12-2013

Adaptive regret minimization in bounded-memory games

Jeremiah BLOCKI

Nicolas CHRISTIN

Anupam DATTA

Arunesh SINHA

Singapore Management University, aruneshs@smu.edu.sg

Adaptive Regret Minimization in Bounded-Memory Games *

Jeremiah Blocki, Nicolas Christin, Anupam Datta, and Arunesh Sinha

Carnegie Mellon University, Pittsburgh, PA
{jblocki, nicolasc, danupam, aruneshs}@cmu.edu

Abstract Organizations that collect and use large volumes of personal information often use security audits to protect data subjects from inappropriate uses of this information by authorized insiders. In face of unknown incentives of employees, a reasonable audit strategy for the organization (defender) is one that minimizes his regret. While regret minimization has been extensively studied in repeated games, a repeated game cannot capture the full complexity of the interaction between the organization (defender) and an insider (adversary) that arises from dependence of rewards and actions on history. We introduce a richer class of games called *bounded memory games*, which can provide a more accurate model of the audit process. The standard notion of regret for repeated games is no longer suitable because actions and rewards can depend on the history of play. To account for this generality, we introduce the notion of k*-adaptive regret*, which compares the reward obtained by playing actions prescribed by the algorithm against a hypothetical k*-adaptive adversary* with the reward obtained by the best expert in hindsight against the same adversary. Roughly, a hypothetical k-adaptive adversary adapts her strategy to the defender's actions exactly as the real adversary would within each window of k rounds. A k-adaptive adversary is a natural model for temporary employees who stay for a certain number of audit cycles and are then replaced by a different person. Our definition is parametrized by a set of experts, which can include both fixed and adaptive defender strategies.

We investigate the inherent complexity of and design algorithms for adaptive regret minimization in bounded memory games of perfect and imperfect information. We prove a hardness result showing that, with imperfect information, any k-adaptive regret minimizing algorithm (with fixed strategies as experts) must be inefficient unless $\mathsf{NP} = \mathsf{RP}$ even when playing against an oblivious adversary. In contrast, for bounded memory games of perfect and imperfect information we present approximate 0-adaptive regret minimization algorithms against an oblivious adversary running in time $n^{O(1)}$.

* This work was partially supported by the U.S. Army Research Office contract "Perpetually Available and Secure Information Systems" (DAAD19-02-1-0389) to Carnegie Mellon CyLab, the NSF Science and Technology Center TRUST, the NSF CyberTrust grant "Privacy, Compliance and Information Risk in Complex Organizational Processes," the AFOSR MURI "Collaborative Policies and Assured Information Sharing," and HHS Grant no. HHS 90TR0003/01. Jeremiah Blocki was also partially supported by a NSF Graduate Fellowship. Arunesh Sinha was also partially supported by the CMU CIT Bertucci Fellowship. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

1 Introduction

Online learning algorithms that minimize regret provide strong guarantees in situations that involve repeatedly making decisions in an uncertain environment. There is a well developed theory for regret minimization in repeated games [1]. The goal of this paper is to study regret minimization for a richer class of settings. As a motivating example consider a hospital (defender) where a *series* of temporary employees or business affiliates (adversary) access patient records for legitimate purposes (e.g., treatment or payment) or inappropriately (e.g., out of curiosity about a family member or for financial gain). The hospital conducts audits to catch the violators, which involves expending resources in the form of time spent in human investigation. On the other hand, violations that are missed internally and caught externally (by Government audits, patient complaints, etc.) also result in various losses such as repuation loss, loss due to litigation, etc. The hospital wants to minimize its overall loss by balancing the cost of audits with the risk of externally detected violations. In these settings with unknown adversary incentives, a reasonable strategy for the defender is one that minimizes her regret.

Modeling this interaction as a repeated game of imperfect information is challenging because this game has two additional characteristics that are not captured by a repeated game model: (1) *History-dependent rewards*: The payoff function depends not only on the current outcome but also on previous outcomes. For example, when a violation occurs the hospital might experience a greater loss if other violations have occured in recent history. (2) *History-dependent actions*: Both players may *adapt* their strategies based on history. For example, if many violations have been detected and punished in recent history then a rational employee might choose to lay low rather than committing another violation.

Instead, we capture this form of history dependence by introducing *bounded memory games*, a subclass of stochastic games [1]. In each round of a two-player bounded-memory-m game, both players simultaneously play an action, observe an outcome and receive a reward. In contrast to a repeated game, the payoffs may depend on the state of the game. In contrast to a *general* stochastic game, the rewards may *only* depend on the outcomes from the last m rounds (e.g., violations that were caught in the last m rounds) as well as the actions of the players in the current round.

In a bounded memory game, the standard notion of regret for a repeated game is not suitable because the adversary may adapt her actions based on the history of play. To account for this generality, we introduce (in Section 4) the notion of *k-adaptive regret*, which compares the reward obtained by playing actions prescribed by the algorithm against a hypothetical *k-adaptive adversary* with the reward obtained by the best expert in hindsight against the same adversary. Roughly, a hypothetical k-adaptive adversary plays exactly the same actions as the real adversary except in the last k rounds where she adapts her strategy to the defender's actions exactly as the real adversary would. When $k = 0$, this definition coincides with the standard definition of an *oblivious ad-*

[1] Stochastic games [2] are expressive enough to model history dependence. However, there is no regret minimization algorithm for the *general class of stochastic games*. While we do not view this result as surprising or novel, we include it in the full version [3] of this paper for completeness.

versary considered in defining regret for repeated games. When $k = \infty$ we get a *fully adaptive adversary*. A k-adaptive adversary is a natural model for temporary employees (e.g., residents, contractors) who stay for a certain number of audit cycles and are then replaced by a different person. Our definition is parameterized by a set of experts, which can include both fixed and adaptive defender strategies. In section 5 we use the example of a police chief enforcing the speed limit at a popular tourist destination (or a hospital auditing accesses to the patient records made by residents) to illustrate the power of k-adaptive regret minimization when the defender plays against a series of temporary adversaries.

Next, we investigate the inherent complexity of and design algorithms for adaptive regret minimization in bounded-memory games of perfect and imperfect information. Our results are summarized in Table 1. We prove a hardness result (Section 6; Theorem 1) showing that, with imperfect information, any k-adaptive regret minimizing algorithm (with fixed strategies as experts) must be inefficient unless $\mathsf{NP} = \mathsf{RP}$ even when playing against an oblivious adversary and even when $k = 0$. In fact, the result is even stronger and applies to any γ-approximate k-adaptive regret minimizing algorithm (ensuring that the regret bound converges to γ rather than 0 as the number of rounds $T \to \infty$) for $\gamma < \frac{1}{8n^\beta}$ where n is the number of states in the game and $\beta > 0$. Our hardness reduction from MAX3SAT uses the state of the bounded-memory game and the history-dependence of rewards in a critical way.

We present an inefficient k-adaptive regret minimizing algorithm by reducing the bounded-memory game to a repeated game. The algorithm is inefficient for bounded-memory games when the number of experts is exponential in the number of states of the game (e.g., if all fixed strategies are experts). In contrast, for bounded-memory games of perfect information, we present an efficient $n^{O(1/\gamma)}$ time γ-approximate 0-adaptive regret minimization algorithm against an oblivious adversary for any constant $\gamma > 0$ (Section 7;Theorem 4). We also show how this algorithm can be adapted to get an efficient γ-approximate 0-adaptive regret minimization algorithm for bounded-memory games of imperfect information (Section 7;Theorem 5). The main novelty in these algorithms is an implicit weight representation for an exponentially large set of adaptive experts, which includes all fixed strategies.

	Imperfect Information	Perfect Information
Oblivious Regret ($k = 0$)	Hard (Theorem 1) APX (Theorem 5)	APX (Theorem 4)
k-Adaptive Regret ($k \geq 1$)	Hard (Theorem 1)	Hard (Full Version [3])
Fully Adaptive Regret ($k = \infty$)	X (Full Version [3])	X (Full Version [3])

Table 1. Regret Minimization in Bounded Memory Games
X - no regret minimization algorithm exists
Hard - unless $\mathsf{NP} = \mathsf{RP}$ no regret minimization algorithm is efficiently computable
APX - efficient approximate regret minimization algorithms exist.

2 Related Work

A closely related work is the Regret Minimizing Audit (RMA) mechanism of Blocki et al. [4], which uses a repeated game model for the audit process. RMA deals with history-dependent rewards under certain assumptions about the defender's payoff function, but it does not consider history-dependent actions. While RMA provides strong performance guarantees for the defender against a byzantine adversary, the performance of RMA may be far from optimal when the adversary is rational (or nearly rational). In subsequent work the same authors [5] introduced a model of a nearly rational adversary who behaves in a rational manner most of the time. A nearly rational adversary can usually be deterred from committing policy violations by high inspection and punishment levels. They suggested that the defender commit to his strategy before each audit round (e.g., by publically releasing its inspection and punishment levels) as in a Stackelberg game [6]. However, the paper gives no efficient algorithm for computing the Stackelberg equilibrium.

More recent work by Blocki et al. introduced the notion of Audit Games [7] — a simplified game theoretic model of the audit process in which the adversary is purely rational (unlike the nearly rational adversary of [5]). Audit Games generalize the model of Security Games [8] by including punishment level as part of the defenders action space. Because the punishment parameter introduces quadratic constraints into the optimization problem that must be solved to compute the Stackelberg equilibria, this apparently small change makes it difficult to find the Stackelberg equilibria. The primary technical contribution of [5] is an efficient algorithm for computing the Stackelberg equilibrium of Audit Games. There are two potential advantages of the k-adaptive regret framework compared with the Stackelberg equilibria appraoch: (1) The k-adaptive regret minimization algorithm can be used even if the adversary's incentives are unknown, and (2) A k-adaptive adversary is a better model for a short term adversary (e.g., contractors, tourists) who may not informed about the defender's policy — and therefore may not even know what the 'rational' best response is in a Stackelberg game. See section 5 for additional discussion.

Stochastic games were defined by Shapley [2]. Much of the work on stochastic games has focused on finding and computing equilibria for these games [2, 9]. There has been lot of work in regret minimization for repeated games [1]. Regret minimization in stochastic games has not been the subject of much research. Papadimitriou and Yannakakis showed that many natural optimization problems relating to stochastic games are hard [10]. These results don't apply to bounded memory games. Golovin and Krause recently showed that a simple greedy algorithm can be used when a stochastic optimization problem satisfies a property called adaptive submodularity [11]. In general, bounded memory games do not satisfy this property. Even-Dar, et al., show that regret minimization is possible for a class of stochastic games (Markov Decision Processes) in which the adversary chooses the reward function at each state but does not influence the transitions [12]. They also prove that if the adversary controls the reward function and the transitions, then it is NP-Hard to even approximate the best fixed strategy. Mannor and Shimkin [13] show that if the adversary completely controls the transition model (a Controlled Markov Process) then it is possible to separate the stochastic game into a series of matrix games and efficiently minimize regret in each

matrix game. Bounded-memory games are a different subset of stochastic games where the transitions and rewards are influenced by both players. While our hardness proof shares techniques with Even-Dar, et al., [12], there are significant differences that arise from the bounded-memory nature of the game. We provide a detailed comparison in Section 6.

In a recent paper, Even-Dar, et al., [14] handle a few specific global cost functions related to load balancing. These cost functions depend on history. In their setting, the adversary obliviously plays actions from a joint distribution. In contrast, we consider arbitrary cost functions with bounded dependence on history and adaptive adversaries.

Takimoto and Warmuth [15] developed an efficient online shortest path algorithm. In their setting the experts consists of all fixed paths from the source to the destination. Because there may be exponentially many paths their algorithm must use an implicit weight representation. Awerbuch and Kleinberg later provided a general framework for online linear optimization [16]. In our settings, an additional challenge arises because *experts adapt to adversary actions*. See Section 7 for a more detailed comparison.

Farias, et al., [17] introduce a special class of adversaries that they call "flexible" adversaries. A defender playing against a flexible adversary can minimize regret by learning the average expected reward of every expert. Our work differs from theirs in two ways. First, we work with a stochastic game as opposed to a repeated game. Second, our algorithms can handle a sequence of different k-adaptive adversaries instead of learning a single flexible adversary strategy. A single k-adaptive strategy is flexible, but a sequence of k-adaptive adversaries is not.

3 Preliminaries

Bounded-memory games are a sub-class of stochastic games, in which outcomes and states satisfy certain properties. Formally, a two-player stochastic game between an attacker A and a defender D is given by $(\mathcal{X}_D, \mathcal{X}_A, \Sigma, P, \tau)$, where $\mathcal{X}_A$ and $\mathcal{X}_D$ are the actions spaces for players A and D, respectively, Σ is the state space, $P : \Sigma \times \mathcal{X}_D \times \mathcal{X}_A \rightarrow [0,1]$ is the payoff function and $\tau : \Sigma \times \mathcal{X}_D \times \mathcal{X}_A \times \{0,1\}^* \rightarrow \Sigma$ is the randomized transition function linking the different states. Thus, the payoff during round t depends on the current state (denoted σ^t) in addition to the actions of the defender (d^t) and the adversary (a^t). We use $n = |\Sigma|$ to denote the number of states.

A *bounded-memory game with memory* m ($m \in \mathbb{N}$) is a stochastic game with the following properties: (1) The game satisfies independent outcomes, and (2) The states $\Sigma = \mathcal{O}^m$ encode the last m outcomes, i.e., $\sigma^i = \left(O^{i-1}, \ldots, O^{i-m}\right)$. An outcome of a given round of play is a signal observed by both players (called "public signal" in games [18]). Outcomes depend probabilistically on the actions taken by the players. We use $\mathcal{O}$ to denote the outcome space and $O^t \in \mathcal{O}$ to denote the outcome during round t. We say that a game satisfies *independent outcomes* if O^t is conditionally independent of $\left(O^1, ..., O^{t-1}\right)$ given d^t and a^t. Notice that the defender and the adversary in a game with independent outcomes may still select their actions based on history. However, once those actions have been selected, the outcome is independent of the game history. Note that a repeated game is a bounded-memory-0 game (a bounded-memory game with memory $m = 0$).

A game in which players only observe the outcome O^t after round t but not the actions taken during a round is called an *imperfect information* game. If both players also observe the actions then the game is a *perfect information* game.

The *history* of a game $H = \left(O^1, O^2, \ldots, O^i, \ldots, O^t\right)$, is the sequence of outcomes. We use H_k to denote the k most recent outcomes in the game (i.e., $H_k = \left(O^{t-k+1}; \ldots; O^t\right)$), and $t = |H|$ to denote the total number of rounds played. We use H^i to denote the first i outcomes in a history (i.e., $H^i = \left(O^1, \ldots, O^i\right)$), and $H; H'$ to denote concatenation of histories H and H'.

A *fixed strategy* for the defender in a stochastic game is a function $f : \Sigma \rightarrow \mathcal{X}_D$ mapping each state to a fixed action. F denotes the set of all fixed strategies.

4 Definition of Regret

As discussed earlier, regret minimization in repeated games has received a lot of attention [19]. Unfortunately, the standard definition of regret in repeated games does not directly apply to stochastic games. In a repeated game, regret is computed by comparing the performance of the defender strategy D with the performance of a fixed strategy f. However, in a stochastic game, the actions of the defender and the adversary in round i influence payoffs in each round for the rest of the game. Thus, it is unclear how to choose a meaningful fixed strategy f as a reference. We solve this conundrum by introducing an adversary-based definition of regret.

4.1 Adversary Model

We define a parameterized class of adversaries called k-adaptive adversaries, where the parameter k denotes the level of adaptiveness of the adversary. Formally, we say that an agent is *k-adaptive* if its strategy $A(H)$ is defined by a function $f : \mathcal{O}^* \times \mathbb{N} \rightarrow \mathcal{X}_A$ such that $A(H) = f\left(H_i, t\right)$, where $i = t \mod (k+1)$. Recall that H_i is the i most recent outcomes, and $t = |H|$.

As special cases we define an *oblivious adversary* $(k = 0)$ and a *fully adaptive adversary* $(k = \infty)$. Oblivious adversaries essentially play without any memory of the previous outcomes. Fully adaptive adversaries, on the other hand, choose their actions based on the entire outcome history since the start of the game. k-adaptive adversaries lie somewhere in between. At the start of the game, they act as fully adaptive adversaries, playing with the entire outcome history in mind. But, different from fully adaptive adversaries, every k rounds, they "forget" about the entire history of the game and act as if the whole game was starting afresh. As discussed earlier, there are numerous practical instances where k-adaptive adversaries are an appropriate model; for instance, in games in which one player (e.g., a firm) has a much longer length of play than the adversary (e.g., a temporary employee), it may be judicious to model the adversary as k-adaptive. In particular, k-adaptive adversaries are similar to the notion of "patient" players in long-run games discussed by [20]. Their notion of "fully patient" players correspond to fully adaptive adversaries, "myopic" players correspond to oblivious adversaries, and "not myopic but less patient" players correspond to k-adaptive adversaries.

Another possible adversary definition could be to consider a sliding window of size k as the adversary memory. But, because such an adversary can play actions to remind herself of events in the arbitrary past, her memory is not actually bounded by k, and regret minimization is not possible. See the full version [3] of this paper for details.

$\mathcal{A}_D^K$ and $\mathcal{A}_A^K$ denote all possible K-adaptive strategies for the defender and adversary, respectively.

4.2 k-Adaptive Regret

Suppose that the defender D and the adversary A have produced history H in a game G lasting T rounds. Let $a^1, ..., a^T$ denote the sequence of actions played by the adversary. In hindsight we can construct a hypothetical k-adaptive adversary A_k as follows:

$$A_k\left(H'\right) = A\left(H^{t-i}; H_i'\right) \ ,$$

where $t = |H'|$ and $i = t \mod (k+1)$. In other words, the hypothetical k-adaptive adversary replicates the plays the real adversary made in the actual game regardless of the strategy of the defender he is playing against, *except* for the last i rounds under consideration where he adapts his strategy to the defender's actions in the same manner the real adversary would.

Abusing notation slightly we write $P\left(f, A, G, \sigma_0, T\right)$ to denote the expected payoff the defender would receive over T rounds of G given that the defender plays strategy f, the adversary uses strategy A and the initial state of the bounded-memory game G is σ_0. We use $\bar{P}\left(f, A, G, T\right) = P\left(f, A, G, \sigma_0, T\right)/T$ to denote the average per-round payoff. We use

$$\bar{R}_k\left(D, A, G, T, S\right) = \max_{f \in S} \bar{P}\left(f, A_k, G, T\right) - \bar{P}\left(D, A_k, G, T\right) \ ,$$

to denote the *k-adaptive regret* of the defender strategy D using a fixed set S of experts against an adversary strategy A for T rounds of the game G.

Definition 1. *A defender strategy D using a fixed set S of experts is a* γ-approximate k-adaptive regret minimization algorithm *for the class of games $\mathcal{G}$ if and only if for every adversary strategy A, every $\epsilon > 0$ and every game $G \in \mathcal{G}$ there exists $T' > 0$ such that $\forall T > T'$*

$$\bar{R}_k\left(D, A, G, T, S\right) < \epsilon + \gamma \ .$$

If $\gamma = 0$ then we simply refer to D as a k-adaptive regret minimization algorithm. If D runs in time poly $(n, 1/\epsilon)$ we call D efficient.

k-adaptive regret considers a k-adaptive hypothetical adversary who can adapt within each window of size (at most) $k+1$. Intuitively, as k increases this measure of regret is more meaningful (as the hypothetical adversary increasingly resembles the real adversary), albeit harder to minimize.

There are two important special cases to consider: $k = 0$ (oblivious regret) and $k = \infty$ (adaptive regret). Adaptive regret is the strongest measure of regret. Observe that if the actual adversary is k-adaptive then the hypothetical adversary A_∞ is same as

the hypothetical adversary A_k, and hence $\bar{R}_\infty = \bar{R}_k$. Also, if the actual adversary is oblivious then $\bar{R}_\infty = \bar{R}_0 = \bar{R}_k$.

In this paper $\mathcal{G}$ will typically denote the class of perfect/imperfect information bounded-memory games with memory m. We are interested in expert sets S which contain all of the fixed strategies $F \subseteq S$.

5 Audit Examples

As an example, consider the interaction between a police chief (defender) and drivers (adversary) at a popular tourist destination. The police chief is given the task of enforcing speed limits on local roads. Each day the police chief may deploy resources (e.g., radar, policemen) to monitor local roads, and drivers decide whether or not to speed or not.

Repeated Game We first model the interaction above using a repeated game. We will consider a simple version of this interaction in which the defender has two actions

$$\mathcal{X}_D = \{\mathbf{HI}, \mathbf{LI}\} \ ,$$

and the adversary has two actions

$$\mathcal{X}_A = \{\mathbf{S}, \mathbf{DS}\} \ .$$

Here, **HI/LI** stands for high/low inspection and **S/DS** stands for speed and don't speed. We consider the defender utilities in table 2.

In this example, the costs of a higher inspection outweigh the benefits of enforcing the policy. In *any* Nash Equilibria the defender will play his dominant strategy — "always play **LI**." Similarly, *any* algorithm that minimizes regret in the standard sense (0-adaptive) — like the regret minimizing audit mechanism from [4] — must eventually converge to the dominant defender strategy **LI**. While this is the best that the defender can do against a byzantine adversary, this may not always be the best result for the defender when playing against a rational adversary. Consider the adversary's utility defined in table 3.

Actions	**S**	**DS**
HI	.19	0.7
LI	0.2	1

Table 2. Speeding Game — Defender Utility P

If the defender plays his dominant strategy then the adversary will always play the action **S** — speed. This action profile results in average utility 0.2 for the defender and 1 for the adversary. However, if the defender can commit to his strategy in advance then he can play his Stackelberg equilibrium [6] strategy "play **HI** with probability 0.2 and **LI** with probability 0.8." A rational adversary will respond by playing her best response — the action that maximizes her utility given the defenders commitment. In this case the adversary's best response is to play **DS**. The resulting utility for the defender is 0.94!

There are two practical challenges with adopting this approach: (1) If the utility of the adversary is unknown then the defender cannot compute the Stackelberg equilibrium. (2) Even

Actions	**S**	**DS**
HI	0	0.8
LI	1	0.8

Table 3. Speeding Game — Adversary Utility

if the defender commits to playing a Stackelberg equilibrium it is unlikely that many drivers will respond in purely rational manner for the simple reason that they are uniformed (e.g., a tourist may not know whether or not speed limits are aggressively enforce in an unfamiliar area). If the adversary can learn the Stackelberg Equilibrium from a history of the defender's actions, then she might adapt her play to the best response strategy over time. However, each tourist has a limited time window in which she can make these observations and adjust her behavior (e.g., the tourist leaves after at most k days).

Bounded Memory Game Model with k-adaptive regret We model the interaction above using bounded memory games with k-adaptive adversary model. In each round of our bounded memory game the defender and the adversary play an action profile, and observe an outcome — a public signal. The action space in our bounded memory game is identical to the repeated game, and the outcome $\mathcal{O} = \{\mathbf{HI}, \mathbf{LI}\}$ is simply the defender's action. That is we assume that our tourist driver can observe the defender's inspection level in each round (e.g., by counting the number of police cars by the side of the road). The defender's payoff function is identical to table 2 — the defender's payoff is independent of the current state (e.g., rewards in this particular bounded memory game are not history-dependent). A k-adaptive regret minimization algorithm could be run without a priori knowledge of the adversary's utility, and will converge to the optimal fixed strategy against any k-adaptive adversary (e.g., any sequence of k-adaptive tourist strategies).

It is reasonable to use a k-adaptive strategy to model the behavior of our tourist drivers. Each tourist initially has no history of the defender's actions — during the first day of her visit a tourist must make the decision about whether or not to speed without any history of the defender's actions. After the first day the tourist may adapt his behavior based on previous outcomes. For example, a tourist might adopt the following k-adaptive strategy: $\mathcal{A}_1$ = "Play **DS** on the first day, and on the remaining $(k-1)$ days play **S** if the defender has never played **HI** previously, otherwise play **DS**." After k days the tourist leaves and a new tourist arrives. This new tourist may adopt a different k-adaptive strategy (e.g., $\mathcal{A}_2$ = "Play **S** on the first day, and on the remaining $(k-1)$ days play **S** if the defender has never played **HI** previously, otherwise play **DS**.").

We set the memory of our bounded memory game to be $m = k$. Now the fixed defender strategies F in our bounded memory game include strategies like f = "play **HI** every k'th round". Suppose for example that $k = 7$ and the defender plays f. In this case the sequence of rewards that the defender would see against the first k-adaptive adversary $\mathcal{A}_1$ would be $(0.7, 1, 1, 1, 1, 1, 1)$. The sequence of rewards that the defender would see against the second k-adaptive adversary $\mathcal{A}_2$ would be $(0.19, 1, 1, 1, 1, 1, 1)$. It is easy to verify that this is the optimal result for the defender — if the defender does not play **HI** on the first day then the 7-adaptive adversary will speed on day 2. A k-adaptive regret minimization algorithm could be run without a priori knowledge of the adversary's utility, and will converge to the optimal fixed strategy against any k-adaptive adversary (e.g., any sequence of k-adaptive tourist strategies).

Remark 1. A k-adaptive adversary is also an appropriate model for a temporary employee at the hospital so we could also consider the interaction between a hospital (defender) and a resident (adversary) at the hospital. The actions **S** and **DS**(e.g., "speed" and "don't speed") would be replaced with corresponding actions **B** and **V** (e.g., "behave" and "violate").

Unfortunateley, we are able to prove that there is no efficient k-adaptive regret minimization algorithm for general bounded memory games. However, our results do not rule out the posibility of an efficient γ-approximate k-adaptive regret minimization algorithm. Finding an efficient γ-approximate k-adaptive regret minimization algorithms is an important open problem.

6 Hardness Results

In this section, we show that unless NP = RP no oblivious regret minimization algorithm which uses the fixed strategies F as experts can be efficient in the imperfect information setting. In the full version [3] of this paper we explain how our hardness reduction can be adapted to prove that there is no efficient k-adaptive regret minimization algorithm in the perfect information setting for $k \geq 1$.

Specifically, we consider the subclass of bounded-memory games $\mathcal{G}$ with the following properties: $|\mathcal{O}| = O(1)$, $m = O(\log n)$, $|\mathcal{X}_A| = O(1)$, $|\mathcal{X}_D| = O(1)$ and imperfect information. Any $G \in \mathcal{G}$ is a game of imperfect information (on round t the defender observes O^t, but not a^t) with $O(n)$ states. Our goal is to prove the following theorem:

Theorem 1. *For any $\beta > 0$ and $\gamma < 1/8n^{\beta}$ there is no efficient γ-approximate oblivious regret minimization algorithm which uses the fixed strategies F as experts against oblivious adversaries for the class of imperfect information bounded-memory-m games unless* NP = RP.

Given a slightly stronger complexity-theoretic assumption called the randomized exponential time hypothesis [21] we can prove a slightly stronger hardness result. The randomized exponential time hypothesis says that no randomized algorithm running in time $2^{o(n)}$ can solve SAT.

Theorem 2. *Assume that the randomized exponential time hypothesis is true. Then for any $\gamma < 1/\left(8\log^2 n\right)$ there is no efficient γ-approximate oblivious regret minimization algorithm which uses the fixed strategies F as experts against oblivious adversaries for the class of imperfect information bounded-memory-m games.*

The proofs of Theorems 1 and 2 use the fact that it is hard to approximate MAX3SAT within any factor better than $\frac{7}{8}$ [22]. This means that unless NP = RP then for every constant $\beta > 0$ and every randomized algorithm S in RP, there exists a MAX3SAT instance ϕ such that the expected number of clauses in ϕ unsatisfied by $S(\phi)$ is $\geq \frac{1}{8} - \beta$ even though there exists an assignment satisfying $(1 - \beta)$ fraction of the clauses in ϕ.

We reduce a MAX3SAT formula ϕ with variables $x_1, ..., x_n$ and clauses $C_1, ..., C_\ell$ to a bounded-memory game G described formally below. We provide a high level overview of the game G before describing the details. The main idea is to construct G so that the rewards in G are related to the fraction of clauses of ϕ that are satisfied.

In G, for each variable x there is a state σ_x associated with that variable. The oblivious adversary controls the transitions between variables. This allows the oblivious adversary A_R to partition the game into stages of length n, such that during each stage the adversary causes the game to visit each variable exactly once (each state is associated with a variable). During each stage the adversary picks a clause C at random. In G we have $0, 1 \in \mathcal{X}_D$. Intuitively, the defender chooses assignment $x = 1$ by playing the action 1 while visiting the variable x. The defender receives a reward if and only if he succeeds in satisfying the clause C.

The game G is defined as follows:
Defender Actions: $\mathcal{X}_D = \{0, 1, 2\}$
Adversary Actions: $\mathcal{X}_A = \{0, 1\} \times \{0, 1, 2, 3\}$
Outcomes and States: Each round i produces two outcomes

$$\tilde{O}^i = \vec{a}^i[1] \quad \text{and} \quad \hat{O}^i = \begin{cases} 1 & \text{if } d^i = 2 \text{ or } d^i = a^i[2]; \\ 0 & \text{otherwise.} \end{cases}$$

Observe that these outcomes satisfy the independent outcomes requirement for bounded-memory games. There are $n = 2^{m+1}$ states, where σ^i is the state at round i, where

$$\sigma^i = \left(\langle \tilde{O}^{i-1}, \ldots, \tilde{O}^{i-m} \rangle, \hat{O}^{i-1} \right) .$$

Observe that each state encodes the last m outcomes $\tilde{O}$ and the last outcome $\hat{O}^i$. Intuitively, the last m outcomes $\tilde{O}^i$ are used to denote the variable x_i, while $\hat{O}^i$ is 1 if the defender has already received a reward during the current phase.

The defender actions $0, 1$ correspond to the truth assignments $0, 1$. The defender receives a reward for the correct assignment. The defender is punished if he attempts to obtain a reward in any phase after he has already received a reward in that phase. Once the defender has already received a reward he can play the special action 2 to avoid getting punished. The intuitive meaning of the adversary's actions is explained below.

If we ignore the outcome $\hat{O}$ then the states form a De Bruijn graph [23] where each node corresponds to a variable of ϕ. Notice that the adversary completely controls the outcomes $\tilde{O}$ with the first component of his action $\vec{a}[1]$. By playing a De Bruijn sequence $S = s_1...s_n$ the adversary can guarantee that we repeatedly take a Hamiltonian cycle over states(for an example see Figure 1).

Rewards:[2]

$$P\left(\sigma^i, d^i, a^i\right) = \begin{cases} -1 & \text{if } \hat{O}^{i-1} = 1 \text{ and } d^i \neq 2 \text{ and } \vec{a}^i[2] \neq 3; \\ 1 & \text{if } d^i \neq 2 \text{ and } d^i = \vec{a}^i[2] \text{ and } \hat{O}^{i-1} = 0; \\ 0 & \text{otherwise.} \end{cases}$$

[2] We use payoffs in the range $[-1, 1]$ for ease of presentation. These payoffs can easily be rescaled to lie in $[0, 1]$.

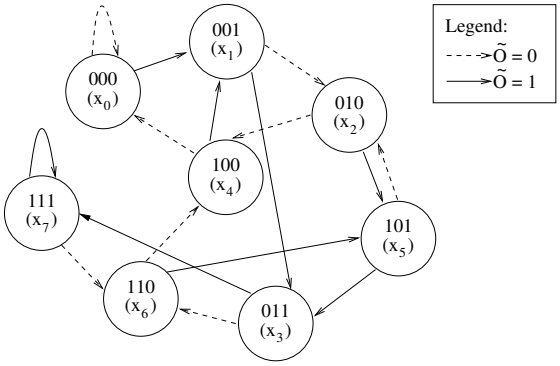

Figure 1. De Bruijn example

An intuitive interpretation of the reward function is presented in parallel with the adversary strategy.

Adversary Strategy: The first component of the adversary's action ($\vec{a}[1]$) controls the transitions between variables. The adversary will play the action $\vec{a}^i[2] = 1$ (resp. $\vec{a}^i[2] = 0$) whenever the corresponding variable assignment $x_i = 1$ (resp. $x_i = 0$) satisfies the clause that the adversary chose for the current phase.

If neither variable assignment satisfies the clause (if $x_i \notin C$ and $\bar{x}_i \notin C$) then the adversary plays $\vec{a}^i[2] = 2$. This ensures that a defender can only be rewarded during a round if he satisfies the clause C, which happens when $d^i = \vec{a}^i[2] = 0$ or 1.

Notice that whenever $\hat{O} = 1$ there is no way to receive a positive reward. The defender may want the game G to return to a state where $\hat{O} = 0$, but unless the adversary plays the special action $\vec{a}^i[2] = 3$ he is penalized when this happens. The adversary action $\vec{a}^i[2] = 3$ is a special 'reset phase' action. By playing $\vec{a}^i[2] = 3$ once at the end of each phase the adversary can ensure that the maximum payoff the defender receives during any phase is 1. See Figure 1 for a formal description of the adversary strategy.

• **Input:** MAX3SAT instance ϕ, with variables $x_1, \ldots, x_{n-1}$, and clauses $C_1, \ldots, C_\ell$. Random string $R \in \{0,1\}^*$

• **De Bruijn sequence:** $s_0, ..., s_{n-1}$

• **Round** t**:** Set $i \leftarrow t \mod n$.

1. **Select Clause:** If $i = 0$ then select a clause C uniformly at random from $C_1, ..., C_\ell$ using R.

2. **Select Move:**

$$a^i = \begin{cases} (s_i, 3) & \text{if } i = 0; \\ (s_i, 1) & \text{if } x_i \in C; \\ (s_i, 0) & \text{if } \bar{x}_i \in C; \\ (s_i, 2) & \text{otherwise.} \end{cases}$$

Figure 2. Oblivious Adversary: A_R

Analysis: At a high level, our hardness argument proceeds as follows:

1. If there is an assignment that satisfies $(1 - \beta)$ fraction of the clauses in ϕ, then there is a fixed strategy that performs well in expectation (see Claim 1).
2. If there a fixed strategy that performs well in expectation, then any γ-approximate oblivious regret minimization algorithm will perform well in expectation (see Claim 2).
3. If an efficiently computable strategy D performs well in expectation, then

there is an efficiently computable randomized algorithm S to approximate MAX3SAT (see Claim 3). This would imply that $\mathsf{NP} = \mathsf{RP}$.

Claim 1 *Suppose that there is a variable assignment that satisfies $(1-\beta)\cdot\ell$ of the clauses in ϕ. Then there is a fixed strategy f such that $E_R\left[\bar{P}\left(f, A_R, G, n\right)\right] \geq (1-\beta)/n$, where R is used to denote the random coin tosses of the oblivious adversary.*

Claim 2 *Suppose that D is an $\left(\frac{1}{8n} - \frac{3\beta}{n}\right)$-approximate oblivious regret minimization algorithm against the class of oblivious adversaries and there is a variable assignment that satisfies $(1-\beta)$ fraction of the clauses in ϕ. Then for $T = poly(n)$*

$$E_R\left[\bar{P}\left(D, A_R, G, T\right)\right] \geq \frac{7}{8n} + \frac{\beta}{n} ,$$

where R is used to denote the random coin tosses of the oblivious adversary.

Claim 3 *Fix a polynomial $p(\cdot)$ and let $\alpha = n \cdot E_R\left[\bar{P}\left(D, A_R, G, T\right)\right]$, where $T = p(n)$ and D is any polynomial time computable strategy. There is a polynomial time randomized algorithm S which satisfies α fraction of the clauses from ϕ in expectation.*

The proofs of these claims can be found in the full version [3] of this paper.

Proof of Theorem 1. The key point is that if an algorithm S runs in time $O\left(p(n)\right)$ on instances of size n^β for some polynomial $p(n)$ then on instances of size n S runs in time $O\left(p\left(n^{1/\beta}\right)\right)$ which is still polynomial time. Unless $\mathsf{NP} = \mathsf{RP}$ $\forall \epsilon, \beta > 0$ and every algorithm S running in time poly(n), there exists an integer n and a MAX3SAT formula ϕ with n^β variables such that

1. There is an assignment satisfying at least $(1-\epsilon)$ of the clauses in ϕ.
2. The expected fraction of clauses in ϕ satisfied by S is $\leq \frac{7}{8} + \epsilon$.

If we reduce from a MAX3SAT instance with n^β variables we can construct a game with $O(n)$ states ($n^{1-\beta}$ copies of each variable). One Hamiltonian cycle would now corresponds to $n^{1-\beta}$ phases of the game. This means that the expected average reward of the optimal fixed strategy is at least

$$\max_{f \in F} E_R\left[\bar{P}\left(f, A_R, G, T\right)\right] \geq \frac{n^{1-\beta}\left(1-\epsilon\right)}{n} ,$$

while the expected average reward of an efficient defender strategy D is at most

$$E_R\left[\bar{P}\left(D, A_R, G, T\right)\right] \leq \frac{n^{1-\beta}\left(\frac{7}{8} + \epsilon\right)}{n} .$$

Therefore, the expected average regret is at least

$$\bar{R}_0\left(D, A_R, G, T, F\right) \geq \left(\frac{1}{8} - 2\epsilon\right) n^{-\beta} .$$

□

The proof of theorem 2 is similar to the proof of theorem 1. It can be found in the full version [3] of this paper.

Our hardness reduction is similar to a result from Even-Dar, et al., [12]. They consider regret minimization in a Markov Decision Process where the adversary controls the transition model. Their game is not a bounded-memory game; in particular it does not satisfy our *independent outcomes* condition. The current state in their game can depend on the last n actions. In contrast, we consider bounded-memory games with $m = O(\log n)$, so that the current state only depends on the last m actions. This makes it much more challenging to enforce guarantees such as "the defender can only receive a reward once in each window of n rounds"—a property that is used in the hardness proof. The adversary is oblivious so she will not remember this fact, and the game itself cannot record whether a reward was given $m+1$ rounds ago. We circumvented this problem by designing a payoff function in which the defender is penalized for allowing the game to "forget" when the last reward was given, thus effectively enforcing the desired property.

7 Regret Minimization Algorithms

In section 7.1 we present a reduction from bounded-memory games to repeated games. This reduction can be used to create a k-adaptive regret minimizing algorithm (Theorem 3). This is significant because there is no k-adaptive regret minimization algorithm for the general class of stochastic games. A consequence of Theorem 1 is that when the expert set includes all fixed strategies F we cannot hope for an efficient algorithm unless $\mathsf{NP} = \mathsf{RP}$. In section 7.2 we present an efficient *approximate* 0-adaptive regret minimization algorithm for bounded-memory games of perfect information. The algorithm uses an implicit weight representation to efficiently sample the experts and update their weights. Finaly, we show how this algorithm can be adapted to obtain an efficient approximate 0-adaptive regret minimization algorithm for bounded-memory games of *imperfect* information.

7.1 Reduction to Repeated Games

All of our regret minimization algorithms work by first reducing the bounded-memory game G to a repeated game $\rho(G, K)$. One round of the repeated game $\rho(G, K)$ corresponds to K rounds of G. Before each round of $\rho(G, K)$ both players commit to an adaptive strategy. In $\rho(G, K)$ the reward that the defender gets for playing a strategy $f \in \mathcal{A}_D^K$ is the reward that the defender would have received for using the strategy f for the next K rounds of the actual game G if the initial state were σ_0: $P(f, g, \rho(G, K)) = P(f, g, G, \sigma_0, K)$.

The rewards in $\rho(G, K)$ may be different from the actual rewards in G because the initial state before each K rounds might not be σ_0. Claim 4 bounds the difference between the hypothetical losses from $\rho(G, K)$ and actual losses in G using the bounded-memory property. The proof of Claim 4 is in the full version of this paper [3].

Claim 4 *For any adaptive defender strategy $f \in \mathcal{A}_D^K$ and any adaptive adversary strategy $g \in \mathcal{A}_A^K$ and any state σ of G we have $|P(f, g, G, \sigma, K) - P(f, g, G, \sigma_0, K)| \leq m$.*

The key idea behind our k-adaptive regret minimization algorithm BW is to reduce the original bounded-memory game to a repeated game $\rho(G, K)$ of imperfect information ($K \equiv 0 \mod k$). In particular we obtain the regret bound in Theorem 3. Details and proofs can be found in the full version of this paper [3].

Theorem 3. *Let G be any bounded-memory-m game with n states and let A be any adversary strategy. After playing T rounds of G against A, BW (G, K) achieves regret bound*

$$\bar{R}_k(\mathsf{BW}, A, G, T, S) < \frac{m}{T^{1/4}} + 4\frac{\sqrt{N \log N}}{T^{1/4}} ,$$

where $N = |S|$ is the number of experts, A is the adversary strategy and K has been chosen so that $K = T^{1/4}$ and $K \equiv 0 \mod k$.

Intuitively, the $m/T^{1/4} = m/K$ term is due to modeling loss from Claim 4 and the other term comes from the standard regret bound of [24].

7.2 Efficient Approximate Regret Minimization Algorithms

In this section we present EXBW (Efficient approXimate Bounded Memory Weighted Majority), an efficient algorithm to approximately minimize regret against an oblivious adversary in bounded-memory games with perfect information. The set of experts $\mathcal{E}$ used by our algorithms contains the fixed strategies F as well as all K-adaptive strategies $\mathcal{A}_D^K$ ($K = m/\gamma$). We prove the following theorem

Theorem 4. *Let G be any bounded-memory-m game of perfect information with n states and let A be any adversary strategy. Playing T rounds of G against A, EXBW runs in total time $Tn^{O(1/\gamma)}$ and achieves regret bound*

$$\bar{R}_0(\mathsf{EXBW}, A, G, T, \mathcal{E}) \leq \gamma + O\left(\frac{m}{\gamma}\sqrt{\frac{\frac{m}{\gamma} n \log(N)}{T}}\right) ,$$

where K has been set to m/γ and $N = \left|\mathcal{A}_D^K\right| = (|\mathcal{X}_D|)^{n^{1/\gamma}}$ is the number of K-adaptive strategies.

In particular, for any constant γ there is an efficient γ-approximate 0-adaptive regret minimization algorithm for bounded-memory games of perfect information. We can adapt this algorithm to get EXBWII (Efficient approXimate Bounded Memory Weighted Majority for Imperfect Information Games), an efficient approximate 0-adaptive regret minimization algorithm for games of imperfect information using a sampling strategy described in the full version of this paper [3].

Theorem 5. *Let G be any bounded-memory-m game of imperfect information with n states and let A be any adversary strategy. There is an algorithm* EXBWII *that runs in total time $Tn^{O(1/\gamma)}$ playing T rounds of G against A, and achieves regret bound*

$$\bar{R}_0\left(\text{EXBWII}, A, G, T, \mathcal{E}\right) \leq 2\gamma + O\left(\frac{mn^{1/\gamma}}{\gamma^2}\sqrt{\frac{\frac{mn^{1/\gamma}}{\gamma} n \log\left(N\right)}{T}}\right) .$$

where K has been set to m/γ and $N = \left|\mathcal{A}_D^K\right| = \left(\left|\mathcal{X}_D\right|\right)^{n^{1/\gamma}}$ is the number of K-adaptive strategies.

The regret bound of Theorem 4 is simply the regret bound achieved by the standard weighted majority algorithm [25] plus the modeling loss term from Claim 4. The main challenge is to provide an efficient simulation of the weighted majority algorithm. There are an exponential number of experts so no efficient algorithm can explicitly maintain weights for each of these experts. To simulate the weighted majority algorithm EXBW implicitly maintains the weight of each expert.

To simulate the weighted majority algorithm we must be able to *efficiently sample* from our weighted set of experts (see $\mathbf{Sample}\left(\mathcal{E}\right)$) and efficiently update the weights of each expert in the set after each round of $\rho\left(G, K\right)$ (see update weight stage of EXBW).

Meet the Experts Instead of using F as the set of experts, EXBW uses a larger set of experts $\mathcal{E}$ ($F \subset \mathcal{E}$). Recall that a K-adaptive strategy is a function f mapping the K most recent outcomes H_K to actions. We use a set of K-adaptive strategies $E = \left\{f_\sigma : \sigma \in \Sigma\right\} \subset \mathcal{A}_D^K$ to define an expert E in $\rho\left(G, K\right)$: if the current state of the real bounded-memory game G is σ then E uses the K-adaptive strategy f_σ in the next round of $\rho\left(G, K\right)$ (i.e., the next K rounds of G). $\mathcal{E}$ denotes the set of all such experts.

Maintaining Weights for Experts Implicitly To implicitly maintain the weights of each expert $E \in \mathcal{E}$ we use the concept of a game trace. We say that a game trace $p = \sigma, d^1, O^1, ..., d^{i-1}, O^{i-1}, d^i$ is consistent with an expert E if $f_\sigma\left(O^1, ..., O^{j-1}\right) = d^j$ for each j. We define the set $\mathcal{C}\left(E\right)$ to be the set of all such consistent traces of maximum length K and $\mathcal{C} = \bigcup_{E \in \mathcal{E}} \mathcal{C}\left(E\right)$ denotes the set of all traces consistent with some expert $E \in \mathcal{E}$. EXBW maintains a weight w_p on each trace $p \in \mathcal{C}$. The weight of an expert E is then defined to be $W_E = \prod_{p \in \mathcal{C}(E)} w_p$.

Given adversary actions $\vec{a} = a_1, ..., a_K$ and a trace $p = \sigma, d^1, O^1, ..., d^{i-1}, O^{i-1}, d^i$ we define $\mathcal{R}\left(\vec{a}, \sigma', p\right)$.

$$\mathcal{R}\left(\vec{a}, \sigma', p\right) = \begin{cases} 0 & \text{if } \sigma \neq \sigma'; \\ \prod_{j<i} \Pr\left[O^j \,\middle|\, a^j, d^j\right] & \text{otherwise;} \end{cases}$$

Intuitively, $\mathcal{R}\left(\vec{a}, \sigma', p\right)$ is the probability that each outcome of p would have occurred given the adversary actions were $\vec{a}$ and the initial state was σ'. We use $\ell\left(p, \vec{a}, \sigma'\right)$ to denote the payment that the defender received for playing d^i (the last action in p). Formally $\ell\left(p, \vec{a}, \sigma'\right) = P\left(\sigma_p^f, d^i, a^i\right) \mathcal{R}\left(\vec{a}, \sigma', p\right)$, where σ_p^f denotes the state reached following the trace p (after observing outcomes $O^1, ..., O^{i-1}$ starting from σ_0) and d^i is the final defender

action in the trace. Notice that in the imperfect information setting the defender could not compute ℓ because he would not observe the adversary's actions $\vec{a}$.

Updating Weights Efficiently While updating weights EXBW maintains the invariant that $w_p = \beta^{\sum_{j=1}^{T/K} \ell\left(p,\vec{a}^j,\sigma^{jK}\right)}$, where σ^{jK} is the state of G after jK rounds and $\vec{a}^t$ is the actions the adversary played during the j'th round of $\rho\left(G,K\right)$. The standard weighted majority algorithm maintains the invariant that $W_E = \beta^{\sum_{j=1}^{T/K} P\left(E,\vec{a}^t,\rho(G,K)\right)}$. Claim 5 implies that EXBW also maintains this invariant with its implicit weight representation — the proof of Claim 5 is in the full version of this paper [3].

Claim 5

$$\prod_{p\in\mathcal{C}(E)} \beta^{\sum_{j=1}^{T/K} \ell\left(p,\vec{a}^j,\sigma^{jK}\right)} = \beta^{\sum_{j=1}^{T/K} P\left(E,\vec{a}^j,\rho(G,K)\right)} .$$

Sampling Experts Efficiently We can also efficiently sample from $\mathcal{E}$ using dynamic programming (see $\mathbf{Sample}\left(\mathcal{E}\right)$). Using the notation $p \sqsubset p'$ for p' extends p we can define $\hat{w}_p$. Intuitively, $\hat{w}_{p;O;d}$ represents the weight of the action d from history $p;O$. Using dynamic programming we can efficiently compute $\hat{w}_p$ for each trace p because there are only $n^{O(1/\gamma)}$ such traces. Using the weights $\hat{w}_p$ we can efficiently sample from $\mathcal{E}$. We use $p;O;d$ to denote a new game trace which contains all of the outcomes/actions in p appended with O and d.

$$\hat{w}_p = \sum_{E:p\in\mathcal{C}(E)} \prod_{p'\in\mathcal{C}(E)\wedge p\sqsubset p'} w_{p'}$$

Algorithm: EXBW $\left(\gamma, G\right)$

- **Initialize:** $K = m/\gamma$
- **Construct:** $\rho\left(G,K\right)$
- **Each Round:**

1. $\sigma \leftarrow G.CurrentState$
2. $E \leftarrow \mathbf{Sample}\left(\mathcal{E}\right)$
3. Play E
4. Observe adversary actions

$$\vec{a} = a^1, ..., a^K .$$

5. **Update Weights:** For each $p \in \mathcal{C}$

A. Compute $\ell\left(p,\vec{a},\sigma\right)$

B. Set $w_p \leftarrow w_p \times \beta^{\ell(p,\vec{a},\sigma)}$.

Algorithm: $\mathbf{Sample}\left(\mathcal{E}\right)$

- For each trace $p \in \mathcal{C}$ recursively compute $\hat{w}_p$ using the formula:

$$\hat{w}_p = \sum_{O\in\mathcal{O}} \sum_{d\in\mathcal{X}_D} \beta^{\sum_{t=1}^{T} \ell\left(p;O;d,\vec{a}^t,\sigma^{Kt}\right)} \hat{w}_{p;O;d} .$$

- **Build Strategy** E: For each $p \in \mathcal{C}$ and $O \in \mathcal{O}$, randomly select $d \in \mathcal{X}_D$

$$\Pr\left[d \,|\, p,\ O\right] = \frac{\hat{w}_{p;O;d}}{\sum_{d'\in\mathcal{X}_D} \hat{w}_{p;O;d'}} .$$

- E play d any time it observes history $p;O$.

Claim 6 says that $\mathbf{Sample}\left(\mathcal{E}\right)$ outputs each expert E with probability proportional to W_E.

Claim 6 *For each expert $E \in \mathcal{E}$ Algorithm $\mathbf{Sample}\left(\mathcal{E}\right)$ outputs E with probability*

$$\Pr\left[E\right] \propto W_E .$$

Given $\mathbf{Sample}\left(\mathcal{E}\right)$ it is straightforward to simulate the standard weighted majority algorithm. To update weights EXBW simply loops through all traces $p \in \mathcal{C}$ applying the update rule $w_p = w_p \times \beta^{\ell\left(p,\vec{a}^t,\sigma^{tK}\right)}$, where β is a learning parameter we tune later.

The formal proof of Theorem 4 can be found in the full version along with the proof of claim 6.

At a high level our algorithm is similar to the online shortest path algorithm developed by Takimoto and Warmuth [15]. In their work, they consider the set of all source-destination paths in a graph as experts. Since there are exponentially many paths they also maintain the weights of the experts implicitly. In their setting, the defender completely controls the chosen path. In contrast, our experts adapt to adversary actions. The challenge was constructing a new implicit weight representation which works for K-adaptive strategies.

Using this implicit weight representation we could have also used the general barycentric spanner approach to online linear optimization developed by Awerbuch and Kleinberg [16] to design a γ-approximate 0-adaptive regret minimization algorithm running in time $n^{O(1/\gamma)}$. However, we are able to achieve better regret bounds in theorem 4 by simulating the weighted majority algorithm. Awerbuch and Kleinberg [16, Theorem 2.8] achieve the average regret bound $O\left(Md^{5/3}/T^{1/3}\right)$, where d is the dimension of the problem space and M is a bound on the cost vectors. By comparison our regret bounds in Theorems 4 and 5 tend to 0 with $1/\sqrt{T}$. In our setting, the dimension of the problem space is $d = O\left(n^{(1/\gamma)}\right)$ (the number of nodes in the decision tree), and $M = K = m/\gamma$ is the upper bound on the cost vector in each round of $\rho\left(G, K\right)$. The average regret bound would be $O\left(\frac{m}{\gamma} n^{5/(3\gamma)}/T^{1/3}\right)$. the regret bound is proportional to $\sqrt{n^{1/\gamma}/T}$. By comparison Theorem 4 has a $\sqrt{n^{1/\gamma}}$ in the numerator.

The standard regret minimization trick for dealing with imperfect information in a repeated game is to break the game up into phases and perform random sampling in each round to estimate the cost of each expert and update weights. The challenge in adapting EXBW is that there are exponentially many experts in $\mathcal{E}$. Our key idea was to estimate $\ell\left(p, \vec{a}, \sigma\right)$ for each $p \in \mathcal{C}$ so there are only $n^{O(1/\gamma)}$ samples to take in each phase. We can then update the implicit weight representation using the estimated values $\ell\left(p, \vec{a}, \sigma\right)$.

8 Open Questions

In this paper, we defined a new class of games called bounded-memory games, introduced several new notions of regret, and presented hardness results and algorithms for regret minimization in this subclass of stochastic games. Because both the games and the notions of regret we study in this paper rely on novel definitions, they raise a number of interesting open problems: (1) To what extent can the hardness results of Theorems 1 and 2 be further improved? ($\gamma = 1/\log n$?) Could similar hardness results apply to games with perfect information? (2) Is there an efficient *non-approximate* oblivious regret minimization algorithm for bounded-memory games with perfect information? (3) Is there a γ-approximate oblivious regret minimization algorithm with running time $n^{o(1/\gamma)}$? For example, could one design a γ-approximate oblivious regret minimization algorithm with running time $n^{-\log \gamma}$? (4) For repeated games ($m = 0$) is there an efficient γ-approximate k-adaptive regret minimization algorithm if we use $\mathcal{A}_D^K$ as our set of experts ($K = \log n$)?

References

1. Blum, A., Mansour, Y.: Learning, regret minimization, and equilibria. Algorithmic Game Theory (2007) 79–102
2. Shapley, L.: Stochastic games. Proceedings of the National Academy of Sciences of the United States of America **39**(10) (1953) 1095
3. Blocki, J., Christin, N., Datta, A., Sinha, A.: Adaptive regret minimization in bounded-memory games. CoRR **abs/1111.2888** (2011)
4. Blocki, J., Christin, N., Datta, A., Sinha, A.: Regret minimizing audits: A learning-theoretic basis for privacy protection. In: Computer Security Foundations Symposium, 2011. CSF'11. 24th IEEE, IEEE (2011) 312–327
5. Blocki, J., Christin, N., Datta, A., Sinha, A.: Audit mechanisms for provable risk management and accountable data governance. In: GameSec. (2012)
6. Von StackelberG, H.: Market structure and equilibrium. Springer (2011)
7. Blocki, J., Christin, N., Datta, A., Procaccia, A.D., Sinha, A.: Audit games. In: IJCAI. (2013)
8. Tambe, M.: Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned. Cambridge University Press (2011)
9. Mertens, J., Neyman, A.: Stochastic games. International Journal of Game Theory **10**(2) (1981) 53–66
10. Papadimitriou, C., Tsitsiklis, J.: The complexity of optimal queueing network control (1999)
11. Golovin, D., Krause, A.: Adaptive submodularity: A new approach to active learning and stochastic optimization. CoRR **abs/1003.3967** (2010)
12. Even-Dar, E., Kakade, S., Mansour, Y.: Experts in a Markov decision process. In: Advances in neural information processing systems 17: proceedings of the 2004 conference, The MIT Press (2005) 401
13. Mannor, S., Shimkin, N.: The empirical bayes envelope and regret minimization in competitive markov decision processes. Mathematics of Operations Research (2003) 327–345
14. Even-Dar, E., Mannor, S., Mansour, Y.: Learning with global cost in stochastic environments. In: COLT: Proceedings of the Workshop on Computational Learning Theory. (2010)
15. Takimoto, E., Warmuth, M.: Path kernels and multiplicative updates. The Journal of Machine Learning Research **4** (2003) 773–818
16. Awerbuch, B., Kleinberg, R.: Online linear optimization and adaptive routing. Journal of Computer and System Sciences **74**(1) (2008) 97–114
17. Farias, D.P.D., Megiddo, N.: Combining expert advice in reactive environments. J. ACM **53** (September 2006) 762–799
18. Fudenberg, D., Tirole, J.: Game theory. MIT Press (1991)
19. Blum, A., Mansour, Y.: From external to internal regret. Learning Theory (2005) 621–636
20. Celentani, M., Fudenberg, D., Levine, D., Pesendorfer, W.: Maintaining a reputation against a patient opponent. Econometrica **64** (1996) 691–704
21. Impagliazzo, R., Paturi, R.: On the complexity of k-sat. Journal of Computer and System Sciences **62**(2) (2001) 367–375
22. Hastad, J.: Some optimal inapproximability results. Journal of the ACM (JACM) **48**(4) (2001) 798–859
23. Good, I.J.: Normal recurring decimals. Journal of the London Mathematical Society **1**(3) (1946) 167
24. Auer, P., Cesa-Bianchi, N., Freund, Y., Schapire, R.: Gambling in a rigged casino: The adversarial multi-armed bandit problem. In: FOCS, Published by the IEEE Computer Society (1995) 322
25. Littlestone, N., Warmuth, M.: The weighted majority algorithm. In: Proceedings of FOCS. (1989) 256–261