From External to Internal Regret

Avrim Blum* AVRIM@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University,
Pittsburgh, PA 15213.

Yishay Mansour† MANSOUR@CS.TAU.AC.IL
School of Computer Science,
Tel Aviv University,
Tel Aviv, ISRAEL.

Editor: Ron Meir

Abstract

External regret compares the performance of an online algorithm, selecting among N actions, to the performance of the best of those actions in hindsight. Internal regret compares the loss of an online algorithm to the loss of a modified online algorithm, which consistently replaces one action by another.

In this paper we give a simple generic reduction that, given an algorithm for the external regret problem, converts it to an efficient online algorithm for the internal regret problem. We provide methods that work both in the *full information* model, in which the loss of every action is observed at each time step, and the *partial information* (bandit) model, where at each time step only the loss of the selected action is observed. The importance of internal regret in game theory is due to the fact that in a general game, if each player has sublinear internal regret, then the empirical frequencies converge to a correlated equilibrium.

For external regret we also derive a quantitative regret bound for a very general setting of regret, which includes an arbitrary set of modification rules (that possibly modify the online algorithm) and an arbitrary set of time selection functions (each giving different weight to each time step). The regret for a given time selection and modification rule is the difference between the cost of the online algorithm and the cost of the modified online algorithm, where the costs are weighted by the time selection function. This can be viewed as a generalization of the previously-studied *sleeping experts* setting.

1. Introduction

The motivation behind regret analysis might be viewed as the following: we design a sophisticated online algorithm that deals with various issues of uncertainty and decision making, and sell it to a client. Our online algorithm runs for some time and incurs a certain loss. We would like to avoid the embarrassment that our client will come back to us and claim that in *retrospect* we could have incurred a much lower loss if we used his simple alternative policy

*. This work was supported in part by NSF grants CCR-0105488 and IIS-0312814.
†. The work was done while the author was a fellow in the Institute of Advance studies, Hebrew University. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778, by a grant no. 1079/04 from the Israel Science Foundation and an IBM faculty award. This publication only reflects the authors' views.

π. The regret of our online algorithm is the difference between the loss of our algorithm and the loss using π. Different notions of regret quantify differently what is considered to be a "simple" alternative policy.

At a high level one can split alternative policies into two categories. The first consists of alternative policies that are independent from the online algorithm's action selection, as is done in *external regret*. External regret, also called the *best expert* problem, compares the online algorithm's cost to the best of N actions in retrospect (see Hannan (1957); Foster and Vohra (1993); Littlestone and Warmuth (1994); Freund and Schapire (1995, 1999); Cesa-Bianchi et al. (1997)). This implies that the simple alternative policy performs the same action in all time steps, which indeed is quite simple. Nonetheless, one important application of external regret is a general methodology for developing online algorithms whose performance matches that of an optimal static offline algorithm, by modeling the possible static solutions as different actions.

The second category of alternative policies are those that consider the online sequence of actions and suggest a simple modification to it, such as "every time you bought IBM, you should have bought Microsoft instead." This notion is captured by *internal regret*, introduced in Foster and Vohra (1998). Specifically, internal regret allows one to modify the online action sequence by changing every occurrence of a given action i to an alternative action j. Specific low internal regret algorithms were derived by Hart and Mas-Colell (2000), Foster and Vohra (1997, 1998, 1999), and Cesa-Bianchi and Lugosi (2003), where the use of the approachability theorem of Blackwell (1956) has played an important role in some of the algorithms.

One of the main contributions of our work is to show a simple online way to efficiently convert any low external regret algorithm into a low *internal* regret algorithm. Our guarantee is somewhat stronger than internal regret and we call it *swap regret*, which allows one to *simultaneously* swap multiple pairs of actions. (If there are N actions total, then swap-regret is bounded by N times the internal regret.) Using known results for external regret we can derive a swap regret bound of $O(\sqrt{TN \log N})$, where T is the number of time steps, which is the best known bound on swap regret for efficient algorithms. We also show an $\Omega(\sqrt{TN})$ lower bound for the case of randomized online algorithms against an adaptive adversary.

The importance of internal and swap regret is due to their tight connection to correlated equilibria, introduced by Aumann (1974). For a general-sum game of any finite number of players, a distribution Q over the joint action space is a correlated equilibrium if every player would have zero internal regret when playing it. In a repeated game scenario, if each player uses an action selection algorithm whose regret of this form is sublinear in T, then the empirical distribution of the players actions converges to a correlated equilibrium (see, e.g., Hart and Mas-Colell (2000)), and in fact, the benefit of a deviation from a correlated equilibrium is bounded exactly by R/T, where R is the largest swap regret of any of the players.

We also extend our results to the *partial information model*, also called the *adversarial multi-armed bandit* (MAB) problem in Auer et al. (2002b). In this model, the online algorithm only gets to observe the loss of the action actually selected, and does not see the losses of the actions not chosen. For example, if you are driving to work and need to select which of several routes to take, you only observe the travel time on the route actually taken.

If we view this as an online problem, each day selecting which route to take on that day, then this fits the MAB setting. Furthermore, the route-choosing problem can be viewed as a general-sum game: your travel time depends on the choices of the other drivers as well. Thus, if *every* driver uses a low internal-regret algorithm, then the uniform distribution over observed traffic patterns will converge to a correlated equilibrium. For the MAB problem, our combining algorithm requires additional assumptions on the base external-regret MAB algorithm: a smoothness in behavior when the actions played are taken from a somewhat different distribution than the one proposed by the algorithm. Luckily, these conditions are satisfied by existing external-regret MAB algorithms such as that of Auer et al. (2002b). For the multi-armed bandit setting, we derive an $O(N\sqrt{NT \log N})$ swap-regret bound. Thus, after $T = O(\frac{1}{\epsilon^2}N^3 \log N)$ rounds, the empirical distribution on the history is an ϵ-correlated equilibrium. In a recent work, Stoltz (2005) gives an improved swap regret bound of $O(N\sqrt{T \log N})$. (The work of Hart and Mas-Colell (2001) also gives a multi-armed bandit algorithm whose swap regret is sublinear in T, but does not derive explicit bounds.)

One can also envision broader classes of regret. Lehrer (2003) defines a notion of *wide range regret* that allows for arbitrary action-modification rules, which might depend on history, and also Boolean time selection functions (that determine which subset of times is relevant). Using the approachability theorem, he shows a scheme that in the limit achieves no regret (i.e., regret is sublinear in T). While Lehrer (2003) derives the regret bounds in the limit, we derive finite-time regret bounds for this setting. We show that for any family of N actions, M time selection functions and K modification rules, the maximum regret with respect to any selection function and modification rule is bounded by $O(\sqrt{TN \log(MK)})$. Our model also handles the case where the time selection functions are not Boolean, but rather real valued in $[0, 1]$.

This latter result can be viewed as a generalization of the *sleeping experts* setting of Freund et al. (1997) and Blum (1997). In the sleeping experts problem, we again have a set of experts, but on any given time step, each expert may be awake (making a prediction) or asleep (not predicting). This is a natural model for combining a collection of if-then rules that only make predictions when the "if" portion of the rule is satisfied, and this setting has had application in domains ranging from managing a calendar (Blum, 1997) and text-categorization (Cohen and Singer, 1999) to learning how to formulate web search-engine queries (Cohen and Singer, 1996). By converting each such sleeping-expert into a pair $\langle$expert, time-selection function$\rangle$, we achieve the desired guarantee that for each sleeping-expert, our loss *during the time that expert was awake* is not much more than its loss in that period. Moreover, by using non-Boolean time-selection functions, we can naturally handle prediction rules that have varying degrees of confidence in their predictions and achieve a confidence-weighted notion of regret.

We also study the case of deterministic Boolean prediction in the setting of time selection functions. We derive a deterministic online algorithm whose number of weighted errors, with respect to any time selection function from our class of M selection functions, is at most $3OPT + 2 + 2\log_2 M$, where OPT is the cumulative loss of the best constant prediction for that time selection function.

Related work. A different conversion procedure from external to internal regret was given independently by Stoltz and Lugosi (2005), yet the approach there is very different from the one developed here. Further results regarding the relation between external and internal regret appear in Stoltz and Lugosi (2007) and for the multi-armed bandit setting in Cesa-Bianchi et al. (2006). In comparison to Stoltz and Lugosi (2007), we are able to achieve a better swap regret guarantee in polynomial time. (A straightforward application of Stoltz and Lugosi (2007) to swap regret would require time-complexity $\Omega(N^N)$; alternatively, they can achieve a good internal-regret bound in polynomial time, but then their swap regret bound becomes worse by a factor of $\sqrt{N}$.) On the other hand, their work is applicable to any convex loss function while our work is restricted to linear loss functions. See also Stoltz (2005), Section 1.3, for a discussion of the different procedures.

2. Model and Preliminaries

We assume an adversarial online model where there are N available actions $\{1, \ldots, N\}$. At each time step t, an online algorithm H selects a distribution p^t over the N actions. After that, the adversary selects a loss vector $\ell^t \in [0,1]^N$, where $\ell_i^t \in [0,1]$ is the loss of the i-th action at time t. In the *full information model*, the online algorithm receives the loss vector ℓ^t and experiences a loss $\ell_H^t = \sum_{i=1}^N p_i^t \ell_i^t$. In the *partial information model*, the online algorithm receives $(\ell_{k^t}^t, k^t)$, where k^t is distributed according to p^t, and $\ell_H^t = \ell_{k^t}^t$ is its loss. The loss of the i-th action during the first T time steps is $L_i^T = \sum_{t=1}^T \ell_i^t$, and the loss of H is $L_H^T = \sum_{t=1}^T \ell_H^t$. The aim for the external regret setting is to design an online algorithm that will be able to approach the best action, namely, to have a loss close to $L_{min}^T = \min_i L_i^T$. Formally we would like to minimize the external regret $R = L_H^T - L_{min}^T$.

We introduce a notion of a *time selection* function. A time selection function I is a function over the time steps mapping each time step to $[0,1]$. That is, $I : \{1, \ldots, T\} \to [0,1]$. The loss of action j using time-selector I is $L_{j,I}^T = \sum_t I(t)\ell_j^t$. Similarly we define $L_{H,I}$, the loss of the online algorithm H with respect to time selection function I, as $L_{H,I}^T = \sum_t I(t)\ell_H^t$, where ℓ_H^t is the loss of H at time t.[1] This notion of experts with time selection is very similar to the notion of "sleeping experts" studied in Freund et al. (1997). Specifically, for each action j and time selection function I, one can view the pair (j, I) as an expert that is "awake" when $I(t) = 1$ and "asleep" when $I(t) = 0$ (and we could view it as "partially awake" when $I(t) \in (0,1)$).

We also consider modification rules that modify the actions selected by the online algorithm, producing an alternative strategy we will want to compete against. A *modification rule* F has as input the history and an action choice and outputs a (possibly different) action. (We denote by F^t the function F at time t, including any dependency on the history.) Given a sequence of probability distributions p^t used by an online algorithm H, and a modification rule F, we define a new sequence of probability distributions $f^t = F^t(p^t)$, where $f_i^t = \sum_{j:F^t(j)=i} p_j^t$. The loss of the modified sequence is $L_{H,F} = \sum_t \sum_i f_i^t \ell_i^t$. Similarly, given a time selection function I and a modification rule F we define $L_{H,I,F} = \sum_t \sum_i I(t) f_i^t \ell_i^t$.

1. We can let the time selector depend on the history up to time t, rather than the time t itself, and all the results presented would be the same.

In our setting we assume a finite class of N actions, $\{1, \ldots, N\}$, a finite set $\mathcal{F}$ of K modification rules, and a finite set $\mathcal{I}$ of M time selection function. Given a sequence of loss vectors, the regret of an online algorithm H with respect to the N actions, the K modification rules, and the M time selection functions, is

$$R_H^{\mathcal{I},\mathcal{F}} = \max_{I \in \mathcal{I}} \max_{F \in \mathcal{F}} \{L_{H,I} - L_{H,I,F}\}.$$

Note that the external regret setting is equivalent to having a single time-selection function ($I(t) = 1$ for all t) and a set $\mathcal{F}^{ex}$ of N modification rules F_i, where F_i always outputs action i. For internal regret, the set $\mathcal{F}^{in}$ consists of $N(N-1)$ modification rules $F_{i,j}$, where $F_{i,j}(i) = j$ and $F_{i,j}(i') = i'$ for $i' \neq i$, plus the identity function. That is, the internal regret of H is

$$\max_{F \in \mathcal{F}^{in}} \{L_H - L_{H,F}\} \;=\; \max_{i,j} \sum_t p_i^t(\ell_i^t - \ell_j^t).$$

We also define an extension to internal regret that we call *swap regret*. This case has $\mathcal{F}^{sw}$ include all N^N functions $F : \{1, \ldots, N\} \to \{1, \ldots, N\}$, where the function F swaps the current online action i with $F(i)$ (which can be the same or a different action).[2]

A few simple relationships between the different types of regret: since $\mathcal{F}^{ex} \subseteq \mathcal{F}^{sw}$ and $\mathcal{F}^{in} \subseteq \mathcal{F}^{sw}$, both external and internal regret are upper-bounded by swap-regret. Also, swap-regret is at most N times larger than internal regret. On the other hand, even with $N = 3$, there are simple examples that separate internal and external regret (see, e.g., Stoltz and Lugosi (2005)).

Correlated Equilibria and Swap Regret

We briefly sketch the relationship between correlated equilibria and swap regret.

Definition 1 *A game $G = \langle M, (A_i), (s_i) \rangle$ has a finite set M of m players. Player i has a set A_i of N actions and a loss function $s_i : A_i \times (\times_{j \neq i} A_j) \to [0,1]$ that maps the action of player i and the actions of the other players to a real number. (We have scaled losses to $[0,1]$.)*

The aim of each player is to minimize its loss. A correlated equilibrium is a distribution P over the joint action space with the following property. Imagine a correlating device draws a vector of actions $\vec{a}$ using distribution P over $\times A_i$, and gives player i the action a_i from $\vec{a}$. (Player i is not given any other information regarding $\vec{a}$.) The probability distribution P is a correlated equilibrium if, for each player, it is its best response to play the suggested action (provided that the other players do not deviate).

We now define an ϵ-correlated equilibrium.

Definition 2 *A joint probability distribution P over $\times A_i$ is an ϵ-correlated equilibrium if for every player j and for any function $F : A_j \to A_j$, we have $E_{a \sim P}[s_j(a_j, a^{-j})] \leq E_{a \sim P}[s_j(F(a_j), a^{-j})] + \epsilon$, where a^{-j} denotes the joint actions of the other players.*

2. Note that in swap and external regret, the modification functions do not depend on history. In Section 7 we consider general modification functions.

In other words, P is an ϵ-correlated equilibrium if the expected incentive to deviate is at most ϵ for every player.

The following theorem relates the empirical distribution of the actions performed by each player, their swap regret, and the distance from a correlated equilibrium (see also, Foster and Vohra (1997, 1998) and Hart and Mas-Colell (2000)).

Theorem 3 *Let $G = \langle M, (A_i), (s_i) \rangle$ be a game and assume that for T time steps each player follows a strategy that has swap regret of at most $R(T, N)$. The empirical distribution Q of the joint actions played by the players is an $(R(T, N)/T)$-correlated equilibrium, and the loss of each player equals, by definition, its expected loss on Q.*

The above states that the payoff of each player is its payoff in some approximate correlated equilibrium. In addition, it relates the swap regret to the distance from a correlated equilibrium. Note that if the average swap regret vanishes then the procedure converges, in the limit, to the set of correlated equilibria (see Hart and Mas-Colell (2000) and Foster and Vohra (1997, 1999)).

3. Generic reduction from external to swap regret

We now give a black-box reduction showing how any algorithm A achieving good external regret can be used as a subroutine to achieve good swap regret as well. The high-level idea is as follows. We will instantiate N copies of the external-regret algorithm. At each time step, these algorithms will each give us a probability vector, which we will combine in a particular way to produce our own probability vector p. When we receive a loss vector ℓ, we will partition it among the N algorithms, giving algorithm A_i a fraction p_i (p_i is our probability mass on action i), so that A_i's belief about the loss of action j is $\sum_t p_i^t \ell_j^t$, and matches the cost we would incur putting i's probability mass on j. In the proof, algorithm A_i will in some sense be responsible for ensuring low regret of the $F_{i,j}$ variety. The key to making this work is that we will be able to define the p's so that the sum of the losses of the algorithms A_i on their own loss vectors matches our overall true loss.

To be specific, let us formalize what we mean by an external regret algorithm.

Definition 4 *An algorithm A has external regret $R(L_{min}, T, N)$ if for any sequence of T losses ℓ^t such that some action has total loss at most L_{min}, for any action $j \in \{1, \ldots, N\}$ we have*

$$L_A^T = \sum_{t=1}^{T} \ell_A^t \leq \sum_{t=1}^{T} \ell_j^t + R(L_{min}, T, N) = L_j^T + R(L_{min}, T, N) .$$

We assume we have N algorithms A_i (which could all be identical or different) such that A_i has external regret $R_i(L_{min}, T, N)$. We combine the N algorithms as follows. At each time step t, each algorithm A_i outputs a distribution q_i^t, where $q_{i,j}^t$ is the fraction it assigns action j. We compute a vector p such that $p_j^t = \sum_i p_i^t q_{i,j}^t$. That is, $p = pQ$, where p is the row-vector of our probabilities and Q is the matrix of $q_{i,j}$. (We can view p as a stationary distribution of the Markov Process defined by Q, and it is well known such a p exists and is efficiently computable.) For intuition into this choice of p, notice that it implies we can consider action selection in two equivalent ways. The first is simply using the distribution p

to select action j with probability p_j. The second is to select algorithm A_i with probability p_i and then to use algorithm A_i to select the action (which produces distribution pQ).

When the adversary returns ℓ^t, we return to each A_i the loss vector $p_i^t \ell^t$. So, algorithm A_i experiences loss $(p_i^t \ell^t) \cdot q_i^t = p_i^t (q_i^t \cdot \ell^t)$.

Now we consider the guarantee that we have for algorithm A_i, namely, for any action j,

$$\sum_{t=1}^{T} p_i^t (q_i^t \cdot \ell^t) \quad \leq \quad \sum_{t=1}^{T} p_i^t \ell_j^t + R_i(L_{min}, T, N) \; . \tag{1}$$

If we sum the losses of the N algorithms at any time t, we get $\sum_i p_i^t (q_i^t \cdot \ell^t) = p^t Q^t \ell^t$, where p^t is the row-vector of our probabilities, Q^t is the matrix of $q_{i,j}^t$, and ℓ^t is viewed as a column-vector. By design of p^t, we have $p^t Q^t = p^t$. So, the sum of the perceived losses of the N algorithms is equal to our actual loss $p^t \ell^t$.

Therefore, summing equation (1) over all N algorithms, the left-hand-side sums to L_H^T. Since the right-hand-side of equation (1) holds for any j, we have that for any function $F : \{1, \ldots, N\} \to \{1, \ldots, N\}$,

$$L_H^T \leq \sum_{i=1}^{N} \sum_{t=1}^{T} p_i^t \ell_{F(i)}^t + \sum_{i=1}^{N} R_i(L_{min}, T, N).$$

We have therefore proven the following theorem.

Theorem 5 *For any N algorithms A_i with regret R_i, for every function $F : \{1, \ldots, N\} \to \{1, \ldots, N\}$, for any sequence of T losses ℓ^t such that some action has total loss at most L_{min}, the above algorithm satisfies*

$$L_H \leq L_{H,F} + \sum_{i=1}^{N} R_i(L_{min}, T, N),$$

i.e., the swap-regret of H is at most $\sum_{i=1}^{N} R_i(L_{min}, T, N)$.

A typical *optimized experts algorithm*, such as in Littlestone and Warmuth (1994), Freund and Schapire (1995), Auer et al. (2002b), and Cesa-Bianchi et al. (1997), will have $R(L_{min}, T, N) = O(\sqrt{L_{min} \log N} + \log N)$. (Alternatively, Corollary 15 can be also used to deduce the above bound.) We can immediately derive the following corollary.

Corollary 6 *Using an optimized experts algorithm as the A_i, for every function $F : \{1, \ldots, N\} \to \{1, \ldots, N\}$, we have that*

$$L_H \leq L_{H,F} + O(N\sqrt{T \log N}) \; .$$

We can perform a slightly more refined analysis of the bound by having L_{min}^i be the minimum loss for an action in A_i. Note that $\sum_{i=1}^{N} L_{min}^i \leq T$, since we scaled the losses given to algorithm A_i at time t by p_i^t. By convexity of the square-root function, this implies that $\sum_{i=1}^{N} \sqrt{L_{min}^i} \leq \sqrt{NT}$, which implies the worst case regret is $O(\sqrt{TN \log N})$.[3]

3. We need to use here an external regret algorithm which does not need to have as an input the value of L_{min}^i. An example of such an algorithm is Corollary 2 in Cesa-Bianchi et al. (2005), which guarantees an external regret of at most $O(\sqrt{L_{min} \log N} + \log N)$.

Corollary 7 *Using optimized experts algorithms as the A_i, for every function $F : \{1, \ldots, N\} \rightarrow \{1, \ldots, N\}$, we have that*

$$L_H \leq L_{H,F} + O(\sqrt{TN \log N}) .$$

One strength of the above general reduction is it ability to accommodate new regret minimization algorithms. For example, using the algorithms of Cesa-Bianchi et al. (2005) one can get a more refined regret bound, which depends on the second moment.

4. Lower bounds on swap regret

Notice that while good algorithms for external regret achieve bounds of $O(\sqrt{T \log N})$, our swap-regret bounds are roughly $O(\sqrt{TN \log N})$. Or, to put it another way, imagine we are interested in the number of time steps T needed to achieve an average regret of ϵ per time step (a total regret of ϵT). Then, for external regret we have algorithms that can do this in $T = O(\epsilon^{-2} \log N)$ steps, whereas our bounds require $T = O(\epsilon^{-2} N \log N)$ steps for the case of swap-regret. From the point of view of equilibrium, this means that while for *two-player zero-sum* games such algorithms will achieve approximate minimax-optimality in $O(\log N)$ steps when played against each other, for *general-sum* games we seem to need $O(N \log N)$ steps to achieve an ϵ-correlated equilibrium. A natural question is whether this is best possible. In particular, is it possible to guarantee swap-regret at most ϵT in a number of time steps that is sublinear in the size of the action space N?

We give here a partial answer: a lower bound of $\Omega(\sqrt{TN})$ on swap-regret but in a more adversarial model. Specifically, we have defined swap regret with respect to the *distribution* p^t produced by the player, rather than the actual action a_t selected from that distribution. In the case that the adversary is oblivious (does not depend on the player's action selection) then the two models have the same expected regret. However we will consider an *adaptive* adversary, whose choices may depend on the player's action selection in previous rounds. In this setting (adaptive adversary *and* regret defined with respect to the action selected from p^t rather than p^t itself) we derive an $\Omega(\sqrt{TN})$ lower bound.

Before presenting these bounds, we first mention one subtle issue. For a given stochastic adversary, the optimal policy for minimizing *loss* may not be the optimal policy for minimizing swap-regret. For example, consider a process in which $\{0, 1\}$ losses are generated by a fair coin, except that in time steps $t \in ((i-1)T/N, iT/N]$, action i has loss of 1 with probability only $1/2 - 1/T$. In this case, the optimal policy for minimizing expected *loss* uses action $i = 1$ for the first T/N steps, then action $i = 2$ for the next T/N steps, and so forth. However, because of the variance of the coin flips, in retrospect each action played can be swapped with another for an expected gain of $\Omega(\sqrt{(T \log N)/N})$ each (see, e.g., Feller (1968, 1971)), giving a total swap-regret of $\Omega(\sqrt{TN \log N})$ for this policy. On the other hand, a policy that just picks a single fixed action would have swap-regret only $O(\sqrt{T \log N})$ even though its expected loss is slightly higher.

We now present our lower bound, first giving a somewhat weaker version whose proof is simpler but contains the main ideas, and then giving the stronger version. Notice that even the weaker version (Theorem 8) implies that $\Omega(N/\epsilon)$ time steps are necessary in order to achieve an average swap regret ϵ per time step, in the adaptive adversary model.

Theorem 8 *There exists an adaptive adversary such that for any randomized online algorithm A, its expected swap regret* $E[\max_{F \in \mathcal{F}^{sw}} L_A - L_{A,F}]$ *(defined with respect to the actions selected from* p^t *rather than* p^t *itself), is at least* $\min((N-1)/16, T/8)$.

Proof The adversary behaves as follows. At each time step t, all actions i that have been previously played by algorithm A receive a loss of 1. All other actions i receive either (a) a loss chosen uniformly and independently at random from $\{0, 1\}$ if $i \in \{1, \ldots, N/2\}$, or (b) a loss of exactly $1/2$ if $i \in \{N/2 + 1, \ldots, N\}$. The basic idea of the argument now is that so long as there are still actions of type (a) remaining, algorithm A has no good choices: if it chooses to play a previously-played action, then it will incur large loss, whereas if it chooses to play a new action, this will have a good probability of substantially increasing swap regret. The presence of the actions of type (b) with loss exactly 1/2 is not really necessary but helps to simplify the calculations. Formally, we argue as follows.

Assume $T < N/2$. We will give a swap-regret lower bound of $T/8$, implying our desired result. In particular, we will keep track of two quantities: E_F, the expected regret of A with respect to a specific modification rule F, and E_L, the expected loss of A. Modification rule F is defined as follows: the first time t that algorithm A plays some action i, we define $F(i)$ to be whichever action performed best at time t (breaking ties arbitrarily); for actions never played, the value of F does not matter. Now, consider some specific time step t. If algorithm A plays some action i that was never before played, then E_F increases by at least 1/4 (since for $T < N/2$ there must be at least one other unplayed action $j \leq N/2$ and $\mathbf{E}[\max(\ell_i^t - \ell_j^t, 0)] = 1/4$), and E_L increases by 1/2. On the other hand, if algorithm A plays a previously-played action, then E_L increases by 1, and E_F at least does not decrease. Notice that either way, $2E_F + E_L$ increases by at least 1. Therefore, by time T we have $2E_F + E_L \geq T$. Now, if algorithm A is such that $E_F \geq T/8$ then we are done (the expected regret with respect to F is large). On the other hand, if $E_F < T/8$, then this implies $E_L \geq 3T/4$. However, this means that algorithm A has large expected *external* regret since there must exist some unplayed action $j > N/2$ whose total loss is exactly $T/2$. Thus in this case the expected swap-regret of A is at least $T/4$. ■

Theorem 9 *There exists an adaptive adversary such that for any randomized online algorithm A, its expected swap regret* $E[\max_{F \in \mathcal{F}^{sw}} L_A - L_{A,F}]$ *(defined with respect to the actions selected from* p^t *rather than* p^t *itself), is at least* $\sqrt{TN}/160 - 1$, *for* $N \leq T \leq \frac{1}{\sqrt{N}} e^{N/288}$.

Proof The adversary is the same as that used in the proof of Theorem 8, except now it waits until an action is played $8T/N$ times before causing it to deterministically receive a loss of 1. Specifically, all actions played by algorithm A at least $8T/N$ times receive a loss of 1, and all other actions i receive either (a) a loss chosen uniformly and independently at random from $\{0, 1\}$ if $i \in \{1, \ldots, N/2\}$, or (b) a loss of exactly 1/2 if $i \in \{N/2+1, \ldots, N\}$. Let us call an action that has been played less than $8T/N$ times a "low-loss action" and those played at least $8T/N$ times a "1-loss action". Let T^ℓ denote the number of times the algorithm plays low-loss actions (which could be a random variable depending on the algorithm). Notice that the expected loss of the algorithm is $\mathbf{E}[T^\ell/2 + (T - T^\ell)] = T - \mathbf{E}[T^\ell/2]$.

We break the argument into two cases based on $\mathbf{E}[T^\ell]$. The simpler case is $\mathbf{E}[T^\ell] \leq 3T/4$ (i.e., the algorithm plays many 1-loss actions). In that case, the expected loss of the algorithm is at least $5T/8$. On the other hand, there must be some action of total loss at most $T/2$ because it is not possible for the algorithm to have played all actions $i \in \{N/2+1, \ldots, N\}$ for $8T/N$ times each. So, the expected regret is at least $T/8 \geq \sqrt{TN}/8$.

We now analyze the case that $\mathbf{E}[T^\ell] > 3T/4$. Let $\mathcal{T}_i$ denote the time steps in which the algorithm plays i, and let $T_i = |\mathcal{T}_i|$. Define modification rule F such that $F(i)$ is the action of least total loss in time steps $\mathcal{T}_i$. We will argue later that with probability $1-\lambda$ (where we will set $\lambda = 1/T$), for all i the action $F(i)$ has loss at most $T_i/2 - \sqrt{T_i}/5$ during time steps $\mathcal{T}_i$. So, letting R denote the swap-regret of algorithm A, the expected swap-regret $\mathbf{E}[R]$ is at least the difference between its expected loss and $\mathbf{E}[\sum_i T_i/2 - \sqrt{T_i}/5] + \lambda T$:

$$\mathbf{E}[R] \;\geq\; T - \mathbf{E}\left[\frac{T^\ell}{2}\right] - \mathbf{E}\left[\sum_{i=1}^{N} \frac{T_i}{2} - \frac{\sqrt{T_i}}{5}\right] - \lambda T \geq \frac{1}{5}\mathbf{E}\left[\sum_{i=1}^{N} \sqrt{T_i}\right] - \lambda T,$$

where we use the fact that $\sum_i T_i = T$ and $T^\ell \leq T$.

The number of actions i such that $T_i \geq T/(4N)$ is at least $(T^\ell - T/4)/(8T/N)$, since even if one considers only the time steps in which low-loss actions are played, at most $T/4$ of them can involve playing actions with $T_i < T/(4N)$, and for the rest, any given action i can occur at most $8T/N$ times. Since $\mathbf{E}[T^\ell] \geq 3T/4$, the *expected* number of such actions is at least $(T/2)/(8T/N) = N/16$ and therefore,

$$\mathbf{E}[R] \;\geq\; \frac{1}{5}\mathbf{E}\left[\sum_{i=1}^{N} \sqrt{T_i}\right] - \lambda T \;\geq\; \frac{N}{80}\sqrt{\frac{T}{4N}} - \lambda T \;=\; \frac{\sqrt{TN}}{160} - \lambda T.$$

It remains to show that with high probability, all actions $F(i)$ have loss at most $T_i/2 - \sqrt{T_i}/5$ during time steps $\mathcal{T}_i$. First, note that in K coin tosses, with probability at least $1/3$ we have at most $K/2 - \sqrt{K}/5$ heads. Fix an action i and any given value of T_i. So, by Hoeffding bounds, if $N/2$ coins (corresponding to actions $1, \ldots, N/2$) are each tossed $K = T_i$ times, with probability at least $e^{-N(1/3-1/4)^2}$ we have over $N/8$ (i.e., $1/4$ of them) with at most $T_i/2 - \sqrt{T_i}/5$ heads. Thus, even if the algorithm could decide which (at most $N/8$) actions to turn into 1-loss actions after the fact, with probability at least $e^{-N(1/3-1/4)^2}$ there will still be at least one with loss at most $T_i/2 - \sqrt{T_i}/5$. Summing over all i and all possible values of T_i yields a failure probability at most $NTe^{-N(1/3-1/4)^2} = \lambda$. For $T \leq \frac{1}{\sqrt{N}}e^{N/288}$, this is at most $1/T$, completing the proof. ■

5. Reducing External to Swap Regret in the Partial information model

In the full information setting the learner gets, at the end of each time step, full information on the costs of all the actions. In the partial information (multi-arm bandit) model, the learner gets information only about the action that was selected. In some applications this is a more plausible model regarding the information the learner can observe.

The reduction in the partial information model is similar to the one of the full information model, but with a few additional complications. We are given N partial information algorithms A_i. At each time step t, each algorithm A_i outputs a distribution q_i^t. Our master online algorithm combines them to some distribution p^t which it uses. Given p^t it receives a feedback, but now this includes information only regarding one action, i.e., it receives $(\ell_{k^t}^t, k^t)$, where k^t is distributed according to p^t. We take this feedback and distribute to each algorithm A_i a feedback (c_i^t, k^t), such that $\sum_i c_i^t = \ell_{k^t}^t$. The main technical difficulty is that now the action selected, k^t, is distributed according to p^t and not q_i^t. (For example, it might be that A_i has $q_{i,j}^t = 0$ but it receives feedback about action j. From A_i's point of view this is impossible! Or, more generally, A_i might start noticing it seems to have a very bad random-number generator.) For this reason, for the reduction to work we need to make a stronger assumption about the guarantees of the algorithms A_i, which luckily is implicit in the algorithms of Auer et al. (2002b). Since results of Auer et al. (2002b) are stated in terms of maximizing gain rather then minimizing loss we will switch to this notation, e.g., define the benefit of action j at time t to be $b_j^t = 1 - \ell_j^t$.

We start by describing our MAB algorithm *SR_MAB*. Initially, we are given N partial information algorithms A_i. At each time step t, each A_i gives a *selection distribution* q_i^t over actions. Given all the selection distributions we compute an *action distribution* p^t. We would like to keep two sets of gains: one is the *real gain*, denoted by b_i^t, and the other is the gain that the MAB algorithm A_i observes, $g_{A_i}^t$. Given the action distribution p^t the adversary selects a vector of real gains b_i^t. Our MAB algorithm *SR_MAB* receives a single feedback $(b_{k^t}^t, k^t)$ where k^t is a random variable that with probability p_j^t equals j. Algorithm *SR_MAB*, given b^t, returns to each A_i a pair $(g_{A_i}^t, k^t)$, where the observed gain $g_{A_i}^t$ is based on $b_{k_t}^t$, p^t and q_i^t. Again, note that k^t is distributed according to p^t, which may not equal q_i^t: it is for this reason we need to use an MAB algorithm that satisfies certain properties (stated in Lemma 10).

In order to specify our MAB algorithm, *SR_MAB*, we need to specify how it selects the action distribution p^t and the observed gains $g_{A_i}^t$. As in the full information case, we compute an action distribution p^t such that $p_j^t = \sum_i p_i^t q_{i,j}^t$. That is, $p = pQ$, where p is the row-vector of our probabilities and Q is the matrix of $q_{i,j}$. Given p^t the adversary returns a real gain $(b_{k^t}^t, k^t)$, namely, the real gain is of our algorithm $b_{k^t}^t$. We return to each algorithm A_i an observed gain of $g_{A_i}^t = p_i^t b_{k^t}^t q_{i,k^t} / p_{k^t}^t$. (In general, define $g_{i,j}^t = p_i^t b_j^t q_{i,j}^t / p_j^t$, if $j = k^t$ and $g_{i,j}^t = 0$ if $j \neq k^t$.)

First, we will show that $\sum_{i=1}^N g_{A_i}^t = b_{k^t}^t$ which implies that $g_{A_i}^t \in [0,1]$. From the property of the distribution p^t we have that,

$$\sum_{i=1}^{N} g_{A_i}^t = \sum_{i=1}^{N} \frac{p_i^t b_{k^t}^t q_{i,k^t}}{p_{k^t}^t} = \frac{p_{k^t}^t b_{k^t}^t}{p_{k^t}^t} = b_{k^t}^t.$$

This shows that we distribute our real gain among the algorithms A_i; that is, that the sum of the observed gains equals the real gain. In addition, it bounds the observed gain that each algorithm A_i receives. Namely, $0 \leq g_{A_i}^t \leq b_{k^t}^t \leq 1$.

In order to describe the guarantee that each external regret multi-arm bandit algorithm A_i is required to have, we need the following additional definition. At time t let $X_{i,j}^t$ be a random variable such that $X_{i,j}^t = g_{i,j}^t / q_{i,j}^t$ if $j = k^t$ and $X_{i,j}^t = 0$ otherwise. The expectation of $X_{i,j}^t$ is,

$$E_{k^t \sim p^t}[X_{i,k^t}^t] \;=\; p_{k^t}^t \frac{g_{i,k^t}^t}{q_{i,k^t}^t} \;=\; p_{k^t}^t \frac{p_i^t b_{k^t}^t}{p_{k^t}^t} \;=\; p_i^t b_{k^t}^t.$$

Lemma 10 (Auer et al. (2002b)) *There exists a multi-arm bandit algorithm, A_i, such that for any sequence of observed gains $g_{i,j}^t \in [0,1]$ it outputs actions distributions q_i^t, and for any sequence of selected actions k^t, and for any action r and parameter $\gamma \in (0,1]$, then,*

$$G_{A_i,g^t} \equiv \sum_{t=1}^{T} g_{A_i}^t \equiv \sum_{t=1}^{T} g_{i,k^t}^t \geq (1-\gamma)\sum_{t=1}^{T} X_{i,r}^t - \frac{N \ln N}{\gamma} - \frac{\gamma}{N}\sum_{t=1}^{T}\sum_{j=1}^{N} X_{i,j}^t, \qquad (2)$$

where $X_{i,j}^t$ is a random variable such that $X_{i,j}^t = g_{i,j}^t / q_{i,j}^t$ if $j = k^t$ and $X_{i,j}^t = 0$ otherwise.

Note that in Auer et al. (2002b) the action distribution is identical to the selection distribution, i.e. $p^t \equiv q^t$, and the observed and real gain are identical, i.e., $g^t \equiv b^t$. Auer et al. (2002b) derive the external regret bound by taking the expectation with respect to the action distribution (which is identical to the selection distribution). In our case we separate the real gain from the observed gain, which adds another layer of complication. (Technically, the distribution p^t is a random variable that depends on the history $\mathcal{H}^{t-1}$ up to time t, i.e., the observed actions $k^1, \ldots k^{t-1}$ and well as the observed gains $b_{k^1}^1, \ldots b_{k^{t-1}}^{t-1}$. For this reason we take the expectation with respect to $\mathcal{H}^{t-1}$ every time we refer to p^t. For simplicity we make this dependency implicitly in the expectations $E[\cdot]$.) We define the expected benefit of SR_MAB to be $B_{SR\_MAB} = E[\sum_{t=1}^{T} b_{SR\_MAB}^t]$ and for a function $F : \{1, \ldots, N\} \rightarrow \{1, \ldots, N\}$ we define $B_{SR\_MAB,F} = E[\sum_{t=1}^{T}\sum_{i=1}^{N} p_i^t b_{F(i)}^t]$. We now state our main theorem regarding the partial information model.

Theorem 11 *Given a multi-arm bandit algorithm satisfying Lemma 10 (such as the algorithm of Auer et al. (2002b)), it can be converted to a master online algorithm SR_MAB, such that*

$$B_{SR\_MAB} \geq \max_F B_{SR\_MAB,F} - N \cdot R^{MAB}(B_{max}, T, N) \;,$$

where the expectation is over the observed actions of SR_MAB, B_{max} bounds the maximum benefit of any algorithm and $R^{MAB}(B,T,N) = O(\sqrt{BN \log N})$.

Proof Let the total observed gain of algorithm A_i be $G_{A_i} = \sum_{t=1}^{T} g_{A_i}^t = \sum_{t=1}^{T} g_{i,k^t}^t$. Since we distribute our gain between the A_i, i.e., $\sum_{i=1}^{N} g_{A_i}^t = b_{SR\_MAB}^t$, we have that $B_{SR\_MAB} = E[\sum_{t=1}^{T} b_{SR\_MAB}^t] = \sum_{i=1}^{N} E[G_{A_i}]$. Since $g_{i,j}^t \in [0,1]$, by Lemma 10, this implies that for any action r, after taking the expectation, we have

$$\begin{aligned} E[G_{A_i}] &= E[\sum_{t=1}^{T} E_{p^t}[g_{i,k^t}^t]] \\ &\geq (1-\gamma)E[\sum_{t=1}^{T} E_{p^t}[X_{i,r}^t]] - \frac{N \ln N}{\gamma} - \frac{\gamma}{N} E\left[\sum_{t=1}^{T}\sum_{j=1}^{N} E_{p^t}[X_{i,j}^t]\right] \end{aligned}$$

$$
\begin{aligned}
&= (1-\gamma)E[\sum_{t=1}^{T} p_i^t b_r^t] - \frac{N \ln N}{\gamma} - \frac{\gamma}{N} E\left[\sum_{t=1}^{T}\sum_{j=1}^{N} p_i^t b_j^t\right] \\
&\geq (1-\gamma)B_{i,r} - \frac{N \ln N}{\gamma} - \frac{\gamma}{N}\sum_{j=1}^{N} B_{i,j} \\
&\geq B_{i,r} - O(\sqrt{B_{max} N \ln N}) = B_{i,r} - R^{MAB}(B_{max}, N, T) \ ,
\end{aligned}
$$

where $B_{i,r} = E[\sum_{t=1}^{T} p_i^t b_r^t]$, $B_{max} \geq \max_{i,j} B_{i,j}$ and $\gamma = \min\{\sqrt{(N \ln N)/B_{max}}, 1\}$.

For swap regret, we compare the expected benefit of SR_MAB to that of $\sum_{i=1}^{N} \max_j B_{i,j}$. Therefore,

$$
B_{SR\_MAB} = \sum_{i=1}^{N} E[G_{A_i}] \geq \max_{F} \sum_{i=1}^{N} B_{i,F(i)} - N \cdot R^{MAB}(B_{max}, T, N)
$$

which completes the proof of the theorem. ■

6. External Regret with Time-Selection Functions

We now present a simple online algorithm that achieves a good external regret bound in the presence of time selection functions, generalizing the *sleeping experts* setting. Specifically, our goal is for each action a, and each time-selection function I, that our total loss during the time-steps selected by I should be not much more than the loss of a during those time steps. More generally, this should be true for the losses *weighted* by I when $I(t) \in [0,1]$. The idea of the algorithm is as follows. Let $R_{a,I}$ be the regret of our algorithm with respect to action a and time selection function I. That is, $R_{a,I} = \sum_t I(t)(\ell_H^t - \ell_a^t)$. Let $\tilde{R}_{a,I}$ be a less-strict notion of regret in which we multiply our loss by some $\beta \in (0,1)$, that is, $\tilde{R}_{a,I} = \sum_t I(t)(\beta \ell_H^t - \ell_a^t)$. What we will do is give to each action a and time selection function I a weight $w_{a,I}$ that is exponential in $\tilde{R}_{a,I}$. We will prove that the sum of our weights never increases, and thereby be able to easily conclude that none of the $\tilde{R}_{a,I}$ can be too large.

Specifically, for each of the N actions and the M time selection functions we maintain a weight $w_{a,I}^t$. We update these weights using the rule $w_{a,I}^{t+1} = w_{a,I}^t \beta^{I(t)(\ell_a^t - \beta \ell_H^t)}$, where ℓ_H^t is the loss of our online algorithm H at time t. (Initially, $w_{a,I}^0 = 1$.) Equivalently, $w_{a,I}^t = \beta^{-\tilde{R}_{a,I}^t}$, where $\tilde{R}_{a,I}^t$ is the "less-strict" regret mentioned above up to time t.

At time t we define $w_a^t = \sum_I I(t) w_{a,I}^t$, $W^t = \sum_a w_a^t$ and $p_a^t = w_a^t / W^t$. Our distribution over actions at time t is p^t. The following claim shows that the weights remain bounded.

Claim 12 *At any time t we have $0 \leq \sum_{a,I} w_{a,I}^t \leq NM$.*

Proof Initially, at time $t = 0$, the claim clearly holds. Observe that at time t we have the following identity,

$$
W^t \ell_H^t = W^t \sum_a p_a^t \ell_a^t = \sum_a w_a^t \ell_a^t = \sum_a \sum_I I(t) w_{a,I}^t \ell_a^t. \tag{3}
$$

For the inductive step we show that the sum of the weights can only decrease. Note that for any $\beta \in [0,1]$ and $x \in [0,1]$ we have $\beta^x \le 1-(1-\beta)x$ and $\beta^{-x} \le 1+(1-\beta)x/\beta$. Therefore,

$$\begin{aligned}
\sum_a \sum_I w_{a,I}^{t+1} &= \sum_a \sum_I w_{a,I}^t \beta^{I(t)(\ell_a^t - \beta \ell_H^t)} \\
&= \sum_a \sum_I w_{a,I}^t \beta^{I(t)\ell_a^t} \beta^{-\beta I(t)\ell_H^t)} \\
&\le \sum_a \sum_I w_{a,I}^t (1-(1-\beta)I(t)\ell_a^t)(1+(1-\beta)I(t)\ell_H^t) \\
&\le \left(\sum_a \sum_I w_{a,I}^t\right) - (1-\beta)\left(\sum_{a,I} I(t) w_{a,I}^t \ell_a^t\right) + (1-\beta)\left(\sum_{a,I} I(t) w_{a,I}^t \ell_H^t\right) \\
&= \left(\sum_a \sum_I w_{a,I}^t\right) - (1-\beta)W^t \ell_H^t + (1-\beta)W^t \ell_H^t \quad \text{(using eqn. (3))} \\
&= \left(\sum_a \sum_I w_{a,I}^t\right) \quad ,
\end{aligned}$$

which completes the proof of the claim. ■

We use the above claim to bound the weight of any action a and time-selection function I.

Corollary 13 *For every action a and time selection I we have*

$$w_{a,I}^t = \beta^{L_{a,I} - \beta L_{H,I}} \le MN,$$

where $L_{H,I} = \sum_t I(t)\ell_H^t$ is the loss of the online algorithm with respect to time-selection function I.

A simple algebraic manipulation of the above implies the following theorem

Theorem 14 *For every action a and every time selection function $I \in \mathcal{I}$ we have*

$$L_{H,I} \le \frac{L_{a,I} + \frac{\log NM}{\log \frac{1}{\beta}}}{\beta} .$$

We can optimize for β in advance, or do it dynamically using Auer et al. (2002a), establishing:

Corollary 15 *For every action a and every time selection function $I \in \mathcal{I}$ we have*

$$L_{H,I} \le L_{a,I} + O(\sqrt{L_{min} \log NM} + \log MN),$$

where $L_{min} = \max_I \min_a \{L_{a,I}\}$.

One can get a more refined regret bound of $O(\sqrt{L_{min,I}\log NM}+\log MN)$ with respect to each time selection function $I \in \mathcal{I}$, where $L_{min,I} = \min_a\{L_{a,I}\}$. This is achieved by keeping a parameter β_I for each time selection function $I \in \mathcal{I}$. As before we then set $w^t_{a,I} = \beta_I^{-\tilde{R}^t_{a,I}}$, where $\tilde{R}^t_{a,I} = \sum_{t'\leq t} I(t')(\beta_I \ell^{t'}_H - \ell^{t'}_a)$. We then let $w^t_a = \sum_I (1-\beta_I)I(t)w^t_{a,I}$, $W^t = \sum_a w^t_a$ and $p^t_a = w^t_a/W^t$. The proof of Claim 12 holds in a similar way, and from that one can derive, analogously, the more refined regret bound, as stated in the following theorem.

Theorem 16 *For every action a and every time selection function $I \in \mathcal{I}$ we have*

$$L_{H,I} \leq L_{a,I} + O(\sqrt{L_{min,I}\log NM} + \log MN),$$

where $L_{min,I} = \min_a\{L_{a,I}\}$.

7. Arbitrary time selection and modification rules

In this section we combine the techniques from Sections 3 and 6 to derive a regret bound for the general case where we assume that there is a finite set $\mathcal{I}$ of M time selection functions, and a finite set $\mathcal{F}$ of K modification rules. Our goal is to design an algorithm such that for any time selection function $I \in \mathcal{I}$ and any $F \in \mathcal{F}$, we have that $L_{H,I}$ is not much larger than $L_{H,I,F}$. Essentially, our results can be viewed as deriving explicit rates for the wide range regret of Lehrer (2003).

We maintain at time t a weight $w^t_{j,I,F}$ per action j, time selection I and modification rule F. Initially $w^0_{j,I,F} = 1$. We set

$$w^{t+1}_{j,I,F} = w^t_{j,I,F}\beta^{p^t_j I(t)(\ell^t_{F(j)} - \beta\ell^t_{H,j})},$$

and let $W^t_{j,F} = \sum_I I(t)w^t_{j,I,F}$, $W^t_j = \sum_F W^t_{j,F}$, and $\ell^t_{H,j} = \sum_F W^t_{j,F}\ell^t_{F(j)}/W^t_j$.

We use the weights to define a distribution p^t over actions as follows. We select a distribution p^t such that

$$p^t_i \;=\; \sum_{j=1}^N p^t_j \sum_{F:F(j)=i} \frac{W^t_{j,F}}{W^t_j} \quad . \tag{4}$$

I.e., p is the stationary distribution of the associated Markov chain. Notice that the definition of p implies that the loss of H at time t can either be viewed as $\sum_i p^t_i \ell^t_i$ or as $\sum_j p_j \sum_F (W^t_{j,F}/W^t_j)\ell^t_{F(j)} = \sum_j p^t_j \ell^t_{H,j}$. The following claim bounds the magnitude of the weights.

Claim 17 *For every action j, at any time t we have $0 \leq \sum_{I,F} w^t_{j,I,F} \leq MK$.*

Proof This clearly holds initially at $t = 0$. For any $t \geq 0$ we show that $\sum_{I,F} w^{t+1}_{j,I,F} \leq \sum_{I,F} w^t_{j,I,F}$. Recall that for $\beta \in [0,1]$ and $x \in [0,1]$ we have $\beta^x \leq 1-(1-\beta)x$ and $\beta^{-x} \leq 1 + (1-\beta)x/\beta$.

$$\sum_{I,F} w^{t+1}_{j,I,F} \;=\; \sum_{I,F} w^t_{j,I,F}\beta^{p^t_j I(t)(\ell^t_{F^t(j)} - \beta\ell^t_{H,j})}$$

$$
\begin{aligned}
&\le \sum_{I,F} w^t_{j,I,F}(1-(1-\beta)p^t_j I(t)\ell^t_{F^t(j)})(1+(1-\beta)p^t_j I(t)\ell^t_{H,j}) \\
&\le \left(\sum_{I,F} w^t_{j,I,F}\right) - (1-\beta)p^t_j \sum_F \ell^t_{F^t(j)} \sum_I I(t) w^t_{j,I,F} + (1-\beta)p^t_j \ell^t_{H,j} \sum_{I,F} I(t) w^t_{j,I,F} \\
&= \left(\sum_{I,F} w^t_{j,I,F}\right) - (1-\beta)p^t_j \sum_F \ell^t_{F^t(j)} W^t_{j,F} + (1-\beta)p^t_j \ell^t_{H,j} W^t_j \\
&= \left(\sum_{I,F} w^t_{j,I,F}\right) - (1-\beta)p^t_j W^t_j \ell^t_{H,j} + (1-\beta)p^t_j W^t_j \ell^t_{H,j} \\
&= \sum_{I,F} w^t_{j,I,F} \ ,
\end{aligned}
$$

where in the second to last equality we used the identity $\sum_F \ell^t_{F^t(j)} W^t_{j,F} = \ell^t_{H,j} W^t_j$. ■

The following theorem derives the general regret bound.

Theorem 18 *For every time selection $I \in \mathcal{I}$ and modification rule $F \in \mathcal{F}$, we have that*

$$L_{H,I} \le L_{H,I,F} + O(\sqrt{TN \log MK}) \ .$$

Proof Consider a time selection function $I \in \mathcal{I}$ and a modification function $F \in \mathcal{F}$. By Claim 17 we have that,

$$w^T_{j,I,F} = \beta^{(\sum_t p^t_j I(t)\ell^t_{F^t(j)}) - \beta(\sum_t p^t_j I(t)\ell^t_{H,j})} \le MK \ ,$$

which is equivalent to

$$\left(\sum_t I(t) p^t_j \ell^t_{H,j}\right) \le \frac{1}{\beta}\left(\sum_t I(t) p^t_j \ell^t_{F^t(j)}\right) + \frac{\log MK}{\beta \log \frac{1}{\beta}} \ .$$

Notice that $\sum_{j,t} I(t) p^t_j \ell^t_{H,j} = \sum_{i,t} I(t) p^t_i \ell^t_i$, by the definition of the p_i's in Equation (4). Summing over all actions j this sum is $L_{H,I}$. Therefore,

$$
\begin{aligned}
L_{H,I} = \sum_{j=1}^N \left(\sum_t I(t) p^t_j \ell^t_{H,j}\right) &\le \sum_{j=1}^N \frac{1}{\beta}\left(\sum_t I(t) p^t_j \ell^t_{F^t(j)}\right) + \frac{N \log MK}{\beta \log \frac{1}{\beta}} \\
&= \frac{1}{\beta} L_{H,I,F} + \frac{N \log MK}{\beta \log \frac{1}{\beta}},
\end{aligned}
$$

where $L_{H,I}$ is the cost of the online algorithm at time selection I and $L_{H,I,F}$ is the cost of the modified output sequence at time selection I. Optimizing for β we derive the theorem. ■

8. Boolean Prediction with Time Selection

In this section we consider the case that there are two actions $\{0,1\}$, and the loss function is such that at every time step t one action has loss 1 and the other has loss 0. Namely, we assume that the adversary returns at time t an action $o^t \in \{0,1\}$, and the loss of action a^t is 1 if $a^t \neq o^t$ and 0 if $a^t = o^t$. Our objective here is to achieve good bounds with a deterministic algorithm.

For each time selection function $I \in \mathcal{I}$, action $a \in \{0,1\}$, and time t, our online Boolean prediction algorithm maintains a weight $w^t_{a,I}$. Initially we set $w^0_{a,I} = 1$ for every action $a \in \{0,1\}$ and time selection function $I \in \mathcal{I}$. At time t, for each action $a \in \{0,1\}$, we compute $w^t_a = \sum_I I(t) w^t_{a,I}$, and predict $a^t = 1$ if $w^t_1 \geq w^t_0$, and otherwise predict $a^t = 0$. The *weighted errors* of our online Boolean prediction algorithm, during the time selection function $I \in \mathcal{I}$, is $\sum_{t:o_t \neq a^t} I(t)$.

Following our prediction we observe the adversary action o^t. If no error occurred (i.e., $a_t = o_t$) then all the weights at time $t+1$ equal the weights at time t. If an error occurred (i.e., $a_t \neq o_t$) then we update the weights as follows. For every time selection function $I \in \mathcal{I}$ we set the weight of action b to $w^{t+1}_{b,I} = w^t_{b,I} 2^{cI(t)}$, where $c = -1$ if $b \neq o_t$ and $c = 1/2$ if $b = o_t$. This can be viewed as a version of the Balanced-Winnow algorithm of Littlestone (1988). We establish the following claim,

Claim 19 *At any time t we have $0 \leq \sum_{a,I} w^t_{a,I} \leq 2M$.*

Proof Clearly this holds at time $t = 0$. When an error is performed, we have that $w^t_{error} \geq w^t_{correct}$, where *correct* $= o^t$ and *error* $= 1 - o^t$. The additive change in the weights is at most $(\sqrt{2}-1)w^t_{correct} - w^t_{error}/2 < 0$, which completes the proof. ∎

For a time selection function $I \in \mathcal{I}$, let $v_{a,I} = \sum_{t:o_t=a} I(t)$. The *preferred action* for a time selection function I is 1 if $v_{1,I} \geq v_{0,I}$ and 0 otherwise. Let $OPT(I)$ be the weighted errors of the preferred action during time selection function I. W.l.o.g., assume that the preferred action for I is 1, which implies that $OPT(I) = v_{0,I}$. By Claim 19 we have that $w^t_{1,I} \leq 2M$. The total decrease in $w^t_{1,I}$ is bounded by a factor of $2^{-v_{0,I}}$. Since $w^T_{1,I} \leq 2M$ this implies that $2^{x/2 - v_{0,I}} \leq 2M$, where x is the total increase, which implies that

$$x \leq 2v_{0,I} + 2 + 2\log_2 M \ .$$

The weighted errors of our online Boolean prediction algorithm, during time selection function $I \in \mathcal{I}$, i.e., $\sum_{t:a^t \neq o^t} I(t)$, is at most $x + v_{0,I}$, while the preferred action makes only $v_{0,I}$ weighted errors. This implies that the weighted errors of our online Boolean prediction algorithm during time selection function I is bounded by $x + v_{0,I} = 3v_{0,I} + 2 + 2\log_2 M$, which establishes the following theorem.

Theorem 20 *For every $I \in \mathcal{I}$, our online algorithm makes at most $3OPT(I) + 2 + 2\log_2 M$ weighted errors.*

9. Conclusion and open problems

In this paper we give general reductions by which algorithms achieving good external regret can be converted to algorithms with good internal or swap regret, and in addition we develop algorithms for a generalization of the sleeping experts scenario including both real-valued time-selection functions and a finite set of modification rules.

A key problem left open by this work is whether it is possible to achieve swap-regret that has a sublinear or even logarithmic dependence on N. Specifically, for *external* regret, existing algorithms achieve regret ϵT in time $T = O(\frac{1}{\epsilon^2} \log N)$, but our algorithms for swap-regret achieve regret ϵT only by time $T = O(\frac{1}{\epsilon^2} N \log N)$. We have shown that sublinear dependence is not possible against an adaptive adversary with swap-regret defined with respect to the actions actually chosen from the algorithm's distribution, but we do not know whether there is a comparable lower bound in the distributional setting (where swap-regret is defined with respect to the distributions p^t themselves). In particular, an algorithm with lower dependence on N would imply a more efficient (in terms of number of rounds) procedure for achieving an approximate correlated equilibrium. Ideally, one would like to achieve approximate correlated equilibrium in a number of rounds that is only logarithmic in N, much as can be done for approximate minimax optimality in 2-player zero-sum games.

Acknowledgements

We would like to thank Gilles Stoltz for a number of helpful comments and suggestions.

References

Peter Auer, Nicolò Cesa-Bianchi, , and Claudio Gentile. Adaptive and self-confident on-line learning algorithms. *JCSS*, 64(1):48–75, 2002a. (A preliminary version has appeared in Proc. 13th Ann. Conf. Computational Learning Theory.).

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002b.

R. J. Aumann. Subjectivity and correlation in randomized strategies. *Journal of Mathematical Economics*, 1:67–96, 1974.

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

A. Blum. Empirical support for winnow and weighted-majority based algorithms: results on a calendar scheduling domain. *Machine Learning*, 26:5–23, 1997.

Nicolò Cesa-Bianchi, Yoav Freund, David P. Helmbold, David Haussler, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *Journal of the Association for Computing Machinery (JACM)*, 44(3):427–485, 1997.

Nicolò Cesa-Bianchi and Gábor Lugosi. Potential-based algorithms in on-line prediction and game theory. *Machine Learning*, 51(3):239–261, 2003.

Nicolò Cesa-Bianchi, Gábor Lugosi, and Gilles Stoltz. Regret minimization under partial monitoring. *Mathematics of Operations Research*, 31:562–580, 2006.

Nicolò Cesa-Bianchi, Yishay Mansour, and Gilles Stoltz. Improved second-order bounds for prediction with expert advice. In *Proceedings of the Eighteenth Annual Conference on Computational Learning Theory*, 2005.

W. Cohen and Y. Singer. Learning to query the web. In *AAAI Workshop on Internet-Based Information Systems*, 1996.

W. Cohen and Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems*, 17(2):141–173, 1999.

William Feller. *An introduction to probability theory and its applications. - Vol. 1.* Wiley, 1968.

William Feller. *An introduction to probability theory and its applications. - Vol. 2.* Wiley, 1971.

D. Foster and R. Vohra. Calibrated learning and correlated equilibrium. *Games and Economic Behavior*, 21:40–55, 1997.

D. Foster and R. Vohra. Asymptotic calibration. *Biometrika*, 85:379–390, 1998.

D. Foster and R. Vohra. Regret in the on-line decision problem. *Games and Economic Behavior*, 29:7–36, 1999.

Dean P. Foster and Rakesh V. Vohra. A randomization rule for selecting forecasts. *Operations Research*, 41(4):704–709, July–August 1993.

Y. Freund, R. Schapire, Y. Singer, and M. Warmuth. Using and combining predictors that specialize. In *Proceedings of the 29th Annual Symposium on Theory of Computing*, pages 334–343, 1997.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Euro-COLT*, pages 23–37. Springer-Verlag, 1995. Also, JCSS 55(1): 119-139 (1997).

Yoav Freund and Robert E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79–103, 1999. (A preliminary version appeared in the Proceedings of the Ninth Annual Conference on Computational Learning Theory, pages 325–332, 1996.).

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

S. Hart and A. Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68:1127–1150, 2000.

S. Hart and A. Mas-Colell. A reinforcement procedure leading to correlated equilibrium. In Wilhelm Neuefeind Gerard Debreu and Walter Trockel, editors, *Economic Essays*, pages 181–200. Springer, 2001.

E. Lehrer. A wide range no-regret theorem. *Games and Economic Behavior*, 42:101–115, 2003.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108:212–261, 1994.

Gilles Stoltz. *Incomplete information and internal regret in prediction of individual sequences*. PhD thesis, Dept. of Mathematics, University Paris XI, ORSAY, 2005.

Gilles Stoltz and Gábor Lugosi. Internal regret in on-line portfolio selection. *Machine Learning*, 59(1-2):125–159, 2005.

Gilles Stoltz and Gábor Lugosi. Learning correlated equilibria in games with compact sets of strategies. *Games and Economic Behavior*, 59:187–209, 2007.