Adversarial Competition and Collusion in Algorithmic Markets

Luc Rocher [1,3,†,✉], **Arnaud J. Tournier** [2,3,†], **Yves-Alexandre de Montjoye** [2,3,✉]

[1]Oxford Internet Institute, University of Oxford, UK; [2]Department of Computing, Imperial College London, UK; [3]Data Science Institute, Imperial College London, UK

✉ **For correspondence:**
luc.rocher@oii.ox.ac.uk
demontjoye@imperial.ac.uk

[†]These authors contributed equally to this work.

Abstract

Algorithms are now playing a central role in digital marketplaces, setting prices and automatically responding in real time to competitors' behavior. The deployment of automated pricing algorithms is scrutinized by economists and regulatory agencies, concerned about its impact on prices and competition. Existing research has so far been limited to cases where all firms use the same algorithm, suggesting that anti-competitive behaviour might spontaneously arise in that setting. Here, we introduce and study a general anti-competitive mechanism, adversarial collusion, where one firm manipulates other sellers that use their own pricing algorithm. We propose a network-based framework to model the strategies of pricing algorithms on iterated 2 and 3-firm markets. In this framework, an attacker learns to endogenize competitors' algorithms and then derive a strategy to artificially increase its profit at the expense of competitors. Facing a drastic loss of profits, competitors will eventually intervene and revise or turn off their pricing algorithm. To disincentivize this intervention, we show that the attacker can instead unilaterally increase both its profits and the profits of competitors. This leads to a collusive outcome with symmetric and supra-competitive profits, sustainable in the long run. Together, our findings highlight the need for policymakers and regulatory agencies to consider adversarial manipulations of algorithmic pricing, which might currently fall outside of the scope of current competition laws.

Introduction

Online commerce, which reached 35% of all retail sales in the UK in 2020 [1], is increasingly dominated by digital marketplaces [2]. Facilitated by digital technologies and a growing international logistic network, retailers now compete globally on these platforms. Platform sellers use sophisticated algorithms to set prices, responding in real time to competitors' prices, stock availability, and variable consumer demand [3, 4]. According to a recent study, more than one third of all sellers on Amazon Marketplace already use pricing algorithms [5].

The reliance on algorithms that automatically set prices on marketplaces introduces new risks for buyers and sellers. An important risk is enabling anti-competitive collusion [3, 6], a type of non-cooperative game where participating firms decide to coordinate their prices at the expense of consumers. In practice, such situations are only stable as long as each participant finds it rational to abide by the collusive strategy, similarly to iterated prisoner's dilemmas [7]. However, recent works on machine learning (ML) agents in iterated prisoner's dilemmas suggest that algorithms could learn to collaborate in such non-cooperative games [8].

An emerging literature has scrutinized competition in many algorithmic markets—from airline tickets to gasoline stations [9]—where firms use the same pricing algorithm, e.g., purchased from a third party [10]. In these 'hub-and-spoke' settings, researchers are concerned that modern algorithms could learn over time to tacitly coordinate their prices [3, 6]. Tesauro et al. showed that pricing algorithms learn to increase profitability and damp out or eliminate cyclic price "wars" [11]. Klein further showed that pricing algorithms can converge to strategies that sustain supra-competitive equilibria, without explicit communication [12]. Finally, Calvano et al. showed that algorithms can tacitly learn to consistently charge supra-competitive prices and might learn punishment mechanisms against players trying to defect [13, 14].

Recent theoretical studies in economics and game theory suggest that algorithms may have the potential to anticipate competitors' prices [15] and unilaterally undermine their profits [16]. For instance, in 2011, the retail price of the book "The Making of a Fly" progressively rose to $23,698,655.93 on Amazon due to two naive algorithms each anticipating that the other algorithm would not update its price [17]. Since then, scholars have argued that machine learning pricing algorithms may eventually learn to communicate with competitors [18], which could help coordinate their strategies [19].

In computer science, adversarial techniques have been developed and deployed against a wide range of machine learning algorithms. For instance, keywords have been artificially added at the end of emails to avoid being flagged as spam [20], stickers have been put on road signs to lead autonomous cars off road [21], and artificial fingers have been made from gelatin to fool sophisticated fingerprint systems [22]. These examples have opened a new field of research studying the vulnerabilities of algorithms against adversaries trying to exploit them [23–26].

Here, we propose an adversarial model to study competition in digital marketplaces. We develop a network-based approach to learn the strategy of algorithmic competitors, and validate it against three algorithms from the literature in a standard iterated 2-firm market [11–13]. We show how this knowledge can be used by an attacker to find the best stationary response against each of these algorithms. This stationary response allows an attacker to maximize its long term profits while competitors incurs a loss of profits. In practice, facing a drastic loss of profits, competitors will eventually intervene and revise or turn off their pricing algorithms. To disincentivize this revision, we show that the attacker can instead unilaterally increase both its individual profits and the profits of its competitor. To do so, the attacker derives a pricing strategy with the intent to maximize both its profits and the competitors' profits. We call this new unilateral collusion mechanism 'adversarial collusion' and show how it can be extended to a 3-firm market case.

Adversarial collusion raises new regulatory and enforcement questions. In particular, it might, depending on the circumstances, fall outside of the scope of current competi-

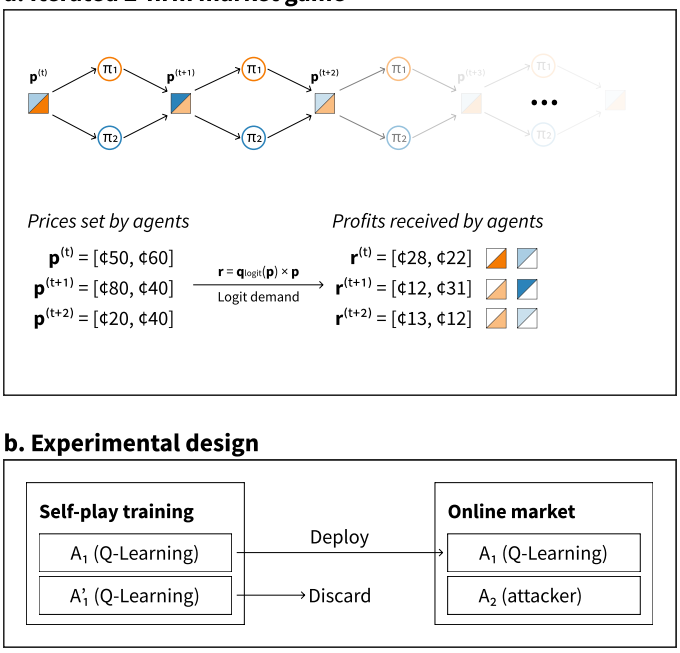


Figure 1. Representation of the 2-firm market game and experimental design. (a) We model an iterated market game, where the competitor and the attacker follow their respective private pricing strategies π_1 and π_2. We show the prices set and profits received in the game using the logit demand model from the CAL setting. **(b)** We follow a standard experimental design where, in a first step, the Q-Learning agent A_1 is trained in self-play against a similar Q-Learning agent A'_1. After discarding A'_1, the agent A_1 is deployed on a market where it faces our attacker A_2.

tion laws in both the European Union and the United States. Our results furthermore emphasize the need for regulatory agencies to provide guidance on pricing algorithms, including on the objective function, on the non-stationarity of strategies, and on how to assess the robustness of pricing algorithms to adversarial collusion.

Results

We study a standard setup consisting of a horizontal iterated market game with n distinct firms, all selling the same good online [27]. We denote by $\mathrm{p} = (p_1, \dots, p_n)$ the vector of prices set by each of the agents, and $q_i(\mathrm{p})$ the overall consumer demand of the market for A_i's good in response to the prices p, for which agents have no prior knowledge. After each iteration of the market, agents independently update their respective price depending on the profits they previously obtained. Formally, at each step t, agent A_i observes the prices $(p_1^{(t)}, \dots, p_n^{(t)})$ on the market, sells $q_i(\mathrm{p}^{(t)})$ goods produced at a marginal cost c_i, receives a profit $r_i^{(t)} = q_i(\mathrm{p}^{(t)})\,(p_i^{(t)} - c_i)$, and decides to set its new price $p_i^{(t+1)} = \pi_i(\mathrm{p}^{(t)})$ by following its private pricing strategy π_i.

To simulate the purchasing behaviors of real-world markets, scholars have relied on synthetic consumer demand models. These demand models are always symmetric and deterministic, solely driven by prices. Table S1 summarises the major demand functions q used in the literature, each modelling different price sensitivities for consumers, including the classical model of Bertrand competition [11–13]. Following standard practices in

the literature [11–13], we set marginal costs $c_i = 0$ for all agents and discretize prices within $[0, 1]$: for each agent A_i, $p_i \in P_M = \{\frac{k}{M} \mid 0 \leq k \leq M\}$ for a positive granularity factor M (set to $M = 25$ here). Thus, a price $p_i = 0$ corresponds to a situation where A_i is pricing its good at the marginal cost $c_i = 0$, making zero profits regardless of the number of customers.

Adversarial pricing strategies

The increasing use of sophisticated algorithms to set prices on online markets creates incentives for competitors to decode strategies, detect weaknesses, and exploit them [15, 18]. Inspired by recent advances in adversarial machine learning, we here propose a new unilateral adversarial approach to learn and exploit the algorithmic strategies of competitors. Below, we initially focus on an iterated 2-firm market game with two pricing algorithms (Fig. 1a). The competitor A_1 uses a stationary private pricing algorithm trained to maximize its own long-term profits, which the attacker A_2 has no prior knowledge of (Fig. 1b). During a preliminary exploration phase (Fig. 2b), the attacker A_2 progressively learns, by selling goods on the market, the private pricing strategy π_1 of its competitor A_1. Using this knowledge, A_2 then maximizes its own profits by anticipating the reactions of its competitor and finding the best response.

Formally, in the exploration phase, the attacker A_2 learns a graph representation $G = (V, E)$ of the competitor's pricing strategy π_1 on the market (Fig. 2a). This characteristic graph G encodes the response of A_1 to price changes by A_2. It comprises all prices as vertices and all admissible transitions as edges (Figs. S1-3):

$$\begin{cases} V = \{(p_1, p_2) \mid p_1 \in P_M, p_2 \in P_M\} \\ E = \{(p_1, p_2) \to (\hat{\pi}_1(p_1, p_2), p_2') \mid (p_1, p_2) \in V, p_2' \in P_M\} \end{cases}$$

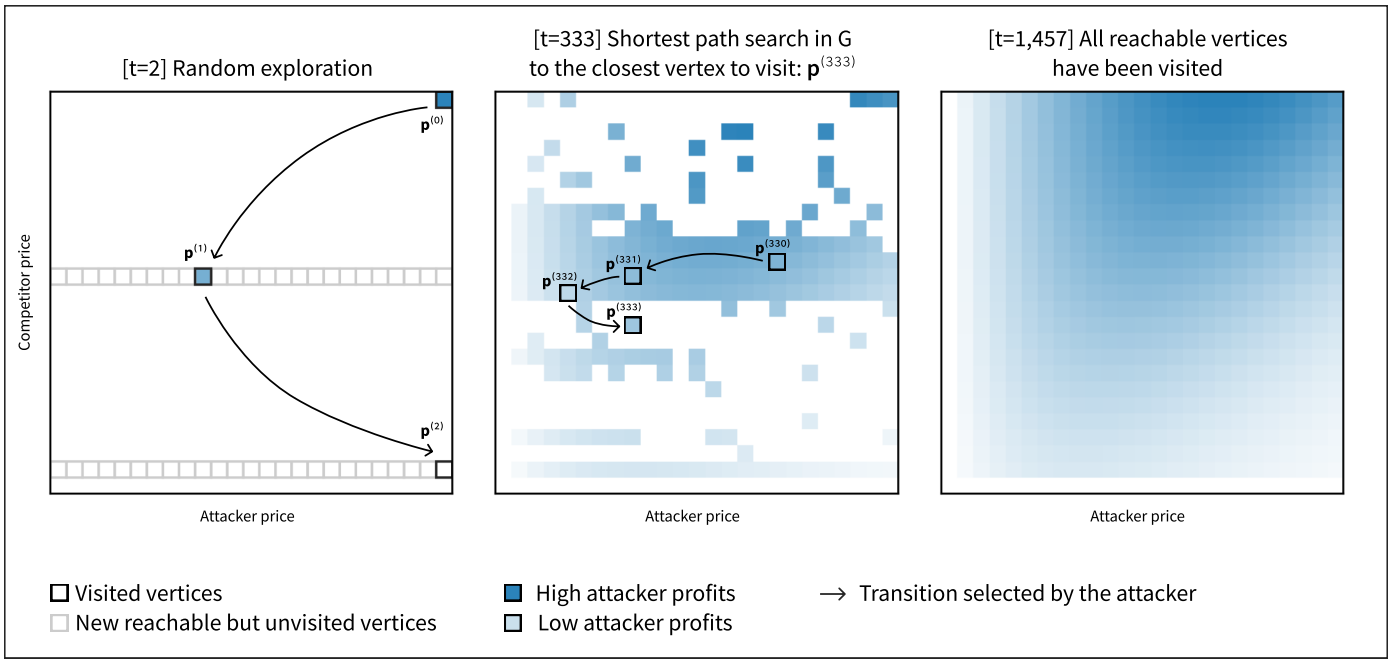
with $\hat{\pi}_1$ an approximation of the pricing strategy of A_1. The strategy $\hat{\pi}_1$ can be learned over time from any initial configuration of the market, by following our exploration procedure until all accessible vertices have been explored (see SI Methods).

In the exploitation phase, the attacker A_2 searches for price sequences in the characteristic graph G leading to high profits in the long run. From G, the attacker computes the policy graph (Fig. 2b) containing the optimal price sequence C to maximize, from any initial position, a given objective function f over time (see SI Methods).

To understand the impact of adversarial pricing on markets, we record the increase in profits obtained by both A_1 and A_2 post learning. Formally, starting from any initial market configuration (p_1, p_2), the competitor A_1 follows its private learned strategy π_1. In line with the algorithmic competition literature, we consider as competitor (A_1) three stationary Q-Learning algorithms: TES introduced by Tesauro and Kephart [11], KLN by Klein [12], and CAL by Calvano et al. [13]) trained according to their respective procedures and environments. These algorithms learn to set the next price $p_i^{(t+1)}$ given any configuration of the market $\mathbf{p}^{(t)}$. Their objective is to maximise not only their immediate profit $r_i^{(t+1)}$, but also their own future profits. Their pricing strategy $\pi_i^{(t)}$ is refined over time until convergence to π_i. These three algorithms slightly differ in their meta-parameters (e.g. choice of the reward function and exploration strategy, see SI Methods) and the overall computational cost of their training until convergence.

Fig. 3 illustrates how our method enables the attacker A_2 to (a) learn the pricing strategy π_1 of its competitor A_1, and (b) derive a strategy π_2 that exploits this information to

a. Exploration phase



b. Learned policy graph

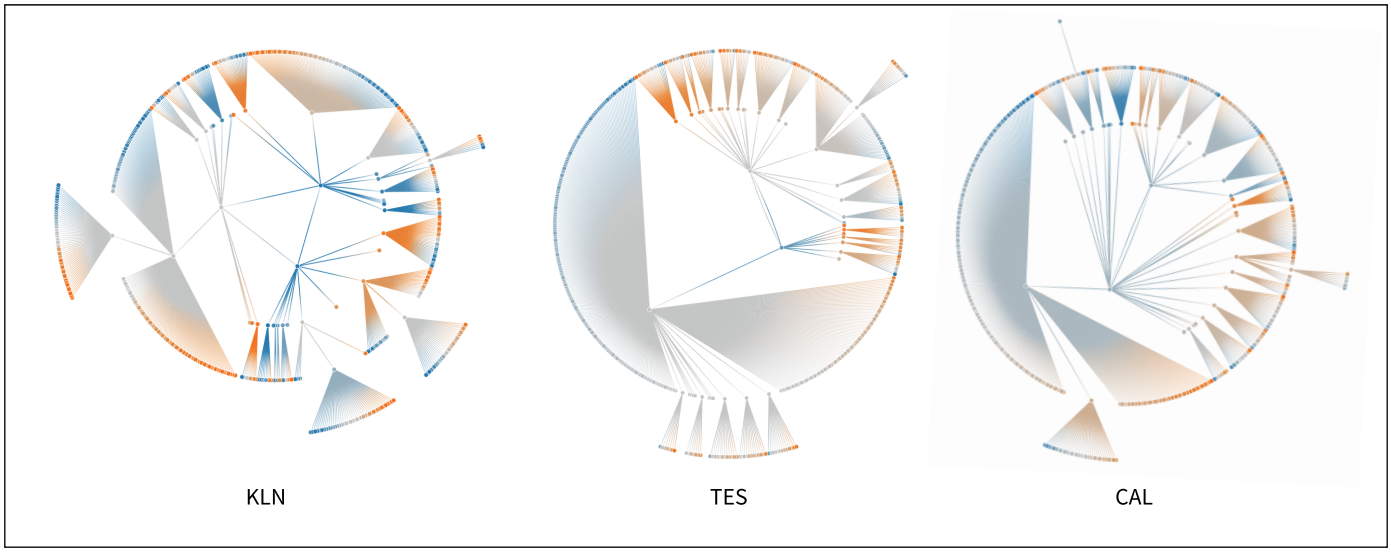


Figure 2. Representation of the exploration and adversarial pricing strategies. (a) During the exploration phase, the attacker forms the characteristic graph *G* composed of vertices (market configurations) and edges (market transitions). It explores vertices until all accessible vertices have been visited. Two exploration methods are combined: random exploration to visit new vertices (left), shortest path search in *G* to reach the closest unvisited vertex (center). The profits obtained by the attacker in each visited state are represented as a shade of blue (maximal in the top-right quadrant, when attacker prices are high and competitor prices higher), here in the CAL setting with $M = 25$. **(b)** Using *G*, the attacker forms a policy graph to maximize its own profits. The policy graph contains only 1 outgoing edge from each vertex and encodes the optimal sequence of prices to set to maximize its profits. We visualize the policy graph as a directed tree leading to a price sequence *C* (center of each graph), along which the attacker iterates.

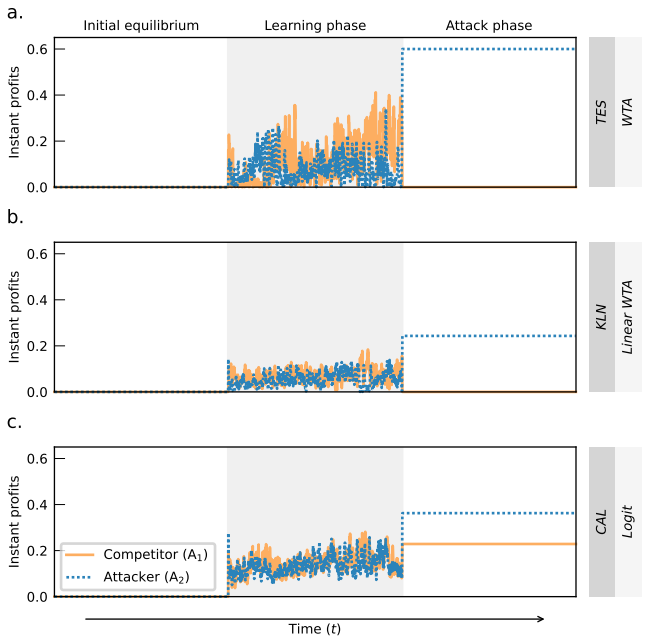


Figure 3. Adversarial pricing enables one firm to increase its profits after learning over time how its competitor reacts to price changes. We compare the profits obtained over time by each competitor (a. TES, b. KLN, c. CAL) in an iterated 2-firm market game. From an initial competitive Nash equilibrium, the attack is carried out in two phases. In the exploration phase, the attacker explores all market price configurations, leading to a wide range of observed profits. This phase ends when the attacker detects that no new market configuration can be explored. Finally, the attacker nudges its competitor towards the best price sequence C to maximize its own profits in the exploitation phase.

maximize its own profits r_2. During the exploration phase, A_2 discovers the best price sequence C. Then, iterating upon C, A_2 obtains high profits r_2 compared to r_1 for A_1. In the first example (Fig. 3a), TES obtains null profits ($r_1 = 0$) while A_2 obtains high profits ($r_2 = 0.600$). In this setting, A_2 sets prices just under A_1, with customers buying from the lower-price firm. In the second example, Fig. 3b shows that KLN obtains null profits ($r_1 = 0$) while A_2 obtains high profits ($r_2 = 0.243$). Compared to the first duopoly, profits are lower since customers purchase less high-priced goods in this setting with linear demand. Finally, in a market where customers are less sensitive to prices (Fig. 3c), A_2 obtains high profits ($r_2 = 0.363$) while CAL still obtains positive profits ($r_1 = 0.229$). The duration of the exploration phase, happening independently from the market iterations, is solely determined by the reactivity of A_1 to price changes (see Discussion).

Adversarial collusion

So far, we have considered a stationary competitor A_1. However, strong competitive strategies, such as the adversarial one presented above, will typically lead to a sudden decrease in profits for other competitors. In response, under-performing firms are likely to intervene, e.g. by revising or even turning off their pricing algorithm [15, 18], limiting the attacker's ability to obtain high profits over time. We test this scenario in Supplementary Note 1, showing that A_1 can restore high individual profits by retraining its

Experimental conditions		Average profits of competitor A_1 (r_1) and attacker A_2(r_2)			
		Adversarial Competition		Adversarial Collusion	
Competitor A_1	Demand function	r_1	r_2	r_1	r_2
TES	Strict WTA	0.000	0.600	0.300	0.300
KLN	Linear WTA	0.000	0.243	0.091	0.091
CAL	Logit	0.229	0.363	0.255	0.255

Table 1. Supra-competitive profits achieved using adversarial collusion for the three studied settings. We report the profits r_1 and r_2 obtained in a 2-firm setting against each of the three agents from the literature. The Adversarial Competition method **(a)** maximizes the attacker profits r_2 while the Adversarial Collusion **(b)** maximizes the profits of all agents. The reported profits are averaged over all 25 × 25 initial configurations of the market, and measured after both the competitor and the attacker have converged.

Q-Learning algorithm. However, Figs. S4-6 show that, in an arms race scenario where A_1 and A_2 iteratively retrain their respective algorithm and attack, profits slowly decrease overall. This suggests that the high individual profits initially obtained by the attacker are not necessarily sustainable if the competitor retrains its algorithm.

An attacker might thus decide to increase profits for both itself and its competitor. Formally, we propose an unilateral adversarial strategy to achieve high sustainable profits in a 2-firm market. To disincentivize A_1 from changing its pricing strategy π_1 to an unknown strategy π_1', A_2 now decides to increase the profits r_1 and r_2 equally. We show empirically how A_2 can estimate the profits r_1 of A_1 (see SI Methods) and then maximize an arithmetic mean objective $f(\mathrm{r}) = (r_1 + r_2)/2$ using our adversarial approach, achieving high profits for both A_1 and A_2.

Table 1 shows that adversarial collusion yields supra-competitive and symmetric profits against the three algorithms TES, KLN, CAL after exploration, consistently higher than the initial equilibrium. The variations observed across competitors are the results of multiple factors, such as the demand function of the market (see Discussion).

Optimal Exploration Strategy

During the exploration phase, the attacker only learns a partial representation of the competitor strategy, limited to visited market configurations. To evaluate the robustness of the exploitation strategy to unexplored configurations, we compare our results against profits obtained with perfect knowledge of the competitor strategy π_1. In our experiments, we show that, without initial knowledge, our attacker consistently achieves profits equal to a perfect-knowledge attacker's profits and from any initial configuration of the market (p_1, p_2). Our attacker explores all market configurations that can be reached, using a breadth-first search algorithm. At the end of the exploration phase, the resulting characteristic graph G forms a connected component of the perfect-knowledge graph G^*, containing a fraction τ_G of explored vertices compared to G^*. In practice, analyzing the size of the connected component G, we find that the fraction τ_G of reachable vertices converges to $\tau_G = 1$ in all three scenarios.

To limit the cost of learning G, the attacker could decide to end the exploration early. Fig. 4 shows that the best achievable price sequence is often found early during the exploration phase, opening the door to the development of early stopping criteria. For in-

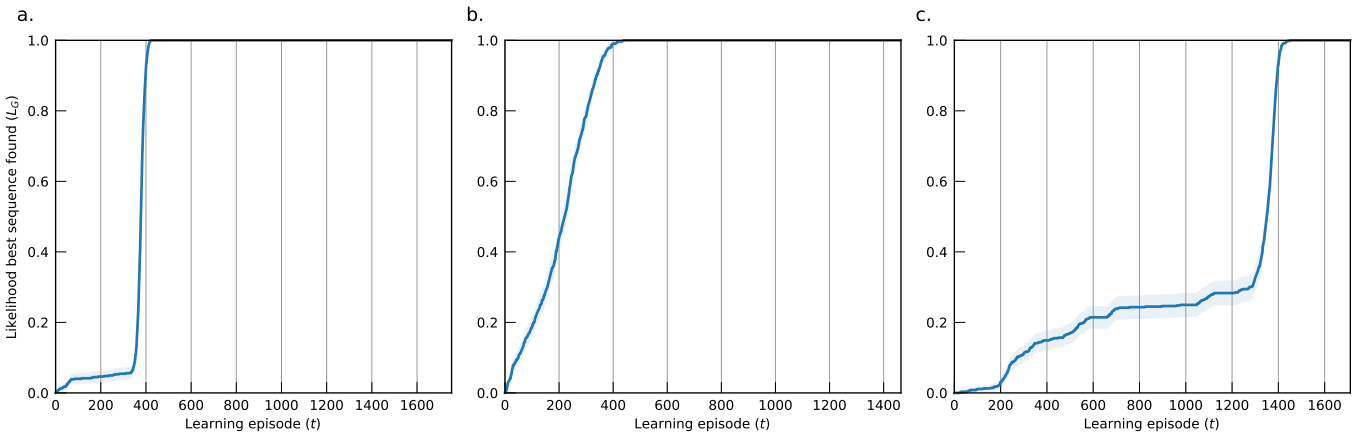


Figure 4. The exploration phase efficiently discovers the optimal price sequence to maximize the profits of both the attacker and the competitor. We report the likelihood L_G of having found the optimal sequence, averaged over all 25 × 25 initial market configuration, as well as its 95% confidence interval. Each panel displays the likelihood L_G for each of the three studied scenarios (a. TES, b. KLN, c. CAL), increasing from L_G = 0 initially to L_G = 100% across all scenarios and for all 25 × 25 initial market conditions.

stance, in our experiments, only 420 steps against TES, 438 steps against KLN, and 1451 steps against CAL are sufficient for the attacker to find C^* with L_G = 1 probability from any initial configuration. However, there can be multiple price sequences yielding high profits for both the competitor and the attacker. Fig. 5 shows that high supra-competitive profits, although not as high as with C^*, can be obtained in less steps.

Extension to 3-firm markets

When more than two firms are involved in a market, economics and machine learning scholars have raised doubt about the likelihood of algorithmic collusion and the sustainability of potential collusive outcomes [28].

To test the potential for collusive outcomes with more than two firms, we extend our adversarial mechanism to a 3-firm market. We follow the same experimental setup as above, with two competitors A_1 and A_2 and an attacker A_3. The competitors A_1 and A_2 use a stationary private pricing algorithm trained to maximize their own respective long-term profits in a 3-firm market. We set the characteristic graph G to encode the response of A_1 and A_2 to price changes by A_3 (see Supplementary Note 2). With M = 25, G has now $|V| = M^3$ vertices and $|E| = |V| \times M = M^4$ edges.

In a single exploration stage, the attacker learns the pricing strategies of the two competitors, encoded in the characteristic graph G. Table S3 shows that the attacker can successfully reach both high adversarial competition profits and high adversarial collusion profits (maximising the profits of all three firms in the market). In a general setting with n distinct firms, a similar approach leads to a graph containing M^n vertices and M^{n+1} edges. Large values of n and M can make the exploration prohibitively long and the optimization algorithm (running in $O(M^{2n+1})$, see SI Methods) computationally expensive. In such cases, beyond the scope of this article, different approaches relying on modern deep reinforcement learning [29, 30] could overcome the computational costs of our method.

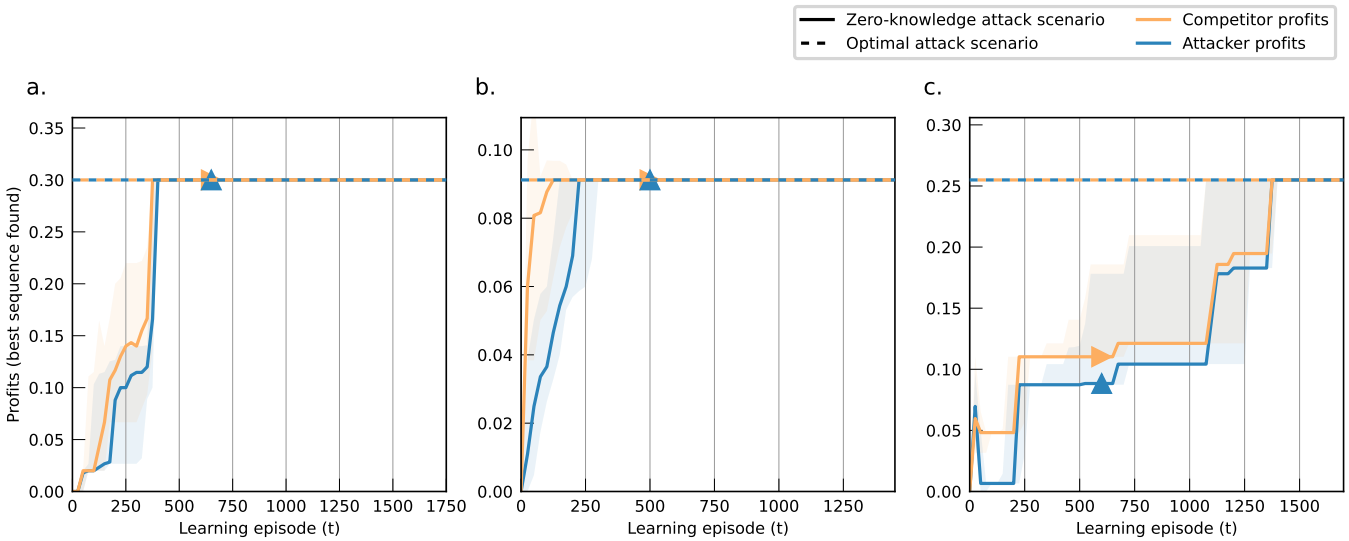


Figure 5. Profits obtained with the best price sequence currently found at time *t*, showing that the attacker can find a good price sequence even with limited exploration. We report the profits r_1 (competitor A_1) and r_2 (attacker A_2) for the best sequence found within the explored vertices (market price configuration) during the exploration phase for each of the three studied scenarios (a. TES, b. KLN, c. CAL). We display the median profits along with the 25% and 75% quartiles, averaged over each of the 25 × 25 initial market configuration. For instance, stopping after only half of the complete exploration phase duration would yield the optimal profits against TES and KLN„ and 39% of the optimal profits against CAL (median profits, filled triangles). The curves are not necessarily monotonous and profits symmetric: the adversary estimates the competitor's profits by assuming symmetry of the demand function and marginal costs (see Methods), therefore visits configurations where the competitor's profits cannot be estimated yet. Once all vertices have been explored, all profits can be estimated, and the best sequence corresponds to symmetric profits.

Conclusion

The competition literature has long theorized how sellers can learn the pricing strategies of their competitors [15, 31, 32]. Our results show that this is both technologically feasible and practical against stationary pricing algorithms. Our attacker is able to efficiently learn its competitor's strategy, opening up a new collusion mechanism on digital marketplaces: adversarial collusion. Together, our findings emphasize the need for policymakers and regulatory agencies to consider how adversarial mechanisms could subtly undermine the competitiveness of online markets and harm consumers [6].

Broader impact statement

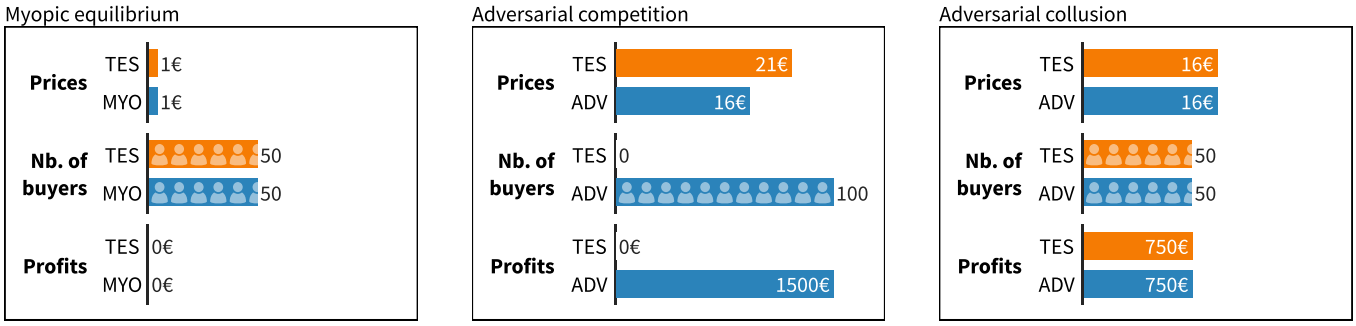
In our research, we use a standard economic setting to study pricing algorithms in silico, on which we have identified new risks for consumers. After careful consideration, we believe publishing our findings benefits society and largely outweighs the risks to consumers and businesses. Adversarial attacks are already known in the computer science literature and could already be exploited in practice. Our research allows policy discussion to progress and regulators to make informed decisions by quantifying the impact of adversarial collusion (see Fig. 6). We hope our findings encourage the implementation of security measures in online markets and further empirical research measuring adversarial harms in real-world online markets.

Our results raise important regulatory and enforcement questions. While a complete analysis is beyond the scope of this work, adversarial collusion might, depending on the circumstances, fall outside the scope of current competition laws [6]. On the one hand, in Europe, judgments from the Court of Justice of the European Union (CJEU) [33] seem to give firms significant leeway to unilaterally set prices if there is no coordination. In the US, Department of Justice officials suggested that algorithmic collusion is not illegal without an agreement among participants [34]. On the other hand, adversarial collusion might be considered a concerted practice, e.g., under Art. 101 of the Treaty on the Functioning of the European Union (TFEU), if it was proven that the competitor had become aware that its algorithm was being manipulated. Similarly, if the attacker was in a dominant position, having their algorithm explicitly optimize for shared profits might be considered as disincentivizing competitors from becoming more efficient, and therefore abusive. From an enforcement perspective and given the error-cost, agencies might thus want to consider explicitly restricting maximization of shared profits, e.g. by making it a rebuttable presumption. They might also consider the use of weak algorithms to be equivalent to price signaling—a controversial practice—and give further guidance on how to assess the robustness of pricing algorithms.

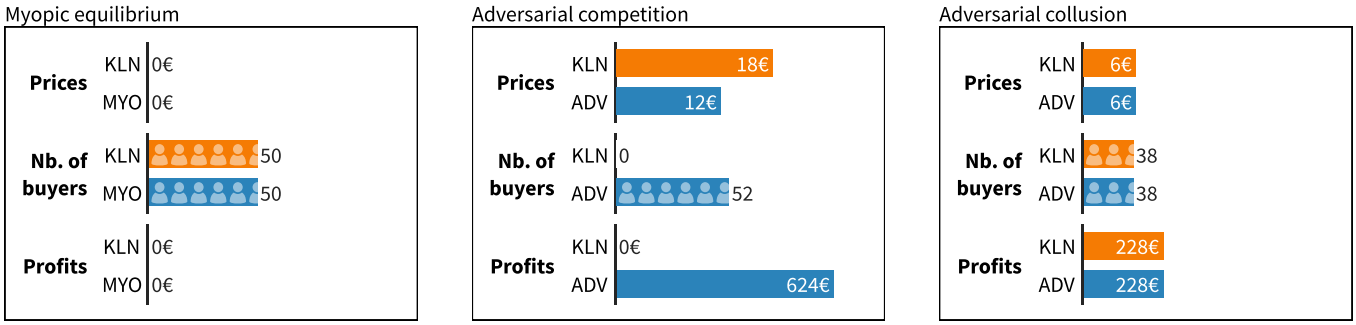
Discussion

Beyond the machine learning pricing algorithms we considered, simpler algorithms used in practice are also likely to be vulnerable to targeted attacks. For instance, the well-know tit-for-tat algorithm [35]—deployed by the competitor—could be trivially led by an attacker to collude and offer stable supra-competitive prices. Similarly, the algorithm responsible for pricing the book "The Making of a Fly" up to $23,698,655.93 on Amazon was simply adjusting its price to 1.270589 times the price offered by another seller [17]. This

a. Winner-take-all demand (e.g., prescription drug market, marginal cost 1€)



b. Linear demand (e.g., access to scientific articles, marginal cost 0€)



c. Logit demand (e.g., hotel room booking, marginal cost 50€)

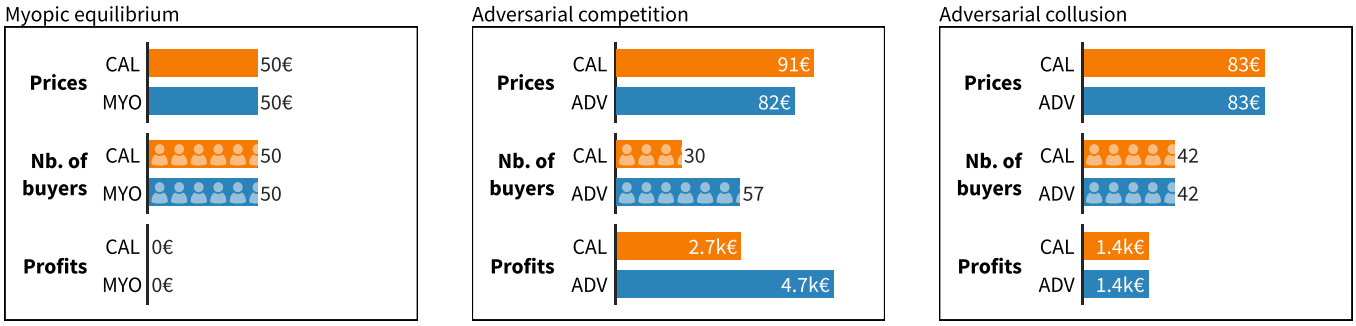


Figure 6. Harms caused to consumers by adversarial competition and collusion. We report the effect of adversarial attacks on example markets, for the three considered scenarios, compared to a baseline setting where TES, KLN, CAL face a myopic firm pricing at the marginal cost. As an example, we renormalize prices from [0€, 1€] (without marginal cost) to realistic intervals (with marginal cost), and report the effects of each attack on prices, number of buyers, and profits. **(a)** Winner-take-all demand, marginal cost of 1€, and prices between 1€ and 26€, simulating for instance an elastic prescription market where buyers will always buy and from the cheapest seller. **(b)** Linear demand, marginal cost of 0€, prices between 0€ and 25€, simulating for instance online access to scientific articles (inelastic market with consumers buying access from the cheapest provider, with more consumers buying the lower the price is). **(c)** Logit demand, marginal cost of 50€, prices between 50€ and 100€, simulating for instance hotel room booking in a city (elastic market with consumer more likely to book from hotels with less expensive rooms but taking into accounts other factors).

algorithm too is likely to be vulnerable to targeted adversarial attacks, e.g. by estimating the maximum price consumers would be willing to pay for the book and nudging the algorithm to reach and stay at that price, instead of reaching unrealistically high prices. These real-world examples reinforce the importance to consider the impact of adversarial mechanisms on competition.

To learn the competitor's strategy π_1, our algorithm varies its price and observes how A_1 reacts over time. If goods are sold during that time, the attacker A_2 might incur a profit r_2 lower than r_1. In our experiments, learning the characteristic graph G is fast, taking at most 1753 steps against TES, 1464 steps against KLN, and 1562 steps against CAL. Empirical evidence suggests that pricing algorithms in online platforms are often highly reactive to price changes. Uber, for instance, reportedly updates prices every 3 to 5 minutes [36], while algorithms used by Amazon vendors update their prices hundreds to thousands of times per day [5]. This, along with early-stopping mechanisms and strategies to learn π_1 when the demand is low (e.g., at night), would strongly limit the cost of the exploration phase.

In our setting, the attacker starts its exploration after the competitor has finished learning its pricing policy π_1. In practice, the attacker can determine when π_1 becomes deterministic by monitoring the competitor's actions. However, if the attack starts while the competitor is still learning, other classes of adversarial machine learning techniques could be employed. In such cases, the attacker could force the competitor to learn to always set high prices by using action or online data poisoning, tweaking prices while the competitor learns to maximise its profits, to undermine learning objectives [37, 38].

We considered the standard economic setting to study pricing algorithms, an iterated 2-firm market where reinforcement learning (RL) algorithms are trained independently with single-agent methods [11–13, 39]. These single-agent methods were developed for an agent evolving alone in a fixed and stationary environment [40], meaning this environment is not changing over time in response to the agent learning. In the standard training phase studied above, each Q-Learning learning agent interacts with an environment that mutates when the other Q-Learning agent updates its policy. Modern RL methods in multiplayer iterated settings are typically multi-agent [29, 30, 41]. While our experiments provides initial insights on adversarial collusion, recent studies have also shown that even state of the art multi-agent RL agents can be exploited by adversarial policies [42]. More work would be needed to assess the effectiveness of our attack against algorithms trained using multi-agent RL techniques.

Adversarial collusion, measured in our experiments against a range of competitors and in multiple market settings, consistently obtains supra-competitive profits for all agents on the market. These profits vary across experiments (see Table 1). These variations could stem from many factors, such as the training procedures or meta-parameters used by competitors' algorithms (see SI Methods). However, we hypothesize that the consumer demand, shown in theory and practice to strongly impact the learned strategies of pricing algorithms [13, 39], is mainly responsible for the observed variations of profits in our experiments.

To study the risk of adversarial collusion, in line with the literature, we assumed that consumers are uniformly modelled by one demand model and that firms assign the same price to each consumer. In practice, however, firms regularly use data-driven 'personalised pricing' strategies, e.g., matching insurance premiums to consumer risk profiles

or adapting shipping price based on the user location [4, 43]. The evidence of adversarial collusion could be more difficult for external auditors to detect when firms use price discrimination—adapting prices to consumer profiles.

In our experiments, the competitor can quickly notice that their profits have changed once the attacker executes its attack. This sends a clear signal to the competitor who could then revise its strategy (see Supplementary Note 1). In adversarial machine learning, attacks are often designed to be hard to detect or even unnoticeable [23–26]. In Supplementary Note 3, we show that, in our setting, the attacker can aim to maximise its own profits while keeping the competitor's profits stable. We designed a constrained objective obtaining high profits r_2 of 0.360, 0.600, 0.186 while the respective competitors CAL, TES, KLN receive profits within $\frac{1}{M} = 0.04$ currency units of their profits before the attack. Although the competitor's profits remain stable, the competitor can still detect that it receives substantially lower profits than the attacker. We believe that adversarial collusion, i.e. maximizing *symmetric* joint profits of both agents, is more stable in the long term than attempting to keep the competitor's profits stable.

Code Availability

The source code to reproduce the results of this article is available on the Open Science Framework (OSF) repository at http://dx.doi.org/10.17605/osf.io/2yuvm [44]. Data files to reproduce figures are also available in the OSF repository.

Author Contributions Statement

L.R. and A.J.T. contributed to conceptualization, methodology development, software development, experimental validation, and writing. Y.-A.d.M. contributed to conceptualization, methodology development, and writing.

Acknowledgments

We thank all members of the Computational Privacy Group for discussions and suggestions. We also thank J. Cremer and H. Piffaut for comments on earlier versions of the manuscript.

Competing interests Statement

L.R. acknowledges support from EPSRC (EP/W016419/1). A.J.T. and Y.-A.d.M. acknowledge that they received no funding in support for this research. The authors declare that they have no competing interests.

References

1. Dalgleish, R. *Retail sales, Great Britain* Accessed: 2023-01-01. Office for National Statistics. https://www.ons.gov.uk/businessindustryandtrade/retailindustry/bulletins/retailsales/december2020.
2. Buck, R. *et al. Perspectives on Retail and Consumer Goods: Issue 8* McKinsey & Company. https://www.mckinsey.com/~/media/mckinsey/industries/retail/our%20insights/perspectives%20on%20retail%20and%20consumer%20goods%20number%208/perspectives-on-retail-and-consumer-goods_issue-8.pdf.
3. Competition and Markets Authority. *Pricing algorithms: Economic working paper on the use of algorithms to facilitate collusion and personalised pricing* https://gov.uk/government/publications/pricing-algorithms-research-collusion-and-personalised-pricing. Oct. 2018.
4. OECD. *Algorithms and Collusion: Competition Policy in the Digital Age* http://www.oecd.org/daf/competition/Algorithms-and-colllusion-competition-policy-in-the-digital-age.pdf. 2017.
5. Chen, L., Mislove, A. & Wilson, C. *An Empirical Analysis of Algorithmic Pricing on Amazon Marketplace* in *Proc Int. Conf. World Wide Web* (ACM, Montréal, Québec, Canada, Apr. 2016), 1339–1349.
6. Thomas, S. Harmful Signals: Cartel Prohibition and Oligopoly Theory in the Age of Machine Learning. *J. Compet. Law Econ* **15,** 159–203 (Oct. 2019).
7. Axelrod, R. in *The dynamics of norms* (eds Lowenstein, C. & Lowenstein, M.) 1–16 (Cambridge University Press, 1987).
8. Lerer, A. & Peysakhovich, A. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. arXiv: 1707.01068 [cs.AI] (July 2017).
9. Assad, S., Clark, R., Ershov, D. & Xu, L. Algorithmic Pricing and Competition: Empirical Evidence from the German Retail Gasoline Market (2020).
10. Petit, N. Antitrust and artificial intelligence: a research agenda. *J. Eur. Compet. Law Pract.* **8,** 361–362 (2017).
11. Tesauro, G. & Kephart, J. O. Pricing in agent economies using multi-agent Q-learning. *Auton. Agent. Multi. Agent. Syst.* **5,** 289–304 (2002).
12. Klein, T. Autonomous Algorithmic Collusion: Q-Learning Under Sequential Pricing. *RAND J. Econ.* **52,** 538–558 (2021).
13. Calvano, E., Calzolari, G., Denicolò, V. & Pastorello, S. Artificial Intelligence, Algorithmic Pricing, and Collusion. *Am. Econ. Rev.* **110,** 3267–97 (Oct. 2020).
14. Calvano, E., Calzolari, G., Denicolò, V., Harrington, J. E. & Pastorello, S. Protecting consumers from collusive prices due to AI. *Science* **370,** 1040–1042. eprint: https://science.sciencemag.org/content/370/6520/1040.full.pdf (2020).
15. Salcedo, B. *Pricing algorithms and tacit collusion* http://brunosalcedo.com/docs/collusion.pdf. 2015.
16. Press, W. H. & Dyson, F. J. Iterated Prisoner's Dilemma contains strategies that dominate any evolutionary opponent. *Proc. Natl. Acad. Sci. U. S. A.* **109,** 10409–10413 (June 2012).

17. Eisen, M. *Amazon's $23,698,655.93 book about flies* https://www.michaeleisen.org/blog/?p=358. 2011.
18. Kühn, K.-U. & Tadelis, S. *Algorithmic Collusion* https://cresse.info/wp-content/uploads/2020/02/2017_sps5_pr2_Algorithmic-Collusion.pdf. 2017.
19. Crandall, J. W. *et al.* Cooperating with machines. *Nat. Commun.* **9,** 1–12 (2018).
20. Dalvi, N., Domingos, P., Sanghai, S. & Verma, D. *Adversarial classification* in *Proc. Int. Conf. Knowl. Discovery Data Mining* (ACM, 2004), 99–108.
21. Eykholt, K. *et al. Robust physical-world attacks on deep learning visual classification* in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.* (IEEE, 2018), 1625–1634.
22. Matsumoto, T., Matsumoto, H., Yamada, K. & Hoshino, S. *Impact of artificial" gummy" fingers on fingerprint systems* in *Optical Security and Counterfeit Deterrence Techniques IV* **4677** (2002), 275–289.
23. Biggio, B. & Roli, F. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition* **84,** 317–331 (2018).
24. Su, J., Vargas, D. V. & Sakurai, K. One pixel attack for fooling deep neural networks. *IEEE Trans. Evol. Comput.* **23,** 828–841 (2019).
25. Carlini, N. & Wagner, D. *Audio adversarial examples: Targeted attacks on speech-to-text* in *2018 IEEE Security and Privacy Workshops (SPW)* (IEEE, 2018), 1–7.
26. Lei, Q. *et al.* Discrete adversarial attacks and submodular optimization with applications to text classification. *Proceedings of Machine Learning and Systems* **1,** 146–165 (2019).
27. Ivaldi, M., Jullien, B., Rey, P., Seabright, P. & Tirole, J. in *The Political Economy of Antitrust* (eds Ghosal, V. & J. Stennek, J.) 217–239 (Elsevier, 2007).
28. Schwalbe, U. Algorithms, Machine Learning, and Collusion. *J. Compet. Law Econ.* **14,** 568–607 (Dec. 2018).
29. Silver, D. *et al.* Mastering the game of Go without human knowledge. *Nature* **550,** 354–359 (2017).
30. Vinyals, O. *et al.* Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* **575,** 350–354 (2019).
31. Milgrom, P. & Roberts, J. Rationalizability, learning, and equilibrium in games with strategic complementarities. *Econometrica* **58,** 1255–1277 (1990).
32. Schlosser, R. & Boissier, M. *Dynamic pricing under competition on online marketplaces: A data-driven approach* in *Proc. Int. Conf. Knowl. Discovery Data Mining* **24** (ACM, 2018), 705–714.
33. Council of European Union. Commission Decision of 19 December 1984 relating to a proceeding under Article 85 of the EEC Treaty (IV/29.725 - Wood pulp). *OJ* **L 85,** 1–52 (Mar. 1985).
34. Guniganti, P. *US DOJ deputy: algorithmic cartel requires agreement* Accessed: 2021-4-16. GCR. https://globalcompetitionreview.com/us-doj-deputy-algorithmic-cartel-requires-agreement.
35. Axelrod, R. & Hamilton, W. D. The Evolution of Cooperation. *Science* **211,** 1390–1396 (1981).

36. Chen, L., Mislove, A. & Wilson, C. *Peeking beneath the hood of Uber* in *Proc. 2015 Internet Measurement Conf.* (ACM, 2015), 495–508.

37. Liu & Lai. Provably Efficient Black-Box Action Poisoning Attacks Against Reinforcement Learning. *Adv. Neural Inf. Process. Syst.* (2021).

38. Zhang, X., Zhu, X. & Lessard, L. *Online Data Poisoning Attacks* in *Proceedings of the 2nd Conference on Learning for Dynamics and Control* (eds Bayen, A. M. *et al.*) **120** (PMLR, 2020), 201–210.

39. Hansen, K. T., Misra, K. & Pai, M. M. Frontiers: Algorithmic Collusion: Supra-competitive Prices via Independent Algorithms. *Marketing Science* **40,** 1–12 (2021).

40. Sutton, R. S. & Barto, A. G. *Reinforcement Learning: An Introduction* (MIT Press, Nov. 2018).

41. Busoniu, L., Babuska, R. & De Schutter, B. A comprehensive survey of multiagent reinforcement learning. *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.* **38,** 156–172 (2008).

42. Wang, T. T. *et al.* Adversarial Policies Beat Professional-Level Go AIs. arXiv: 2211.00241 [cs.LG] (Nov. 2022).

43. Ariel, E. & Maurice, E. S. Virtual Competition: The Promise and Perils of the Algorithm-Driven Economy. *Harvard University Press* (2016).

44. Rocher, L. & Tournier, A. J. *Cabale: a Reproducible Python Environment for Adversarial Collusion Experiments* https://dx.doi.org/10.17605/osf.io/2yuvm.

Supplementary Information for: Adversarial Competition and Collusion in Algorithmic Markets

Contents

Supplementary Materials and Methods **ii**

Framework . . . ii
Algorithmic pricing . . . ii
Q-Learning agents in the literature . . . ii
Network-based approach . . . iii
Measuring average asymptotic profits . . . iv
Comparison to hub-and-spoke . . . iv
Exploration algorithm . . . iv

Supplementary notes **vi**

Supplementary note 1: arms race experiment . . . vi
Supplementary note 2: 3-firm experiment . . . vi
Supplementary note 3: unnoticeable pricing objective . . . vi

Supplementary tables S1 to S4 **viii**

Supplementary figures S1 to S6 **x**

Supplementary Materials and Methods

Framework

In this article, we have considered a symmetrical game where marginal costs are equal between all agents, i.e. for all agents $c_i = c$. Any agent A_i can thus infer the profit $r_j^{(t)}$ obtained by its competitor A_j by observing the demand $q_j(\mathrm{p}^{(t)})$ they received for the price $p_j^{(t)}$ they offered. In practice, prices can often be monitored in real time, while the demand received by agents could, e.g., be estimated by a learning algorithm or deduced a posteriori from accounting documents.

Algorithmic pricing

The algorithmic collusion literature has proposed Q-Learning (QL) models [1] for agents to maximize not only their immediate profit

$$r_i^{(t+1)} = q_i(\mathrm{p}^{(t+1)})(p_i^{(t+1)} - c_i)$$

but the total $G_i^{(t+1)}$ of all future discounted profits:

$$G_i^{(t+1)} = \sum_{k \geq 0} \delta^k r_i^{(t+k+1)}$$

To obtain a pricing strategy maximizing G_i, models are trained in self-play, i.e. against a similar but distinct opponent. In self-play, each agent A_i dynamically improves its own strategy over time until convergence. Given current prices $\mathrm{p}^{(t)}$ on the market, A_i searches the next price $p_i^{(t+1)}$ that maximizes $G_i^{(t+1)}$. The optimal policy function $\pi_i^*(\mathrm{p}^{(t)}) = p_i^{(t+1)}$ is unknown to A_i. Instead, A_i relies on an estimated policy $\pi_i^{(t)}$, refined over time using the obtained profits $\{r_i^{(0)}, r_i^{(1)}, \dots, r_i^{(t)}\}$. Formally, at each time step t, A_i updates an internal tensor (usually called matrix) $Q_i^{(t)}(\mathrm{p}^{(t)}, p_i^{(t+1)})$. This matrix reflects an estimated score for selecting the price $p_i^{(t+1)}$ at step $t+1$, knowing that prices on the market at step t are $\mathrm{p}^{(t)}$. Using this matrix:

$$\pi_i^{(t)}\left(\mathrm{p}^{(t)}\right) = \arg\max_p Q_i^{(t)}\left(\mathrm{p}^{(t)}, p\right)$$

Many methods have been proposed to update the matrix Q_i over time. They all aim to converge to an optimal matrix Q_i^* associated with the optimal pricing policy π_i^*. To sufficiently explore all the cells of the Q-matrix during training, random prices are usually selected with a probability ϵ. This procedure, known as ϵ-greedy iteration, has been shown to improve performances of QL agents when the parameter ϵ is small or decreases towards 0 over time [1].

Q-Learning agents in the literature

The pricing model proposed by Tesauro et al. (TES) [2] is a QL agent. The Q-matrix associated with the decision function is initialized with instantaneous payoff values, and updated iteratively using off-policy iteration [1]:

$$Q_i^{(t+1)}\left(\mathrm{p}^{(t)}, p^{(t+1)}\right) = Q_i^{(t)}\left(\mathrm{p}^{(t)}, p_i^{(t+1)}\right) + \alpha_t\left[r_i^{(t+1)} + \delta \arg\max_p Q_i^{(t)}\left(\mathrm{p}^{(t)}, p\right) - Q_i^{(t)}\left(\mathrm{p}^{(t)}, p_i^{(t+1)}\right)\right]$$

with a discount factor $\delta = 0.9$ and learning rate $\alpha_t = 0.1/(1 + 0.01t)$ decreasing over time. TES is trained for $T = 20,000$ iterations.

Similarly, the pricing model proposed by Klein (KLN) [3] is a QL agent. The Q-matrix is initialized at zero, and updated with sequential Q-learning:

$$Q_i^{(t+1)}\left(\mathrm{p}^{(t)}, p^{(t+1)}\right) = Q_i^{(t)}\left(\mathrm{p}^{(t)}, p_i^{(t+1)}\right) + \alpha\left[r_i^{(t+1)} + \delta\, r_i^{(t+2)} + \delta^2 \arg\max_p Q_i^{(t)}\left(\mathrm{p}^{(t)}, p\right) - Q_i^{(t)}\left(\mathrm{p}^{(t)}, p_i^{(t+1)}\right)\right]$$

with a fixed learning rate $\alpha = 0.30$, a discount factor $\delta = 0.95$, and a decreasing $\epsilon_t = 10^{-6t/T}$. KLN is trained for $T = 10$ million iterations.

Finally, the pricing model proposed by Calvano et al. (CAL) [4] is also a QL agent. The Q-matrix is initialized with discounted payoff values, and updated with off-policy iteration, similarly to TES. A fixed learning rate $\alpha = 0.15$, a discount factor $\delta = 0.95$, and a decreasing $\epsilon_t = \exp(-4\ 10^{-6}\ t)$ are used during training. The training continues until convergence is achieved, or stops when $T = 1$ million iterations have passed.

For all three models TES, KLN, CAL, the training aid agent is discarded once convergence is achieved.

Network-based approach

We develop a network-based method for the attacker A_2 to learn the strategy of its algorithmic competitor A_1 in the iterated 2-firm market game. In line with previous works [2–5], we consider a symmetric demand function between agents, i.e. $q_1(i,j) = q_2(j,i)$ for all $i, j \in P_M$, and assume that the competitor's pricing algorithm has converged and is now deterministic (see previous section).

At each time step t, A_1 and A_2 observe the current prices $\mathrm{p}^{(t)} = (p_1^{(t)}, p_2^{(t)})$ of the market and decide resp. on new prices $p_1^{(t+1)} = \pi_1(\mathrm{p}^{(t)})$ and $p_2^{(t+1)} = \pi_2(\mathrm{p}^{(t)})$. We consider a scenario where A_2 decides to use an adversarial policy π_2 that exploits the private policy π_1.

First, in the exploration phase, the attacker A_2 learns the characteristic graph $G = (V, E)$ of the competitor's pricing strategy π_1 on the market (see algorithm 2 in Section). To do so, A_2 maintains both a list of unexplored states $\mathrm{p} = \{(p_1, p_2), \ldots\}$, and a list of reachable states from explored states. A_2 progressively builds G by constantly updating the direction of exploration towards the closest unexplored state. Formally, the attacker computes the shortest path to each unexplored state using a breadth-first search (BFS) [6] and then selects prices along the path leading to the closest unexplored state. Once that state is reached, the process repeats until all states have been explored. An early stopping criterion could also be used to stop the learning phase before all states have been explored; for instance, after a given amount of elapsed time, after a given number of states have been explored, or when highly profitable price sequences have been discovered (see Discussion).

During this exploration phase, the attacker A_2 also progressively learns the profits received by the agent A_1 by assuming symmetry of the demand function and marginal costs. For each visited market configuration $v = (p_1^{(t)}, p_2^{(t)})$, A_2 receives the profits $r_2(v)$ and thus learns $\hat{r_1}(v') = r_2(v)$ with $v' = (p_2^{(t)}, p_1^{(t)})$. At the end of the exploration phase, the attacker also assumes that $\hat{r_1}(v') = r_2(v) = 0$ for any unexplored state v.

Second, in the exploitation phase, the attacker aims to obtain the best response to the competitor's algorithm in order to maximize a given objective function f in the long run. This means that, from any initial market configuration $(p_1^{(0)}, p_2^{(0)})$, the attacker's policy π_2 should be optimal according to the following criterion:

$$\pi_2 = \arg\max_{\pi} \sum_{t=0}^{+\infty} f_{\pi_1,\pi}(t, p_1^{(0)}, p_2^{(0)})$$

A heuristic to obtain π_2 from any initial market configuration is for A_2 to find a reachable vertex $v \in V$ or cycle of vertices $C = (v_1, v_2, \dots, v_k = v_1)$ maximizing the objective function $f(C)$. Here, we consider the following objective functions corresponding to our Adversarial Competition and Adversarial Collusion approaches:

$$f_{\text{comp}}(C) = \frac{1}{|C|} \sum_{v \in C} r_2(v)$$

$$f_{\text{coll}}(C) = \frac{1}{|C|} \sum_{v \in C} \frac{1}{2}(\hat{r_1}(v) + r_2(v))$$

Searching C that maximizes f is known as a maximum cycle mean problem in graph theory. We use a dynamic programming method developed by Karp [7] to solve it efficiently in $O(|V|\,|E|)$.

Finally, the attacker A_2 searches for the shortest path to C using a breadth-first search (BFS) in $O(|V| + |E|)$, and selects prices along this path until C is reached. Once the cycle C is reached, A_2 selects the next price to stay on C.

Measuring average asymptotic profits

The initial market conditions can affect the measured outcomes at the end of an experiment. To account for these variations, the reported profits are always averaged over all initial market configurations. For $M = 25$, we systematically perform $M \times M$ experiments starting from each initial market configuration (p_1, p_2).

Furthermore, from the initial configuration of the market, an initial burn-in period of 100 iterations is discarded before measuring asymptotic profits. After that initial period, the profits are computed and averaged over 1000 steps.

Comparison to hub-and-spoke

The mechanism we introduced here constitutes a new unilateral collusion mechanism, different from the traditional hub-and-spoke setting. We show in our experiments that supra-competitive prices can be systematically achieved even if A_1 and A_2 do not share the same algorithm. Starting from any initial configuration (p_1, p_2) of the market, we compare the adversarial collusion and hub-and-spoke asymptotic profits. We show that our adversarial collusion profits are similar or higher than in the hub-and-spoke setting (+1.510^4% against TES, +0% against KLN, and 0% against CAL, see Table S2).

Exploration algorithm

The exploration is implemented with two algorithms. The algorithm 1 initializes an empty characteristic graph and a list of prices to explore from each unvisited market configuration. The algorithm 2 progressively complete the vertices and edges of G, by successively visiting configurations using a combination of random exploration and shortest path search (Floyd–Warshall).

Algorithm 1 Initialising the states to explore

procedure Init(M)
    $unexploredPrices \leftarrow \{\}$ empty dictionary
    $G \leftarrow$ EmptyGraph()
    for $i \leftarrow 1$ to M **do**
        for $j \leftarrow 1$ to M **do**
            $unexploredPrices[(i,j)] \leftarrow \{k \mid k \in [1, M]\}$
            AddVertex($(i,j), G$)
        end for
    end for
    Return $G, unexploredPrices$
end procedure

Algorithm 2 Performing the exploration until a stopping criterion is reached

procedure Explore($unexploredPrices, G, s$)
    if StoppingCriterion() **then** default criterion is $unexploredPrices = \emptyset$
        Return G
    else
        if $unexploredPrices[s] \neq \emptyset$ **then**
            $price \leftarrow$ UniformRandom($unexploredPrices[s]$)
        else
            $stack \leftarrow$ ShortestPath($unexploredPrices, G, s$) Breadth First Search, only the last transition is unexplored
            while $stack \neq \emptyset$ **do**
                $price \leftarrow$ Pop($stack$)
                $s \leftarrow$ ExecuteMarket($\pi_1(s), price$)
            end while
        end if
        Remove($unexploredPrices[s], price$)
        $s' \leftarrow$ ExecuteMarket($\pi_1(s[0]), price$)
        AddEdge(s, s', G)
        Explore($unexploredPrices, G, s'$)
    end if
end procedure

Supplementary notes

Supplementary note 1: arms race experiment

We simulate an arms race situation where the competitor A_1 can now re-train its pricing algorithm after being attacked by A_2. Once the agent has fully retrained its algorithm against the attacker, the attacker re-initiates the attack from the start. Over time, both the agent and the attacker battle to find a dominant pricing strategy.

Figures S4 to S6 report the profits obtained by A_1 and A_2 after each retraining, over 10 successive episodes. In all three scenarios, the profits obtained by both agents decrease over time. When facing TES, the attacker obtains high profits during three episodes until both agents start receiving null profits. When facing KLN, the attacker obtains high profits each times it explores and run its attack while KLN restores lower profits by retraining its algorithm; over time, both agents receive lower and lower profits. Finally, when facing CAL, the attacker obtains high profits each times it explores and runs its attack while KLN also restores its profits by retraining its algorithm; both agents receiving lower profits after the first episode.

Supplementary note 2: 3-firm experiment

In a 3-firm setting, the characteristic graph G encodes the response of A_1 and A_2 to price changes by A_3. It comprises all prices as vertices and all admissible transitions as edges:

$$\begin{cases} V = \{(p_1, p_2, p_3) \mid p_1 \in P_M, p_2 \in P_M, p_3 \in P_M\} \\ E = \{(p_1, p_2, p_3) \to (\hat{\pi}_1(p_1, p_2, p_3), \hat{\pi}_2(p_1, p_2, p_3), p'_3) \mid (p_1, p_2, p_3) \in V, p'_3 \in P_M\} \end{cases}$$

with $\hat{\pi}_1$ and $\hat{\pi}_2$ approximations of the pricing strategy of A_1 and A_2, learned over time from any initial configuration of the market by following our exploration procedure until all accessible vertices have been explored.

We consider two attack-scenarios: (a) the attacker A_3 maximizes its own individual profits (b) the attacker aims to increase the joints profits of all three firms (A_1, A_2, and A_3). In the first case, the objective function is the same as the one used in the main text. In the second case, the objective function becomes:

$$f_{\text{coll}}(C) = \frac{1}{|C|} \sum_{v \in C} \frac{1}{3} \left(\hat{r_1}(v) + \hat{r_2}(v) + r_3(v)\right) \Delta(v)$$

with

$$\Delta(v) = \begin{cases} 1 \text{ if } \max(\hat{r_1}(v), \hat{r_2}(v), r_3(v)) - \min(\hat{r_1}(v), \hat{r_2}(v), r_3(v)) \leq \delta \\ 0 \text{ otherwise} \end{cases}$$

Above, we relax the guarantee of equal profits at each step of the pricing sequence C by allowing the estimated profits to differ by at most δ between the three firms. We empirically set $\delta = \frac{1}{2M}$. Without relaxation, we noticed that with more firms, fewer vertices have same equal profits for all firms. Relaxing this guarantee allows to achieve high profits for all firms but not perfectly symmetric anymore.

Supplementary note 3: unnoticeable pricing objective

To maximise the profits of A_2 while keeping the profits of A_1 stable, we considered a variation of the $f_{\text{comp}}(\cdot)$ objective by adding the following constraint:

$$\left| r_1^{\leftarrow} - \frac{1}{|C|} \sum_{v \in C} \hat{r_1}(v) \right| \leq \frac{1}{M}$$

where $r_1^{\leftarrow}$ is the average profit over time obtained by A_1 before the attack starts.

Finding the best sequence of prices for A_2 subject to this constraint is an objective not tractable with the Karp's algorithm. Instead, we solve it using an exhaustive method, by iterating over all cycles of length lower than $k = 4$ in the characteristic graph.

Supplementary tables S1 to S4

Demand Type	Expression
Winner-Take-All (WTA)	$q(p) = 1_{(p=\min p)} \frac{1}{\|p=\min p\|_1}$
WTA with downward linear slope	$q(p) = 1_{(p=\min p)} \frac{1-p}{\|p=\min p\|_1}$
Logit	$q(p) = \frac{\exp(\frac{a-p}{\mu})}{\|\exp(\frac{a-p}{\mu})\|_1 + \exp(\frac{a_0}{\mu})}$

Table S1. Three major classes of demand functions are used in the algorithmic pricing literature to model the purchasing behaviors of consumers. They model the effect of increased competition or changes of prices on the consumer preferences, with different assumptions regarding consumer choices.

Experimental conditions		Average profits of the agents A_1 (r_1) and A_2(r_2)		
Agent A_1	Demand function	Hub-and-spoke	Adversarial Competition	Adversarial Collusion
TES	Strict WTA	0.002 v. 0.003	0.000 v. 0.600	0.300 v. 0.300
KLN	Linear WTA	0.091 v. 0.091	0.000 v. 0.243	0.091 v. 0.091
CAL	Logit	0.278 v. 0.233	0.229 v. 0.363	0.255 v. 0.255

Table S2. Supra-competitive profits achieved using adversarial vs. hub-and-spoke collusion settings for the three studied settings. We report the profits obtained by competitors from the literature in a 2-firm setting, against each of the three agents A_2: (a) $A_2 = A_1$ (hub-and-spoke), (b-c) our two attacker. The Adversarial Competition method (b) maximizes the attacker profits r_2, while the Adversarial Collusion (c) maximizes the profits of all agents. The reported profits are averaged over all $M^2 = 625$ initial configurations of the market, and measured after both the agent and the attacker have converged to an equilibrium. The logit demand uses the same parameters as set by Calvano et al. [4]: $a = 2$ and $\mu = 1/4$.

Experimental conditions		Average profits of A_1, A_2, A_3					
Competitors (A_1, A_2)	Demand function	Adversarial Competition			Adversarial Collusion		
		r_1	r_2	r_3	r_1	r_2	r_3
TES	Strict WTA	0.007	0.007	0.280	0.025	0.044	0.020
KLN	Linear WTA	0.000	0.000	0.238	0.060	0.060	0.064
CAL	Logit	0.132	0.167	0.277	0.139	0.134	0.142

Table S3. Adversarial competition and collusion in a 3-firm setting for the three studied settings. We report the profits r_1 (competitor), r_2 (competitor), and r_3 (attacker) obtained against each of the three agents from the literature. The Adversarial Competition method (a) maximizes the attacker profits r_3, while the Adversarial Collusion (b) maximizes the relaxed joint profits of A_1, A_2, and A_3 (see Section). The reported profits are measured after both the competitor and the attacker have converged to an equilibrium.

Experimental conditions		Pricing strategies of the agent A_2	
Agent A_1	Demand function	Hub-and-spoke	Adversarial unnoticeable
TES	Strict WTA	0.002 v. 0.003	0.000 v. 0.600
KLN	Linear WTA	0.091 v. 0.091	0.062 v. 0.186
CAL	Logit	0.278 v. 0.233	0.242 v. 0.360

Table S4. Supra-competitive profits achieved using adversarial unnoticeable vs. hub-and-spoke collusion settings for the three studied settings. We report the profits obtained by competitors from the literature in a 2-firm setting, against each of the three agents A_2: (a) $A_2 = A_1$ (hub-and-spoke), (b) our unnoticeable attacker. The unnoticeable method (b) maximizes the profits of the attacker, subject to the competitor profits staying within $\frac{1}{M} = 0.04$ currency units of their profits before the attack.

Supplementary figures S1 to S6

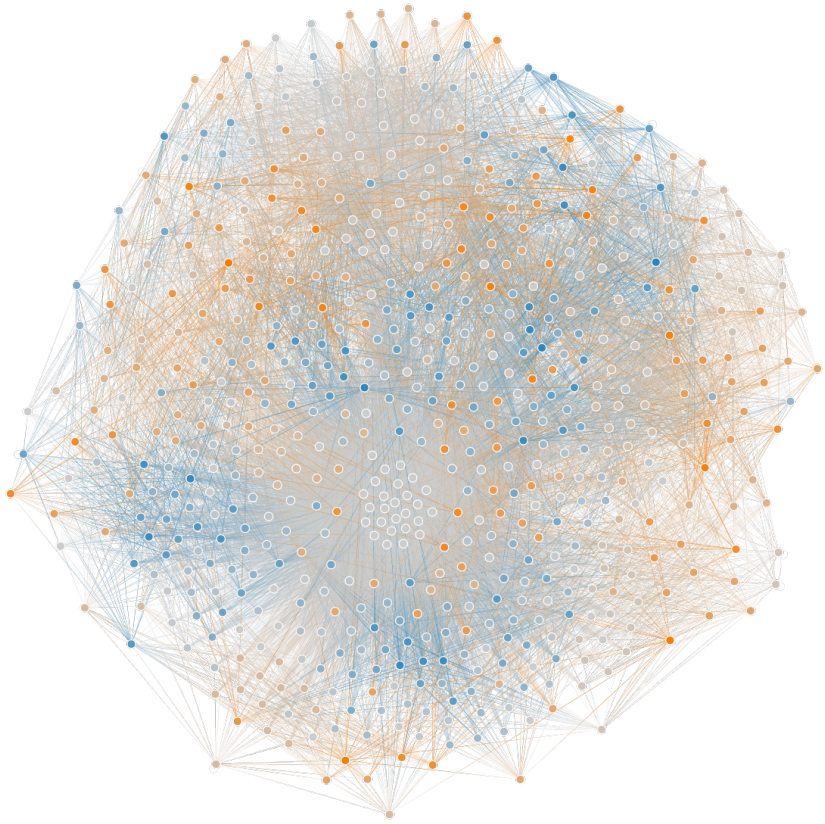


Figure S1. Characteristic graph *G* in the TES experiment. Each node contains *M* = 25 outgoing edges. We represent in blue the vertices with higher attacker profits and in orange the vertices with higher competitor profits. The network forms a single connected component, with no visible structure between higher attacker and competitor profits.

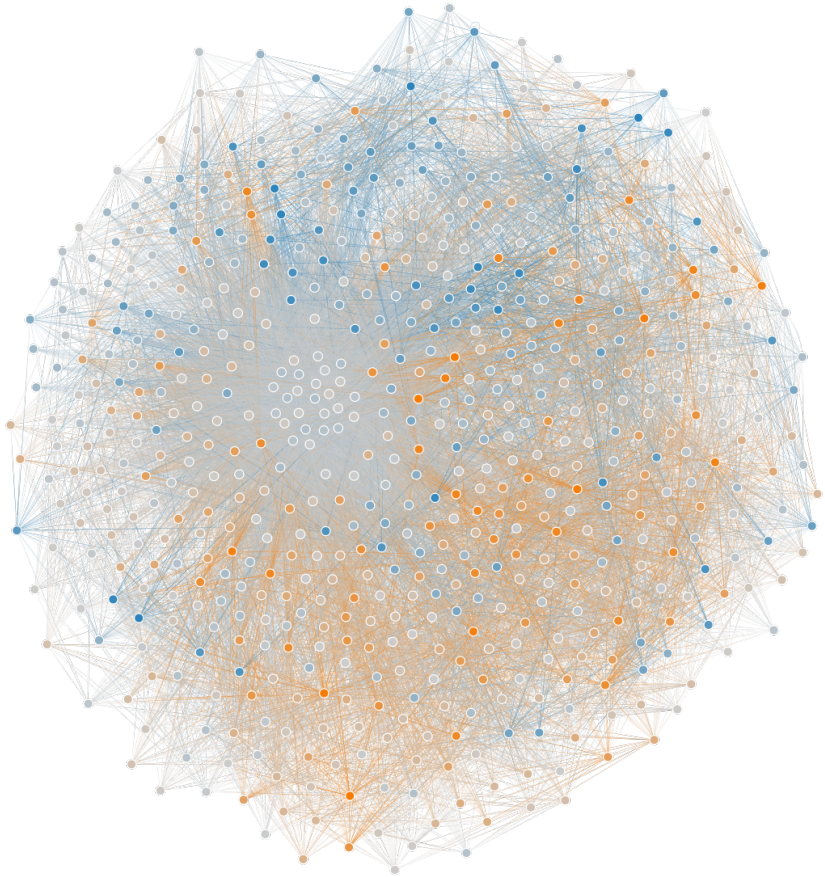


Figure S2. Characteristic graph G in the CAL experiment. Each node contains $M = 25$ outgoing edges. We represent in blue the vertices with higher attacker profits and in orange the vertices with higher competitor profits. The network forms a single connected component, with no visible structure between higher attacker and competitor profits.

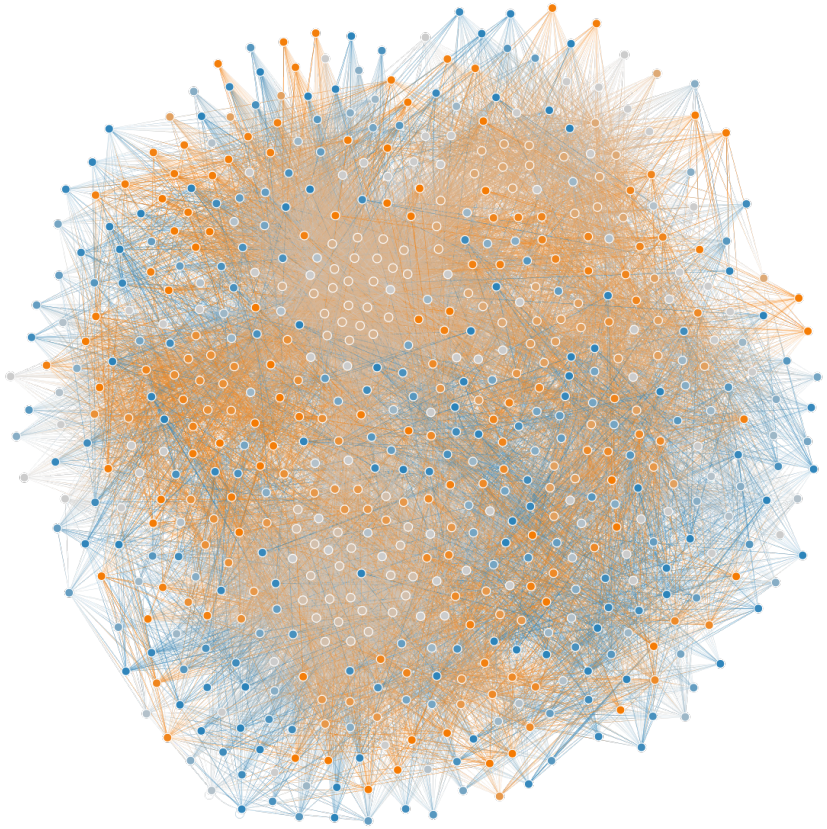


Figure S3. Characteristic graph G in the KLN experiment. Each node contains M = 25 outgoing edges. We represent in blue the vertices with higher attacker profits and in orange the vertices with higher competitor profits. The network forms a single connected component, with no visible structure between higher attacker and competitor profits.

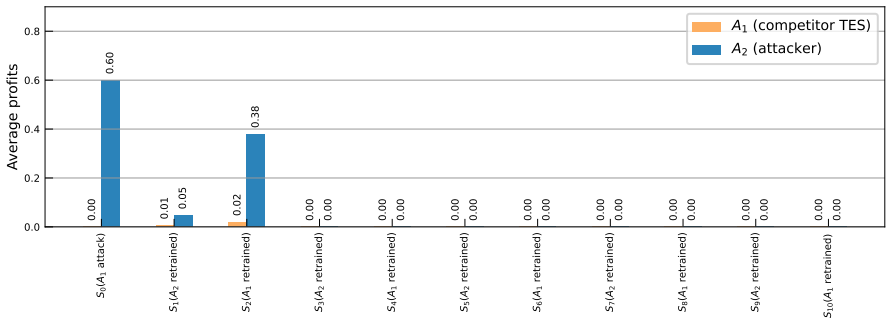


Figure S4. Profits achieved by each agent over time in an arms race scenario in the TES experimental setup. Reported profits are measured after the agent has fully retrained its pricing algorithm (when A_1 is retrained) or after the attacker has finished exploring the new pricing strategy of the agent (when A_2 is retrained) .

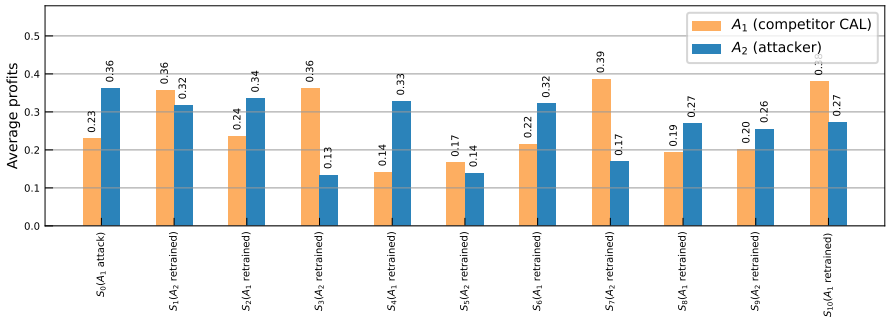


Figure S5. Profits achieved by each agent over time in an arms race scenario in the CAL experimental setup. Reported profits are measured after the agent has fully retrained its pricing algorithm (when A_1 is retrained) or after the attacker has finished exploring the new pricing strategy of the agent (when A_2 is retrained) .

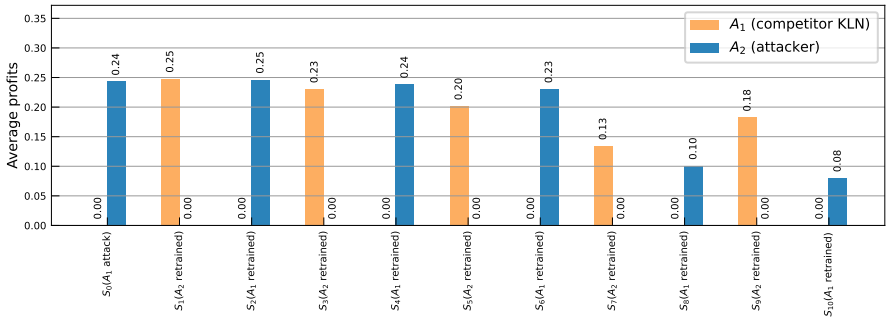


Figure S6. Profits achieved by each agent over time in an arms race scenario in the KLN experimental setup. Reported profits are measured after the agent has fully retrained its pricing algorithm (when A_1 is retrained) or after the attacker has finished exploring the new pricing strategy of the agent (when A_2 is retrained) .

References

1. Sutton, R. S. & Barto, A. G. *Reinforcement Learning: An Introduction* (MIT Press, Nov. 2018).
2. Tesauro, G. & Kephart, J. O. Pricing in agent economies using multi-agent Q-learning. *Auton. Agent. Multi. Agent. Syst.* **5,** 289–304 (2002).
3. Klein, T. Autonomous Algorithmic Collusion: Q-Learning Under Sequential Pricing. *RAND J. Econ.* **52,** 538–558 (2021).
4. Calvano, E., Calzolari, G., Denicolò, V. & Pastorello, S. Artificial Intelligence, Algorithmic Pricing, and Collusion. *Am. Econ. Rev.* **110,** 3267–97 (Oct. 2020).
5. Hansen, K. T., Misra, K. & Pai, M. M. Frontiers: Algorithmic Collusion: Supra-competitive Prices via Independent Algorithms. *Marketing Science* **40,** 1–12 (2021).
6. Moore, E. F. *The shortest path through a maze* in *Proc. Int. Symp. Switching Theory* (1959), 285–292.
7. Karp, R. M. A characterization of the minimum cycle mean in a digraph. *Discrete Math.* **23,** 309–311 (1978).