Review of Industrial Organization (2019) 55:155–171
https://doi.org/10.1007/s11151-019-09689-3

Algorithmic Pricing What Implications for Competition Policy?

Emilio Calvano[1,3] · **Giacomo Calzolari**[2,3,4] · **Vincenzo Denicolò**[1,2] · **Sergio Pastorello**[1]

Published online: 9 February 2019

Abstract
Pricing decisions are increasingly in the "hands" of artificial algorithms. Scholars and competition authorities have voiced concerns that those algorithms are capable of sustaining collusive outcomes more effectively than can human decision makers. If this is so, then our traditional policy tools for fighting collusion may have to be reconsidered. We discuss these issues by critically surveying the relevant law, economics, and computer science literature.

Keywords Algorithmic pricing · Competition policy · Artificial intelligence · Machine learning · Collusion

JEL Classification D42 · D82 · L42

✉ Giacomo Calzolari
giacomo.calzolari@eui.eu

Emilio Calvano
emilio.calvano@unibo.it

Vincenzo Denicolò
vincenzo.denicolo@unibo.it

Sergio Pastorello
sergio.pastorello@unibo.it

[1] Dipartimento di Scienze Economiche, University of Bologna, Piazza Scaravilli 2, 40126 Bologna, Italy

[2] CEPR, London, UK

[3] Toulouse School of Economics, Toulouse, France

[4] European University Institute and University of Bologna, Via delle fontanelle 18, 50014 Florence, Italy

1 Introduction

The geographical fragmentation of markets has dramatically fallen in Europe in the last decades. The European Common Market has removed artificial barriers to the free movement of goods, technological progress has significantly decreased transportation costs, and the advent of electronic commerce has greatly enlarged the set of potential suppliers to which a typical buyer has access. As a result, for many types of goods Europe is today a single geographical market.

This process has brought about substantial benefits but has also created new challenges. In this paper, we focus on one side-effect of electronic commerce: the diffusion of algorithmic pricing. Firms' pricing decisions are increasingly delegated to software programs that incorporate the latest developments of artificial intelligence. While pricing algorithms have been used by airline companies for decades,[1] only more recently have they been adopted in other sectors, such as financial markets and the hotel and insurance industries. Still more recently, the diffusion of Algorithmic Pricing (AP) has extended beyond these domains and is now much more widespread. For example, Chen et al. (2016) document that a significant fraction of sellers in a large online marketplace (Amazon US), where many different types of goods are traded, adopted algorithmic pricing in 2015.[2]

AP does not seem to be a fad. As increasing numbers of transactions take place in digital environments, and the software technology further improves, it is likely that the demand for AP will keep increasing. And the supply of AP is likely to keep up with the demand. AP has become affordable even for small businesses, as off-the-shelf machine learning solutions and computing capability are now being supplied by tech giants such as Amazon, Google and Microsoft. More entry in this new industry may well occur in the future. Recent developments—such as the Distributed Digital Ledgers and the Internet of Things—may further fuel the growth in the demand for and supply of AP.

The diffusion of AP raises various concerns for competition policy and regulation. One concern is that AP tremendously enlarges the scope for price discrimination. While the competitive effects of price discrimination are generally uncertain, with AP prices may be conditioned not only on relatively innocent information such as the timing of the purchase or the firm's residual capacity, but also on the buyer's entire past purchasing history. Such conditional pricing may lead to consumer poaching; or it could lead to the use of exclusivity or market-share discounts –both of which may have anti-competitive effects. Furthermore, the commercial exploitation of information that should arguably remain private may raise issues of privacy.[3]

[1] British Airways seems to have been the first company to use pricing algorithms in the 1970s.

[2] The European Commission's 2017 "Final report on the E-commerce Sector Inquiry" concludes that "*A majority of retailers track the online prices of competitors. Two-thirds of them use software programs that autonomously adjust their own prices based on the observed prices of competitors*".

[3] AP could provide a competing explanation (and thus an identification challenge) for the evidence of higher online (relative to offline) prices in some markets. The prevalent explanation is that of an increase in the match quality (Ellison and Ellison 2018). AP could also speak to the question of what is causing (online) price dispersion both in the cross-section and in the time-series in seemingly homogenous product markets (Chen et al. 2016).

Another source of concern is the possibility that AP may facilitate collusion: this concern has been repeatedly voiced in recent years, both in the popular press[4] and in the academic literature. In particular, Ezrachi and Stucke (2015) and Mehra (2016) were the first in the law literature to point explicitly to the risk that AP may inhibit competition and effectively sustain collusion with no need of human intervention. The issue is now on the radars of various antitrust agencies, such as the US Federal Trade Commission and the European Commission.[5]

These concerns may seem somewhat speculative as, so far, the evidence of digital cartels seems limited. To the best of our knowledge, the only antitrust case that involved algorithmic pricing was the US's and UK's agencies successful challenge of a pricing software that was allegedly designed to coordinate the price of posters by multiple online sellers.[6] Commentators have also pointed to a few specific examples: such as software programs that appear to have escalated the price of a second-hand book to the millions of dollars[7]; or, more important, the use of the same pricing algorithm for car parts by several manufacturers, which allegedly resulted in billions of extra profits in the European market.[8] However, we have not seen much real action on the antitrust front so far.

In the light of this evidence—or lack thereof—the optimistic view of AP is that these programs are not really more conducive to collusive outcomes than is human pricing. According to this view, the concerns mentioned above are exaggerated. The pessimistic view, in contrast, maintains that the anticompetitive potential of AP has not fully materialized yet, as AP is still in its infancy. Furthermore, pessimists argue that antitrust authorities may have refrained from intervening because they are not well equipped to cope with this new form of collusion.

In this paper, we contribute to this debate, which so far has involved for the most part law scholars and, especially, computer scientists.[9] Bringing an economic perspective into the debate may be useful, as AP raises a number of important economic questions. Can "intelligent" pricing algorithms learn to collude? Is AP collusion any different from collusion among humans? Is AP conducive to collusion more often than what humans could do? If the answers to these questions are affirmative, how can we detect algorithmic collusion? What are the appropriate new standards for competition policy?

[4] For example, the *New Yorker* asked what happens "*When bots collude*" (April 25, 2015), and the *Financial Times* wrote of "*Digital cartels*" (January 8, 2017).

[5] The Acting Chairman of the U.S. Federal Trade Commission M. Ohlhausen "Should We Fear the Things That Go Beep in the Night? Some Initial Thoughts on the Intersection of Antitrust Law and Algorithmic Pricing," remarks at the "Concurrences Antitrust in the Financial Sector Conference," New York, May 23, 2017. OECD Roundtable on Algorithms and Collusion, June 2017. The European Commissioner for Competition M. Vestager, "Algorithms and Competition," remarks at the Bundeskarellamt 18th Conference on Competition, Berlin, March 16, 2017.

[6] Wired Magazine; U.S. v. Topkins, 2015 and CMA case 2015 n. 50223.

[7] See Olivia Solon, 2015, "How a book about flies came to be priced $24 million on Amazon".

[8] See https://theblacksea.eu/stories/article/en/car-parts-probe.

[9] Harrington (2017) develops a legal approach for collusion with AP that is grounded in economic analysis.

In the next sections, we shall briefly address these questions. Since economic research on AP is still in its infancy, we are not in a position to provide answers. Our aim is, more modestly, to clarify the terms of the debate.

2 Adaptive and Learning Algorithms

At the cost of oversimplifying, one can distinguish between two classes of algorithms for pricing: "adaptive" and "learning" algorithms. We discuss them separately as the competitive concerns that they raise may be different.

2.1 Adaptive Algorithms

First-generation pricing algorithms are adaptive in nature. These algorithms are, essentially, sets of rules that dictate optimal responses to specific contingencies. Dynamic pricing for revenue management, which has been used for some time in hotel booking and airline services, belongs to this class.

Adaptive pricing algorithms typically perform two activities: estimation and optimization. Accordingly, they may be viewed as comprising an estimation module and an optimization module. The estimation module estimates market demand using past volumes and prices, and possibly other control variables. The optimization module then chooses the optimal price given the demand estimate and observed past behavior of rivals.

When market conditions are known, so that the estimation function is inactive, adaptive algorithms essentially set a firm's price as a function of rival's past prices. This adaptive behavior may be more or less sophisticated. In some cases, the "optimization" module actually boils down to a fixed and perhaps somewhat arbitrary adjustment rule. For example, an algorithm may set the own price as a fraction (or multiple) of the rival's price. In other cases, the optimizing behavior is more sophisticated. For example, the algorithm may calculate a best response to rivals' hypothetical strategies. Examples include: "best response dynamics" (where the firm's price is the best response to the competitors' last period prices); "fictitious play" (where the firm plays a best response to a fictitious mixed strategy which is taken to be the past price distribution); and what is called, perhaps imprecisely, "Bayesian learning" (where the firm plays a best response to a weighted average of past previous prices with exponentially declining weights).[10]

These forms of adaptive behavior, where a player plays a static best response to some combination of the rivals' past strategies, have been theoretically analyzed by Milgrom and Roberts (1990). They show that in supermodular games (a class of games with strategic complementarities that includes the typical Bertrand pricing game), such adaptive behavior generally converges to outcomes that do not exhibit

[10] The term may be imprecise as these algorithms learn only in a very limited sense, as we shall clarify below.

collusion. For example, in pricing games the system converges to prices that are no higher than the Nash equilibrium prices of the one-shot pricing game.[11]

This result has been taken to suggest that AP does not really make collusion any easier to achieve. To produce collusive outcomes, the software must not play static best responses; rather, it must be instructed to condition its action on the rivals' past behavior in a collusive fashion.

Of course, not all types of such conditioning lead to collusive outcomes. For example, it may be easy to design rules that mechanically lead to high prices. However, collusion requires that the prices be *profitably* high.[12] The problem is that prices may be profitably high in many different ways, which may benefit different firms to different degrees. Adaptive algorithms must therefore be instructed to coordinate on one of many possible outcomes. Furthermore, adaptive algorithms must be instructed to support such coordination by a system of punishments in the event that the rivals defect. Both types of instructions must be fed into the software: Adaptive algorithms cannot collude unless they are designed by their programmers to do so.

But if this is so, then the programmers must solve exactly the same coordination problems as human price makers. From this observation, skeptics draw two conclusions: First it is unlikely that independent programmers—or independent managers who are instructing their own programmers—can achieve any significant degree of coordination without explicitly communicating. This implies that AP collusion may be proved by exactly the same type of evidence as for traditional collusion: minutes of meetings, phone calls, emails, etc. In other words, antitrust policies can continue to focus on the "meeting of the minds", which is the hallmark of explicit collusion.

On top of that, and this is the second conclusion, collusive AP software must include lines of coding that reveal the programmers', or the managers', collusive intent (as in *U.S. v. Topkins*, 2015). This in fact contributes to generate the "hard" evidence that antitrust authorities seek. In sum, AP collusion is harder to achieve and, if anything, easier to detect than human collusion.

Still, even adaptive algorithms differ from humans in one important way: the frequency of interactions. AP may react to rivals' actions much more quickly than do human beings. This property of AP has been emphasized by Ezrachi and Stucke (2015) and Mehra (2016). As is well known, simple models of collusion predict that more frequent interaction makes collusion easier to sustain, as defection is punished more promptly and hence the gains from defection are reaped for a shorter time. From this, Ezrachi and Stucke (2015) and Mehra (2016) conclude that collusion is more likely with AP.

In fact, the result that more rapid responses facilitate collusion has been questioned in more recent theoretical contributions. Sannikov and Skrzypacz (2007) argue that faster interaction may actually impede collusion under imperfect

[11] Whether this result survives when market conditions are not stationary-- and hence the estimation function of the pricing algorithm is active-- is an open question.

[12] For example, the second-hand book episode seems to have been generated by adaptive algorithms which would set the own price as a multiple of the rival's price. If two firms adopt a pricing rule of the type $p_i = a_i p_j$, the system explodes whenever $a_i a_j > 1$. As a results prices may get very high indeed, but the outcome will be poor in terms of profit maximization.

observability of the rivals' actions. The reason for this is that responding too quickly to noisy information may unravel the collusive scheme.[13]

In any case, even if the traditional result holds so that AP does make collusion easier, with adaptive algorithms there seems to be no reason to change the traditional policy approach. Antitrust authorities should look for the usual type of evidence of direct communication among the competing parties; the only difference is that they should focus on codes and programmers as witness or complicit in addition to final decision makers.

2.2 Learning Algorithms

Second-generation algorithmic pricing is based on more recent developments in computer science, which belong to the field of Artificial Intelligence and, more specifically, on Machine Learning (ML). Rather than specifying a pricing problem and instructing the software to solve it, with ML the software learns how to solve the task from experience.[14]

To gain such experience, ML algorithms experiment with strategies that would be sub-optimal according to their current knowledge. Experimentation is costly in that it entails, in expectation, a sacrifice of profits. However, it is valuable as it allows learning from more diverse situations.[15]

Fudenberg and Levine (2016, p. 157) define learning as "passive" when:

> players have no incentive to change their actions to gain additional information.

This is precisely what adaptive algorithms do: As time passes and new information becomes available, the estimation of market conditions may improve. ML pricing algorithms, in contrast, exhibit a different, more "active" type of learning: They experiment to learn the consequences of possibly suboptimal prices.

With ML, there is no need to specify a model of the market, estimate the model, and solve for the optimal strategy. The programmer instead chooses only which variables the strategy should be conditioned on, how frequently the program must experiment, and how much weight to give to more recent experience relative to the cumulated stock of knowledge. Starting from any arbitrary assessment of the value of the feasible strategies, the program then updates these values. In this fashion, the algorithm is "model-free" and learns to play optimally from experience.

For example, a ML program designed to play chess need not be fed with any notion of chess strategy. All the program needs to know are the rules of the game,

[13] Sannikov and Skrzypacz (2007) derive this result by assuming that agents optimally extract the signal from the noisy information that they receive. Whether the result continues to be true also for AP remains to be investigated.

[14] ML techniques have been developed and are currently adopted for a large number of applications. By far the most popular in the social sciences are those of classifying tasks (with supervised learning) and those meant to uncover hidden structure in big datasets (with unsupervised learning).

[15] If the environment is stationary, experimentation is crucial initially but may optimally vanish eventually; in a dynamic environment, in contrast, it may be optimal to keep experimenting forever.

and an initial assessment of the value of each possible position. This can be extremely simple (for example, the difference between the total value of own pieces and the opponent's pieces, and $\pm\infty$ if checkmate follows in one move), and it could even be more arbitrary. The program learns from experience how to assess every possible position, and hence how to play optimally.

As mentioned above, during the learning phase suboptimal decisions may be taken frequently. This is costly, and the learning phase may last for quite a while. (In practice, the problem can be alleviated by letting the program learn in a simulated environment before using it for making real pricing choices.) After the initial phase, however, the fact of being model-free gives ML pricing algorithms a great advantage over adaptive algorithms and first-generation revenue management tools—especially in complex and quickly changing environments.

From the viewpoint of competition policy, however, the concern is that ML pricing algorithms may learn not only how to cope with a rapidly changing market environment, but also how to cope with competitors. Since collusion is profitable, algorithms that learn from experience "too well" may actually learn to collude—even if they have not been specifically designed to do so.

There is no doubt that sustaining collusion is a daunting task, because firms must coordinate on both a collusive outcome and a punishments mechanism. However, thanks to their proclivity to explore, ML algorithms might solve both coordination problems quite effectively. An important theoretical problem is therefore to ascertain whether, and to what extent, learning algorithms may learn to collude.

This problem is still open. Various factors hint to a positive answer. For example, Milgrom and Roberts' negative result does not apply to ML algorithmic pricing, whose behavior is not adaptive. Furthermore, ML software is known to outperform humans in other fields. For example, ML software that plays chess can beat the current world champion easily. This suggests that ML programs may learn to collude more effectively than do humans.

But these arguments are too loose to be the basis for policy, and the question deserves a more systematic treatment. Differently from adaptive algorithms, learning algorithms may come to solve the coordination problems even if they have been designed innocently. Programmers and managers who delegated the task, need not instruct the program to coordinate on a specific outcome, nor to adopt a specific system of punishment. Therefore, they need not communicate to achieve efficient coordination. This implies that to the extent that learning algorithms do collude, then this type of collusion poses new challenges to competition policy.

3 Q-Learning in a Simple Pricing Game

In this section we focus on a specific class of algorithms: Q-learning algorithms. These algorithms are well understood, relatively simple, and they form the basis for more sophisticated algorithms.

Generally, Q-learning tools tackle the problem of finding an optimal policy in Markov Decision Problems (or MDPs) or similar problems.[16] A MPD is a formal framework that analyze repeated decision-making in dynamic and stochastic environments. To be concrete: consider the problem of a price-setting firm in an oligopoly market. In every period the firm: (1) observes relevant information such as the price that its rivals charged in previous periods or the state of demand; (2) sets its own price; and (3) collects the resulting profits. The firm's problem is that of finding the pricing policy that maximizes the present value of its profits. A policy is a mapping from what it observes—the "state"—to its control variable: the price. The Q-Learning algorithm is a tool that is designed to "crack" this decision problem through a process of experimentation. Experimenting allows the program to learn the policy that maximizes long run profits.[17]

Q-learning algorithms are particularly appealing for a number of reasons: First, they learn the optimal policy while playing and thus do not need to be trained with data (in contrast with supervised learning). They can be put immediately into production. Second, they do not require knowledge of the effect of one's actions on the environment (that is the effect of actions on future states). For example, in the pricing task they do not require the programmer to specify the consumers' or rivals' future reaction to one's price. Finally, they can confront a great deal of uncertainty and work with very little information.

How does the algorithm learn? A key ingredient is the Q-matrix (from Q-learning), which stores an estimate of the present value of choosing a particular action in a particular state. For concreteness: Suppose that the pricing task for firm i is to choose every period from one of two exogenously given prices, $p_i = 1$ or $p_i = 2$. The firm observes the prices charged last period by its rival p_j and keeps track of its own past price. These two prices account for the present state $\mathrm{s} = (p_i, p_j)$. There are thus 4 possible states, one for each combination of last period prices. In this simple example, the Q-matrix is a 4×2 matrix:

$\mathrm{Q_i(s,p_i)}$	$p_i = 1$	$p_i = 2$
$\mathrm{s_1} = (1,1)$		
$\mathrm{s_2} = (1,2)$		
$\mathrm{s_3} = (2,1)$		
$\mathrm{s_4} = (2,2)$		

The rows indicate the four possible distinct states, and the *kh*th entry can be interpreted as an estimate of the present value of the stream of profits that are obtained by following the choice of price h in state k.

[16] For a textbook introduction to Q-learning see Sutton and Barto (1998).

[17] To economists, the analysis of a MDP may be reminiscent of the analysis of Markov Perfect industry dynamics, which are summarized in Doraszelski and Pakes (2007). There are however substantial differences: In that literature, the numerical methods for equilibrium identification rely on the industry structure and solve systems of firms' optimality conditions. Here instead, the algorithms are model free and learn with experimentation.

The matrix is initialized with some arbitrarily assigned values in the first period and is then updated on the basis of experience. The updating takes places as follows. Let π_i denote the observed period profit of firm i that results from charging price p_i. Let the present state be s—the prices that were set by firms in the previous period—and the future state $s' = (p_i, p_j)$ which is determined by the current price of firm i and that of the other firm j, respectively p_i and p_j. At the end of the current period, the new value of the Q-matrix at the (s, p_i) cell then is updated as follows:

$$Q_i^{new}(s, p_i) = (1 - \alpha)Q_i(s, p_i) + \alpha\left[\pi_i + \gamma \max_{q_i} Q_i(s', q_i)\right]$$

where the positive parameter $\alpha < 1$ is the *learning rate* and $\gamma < 1$ is a discount factor.

The Q-learning equation is reminiscent of a Bellman equation in dynamic programming.[18] Analytically, the learning dimension is captured by the term inside the square brackets in the previous expression. This term captures the present and future consequences of a price p_i as opposed to the current value $Q_i(s, p_i)$ that associated with that price (the first term in the right-hand side of the equation). Given the chosen price p_i and initial state s, the corresponding cell (and only that cell) is updated. Notice that the updating takes into account not only the current realized profit π_i, but also the dynamic (future) gains that can be obtained once the environment evolves into the new state s'. In this way, strategies that perform well are reinforced, as their Q-value increases (this is why Q-learning is an instance of Reinforcement Learning), while none of the other cells are affected.

The choice of a price level balances two needs: gathering new information (exploring) and reaping profits (exploiting). In other words, exploitation means that the information already gathered is used to choose the action that corresponds to the higher estimate of the present value. This is the price (action) x that corresponds to the largest value $Q_i(s, p_i)$ in state s. The balancing between exploitation and exploration is simple: In every period, the algorithm explores (and thus does not exploit) with an exogenously given probability ε and exploits setting the currently optimal price with probability $1 - \varepsilon$. When it explores, the algorithm may choose the other price—with the lowest Q-value in state s—in a simple two-actions environment. More generally, it randomizes with a uniform probability among all feasible choices. The probability of exploration ε can be a time invariant parameter, in which case the "ε-*Greedy*" *algorithm* keeps experimenting with constant probability. A common alternative is the *Boltzmann exploration algorithm* where the probability of selecting price p_i at time t is

$$\frac{e^{Q_i(s,p_i)/\beta}}{\sum_p e^{Q_i(s,p)/\beta}}$$

[18] Q-learning is also related to the idea of active learning in game theory (see Fudenberg and Levine 2016). It is normally associated with the more general class of "reinforcement learning" models where an agent learns by interacting with the environment, perceiving its state, and taking actions with trials and errors. Those actions that are associated with a positive consequence (reward) then have higher chances of being chosen in the future: They are reinforced by the agent's behavior.

and β is a "temperature" that monotonically decreases with time and tends to zero. Always favoring actions with higher Q-values, at the beginning when β is large, the algorithm explores significantly and for very low β it almost always takes the "greedy" optimal action. This dependence of the "temperature" on time favors exploration at early stages, and should help convergence over time.

A key property is that Q-Learning is model-free, as one does not need to specify a demand function and to conjecture about the rival's behavior. All that is needed to calculate the next period strategy is observed at the end of the current period.

What does the algorithm learn? Does it learn and converge to the "optimal policy": the mapping from states into actions that maximizes the present value of the future stream of profits? For optimization problems that do not involve strategic interaction, Watkins and Dayan (1992) showed that Q-learning eventually always finds the optimal policy—provided that certain conditions are met. Specifically, they show that if the environment is stationary and there is enough exploration, so that all the cells of the matrix have been "visited" often enough, then under certain technical conditions the Q matrix converges to a matrix that induces the agent to play the optimal policy.

With many players (or multi-agent environments in the jargon of computer scientists) and strategic interaction, there exists no general result of convergence. The difficulty is that learning is here a "moving target" because algorithms simultaneously explore and affect the environment and how it responds to their individual experimentation. Formally, in a strategic environment it not true that the state representation of the past contains all of the information that is relevant for decisions (the Markov property), and the environment is not stationary. Although this is certainly a limitation, it does not imply that Q-learning algorithms do not perform in strategic environments—which remains an empirical issue, at least at this stage.

Simple variants of Q-learning assume no memory, which in the previous description involved one past period and its state s. With no memory the algorithm cannot account for a current action's future consequences and it must thus be $\gamma = 0$. This was the case, for example, in some early economic applications such as Roth and Erev (1995), Erev and Roth (1998) and Sarin and Vahid (2001).

Relatedly, a fully model-free Q-learning approach would completely disregard the mutual interdependence. In our previous environment this would amount to disregarding the other firm's (present and past) price altogether. This simple application of single-player Q-learning to strategic environments, clearly cannot guarantee convergence because, again, the Markov Property would be lost, with other firms' actions entering the noise of the (not-stationary) environment.[19]

The computer science literature identifies three important challenges for algorithms: stability; adaptation; and scalability.[20] Stability refers to convergence to a policy that, after a certain number of repetitions, does not change. In our previous environment, a policy identifies, for each state s (i.e. past prices) what is the price

[19] Still, Busoniu et al. (2008) account for cases in which these algorithms have show good performances in terms of convergence.

[20] For a comprehensive survey see Busoniu et al. (2008).

to set in the current period. It is then stable if after some time t, the chosen price for a given state s is always the same.[21] Adaptation instead guarantees that a good performance of the algorithm is maintained even if some of the players change their policy. Finally, scalability pertains to the possibility to apply the algorithms to richer environments with many players, states, and actions (which could be also continuous variables). The challenge here is that the tabular representation and storage of extremely large Q-matrixes becomes quickly non-viable. The computer science literature has in recent years worked intensively to address these challenges—independently or in combination—with a trade-off that is often seen between stability and adaptation (see Bloembergen et al. 2015, for a recent survey).

4 An Agenda for Future Research

Armed with the Q-learning algorithm of the previous section, it is relatively simple to run experiments. Our own ongoing investigations show that in the simple prisoners' dilemma it is easy to observe Q-learning algorithms that converge to cooperative actions (and strategies such as tit-for-tat and one-period-punishment). Depending on parameter values (for $\gamma, \alpha, \varepsilon$), one can easily obtain a frequency of cooperative actions as high as 95%. Cooperation has been identified also by other papers that study some forms of market competition with Q-learning, such as Tesauro and Kephart (2002), Xie and Chen (2004), Waltman and Kaymak (2008) and Dogan and Guner (2015).

While these results may be suggestive, there is still much research to be done in order to understand to what extent and under what conditions ML pricing algorithms may learn to collude. In this section we briefly discuss a number of robustness checks and challenges that should be addressed before drawing reliable conclusions. Some of these are taken on in Calvano et al. (2018) where algorithmic collusion is explored in the much complex Bertrand oligopoly setting, with price competition and logit-differentiated demand.

The simple prisoners' dilemma contains important elements of price competition: mutual interest to set high prices, together with individual incentives to undercut. However, it is simple and probably simplistic description of actual interactions in markets. Are algorithms able to cooperate also in complex and realistic environments? For example, just enlarging the set of possible prices would make the coordination problem more complex and affect the scalability of the algorithms. Even if theoretical models often sidestep the issue of what to coordinate on, it seems natural to conjecture that this would make collusion more difficult to achieve.

Introducing asymmetries and increasing the number of firms also make collusion more difficult, as has been shown by a number of theoretical studies. Still, it would be interesting to check whether these predictions—which have been obtained for

[21] Note that this does not imply that the values of the Q matrix remain unchanged, but only that the ranking between $Q_i(s,1)$ and $Q_i(s,2)$ is unchanged.

rational agents—hold also for Q-learning algorithms when considering the issue of scalability explicitly.

Another important element is intrinsic uncertainty: theory suggests that uncertainty may make cooperation more difficult, as players might misinterpret negative outcomes as due to rivals' defection rather than bad market conditions. Is this the case also for algorithms?

Last but not least, the analysis should be extended to learning algorithms that are more sophisticated than Q-learning. It seems likely that pricing algorithms that are adopted in practice indeed incorporate the latest developments of Artificial Intelligence and hence are more clever than simple Q-learning.

Some of these developments may be useful to address the challenges that were mentioned above. For example, enlarging the set of strategies—which is necessary to understand whether algorithms may learn what strategies to coordinate on-may exponentially increase the computational burden. However, very recent advances in the development of Neural Networks and Deep Learning methods allowed successfully coping with high-dimensional state spaces. They do so by giving the algorithms the ability to generalize from experience. That is, the algorithm extrapolates what would happen in states that have not been visited. One way to think about this is that the program uses data that have been gathered on visited states to estimate Q values for unvisited "close enough" states.[22]

Tampuu et al. (2017), for example, use a Deep Q-network, that is similar to simple Q-learning that we illustrated above; the add an approximation of the Q matrices with a non-linear neural network. Although they acknowledge that, as in our case, there is no proof of convergence for their learning algorithm, they are able to document several interesting facts. For example, they show how the multiple agents' behavior spans from very competitive in zero-sum games, to significant cooperation when less confrontational payoffs appear in the stage game.[23]

Sophisticated algorithms may also learn how their rivals learn, and how to communicate with each other. For example, second-generation Q-Learning algorithms such as those that were investigated by Hu and Wellman (2003) may rely on estimates of the other players' Q matrices. Each algorithm tracks ad replicates the Q-table of all of the other players. Necessary conditions for this are that all agents use the same algorithms, all actions are publicly observable and rewards (profits) are measurable with actions.[24] Other recent algorithms—which are called "joint-action learners"—observe the actions of others to estimate their policy directly. Clearly, these approaches may be better at reaching cooperation, although they need to rely on guesses of other algorithms' future actions. Both approaches—tracking other Q-tables and joint action learners—in any case face the issue of scalability as the

[22] In a celebrated experiment Mnih et al. (2015) show how a deep reinforcement ML algorithm learned to solve complex tasks such as playing classic Atari videogames better than humans using as the state the color of 210×160 pixels on the screen with a 128-colour palette and the scoreplay.

[23] Other recent and promising examples about cooperation in a multi-agent setting is the work at Facebook AI Research: e.g., by Lerer and Peysakhovich (2018).

[24] They are not sufficient because, for example, the Max operator in the formula for Q-learning may lead to non-unique solutions.

dimensionality further increases dramatically with the number of actions, states, and other agents.

As for communication, we know from a vast literature, starting from Cooper et al. (1990), that communication plays a key role for cooperation among humans—even if it takes a form as simple as "cheap talk." In this respect, the lack of communication among pricing algorithms may significantly limit the emergence of cooperation.[25] But this limitation may be overcome by algorithms that develop their own language to communicate. Although restricted to cooperative tasks (i.e. where rewards of all algorithms are aligned, which would not apply to market competition), Sukhbaatar et al. (2016) have shown algorithms that in cooperative tasks have the ability to learn (also) to communicate amongst themselves. In a recent work, Crandall et al. (2018) have associated a state-of-the-art ML type of algorithm with some form of signaling. They showed that this approach allows cooperation to be sustained in a variety of different environments –and even when algorithms interact with humans.[26] Incidentally, we note that the latter observation refers to an environment with human and machine interaction, which is another challenge for algorithms in markets –even if one of the more spectacular successes of AI rests precisely in algorithms beating human champions at chess, Go, checkers and poker.

Ultimately, a key question is whether ML software will be more effective to coordinate and collude than are managers. The ability of ML software to "easily" coordinate and to outperform (the best) humans in many strategic environments seems to lean towards a positive answer. However, we certainly need further investigations and we must account for several and different factors. For example, experienced and knowledgeable managers may already know what a collusive outcome is. Risk aversion and concerns about reputation—which are irrelevant for a software programs, may instead refrain managers from engaging in tacit collusion. Simple and sometimes biased rules of thumb could limit managers' incentive to look for even higher profits with collusion.[27] A possible approach to check if ML software is more prone to collusion is to compare its behavior with that of humans in lab experiments (even though university students seem not to be good benchmarks for actual managers).

5 Conclusions and Implications for Policy

Our understanding of whether and how pricing algorithms may coordinate on high prices, and if collusion among algorithms is easier than among humans, is still very limited. More multidisciplinary research is necessary to ascertain whether AP really

[25] Cooper and Kühn (2014) have shown that communication between humans helps cooperation by clarifying how individuals think about the environment and whether they really mean punishing deviations and by making social punishments and rewards explicit.

[26] Salcedo (2015) presents a theoretical model that shows that collusion with algorithms is not only possible, but also inevitable. However, the result relies on algorithms' ability to read into other algorithms and thus learn their "intentions," and on some commitment not to revise the algorithms in use.

[27] With this respect it is interesting to notice that the use of learning and optimizing software may induce firms to behave more like the dynamic profit-maximizing firms of standard economic theory.

poses new problems for competition policy, and how these problems should be addressed. Yet, the analysis developed so far may shed some light on the current debate and the proposed approaches to policy.

Broadly, one may distinguish three possible approaches. The first one, based on the optimistic view that AP does not really pose any new problem, is to stick to current policy. The second approach is to regulate the introduction of pricing algorithms *ex ante*, pretty much in the same way as the commercialization of new drugs: Any new pricing algorithm should be tested by a regulatory agency to ascertain whether it exhibits a tendency to collude (in which case it would be prohibited) or not (in which case it would be approved). Finally, the third approach is to regulate *ex post*, as competition policy typically does, but with the use of different legal standards from the current ones. In particular, this approach calls for a re-assessment of the contentious issue of the prohibition of tacit collusion.

A fourth possible option is an outright prohibition of algorithmic pricing. However, there is a wide consensus that pricing algorithms may deliver big efficiency gains by allowing more efficient pricing. A per se prohibition of AP is therefore unlikely to be optimal—even if we set aside the enormous problem of implementing such prohibition in practice.

Let us start then from current policy: economists generally define collusion as a reward-punishment scheme that leads to prices and profits that are above some competitive benchmark (see, for instance, Harrington 2017). This scheme may be agreed upon by the parties explicitly or tacitly. The effects of tacit collusion are not different from those of explicit collusion. The difference between the two lies mainly in the greater difficulty of achieving coordination without communication.

The current legal standard for collusion, however, requires not only coordination on supracompetitive prices, but also some conscious and mutually accepted agreement among firms—a "meeting of minds"—to restrain competition. In other words, current policy prohibits explicit but not tacit collusion. For example, parallelism in pricing is not enough to prove collusion, as it can be the outcome, for instance, of independent reactions to common shocks.

From an economic point of view, current policy can be rationalized on the basis of an assessment of the likelihood and the costs of making mistakes—false positives and false negatives. The implicit presumption must be that it is quite unlikely that coordination is reached without explicit agreement (so that false negatives are rare), and that there are no precise methods for inferring collusion from observed price movements (so that false positives would be frequent). If this presumption is correct, then it indeed makes sense to require direct rather than circumstantial evidence of an agreement among the parties.

Those who claim that AP does not call for any change in current policy explicitly or implicitly argue that AP does not radically modify this assessment of the frequency of false positives and false negatives. In other words, they explicitly or implicitly argue that communication—among programmers rather than among final decision makers—is often essential for collusion, and that detecting implicit collusion is extremely difficult. Our discussion above suggests that these claims may be true for first-generation, adaptive algorithms but could be instead seen as inappropriate for learning algorithms.

Things are different, however, for the more sophisticated pricing algorithms that are currently available, which are capable of learning from experience. These algorithms do not need to communicate to learn to collude and—in fact, at least in their first incarnations—are not capable of communicating. Furthermore, these algorithms may learn to cooperate without any explicit collusive intent on the part of their programmers. It seems that if current policy does not change, AP may significantly increase the risk of false negatives.

The second policy approach—ex ante regulation—has been proposed both by lawyers such as Ezrachi and Stucke (2015) and economists such as Harrington (2017). Specifically, Ezrachi and Stucke (2015) proposed a "sand-box" approach (similar to the one that has been currently adopted for fintech firms in the UK) where one tries to nurture algorithms in virtual markets and monitor the association between their properties and thus the observed outcomes. Harrington (2017) proposes to check the behavior of specific AP and define a black list of algorithms that would become unlawful per se.

This regulatory approach would represent a form of public intervention much more intrusive than competition policy, which acts ex post rather than ex ante. Normally, such intrusive intervention is reserved for cases where market failure is evident and costly, and the efficiency losses due to the regulation are limited. It is unclear whether these conditions are met in the case of AP.

Another problem is that the collusive properties of pricing algorithms may depend on which other algorithms they interact with. Suppose that algorithm A has been approved on the basis of evidence that it does not tend to collude with existing algorithms B, C, and D. Suppose, however, that subsequently a new, superior algorithm E is developed. And suppose that the new algorithm E tends to collude with A, but not with B, C, and D. Which algorithm should be prohibited? E is better than A, so on efficiency grounds E should be approved and A prohibited. But A was approved at the outset, and it may be costly to outlaw it at a later stage (for example, firms may have sunk investments in technology A).

The difficulties of pursuing the first two approaches may suggest taking seriously the third one, ex post intervention. If algorithmic pricing indeed makes collusion easier—and, in particular, if it dispenses with the need for direct communication among the parties—then the likelihood that current policy may lead to false negatives may be significantly higher. If this is so, then the balance between explicit and tacit collusion that underlies current policy may have to be reconsidered.

As is well known, however, detecting tacit collusion is problematic. Future research should therefore focus not only on the possibility that AP may facilitate collusion, but also on the new features that tacit collusion may exhibit under AP.

At present, we simply do not know enough about AP to make definitive policy recommendations. We believe that this is a field where new theoretical and empirical research may substantially pay off in terms of better policy.

Acknowledgements We thank the editor Larry White, the guest editors Christos Genakos, Michael Pollitt, and the discussant Patrick Legros and participants at the conference "Celebrating 25 Years of the EU Single Market" organized by the Review of Industrial Organization in Cambridge Judge Business School, 2018. Financial support from the Digital Chair of the Toulouse school of economics is gratefully acknowledged.

References

Bloembergen, D., Tuyls, K., Hennes, D., & Kaisers, M. (2015). Evolutionary dynamics of multi-agent learning: A survey. *Journal of Artificial Intelligence Research, 53,* 659–697.

Busoniu, L., Babuska, R., & De Schutter, B. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, *38*(2), 156–172.

Calvano, E., Calzolari, G., Denicolò, V., & Pastorello, S. (2018). *Artificial intelligence, algorithmic pricing and collusion*. CEPR Discussion Paper13405.

Chen, L., Mislove, A., & Wilson, C. (2016). An empirical analysis of algorithmic pricing on Amazon marketplace. In *Proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee.

Cooper, R. W., DeJong, D. V., Forsythe, R., & Ross, T. W. (1990). Selection criteria in coordination games: Some experimental results. *The American Economic Review, 80*(1), 218–233.

Cooper, D. J., & Kühn, K. U. (2014). Communication, renegotiation, and the scope for collusion. *American Economic Journal: Microeconomics, 6*(2), 247–278.

Crandall, J. W., Oudah, M., Tennom, Ishowo-Oloko, F., Abdallah, S., Bonnefon, J., et al. (2018). Cooperating with machines. *Nature Communications, 9*(233), 2018.

Dogan, I., & Guner, A. R. (2015). A reinforcement learning approach to competitive ordering and pricing problem. *Expert Systems, 32,* 39–47.

Doraszelski, U., & Pakes, A. (2007). A framework for applied dynamic analysis in IO. In M. Armstrong & R. H. Porter (Eds.), *Handbook of industrial organization* (Vol. 3, Chapter 4). Amsterdam: Elsevier Science.

Ellison, G., & Ellison, S. F. (2018). *Match quality, search, and the Internet market for used books* (No. w24197). National Bureau of Economic Research.

Erev, I., & Roth, A. E. (1998). Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *American Economic Review, 88,* 848–881.

Ezrachi, A., & Stucke, M. E. (2015). Artificial intelligence and collusion: When computers inhibit competition. Oxford Legal Studies Research Paper No. 18/2015, University of Tennessee Legal Studies Research Paper No. 267.

Fudenberg, D., & Levine, D. K. (2016). Whither game theory? Towards a theory of learning in games. *The Journal of Economic Perspectives, 30*(4), 151–169.

Harrington, J. E. (2017). *Developing competition law for collusion by autonomous agents*. Working paper, The Wharton School, University of Pennsylvania.

Hu, J., & Wellman, M. P. (2003). Nash Q-learning for general-sum stochastic games. *Journal of machine learning research, 4,* 1039–1069.

Lerer, A., & Peysakhovich, A. (2018). Maintaining cooperation in complex social dilemmas using deep reinforcement learning. arXiv preprint arXiv:1707.01068.

Mehra, S. K. (2016). Antitrust and the robo-seller: Competition in the time of algorithms. *Minnesota Law Review*, *100*, 1323–75.

Milgrom, P. R., & Roberts, D. J. (1990). Rationalizability, learning, and equilibrium in games with strategic complementarities. *Econometrica*, *58*, 1255–1277.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., et al. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529–533. https://doi.org/10.1038/nature14236. **(PMID: 25719670)**.

Roth, A. E., & Erev, I. (1995). Learning in extensive-form games: Experimental data and simple dynamic models in the intermediate term. *Games and Economic Behavior, 8,* 164–212.

Salcedo, B. (2015). *Pricing algorithms and tacit collusion*. Working paper Pennsylvania State University.

Sannikov, Y., & Skrzypacz, A. (2007). Impossibility of collusion under imperfect monitoring with flexible production. *American Economic Review*, *97*, 1794–1823.

Sarin, R., & Vahid, F. (2001). Predicting how people play games: A simple dynamic model of choice. *Games and Economic Behavior, 34,* 104–122.

Sukhbaatar, S., Szlam, A., & Fergus, R. (2016). Learning multiagent communication with backpropagation. arXiv, preprint arXiv:1605.07736.

Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. Cambridge: MIT press.

Tampuu, A., Matiisen, T., Kodelja, D., Kuzovkin, I., Korjus, K., Aru, J., et al. (2017). Multiagent cooperation and competition with deep reinforcement learning. *PLoS ONE, 12*(4), e0172395.

Tesauro, G., & Kephart, J. O. (2002). Pricing in agent economics using multi-agent Q-learning. *Autonomous Agents and Multi-Agent Systems, 5,* 289–304.
United States v. David Topkins. (2015). Plea agreement. https://www.justice.gov/atr/casedocument/file/513586/download
Waltman, L., & Kaymak, U. (2008). Q-learning agents in a Cournot oligopoly model. *Journal of Economic Dynamics and Control, 32*(10), 3275–3293.
Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine Learning, 8*(279), 292.
Xie, M., & Chen, J. (2004). Studies on horizontal competition among homogeneous retailers through agent-based simulations. *Journal of Systems Science and Systems Engineering, 13,* 490–505.