

# 15-859(B) Machine Learning Theory

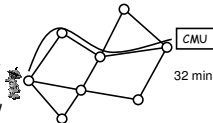
Bandit Problems, Game Theory,  
Connections between online learning  
and GT

Avrim Blum

Start with recap

## Consider the following setting...

- w Each morning, you need to pick one of  $N$  possible routes to drive to work.
- w But traffic is different each day.
  - n Not clear a priori which will be best.
  - n When you get there you find out how long your route took. (And maybe others too or maybe not.)
- w Want a strategy for picking routes so that in the long run, whatever the sequence of traffic patterns has been, you've done nearly as well as the best fixed route in hindsight. (In expectation, over internal randomness in the algorithm)



## "No-regret" algorithms for repeated decisions

General framework:

- w Algorithm has  $N$  options. World chooses cost vector. Can view as matrix like this (maybe infinite # cols)

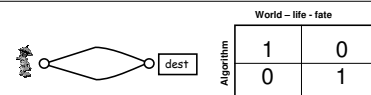
Algorithm	World - life - fate	

- w At each time step, algorithm picks row, life picks column.
  - n Alg pays cost for action chosen.
  - n Alg gets column as feedback (or just its own cost in the "bandit" model).
  - n Need to assume some bound on max cost. Let's say all costs between 0 and 1.

## "No-regret" algorithms for repeated decisions

- w At each time step, algorithm picks row, life picks column.
- Define **average regret** in  $T$  time steps as:  
(avg per-day cost of algo) - (avg per-day cost of best fixed row in hindsight).
- Alg gets column as feedback (or just its own cost in the "bandit" model).
- We want this to go to 0 or better as  $T$  gets large.
- Need to assume some bound on max cost. Let's say all costs between 0 and 1.
- [called a "no-regret" algorithm]

## To be clear...



- w View of world/life/fate: unknown sequence LRLRLRR...
- w Goal: do well (in expectation) no matter what the sequence is.
- w Note: Not trying to compete with best adaptive strategy - just best fixed choice in hindsight.
- w Algorithms must be randomized or else it's hopeless.
- w No-regret algorithms can do much better than playing minimax optimal, and never much worse.

## History and development (abridged)

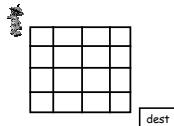
- w [Hannan'57, Blackwell'56]: Alg. with regret  $O((N/T)^{1/2})$ .
  - n Re-phrasing, need only  $T = O(N/\epsilon^2)$  steps to get time-average regret down to  $\epsilon$ . (will call this quantity  $T_\epsilon$ )
  - n Optimal dependence on  $T$  (or  $\epsilon$ ). Game-theorists viewed #rows  $N$  as constant, not so important as  $T$ , so pretty much done.

## History and development (abridged)

- w [Hannan'57, Blackwell'56]: Alg. with regret  $O((N/T)^{1/2})$ .
  - n Re-phrasing, need only  $T = O(N/\epsilon^2)$  steps to get time-average regret down to  $\epsilon$ . (will call this quantity  $T_\epsilon$ )
  - n Optimal dependence on  $T$  (or  $\epsilon$ ). Game-theorists viewed #rows  $N$  as constant, not so important as  $T$ , so pretty much done.
- w Learning-theory 80s-90s: "combining expert advice". Imagine large class  $C$  of  $N$  prediction rules.
  - n Perform (nearly) as well as best  $f \in C$ .
  - n [LittlestoneWarmuth'89]: Weighted-majority algorithm
    - 1  $E[\text{cost}] \leq \text{OPT}(1+\epsilon) + (\log N)/\epsilon$ .
    - 1 Regret  $O((\log N)/T)^{1/2}$ .  $T_\epsilon = O((\log N)/\epsilon^2)$ .
  - n Optimal as fn of  $N$  too, plus lots of work on exact constants, 2<sup>nd</sup> order terms, etc. [CFHHSW93]...
- w Extensions to bandit model (adds extra factor of  $N$ ).

## Efficient implicit implementation for large $N$ ...

- w Bounds have only log dependence on # choices  $N$ .
- w So, conceivably can do well when  $N$  is exponential in natural problem size, if only could implement efficiently.
- w E.g., case of paths...



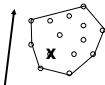
- w  $n \times n$  grid has  $N = (2n \text{ choose } n)$  possible paths.
- w Recent years: series of results giving efficient implementation/alternatives in various settings, plus extensions to bandit model.

## Efficient implicit implementation for large $N$ ...

- w Recent years: series of results giving efficient implementation/alternatives in various settings:
  - n [HelmboldSchapire97]: best pruning of given DT.
  - n [BChawlaKalai02]: list-update problem.
  - n [TakimotoWarmuth02]: online shortest path in DAGs.
  - n [KalaiVempala03]: elegant setting generalizing all above
    - 1 Online linear optimization
  - n [Zinkevich03]: elegant setting generalizing all above
    - 1 Online convex optimization
  - n [AwerbuchKleinberg04][McMahanB04]: [KV]  $\rightarrow$  bandit model
  - n [Kleinberg,FlaxmanKalaiMcMahan05]: [Z03]  $\rightarrow$  bandit model
  - n [DaniHayes06]: improve bandit convergence rate
  - n [GolovinStreeter08]: online submodular fn maximization
- More...

## [Kalai-Vempala'03] and [Zinkevich'03] settings

- [KV] setting:
  - w Implicit set  $S$  of feasible points in  $\mathbb{R}^m$ . (E.g.,  $m = \# \text{edges}$ ,  $S = \{\text{indicator vectors } 011010010 \text{ for possible paths}\}$ )
  - w Assume have oracle for offline problem: given vector  $c$ , find  $x \in S$  to minimize  $c \cdot x$ . (E.g., shortest path algorithm)
  - w Use to solve online problem: on day  $i$ , must pick  $x_i \in S$  before  $c_i$  is given.
  - w  $(c_1 \cdot x_1 + \dots + c_T \cdot x_T) / T \rightarrow \min_{x \in S} (c_1 + \dots + c_T) / T$ .
- [Z] setting:
  - w Assume  $S$  is convex.
  - w Allow  $c(x)$  to be a convex function over  $S$ .
  - w Assume given any  $y$  not in  $S$ , can algorithmically find nearest  $x \in S$ .



## Plan for today and next time

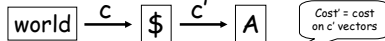
- w Bandit algorithms
- w Sleeping experts
- w Game theory
- w Connections between online learning and game theory
- w But first, a quick discussion of  $[0,1]$  vs  $\{0,1\}$  costs for RWM algorithm

## [0,1] costs vs {0,1} costs.

We analyzed Randomized Wtd Majority for case that all costs in {0,1} (correct or mistake).

Here is a simple way to extend to [0,1].

- w Given cost vector  $c$ , view  $c_i$  as bias of coin. Flip to create boolean vector  $c'$ , s.t.  $E[c'_i] = c_i$ . Feed  $c'$  to alg A.



- w For any sequence of vectors  $c'$ , we have:
    - o  $E_A[\text{cost}'(A)] \leq \min_i \text{cost}'(i) + [\text{regret term}]$
  - w So,  $E_{\$}[E_A[\text{cost}'(A)]] \leq E_{\$}[\min_i \text{cost}'(i)] + [\text{regret term}]$
  - w LHS is  $E_A[\text{cost}(A)]$ .
  - w  $\text{RHS} \leq \min_i E_{\$}[\text{cost}'(i)] + [\text{r.t.}] = \min_i [c_i] + [\text{r.t.}]$
- In other words, costs between 0 and 1 just make the problem easier...

## Experts → Bandit setting

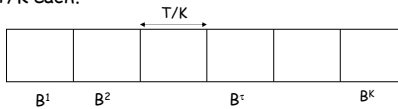
- w In the bandit setting, only get feedback for the action we choose. Still want to compete with best action in hindsight.
- w [ACFS02] give algorithm with cumulative regret  $O((TN \log N)^{1/2})$ . [average regret  $O((N \log N)/T)^{1/2}$ ].
- w Here, will give more generic, simpler approach but with worse bounds ( $T^{1/2} \rightarrow T^{2/3}$ ).

## Experts → Bandit setting

Given: algorithm A for full-info setting with regret  $\leq R(T)$ .  
Goal: use in black-box manner for bandit problem.

Preliminaries:

- w First, suppose we break our  $T$  time steps into  $K$  blocks of size  $T/K$  each.

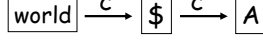


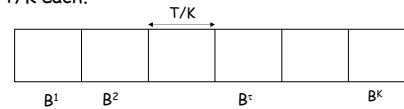
- w Use same distrib throughout block and update based on average cost vector  $c^*$  for block  $\tau$ .
- w Then, will get regret  $\leq R(K) \times T/K$ . Because really paying  $T/K \times c^*$  per block
- w What if we instead update on cost vector  $c' \in [0,1]^N$  that's a random variable whose expectation is correct?

## Experts → Bandit setting

Given: algorithm A for full-info setting with regret  $\leq R(T)$ .  
Goal: use in black-box manner for bandit problem.

Preliminaries:

- w First, suppose  blocks of size  $T/K$  each.

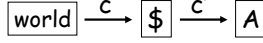


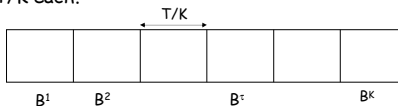
- w Use same distrib throughout block and update based on average cost vector  $c^*$  for block  $\tau$ .
- w Then, will get regret  $\leq R(K) \times T/K$ . Because really paying  $T/K \times c^*$  per block
- w What if we instead update on cost vector  $c' \in [0,1]^N$  that's a random variable whose expectation is correct?

## Experts → Bandit setting

Given: algorithm A for full-info setting with regret  $\leq R(T)$ .  
Goal: use in black-box manner for bandit problem.

Preliminaries:

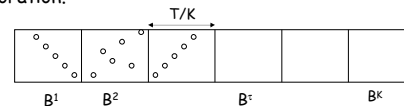
- w First, suppose  blocks of size  $T/K$  each.



- w Do at least as well by {0,1}→[0,1] argument. Still get regret bound  $R(K) \times T/K$ .
- w How does this help us for bandit problem?
- w What if we instead update on cost vector  $c' \in [0,1]^N$  that's a random variable whose expectation is correct?

## Experts → Bandit setting

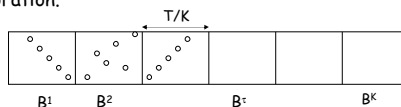
- w For bandit problem, for each action, pick random time step in each block to try it as "exploration".
- w Define  $c'$  only wrt these exploration steps.
- w Just have to pay an extra at most  $NK$  for cost of this exploration.



- w Do at least as well by {0,1}→[0,1] argument. Still get regret bound  $R(K) \times T/K$ .
- w How does this help us for bandit problem?
- w What if we instead update on cost vector  $c' \in [0,1]^N$  that's a random variable whose expectation is correct?

## Experts → Bandit setting

- For bandit problem, for each action, pick random time step in each block to try it as "exploration".
- Define  $c'$  only wrt these exploration steps.
- Just have to pay an extra at most  $NK$  for cost of this exploration.



- Final bound:  $R(K) \times T/K + NK$ .
- Using  $K = (T/N)^{2/3}$  and bound from RWM, get cumulative regret bound of  $O(T^{2/3} N^{1/3} \log N)$ .

## A natural generalization

- A natural generalization of our regret goal is: what if we also want that on rainy days, we do nearly as well as the best route for rainy days.
- And on Mondays, do nearly as well as best route for Mondays.
- More generally, have  $N$  "rules" (on Monday, use path  $P$ ). Goal: simultaneously, for each rule  $i$ , guarantee to do nearly as well as it *on the time steps in which it fires*.
- For all  $i$ , want  $E[\text{cost}_i(\text{alg})] \leq (1+\epsilon)\text{cost}_i(i) + O(\epsilon^{-1} \log N)$ .  
( $\text{cost}_i(X)$  = cost of  $X$  on time steps where rule  $i$  fires.)
- Can we get this? (Going back to full-info setting)

## A natural generalization

- This generalization is esp natural in machine learning for combining multiple if-then rules.
- E.g., document classification. Rule: "if <word- $X$ > appears then predict < $Y$ >". E.g., if has football then classify as sports.
- So, if 90% of documents with football *are* about sports, we should have error  $\leq 11\%$  on them.

"Specialists" or "sleeping experts" problem.

- Assume we have  $N$  rules, explicitly given.
- For all  $i$ , want  $E[\text{cost}_i(\text{alg})] \leq (1+\epsilon)\text{cost}_i(i) + O(\epsilon^{-1} \log N)$ .  
( $\text{cost}_i(X)$  = cost of  $X$  on time steps where rule  $i$  fires.)

## A simple algorithm and analysis (all on one slide)

- Start with all rules at weight 1.
- At each time step, of the rules  $i$  that fire, select one with probability  $p_i \propto w_i$ .
- Update weights:
  - If didn't fire, leave weight alone.
  - If did fire, raise or lower depending on performance compared to weighted average:
    - $r_i = [\sum_j p_j \text{cost}(j)] / (1+\epsilon) - \text{cost}(i)$
    - $w_i \leftarrow w_i (1+\epsilon)^{r_i}$
  - So, if rule  $i$  does exactly as well as weighted average, its weight drops a little. Weight increases if does better than weighted average by more than a  $(1+\epsilon)$  factor. This ensures sum of weights doesn't increase.
- Final  $w_i = (1+\epsilon)^{E[\text{cost}_i(\text{alg})] / (1+\epsilon) - \text{cost}_i(i)}$ . So, exponent  $\leq \epsilon^{-1} \log N$ .
- So,  $E[\text{cost}_i(\text{alg})] \leq (1+\epsilon)\text{cost}_i(i) + O(\epsilon^{-1} \log N)$ .

## Next Topic: Game Theory

## 2-Player Zero-Sum games

- Two players  $R$  and  $C$ . Zero-sum means that what's good for one is bad for the other.
- Game defined by matrix with a row for each of  $R$ 's options and a column for each of  $C$ 's options. Matrix tells who wins how much.
  - an entry  $(x,y)$  means:  $x$  = payoff to row player,  $y$  = payoff to column player. "Zero sum" means that  $y = -x$ .
- E.g., penalty shot:

		Left	Right	goalie
shooter	Left	(0,0)	(1,-1)	GOAALL!!!
	Right	(1,-1)	(0,0)	No goal

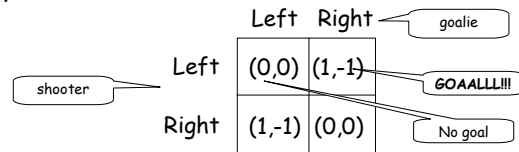
### Game Theory terminology

- Rows and columns are called pure strategies.
- Randomized algs called mixed strategies.
- "Zero sum" means that game is purely competitive.  $(x,y)$  satisfies  $x+y=0$ . (Game doesn't have to be fair).



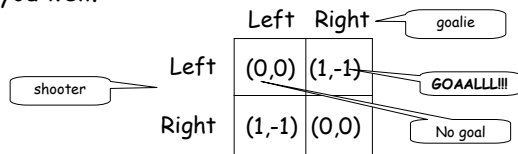
### Minimax-optimal strategies

- Minimax optimal strategy is a (randomized) strategy that has the best guarantee on its expected gain, over choices of the opponent. [maximizes the minimum]
- I.e., the thing to play if your opponent knows you well.



### Minimax-optimal strategies

- Can solve for minimax-optimal strategies using Linear programming
- No-regret strategies will do nearly as well or better.
- I.e., the thing to play if your opponent knows you well.



### Minimax Theorem (von Neumann 1928)

- Every 2-player zero-sum game has a unique value  $V$ .
- Minimax optimal strategy for  $R$  guarantees  $R$ 's expected gain at least  $V$ .
- Minimax optimal strategy for  $C$  guarantees  $C$ 's expected loss at most  $V$ .

Existence of no-regret strategies gives one way of proving the theorem.

Can use notion of minimax optimality to explain bluffing in poker

### Simplified Poker (Kuhn 1950)

- Two players  $A$  and  $B$ .
- Deck of 3 cards: 1,2,3.
- Players ante \$1.
- Each player gets one card.
- $A$  goes first. Can bet \$1 or pass.
  - If  $A$  bets,  $B$  can call or fold.
  - If  $A$  passes,  $B$  can bet \$1 or pass.
    - If  $B$  bets,  $A$  can call or fold.
- High card wins (if no folding). Max pot \$2.

- Two players A and B. 3 cards: 1,2,3.
- Players ante \$1. Each player gets one card.
- A goes first. Can bet \$1 or pass.
  - If A bets, B can call or fold.
  - If A passes, B can bet \$1 or pass.
    - If B bets, A can call or fold.

### Writing as a Matrix Game

- For a given card, A can decide to
  - Pass but fold if B bets. [PassFold]
  - Pass but call if B bets. [PassCall]
  - Bet. [Bet]
- Similar set of choices for B.

### Can look at all strategies as a big matrix...

	[FP,FP,CB]	[FP,CP,CB]	[FB,FP,CB]	[FB,CP,CB]
[PF,PF,PC]	0	0	-1/6	-1/6
[PF,PF,B]	0	1/6	-1/3	-1/6
[PF,PC,PC]	-1/6	0	0	1/6
[PF,PC,B]	-1/6	-1/6	1/6	1/6
[B,PF,PC]	1/6	-1/3	0	-1/2
[B,PF,B]	1/6	-1/6	-1/6	-1/2
[B,PC,PC]	0	-1/2	1/3	-1/6
[B,PC,B]	0	-1/3	1/6	-1/6


### And the minimax optimal strategies are...

- A:
  - If hold 1, then 5/6 PassFold and 1/6 Bet.
  - If hold 2, then  $\frac{1}{2}$  PassFold and  $\frac{1}{2}$  PassCall.
  - If hold 3, then  $\frac{1}{2}$  PassCall and  $\frac{1}{2}$  Bet.
 Has both bluffing and underbidding...
- B:
  - If hold 1, then 2/3 FoldPass and 1/3 FoldBet.
  - If hold 2, then 2/3 FoldPass and 1/3 CallPass.
  - If hold 3, then CallBet
 Minimax value of game is -1/18 to A.

Now, to General-Sum games...

### General-sum games

- In general-sum games, can get win-win and lose-lose situations.
- E.g., "what side of sidewalk to walk on?":

		Left	Right	
you	Left	(1,1)	(-1,-1)	person walking towards you 
	Right	(-1,-1)	(1,1)	

### General-sum games

- In general-sum games, can get win-win and lose-lose situations.
- E.g., "which movie should we go to?":

	Adventureland	Monsters
Adventureland	(8,2)	(0,0)
Monsters	(0,0)	(2,8)

No longer a unique "value" to the game.

## Nash Equilibrium

- A Nash Equilibrium is a stable pair of strategies (could be randomized).
- Stable means that neither player has incentive to deviate on their own.
- E.g., "what side of sidewalk to walk on":

	Left	Right
Left	(1,1)	(-1,-1)
Right	(-1,-1)	(1,1)

NE are: both left, both right, or both 50/50.

## Uses

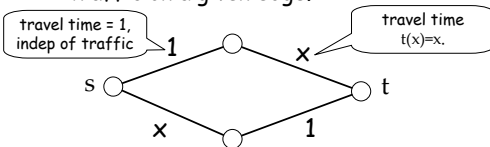
- Economists use games and equilibria as models of interaction.
- E.g., pollution / prisoner's dilemma:
  - (imagine pollution controls cost \$4 but improve everyone's environment by \$3)

	don't pollute	pollute
don't pollute	(2,2)	(-1,3)
pollute	(3,-1)	(0,0)

Need to add extra incentives to get good overall behavior.

## NE can do strange things

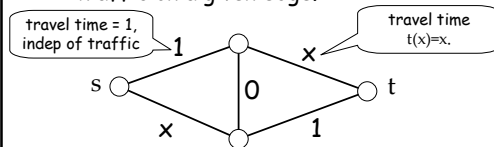
- Braess paradox:
  - Road network, traffic going from s to t.
  - travel time as function of fraction x of traffic on a given edge.



Fine. NE is 50/50. Travel time = 1.5

## NE can do strange things

- Braess paradox:
  - Road network, traffic going from s to t.
  - travel time as function of fraction x of traffic on a given edge.



Add new superhighway. NE: everyone uses zig-zag path. Travel time = 2.

## Existence of NE

- Nash (1950) proved: any general-sum game must have at least one such equilibrium.
  - Might require randomized strategies (called "mixed strategies")
- This also yields minimax thm as a corollary.
  - Pick some NE and let  $V$  = value to row player in that equilibrium.
  - Since it's a NE, neither player can do better even knowing the (randomized) strategy their opponent is playing.
  - So, they're each playing minimax optimal.

## Existence of NE

- Proof will be non-constructive.
- Unlike case of zero-sum games, we **do not know any** polynomial-time algorithm for finding Nash Equilibria in  $n \times n$  general-sum games. [known to be "PPAD-hard"]
- Notation:
  - Assume an  $n \times n$  matrix.
  - Use  $(p_1, \dots, p_n)$  to denote mixed strategy for row player, and  $(q_1, \dots, q_n)$  to denote mixed strategy for column player.

### Proof

- We'll start with Brouwer's fixed point theorem.
  - Let  $S$  be a compact convex region in  $\mathbb{R}^n$  and let  $f: S \rightarrow S$  be a continuous function.
  - Then there must exist  $x \in S$  such that  $f(x)=x$ .
  - $x$  is called a "fixed point" of  $f$ .
- Simple case:  $S$  is the interval  $[0,1]$ .
- We will care about:
  - $S = \{(p,q): p,q \text{ are legal probability distributions on } 1,\dots,n\}$ . I.e.,  $S = \text{simplex}_n \times \text{simplex}_n$

### Proof (cont)

- $S = \{(p,q): p,q \text{ are mixed strategies}\}$ .
- Want to define  $f(p,q) = (p',q')$  such that:
  - $f$  is continuous. This means that changing  $p$  or  $q$  a little bit shouldn't cause  $p'$  or  $q'$  to change a lot.
  - Any fixed point of  $f$  is a Nash Equilibrium.
- Then Brouwer will imply existence of NE.

### Try #1

- What about  $f(p,q) = (p',q')$  where  $p'$  is best response to  $q$ , and  $q'$  is best response to  $p$ ?
- Problem: not necessarily well-defined:
  - E.g., penalty shot: if  $p = (0.5,0.5)$  then  $q'$  could be anything.

	Left	Right
Left	(0,0)	(1,-1)
Right	(1,-1)	(0,0)

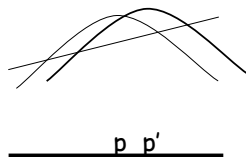
### Try #1

- What about  $f(p,q) = (p',q')$  where  $p'$  is best response to  $q$ , and  $q'$  is best response to  $p$ ?
- Problem: also not continuous:
  - E.g., if  $p = (0.51, 0.49)$  then  $q' = (1,0)$ . If  $p = (0.49,0.51)$  then  $q' = (0,1)$ .

	Left	Right
Left	(0,0)	(1,-1)
Right	(1,-1)	(0,0)

### Instead we will use...

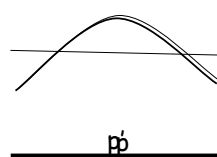
- $f(p,q) = (p',q')$  such that:
  - $q'$  maximizes [(expected gain wrt  $p$ ) -  $\|q-q'\|^2$ ]
  - $p'$  maximizes [(expected gain wrt  $q$ ) -  $\|p-p'\|^2$ ]



Note: quadratic + linear = quadratic.

### Instead we will use...

- $f(p,q) = (p',q')$  such that:
  - $q'$  maximizes [(expected gain wrt  $p$ ) -  $\|q-q'\|^2$ ]
  - $p'$  maximizes [(expected gain wrt  $q$ ) -  $\|p-p'\|^2$ ]



Note: quadratic + linear = quadratic.



### Instead we will use...

- $f(p,q) = (p',q')$  such that:
  - $q'$  maximizes [(expected gain wrt  $p$ ) -  $\|q-q'\|^2$ ]
  - $p'$  maximizes [(expected gain wrt  $q$ ) -  $\|p-p'\|^2$ ]
- $f$  is well-defined and continuous since quadratic has unique maximum and small change to  $p,q$  only moves this a little.
- Also fixed point = NE. (even if tiny incentive to move, will move little bit).
- So, that's it!

### One more interesting game

"Ultimatum game":

- Two players "Splitter" and "Chooser"
- 3<sup>rd</sup> party puts \$10 on table.
- Splitter gets to decide how to split between himself and Chooser.
- Chooser can accept or reject.
- If reject, money is burned.

### One more interesting game

"Ultimatum game": E.g., with \$4

		1	2	3
Chooser: how much to accept	1	(1,3)	(2,2)	(3,1)
	2	(0,0)	(2,2)	(3,1)
	3	(0,0)	(0,0)	(3,1)

Splitter: how much to offer chooser

Internal regret and correlated equilibria

What if everyone started using no-regret algs?

- w What if changing cost function is due to other players in the system optimizing for themselves?
- w In zero-sum games, empirical frequencies quickly approaches minimax optimal.
- w In general-sum games, does behavior quickly (or at all) approach a Nash equilibrium? (after all, a Nash Eq is exactly a set of distributions that are no-regret wrt each other).
- w Well, unfortunately, no.

A bad example for general-sum games

- w Augmented Shapley game from [Z04]: "RPSF"
  - First 3 rows/cols are Shapley game (rock / paper / scissors but if both do same action then both lose).
  - 4<sup>th</sup> action "play foosball" has slight negative if other player is still doing r/p/s but positive if other player does 4<sup>th</sup> action too.
  - NR algs will cycle among first 3 and have no regret, but do worse than only Nash Equilibrium of both playing foosball.
- w We didn't really expect this to work given how hard NE can be to find...

## What can we say?

- W If algorithms minimize "internal" or "swap" regret, then empirical distribution of play approaches *correlated equilibrium*.
  - n Foster & Vohra, Hart & Mas-Colell,...
  - n Though doesn't imply play is stabilizing.

What are internal regret and correlated equilibria?

## More general forms of regret

1. "best expert" or "external" regret:
  - Given  $n$  strategies. Compete with best of them in hindsight.
2. "sleeping expert" or "regret with time-intervals":
  - Given  $n$  strategies,  $k$  properties. Let  $S_i$  be set of days satisfying property  $i$  (might overlap). Want to simultaneously achieve low regret over each  $S_i$ .
3. "internal" or "swap" regret: like (2), except that  $S_i$  = set of days in which we chose strategy  $i$ .

## Internal/swap-regret

- E.g., each day we pick one stock to buy shares in.
  - Don't want to have regret of the form "every time I bought IBM, I should have bought Microsoft instead".
- Formally, regret is wrt optimal function  $f: \{1, \dots, N\} \rightarrow \{1, \dots, N\}$  such that every time you played action  $j$ , it plays  $f(j)$ .
- Motivation: connection to correlated equilibria.

## Internal/swap-regret

### "Correlated equilibrium"

- Distribution over entries in matrix, such that if a trusted party chooses one at random and tells you your part, you have no incentive to deviate.
- E.g., Shapley game.

	R	P	S
R	-1,-1	-1,1	1,-1
P	1,-1	-1,-1	-1,1
S	-1,1	1,-1	-1,-1

## Internal/swap-regret

- If all parties run a low internal/swap regret algorithm, then empirical distribution of play is an apx correlated equilibrium.
  - Correlator chooses random time  $t \in \{1, 2, \dots, T\}$ . Tells each player to play the action  $j$  they played in time  $t$  (but does not reveal value of  $t$ ).
  - Expected incentive to deviate:  $\sum_j \Pr(j) (\text{Regret}|j) = \text{swap-regret of algorithm}$
  - So, this says that correlated equilibria are a natural thing to see in multi-agent systems where individuals are optimizing for themselves

## Internal/swap-regret, contd

Algorithms for achieving low regret of this form:

- Foster & Vohra, Hart & Mas-Colell, Fudenberg & Levine.
- Can also convert any "best expert" algorithm into one achieving low swap regret.
- Unfortunately, time to achieve low regret is linear in  $n$  rather than  $\log(n)$ ...

### Internal/swap-regret, contd

Can convert any "best expert" algorithm  $A$  into one achieving low swap regret. Idea:

- Instantiate one copy  $A_i$  responsible for expected regret over times we play  $i$ .
- Each time step, if we play  $p=(p_1,\dots,p_n)$  and get cost vector  $c=(c_1,\dots,c_n)$ , then  $A_i$  gets cost-vector  $p_i c$ .
- If each  $A_i$  proposed to play  $q_i$ , so all together we have matrix  $Q$ , then define  $p = pQ$ .
- Allows us to view  $p_i$  as prob we chose action  $i$  or prob we chose algorithm  $A_i$ .