The Contextual Bandits Problem

A New, Fast, and Simple Algorithm

Alekh Agarwal (MSR)
Daniel Hsu (Columbia)
Satyen Kale (Yahoo)
John Langford (MSR)
Lihong Li (MSR)
Rob Schapire (MSR/Princeton)

Example: Ad/Content Placement

- repeat:
  1. website visited by user (with profile, browsing history, etc.)
  2. website chooses ad/content to present to user
  3. user responds (clicks, leaves page, etc.)
- goal: make choices that elicit desired user behavior

Example: Medical Treatment

- repeat:
  1. doctor visited by patient (with symptoms, test results, etc.)
  2. doctor chooses treatment
  3. patient responds (recovers, gets worse, etc.)
- goal: make choices that maximize favorable outcomes

The Contextual Bandits Problem

- repeat:
  1. learner presented with context
  2. learner chooses an action
  3. learner observes reward (but only for chosen action)
- goal: learn to choose actions to maximize rewards

The Contextual Bandits Problem

- repeat:
  1. learner presented with context
  2. learner chooses an action
  3. learner observes reward (but only for chosen action)
- goal: learn to choose actions to maximize rewards
- general and fundamental problem: how to learn to make intelligent decisions through experience

Issues

- classic dilemma:
    - exploit what has already been learned
    - explore to learn which behaviors give best results

Issues

- classic dilemma:
  - exploit what has already been learned
  - explore to learn which behaviors give best results
- in addition, must use context effectively
  - many choices of behavior possible
  - may never see same context twice

Issues

- classic dilemma:
  - exploit what has already been learned
  - explore to learn which behaviors give best results
- in addition, must use context effectively
  - many choices of behavior possible
  - may never see same context twice
- selection bias: if explore while exploiting, will tend to get highly skewed data

Issues

- classic dilemma:
  - exploit what has already been learned
  - explore to learn which behaviors give best results
- in addition, must use context effectively
  - many choices of behavior possible
  - may never see same context twice
- selection bias: if explore while exploiting, will tend to get highly skewed data
- efficiency

This Talk

- new and general algorithm for contextual bandits
- optimal statistical performance
- far faster and simpler than predecessors

Formal Model

- repeat

  1a. learner observes context x_t

  2. learner selects action $a_t \in \{1, \ldots, K\}$
  3. learner receives observed reward $r_t(a_t)$

Formal Model

- repeat
  1a. learner observes context x_t
  1b. reward vector $\mathbf{r}_t \in [0, 1]^K$ chosen (but not observed)
  2. learner selects action $a_t \in \{1, \ldots, K\}$
  3. learner receives observed reward $r_t(a_t)$

Formal Model

- repeat
  1a. learner observes context x_t
  1b. reward vector $\mathbf{r}_t \in [0,1]^K$ chosen (but not observed)
  2. learner selects action $a_t \in \{1, \ldots, K\}$
  3. learner receives observed reward $r_t(a_t)$
- goal: maximize total reward:

$$\sum_{t=1}^{T} r_t(a_t)$$

Formal Model

- repeat
  1a. learner observes context x_t
  1b. reward vector $\mathbf{r}_t \in [0,1]^K$ chosen (but not observed)
  2. learner selects action $a_t \in \{1, \dots, K\}$
  3. learner receives observed reward $r_t(a_t)$
- goal: maximize total reward:

$$\sum_{t=1}^{T} r_t(a_t)$$

- assume pairs $(x_t, \mathbf{r}_t)$ chosen at random i.i.d.

Example

	Actions		
Context	1	2	3
$(Male, 50, \ldots)$			

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	0.2	0.0

Example

Context	Actions 1	2	3
$(\mathit{Male}, 50, \ldots)$	1.0	0.2	0.0

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0

total reward $= 0.2 +$

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$			

total reward $= 0.2 +$

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	1.0

total reward $= 0.2 +$

Example

Context	Actions 1	2	3
$(\mathit{Male}, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(\mathit{Female}, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$

total reward $= 0.2 + 1.0 +$

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$			

total reward $= 0.2 + 1.0 +$

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	0.1	0.7

total reward $= 0.2 + 1.0 +$

Example

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

total reward $= 0.2 + 1.0 + 0.1 + \cdots$

Special Case: Multi-armed Bandit Problem

- no context
- try to do as well as best single action

Special Case: Multi-armed Bandit Problem

- no context
- try to do as well as best single action
  - tacitly assuming there is one action that gives high rewards
  - e.g.: single treatment/ad/content that is right for entire population

Policies

- in contextual bandits setting, can use context to choose actions
- may exist good policy (decision rule) for choosing actions based on context

Policies

- in contextual bandits setting, can use context to choose actions
- may exist good policy (decision rule) for choosing actions based on context
- e.g.:

If $(\text{sex} = \text{male})$	choose action 2
Else if $(\text{age} > 45)$	choose action 1
else	choose action 3

Policies

- in contextual bandits setting, can use context to choose actions
- may exist good policy (decision rule) for choosing actions based on context
- e.g.:

If (sex = male)	choose action 2
Else if (age $>$ 45)	choose action 1
else	choose action 3

- policy π : (context x) $\mapsto$ (action a)

Policies

- in contextual bandits setting, can use context to choose actions
- may exist good policy (decision rule) for choosing actions based on context
- e.g.:

If (sex = male)	choose action 2
Else if (age $>$ 45)	choose action 1
else	choose action 3

- policy π : (context x) $\mapsto$ (action a)
- learner generally working with some rich policy space Π
  - e.g.: all decision trees ("if-then-else" rules)

Policies

- in contextual bandits setting, can use context to choose actions
- may exist good policy (decision rule) for choosing actions based on context
- e.g.:

  If (sex = male) choose action 2
  Else if (age $>$ 45) choose action 1
  else choose action 3

- policy π : (context x) $\mapsto$ (action a)
- learner generally working with some rich policy space Π
  - e.g.: all decision trees ("if-then-else" rules)
  - assume Π finite, but typically extremely large
  - tacit assumption:
    $\exists$ (unknown) policy $\pi \in \Pi$ that gives high rewards

Learning with Context and Policies

- goal: learn through experimentation to do (almost) as well as best $\pi \in \Pi$
- policies may be very complex and expressive
  $\Rightarrow$ powerful approach

Learning with Context and Policies

- goal: learn through experimentation to do (almost) as well as best $\pi \in \Pi$
- policies may be very complex and expressive
  $\Rightarrow$ powerful approach
- challenges:
  - Π extremely large
  - need to be learning about all policies simultaneously while also performing as well as the best
  - when action selected, only observe reward for policies that would have chosen same action
  - exploration versus exploitation on a gigantic scale!

Formal Model (*revisited*)

- repeat

  1a. learner observes context x_t

  1b. reward vector $\mathbf{r}_t \in [0,1]^K$ chosen (but not observed)

  2. learner selects action $a_t \in \{1, \ldots, K\}$

  3. learner receives observed reward $r_t(a_t)$

- goal: want high total (or average) reward
  relative to best policy $\pi \in \Pi$

Formal Model (*revisited*)

- repeat
  1a. learner observes context x_t
  1b. reward vector $\mathbf{r}_t \in [0,1]^K$ chosen (but not observed)
  2. learner selects action $a_t \in \{1,\ldots,K\}$
  3. learner receives observed reward $r_t(a_t)$
- goal: want high total (or average) reward
  relative to best policy $\pi \in \Pi$
  - i.e., want small regret:

$$\underbrace{\frac{1}{T}\sum_{t=1}^{T} r_t(a_t)}_{\text{learner's average reward}}$$

Formal Model (*revisited*)

- repeat

  1a. learner observes context x_t
  1b. reward vector $\mathbf{r}_t \in [0,1]^K$ chosen (but not observed)
  2. learner selects action $a_t \in \{1,\ldots,K\}$
  3. learner receives observed reward $r_t(a_t)$

- goal: want high total (or average) reward
  relative to best policy $\pi \in \Pi$
  - i.e., want small regret:

$$\underbrace{\max_{\pi\in\Pi} \frac{1}{T}\sum_{t=1}^{T} r_t(\pi(x_t))}_{\text{best policy's average reward}} - \underbrace{\frac{1}{T}\sum_{t=1}^{T} r_t(a_t)}_{\text{learner's average reward}}$$

An Algorithm that Solves this Problem

[Auer, Cesa-Bianchi, Freund, Schapire]

- Exp4 solves this problem
  - maintains weights over all policies in Π

An Algorithm that Solves this Problem

[Auer, Cesa-Bianchi, Freund, Schapire]

- Exp4 solves this problem
  - maintains weights over all policies in Π
- regret is essentially optimal:

$$O\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$$

- even works for adversarial (i.e., non-random, non-iid) data

An Algorithm that Solves this Problem

[Auer, Cesa-Bianchi, Freund, Schapire]

- Exp4 solves this problem
  - maintains weights over all policies in Π
- regret is essentially optimal:

$$O\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$$

- even works for adversarial (i.e., non-random, non-iid) data
- but: time/space are linear in $|\Pi|$
  - too slow if $|\Pi|$ gigantic

An Algorithm that Solves this Problem

[Auer, Cesa-Bianchi, Freund, Schapire]

- Exp4 solves this problem
  - maintains weights over all policies in Π
- regret is essentially optimal:

$$O\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$$

- even works for adversarial (i.e., non-random, non-iid) data
- but: time/space are linear in $|\Pi|$
  - too slow if $|\Pi|$ gigantic
- seems hopeless to do better for fully general policy spaces
- this talk: aim for time/space only poly$(\log |\Pi|)$ when Π is "well structured"

The (Fantasy) Full-Information Setting

- say see rewards for all actions

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(Male, 50, \ldots)$			

$\square$ = learner's action

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	0.2	0.0

$\square$ = learner's action

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0

$\square$ = learner's action

learner's total reward = $0.2 +$

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(\mathit{Male}, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(\mathit{Female}, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(\mathit{Female}, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

learner's total reward $= 0.2 + 1.0 + 0.1 + \cdots$

$\boxed{}$ = learner's action

The (Fantasy) Full-Information Setting

- say see rewards for all actions

	Actions		
Context	1	2	3
$(\mathit{Male}, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(\mathit{Female}, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(\mathit{Female}, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\boxed{}$ = learner's action

learner's total reward $= 0.2 + 1.0 + 0.1 + \cdots$

- for any π, can compute rewards would have received

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\square$ = learner's action

$\bigcirc$ = π's action

$$\text{learner's total reward} = 0.2 + 1.0 + 0.1 + \cdots$$
$$\pi\text{'s total reward} = 0.0 + 1.0 + 0.5 + \cdots$$

- for any π, can compute rewards would have received

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(\textit{Male}, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(\textit{Female}, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(\textit{Female}, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

= learner's action

= π's action

$$\text{learner's total reward} = 0.2 + 1.0 + 0.1 + \cdots$$
$$\pi\text{'s total reward} = 0.0 + 1.0 + 0.5 + \cdots$$

- for any π, can compute rewards would have received
  - average is good estimate of π's expected reward

The (Fantasy) Full-Information Setting

- say see rewards for all actions

Context	Actions 1	2	3
$(\mathit{Male}, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(\mathit{Female}, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(\mathit{Female}, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\square$ = learner's action

$\bigcirc$ = π's action

learner's total reward $= 0.2 + 1.0 + 0.1 + \cdots$

π's total reward $= 0.0 + 1.0 + 0.5 + \cdots$

- for any π, can compute rewards would have received
  - average is good estimate of π's expected reward
- choose empirically best $\pi \in \Pi$
- regret $= O\left(\sqrt{\frac{\ln|\Pi|}{T}}\right)$

"Arg-Max Oracle" (AMO)

- to apply, just need "oracle" (algorithm/subroutine) for finding best $\pi \in \Pi$ on observed rewards
- input: $(x_1, \mathbf{r}_1), \ldots, (x_T, \mathbf{r}_T)$
  $x_t =$ context
  $\mathbf{r}_t = (r_t(1), \ldots, r_t(K)) =$ rewards for all actions
- output:

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{t=1}^{T} r_t(\pi(x_t))$$

"Arg-Max Oracle" (AMO)

- to apply, just need "oracle" (algorithm/subroutine) for finding best $\pi \in \Pi$ on observed rewards
- input: $(x_1, \mathbf{r}_1), \ldots, (x_T, \mathbf{r}_T)$

  x_t = context

  $\mathbf{r}_t = (r_t(1), \ldots, r_t(K))$ = rewards for all actions
- output:

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{t=1}^{T} r_t(\pi(x_t))$$

- really just (cost-sensitive) classification:

context	$\leftrightarrow$	example
action	$\leftrightarrow$	label/class
policy	$\leftrightarrow$	classifier
reward	$\leftrightarrow$	gain/(negative) cost

"Arg-Max Oracle" (AMO)

- to apply, just need "oracle" (algorithm/subroutine) for finding best $\pi \in \Pi$ on observed rewards
- input: $(x_1, \mathbf{r}_1), \ldots, (x_T, \mathbf{r}_T)$

  $x_t =$ context

  $\mathbf{r}_t = (r_t(1), \ldots, r_t(K)) =$ rewards for all actions
- output:

$$\hat{\pi} = \arg\max_{\pi \in \Pi} \sum_{t=1}^{T} r_t(\pi(x_t))$$

- really just (cost-sensitive) classification:

context	$\leftrightarrow$	example
action	$\leftrightarrow$	label/class
policy	$\leftrightarrow$	classifier
reward	$\leftrightarrow$	gain/(negative) cost

- so: if have "good" classification algorithm for Π, can use to find good policy (in full-information setting)

But in the Bandit Setting...

- ...only see rewards for actions taken

But in the Bandit Setting...

- ...only see rewards for actions taken

	Actions		
Context	1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\boxed{}$ = learner's action

But in the Bandit Setting...

- ...only see rewards for actions taken

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\boxed{}$ = learner's action

But in the Bandit Setting...

- ...only see rewards for actions taken

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\boxed{}$ = learner's action

learner's total reward $= 0.2 + 1.0 + 0.1 + \cdots$

But in the Bandit Setting...

- ...only see rewards for actions taken

	Actions		
Context	1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\square$ = learner's action

$\bigcirc$ = π's action

learner's total reward $= 0.2 + 1.0 + 0.1 + \cdots$

- for any policy π, only observe π's rewards on subset of rounds

But in the Bandit Setting...

- ...only see rewards for actions taken

	Actions		
Context	1	2	3
(*Male*, 50, ...)	1.0	$\boxed{0.2}$	0.0
(*Female*, 18, ...)	1.0	0.0	$\boxed{1.0}$
(*Female*, 48, ...)	0.5	$\boxed{0.1}$	0.7
⋮		⋮	

$\square$ = learner's action

$\bigcirc$ = π's action

$$\text{learner's total reward} = 0.2 + 1.0 + 0.1 + \cdots$$

$$\pi\text{'s total reward} = ?? + 1.0 + ?? + \cdots$$

- for any policy π, only observe π's rewards on subset of rounds

But in the Bandit Setting...

- ...only see rewards for actions taken

Context	Actions 1	2	3
$(Male, 50, \ldots)$	1.0	$\boxed{0.2}$	0.0
$(Female, 18, \ldots)$	1.0	0.0	$\boxed{1.0}$
$(Female, 48, \ldots)$	0.5	$\boxed{0.1}$	0.7
$\vdots$		$\vdots$	

$\square$ = learner's action

$\bigcirc$ = π's action

$$\text{learner's total reward} = 0.2 + 1.0 + 0.1 + \cdots$$

$$\pi\text{'s total reward} = ?? + 1.0 + ?? + \cdots$$

- for any policy π, only observe π's rewards on subset of rounds
- might like to use AMO to find empirically good policy
- problems:
  - only see some rewards
  - observed rewards highly biased
    (due to skewed choice of actions)

Key Question

- still: AMO is a natural primitive
- key question: can we solve the contextual bandits problem given access to AMO?

Key Question

- still: AMO is a natural primitive
- key question: can we solve the contextual bandits problem given access to AMO?
- can we use an AMO on bandit data by somehow:
  - filling in missing data
  - overcoming bias

Key Question

- still: AMO is a natural primitive
- key question: can we solve the contextual bandits problem given access to AMO?
- can we use an AMO on bandit data by somehow:
  - filling in missing data
  - overcoming bias
- want:
  - optimal regret
  - time/space bounds $\mathrm{poly}(\log |\Pi|)$

Key Question

- still: AMO is a natural primitive
- key question: can we solve the contextual bandits problem given access to AMO?
- can we use an AMO on bandit data by somehow:
  - filling in missing data
  - overcoming bias
- want:
  - optimal regret
  - time/space bounds $\mathrm{poly}(\log |\Pi|)$
- AMO is theoretical idealization
- captures structure in policy space
- in practice, can use off-the-shelf classification algorithm

ϵ-Greedy/Epoch-Greedy

[Langford & Zhang]

- partially solved by the ϵ-greedy/epoch-greedy algorithm
- on each round, choose action:
  - according to "best" policy so far (with probability $1 - \epsilon$)
  - uniformly at random (with probability ϵ)

ϵ-Greedy/Epoch-Greedy

[Langford & Zhang]

- partially solved by the ϵ-greedy/epoch-greedy algorithm
- on each round, choose action:
  - according to "best" policy so far (with probability $1 - \epsilon$) [can find with AMO]
  - uniformly at random (with probability ϵ)

ϵ-Greedy/Epoch-Greedy

[Langford & Zhang]

- partially solved by the ϵ-greedy/epoch-greedy algorithm
- on each round, choose action:
  - according to "best" policy so far (with probability $1-\epsilon$) [can find with AMO]
  - uniformly at random (with probability ϵ)
- regret $= O\left(\left(\frac{K \ln |\Pi|}{T}\right)^{1/3}\right)$
- fast and simple, but not optimal

“Monster” Algorithm

[Dudík, Hsu, Kale, Karampatziakis, Langford, Reyzin & Zhang]

- RandomizedUCB (aka “Monster”) algorithm gets optimal regret using AMO
- solves multiple optimization problems using ellipsoid algorithm
- very slow: calls AMO about $\tilde{O}\left(T^4\right)$ times on every round

Main Result

- new, simple algorithm for contextual bandits with AMO access
- (nearly) optimal regret: $\tilde{O}\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$
- fast: calls AMO far less than once per round!
    - on average, calls AMO

$$\tilde{O}\left(\sqrt{\frac{K}{T \ln |\Pi|}}\right) \ll 1$$

times per round

Main Result

- new, simple algorithm for contextual bandits with AMO access
- (nearly) optimal regret: $\tilde{O}\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$
- fast: calls AMO far less than once per round!
    - on average, calls AMO

$$\tilde{O}\left(\sqrt{\frac{K}{T \ln |\Pi|}}\right) \ll 1$$

      times per round
- rest of talk: sketching main ideas of the algorithm

De-biasing Biased Estimates

- selection bias is major problem:
  - only observe reward for single action
  - exploring while exploiting leads to inherently biased estimates

De-biasing Biased Estimates

- selection bias is major problem:
  - only observe reward for single action
  - exploring while exploiting leads to inherently biased estimates
- nevertheless: can use simple trick to get unbiased estimates for all actions

De-biasing Biased Estimates (cont.)

- say $r(a) =$ (unknown) reward for action a
  $p(a) =$ (known) probability of choosing a

De-biasing Biased Estimates (cont.)

- say $r(a) =$ (unknown) reward for action a
  $p(a) =$ (known) probability of choosing a
- define $\hat{r}(a) = \begin{cases} r(a)/p(a) & \text{if } a \text{ chosen} \\ 0 & \text{else} \end{cases}$
- then $\mathrm{E}\left[\hat{r}(a)\right] = r(a)$

De-biasing Biased Estimates (cont.)

- say $r(a) =$ (unknown) reward for action a
  $p(a) =$ (known) probability of choosing a
- define $\hat{r}(a) = \begin{cases} r(a)/p(a) & \text{if } a \text{ chosen} \\ 0 & \text{else} \end{cases}$
- then $\mathrm{E}[\hat{r}(a)] = r(a)$ — unbiased!

$\therefore$ can estimate reward for all actions

De-biasing Biased Estimates (cont.)

- say $r(a) =$ (unknown) reward for action a
  $p(a) =$ (known) probability of choosing a
- define $\hat{r}(a) = \begin{cases} r(a)/p(a) & \text{if } a \text{ chosen} \\ 0 & \text{else} \end{cases}$
- then $\mathrm{E}\left[\hat{r}(a)\right] = r(a)$ — unbiased!

$\therefore$ can estimate reward for all actions

$\therefore$ can estimate expected reward for any policy π:

$$\hat{R}(\pi) = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \hat{r}_\tau(\pi(x_\tau)) = \hat{\mathrm{E}}\left[\hat{r}(\pi(x))\right]$$

De-biasing Biased Estimates (cont.)

- say $r(a) =$ (unknown) reward for action a
  $p(a) =$ (known) probability of choosing a
- define $\hat{r}(a) = \begin{cases} r(a)/p(a) & \text{if } a \text{ chosen} \\ 0 & \text{else} \end{cases}$
- then $\mathrm{E}[\hat{r}(a)] = r(a)$ — unbiased!

∴ can estimate reward for all actions

∴ can estimate expected reward for any policy π:

$$\hat{R}(\pi) = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \hat{r}_\tau(\pi(x_\tau)) = \hat{\mathrm{E}}[\hat{r}(\pi(x))]$$

∴ can estimate regret of any policy π:

$$\widehat{\text{Regret}}(\pi) = \max_{\hat{\pi} \in \Pi} \hat{R}(\hat{\pi}) - \hat{R}(\pi)$$

  - can find maximizing $\hat{\pi}$ using AMO

Variance Control

- estimates are unbiased — done?

Variance Control

- estimates are unbiased — done?
- no! — variance may be extremely large

Variance Control

- estimates are unbiased — done?
- no! — variance may be extremely large
- can show $\text{variance}(\hat{r}(a)) \leq \dfrac{1}{p(a)}$

Variance Control

- estimates are unbiased — done?
- no! — variance may be extremely large
- can show $\text{variance}(\hat{r}(a)) \leq \dfrac{1}{p(a)}$

$\therefore$ to get good estimates, must ensure that $1/p(a)$ not too large

Randomizing over Policies

- need to choose actions (semi-)randomly

Randomizing over Policies

- need to choose actions (semi-)randomly
- approach: on each round,
  - compute distribution $\mathbf{Q}$ over policy space Π
  - randomly pick $\pi \sim \mathbf{Q}$
  - on current context x, choose action $\pi(x)$

Randomizing over Policies

- need to choose actions (semi-)randomly
- approach: on each round,
  - compute distribution $\mathbf{Q}$ over policy space Π
  - randomly pick $\pi \sim \mathbf{Q}$
  - on current context x, choose action $\pi(x)$
- $\mathbf{Q}$ induces distribution over actions (for any x):

$$Q(a|x) = \Pr_{\pi \sim \mathbf{Q}} [\pi(x) = a]$$

Randomizing over Policies

- need to choose actions (semi-)randomly
- approach: on each round,
  - compute distribution $\mathbf{Q}$ over policy space Π
  - randomly pick $\pi \sim \mathbf{Q}$
  - on current context x, choose action $\pi(x)$
- $\mathbf{Q}$ induces distribution over actions (for any x):

$$Q(a|x) = \Pr_{\pi \sim \mathbf{Q}} [\pi(x) = a]$$

- seems will require time/space $O(|\Pi|)$ to compute $\mathbf{Q}$ over space Π
  - will see later how to avoid!

How to Pick $\mathbf{Q}$

- on each round, want to pick $\mathbf{Q}$ with:
  1. low (estimated) regret
     i.e., choose actions think will give high reward

How to Pick $\mathbf{Q}$

- on each round, want to pick $\mathbf{Q}$ with:
  1. low (estimated) regret
     i.e., choose actions think will give high reward
  2. low (estimated) variance
     i.e., ensure future estimates will be accurate

How to Pick $\mathbf{Q}$

- on each round, want to pick $\mathbf{Q}$ with:
  1. low (estimated) regret [exploit]
     i.e., choose actions think will give high reward
  2. low (estimated) variance [explore]
     i.e., ensure future estimates will be accurate

Low Regret

- $\widehat{\text{Regret}}(\pi) =$ estimated regret of π

Low Regret

- $\widehat{\text{Regret}}(\pi) =$ estimated regret of π
- so: estimated regret for random $\pi \sim \mathbf{Q}$ is

$$\sum_{\pi} Q(\pi)\, \widehat{\text{Regret}}(\pi) = \mathrm{E}_{\pi \sim \mathbf{Q}}\left[\widehat{\text{Regret}}(\pi)\right]$$

Low Regret

- $\widehat{\text{Regret}}(\pi) =$ estimated regret of π
- so: estimated regret for random $\pi \sim \mathbf{Q}$ is

$$\sum_{\pi} Q(\pi)\,\widehat{\text{Regret}}(\pi) = \mathrm{E}_{\pi \sim \mathbf{Q}}\left[\widehat{\text{Regret}}(\pi)\right]$$

- want small:

$$\sum_{\pi} Q(\pi)\,\widehat{\text{Regret}}(\pi) \le \text{[small]}$$

Low Variance

- $\frac{1}{Q(a|x)}$ = variance of estimate of reward for action a

Low Variance

- $\dfrac{1}{Q(a|x)} =$ variance of estimate of reward for action a
- so $\dfrac{1}{Q(\pi(x)|x)} =$ variance if action chosen by π

Low Variance

- $\frac{1}{Q(a|x)}$ = variance of estimate of reward for action a
- so $\frac{1}{Q(\pi(x)|x)}$ = variance if action chosen by π
- can estimate expected variance for actions chosen by π:

$$\hat{V}^Q(\pi) = \hat{\mathrm{E}}\left[\frac{1}{Q(\pi(x)|x)}\right] = \frac{1}{t-1}\sum_{\tau=1}^{t-1}\frac{1}{Q(\pi(x_\tau)|x_\tau)}$$

Low Variance

- $\dfrac{1}{Q(a|x)}$ = variance of estimate of reward for action a
- so $\dfrac{1}{Q(\pi(x)|x)}$ = variance if action chosen by π
- can estimate expected variance for actions chosen by π:

$$\hat{V}^Q(\pi) = \hat{\mathrm{E}}\left[\frac{1}{Q(\pi(x)|x)}\right] = \frac{1}{t-1}\sum_{\tau=1}^{t-1}\frac{1}{Q(\pi(x_\tau)|x_\tau)}$$

- want small:

$$\hat{V}^Q(\pi) \leq [\text{small}] \quad \text{for all } \pi \in \Pi$$

Low Variance

- $\dfrac{1}{Q(a|x)}$ = variance of estimate of reward for action a
- so $\dfrac{1}{Q(\pi(x)|x)}$ = variance if action chosen by π
- can estimate expected variance for actions chosen by π:

$$\hat{V}^{Q}(\pi) = \hat{\mathrm{E}}\left[\frac{1}{Q(\pi(x)|x)}\right] = \frac{1}{t-1}\sum_{\tau=1}^{t-1}\frac{1}{Q(\pi(x_\tau)|x_\tau)}$$

- want small:

$$\hat{V}^{Q}(\pi) \le [\text{small}] \quad \text{for all } \pi \in \Pi$$

- detail: problematic if $Q(a|x)$ too close to zero

Low Variance

- $\frac{1}{Q^\mu(a|x)}$ = variance of estimate of reward for action a
- so $\frac{1}{Q^\mu(\pi(x)|x)}$ = variance if action chosen by π
- can estimate expected variance for actions chosen by π:

$$\hat{V}^Q(\pi) = \hat{\mathrm{E}}\left[\frac{1}{Q^\mu(\pi(x)|x)}\right] = \frac{1}{t-1}\sum_{\tau=1}^{t-1}\frac{1}{Q^\mu(\pi(x_\tau)|x_\tau)}$$

- want small:

$$\hat{V}^Q(\pi) \leq [\text{small}] \quad \text{for all } \pi \in \Pi$$

- detail: problematic if $Q(a|x)$ too close to zero
  - to avoid, "smooth" probabilities by occassionally picking action uniformly at random:

$$Q^\mu(a|x) = (1 - K\mu)Q(a|x) + \mu$$

Pulling Together

- want $\mathbf{Q}$ such that:

$$\sum_{\pi} Q(\pi)\,\widehat{\text{Regret}}(\pi) \le \text{[small]}$$

$$\hat{V}^{Q}(\pi) \le \text{[small]} \qquad\qquad \text{for all } \pi \in \Pi$$

Pulling Together

- want $\mathbf{Q}$ such that:

$$\sum_{\pi} Q(\pi)\,\widehat{\text{Regret}}(\pi) \le \text{[small]}$$

$$\hat{V}^{Q}(\pi) \le \text{[small]} \qquad \text{for all } \pi \in \Pi$$

$$\sum_{\pi} Q(\pi) = 1$$

Pulling Together

- want **Q** such that:

$$\begin{aligned} &\sum_{\pi} Q(\pi)\, \widehat{\mathrm{Regret}}(\pi) \le C_0 \\ &C_1 \cdot \hat{V}^{Q}(\pi) \le C_0 \qquad\qquad\qquad \text{for all } \pi \in \Pi \\ &\sum_{\pi} Q(\pi) = 1 \end{aligned}$$

- can fill in constants

Pulling Together

- want $\mathbf{Q}$ such that:

$$\sum_\pi Q(\pi)\,\widehat{\mathrm{Regret}}(\pi) \le C_0$$

$$C_1 \cdot \hat{V}^Q(\pi) \le C_0 + \widehat{\mathrm{Regret}}(\pi) \qquad \text{for all } \pi \in \Pi$$

$$\sum_\pi Q(\pi) = 1$$

- can fill in constants
- make easier by:
  - allowing higher variance for policies with higher regret (poor policies can be eliminated even with fairly poor performance estimates)

Pulling Together

- want **Q** such that:

$$\sum_\pi Q(\pi)\, \widehat{\mathrm{Regret}}(\pi) \le C_0$$

$$C_1 \cdot \hat{V}^Q(\pi) \le C_0 + \widehat{\mathrm{Regret}}(\pi) \qquad \text{for all } \pi \in \Pi$$

$$\sum_\pi Q(\pi) \le 1$$

- can fill in constants
- make easier by:
  - allowing higher variance for policies with higher regret (poor policies can be eliminated even with fairly poor performance estimates)
  - only require **Q** to be sub-distribution (can put all remaining mass on $\hat{\pi}$ with maximum estimated reward)

Optimization Problem "OP"

find **Q** such that:

$$\sum_{\pi} Q(\pi)\,\widehat{\mathrm{Regret}}(\pi) \le C_0$$

$$C_1 \cdot \hat{V}^{Q}(\pi) \le C_0 + \widehat{\mathrm{Regret}}(\pi) \quad \text{for all } \pi \in \Pi$$

$$\sum_{\pi} Q(\pi) \;\le\; 1$$

Optimization Problem "OP"

find $\mathbf{Q}$ such that:

$$\sum_{\pi} Q(\pi)\, \widehat{\text{Regret}}(\pi) \leq C_0 \qquad \text{[regret constraint]}$$

$$C_1 \cdot \hat{V}^{Q}(\pi) \leq C_0 + \widehat{\text{Regret}}(\pi) \quad \text{for all } \pi \in \Pi \qquad \text{[variance constraint]}$$

$$\sum_{\pi} Q(\pi) \leq 1 \qquad \text{[sub-distribution]}$$

Optimization Problem "OP"

find $\mathbf{Q}$ such that:

$$\sum_{\pi} Q(\pi)\,\widehat{\mathrm{Regret}}(\pi) \le C_0 \qquad \text{[regret constraint]}$$

$$C_1 \cdot \hat{V}^{Q}(\pi) \le C_0 + \widehat{\mathrm{Regret}}(\pi) \quad \text{for all } \pi \in \Pi \qquad \text{[variance constraint]}$$

$$\sum_{\pi} Q(\pi) \le 1 \qquad \text{[sub-distribution]}$$

- similar to [Dudík et al.]

Optimization Problem "OP"

find $\mathbf{Q}$ such that:

$$\sum_{\pi} Q(\pi)\,\widehat{\text{Regret}}(\pi) \le C_0 \qquad \text{[regret constraint]}$$

$$C_1 \cdot \hat{V}^Q(\pi) \le C_0 + \widehat{\text{Regret}}(\pi) \quad \text{for all } \pi \in \Pi \qquad \text{[variance constraint]}$$

$$\sum_{\pi} Q(\pi) \;\le\; 1 \qquad \text{[sub-distribution]}$$

- similar to [Dudík et al.]
- seems awful:
  - $|\Pi|$ variables
  - $|\Pi|$ constraints
  - constraints involve nasty non-linear functions (recall $\hat{V}^Q(\pi) = \hat{\mathrm{E}}\left[\frac{1}{Q^\mu(\pi(x)|x)}\right]$)
  - not even clear if feasible

If We Can Solve It...

- Theorem: if can solve OP on every round (for appropriate constants), then will get regret

$$\tilde{O}\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$$

.

If We Can Solve It...

- Theorem: if can solve OP on every round (for appropriate constants), then will get regret

$$\tilde{O}\left(\sqrt{\frac{K \ln |\Pi|}{T}}\right)$$

.

- proof idea:
  - regret constraint ensures low regret
    (if estimates are good enough)
  - variance constraint ensures that they actually will be good enough
- essentially same approach as [Dudík et al.]

How to Solve?

- basic idea:
  - find a violated constraint
  - (attempt to) fix it
  - repeat

How to Solve? (cont.)

- $\mathbf{Q} \leftarrow \mathbf{0}$
- repeat:
  1. if $\mathbf{Q}$ "too big" then rescale
     - (i.e., multiply $\mathbf{Q}$ by scalar < 1)
     - ensures sub-distribution and regret constraints are satisfied

How to Solve? (cont.)

- $\mathbf{Q} \leftarrow \mathbf{0}$
- repeat:
  1. if $\mathbf{Q}$ “too big” then rescale
     - (i.e., multiply $\mathbf{Q}$ by scalar < 1)
     - ensures sub-distribution and regret constraints are satisfied
  2. find $\pi \in \Pi$ for which corresponding variance constraint is violated
     a. if none exists, halt and output $\mathbf{Q}$
     b. else $Q(\pi) \leftarrow Q(\pi) + \alpha$ where $\alpha =$ [some formula]

More Detail: Rescaling Step

1. [detailed version]
   if $\sum_{\pi} Q(\pi)(C_0 + \widehat{\text{Regret}}(\pi)) > C_0$ then rescale $\mathbf{Q}$
   (multiply by scalar < 1) so holds with equality

More Detail: Rescaling Step

1. [detailed version]
   if $\sum_{\pi} Q(\pi)(C_0 + \widehat{\text{Regret}}(\pi)) > C_0$ then rescale $\mathbf{Q}$
   (multiply by scalar < 1) so holds with equality

- after this step, have

$$\sum_{\pi} Q(\pi)(C_0 + \widehat{\text{Regret}}(\pi)) \le C_0$$

More Detail: Rescaling Step

1. [detailed version]
   if $\sum_\pi Q(\pi)(C_0 + \widehat{\text{Regret}}(\pi)) > C_0$ then rescale $\mathbf{Q}$ (multiply by scalar < 1) so holds with equality

- after this step, have

$$\sum_\pi Q(\pi)(C_0 + \widehat{\text{Regret}}(\pi)) \leq C_0$$

  which implies:
  - $\sum_\pi Q(\pi) \leq 1$ [sub-distribution]
  - $\sum_\pi Q(\pi)\, \widehat{\text{Regret}}(\pi) \leq C_0$ [regret constraint]

More Detail: Checking Variance Constraints

2. [detailed version]

   find $\pi \in \Pi$ for which $C_1 \cdot \hat{V}^Q(\pi) - \widehat{\text{Regret}}(\pi) > C_0$

   a. if none exists, halt and output $\mathbf{Q}$
   b. else $Q(\pi) \leftarrow Q(\pi) + \alpha$ where $\alpha =$ [some formula]

More Detail: Checking Variance Constraints

2. [detailed version]
   find $\pi \in \Pi$ for which $C_1 \cdot \hat{V}^Q(\pi) - \widehat{\mathrm{Regret}}(\pi) > C_0$
   a. if none exists, halt and output $\mathbf{Q}$
   b. else $Q(\pi) \leftarrow Q(\pi) + \alpha$ where $\alpha =$ [some formula]

- if halts then $C_1 \cdot \hat{V}^Q(\pi) \leq C_0 + \widehat{\mathrm{Regret}}(\pi)$ for all $\pi \in \Pi$
  [variance constraint]

More Detail: Checking Variance Constraints

2. [detailed version]
   find $\pi \in \Pi$ for which $C_1 \cdot \hat{V}^Q(\pi) - \widehat{\text{Regret}}(\pi) > C_0$
   a. if none exists, halt and output $\mathbf{Q}$
   b. else $Q(\pi) \leftarrow Q(\pi) + \alpha$ where $\alpha =$ [some formula]

- if halts then $C_1 \cdot \hat{V}^Q(\pi) \leq C_0 + \widehat{\text{Regret}}(\pi)$ for all $\pi \in \Pi$
  [variance constraint]
- can execute step using AMO:
  - can construct "pseudo-rewards" $\tilde{\mathbf{r}}_\tau$ for which $(\forall \pi)$:

$$C_1 \cdot \hat{V}^Q(\pi) - \widehat{\text{Regret}}(\pi) = \sum_\tau \tilde{r}_\tau(\pi(x_\tau)) + [\text{constant}]$$

More Detail: Checking Variance Constraints

2. [detailed version]
   find $\pi \in \Pi$ for which $C_1 \cdot \hat{V}^Q(\pi) - \widehat{\mathrm{Regret}}(\pi) > C_0$
   a. if none exists, halt and output $\mathbf{Q}$
   b. else $Q(\pi) \leftarrow Q(\pi) + \alpha$ where $\alpha =$ [some formula]

- if halts then $C_1 \cdot \hat{V}^Q(\pi) \le C_0 + \widehat{\mathrm{Regret}}(\pi)$ for all $\pi \in \Pi$
  [variance constraint]
- can execute step using AMO:
  - can construct "pseudo-rewards" $\tilde{\mathbf{r}}_\tau$ for which $(\forall \pi)$:

$$C_1 \cdot \hat{V}^Q(\pi) - \widehat{\mathrm{Regret}}(\pi) = \sum_\tau \tilde{r}_\tau(\pi(x_\tau)) + \text{[constant]}$$

  - so: can maximize with AMO
  - will find violating constraint (if one exists)

More Detail: Checking Variance Constraints

2. [detailed version]
   find $\pi \in \Pi$ for which $C_1 \cdot \hat{V}^Q(\pi) - \widehat{\text{Regret}}(\pi) > C_0$
   a. if none exists, halt and output $\mathbf{Q}$
   b. else $Q(\pi) \leftarrow Q(\pi) + \alpha$ where $\alpha =$ [some formula]

- if halts then $C_1 \cdot \hat{V}^Q(\pi) \leq C_0 + \widehat{\text{Regret}}(\pi)$ for all $\pi \in \Pi$
  [variance constraint]
- can execute step using AMO:
  - can construct "pseudo-rewards" $\tilde{\mathbf{r}}_\tau$ for which $(\forall \pi)$:

$$C_1 \cdot \hat{V}^Q(\pi) - \widehat{\text{Regret}}(\pi) = \sum_\tau \tilde{r}_\tau(\pi(x_\tau)) + \text{[constant]}$$

  - so: can maximize with AMO
  - will find violating constraint (if one exists)

$\therefore$ one AMO call per iteration

Why Does It Work?

- so: if halts, then outputs solution to OP
- but how long will it take to halt (if ever)?
- to answer, analyze using a potential function

A Potential Function

- define potential function to quantify progress:

$$\Phi(\mathbf{Q}) = A \cdot \underbrace{\hat{\mathrm{E}}\left[\mathrm{RE}\left(\text{uniform} \;\|\; Q^{\mu}(\cdot|x)\right)\right]}_{\text{low variance}} + B \cdot \underbrace{\sum_{\pi} Q(\pi)\, \widehat{\text{Regret}}(\pi)}_{\text{low regret}}$$

A Potential Function

- define potential function to quantify progress:

$$\Phi(\mathbf{Q}) = A \cdot \underbrace{\hat{\mathrm{E}}\left[\mathrm{RE}\left(\text{uniform} \;\|\; Q^{\mu}(\cdot|x)\right)\right]}_{\text{low variance}} + B \cdot \underbrace{\sum_{\pi} Q(\pi)\, \widehat{\mathrm{Regret}}(\pi)}_{\text{low regret}}$$

- defined for all nonnegative vectors $\mathbf{Q}$ over Π
  (not just sub-distributions)

A Potential Function

- define potential function to quantify progress:

$$\Phi(\mathbf{Q}) = A \cdot \underbrace{\hat{\mathrm{E}}\left[\mathrm{RE}\left(\text{uniform} \;\|\; Q^{\mu}(\cdot|x)\right)\right]}_{\text{low variance}} + \underbrace{B \cdot \sum_{\pi} Q(\pi)\, \widehat{\mathrm{Regret}}(\pi)}_{\text{low regret}}$$

- defined for all nonnegative vectors $\mathbf{Q}$ over Π (not just sub-distributions)
- properties:
  - $\Phi(\mathbf{Q}) \geq 0$
  - convex

A Potential Function

- define potential function to quantify progress:

$$\Phi(\mathbf{Q}) = A \cdot \underbrace{\hat{\mathrm{E}}\left[\mathrm{RE}\left(\text{uniform} \;\|\; Q^{\mu}(\cdot|x)\right)\right]}_{\text{low variance}} + B \cdot \underbrace{\sum_{\pi} Q(\pi)\, \widehat{\mathrm{Regret}}(\pi)}_{\text{low regret}}$$

- defined for all nonnegative vectors $\mathbf{Q}$ over Π (not just sub-distributions)
- properties:
  - $\Phi(\mathbf{Q}) \geq 0$
  - convex
  - if $\mathbf{Q}$ minimizes Φ then $\mathbf{Q}$ is a solution to OP
    - key proof step: $\partial\Phi/\partial Q(\pi) \propto$ variance constraint for π

  $\therefore$ OP is feasible

Analysis

- algorithm turns out to be (roughly) coordinate descent on Φ
  - each step adjusts $\mathbf{Q}$ along one coordinate direction $Q(\pi)$

Analysis

- algorithm turns out to be (roughly) coordinate descent on Φ
  - each step adjusts $\mathbf{Q}$ along one coordinate direction $Q(\pi)$
- can lower-bound how much Φ decreases on each update
- can also show rescaling step never increases Φ

Analysis

- algorithm turns out to be (roughly) coordinate descent on Φ
  - each step adjusts $\mathbf{Q}$ along one coordinate direction $Q(\pi)$
- can lower-bound how much Φ decreases on each update
- can also show rescaling step never increases Φ
- since $\Phi \geq 0$, gives bound on number of iterations of the algorithm

Analysis

- algorithm turns out to be (roughly) coordinate descent on Φ
  - each step adjusts $\mathbf{Q}$ along one coordinate direction $Q(\pi)$
- can lower-bound how much Φ decreases on each update
- can also show rescaling step never increases Φ
- since $\Phi \geq 0$, gives bound on number of iterations of the algorithm
- Theorem: On round t, algorithm halts after at most

$$\tilde{O}\left(\sqrt{\frac{Kt}{\ln |\Pi|}}\right)$$

iterations (and calls to AMO).

Analysis

- algorithm turns out to be (roughly) coordinate descent on Φ
  - each step adjusts $\mathbf{Q}$ along one coordinate direction $Q(\pi)$
- can lower-bound how much Φ decreases on each update
- can also show rescaling step never increases Φ
- since $\Phi \geq 0$, gives bound on number of iterations of the algorithm
- Theorem: On round t, algorithm halts after at most

$$\tilde{O}\left(\sqrt{\frac{Kt}{\ln|\Pi|}}\right)$$

  iterations (and calls to AMO).
- as corollary, also get bound on sparsity of $\mathbf{Q}$

Epochs and Warm Start

- so far, assumed solve OP from scratch on each round
  - naively, gives $\tilde{O}\left(T^{3/2}\right)$ calls to AMO in T rounds
  - can do much better!

Epochs and Warm Start

- so far, assumed solve OP from scratch on each round
  - naively, gives $\tilde{O}\left(T^{3/2}\right)$ calls to AMO in T rounds
  - can do much better!
- first improvement: since data iid, can use same solution for many rounds, i.e., for long "epochs"
  - gives same (near optimal) regret
  - essentially no computation required on rounds where $\mathbf{Q}$ not updated

Epochs and Warm Start

- so far, assumed solve OP from scratch on each round
  - naively, gives $\tilde{O}\left(T^{3/2}\right)$ calls to AMO in T rounds
  - can do much better!
- first improvement: since data iid, can use same solution for many rounds, i.e., for long "epochs"
  - gives same (near optimal) regret
  - essentially no computation required on rounds where $\mathbf{Q}$ not updated
- second improvement: can initialize algorithm with the previous solution (rather than starting fresh each time)
  - works because each new example can only cause Φ to increase slightly

Epochs and Warm Start (cont.)

- putting together:
  if only update $\mathbf{Q}$ on rounds $1, 4, 9, 16, 25, \ldots$
  - get same (near optimal) regret
  - only need

$$\tilde{O}\left(\sqrt{\frac{KT}{\ln|\Pi|}}\right)$$

calls to AMO total for entire sequence of T rounds

Summary

- new algorithm for contextual bandits problem with AMO access
- (nearly) optimal regret
- simple and fast
- only requires an average of

$$\tilde{O}\left(\sqrt{\frac{K}{T \ln |\Pi|}}\right) \ll 1$$

AMO calls per round

Open Problems and Future Directions

- try out experimentally
- is there an algorithm that uses an online (rather than batch) oracle?
- is there a lower bound on number of AMO calls necessary to solve this problem?
- can we find a similar algorithm that handles adversarial data?