TTIC 31260 Algorithmic Game Theory

03/25/24

Bandit algorithms, internal & swap regret, and correlated equilibria

Your guide:

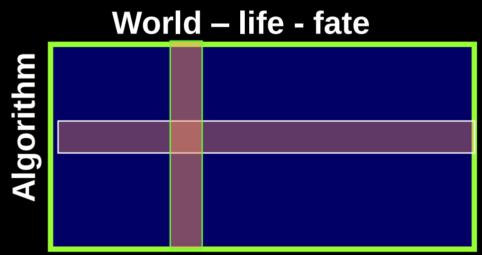
Avrim Blum

[Readings: Ch. 4.4-4.6 of AGT book]

Recap

"No-regret" algorithms for repeated decisions:

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



- At each time step, algorithm picks row, life picks column.
  - Alg pays cost (or gets benefit) for action chosen.
  - Alg gets column as feedback (or just its own cost/benefit in the "bandit" model).
  - Goal: do nearly as well as best fixed row in hindsight.

RWM

$(1-\varepsilon c_1^{\,2})(1-\varepsilon c_1^{\,1})\,1$

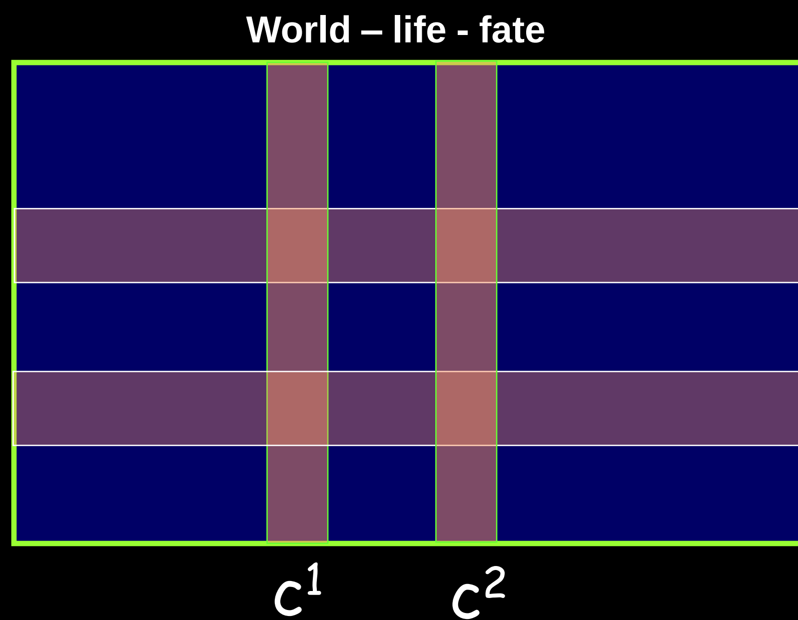
$(1-\varepsilon c_2^{\,2})(1-\varepsilon c_2^{\,1})\,1$

$(1-\varepsilon c_3^{\,2})(1-\varepsilon c_3^{\,1})\,1$

$. \quad . \quad 1$

$. \quad . \quad 1$

$(1-\varepsilon c_n^{\,2})(1-\varepsilon c_n^{\,1})\,1$



scaling so costs in [0,1]

Guarantee: $E[\text{cost}] \leq OPT + 2(OPT \cdot \log n)^{1/2}$

Since $OPT \leq T$, this is at most $OPT + 2(T\log n)^{1/2}$.

So, regret/time step $\leq 2(T\log n)^{1/2}/T \rightarrow 0$.

[ACFS02]: applying RWM to bandit setting

- What if only get your own cost/benefit as feedback?

- Use of RWM as subroutine to get algorithm with cumulative regret $O((TN \log N)^{1/2})$.

  [average regret $O(((N \log N)/T)^{1/2})$.]

- Will do a somewhat weaker version of their analysis (same algorithm but not as tight a bound).

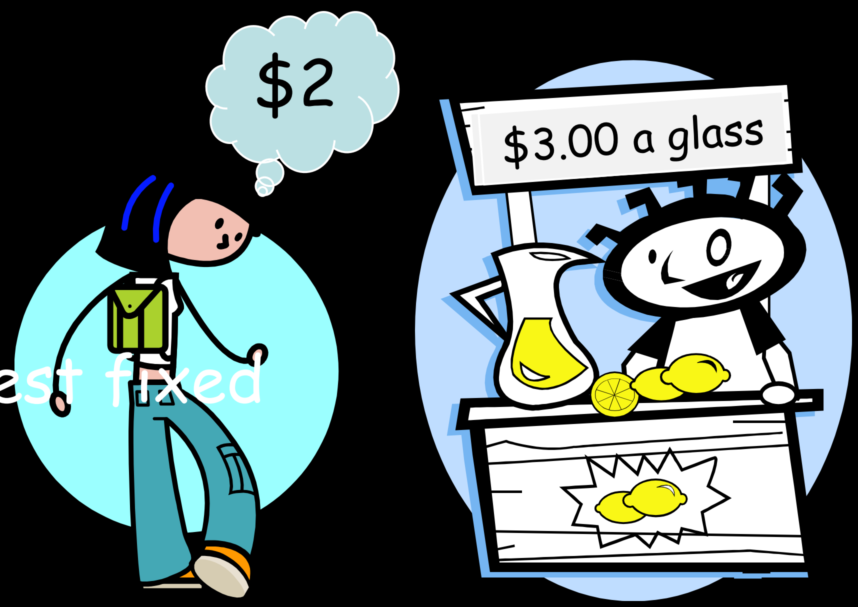
- For fun, talk about it in the context of online pricing…

Online pricing

- Say you are selling lemonade (or a cool new software tool, or bottles of water at the world cup).
- For t=1,2,…T
  - Seller sets price p^t
  - Buyer arrives with valuation v^t
  - If $v^t \geq p^t$, buyer purchases and pays p^t, else doesn't.
  - Repeat.

View each possible price as a different row/expert

- Assume all valuations $\leq$ h.
- Goal: do nearly as well as best fixed price in hindsight.



- If v^t revealed, run RWM. E[gain] $\geq$ OPT$(1-\epsilon) - O(\epsilon^{-1} h \log n)$.

Multi-armed bandit problem

Exponential Weights for Exploration and Exploitation ($\exp^3$)

[Auer,Cesa-Bianchi,Freund,Schapire]

OPT

Expert $i \sim q^t$

Gain g_i^t

q^t

Exp3

Distrib p^t

Gain vector $\hat{g}^t$

$\le nh/\gamma$

RWM

$\widehat{OPT}$

n = #experts

$q^t = (1-\gamma)p^t + \gamma$ unif

$\hat{g}^t = (0,\ldots,0, g_i^t/q_i^t,0,\ldots,0)$

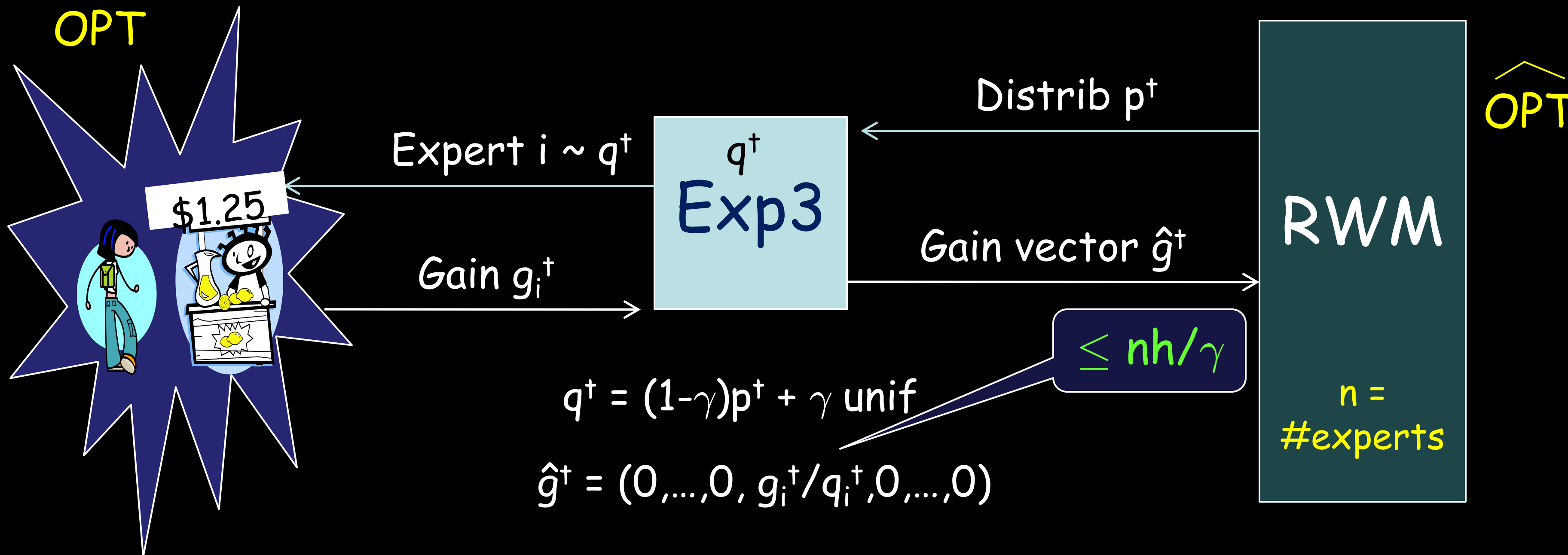
1. RWM believes gain is: $p^t \cdot \hat{g}^t = p_i^t(g_i^t/q_i^t) \equiv g^t_{RWM}$
2. $\sum_t g^t_{RWM} \ge \widehat{OPT}\,(1-\epsilon) - O(\epsilon^{-1}\, nh/\gamma \log n)$
3. Actual gain is: $g_i^t = g^t_{RWM}(q_i^t/p_i^t) \ge g^t_{RWM}(1-\gamma)$
4. $E[\widehat{OPT}] \ge OPT$. Because $E[\hat{g}_j^t] = (1-q_j^t)0 + q_j^t(g_j^t/q_j^t) = g_j^t$,

   so $E[\max_j[\sum_t \hat{g}_j^t]] \ge \max_j [\, E[\sum_t \hat{g}_j^t]\,] = OPT$.

Multi-armed bandit problem

Exponential Weights for Exploration and Exploitation ($\exp^3$)

[Auer,Cesa-Bianchi,Freund,Schapire]



Conclusion ($\gamma = \epsilon$):

$$E[\text{Exp3}] \geq \text{OPT}(1-\epsilon)^2 - O(\epsilon^{-2}\, nh \log(n))$$

Balancing would give $O((\text{OPT}\ nh \log n)^{2/3})$ in bound because of ϵ^{-2}.
But can reduce to ϵ^{-1} and $O((\text{OPT}\ nh \log n)^{1/2})$ more care in analysis.

Summary

Algorithms for online decision-making with strong guarantees on performance compared to best fixed choice.

- Application: play repeated game against adversary. Perform nearly as well as fixed strategy in hindsight.

Can apply even with very limited feedback.

- Application: which way to drive to work, with only feedback about your own paths; online pricing, even if only have buy/no buy feedback.

Internal/Swap Regret
and
Correlated Equilibria

What if all players minimize regret?

- In zero-sum games, empirical frequencies quickly approaches minimax optimal.
- 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. So *if the distributions stabilize*, they must converge to a Nash equil.
- Well, unfortunately, no.

A bad example for general-sum games

- Augmented Shapley game from [Zinkevich04]:
  - First 3 rows/cols are Shapley game (rock / paper / scissors but if both do same action then both lose).
  - 4^{th} action "play foosball" has slight negative if other player is still doing r/p/s but positive if other player does 4^{th} action too.

  RWM will cycle among first 3 and have no regret, but do worse than only Nash Equilibrium of both playing foosball.

- We didn't really expect this to work given how hard NE can be to find…

A bad example for general-sum games

- [Balcan-Constantin-Mehta12]:
  - Failure to converge even in Rank-1 games (games where R+C has rank 1).
  - Interesting because one can find equilibria efficiently in such games.

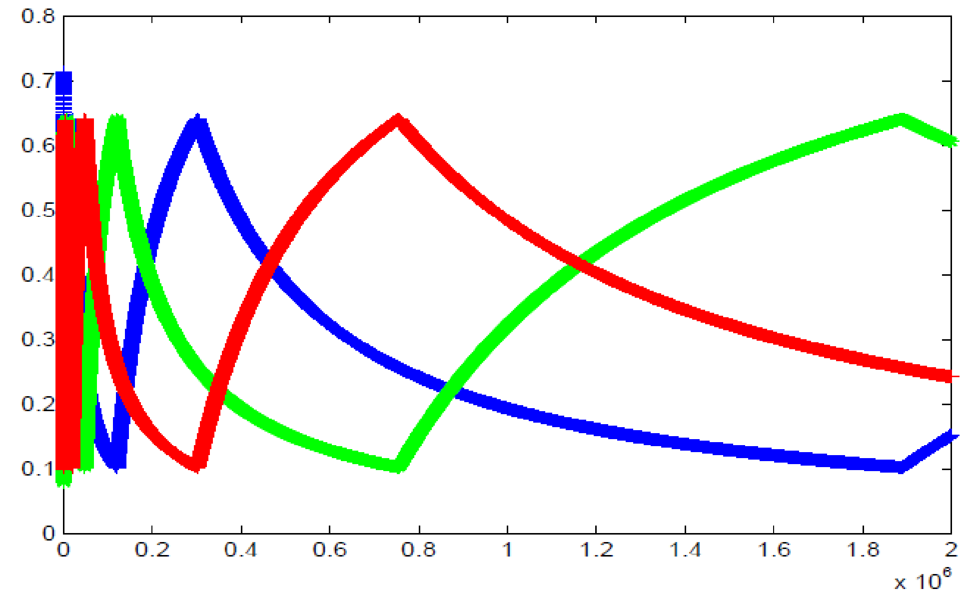


Figure 4. c_is of symmetric Shapley game with $a = 10, b = 1$

What *can we say?*

If algorithms minimize "internal" or "swap" regret, then empirical distribution of play approaches *correlated* equilibrium.

- Foster & Vohra, Hart & Mas-Colell,…
- Though doesn't imply play is stabilizing.

What are internal/swap 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 AT&T, I should have bought Microsoft instead".
- Formally, swap regret is wrt optimal function $f:\{1,\ldots,n\}\rightarrow\{1,\ldots,n\}$ such that every time you played action j, it plays f(j).
- So, competing with the best of these n^n "rewiring" functions.

Formally

- Let c^t denote the cost vector (loss vector) at time t.
- The algorithm's total expected cost (loss) is:

$$\sum_t p^t \cdot c^t = \sum_t \sum_j p_j^t c_j^t \,.$$

- For standard external regret, we are comparing this to the cost (loss) of the best action in hindsight: $\min_i \sum_t c_i^t$.
- For swap regret, we compare to the best rewiring of our probability mass:

$$\min_f \sum_t \sum_j p_j^t c_{f(j)}^t = \boxed{\sum_j \min_i} \sum_t p_j^t c_i^t \,.$$

- In other words, our probability mass on action j gets rewired to action $i = f(j)$.

Note: if you replace the $\sum_j \min_i$ with $\min_i \sum_j$ then you get back to external 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

Correlated equilibrium

- Can solve for CEQ using linear programming.

R C

- Solve for $D_{ij} \geq 0, \sum_{ij} D_{ij} = 1$, such that:
  - For all i, i', $\displaystyle\sum_j D_{ij} R_{ij} \geq \sum_j D_{ij} R_{i'j}$ [Conceptually, divide LHS and RHS by $\sum_j D_{ij}$]
  - For all j, j', $\displaystyle\sum_i D_{ij} C_{ij} \geq \sum_i D_{ij} C_{ij'}$ [Conceptually, divide LHS and RHS by $\sum_i D_{ij}$]

(E.g., Google maps tells each person what route to take, and it's a CEQ if nobody has any incentive to deviate)

- Can't do for Nash since replacing D_{ij} with $p_i q_j$ makes quadratic.

Connection

- If all parties run a low swap regret algorithm, then empirical distribution of play is an apx correlated equilibrium.
  - Correlator chooses random time $t \in \{1,2,\ldots,T\}$. Tells each player to play the action j they played in time t (but does not reveal value of t).
  - If each player had no swap regret, then no matter what action j they are told to play, they will not have any incentive to deviate $\Rightarrow$ correlated equilibrium.
  - Expected incentive to deviate: $\sum_j \Pr(j)(\text{Regret}|j)$ = swap-regret experienced.

Correlated vs Coarse-correlated Eq

In both cases: a distribution over entries in the matrix. Think of a third party choosing from this distr and telling you your part as “advice”.

“Correlated equilibrium”

- You have no incentive to deviate, even after seeing what the advice is.

“Coarse-Correlated equilibrium”

- If only choice is to see and follow, or not to see at all, would prefer the former.

Low external-regret $\Rightarrow$ apx coarse correlated equilib.

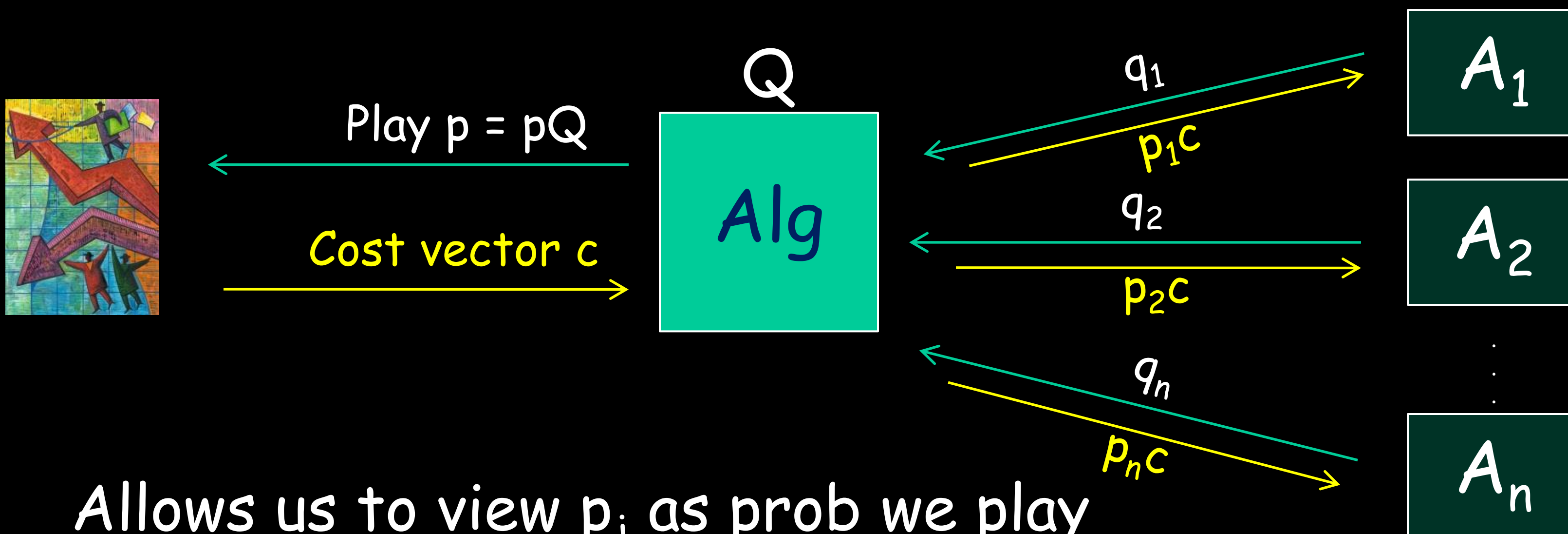
Internal/swap-regret, contd

Algorithms for achieving low regret of this form:

- Foster & Vohra, Hart & Mas-Colell, Fudenberg & Levine.
- Will present method of [BM05] showing how to convert any "best expert" algorithm into one achieving low swap regret.
- Unfortunately, #steps to achieve low swap regret is $O(n \log n)$ rather than $O(\log n)$.

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

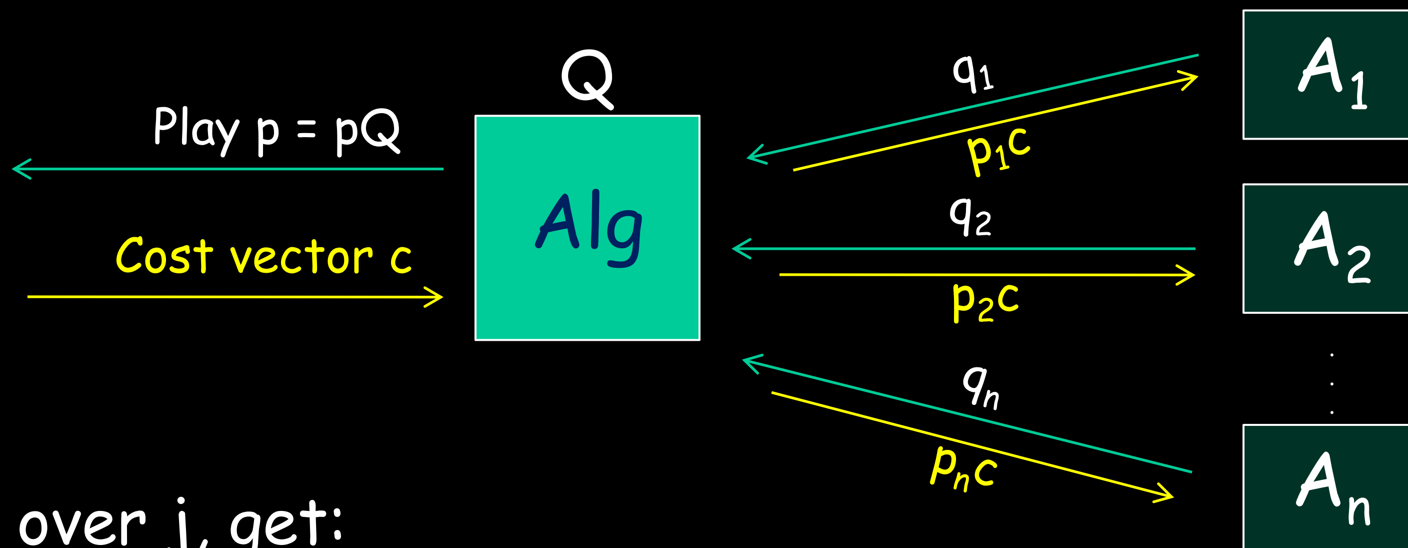
- Instantiate one copy A_j responsible for expected regret over times we play j.



- Allows us to view p_j as prob we play action j, or as prob we play alg A_j.
- Give A_j feedback of $p_j c$.
- A_j guarantees $\sum_t (p_j^t c^t)\cdot q_j^t \leq \min_i \sum_t p_j^t c_i^t$ + [regret term]
- Write as: $\sum_t p_j^t (q_j^t \cdot c^t) \leq \min_i \sum_t p_j^t c_i^t$ + [regret term]

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

- Instantiate one copy A_j responsible for expected regret over times we play j.



- Sum over j, get:

$$\sum_t p^t Q^t c^t \leq \sum_j \min_i \sum_t p_j^t c_i^t + n[\text{regret term}]$$

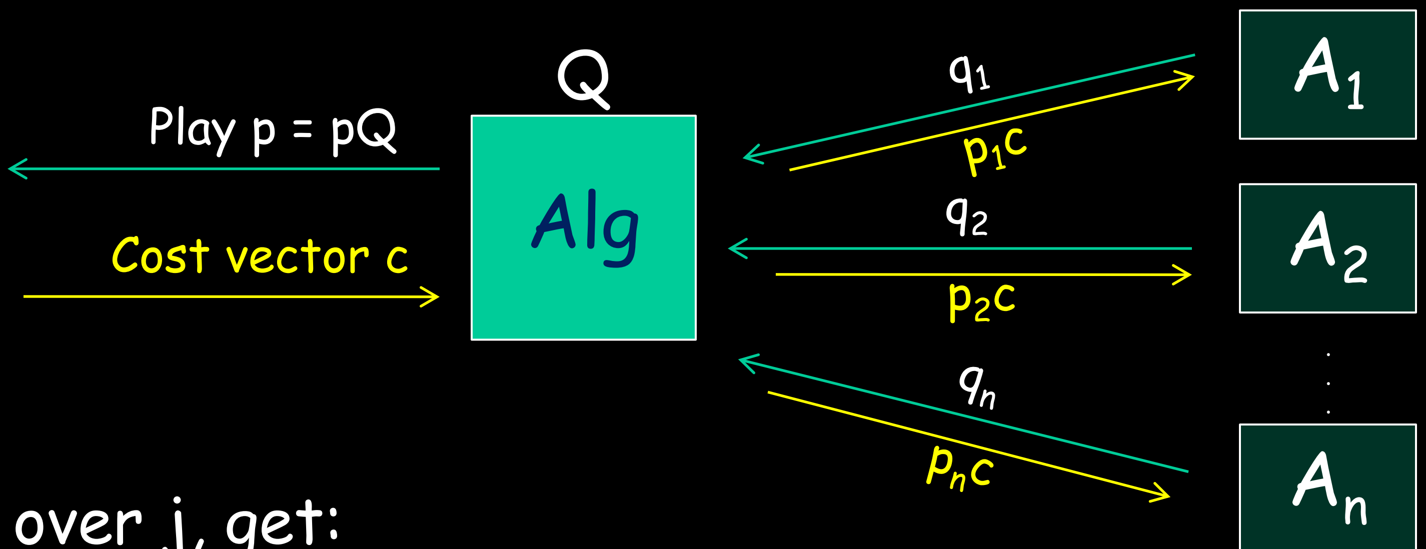
Our total cost

For each j, can move our prob to its own i=f(j)

- Write as: $\sum_t p_j^t (q_j^t \cdot c^t) \leq \min_i \sum_t p_j^t c_i^t + [\text{regret term}]$

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

- Instantiate one copy A_j responsible for expected regret over times we play j.



- Sum over j, get:

$$\sum_t p^t Q^t c^t \le \sum_j \min_i \sum_t p_j^t c_i^t + n[\text{regret term}]$$

Our total cost

For each j, can move our prob to its own i=f(j)

- Get swap-regret at most n times orig external regret.