Motivation - Fishing spots

Setup
- Pick a fishing spot (at the beginning of the day)
- Obtain some amount of fish (at the end of the day)
- Observe other fishermen (noisy side observations)
  - Does not represent what you would get exactly
  - Different fishing lure, motivation, skill...
- Goal: catch as many fish as possible over the time

Side observations - prior work (Mannor and Shamir 2011)
- Playing an action reveals the losses of neighboring spots

Noisy side observations - generalization
- Side observations represented by a weighted graph
- Playing action reveals noisy the losses of neighboring spots

Problem formalization
Learning process
- N actions (nodes of a graph)
- T rounds:
  - Environment (adversary) sets losses for actions
  - Environment makes a graph (not disclosed)
  - Learner picks an action \( i \) to play
  - Learner incurs the loss \( l_{i,j} \) of the action \( i \)
  - Learner observes graph (second neighborhood of \( i \))
  - Learner observes noisy losses \( \ell_{i,j} \) of neighbors \( j \in N(i) \)

\[
\ell_{i,j} = s_{i,j} + (1 - s_{i,j})\xi_{i,j}
\]

where
- \( s_{i,j} \): weight of the edge from node \( i \) to node \( j \) at time \( t \)
- \( \ell_{i,j} \): loss of the action \( j \) at time \( t \)
- \( \xi_{i,j} \): zero mean noise such that \( |\xi_{i,j}| \leq R \)

Goal of the learner: minimizing cumulative regret \( R_T \) defined as

\[
R_T = \sum_{t=1}^{T} \ell_{i,t} - \min_{\{0, 1\}} \sum_{t=1}^{T} \ell_{i,t}
\]

Exp3-type algorithm template
- Compute exponential weights using loss estimates \( \hat{l}_{i,t} \)

\[
\hat{l}_{i,t} = \exp\left(-\frac{1}{\tau} \sum_{t=1}^{T} \ell_{i,j}ight)
\]

- Create a probability distribution such that \( p_{i,j} \propto \hat{l}_{i,j} \)
- Play action \( i \) such that

\[
P(i = j) = p_{i,j} = \frac{\hat{l}_{i,j}}{\sum_{i,j} \hat{l}_{i,j}}
\]

- Create loss estimates (using observability graph)
  - The definition of \( \ell \) defines an Exp3-type algorithm

Loss estimates of algorithms

Desired property of loss estimates: \( \mathbb{E}[\hat{l}_{i,t}] = l_{i,t} \)

Typical estimates:

\[
\hat{l}_{i,t} = \frac{1}{\tau} \sum_{t=1}^{T} \ell_{i,j}
\]

Graphs without weights (known algorithms)

Exp3 (edgeless graph)

\[
\hat{l}_{i,t} = \frac{1}{\tau} \sum_{t=1}^{T} \ell_{i,j} \text{ (arm } i \text{ is observed)}
\]

Hedge (full graph)

\[
\hat{l}_{i,t} = \frac{1}{\tau} \sum_{t=1}^{T} \ell_{i,j} \text{ (arm } i \text{ is observed)}
\]

Exp3-IX (general graph)

\[
\hat{l}_{i,t} = \frac{1}{\tau} \sum_{t=1}^{T} \ell_{i,j} \text{ (arm } i \text{ is observed)}
\]

Graphs with weights (new algorithms)

Exp3-IXt (general graphs with weights and thresholding)
- Use only “reliable” observations with low noise
- Delete all the edges with weights smaller than \( \epsilon \)

Exp3-WIX (general graph with weights)

Special cases of the framework

- Edgeless graphs - Bandit setting
- General graphs - Mannor and Shamir 2011
- Complete graphs - Full information setting

Experimental results

- Exp3 - basic algorithm which ignores all side observations
- Exp3-WIX - our proposed algorithm
- Exp3-IXt - thresholded algorithm (needs to set \( \epsilon \))
- Exp3-IXW - algorithm ignores noise (no guarantees)

Conclusion
- New setting with noisy side observations
- Introduction of effective independence number \( \alpha^* \)
- Exp3-WIX algorithm for the setting
  - Does not need to threshold
  - Does not need to know whole graph
  - Regret bound of order \( \sqrt{NT} \)
- Open questions:
  - Is the effective independence number “right quantity?”
  - Is there a matching lower-bound for Exp3-WIX?
  - Upper-bound of Exp3-WIX matches lower-bound for some cases (e.g., bandits, full information, and setting of Mannor and Shamir 2011)
  - Related lower-bound (Wu et al. 2015) for a stochastic setting with Gaussian noise

\[
R_T = \Omega\left(\sqrt{\frac{NT}{\epsilon}}\right)
\]