Week 5 - Regret Minimization

Introduction to the Regret Concept

- We will consider problem of repeatedly making decision in an uncertain environment.
- We will have:
  - N actions to choose from. (For example rock, paper, scissors)
  - Multiple time steps
  - Algorithm, that probabilistically chooses an action at each times step
  - The environment that makes its "move".
  - The loss function for the each action given by the environment.
- Examples:
  - Choosing the best expert.
  - Choosing the road to the work.
  - Choosing the best strategy in game.

Introduction to the Regret Concept

- We would like to have some guarantees for our policy for action selection
- Even if the loss is not know in the advance and can be chosen arbitrary by the environment.
- We will use notation of the regret.
- The regret tells us, how much could we improve if we have used some alternative policy in **retrospect**.
- The **External regret** compares our performance to the best single in retrospect.

Model Formalization II

- The agent's set of actions is $\mathcal{A}$
- At each time step t the agent chooses a policy $\pi^t \in \Delta(\mathcal{A})$.
- The agent then receives a reward vector $x_t \in \mathcal{R}^{|\mathcal{A}|}$
- The agent's value is the weighted sum $v^t = \sum_{a \in \mathcal{A}} \pi^t(a)\, x^t(a)$.

Model Formalization III

- The cumulative reward up to time T is simply $X_{\pi}^{T} = \sum_{t=1}^{T} \pi^{t} x^{t\top}$.
- External regret of action a, R_a^T then contrasts this to a cumulative reward that would have been received if we rather followed action a at each time step $R_a^T = \sum_{t=1}^{T} x_t(a) - X_{\pi}^{T}$.
- Finally, external regret is then $R^T = \max_{a \in \mathcal{A}} R_a^T$.

Regret with Respect to Optimal Sequence

- Why has to be comparison class G restricted?
- It is not possible to guarantee low regret with respect to the overall optimal sequence of decisions!
- Let G_{all} be the set of all functions mapping times $1 \ldots T$ to actions $\mathcal{A} = 1 \ldots N$.

Theorem

For any online algorithm H there exists a sequence of rewards $x^1 \ldots x^T$ such that the regret $R_{G_{all}}$ is at least $T(1 - 1/N)$.

Deterministic H

- An deterministic algorithm D can at each time step t choose only one action i with $p_i^t = 1$, and assign zero probability to other actions
- First idea - Lets take the action which had the lowest observed loss so far.
- We obtain a greedy algorithm.
- If we use this algorithm for the each player in some game, we play repeatedly, we get the fictitious play algorithm.
- Has greed algorithm any guarantee of low external regret?
- And what about other deterministic algorithms?

Bound on loss of Deterministic Algorithm

- We cant guarantee a low regret with the greedy algorithm.
- In fact for any deterministic algorithm, the loss can be very large.

Theorem

For any deterministic algorithm D, there exists a loss sequence for which $L_D^T = T$ and $L_{min}^T = T/N$.

Lower Bounds for Arbitrary Stochastic Algorithm

- Can the stochastic algorithms do better?
- We will see the lower bounds first.

Theorem

Consider $T < log_2 N$. There exist a stochastic generation of losses such that, for any online algorithm H, we have $E[L_H^T] = T/2$ and $L_{min}^T = 0$.

Theorem

Consider $N = 2$. There exists a stochastic generation of losses such hat for any online algorithm H we have $E[L^T H - L_{min}^T] = \Omega(\sqrt{(}T))$.

Regret matching

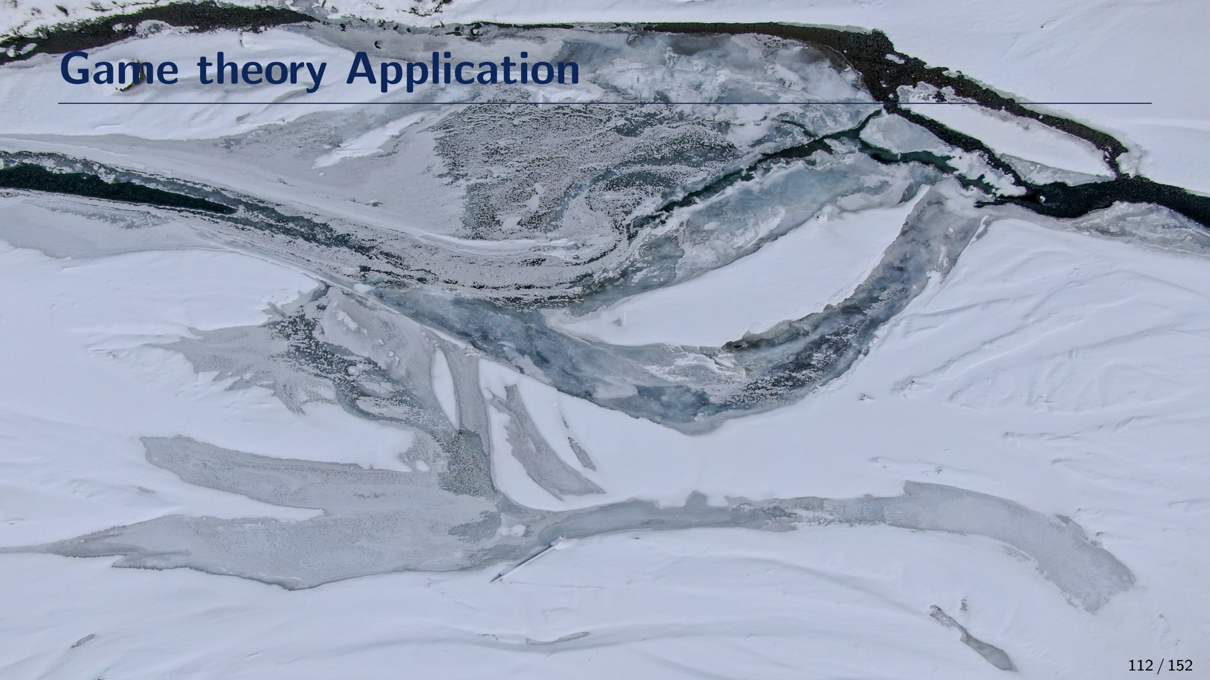
- We would like to have same algorithm with regret close to these bounds.
- The algorithm is surprisingly simple.
- We define the regret of action i at time t as $R_i^t = \sum_{t'=0}^{t-1} l_h^t - l_i^t$.
- We define the positive faction of the regret of action i at time t as $R_i^{t,+} = max(R_i^t, 0)$.
- Algorithm will then choose all actions with no-zero regret, with probability proportional to their positive fraction of the regret.
- $p_i^t = \frac{R_i^(t,+)}{\sum_i R_i^{t,+}}$ if $\sum_i R_i^{t,+} > 0$, $\frac{1}{N}$ otherwise.

Theorem

If we select actions according the regret matching algorithm, over regret is bounded by $O\sqrt(N)\sqrt(T)$.

For the proof see:
`http://www.cs.cmu.edu/~ggordon/ggordon.CMU-CALD-05-112.no-regret.pdf`

Game theory Application



Game theory Application

- Consider normal form game with standard notation.

Definition

The **average** regret of player i at time T is
$R_i^T = \frac{1}{T} max_{\sigma_i^* \in \Sigma_i} \sum_{t=1}^{T} (u_i(\sigma *_i, \sigma_{-1}^t) - u_i(\sigma_t))$

Theorem

If player i selects his actions according the regret matching algorithm, then his average regret is smaller than $\Delta_{u,i}\sqrt{|A_{-i}|}/\sqrt{T}$.

Theorem

In a zero-sum game at time T, if both player's average overall regret is less than ϵ then the **average** strategy profile $\overline{\sigma}$ at time T is a 2ϵ equilibrium.

Convergence in Two Player Zero Sum Game

Theorem

In a two player zero-sum game at time T, if both player's average overall regret is less than ϵ then the **average** strategy profile $\overline{\sigma}$ at time T is a 2ϵ equilibrium.

- Recall the notion of ϵ-Nash equilibrium:

Definition

A strategy profile σ^* it said to be a ϵ-**Nash equilibrium** if for all players i and each his alternate strategy $\sigma_i^{\prime}$, we have that:

$$u_i((\sigma_i^*, \sigma_{-i}^*)) \geq u_i((\sigma_i^{\prime}, \sigma_{-i}^*)) - \epsilon$$

Proof of Convergence

- Let σ_1' be an arbitrary strategy. Since both players have external regret lover than ϵ, we have:

$$\frac{1}{T}\sum_t u_1(\sigma^t) \geq \frac{1}{T}\sum_t u_1(\sigma_1', \sigma_2^t) - \epsilon$$

$$\frac{1}{T}\sum_t u_2(\sigma^t) \geq \frac{1}{T}\sum_t u_2(\sigma_1^t, \overline{\sigma_2}) - \epsilon$$

- We can rewrite these two equations using the property of average strategy:

$$\frac{1}{T}\sum_t u_1(\sigma^t) \geq u_1(\sigma_1', \overline{\sigma_2}) - \epsilon$$

$$\frac{1}{T}\sum_t u_2(\sigma^t) \geq u_2(\overline{\sigma_1}, \overline{\sigma_2}) - \epsilon$$

Proof Continuation I

$$\frac{1}{T}\sum_t u_1(\sigma^t) \geq u_1(\sigma_1', \overline{\sigma_2}) - \epsilon$$

$$\frac{1}{T}\sum_t u_2(\sigma^t) \geq u_2(\overline{\sigma_1}, \overline{\sigma_2}) - \epsilon$$

- Now we will use our assumption, that the game is zero sum. Therefore we have $u_2(\sigma^t) = -u_1(\sigma^t)$. We can tie the both equations together:

$$u_1(\overline{\sigma_1}, \overline{\sigma_2}) + \epsilon \geq \frac{1}{T}\sum_t u_1(\sigma^t) \geq u_1(\sigma_1', \overline{\sigma_2}) - \epsilon$$

- And finally:

$$u_1(\overline{\sigma_1}, \overline{\sigma_2}) \geq u_1(\sigma_1', \overline{\sigma_2}) - 2\epsilon$$

Proof Continuation II

- We know that for any σ_1' we have:

$$u_1(\overline{\sigma_1}, \overline{\sigma_2}) \geq u_1(\sigma_1', \overline{\sigma_2}) - 2\epsilon$$

- When we use the same approach for the second player, we get for that any σ_2' :

$$u_2(\overline{\sigma_1}, \overline{\sigma_2}) \geq u_2(\overline{\sigma_1}, \sigma_2') - 2\epsilon$$

- That makes the strategy profile $(\overline{\sigma_1}, \overline{\sigma_2})$ 2ϵ-nash equilibrium !

Solving Games with Regret Minimization

- We have now the algorithm for solving normal-form games!
- We can choose arbitrary regret minimization algorithm (like regret matching) and let both players to play according the algorithm.
- If we choose regret matching, the asymptotic average regret for each player after T iterations is $O\frac{1}{\sqrt{T}}$.
- When we then take the average strategies for both players, we have $O\frac{1}{\sqrt{T}}$ equilibrium.
- To get the fixed ϵ, we need $O\frac{1}{\epsilon^2}$ iterations of the regret minimization.

Properties of the algorithm

- Very easy to implement.
- Each player needs only to remember regret and average strategy for **his** actions.
- Players do not even have to know the payoff matrix - it does not have to be represented in the memory
- If the second player does not play with some regret minimization algortim, the first player earns as much as the best response to his average strategy in the limit.

Convergence Notes

- We mentioned that the $O\frac{1}{\sqrt{T}}$ is optimal in the general setting.
- But, in zero sum games, both player can cooperate to solve the game.
- They can therefore achieve smaller regret.
- There is know algorithm with $O\frac{ln(T)}{T}$ regret for both players, with assumption that they don't know the payoff matrix at the beginning. `http://dl.acm.org/citation.cfm?id=2133057`
- But the iterations are too slow for the practical use.

Week 5 Homework

1. Implement regret minimization in self-play settings
2. Remember that this is similar to fictitious self-play, except that each player computes their strategy using regret minimization method rather than a best-response
3. Plot the exploitability of the average strategy as well as the current one