**Computational Game Theory** **Spring Semester, 2005/6**

# Lecture 8: May 16, 2006

*Lecturer: Yishay Mansour* *Scribe: Lior Shapira, Eyal David* [1]

## 8.1 Regret Minimization

Lecture 7 dealt with repited games, in which each action was dependent upon a previous actions. In this Lecture, our goal is to build a strategy with good performance when dealing with repeated games. Let us start with a simple model of regret. In this model a player performs a partial optimization on his actions. Following each action he updates his belief an selects the next actions, dependent on the outcome.

## 8.2 Full Information Model

The model is defined as follows:

- Single player
- Actions A={$a_1$,..,$a_N$}
- For each step $t$ the player chooses an action $a_i$ (or a distribution $p^t$ over $A$)
- For each step $t$ we receive a loss $l^t$ where $l_i^t \in [0, 1]$ is the loss of action $i \in A$
- A player's loss at step $t$ is $\sum_{i=1}^{N} p_i^t l_i^t = l_{ON}^t$.
- Accumulative loss for a player is $L_{ON}^T = \sum_{t=1}^{T} \vec{l^t}\vec{p^t} = \sum_{t=1}^{T} l_{ON}^t$

Obviously the loss of a player can be maximized by choosing all losses to be 1. Therefore we must define a way to measure the players achievements. One way is choosing the best action at each step which results in a minimal loss $OPT = \sum_{t=1}^{T} min_i\{l_i^t\}$. This measure is similar to competitive online analysis and in our setting no interesting bound can be achieved.

[1]These notes are based in part on the scribe notes of Eitan Yaffe and Noa Bar-Yosef from 2003/2004

# 8.3 External Regret

Let

$$\begin{aligned} L_i^T &= \sum_{t=1}^{T} l_i^t \text{ , and the accumulated loss of the best action} \\ L_*^T &= min_i L_i^T \end{aligned}$$

We define the external regret $R = L_{ON}^T - L_*^T$ as a way to measure the algorithm's performance and we wish to minimize $R$. This reflects our desire to achieve performance close to the best static choice of action.

## 8.3.1 Minimizing External Regret - Greedy Algorithm

One way to minimize $R$ is by using a greedy algorithm:

- For convenience we'll assume $l_i^t \in \{0,1\}$ (so cumulative loss values will be integers)
- For the $t$ step, we will chose the best action until now, i.e.,

$$a^t = arg \min_i L_i^{t-1}$$

**Theorem 8.1** $L_{ON}^T \leq N \cdot L_*^T + (N-1)$

**Proof:** We define $c_k$ to be the loss of ON(the greedy algorithm) from time $t$, the first time in which $L_*^t = k$ and until time $t'$,the first time in which $L_*^{t'} = k+1$. At time $t$ there are at most $N$ actions with $L_i^t = k$. Each time ONLINE pays 1, the number of actions with a loss of $k$ is reduced by 1. Therefore

$$c_k \leq N \text{ which implies that } L_{ON} = \sum_{k=0}^{L_*^T} c_k \leq N \cdot L_*^T + (N-1)$$

□

**Theorem 8.2** *Each deterministic algorithm D has a series for which* $L_D^T \geq N \cdot L_*^T$

**Proof:** The opponent, at time $t$, defines a loss of 1 on $a^t$, the action that $D$ selects at time $t$ and 0 on the other actions. Algorithm $D$ pays exactly $L_D^T = T$. However, by averaging there is an action i, such that $L_i^T \leq \frac{T}{N}$. This occurs because T "losses" are divided between N actions. And so $L_D^T \geq N \cdot L_*^T$ □

## 8.4 Randomized Algorithms

### 8.4.1 MARK algorithm

Let $B^t = \{i | L_i^t = L_*^t\}$. At time $t+1$ we select $a_i^t$ at random such that $i \in B^t$. I.e.

$$p_i^{t+1} = \begin{cases} \frac{1}{|B^t|} & \text{if } i \in B^t \\ 0 & \text{otherwise} \end{cases}$$

**Claim 8.3** $L_{MARK} \leq (\ln N) \cdot L_*^T + \ln N - 1$

**Proof:** We define $c_k$ as before. We assume that the opponent choose to give a loss of 1 to one action out of $B^t$ (it is always better for the opponent to select out of $B^t$, and in addition it is obviously better to choose two actions in differing rounds rather than the same round). The expected loss for a round therefore is $\frac{1}{|B^t|}$ and so

$$E[c_k] = \sum_{i=1}^{N} \frac{1}{i} \leq \ln(N)$$

and therefore

$$L_{MARK} = E[\sum_{k=0}^{L_*^T} c_k] \leq \ln(N) L_*^T + \ln(N)^{-1} - 1$$

□

### 8.4.2 Weighted Majority algorithm

How can MARK be improved? we notice that performance suffers when $B^t$ is small and so we'll try giving actions a positive probability, even if they aren't in $B^t$.

- We define $w$ such that $\overline{w_i^t} = (\frac{1}{2})^{L_i^{t-1}}$, when initially $\overline{w_i^1} = 1$ (since $L_0^i = 0$)
- The $WM$ algorithm selects a distribution $p_i^t = \frac{w_i^t}{W^t}$ when $W^t = \sum_i \overline{w_i^t}$

The $WM$ algorithm is an exponential smoothing of the greedy algorithm. An action for which the loss is greater than the minimal, receives a probability it would have gotten otherwise which falls exponentially in the difference.

If $l_{WM}^t$ is WM's loss at time $t$ then

- $\overline{w_i^t} = (\frac{1}{2})^{L_i^{t-1} - \frac{1}{2} L_{WM}^{t-1}}$
- $\overline{w_i^{t+1}} = w_i^t (\frac{1}{2})^{l_i^t - \frac{1}{2} l_{WM}^t}$

**Claim 8.4** $0 \leq W^{t+1} \leq W^t \leq N$

**Proof:** By induction on $t$. It's clear that $W^1 \leq N$. We'll prove that $W^{t+1} \leq W^t$.

$$\begin{aligned}
W^{t+1} &= \sum_{i=1}^{N} w_i^{t+1} \\
&= \sum_{i=1}^{N} w_i^t (\frac{1}{2})^{l_i^t} \cdot (\frac{1}{2})^{-\frac{1}{2} l_{WM}^t} \\
&= \sum_{i=1}^{N} w_i^t \cdot 2^{-l_i^t} \cdot 2^{\frac{1}{2} l_{WM}^t} \\
&\leq \sum_{i=1}^{N} w_i^t (1 - \frac{1}{2} l_i^t)(1 + \frac{1}{2} l_{WM}^t) \\
&\leq \sum_{i=1}^{N} w_i^t - \frac{1}{2} \sum_{i=1}^{N} w_i^t l_i^t + \frac{1}{2} \sum_{i=1}^{N} w_i^t l_{WM}^t - ... \\
&= W^t - \frac{w^t}{2} \sum_{i=1}^{N} \frac{w_i^t}{w^t} l_i^t + \frac{1}{2} l_{WM}^t w^t \\
&= W^t - \frac{1}{2} w^t l_{WM}^t + \frac{1}{2} l_{WM}^t w^t = W^t
\end{aligned}$$

We used the linear interpolation showing that $2^{-x} \leq (1 - \frac{1}{2}x)$ and $2^{\frac{1}{2}x} \leq (1 + \frac{1}{2}x)$ for $x \in [0, 1]$
□

**Bound for WM**

From claim 8.4:

$$w_k^t \leq W^t \leq N$$

therefore we choose the best $k^*$ such that

$$2^{-L_k^* + \frac{1}{2} L_{WM}^t} = w_k^t \leq N$$

and therefore

$$\frac{1}{2} L_{WM}^t \leq L_{k^*}^T + \ln(N)$$

$$L_{WM}^t \leq 2 L_{k^*}^T + 2 \ln(N)$$

**Discussion**

As discussed before our goal is to have $L_{ON} \leq L_* + R$ such that $\frac{R}{T} \underset{T\to\infty}{\longrightarrow} 0$. One option is to change the parameter in the WM algorithm $\frac{1}{2}$ with $\beta$ and optimize its value. Using such an optimization we can achieve $R \sim \sqrt{T \log N}$

We present a different online algorithm which achieves

$$\begin{aligned} L_{ON} &\leq L_k + \sqrt{Q_k \ln(N)} + 2\ln(N) \\ Q_k &= \sum_{t=1} T(l_k^t)^2 \leq L_k \leq T \\ Since & \quad l_i^t \in [0,1] \end{aligned}$$

**Upper Bound (finite)**

Instead of losses we'll look at profits (which might be negative or positive). Therefore

$$g_i^t \in [-1, 1]$$

and

$$G_k^t = \sum_{t=1}^{T} g_k^t$$

and so

$$Q_k = \sum_{t=1}^{T} (g_k^t)^2$$

The weights determine the algorithm (same as WM)

$$\begin{aligned} w_i^{t+1} &= w_i^t(1 + \eta g_i^t) \\ w_i^0 &= 1 \end{aligned}$$

The intuition behind this is such that the weight of an action will be exponential in it's profit. For instance if the profit is always 1, the weight will be $(1 + \eta)^T$ and if the profit is always -1 it will be $(1 - \eta)^T$.

**Theorem 8.5**

$$G_{ON}^T \geq G_k^T - \sqrt{Q_k \ln(N)} - 2\ln(N)$$

**Proof:** We bound $\ln \frac{W^T}{W^1}$ from both sides, where $w^t = \sum_{i=1}^{N} w_i^t$ for each $k$.

$$\begin{aligned}
\ln \frac{w^T}{w^1} &\geq \ln \frac{w_k^T}{N} \\
&\quad \text{(From the recursive definition of weights)} \\
&= -\ln(N) + \sum_{t=1}^{T} \ln(1+\eta g_k^t) \\
&\quad \text{(We use the inequality } \ln(1+z) \geq z - z^2 \text{ for } -\tfrac{1}{2} \leq z \leq \tfrac{1}{2}) \\
&\geq -\ln(N) + \sum_{t=1}^{T} \eta g_k^t - \sum_{t=1}^{T} (\eta g_k^t)^2 \\
&= -\ln(N) + \eta G_k^T - \eta^2 Q_k^T \\
&\quad \text{On the other hand...} \\
\ln \frac{W^T}{W^1} &= \sum_{t=1}^{T-1} \ln \frac{W^{t+1}}{W^t} \\
&= \sum_{t=1}^{T-1} \ln[\sum_{i=1}^{N} \frac{w_i^t(1+\eta g_i^t)}{w^t}] \\
&= \sum_{t=1}^{T-1} \ln[\sum_{i=1}^{N} p_i^t(1+\eta g_i^t)] \\
&= \sum_{t=1}^{T-1} \ln[1+\eta \sum_{i=1}^{N} p_i^t g_i^t)] \\
&= \sum_{t=1}^{T-1} \ln[1+\eta g_{ON}^t] \\
\text{(Inequality } \ln(1+z) \leq z) & \\
&\leq \sum_{t=1}^{T-1} \eta g_{ON}^t \\
&= \eta G_{ON}^{T-1}
\end{aligned}$$

Therefore, by combining the bounds, we get

$$\eta G_{ON}^T \geq \eta G_k^T - \eta^2 Q_k^T - \ln N \tag{8.1}$$

or alternatively,

$$G_{ON}^T \geq G_k^T - \eta Q_k^T - \frac{\ln N}{\eta}$$

We set $\eta = \min\{\sqrt{\frac{\ln N}{Q_k^T}}, \frac{1}{2}\}$ and then

$$G_{ON}^T \geq G_k^T - 2\sqrt{Q_k \ln N}$$

Or if $Q_k$ is small

$$G_{ON}^T \geq G_k^T - 2Q_k - 2\ln N$$

Finally

$$R \leq 2\sqrt{Q_k \ln N} \leq \sqrt{T \ln(N)}$$

And therefore

$$\frac{R}{T} = \sqrt{\frac{\ln(N)}{T}} \underset{T \to \infty}{\longrightarrow} 0$$

$\square$

**Lower Bound**

We will discuss 2 aspects of the lower bounds for regret minimization:

1. For $N$ Actions and time $T = \frac{1}{2}\log N$ we will show a lower bound of $R = \Omega(\log N)$. We will assume that for each action we have a cost of 1 with probability $\frac{1}{2}$ and a cost of 0 with probability $\frac{1}{2}$.
The probability to have at time $T$ an action $i$ with $L_i^T = 0$ (an action with 0 loss) is:

$$1 - (1 - (\tfrac{1}{2})^T)^N = 1 - (1 - \tfrac{1}{\sqrt{N}})^N \approx 1 - e^{-\sqrt{N}}$$

For a very large $N$, the expected loss of $L_x$ is boundedly,

$$E[L_*^T] \leq e^{-\sqrt{N}}\tfrac{1}{2}\log N$$

And since for every $ONLINE$ we have $E[L_{ON}] = \frac{1}{2}T$ we get for any algorithm $R$:

$$E[L_R^T] = \tfrac{1}{4}\log N$$

2. For two actions and time $T$ we choose a cost of $(1, 0)$ with probability $\frac{1}{2}$ and a cost of $(0, 1)$ with probability $\frac{1}{2}$.
The ONLINE algorithm loses $\frac{T}{2}$ on average. Because the probabilities are set in advance and are constant over time, when we choose the best possible action the result is around the expected value: $\frac{T}{2} - \Theta(\sqrt{N})$ and we get:

$$Regret = \Omega(\sqrt{T})$$

# 8.5 Partial Information Model

In this game the player chooses a **single action** $a^t$ based on some distribution $p^t$. The opponent then sets the prices $l^t$ based on $p^t$. The player then pays $l_{a^t}$.

## 8.5.1 A simple reduction

We will divide our game into $T/k$ blocks of size $k$, denoted by $X^1 ... X^{T/k}$. Within each group or block of actions $X^j$ we will sample every action $i$ once.

| $k$ | $k$ | $k$ | $k$ | $k$ | $k$ |
|---|---|---|---|---|---|
| $\|l^1 \quad l^k$ | $\|l^1 \quad l^k$ | $\|l^1 \quad l^k$ | $\|l^1 \quad l^k$ | $\|l^1 \quad l^k$ | $\|l^1 \quad l^k\|$ |
| $X^1$ | | | | | $X^{T/k}$ |

At the end of block $X^j$, we gather the loss of the $N$ sampled actions $l_i^j \cdots l_n^j$ and give it to a full information algorithm $ER$. The algorithm returns a distribution $p^{j+1}$, which we use in block $X^{j+1}$ during the non-sampling steps.
Namely,

$$ER(X^1 ... X^t) \longmapsto p^{t+1}$$

The $ER$ algorithm will give us for every action $i \in A$,

$$\sum_{\tau=1}^{T/k} p^\tau \cdot X^\tau \leq \sum_{\tau=1}^{T/k} X_i^\tau + \sqrt{\frac{T}{k} \log N}$$

Now we compute the expected value of $X$:

$$E[X_i^\tau] = \frac{1}{k} \sum_{t \in X^\tau} l_i^t$$

And therefor we have:

$$E[\sum_{\tau=1}^{T/k} p^\tau \cdot X^\tau] \leq E[\sum_{\tau=1}^{T/k} X_i^\tau] + \sqrt{\frac{T}{k} \log N}$$

$$\Downarrow$$

$$\sum_{\tau=1}^{T/k} \frac{1}{k} \sum_{t \in k_\tau} l^t \cdot E[p^\tau(x^1 \cdots x^{t-1})] \leq \frac{1}{k} \sum_{t=1}^{T} l_i^t + \sqrt{\frac{T}{k} \log N}$$

$$\Downarrow$$

$$E[ONLINE] \leq L_i^T + \sqrt{KT \log N} + \tfrac{T}{k} \cdot N.$$

We have an $\frac{T}{k} \cdot N$ sampling cost.
We can optimize this result over $k$ and have:

$$k \cong T^{\frac{1}{3}} N^{\frac{2}{3}} \text{ ,and}$$
$$Regret \sim T^{\frac{2}{3}} N^{\frac{2}{3}}$$