Learning in Network Contexts: Experimental Results from Simulations

Amy Greenwald [1,2]
Department of Computer Science, Brown University
Box 1910, Providence, RI 02912.
amygreen@cs.brown.edu

Eric J. Friedman
Department of Economics, Rutgers University
New Brunswick, NJ 08903
friedman@econ.rutgers.edu

Scott Shenker [3]
Xerox Palo Alto Research Center
3333 Coyote Hill Road, Palo Alto, CA 94304
shenker@parc.xerox.com

December 18, 2000

[1]This work was partially supported by an American Association of University Women Dissertation Fellowship.

[2]The authors would like to thank Dean Foster for extensive discussions and insightful suggestions.

[3]Present address: International Computer Science Institute, 1947 Center Street, Suite 600, Berkeley, CA 94704.

Running Head: Learning in Network Contexts

Abstract

This paper describes the results of simulation experiments performed on a suite of learning algorithms. We focus on games in *network contexts*. These are contexts in which (1) agents have very limited information about the game; (2) play can be extremely asynchronous. There are many proposed learning algorithms in the literature. We choose a small sampling of such algorithms and use numerical simulation to explore the nature of asymptotic play. In particular, we explore the extent to which the asymptotic play depends on three factors, namely: limited information, asynchronous play, and the degree of responsiveness of the learning algorithm.

JEL Subject Classification: C63, C72

Send proofs to:
Amy Greenwald
Computer Science Department
Brown University, Box 1910
Providence, RI 02912

List of symbols:

@	at sign
α	alpha
β	beta
γ	gamma
ϵ	epsilon
κ	kappa
λ	lambda
μ	mu
π	pi
ρ	rho
ϕ	phi
$\mathbf{I}$	boldface indicator function
I	boldface roman numeral 1
II	boldface roman numeral 2
III	boldface roman numeral 3
IV	boldface roman numeral 4
$\subseteq$	subset
$\in$	member
∞	infinity
$\rightarrow$	right arrow
[	left square bracket
]	right square bracket
(	left parenthesis
)	right parenthesis
$\{$	left set bracket
$\}$	right set bracket

List of symbols: (cont.)

%	percentage
$+$	plus
$-$	minus
$\times$	times
$*$	asterisk
$/$	divide
$'$	prime
$=$	equal
$<$	less than
$>$	greater than
$\leq$	less than or equal
$\geq$	greater than or equal
$\neq$	not equal
$\pm$	plus or minus
$\sqrt{}$	square root
$\sum$	summation
$\hat{}$	hat
$\tilde{}$	tilde
	over line
$\mathcal{N}$	calligraphy N
$\mathbb{R}$	real numbers

1 Introduction

While much of classical game theory is based on the assumption of common knowledge, there are many contexts in which this assumption does not apply. Correspondingly, in recent years there has been increased interest in the process by which a set of initially naive agents *learn* through repeated play of a game. The central question concerns the nature of asymptotic play; what set of strategies do the agents learn to play in the long-time limit? (The recent review by Fudenberg and Levine [22] provides an overview of the literature.)

In this paper we focus our attention on learning that occurs in what we call a *network context*. In network contexts, agents interact through the common use of a resource, such as a communication link or a shared database, which is accessed over a network. The interactions of Internet congestion control algorithms where agents share network bandwidth, as described in [44], is perhaps the most studied example of a repeated game in a network context. As the Internet continues to grow, and more resources are shared by remote users, we expect the network context to become increasingly common. As discussed in [17], the network context differs from the traditional game-theoretic context in four important ways.

I. First, agents have very limited *a priori* information. In general, agents are not aware of the underlying characteristics of the shared resource. In other words, they do not know the payoff structure of the game; they know neither their own payoff function, nor that of the other players. In addition, agents are not explicitly aware of the existence of other players. While agents are aware that there may be other agents simultaneously using the shared resource, they do not have any way of directly observing their presence and, consequently, they know nothing about the number or the characteristics of their opponents. In particular, this implies that players *cannot* observe the actions of other players, a standard assumption in many classical models of learning in economics.

II. Second, the payoff function and the agent population are subject to change over time. Shared resources like network links and computer servers periodically crash, and often experience other unpredictable changes in their capabilities, such as upgrades or route changes. In addition, users of these shared resources come and go quite frequently. Thus, when an agent detects a change in his payoff while keeping his own strategy fixed, the agent cannot tell whether this change is due to changing strategies of the other players, changes in the players themselves, or variations in the characteristics of the shared resource.

III. Third, in many, but not all, cases, learning is actually carried out by a computer algorithm, not by a human user. For instance, congestion control algorithms (*e.g.*, TCP) embedded in a computer's operating system control the sharing of a network link. Similarly, automated algorithms can control the retry behavior for query submission to a database. Thus, the learning that takes place in these contexts is explicitly laid out in the form of a well-defined algorithm. Moreover, these algorithms are intended to be quite general in nature, and do not depend on the detailed specifics of a particular situation. In particular, this means that Bayesian learning algorithms are inappropriate, because the initial beliefs depend on the specific context. In any event, the complexity of prior probabilities is such that it is not possible to use Bayesian updating in any realistic setting.

IV. Fourth, in network contexts, games can be played in an asynchronous fashion. There need not be any notion of definable "rounds of play"; users can update their strategies at any time. Moreover, the rates at which agents update their strategies can vary widely, although in general these rates are determined by circumstances and are usually not a strategic variable. For example, automated agents can learn at different rates, depending on their processor speeds and the nature of their learning algorithms. In addition, due to the geographic dispersion of users of the Internet, there can be varying communication delays to a shared resource, which can lead to updating rates that differ by several orders of magnitude.[4] Thus, asynchrony does not arise from Stackelbergian-type strategic manipulation, but merely from properties of communication and computation. Agents closer to the shared resource, or those who have faster processors or smarter algorithms, have the potential to learn more rapidly and effectively.

We focus on contexts that have these four properties: low information content, non-stationary payoffs, automated learning, and asynchrony. We are interested in what happens when a set of automated agents play a game repeatedly in such a context, and we investigate this behavior empirically. We consider a small sampling of learning algorithms, some of which have been well-studied in the literature; for each algorithm we numerically simulate a set of agents using that algorithm and we observe the set of strategies played in the long-time regime. Our simulation experiments can be seen as a natural counterpart to human economic experiments; in particular, Chen [7] investigates some issues closely related to those considered here using human subjects rather than automated learning algorithms.[5]

We concentrate on the extent to which the asymptotic play depends on the amount of information available to the agents, the degree of responsiveness of the learning algorithm, and the level of asynchrony of play. Of particular interest is the extent to which the asymptotic play is contained in the various solution concepts such as Nash equilibria, the set of serially undominated strategies (D^∞), and the less traditional concepts of serially unoverwhelmed strategies (O^∞) and serially Stackelberg-undominated strategies (S^∞) which are discussed below. Our findings suggest that the asymptotic play of games in network contexts can be quite different from that of standard contexts,

[4]As discussed in [17], standard control theoretic results imply that the frequency at which strategies are updated should not be greater than the inverse of the round-trip communication delay to the shared resource; otherwise, instability may result.

[5]Although our present focus is solely on automated agents, experimental evidence (see [7], [9], and [37]) suggests that our results are of relevance in describing human/human and human/machine interactions as well.

where play is typically contained in the serially undominated strategy set.

This paper has 6 sections. Section 2 presents the learning algorithms used in this study, and discusses the relevant solution concepts. Section 3 contains the simulation results for the sample games considered: several simple two-player games, a class of externality games, and the congestion game. In Section 4, we compare these results with the results of simulations in non-network contexts. Section 5 describes related work, and we conclude in Section 6 with a brief discussion of our results. Finally, Appendix A describes the suite of learning algorithms that we simulate in some detail.

2 Background

2.1 Learning Algorithms

The literature is replete with learning algorithms, but but not all of them are applicable in network contexts. Because knowledge of the payoff structure and the other agents is extremely limited, games in a network context are, from a single agent's perspective, most naturally modeled as *games against nature* in which each strategy has some random (and possibly time-varying) payoff about which the agent has no *a priori* knowledge. Consequently, in contrast with belief-based approaches to learning (*e.g.*, Bayesian updating) adopted in much of the literature, learning algorithms for network contexts typically utilize simple updating schemes that do not rely on any detailed assumptions about the structure of the game. Instead, these algorithms employ "trial-and-error" experimentation in an attempt to identify optimal strategies.

The learning algorithms which we simulate are distinguished first of all by their varying degrees of experimentation; we will, for convenience, denote this level of experimentation by the parameter $\epsilon \in [0, 1)$. In static environments, where the payoff structure and the set and characteristics of the other agents is fixed, it may be reasonable to decrease the level of experimentation as time progresses, with experimentation ceasing in the infinite-time limit (*i.e.*, $\epsilon \to 0$ as $t \to \infty$).[6] Many learning algorithms proposed in the literature have this property. In network contexts, however, the environment is not static; the underlying payoff structure, and the population of agents, are subject to change at any time without explicit notification. As a result, agents should be prepared to respond to changing conditions at all times, and should do so in a bounded amount of time. This requires that a non-zero level of experimentation be maintained in the long-time limit, and that future play be more heavily influenced by payoffs obtained in the recent, rather than the distant, past. This second point can be achieved via a parameter $\gamma \in (0, 1]$ which dictates the rate (and typically inverse accuracy) of learning. We call the ability to respond to changes in the environment in bounded time *responsiveness*, and posit that this property is fundamental to learning in network contexts. As we shall see, responsiveness has important implications for the resulting asymptotic play.

The learning algorithms we discuss also differ in the particular criteria that they are designed to satisfy. Perhaps the simplest criterion is that, when playing a static game-against-nature, the algorithm rapidly learns to play (with high probability) the strategy with the highest average payoff.[7] When combined with responsiveness, and a certain monotonicity property, this leads to the class of so-called *reasonable* learning algorithms introduced in [17]. One example of such an algorithm is the *stage* learning algorithm. Stage learners partition the game into *stages*, which consist of $1/\gamma$ rounds of a game. At each round of play, a stage learner chooses its strategy at random based on the probabilities, or weights, it has assigned to each of its strategies.

[6] This is apparent in decision problems such as classic bandit problems [23, 31]. It is less transparent, however, in games with multiple players and strategies.

[7] The formal definition of probabilistic converge in finite time is described in [17]. In this paper we do not formally define convergence, but take a more pragmatic approach which is appropriate for simulations. That is, we say that play has converged when the numerical properties are unchanged by additional iterations as evidenced by simulations.

These weights are updated upon termination of each stage, with weight $1 - \epsilon$ assigned to the pure strategy that obtained the highest average payoffs during the previous stage, and weight $\epsilon/(n-1)$ assigned to all other strategies. Another example of a reasonable learning algorithm, so-called *responsive learning automata* introduced in [16], is a responsive version of simple learning automata (see, for example, Narendra and Thathachar [38]). This algorithm updates weights after every round of play using quite a different method. Another reasonable learning algorithm (for certain choices of parameters) is defined by Roth and Erev [9], and has been used to model human behavior in game-theoretic experiments.

A second criterion, which is a worst-case measure of performance, involves the concept of *regret*. Intuitively, a sequence of plays is optimal if there is no regret for playing a given strategy sequence rather than playing another possible sequence of strategies. Regret comes in two forms: external and internal. A sequence of plays is said to exhibit no external regret if the difference between the cumulative payoffs that are achieved by the learner and those that could be achieved by any other pure strategy is insignificant. The no internal regret optimality criterion is a refinement of the no external regret criterion where the difference between the performance of a learner's strategies and any *remapped* sequence of those strategies is insignificant. By remapped we mean that there is a mapping f of the strategy space into itself such that for every occurrence of a given strategy s in the original sequence the mapped strategy $f(s)$ appears in the remapped sequence of strategies. The learning procedures described in Foster and Vohra [13] and Hart and Mas-Colell [27] satisfy the property of no internal regret. Early no external regret algorithms were discovered by Blackwell [4], Hannan [26], Banos [2], and Megiddo [35]; recently, no external regret algorithms appeared in Cover [8], Freund and Schapire [14], and Auer, Cesa-Bianchi, Freund and Schapire [1].

We investigate six learning algorithms: the reasonable learners discussed above (see [16, 17, 9]), two based on external regret (see [1, 11]), and one based on internal regret (see [27]). Some of these algorithms were initially proposed for quite different settings, in which responsiveness is not necessary and the information level is significantly higher (*e.g.*, agents know their own payoff function). We have extended these learning algorithms for use in network contexts.[8] We call the versions designed for low-information settings *naive*, and those designed for higher information contexts *informed*. We also consider both *responsive* and *non-responsive* variants. Lastly, each agent has a time-scale parameter A that determines the rate at which it updates its strategies. A player updates its strategy (*i.e.*, runs its learning algorithm) only every A rounds, and treats the average payoff during those A rounds as its actual payoff. We are interested in how the asymptotic play depends on whether agents are responsive, whether they are informed, and on the degree of asynchrony (differences in the A values) among the agents.

2.2 Solution Concepts

To describe the asymptotic play, we introduce several solution concepts. We begin with some notation. We restrict our attention to finite games. Let $\mathcal{N} = \{1, \ldots, N\}$ be a finite set of *players*, where $N \in \mathcal{N}$ is the number of players. The finite set of strategies available to player $i \in \mathcal{N}$ is denoted by S_i, with element $s_i \in S_i$. The set of pure strategy profiles is the Cartesian product $S = \prod_i S_i$. In addition, let $S_{-i} = \prod_{j \neq i} S_j$ with element $s_{-i} \in S_{-i}$, and write $s = (s_i, s_{-i}) \in S$. The payoff function $\pi_i : S \to \mathbb{R}$ for player i is a real-valued function on S.

Recall that strategy $s_i \in S_i$ is strictly dominated for player i if there exists some strategy $s_i^* \in S_i$ such that $\pi_i(s_i, s_{-i}) < \pi_i(s_i^*, s_{-i})$ for all $s_{-i} \in S_{-i}$. Let D^∞ denote the serially undominated strategy set: *i.e.*, the set of strategies that remains after the iterated elimination of strictly dominated strategies. Milgrom and Roberts [36] show that the asymptotic play of a set of *adaptive*

[8] Detailed descriptions of these algorithms, in both their original and modified forms, are contained in Appendix A.

learners – learners that eventually learn to play only undominated strategies – eventually lies within D^∞. In addition, it is shown in [16] that certain responsive learners playing synchronously also converge to D^∞. The set D^∞ is widely considered to be an upper bound in terms of solution concepts; that is, it is commonly held that the appropriate solution concept that arises via learning through repeated play is a subset of the serially undominated set.[9] This may indeed be true in standard game-theoretic contexts.

In [16, 17], however, it is shown that in network contexts, where there is potentially asynchrony and responsive learning, play can asymptotically remain outside the serially undominated set. A more appropriate solution concept for such settings is based on the concept of *overwhelmed* strategies. We say that a strategy $s_i \in S_i$ is strictly overwhelmed if there exists some strategy $s_i^* \in S_i$ such that $\pi_i(s_i, s_{-i}) < \pi_i(s_i^*, s'_{-i})$ for all $s_{-i}, s'_{-i} \in S_{-i}$. Let O^∞ denote the set of strategies that remains after the iterated elimination of strictly overwhelmed strategies. It is shown in [17] that the asymptotic play of a set of reasonable learners lies within O^∞, regardless of the level of asynchrony. However, it is conjectured that O^∞ is not a precise solution concept, only an upper bound.

A refinement of O^∞ called S^∞ is introduced and formally defined in [17]. [10] Because it is rather cumbersome, we do not present the precise definition of S^∞. Intuitively, the set S^∞ extends the set D^∞ by allowing for the possibility that play is asynchronous, rather than synchronous as is standard in repeated game theory. More formally, the computation of S^∞ is as follows: pick a specific (non-strict) ordering of the players and construct the extensive form game arising from players moving according to that order; now compute the set of actions which survive the iterated deletion of strictly dominated

[9] Note that this also holds for one-shot games with common knowledge, as the set D^∞ contains all the rationalizable strategies [3, 39].

[10] In [17], a class of "Stackelberg" solution concepts is built from various primitives (such as undominated strategies or correlated equilibria); however for the games studied in this paper most of the "Stackelberg" solution concepts coincide.

strategies in the new game; finally, take the union of these actions for all orderings. Since the ordering is non-strict, asynchronicity defined in this way incorporates synchronicity, from which it follows that $D^\infty \subseteq S^\infty \subseteq O^\infty$.

Another result of interest, due to Foster and Vohra [13], is that a set of no internal regret learners converges to a correlated equilibrium. Note that the support of a set of correlated equilibria is a subset of D^∞; in other words, correlated equilibria do not assign positive probabilities to strategies outside D^∞, but neither do they necessarily converge to Nash equilibria. In contrast, the asymptotic play of a set of no external regret learners need not remain inside D^∞ [11] [24].

In the remainder of this paper, we present the result of simulations of the six learning algorithms on various games. We ask, in particular, whether the asymptotic play converges within the sets D^∞, S^∞, or O^∞. Recall that the term convergence is used informally, both because of experimentation, which precludes true convergence, and our interest in achieving results in finite time. We are interested in which of these concepts, if any, represents an appropriate solution concept for games in network contexts.

3 Simulations in Network Contexts

We consider three sets of games: simple games (two players, few strategies), the congestion game (two players, many strategies), and an externality game (many players, two strategies). The simulations were conducted with varying degrees of asynchrony, ranging from synchronous play to extreme asynchrony with one player acting as the leader (*i.e.*, we vary the value of A from 1 to 10,000 for the leading player and we set $A = 1$ for all other players). The degree of responsiveness is determined by parameters ϵ and γ; for each game, we describe the particular parameter settings.

[11] Note that this remark pertains only to the no external regret criterion, but says nothing about the convergence properties of specific algorithms which are defined to satisfy this criterion, such as those considered in this study.

3.1 Simple Two-Player Games

This subsection presents the results of simulations of four simple two-player games with either two or three strategies per player. The row player is taken to be the leader. The responsive parameter γ was set to .01 for all algorithms, while the degree of experimentation ϵ was set to .025 for the reasonable learning algorithms and .05 for the no regret algorithms.[12] In addition, the no regret algorithms depend on tuning parameters; for the mixing method, $\alpha \sim 1.5$, for multiplicative updating, $\beta = 1$, and for the no internal regret algorithm, $\kappa = 2$. [13] Unless otherwise stated, the simulations described in this paper were run for 10^8 iterations; although play generally converged in far fewer iterations, this rather lengthy simulation time eliminated the transient effects of initial conditions in the final long-run empirical frequency calculations. Initially, all strategies were assigned equal weights.

3.1.1 Game D

The game depicted in Fig. 1 is referred to herein as Game D since it is D-solvable, but it is not S-solvable or O-solvable: *i.e.*, $D^\infty \neq S^\infty = O^\infty$. More specifically, the set D^∞ is a singleton that contains only the strategy profile (T, L), which is the unique Nash equilibrium. On the other hand, $S^\infty = O^\infty = \{T, B\} \times \{L, R\}$. Note that (B, R) is a Stackelberg equilibrium in which the row player is the leader.

The graph depicted in Fig. 2 describes the overall results of simulations of Game D, assuming responsive learning in a naive setting. In particular, Fig. 2

[12] The choice of parameter values reflects the usual trade-off between exploration and exploitation. Increasing the rate of responsiveness γ and the rate of experimentation ϵ leads to increased error in stationary environments, but increased accuracy in non-stationary environments. In our experience, the results are fairly robust to small changes in parameter settings, although we have not formally measured this robustness.

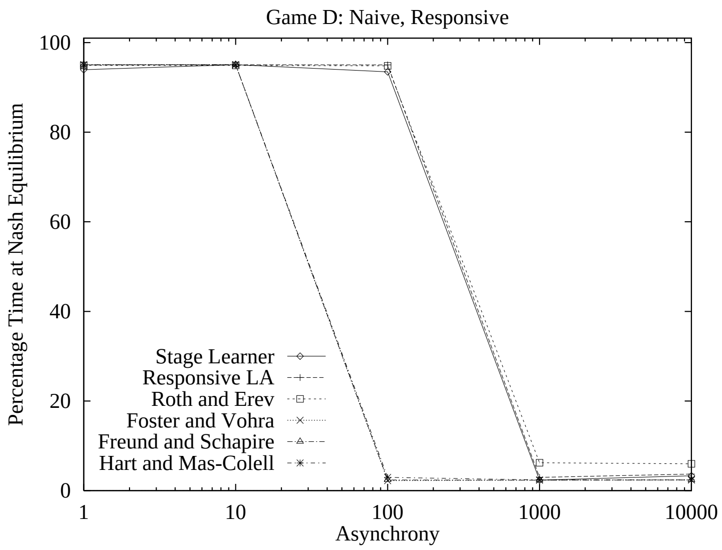
[13] These parameters represent the learning rates in the no regret algorithms. Slower learning rates correspond to higher degrees of accuracy in stationary environments; in non-stationary environments, faster learning rates induce more responsive behavior.

1 \ 2	L	R
T	1,2	3,0
B	0,0	2,1

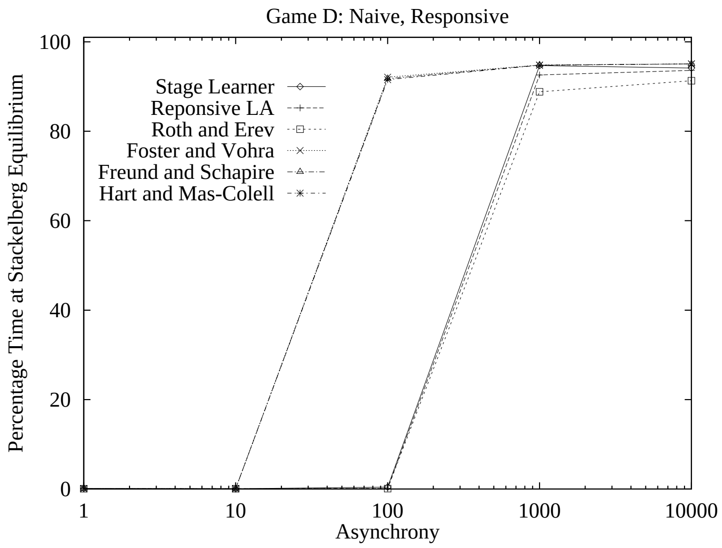
Figure 1: Game D

(a) plots the percentage of time in which the Nash equilibrium solution arises as the degree of asynchrony varies. Asynchrony of 100, for example, implies that the column player is learning 100 times as fast as the column player; thus, the row player is viewed as the leader and the column player the follower. Note that when play is synchronous, all the algorithms converge to the unique Nash solution. In the presence of sufficient asynchrony, however, play does not converge to the Nash solution for any of the algorithms studied. Instead, play converges to the Stackelberg equilibrium, as depicted in Fig. 2 (b). These results demonstrate that D^{∞} does not always contain the asymptotic play. Note that these results are robust; in particular, the results are unchanged even when the game is studied with "noisy" payoffs $\hat{\pi}_i$, where $\hat{\pi}_i = \pi_i \pm \delta$, for small $\delta > 0$.

The transition from Nash equilibrium to Stackelberg equilibrium depicted in Fig. 2 is rather abrupt. This observation prompted us to conduct further simulations at a series of intermediate values to more precisely determine the impact of asynchrony. For the reasonable learning algorithms, the transition between equilibria takes place when A falls between 100 and 1000; for the no regret algorithms, this transition takes place when A lies between 10 and 100. Fig. 3 depicts the details of these transitions in the respective

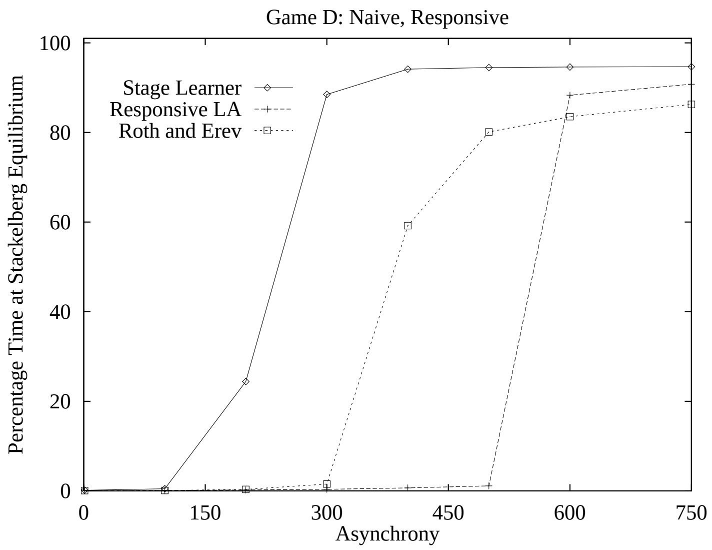


(a) Nash equilibrium

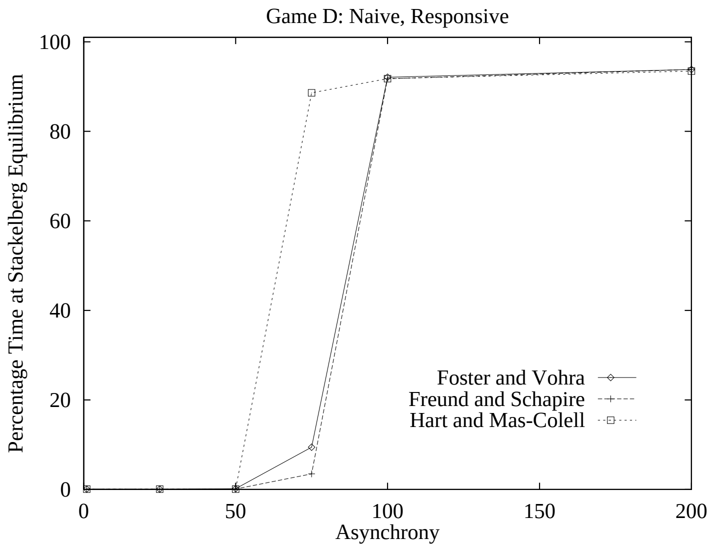


(b) Stackelberg equilibrium

Figure 2: *Convergence to Equilibria in Game D.* (a) plots the percentage of time in which Nash equilibrium arises as the degree of asynchrony varies, while (b) plots the percentage of time in which Stackelberg equilibrium arises.

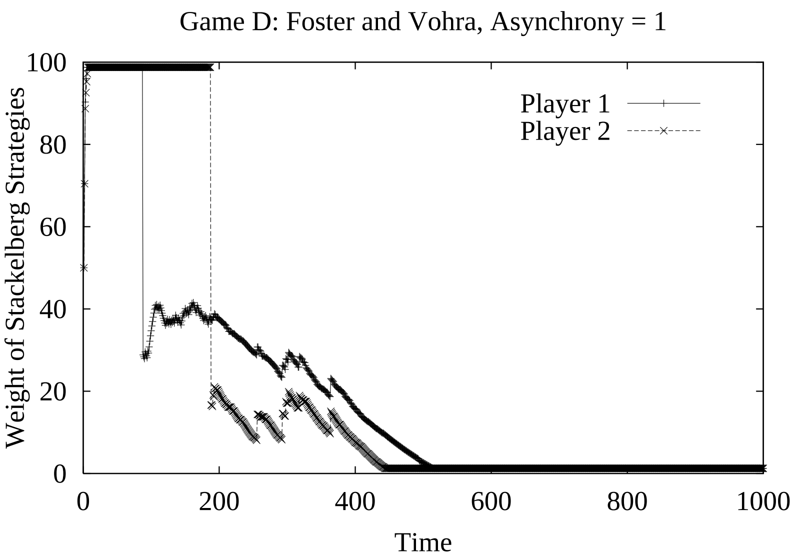


(a) Reasonable Learning Algorithms

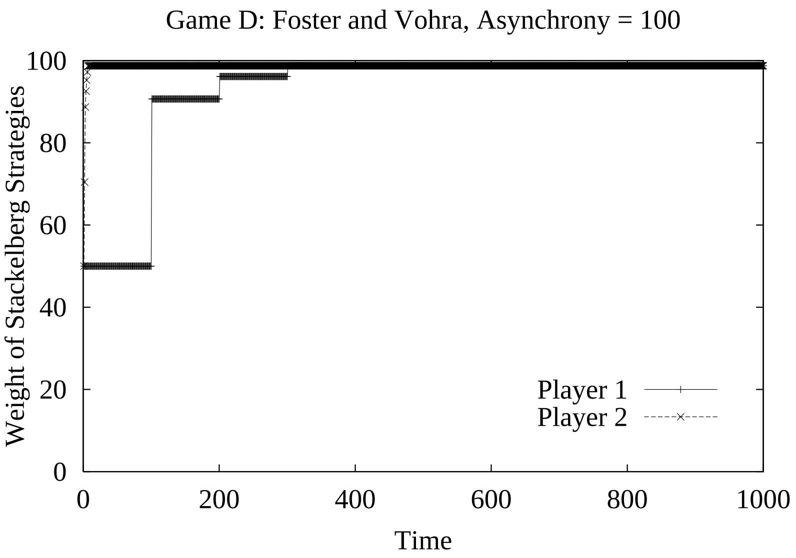


(b) No Regret Learning Algorithms

Figure 3: *Detail of Convergence to Stackelberg Equilibrium in Game D.* (a) Reasonable Learning Algorithms. (b) No Regret Learning Algorithms.



(a) Nash Equilibrium ($A = 1$)



(b) Stackelberg Equilibrium ($A = 100$)

Figure 4: *Convergence to Equilibria in Game D for Foster's and Vohra's algorithm.* (a) plots the weights of the Stackelberg equilibrium strategies over time when $A = 1$; notice that play converges to Nash equilibrium. (b) plots these weights when $A = 100$; notice that play converges to Stackelberg equilibrium.

ranges of asynchrony for the two sets of algorithms. Only reasonable learning algorithms ever clearly exhibit out-of-equilibrium behavior; the no regret algorithms transition directly from one equilibrium to the other.

Recall that (B, R) is the Stackelberg equilibrium in Game D. Fig. 4 plots the changing weights over time of strategy B for player 1 and strategy R for player 2 for the no external regret algorithm due to Foster and Vohra. The individual plays are also plotted; marks at 100 signify play of Stackelberg strategies, and marks at 0 signify play of Nash strategies. For comparison purposes, the synchronous case, where play converges to Nash equilibrium, as well as the asynchronous case with $A = 100$, where play converges to Stackelberg equilibrium, are depicted. Play converges in the former case after roughly 500 iterations, while it converges in the latter case after only about 300 iterations (for the given choices of parameters ϵ and γ). In Fig 4(b), the leader (player 1) slowly learns to play the Stackelberg solution, and because the follower (player 2) is responsive, his weights follow the leader's plays. This behavior is representative of all the learning algorithms considered.

3.1.2 Game O

The next game that is studied in this section is depicted in Fig. 5, and is referred to as Game O, since it is O-solvable. In this game, $\{(T, L)\}$ is the unique Nash equilibrium and $\{(T, L)\} = D^\infty = S^\infty = O^\infty$. Simulations of all the algorithms, for levels of asynchrony ranging from 1 to 10,000, show that Nash equilibrium is played over 95% of the time. In particular, play does not diverge from the Nash equilibrium solution in this O-solvable game, as it did in Game D, regardless of the degree of asynchrony. It has been established that, for reasonable learners, O^∞ is an upper bound on the solution concept. The same result holds for the other algorithms considered, but this is far short of a proof that the O^∞ solution concept applies to them as well. The next game addresses the question of whether the O^∞ solution concept might in fact be too large a set.

1 \ 2	L	R
T	2,2	3,1
B	1,3	0,0

Figure 5: Game O

3.1.3 Prisoners' Dilemma

This section presents the results of simulations of the repeated Prisoners' Dilemma (see Fig. 6). In this game, $\{(D, D)\}$ is the unique Nash (and Stackelberg) equilibrium, and $\{(D, D)\} = D^\infty = S^\infty \neq O^\infty$, since O^∞ is the entire game. The Prisoner's Dilemma provides a simple test of the conjecture that the outcome of responsive learning in network contexts is described by the S^∞ solution concept, rather than the larger solution set O^∞.

Simulations of all the algorithms, for levels of asynchrony ranging from 1 to 10,000, show that in this game the Nash equilibrium is played over 95% of the time. Since play does not diverge significantly from the Nash (and Stackelberg) equilibrium, the asymptotic play is not spread throughout O^∞; instead, it is confined to S^∞.

3.1.4 Game S

The last simple two-player game that is studied is a game in which the players have three strategies. The game is depicted in Fig. 7, and is referred to as Game S. In Game S, $D^\infty = \{T, L\}$; $S^\infty = \{T, L\} \times \{B, R\}$; and O^∞ is the entire game; thus, $D^\infty \neq S^\infty \neq O^\infty$. The results of simulations of Game

1 \ 2	C	D
C	2,2	0,3
D	3,0	1,1

Figure 6: Prisoners' Dilemma

1 \ 2	L	C	R
T	1,2	3,0	2,0
M	0,0	2,1	0,0
B	0,0	2,0	1,1

Figure 7: Game S

S resemble the results of simulations of Game D. [14] Fig. 8 shows that the learning algorithms do not converge to the Nash equilibrium solution of this game when there is asynchrony. Instead, play converges to the Stackelberg equilibrium, as in Game D. This game provides a second test of the conjecture that the outcome of responsive learning in network settings is a strict subset of the O^∞ solution.

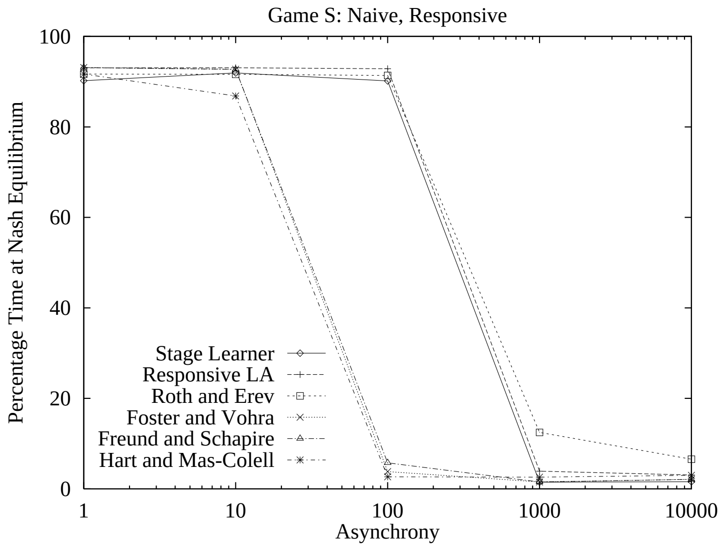
3.2 Externality Games

To test whether similar results apply to games with more than two players, we experiment with externality games. An externality game, as defined in [18], is one in which each agent can choose either to participate or not to participate (in some joint venture) and where the payoffs obtained by a given player depend only on whether that player participates and on the total number of participating players. In this section, we study a related class of games which are D-solvable, and moreover, for certain choices of the parameters, the games in this class are S-solvable and O-solvable as well.

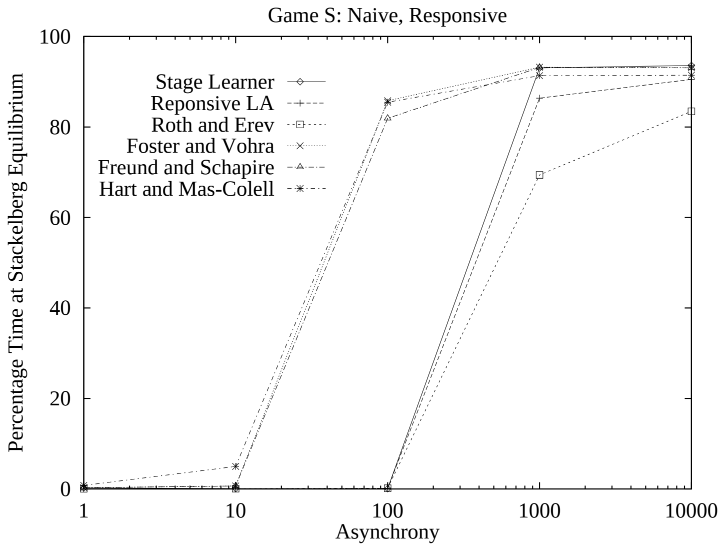
The class of games which we consider (class EG) is a discretization of the non-atomic games discussed in [15]. The set of players $\mathcal{N} = \{0, \ldots, N-1\}$, with $i \in \mathcal{N}$. The players have two possible strategies, namely 0 and 1, where 1 corresponds to participation, and 0 corresponds to non-participation. The number of participants, therefore, is given by $\lambda(s) = \sum_{i \in \mathcal{N}} s_i$, where s_i denotes the strategic choice of player i. Payoffs are determined as follows. The value to player i of participation is $v_i \in \mathbb{R}$, and the cost of participation is $C_i(\lambda)$, where C_i is a nondecreasing function of the externality. Thus, if player i participates, then $s_i = 1$ and $\pi_i(1, s_{-i}) = v_i - C_i(\lambda(s))$. Otherwise, if player i does not participate, then $s_i = 0$, and $\pi_i(s) = \phi\pi_i(1, s_{-i})$, for $\phi \in [0, 1)$. Intuitively, ϕ measures the extent to which players can opt out.

Note that the parameter ϕ does not affect the standard strategic elements of a given game in this class, such as best-replies or dominated strategies. In

[14] Note that in these simulations, the reasonable learning algorithms utilized $\epsilon = .1667$.



(a) Nash Equilibrium



(b) Stackelberg Equilibrium

Figure 8: *Convergence to Equilibria in Game S.* (a) plots the percentage of time in which Nash equilibrium arises as the degree of asynchrony varies, while (b) plots the percentage of time in which Stackelberg equilibrium arises.

particular, if the game is D-solvable for $\phi = 0$ then it is D-solvable for all $\phi \in [0, 1)$. Similarly, varying ϕ does not change the set of Nash equilibria. Moreover, it is straightforward to show that when $\phi = 0$, if the game is D-solvable, then it must also be O-solvable (and therefore, also S-solvable). In contrast, for ϕ sufficiently close to 1, the game is not S-solvable (and therefore, not O-solvable). Thus, by varying ϕ we can create a class of games which are D-solvable but not necessarily S-solvable and not O-solvable.[15]

In our simulations, we consider eight players (*i.e.*, $\mathcal{N} = \{0, \ldots, 7\}$), and we set $v_i = i$ and $C_i(\lambda) = \lambda/\mu$, for $\mu \in \mathbb{R}$. In the first set of simulations, we choose $\mu = 1.9$; we call this Game $\mathrm{EG}_{1.9}$. This game is D-solvable and therefore has a unique Nash equilibrium. Moreover, this implies that for $\phi = 0$, this game must also be O-solvable; however, for ϕ sufficiently close to 1, it is neither S-solvable nor O-solvable (see Appendix B). More specifically, when $\phi > 6/11 = .\overline{54}$, Game $\mathrm{EG}_{1.9}$ has a Stackelberg equilibrium with player 2 as the leader which differs from the Nash equilibrium: the Nash equilibrium for all $\phi \in [0, 1)$ is $s = (0, 0, 0, 1, 1, 1, 1, 1)$, while the Stackelberg equilibrium (with player as the 2 leader and $\phi > 6/11$) is $s = (0, 0, 1, 0, 1, 1, 1, 1)$.

Simulations of Game $\mathrm{EG}_{1.9}$ were conducted using the naive, responsive variants of the no regret learning algorithms.[16] The convergence results in the asynchronous case (for $A = 5,000$) are listed in Table I, for $\epsilon = .02$ and $\gamma = .002$. [17] Simulations of all algorithms show rapid convergence (in the empirical sense described earlier) to the Nash equilibrium (NE) for all values of β in the synchronous case, and to the Stackelberg equilibrium (SE) when $\beta = .6$ and $\beta = .9$ in the asynchronous case. In particular, in Game $\mathrm{EG}_{1.9}$, for certain choices of the parameters, the asymptotic play is not contained in the set D^∞.

[15]Proofs of these claims appear in Appendix B for the class of games considered.

[16]The payoffs were translated by N/μ in simulations of responsive learning automata and the algorithm due to Roth and Erev in order to avoid negative payoffs.

[17]In the algorithm due to Foster and Vohra, $\alpha = 1,000$; in the algorithm due to Freund and Schapire, $\beta = 1$; finally, in the algorithm due to Hart and Mas-Colell, $\kappa = 5$.

Algorithm	$\beta = 0$	$\beta = .5$	$\beta = .6$	$\beta = .9$
Foster and Vohra	NE	NE	SE	SE
Freund and Schapire	NE	NE	SE	SE
Hart and Mas-Colell	NE	NE	SE	SE

Table I: Game $\text{EG}_{1.9}$ with $\gamma = .002, A = 5,000$.

Another game which we simulated in the class EG is Game $\text{EG}_{2.1}$, which is identical to Game $\text{EG}_{1.9}$, except that $\mu = 2.1$. Like Game $\text{EG}_{1.9}$ this game is D-solvable. This implies that for $\phi = 0$, this game must also be O-solvable; however, for ϕ sufficiently close to 1, this game is not O-solvable (see Appendix B). Lastly, unlike Game $\text{EG}_{1.9}$, Game $\text{EG}_{2.1}$, is S-solvable (see Appendix B). This game was simulated assuming the same parameter values as the previous set of simulations, as well as some additional choices for γ. Selected results of these simulations appear in Table II. Notice that regardless of the values of γ and ϕ, and even in the presence of extreme asynchrony, play converges to Nash equilibrium. Thus, as $D^\infty = S^\infty \neq O^\infty$, this game provides further evidence for the conjecture that the asymptotic play of learning in network contexts is contained in S^∞.

Algorithm	$\beta = 0$	$\beta = .5$	$\beta = .6$	$\beta = .9$
Foster and Vohra	NE	NE	NE	NE
Freund and Schapire	NE	NE	NE	NE
Hart and Mas-Colell	NE	NE	NE	NE

Table II: Game $\text{EG}_{2.1}$ with $\gamma \in \{.01, .005, .002, .001\}, A = 10,000$.

3.3 The Congestion Game

So far we have considered games with rather small strategy spaces. In this section, we experiment with a larger strategy space, using an example that arises in computer networks. Consider several agents simultaneously sharing a network link, where each agent controls the rate at which she is transmitting data. If the sum of the transmission rates is greater than the total link capacity, then the link becomes congested and the agents' packets experience high delays and high loss rates. The transmission rates are controlled by each agent's congestion control algorithm, which vary the rates in response to the level of congestion detected.

One can model the interaction of congestion control algorithms as a cost-sharing game where the cost to be shared is the congestion experienced. That is, we can model this as a game where the strategies are the transmission rates r_i and the outcomes are the pairs (r_i, c_i), where c_i is the congestion experienced as function of the strategy profile r. The allocations must obey the sum rule $\sum_i r_i = F(\sum_i c_i)$ where F is a constraint function (*i.e.*, the total congestion experienced must be a function of the total load).

Most current Internet routers use a FIFO packet scheduling algorithm, which results in congestion that is proportional to the transmission rate: *i.e.*, $c_i = [r_i / \sum_j r_j] F(\sum_j c_j)$; this is called the average cost pricing (ACP) mechanism. In contrast, the fair queuing packet scheduling algorithm can be modeled as leading to allocations such that c_i is independent of r_j as long as $r_j \geq r_i$ (this condition, plus anonymity, uniquely specifies the allocations). This is called the Serial mechanism (see [44] for a detailed description).

Chen [7] investigates the two-player congestion game with the following properties: (1) linear utilities $U_i(r_i, c_i) = \alpha_i r_i - c_i$, (2) quadratic congestion $F(x) = x^2$, and (3) a discrete strategy space $S_i = \{1, 2, \ldots, 12\}$. For the choice of parameters $\alpha_1 = 16.1$ and $\alpha_2 = 20.1$, the game defined by the ACP mechanism is D-solvable, but it is not S-solvable or O-solvable. The unique Nash equilibrium is $(4, 8)$ and the Stackelberg equilibrium with player

2 leading is $(2, 12)$. In contrast, the game defined by the Serial mechanism is O-solvable, with the unique Nash equilibrium at $(4, 6)$.

We conducted simulations of both the Serial and the ACP mechanism using the naive, responsive variants of the no regret algorithms with degree of experimentation $\epsilon = .02$. [18,19] In our simulations of the Serial mechanism, the learning algorithms concentrate their play around the Nash equilibrium. In the ACP mechanism, when play is synchronous, the asymptotic behavior again centers around the Nash equilibrium. Given sufficient asynchrony (*e.g.*, $A = 5,000$, when $\gamma = .002$), however, play converges to the Stackelberg equilibrium.

3.4 Discussion

Our results thus far indicate that when responsive learning algorithms play repeated games against one another, play can persist outside the set of serially undominated strategies, given sufficient asynchrony. It comes as no surprise that asynchrony can lead to play outside Nash or correlated equilibria. It is known, for example, that Stackelbergian behavior arises in games among "patient players" [19], or when there exists the ability to make commitments Rosenthal [40] or the capacity to establish reputations [49]. In these prior analyses, the Stackelberg leaders are seen as manipulating the system. The asynchrony that we discuss here comes from a quite different source: the underlying speed of communication and computation of the agents.

In our examples, the trajectory of play is always either largely inside the serially undominated set, or with sufficient asynchrony, it converges to the Stackelberg equilibrium. The transition from one behavior to the other is generally quite sudden, although some of the reasonable learning algorithms raise exception (see Fig. 3). Based on work by Foster and Vohra [13], we

[18] In the algorithm due to Foster and Vohra, $\alpha = 5,000$; in the algorithm due to Freund and Schapire, $\beta = 1$; finally, in the algorithm due to Hart and Mas-Colell, $\kappa = 2,000$.

[19] The payoffs were translated to $\mathbb{R}^+$ in simulations of responsive learning automata and the algorithm due to Roth and Erev in order to avoid negative payoffs.

conjecture that more complicated behavior, with probabilities spread over a wider set of strategy profiles, can arise in more complex games, but always remains within the set S^∞.

4 Simulations in Non-network Contexts

Network contexts differ from standard learning contexts in three important ways: asynchronous play, limited information, and responsive learning. In the previous sections we have looked at how varying asynchrony affects the convergence properties of learners. In this section, we briefly consider the remaining two properties of learning in network contexts.

First we augment the information structure by considering contexts in which learners are *informed*, where such learners know the payoffs that would have occurred had they chosen an alternative action. This typically arises when players (i) know the payoff matrix, and (ii) can observe the actions of the other players. Not surprisingly, in this setting play outside D^∞ does not arise. Unfortunately in network contexts, informed learning is not an option; the basic structure of the Internet is such that learning is inherently uninformed.

Our second consideration, namely responsiveness, is not inevitable, but instead reflects a common (and appropriate) design choice on the Internet. We find the behavior of naive and non-responsive learners, in the presence of asynchrony, to be complex and do not claim to understand such asymptotic behavior; for example, we do not know whether play always converges to D^∞. We demonstrate some of this complexity through simulations of the Shapley game, a classic example for which fictitious play does not converge. Irrespective of these complexities, non-responsive learning is not viable in network contexts due to the non-stationarity of the payoff functions.

The results of this section show that the seemingly obvious conjecture that asynchrony alone leads to Stackelberg behavior is not true in general.

In our simulations, this conjecture holds only when we consider naive *and* responsive learning algorithms, as are relevant for network contexts. If the algorithms are informed, or non-responsive, we do not observe Stackelbergian behavior.

4.1 Informed Learning

Recall that the simulations that lead to asymptotic play outside D^∞ utilize the responsive and naive variants of the set of learning algorithms. In our simulations of Game D (see Fig. 1), responsive but informed learning does *not* lead to play outside D^∞, even in the presence of extreme asynchrony, for enhanced versions of reasonable learning that utilize full payoff information and the original no regret algorithms. Specifically, for levels of asynchrony between 1 and 10,000, simulations of all responsive and informed algorithms show that Nash equilibrium is played over 95% of the time.[20]

Intuitively this occurs because the set of informed learning algorithms compares the current payoff with the potential payoffs of the other possible strategies, *assuming that the other agents keep their strategies fixed.* The key to the Stackelberg solution is that the leader evaluates his payoff in light of the probable responses of other agents. Naive learners, when learning at a slow rate, do this implicitly; that is, they only receive their payoffs after the other players respond to their play. The informed learning algorithms which we consider do not take this reaction into account.

If informed and responsive learning algorithms do indeed converge to D^∞ in general, this might be seen as an argument to consider only informed learning algorithms. However, in network contexts this is not an option; the information about other payoffs is not available and so we are forced to use naive learning algorithms. However, agents do have a choice as whether to

[20] The simulations of Game D discussed in this section and the next depend on the same set of parameter values as in Section 3.1; specifically, $\gamma = .01$ for all algorithms, while $\epsilon = .025$ for the reasonable learning algorithms and $\epsilon = .05$ for the no regret algorithms.

use responsive or non-responsive algorithms, a subject to which we now turn.

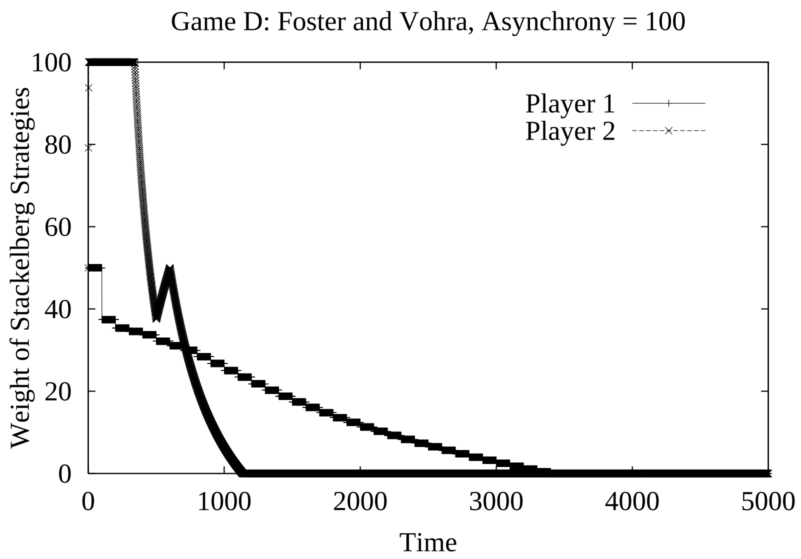
4.2 Non-responsive Learning

We now consider, as a benchmark, non-responsive learners. Simulations of Game D (see Fig. 1) using non-responsive algorithms, for levels of asynchrony between 1 and 10,000, show that the Nash equilibrium is played over 99% of the time for the set of informed algorithms and over 95% of the time for the naive set. Fig. 9 depicts the weight over time of strategy B for player 1 and strategy R for player 2 in simulations of the no external regret learning algorithm due to Foster and Vohra with level of asynchrony 100.

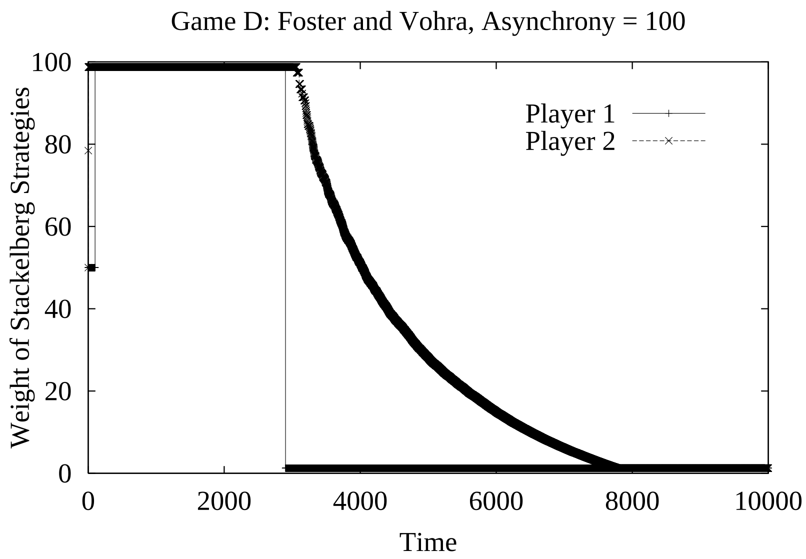
The behavior of informed and non-responsive learners is similar to that of informed and responsive learners, in that they learn to eliminate dominated strategies, yielding convergence to D^∞ (see Fig. 9(a)). This seems reasonable given that the informed, non-responsive algorithms which we simulate are approximately adaptive learners in the sense of Milgrom and Roberts [36], who prove that such adaptive learners converge to D^∞.

In the simulation of naive, non-responsive learners (see Fig. 9(b)), play initiates at the Stackelberg equilibrium, until time almost 3000, when the leader experiments with the Nash equilibrium strategy. Since he is followed shortly thereafter (albeit slowly) by the follower, he never has an incentive to return to the Stackelberg solution. Eventually (near time 8000), the follower's strategy converges to that of Nash as well.

While in our simulations of relatively simple games, the asymptotic play of non-responsive learning algorithms is confined to D^∞, we have no proof that non-responsive, naive learning algorithms in general remain inside D^∞; in fact, the authors are divided as to whether this result holds in the presence of asynchrony. This subject warrants further study. From the perspective of network contexts, however, the analysis of such questions is moot, since non-responsive learners are unsatisfactory for a much simpler reason: their performance is sub-optimal in non-stationary settings.



(a) Informed Version



(b) Naive Version

Figure 9: *Asynchronous, Non-responsive Learning in Game D via Foster's and Vohra's algorithm.* (a) plots the Stackelberg equilibrium strategy weights in the informed case; notice that play quickly converges to Nash equilibrium. (b) plots these weights in the naive case; notice that play again converges to Nash equilibrium.

4.2.1 Non-stationary Environments

Consider a simple, one-player two-strategy game where the payoffs initially are 1 for strategy A and 0 for strategy B. We simulate a non-stationary version of this game where the payoffs are reversed every 5,000 rounds.

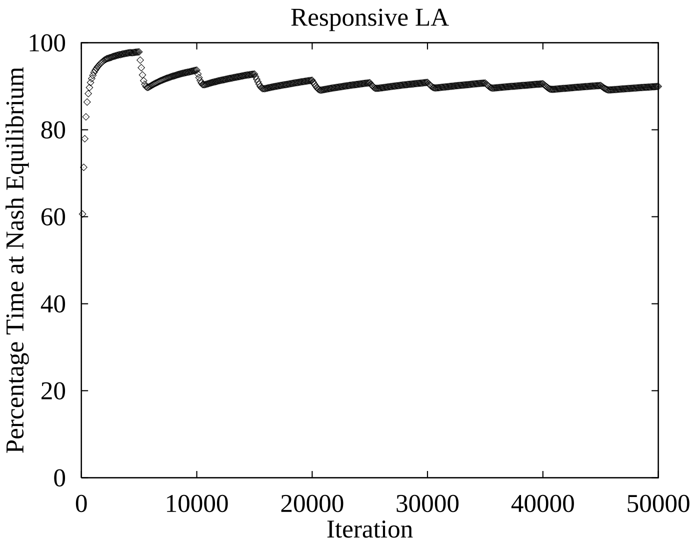
Figs. 10 and 11 case (a) plot the cumulative percentage of time spent playing the optimal strategy in simulations of sample reasonable learning algorithms.[21] All the reasonable learning algorithms – namely stage learning, responsive learning automata, and the algorithm due to Roth and Erev – spend over 90% of their time at the current optimal strategy in the simulated quasi-static environment. In addition, the resulting fluctuations in the weight of strategy A in this game are depicted in (b); observe that the weight of strategy A changes with the state of the environment.

In contrast to the reasonable learning algorithms, the non-responsive, no regret algorithms (both the naive and informed versions) perform poorly in non-stationary environments. Fig. 12 plots the cumulative percentage of time spent playing the Nash equilibrium for Freund's and Schapire's no external regret algorithm, in both its responsive and non-responsive forms.[22] Note that the non-responsive version of the algorithm spends only about 50% of its time playing the currently optimal strategy. This behavior is representative of all the no regret algorithms studied. This is because the non-responsive no regret algorithms fixate on one strategy early on – the one that is initially optimal – and are unable to adjust to future changes in environmental conditions.

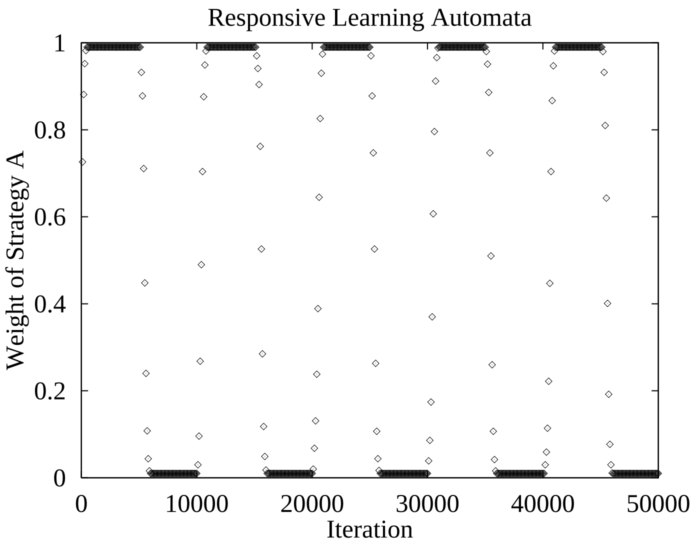
Thus, the criterion of no regret, while perhaps appropriate for learning in static environments, is apparently not sufficient for learning non-stationary payoff functions. Since network contexts are typically non-stationary and since this non-stationarity can be detected only via experimentation (one

[21] The algorithmic parameters for the reasonable learning algorithms were chosen as follows: $\epsilon = \gamma = .01$.

[22] For simulation purposes, in the algorithm due to Freund and Schapire, $\beta = 1$, and in the responsive case, γ was set equal to .01.

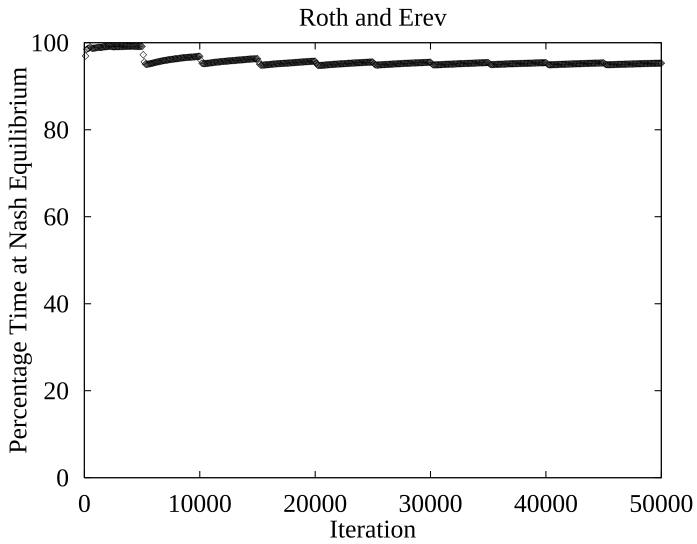


(a) Percentage Time at Nash Equilibrium

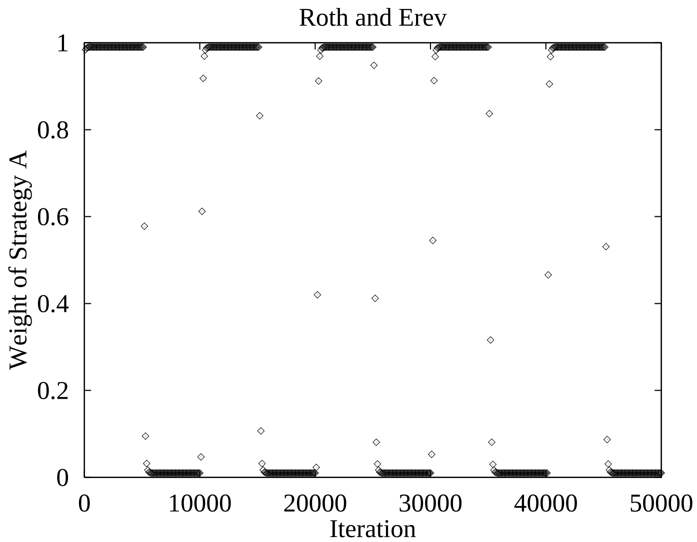


(b) Weight of Strategy A

Figure 10: *Responsive LA in Quasi-static Environment.* (a) Percentage Time at Nash Equilibrium. (b) Weight of Strategy A.

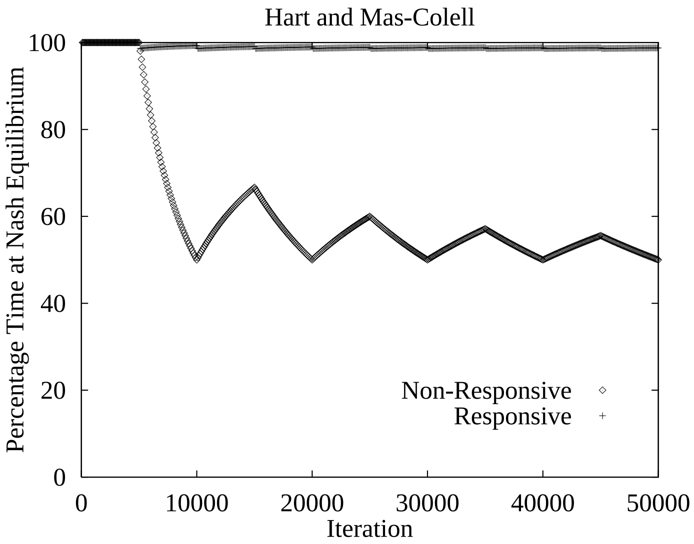


(a) Percentage Time at Nash Equilibrium

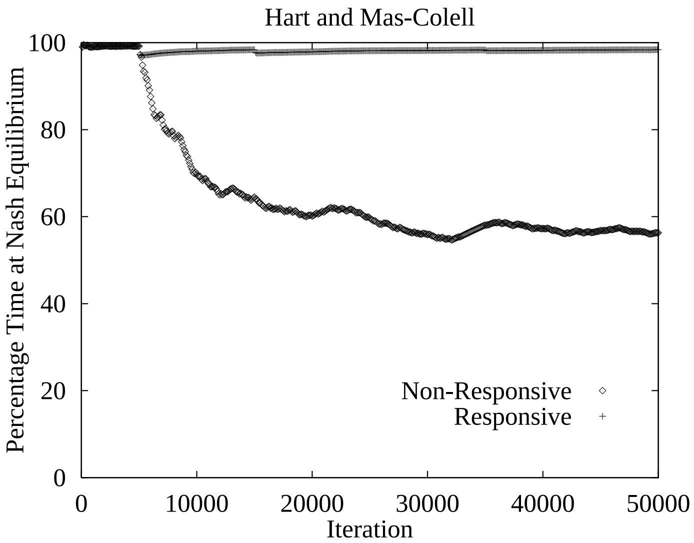


(b) Weight of Strategy A

Figure 11: *Roth's and Erev's Algorithm in Quasi-static Environment.* (a) Percentage Time at Nash Equilibrium. (b) Weight of Strategy A.



(a) Informed Version



(b) Naive Version

Figure 12: *Hart's and Mas-Colell's Algorithm in Quasi-static Environment.* Responsive vs. Non-responsive Learning. (a) Informed Version. (b) Naive Version.

cannot observe the change in the structure of the game directly), the learning algorithms employed should be responsive.

4.2.2 Shapley Game

In this section, we compare the behavior of responsive and non-responsive learners in the Shapley game (see Fig. 4.2.2), a well-known game in which fictitious play, an informed and non-responsive algorithm, does not converge. In this game, fictitious play results in cycling through the cells with 1's in them, namely cells *1, 2, 5, 6, 7*, and *9* in Fig. 4.2.2, with ever-increasing lengths of play in each of these cells [43]. One is led to conjecture that this fascinating behavior arises because of the clear-cut choices made by fictitious play – with probability 1, the strategy with the highest expected payoff is played, leading to abrupt transitions in the trajectory of play.

1 \ 2	*L*	*C*	*R*
T	1,0 1	0,1 2	0,0 3
M	0,0 4	1,0 5	0,1 6
B	0,1 7	0,0 8	1,0 9

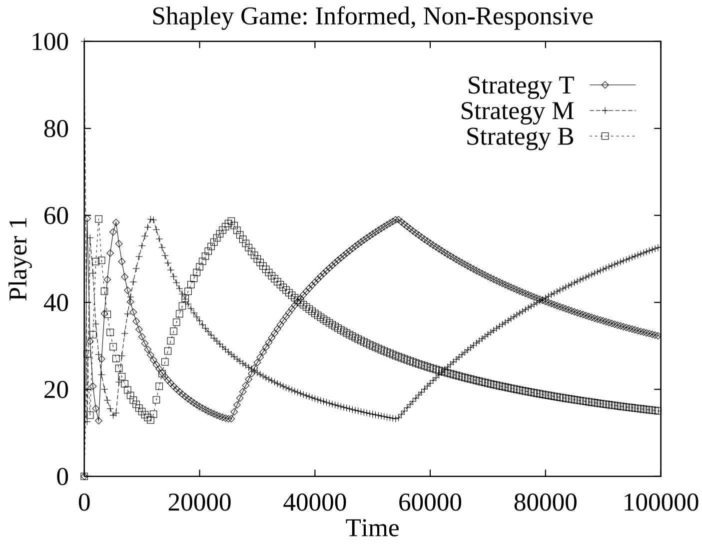
Figure 13: Shapley Game

Surprisingly, in our simulations,[23] we observe behavior similar to that of fictitious play for the non-responsive learning algorithms[24] – both informed (see Figs. 14 (a) and 15 (a)) and naive (see Fig. 14 (b) and 15 (b)) – even though these algorithms do not in general induce discrete changes. Fig. 14 plots the cumulative percentage of time that player 1 plays each of the three strategies; although not depicted here, the behavior patterns of player 2 are identical. In addition, Fig. 15 depicts the joint empirical frequencies of the various strategy profiles after 10^6 iterations, where the x-axis is labeled *1,...,9* corresponding to the cells in Fig. 4.2.2. Via this joint perspective, we see that both informed and naive, non-responsive learners spend very little time playing in cells without a 1 in them. Specifically, in the informed case, the likelihood of play in cells *3*, *4*, and *8* approaches 0, and in the naive case this likelihood approaches ϵ/N.

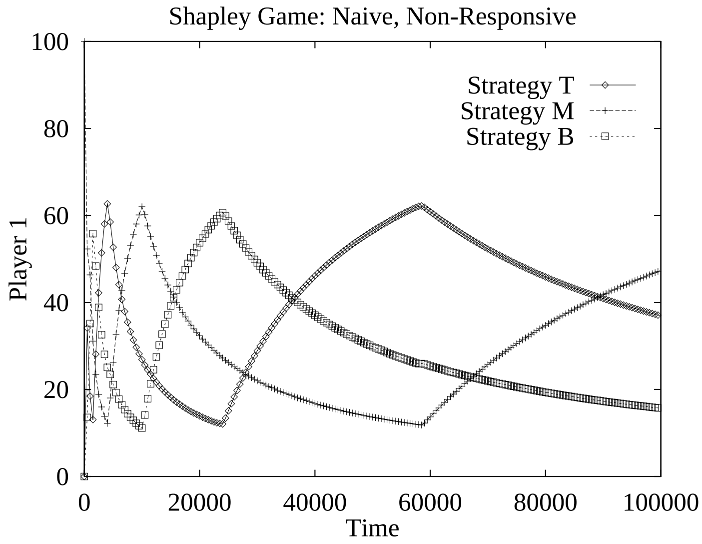
In contrast, the responsive algorithms, while they *do* display the same cycling behavior, the duration of play in each cell does *not* continue to grow. Instead the responsive algorithms spend equal amounts of time in each of the distinguished cells. This is depicted in Fig. 16 (a) (the informed case) and Fig. 16 (b) (the naive case), which plot the cumulative percentage of time player 1 spends playing each of the three strategies. Notice that these graphs converge to 33% for all strategies. In addition, Fig. 17 depicts the joint empirical frequencies of the various strategy profiles after 10^6 iterations. One interesting feature of the set of responsive learning algorithms is that their empirical frequencies converge to that of the fair and Pareto optimal correlated equilibrium; in particular, both players have expected average payoff of 1/2. This follows from the bounded memory of these algorithms.

[23] The graphs present the results of simulations of the algorithm due to Freund and Schapire.

[24] The only exception is the algorithm of Hart and Mas-Colell which is known to converge to correlated equilibrium, and in fact converges to the mixed strategy Nash equilibrium in the Shapley game.

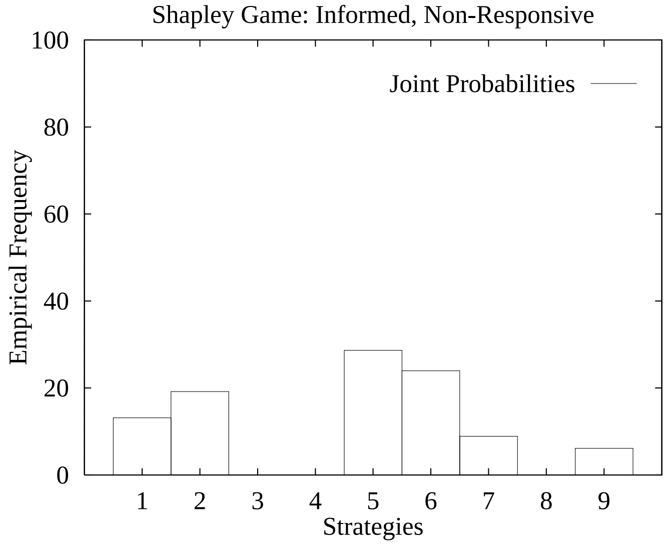


(a) Informed, Non-Responsive

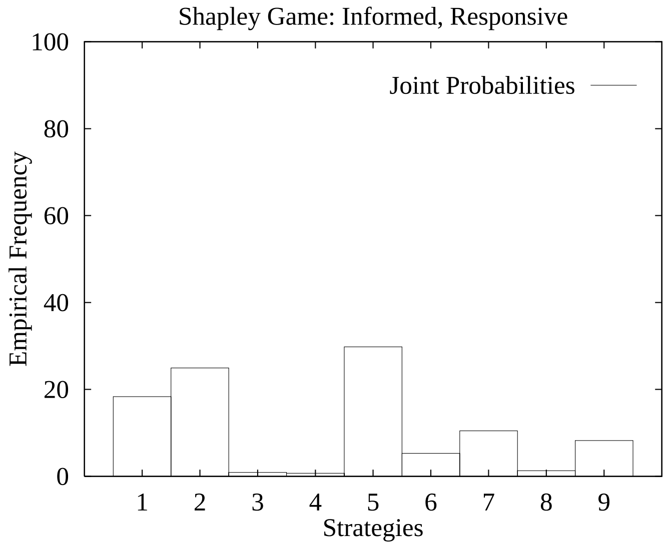


(b) Naive, Non-Responsive

Figure 14: *Non-Responsive Learning in the Shapley Game.* The cumulative percentage of time player 1 plays each of his three strategies assuming non-responsive learning. (a) Informed, Non-Responsive. (b) Naive, Non-Responsive.

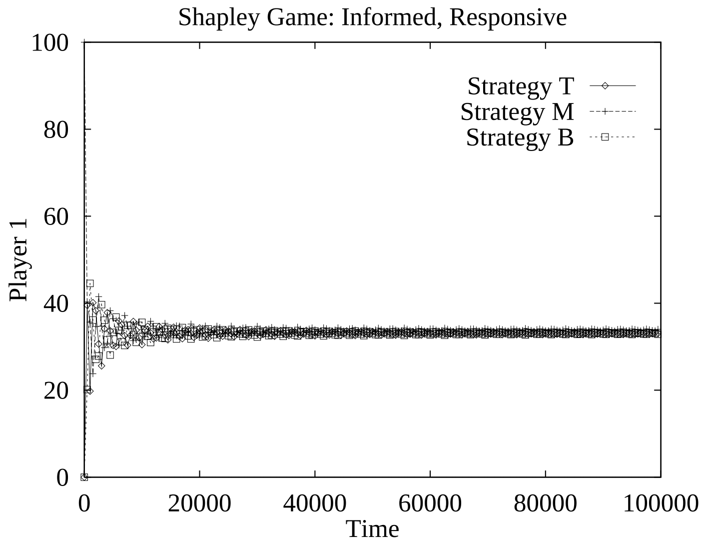


(a) Informed, Non-Responsive

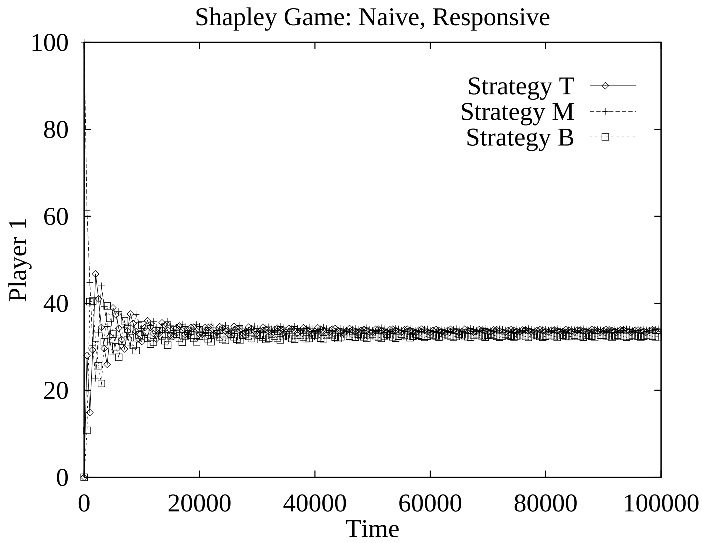


(b) Naive, Non-Responsive

Figure 15: *Joint Empirical Frequencies of Strategy Profiles in the Shapley Game assuming Non-responsive Learning.* The x-axis is labeled *1,...,9*, which corresponds to the cells in Fig. 13. (a) Informed, Non-Responsive. (b) Naive, Non-Responsive.

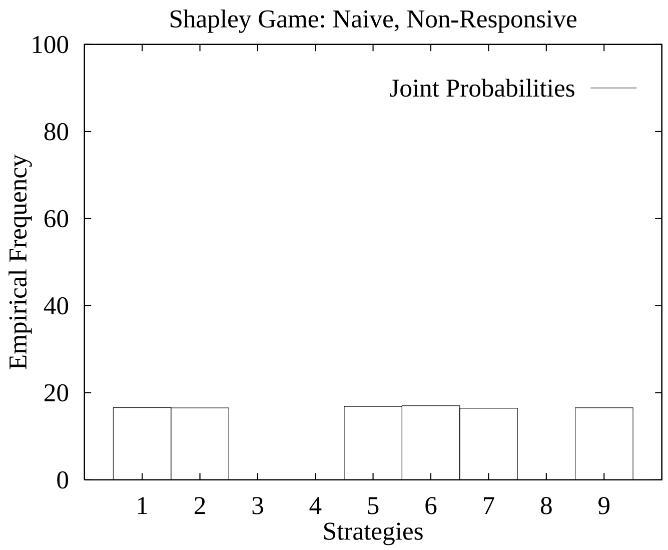


(a) Informed, Responsive

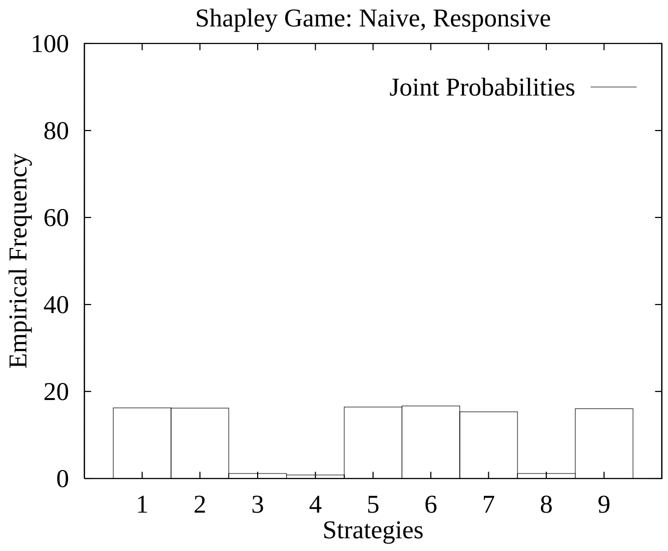


(b) Naive, Responsive

Figure 16: *Responsive Learning in Shapley Game.* Cumulative percentage of time player 1 plays each of his three strategies assuming responsive learning. (a) Informed, Responsive. (b) Naive, Responsive.



(a) Informed, Responsive



(b) Naive, Responsive

Figure 17: *Joint Empirical Frequencies of Strategy Profiles in the Shapley Game assuming Responsive Learning.* The x-axis is labeled *1,...,9*, which corresponds to the cells in Fig. 13. (a) Informed, Responsive. (b) Naive, Responsive.

5 Related Work

There is a vast body of economics literature on learning through repeated play of games, and we make no attempt here to provide a detailed review; for a comprehensive discussion, see the review by Fudenberg and Levine [22]. There is also substantial interest within the artificial intelligence community in the area of multi-agent learning; see the recent special issue of *Machine Learning* on this topic. In this section, we place our work in the context of the varying approaches to learning taken by economics and artificial intelligence researchers.

5.1 Relevance to Economics

The work on learning in economics falls roughly into two camps. The "high-rationality" approach involves learning algorithms that aim to predict their opponents' strategies, and then optimize with respect to those predictions. Prediction methods can be Bayesian (as in Kalai and Lehrer [32]), calibrated (as in Foster and Vohra [13]), or consistent (as in Fudenberg and Levine [21, 20]), just to name a few examples. Typically, the asymptotic play of high-rationality learning is either a correlated or Nash equilibrium. Since these algorithms depend on knowledge of the underlying structure of the game, however, they are not applicable in network contexts, which are of interest in this study.

In contrast, the "low-rationality" approaches to learning are concerned with situations similar to that which concerns us here, where agents have no information about the game other than the payoffs they receive. Examples of such work include Roth and Erev [41], Erev and Roth [10], Borgers and Sarin [5], Mookerji and Sopher [37], and Van Huyck *et al.* [29]; most of these algorithms as they were initially proposed are not responsive, but as we show in Appendix A, they can be made responsive via slight modifications. The focus of these papers is typically on matching the results of human experiments, whereas we focus instead on the nature of the asymptotic play. Chen [7], however, performed experiments on the congestion game discussed in Section 3.3 in which she compared synchronous and asynchronous play, as well as learning in full information one-shot games (zipper design, where play is repeated, but is against different opponents) versus naive learning in repeated games.

5.2 Relevance to Artificial Intelligence

In the context of artificial intelligence research, the present study is related to work in machine learning, where our sample of algorithms would be classified as reinforcement learning schemes. Unlike machine learning in multi-agent systems (see Stone and Veloso [46]) — systems designed to coordinate the interactions of albeit independently motivated, but centrally designed agents — we are interested in multi-agent learning in network contexts, where agent design is outside the realm of system design. Recall that network contexts are characterized by naiveté regarding payoff information, non-stationary environments, and asynchronous interactions.

Traditionally, reinforcement learning algorithms (see textbook by Sutton and Barto [47] and survey by Kaelbling, *et al.* [30]) were designed for single agent learning in stationary environments where payoffs are determined by state. Like learning in network contexts, reinforcement learning is naive in the sense that full payoff information is not assumed to be available to the learner; however, since reinforcement learning algorithms are not designed for use in non-stationary environments, they aim to maximize the discounted sum of expected future returns, rather than exhibit myopic behavior as do the algorithms considered here. The theoretical bounds that are known for reinforcement learning algorithms such as Q-learning [48] do not apply in the non-stationary settings that are typical of network contexts.

Recently, reinforcement learning researchers have expanded their interests to multi-agent learning. Such learning, however, can be tricky simply because as agents learn, their actions change, which implies that their impact on the environment changes, often leading to non-stationarities. For simplicity, most studies in multi-agent learning focus on settings where payoffs are either negatively correlated, as in zero-sum games (see, for example, Littman [33]), or positively correlated, as in coordination games (see, for example, Shoham and Tennenholtz [45]). Two notable exceptions include Wellman and Hu [50], who conducted a theoretical investigation of multi-agent learning in market interactions, and Sandholm and Crites [42], who conducted empirical studies of multi-agent reinforcement learning in the Prisoners' Dilemma. Similarly, this work considers multi-agent learning in positive sum games.

The results described here on convergence to Stackelberg equilibria, rather than traditional game-theoretic solutions such as D^∞, were largely a result of the asynchronous nature of play. Likewise, Huberman and Glance [28] simulate the evolutionary dynamics of both synchronous and asynchronous updating in the repeated Prisoners' Dilemma, and find that rather different behaviors arise. Perhaps of closer interest to the present study, empirical investigations of asynchronous Internet interactions were reported by Lukose and Huberman [34], but their studies concern time correlations arising from such behavior rather than properties of convergence. The convergence of asynchronous machine learning algorithms, however, where players' moves are determined by discrete random processes, have been investigated recently in e-commerce pricing models by Greenwald and Kephart [25].

Finally, regarding theoretical computer science, the algorithms which we refer to as satisfying no regret optimality criteria, are also often described as achieving reasonable competitive ratios. Borodin and El-Yaniv [6] present a thorough discussion of the competitive analysis of on-line algorithms.

6 Conclusion

This paper presented the results of experiments conducted using six learning algorithms, embodying three distinct notions of optimality: one average-case performance measure, and two worst-case performance measures, namely no external and no internal regret. In the suite of relatively simple games which were examined here, all the algorithms exhibited qualitatively similar behavior. Thus, it seems that in network contexts, the key property is not which type of optimality is achieved, but rather, responsiveness.

In low-information settings, where learners are necessarily naive and thus cannot detect changes in the structure of the game directly, algorithms should be able to respond to environmental changes in bounded time. It is shown in this paper that when such naive and responsive learning algorithms operate in asynchronous environments, the asymptotic play need not lie within D^∞. The question of whether play outside D^∞ arises for naive but non-responsive players in asynchronous environments remains open, but presumably would only arise in games with more players and larger strategy sets than have been studied in this paper.

It has been established previously that for reasonable learning algorithms the asymptotic play is contained within O^∞, and it was further conjectured that such play is contained within the smaller set S^∞. Our experimental results are consistent with this conjecture. While these results are suggestive, they are in no way conclusive, and so we are left with the open question of what the appropriate solution concept is for naive and responsive learning algorithms in asynchronous settings.

A Learning Algorithms

For completeness, this appendix describes the learning algorithms which were simulated in this study: responsive learning automata, due to Friedman and Shenker [16], responsive learning via additive updating, due to Roth and

Erev [9], multiplicative updating, due to Freund and Schapire [1], the mixing method, due to Foster and Vohra [11], and finally, learning via no internal regret, due originally to Foster and Vohra [13], and simplified by Hart and Mas-Colell [27]. The algorithms are presented in several varieties, depending upon whether they are applicable in informed or naive settings, and whether or not they are responsive. In informed settings, the counterfactual is known – in other words, players are informed of the potential payoff which they might have achieved had they employed any of their other strategies – while in naive settings, the only information pertaining to the payoff structure that is available at a given time is the payoff of the strategy that is in fact employed at that time. Informally, an algorithm is responsive if it weights recent information more heavily than past information and has a non-zero lower bound on the level of experimentation.[25]

A.1 Notation

The algorithms in this Appendix are described from the point of view of a single player engaging in a game against nature. Note that this point of view gives rise to notation which deviates from that of the main paper. The player chooses strategic actions from a strategy set $N = \{1, \ldots, n\}$, for $n \in \mathbb{N}$. Let $i, j \in N$. The payoff awarded at time t by employing strategy i is denoted π_i^t, and in general, the payoff function at time t is given by $\pi^t : N \to [a, b]$, for $a, b \in \mathbb{R}$. Note that payoffs are assumed to be bounded. For all the types of learning considered in this paper, the algorithm that determines strategic choices is described in terms of a time-dependent vector of probabilities, namely $p^t = (p_1^t, \ldots, p_i^t, \ldots, p_n^t)$, where p_i^t denotes the probability of playing strategy i at time t.

[25] The formal definition of responsiveness is given in [17].

A.2 Responsive Learning Automata

Responsive learning automata (RLAs) are introduced in [16]. This algorithm is designed for network contexts; it is applicable in naive settings and it is are responsive, as the name suggests. It is known that RLAs are reasonable learners, and therefore, RLAs converge to O^∞ [17]; moreover, in the case of synchronous play, RLAs converge to D^∞ [16].

RLAs are a responsive extension of a simple learning automaton (for a survey of the literature, see Narendra and Thathachar [38]). This simple learning automata is constructed by letting $p_i^0 = 1/n$ and using the following update rule when strategy i is played at time t:

$$p_i^{t+1} = p_i^t + \gamma\pi_i^t(1 - p_i^t) \tag{1}$$

$$p_j^{t+1} = p_j^t(1 - \gamma\pi_i^t), \quad \forall j \neq i \in N \tag{2}$$

where γ is a parameter that controls the tradeoff between learning rapidly (for γ close to 1) and accuracy (for γ close to 0). This learning automaton is not responsive. RLAs achieve responsiveness simply by imposing the requirement that the probability of any strategy never fall below $\epsilon/(n-1)$ according to the following update rules:

$$p_i^{t+1} = p_i^t + \gamma\pi_i^t \sum_{j\neq i} a_j^t p_j^t \tag{3}$$

$$p_j^{t+1} = p_j^t(1 - \gamma\pi_i^t a_j^t), \quad \forall j \neq i \in N \tag{4}$$

where $a_j^t = \min\left\{1, \frac{p_j^t - \epsilon/(n-1)}{\gamma\pi_i^t p_j^t}\right\}$.

The following two sections describe alternative learning mechanisms based on additive updating due to Roth and Erev [9] and Foster and Vohra [13].

A.3 Simple Additive Updating

A second example of learning via additive updating is the naive learning algorithm of Roth and Erev [9] which is reasonable, for certain choices of the parameters. The updating scheme is as follows: $p_i^t = q_i^t / \sum_{j \in N} q_j^t$. If strategy i is played at time t then the q_i's are updated as follows:

$$q_i^{t+1} = (1-\gamma) q_i^t + \pi_i^t (1-\epsilon) \tag{5}$$

$$q_j^{t+1} = (1-\gamma) q_j^t + \pi_i^t (\epsilon/(n-1)), \quad \forall j \neq i \in N \tag{6}$$

where γ behaves as in the RLA. It is straightforward, although quite tedious, to show that this algorithm is reasonable.

The following sections describe additive and multiplicative learning rules that satisfy the no external regret optimality criterion.

A.4 Additive Updating Revisited

This section presents an additive updating rule due to Foster and Vohra, known as the mixing method [11], which exhibits no external regret. The mixing method was originally designed for use in informed settings, and consequently updates weights according to the cumulative payoffs achieved by all strategies, including the payoffs that would have been obtained by strategies which were not played. The cumulative payoffs obtained at time t by strategy i (notation ρ_i^t) is computed as follows: $\rho_i^t = \sum_{x=0}^{t} \pi_i^x$.

In the case of two strategies, say A and B, the mixing method updates the weight of strategy i as follows:

$$w_A^{t+1} = \min \left\{ \max \left\{ \frac{1}{2} + \frac{\alpha(\rho_A^t - \rho_B^t)}{2(\rho_A^t + \rho_B^t)}, 0 \right\}, 1 \right\} \tag{7}$$

where the truncations ensure that the probability assigned to strategy A is between 0 and 1. It is shown in Foster and Vohra [11] that the optimal value of α is $\sqrt{T}$, where T is the number of iterations, at which point, the mixing method exhibits no external regret. In general, when the number of strategies $n > 2$, the algorithm utilizes pairwise mixing of strategies via Eq. 7, followed by further mixing of the mixtures.

The mixing method can be modified for use in naive settings by utilizing an estimate of cumulative payoffs that depends only on the payoffs obtained by the strategies that are actually employed and the weights associated with those strategies. In particular, let $\hat{\pi}_i^t = (\mathbf{I}_i^t/p_i^t)\pi_i^t$ where $\mathbf{I}_i^t$ is the indicator function: $\mathbf{I}_i^t = 1$ if strategy i is played at time t, and $\mathbf{I}_i^t = 0$, otherwise. In other words, $\hat{\pi}_i^t$ is equal to 0 if strategy i is not employed at time t; otherwise, $\hat{\pi}_i^t$ is the payoff achieved by strategy i at time t scaled by the likelihood of playing strategy i. Now estimated cumulative payoffs (notation $\hat{\rho}_i^t$) are given by: $\hat{\rho}_i^t = \sum_{x=0}^t \hat{\pi}_i^x$. The naive variant of the mixing method uses $\hat{\rho}_i^t$ in place of ρ_i^t in Eq. 7. This update procedure yields a set of weights which must then be adjusted for use in naive settings, in order to ensure that the space of possible payoffs be adequately explored. This is achieved by imposing an artificial lower bound on the probability with which strategies are played. In particular, let $\hat{p}_i^t = (1-\epsilon)p_i^t + \epsilon/n$, and compute $\hat{\pi}_i^t$ in terms of $\hat{p}_i^t$.

Finally, the mixing method can be modified to achieve responsiveness by utilizing exponential smoothing. In the responsive variant of this algorithm $\tilde{\rho}_i^t$ is substituted for either ρ_i^t or $\hat{\rho}_i^t$, depending on whether the setting is informed or naive. In particular, for $0 < \gamma \leq 1$, in informed settings and naive settings, respectively, $\tilde{\rho}_i^{t+1} = (1-\gamma)\tilde{\rho}_i^t + \pi_i^{t+1}$ and $\tilde{\rho}_i^{t+1} = (1-\gamma)\tilde{\rho}_i^t + \hat{\pi}_i^{t+1}$.

A.5 Multiplicative Updating

This section describes an algorithm due to Freund and Schapire [14] that achieves no external regret in informed settings via multiplicative updating. The development of the variants of the multiplicative updating algorithm is analogous to the development of additive updating.

The multiplicative update rule is computed in terms of the cumulative payoffs achieved by all strategies (namely ρ_i^t) including the surmised payoffs of those strategies which are not played. In particular, the weight assigned to strategy i at time $t+1$, for $\beta > 0$, is given by:

$$p_i^{t+1} = \frac{(1+\beta)^{\rho_i^t}}{\sum_{j \in N}(1+\beta)^{\rho_j^t}} \tag{8}$$

The multiplicative updating rule given in Eq. 8 can be modified in a manner identical to the mixing method, using $\hat{\pi}_i^t$ and $\hat{p}_i^t$ to become applicable in naive settings, and using $\tilde{\rho}_i^t$ to achieve responsiveness. A naive variant of this multiplicative updating algorithm which achieves no external regret appears in Auer, Cesa-Bianchi, Freund, and Schapire [1].

A.6 No Internal Regret

This section describes an algorithm due to Foster and Vohra [13] that achieves no internal regret in informed environments, and a simple implementation due to Hart and Mas-Colell [27]. In addition, the appropriate naive and responsive modifications are presented. Note that learning via no internal regret algorithms converges to correlated equilibrium [12, 27], and therefore converges inside the set D^∞.

Regret can be interpreted as a feeling of remorse over something that happened as a result of one's own actions. Formally, the regret felt by player i at time t is formulated as the difference between the payoffs obtained via strategy i and the payoffs that could have been achieved had strategy j been played whenever i had been played in the past: *i.e.*, $\text{R}_{i\to j}^t = \mathbf{I}_i^t[\pi_j^t - \pi_i^t]$. Cumulative regret is given by: $\text{CR}_{i\to j}^t = \sum_{x=0}^t \text{R}_{i\to j}^x$ and internal regret is defined as $\text{IR}_{i\to j}^t = (\text{CR}_{i\to j}^t)^+$, where $X^+ = \max\{X, 0\}$.

Consider the case of a 2-strategy game, with strategies A and B. The components of the weight vector, namely p_A^{t+1} and p_B^{t+1}, are updated via the following formulae, which reflect cumulative feelings of regret:

$$p_A^{t+1} = \frac{\text{IR}_{B\to A}^t}{\text{IR}_{A\to B}^t + \text{IR}_{B\to A}^t} \qquad \text{and} \qquad p_B^{t+1} = \frac{\text{IR}_{A\to B}^t}{\text{IR}_{A\to B}^t + \text{IR}_{B\to A}^t} \tag{9}$$

If there is significant regret for having played strategy B rather than strategy A, then the algorithm updates weights such that the probability of playing strategy A is increased. This algorithm is due to Foster and Vohra [13]. In general, if strategy i is played at time t,

$$p_j^{t+1} = \frac{1}{\kappa}\text{IR}_{i\to j}^t \qquad \text{and} \qquad p_i^{t+1} = 1 - \sum_{j\neq i} p_j^{t+1} \tag{10}$$

for $\kappa > 0$. This algorithm is due to Hart and Mas-Collell [27].

In a naive setting, it is necessary to compute an estimate of internal regret. Recall that the regret at time t for having played strategy i rather than playing strategy j is given by: $\text{R}^t_{i \to j} = \mathbf{I}^t_i[\pi^t_j - \pi^t_i]$. An estimated measure of expected regret $\hat{\text{R}}^t_{i \to j}$ is given by: $\hat{\text{R}}^t_{i \to j} = \mathbf{I}^t_i[\hat{\pi}^t_j - \hat{\pi}^t_i]$. Note that the expected value of this estimated measure of regret is the the actual measure of regret. Now estimated cumulative internal regret is $\hat{\text{IR}}^t_{i \to j} = (\sum_{x=0}^t \hat{\text{R}}^x_{i \to j})^+$. Finally, weights are updated as in Eq. 10, with $\hat{\text{IR}}^t_{i \to j}$ used in place of $\text{IR}^t_{i \to j}$.

Like the additive and multiplicative updating algorithms, the no internal regret learning algorithm can be made responsive via exponential smoothing of regret. In the informed case, $\tilde{\text{R}}^{t+1}_{i \to j} = (1 - \gamma)\tilde{\text{R}}^t_{i \to j} + \mathbf{1}^{t+1}_i \text{R}^{t+1}_{i \to j}$, and in the naive case, $\tilde{\text{R}}^{t+1}_{i \to j} = (1 - \gamma)\tilde{\text{R}}^t_{i \to j} + \mathbf{1}^{t+1}_i \hat{\text{R}}^{t+1}_{i \to j}$, Finally, it suffices to use $\tilde{\text{IR}}^t_{i \to j} = (\tilde{\text{R}}^t_{i \to j})^+$ as a measure internal regret in the responsive case, where $\tilde{\text{R}}^t_{i \to j}$ is given by the relevant one of the two previous expressions, depending on whether the setting is informed or naive.

This concludes the discussion of the learning algorithms that were selected for simulation in this study. While this appendix is primarily a survey of the literature, it also includes extensions to the algorithms for applicability in network contexts. In future work, we hope to extend the scope of our simulations by considering additional related algorithms.

B Results on the Class EG

In this appendix, we discuss the claims that are made in Section 3.2 that pertain to Game $\text{EG}_{1.9}$ and Game $\text{EG}_{2.1}$. First of all, we note that for $\phi = 0$, both of these games are D-solvable. This can be shown via the general technique of analyzing the best-reply (Cournot) dynamics of the relevant class of games as in [15]; however it can also be shown directly for these simple examples.

Consider Game $\text{EG}_{1.9}$ for a set of 8 players. For player 7, even when

$\lambda = 8$, $\pi_7 = 7 - 8/1.9 > 0$. Thus, participation dominates non-participation for player 7, and similarly for players 5 and 6, implying that $\lambda \geq 3$. Now, given that players 5, 6, and 7 eliminate their dominated strategies, for players 0, 1, and 2, non-participation dominates participation. For example, if player 2 participates, then $\pi_2 \leq 2 - 4/1.9 < 0$. Thus, given that players 0, 1, and 2 eliminate participation because it is dominated, continuing this line of reasoning, we find that non-participation is dominated for players 3 and 4. Finally, the unique outcome via the iterated elimination of dominated strategies is $s = (0, 0, 0, 1, 1, 1, 1, 1)$. By an analogous argument, we note that Game $\mathrm{EG}_{2.1}$ is also D-solvable with outcome $s = (0, 0, 0, 1, 1, 1, 1, 1)$.

The above argument also shows that Game $\mathrm{EG}_{1.9}$ and Game $\mathrm{EG}_{2.1}$ are D-solvable, for $\phi \in (0, 1)$, since dominated strategies are unchanged for these values of ϕ. Moreover, these games are also O-solvable, for $\phi = 0$, since in this case one can show that any dominated strategy is also overwhelmed using the monotonicity properties of the payoff functions. However for ϕ close to 1, neither game is O-solvable. In particular, participation sometimes yields a higher payoff than non-participation, but sometimes this effect is reversed, depending on the strategies of the other players. For example, consider player 7 in Game $\mathrm{EG}_{1.9}$, when $\phi = .9$. Participation with seven other players yields payoffs of $\pi_7 = 2.79$; but non-participation with no other participants yields $\pi_7 = 5.83$, while non-participation with seven participants yields $\pi_7 = 2.51$. In fact, no strategies are overwhelmed either game for ϕ close to 1.

While both Game $\mathrm{EG}_{1.9}$ and Game $\mathrm{EG}_{2.1}$ are D-solvable, and neither are O-solvable, these games differ in terms of S-solvability. In particular, Game $\mathrm{EG}_{2.1}$ is S-solvable, while Game $\mathrm{EG}_{1.9}$ is not. Consider first Game $\mathrm{EG}_{1.9}$; in particular, consider the possible payoffs of player 2 as the leader. If player 2 participates, then player 3 does not, which results in payoffs of $\pi_2(1) = 2 - 5/1.9$; on the other hand, if player 2 opts out, then player 3 does in fact participate, which yields payoffs of $\pi_2(0) = \phi(2 - 6/1.9)$. Consequently, player 2 participates when $\phi > 6/11$, since this implies that $\pi_2(1) > \pi_2(0)$. Therefore, there exists a Stackelberg equilibrium in Game $\mathrm{EG}_{1.9}$ with player

2 as leader which differs from the Nash equilibrium. Now consider Game $EG_{2.1}$. In this game, regardless of whether or not player 2 participates, player 3 participates, since $\pi_3 = 3 - 6/2.1 > 0$. Consequently, player 2 only participates when $\pi_2(1) = 2 - 6/2.1 > \phi(2 - 6/2.1) = \pi_2(0)$ which cannot be satisfied for any $\phi < 1$. A more detailed argument can be used to show that this game is indeed S-solvable.

References

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 322–331. ACM Press, November 1995.

[2] A. Banos. On pseudo games. *The Annals of Mathematical Statistics*, 39:1932–1945, 1968.

[3] B.D. Bernheim. Rationalizable strategic behavior. *Econometrica*, 52(4):1007–1028, 1984.

[4] D. Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

[5] T. Borgers and R. Sarin. Learning through reinforcement and replicator dynamics. Mimeo, 1995.

[6] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, Cambridge, England, 1998.

[7] Y. Chen. Asynchronicity and learning in cost-sharing mechanisms. Mimeo, 1997.

[8] T. Cover. Universal portfolios. *Mathematical Finance*, 1(1):1 – 29, 1991.

[9] I. Erev and A. Roth. On the need for low rationality cognitive game theory: Reinforcement learning in experimental games with unique mixed strategy equilibria. Mimeo, 1996.

[10] I. Erev and A. Roth. On the need for low rationality cognitive game theory: Reinforcement learning in experimental games with unique mixed strategy equilibria. Mimeo, 1996.

[11] D. Foster and R. Vohra. A randomization rule for selecting forecasts. *Operations Research*, 41(4):704–709, 1993.

[12] D. Foster and R. Vohra. Calibrated learning and correlated equilibrium. *Preprint*, 1995.

[13] D. Foster and R. Vohra. Regret in the on-line decision problem. *Games and Economic Behavior*, 21:40–55, 1997.

[14] Y. Freund and R. Schapire. Game theory, on-line prediction, and boosting. In *Proceedings of the 9th Annual Conference on Computational Learning Theory*, pages 325–332. ACM Press, May 1996.

[15] E. Friedman. Learnability in a class of non-atomic games arising on the Internet. *Mimeo*, 1998.

[16] E. Friedman and S. Shenker. Synchronous and asynchronous learning by responsive learning automata. Mimeo, 1996.

[17] E. Friedman and S. Shenker. Learning and implementation on the Internet. Mimeo, 1997.

[18] E.J. Friedman. Dynamics and rationality in ordered externality games. *Games and Economic Behavior*, 16:65–76, 1996.

[19] D. Fudenberg and D. Levine. Reputation and equilibrium selection in games with a patient player. *Econometrica*, 57:759–778, 1989.

[20] D. Fudenberg and D. Levine. Conditional universal consistency. Mimeo, 1995.

[21] D. Fudenberg and D. Levine. Consistency and cautious fictitious play. *Journal of Economic Dynamics and Control*, 19:1065–1089, 1995.

[22] D. Fudenberg and D. Levine. *The Theory of Learning in Games*. MIT Press, Cambridge, 1998.

[23] J. Gittens. *Multi-armed Bandit Allocation Indices*. Wiley, New York, 1989.

[24] A. Greenwald. *Learning to Play Network Games*. PhD thesis, Courant Institute of Mathematical Sciences, New York University, New York, May 1999.

[25] A.R. Greenwald and J.O. Kephart. Shopbots and pricebots. In *Lecture Notes on Artificial Intelligence: Proceedings of the IJCAI Workshop on Agent-mediated Electronic Commerce*. Springer-Verlag, 2000. To Appear.

[26] J. Hannan. Approximation to Bayes risk in repeated plays. In M. Dresher, A.W. Tucker, and P. Wolfe, editors, *Contributions to the Theory of Games*, volume 3, pages 97–139. Princeton University Press, 1957.

[27] S. Hart and A. Mas Colell. A simple adaptive procedure leading to correlated equilibrium. Technical report, Center for Rationality and Interactive Decision Theory, 1997.

[28] B.A. Huberman and N.S. Glance. Evolutionary games and computer simulations. *Proceedings of the National Academy of Sciences USA*, 90:7716–7718, 1993.

[29] J. Van Huyck, R. Battalio, and F. Rankin. Selection dynamics and adaptive behavior without much information. Mimeo, 1996.

[30] L. Kaelbling, M. Littman, and A. Moore. Reinforcement learning: A survey. *Artificial Intelligence Research*, 4:237–285, 1996.

[31] E. Kalai and E. Lehrer. Rational learning leads to Nash equilibria. *Econometrica*, 61(5):1019–1045, 1993.

[32] E. Kalai and E. Lehrer. Rational learning leads to Nash equilibrium. *Econometrica*, 61:1019–1045, 1993.

[33] M. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of Eleventh International Conference on Machine Learning*, pages 157–163. Morgan Kaufmann, 1994.

[34] R. Lukose and B. Huberman. A methodology for managing risk in electronic transactions over the internet. Third International Conference on Computational Economics, June 1997.

[35] N. Megiddo. On repeated games with incomplete information played by non-Bayesian players. *International Journal of Game Theory*, 9:157–167, 1980.

[36] P. Milgrom and J. Roberts. Adaptive and sophisticated learning in normal form games. *Games and Economic Behavior*, 3:82–100, 1991.

[37] D. Mookherjee and B. Sopher. Learning and decision costs in experimental constant sum games. *Games and Economic Behavior*, 19:97–132, 1996.

[38] K. Narendra and M.A.L. Thathachar. Learning automata: A survey. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-4(4):323–334, 1974.

[39] D. Pearce. Rationalizable strategic behavior and the problem of perfection. *Econometrica*, 52:1029–1050, 1984.

[40] R. Rosenthal. A note on the robustness of equilibria with respect to commitment opportunities. *Games and Economic Behavior*, 3:237–243, 1991.

[41] A. Roth and I. Erev. Learning in extensive form games: experimental data and simple dynamic models in the intermediate term. *Games and Economic Behavior*, 8:164–212, 1995.

[42] T. Sandholm and R. Crites. Multiagent reinforcement learning in the iterated prisoners' dilemma. *Biosystems*, 37:147–166, 1995. Special Issue on the Prisoners' Dilemma.

[43] L.S. Shapley. A value for n-person games. In H. Kuhn and A.W. Tucker, editors, *Contributions to the Theory of Games*, volume II, pages 307–317. Princeton University Press, 1953.

[44] S. Shenker. Making greed work in networks: A game-theoretic analysis of switch service disciplines. *IEEE/ACM Transactions on Networking*, 3:819–831, 1995.

[45] Y. Shoham and M. Tennenholtz. Co-learning and the evolution of social activity. Mimeo, 1993.

[46] P. Stone and M. Veloso. Multiagent systems: A survey from a machine learning perspective. Technical Report 193, Carnegie Mellon University, December 1997.

[47] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction.* MIT Press, Massachusetts, 1998.

[48] C. Watkins. *Learning from Delayed Rewards.* PhD thesis, Cambridge University, Cambridge, 1989.

[49] J. Watson. A "reputation" refinement without equilibrium. *Econometrica*, 61:199–206, 1993.

[50] M. Wellman and J. Hu. Conjectural equilibrium in multi-agent learning. *Machine Learning*, 33:179–200, 1998. Special Issue on Multi-agent Learning.