Multi-armed Bandits

Matt Barnes

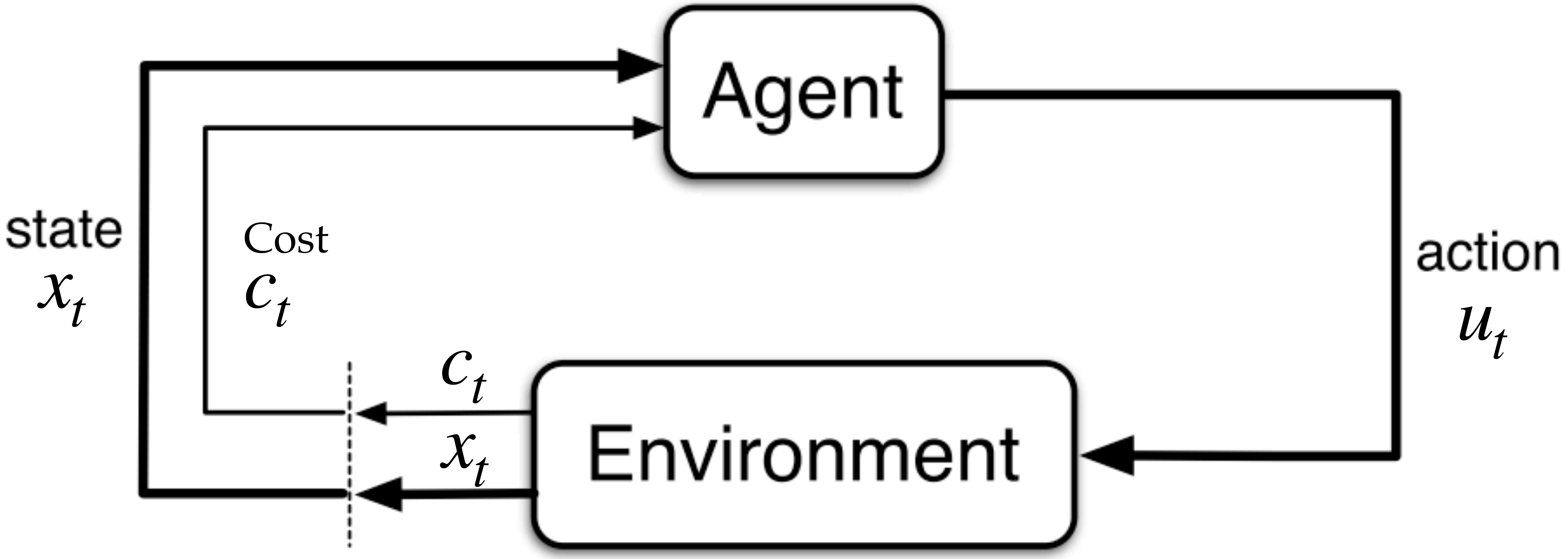
TAs: Matthew Rockett, Gilwoo Lee, Matt Schmittle

Housekeeping

- Remember to fill out course evaluations

Recap

- Markov Decision Processes are a very general class of models, which encompass planning and reinforcement learning.



Recap

"Markov" means that _____ captures all information about the history $x_1, x_2, \ldots, x_t$

The most recent state $\boldsymbol{x_t}$

Recap

- The difference between planning and reinforcement learning is whether the ____________ are known

transition model / dynamics / environment

Recap

- The three general methods for reinforcement learning are...

(1) Model-based

(2) Approximate dynamic programming

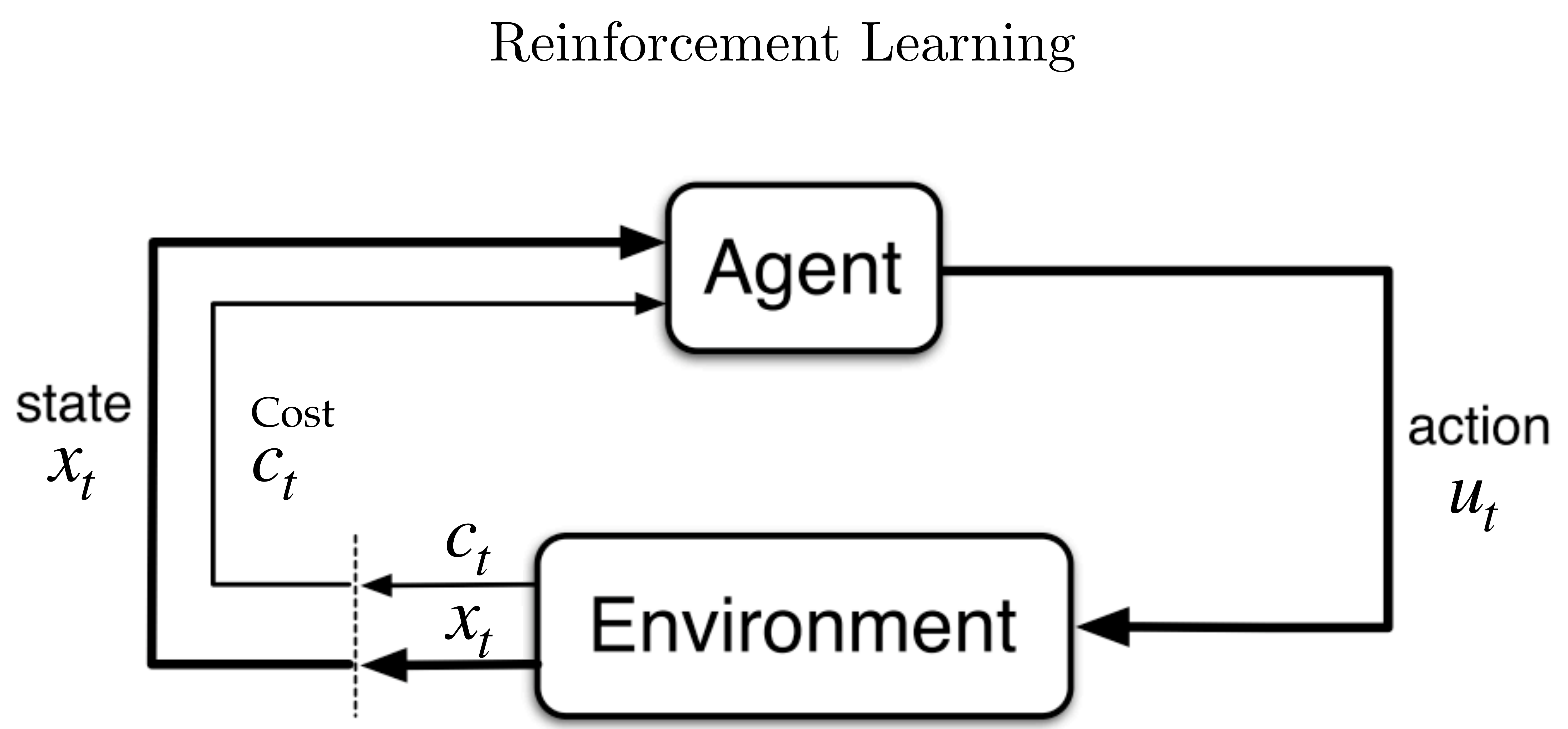
(3) Policy gradient

- (2) and (3) are both ________ methods

Model-free

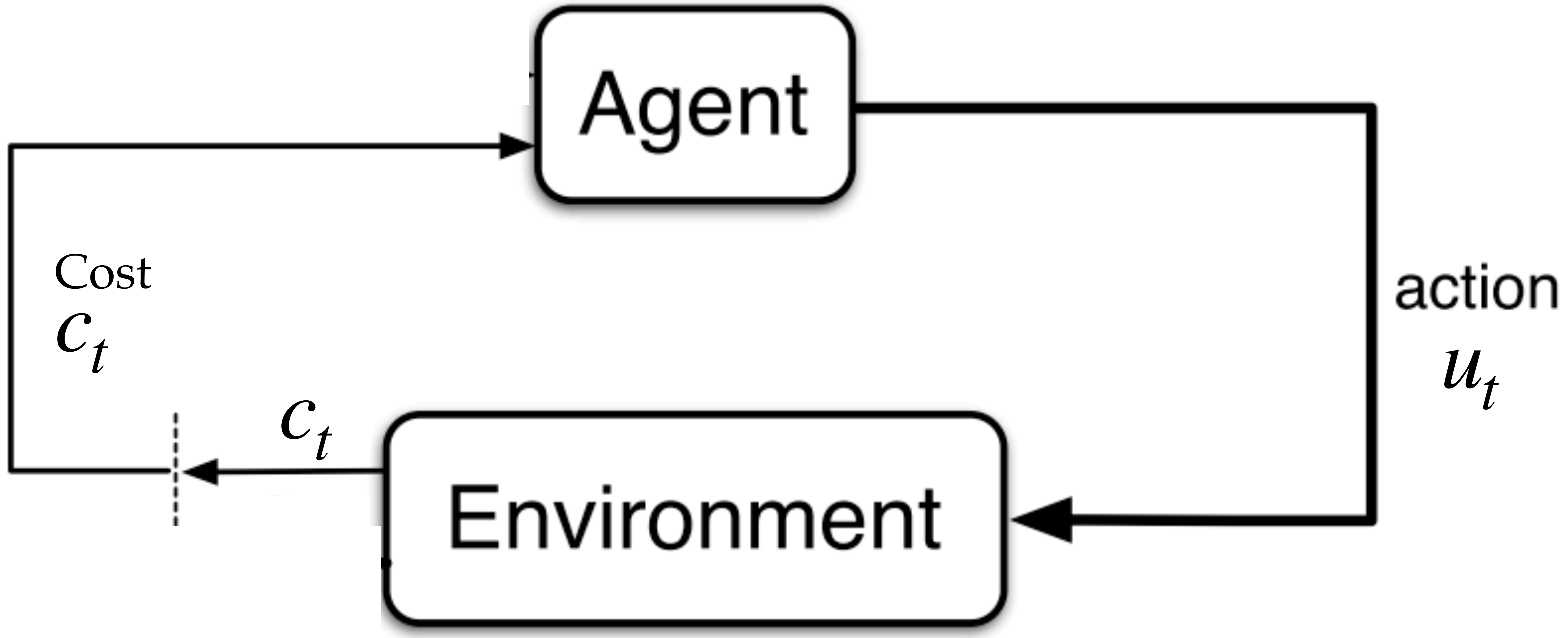
What if the MDP only has a single state?

The MDP view of bandits

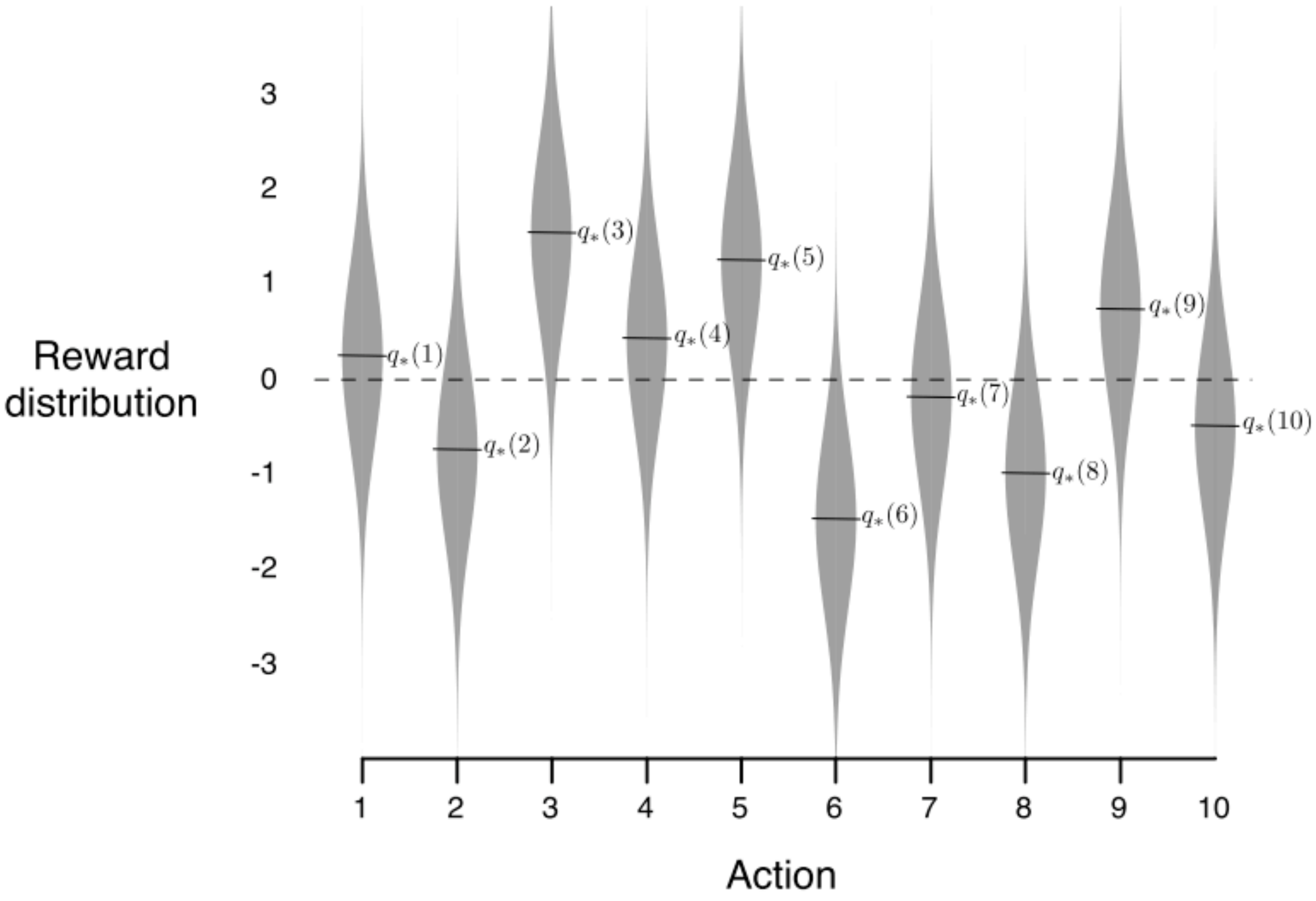


The MDP view of bandits

Multi-armed Bandits



Another view of the bandit problem



The original MAB problem

	u_1	u_2	u_3	u_4
Bandits				
t=1	\$2	-\$2	-\$1	\$10

Agent

Side note: Throughout this lecture I use “reward” and “cost” interchangeably.
You can think about reward as negative cost.

The original MAB problem

	a_1	a_2	a_3	a_4	
Bandits					Agent
t=1	$2	-$2	-$1	$10	$2

The original MAB problem

	a_1	a_2	a_3	a_4	
Bandits					Agent

t=1	$2	-$2	-$1	$10	$2
t=2	$3	-$3	$0	$5	

The original MAB problem

Bandits

	a_1	a_2	a_3	a_4	Agent
t=1	\$2	-\$2	-\$1	\$10	\$2
t=2	\$3	-\$3	\$0	\$5	-\$3

The original MAB problem

	a_1	a_2	a_3	a_4	Agent
Bandits					
t=1	$2	-$2	-$1	$10	$2
t=2	$3	-$3	$0	$5	-$3
t=3	$1	-$4	$1	$1000000	$1

The original MAB problem

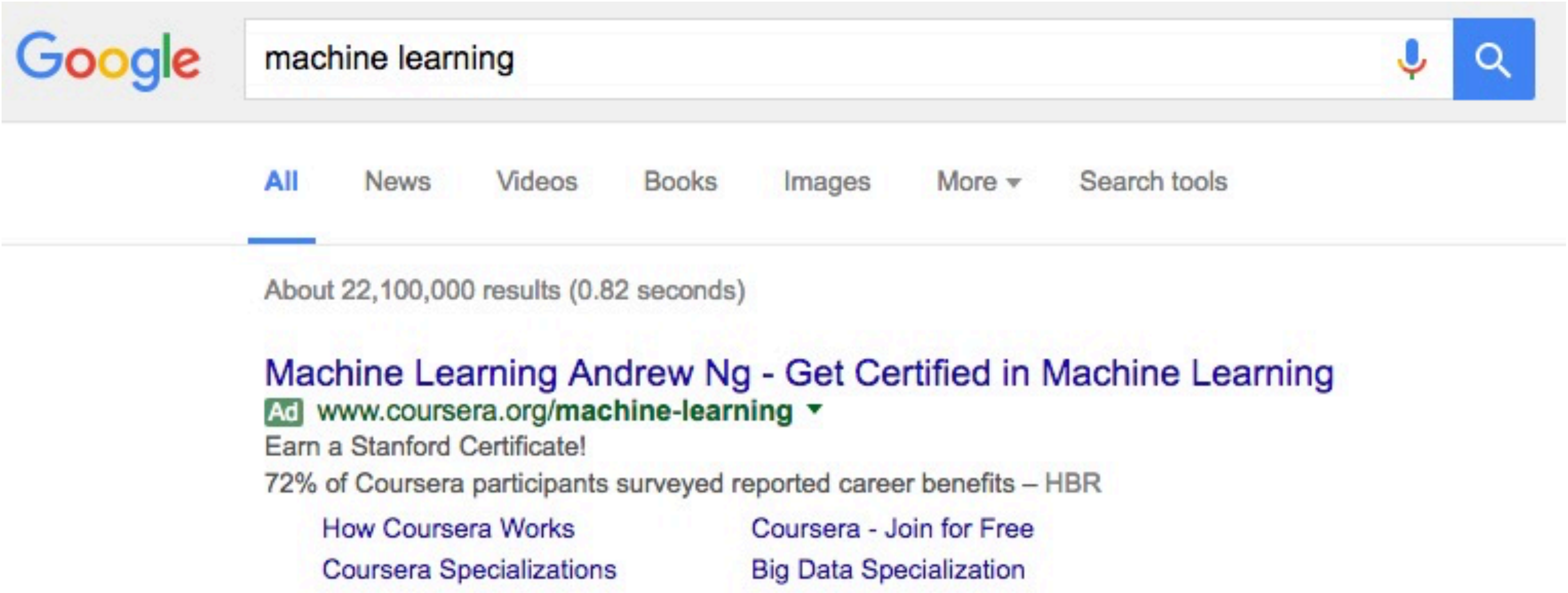
Bandits

	a_1	a_2	a_3	a_4	Agent
t=1	$2	-$2	-$1	$10	$2
t=2	$3	-$3	$0	$5	-$3
t=3	$1	-$4	$1	$1000000	$1
t=4	$3	-$4	-$5	$3	-$5

Real world bandit successes

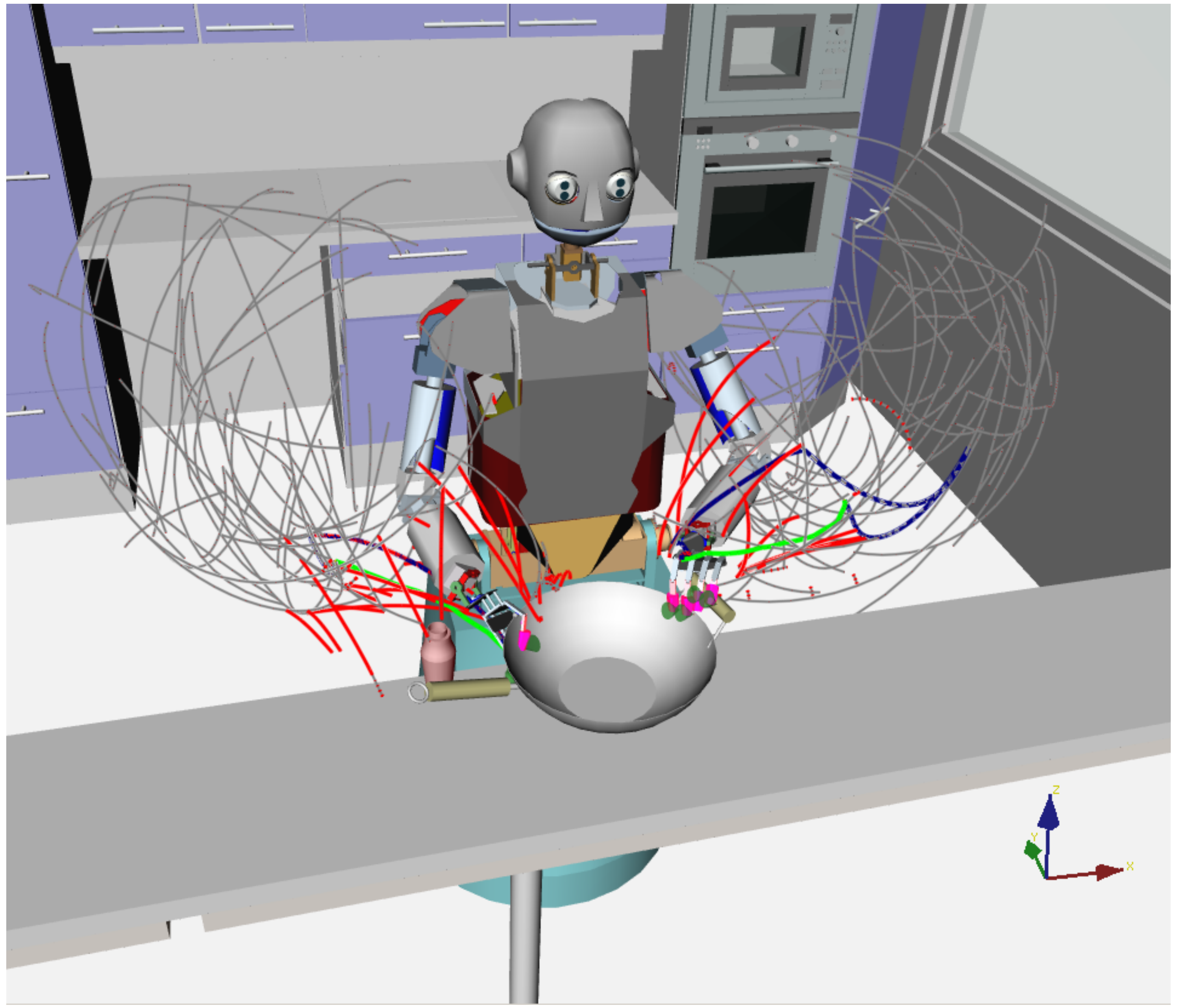
Which advertisement to display?

Reward: user clicks on the selected ad



Other exciting applications

Which grasp? Reward: robot picks up object



Other exciting applications

Which treatment? Reward: patient gets healthy

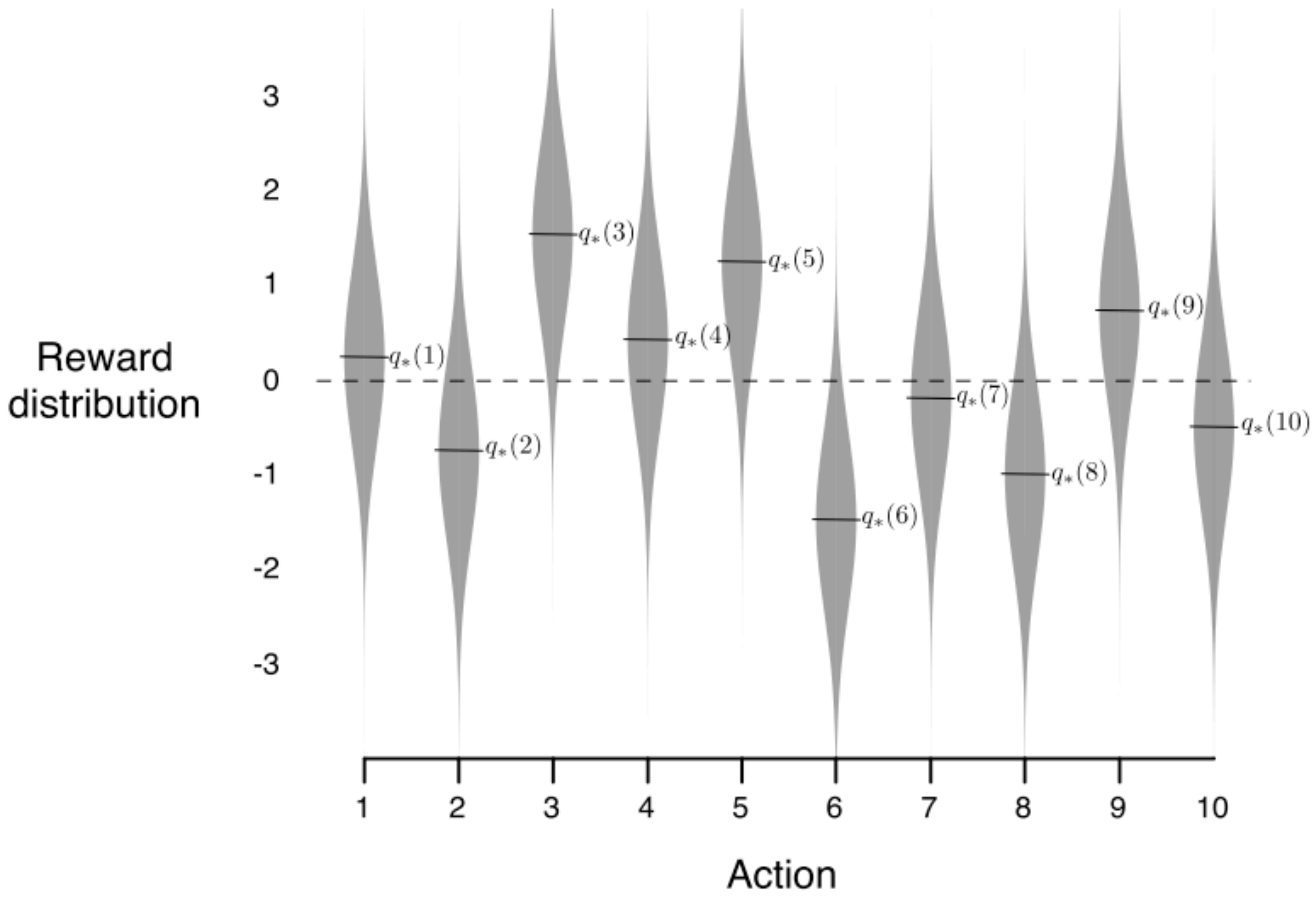
Other exciting applications

Packet routing: Which path to send data along?

The *stochastic* bandit setting

At each time-step t = *1, 2, ...*

- Choose action a_k
- Receive stochastic reward $r_t \sim \mathbb{P}_k(c)$



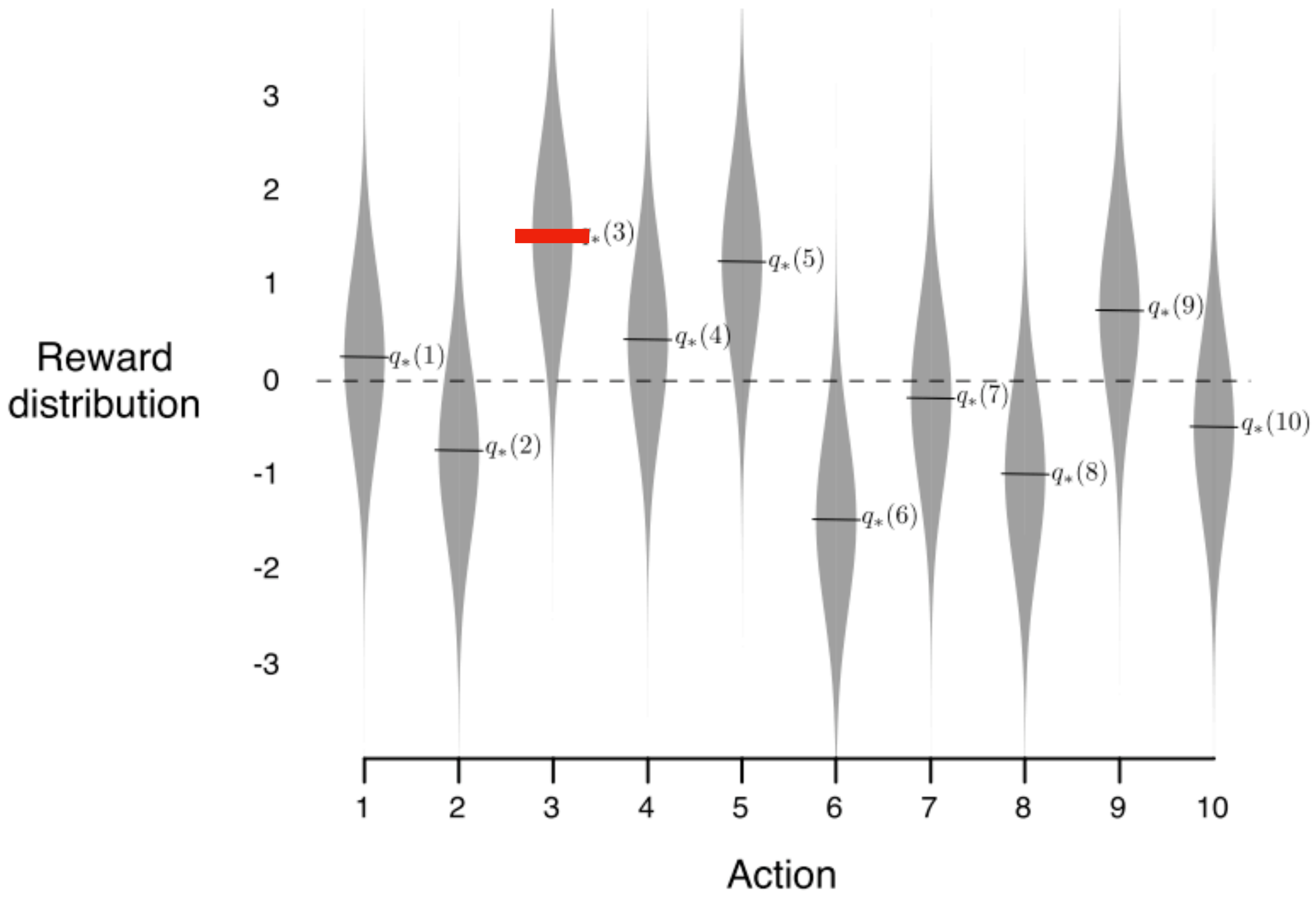
The *stochastic* bandit setting

- What is the best action?

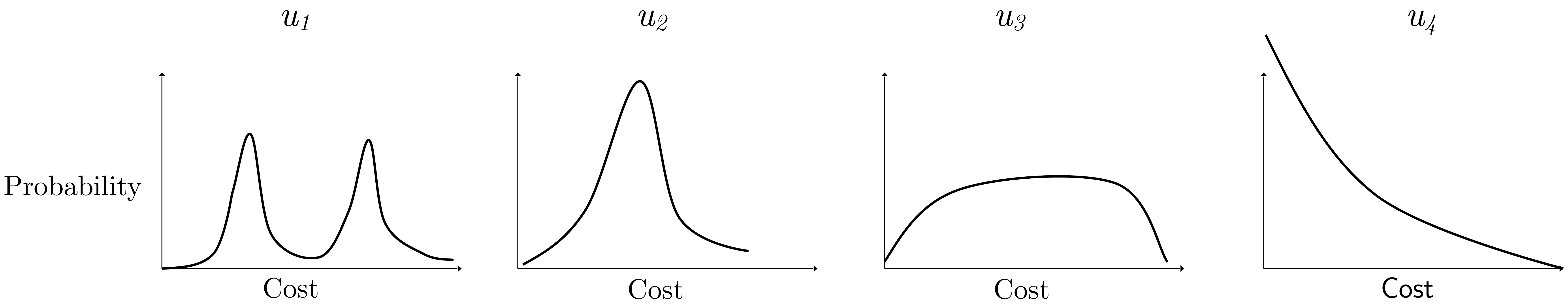
  u_3

- Why?

  Optimal action is one with highest expected reward



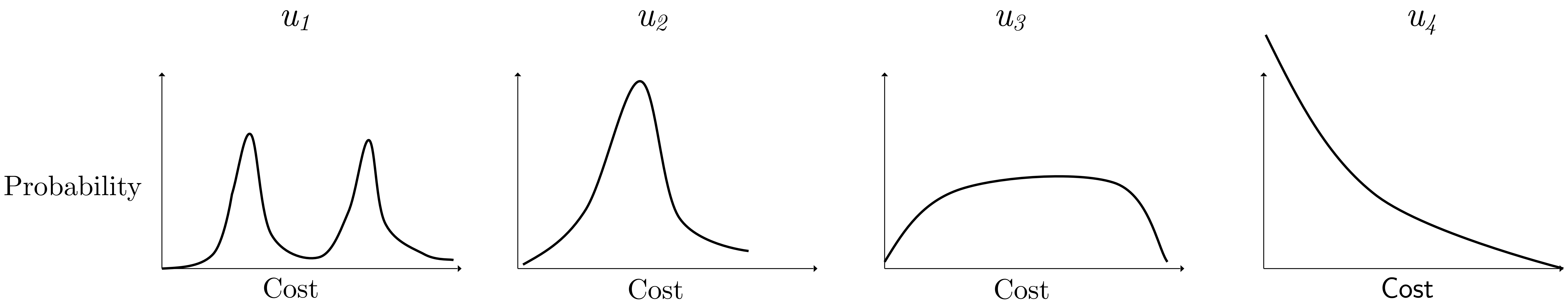
The *stochastic* bandit setting



Key challenge: How do we maximize our gambling earnings?

Requires both
(a) Playing arms we don't know much about (**exploring**)
(b) Earning money on arms we know will pay off well (**exploiting**)

A dumb algorithm

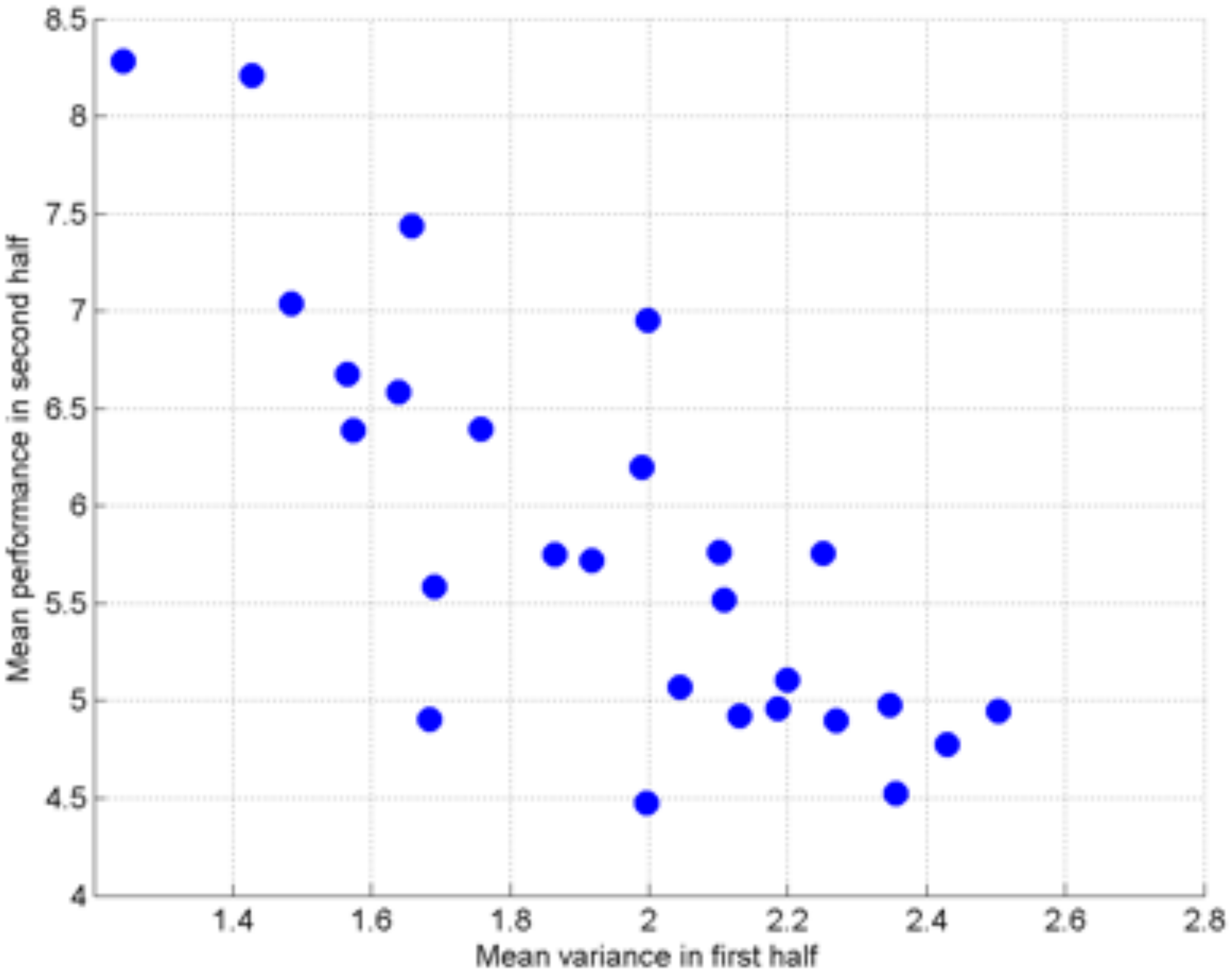


Step 1) **Explore:** Play each arm n times

Step 2) **Exploit:** Choose the arm with the best estimated reward forever

“Exploration vs. Exploitation” is a fundamental tradeoff

(worse performance)



(better performance)

Stafford, Tom, et al. "A novel task for the investigation of action acquisition." PloS one 7.6 (2012): e37749.

“Exploration vs. Exploitation” is a fundamental tradeoff

Key takeaway: Algorithm needs to transition from *exploring* to *exploiting*

A slightly less dumb algorithm: The ε-greedy algorithm

Maintain *estimates* of each action's expected cost. At each time step, choose action a_t according to

- **Exploit:** With probability *(1 - ε)* choose the action with the lowest estimated cost

- **Explore:** With probability ε, choose a random action

Update estimates.

The ε-greedy algorithm

Say we chose action k at time steps $t = t_{k_1}, \ldots, t_{k_{n_k}}$

The estimator of the expected cost is the empirical mean of the observed costs for this action:

$$\hat{c}_k = \frac{1}{n_k} \sum_{i=1}^{n_k} c_{t_{k_i}}$$

Why can't we just choose the action with the lowest estimated cost??

If $\hat{c}$ of the optimal action is less than the true expected cost of some other action, then we will never choose the optimal action!

The ε-greedy algorithm

The downside of ε-greedy

- It's pretty dumb about how it chooses to explore or exploit
- It will keep exploring forever!

A Smart Algorithm:
Upper Confidence Bound (UCB) Algorithm

- Instead of maintaining an estimate of the expected reward, greedily choose actions with the largest *upper confidence bound* on the expected reward

$$\hat{c}_k + \boxed{\sqrt{\frac{2}{n_k}\log\left(\frac{1}{\delta}\right)}}$$

Bonus which encourages exploration

The estimate of the mean (same as ε-greedy)

Number of times we have tried action u_k

Constant, trades off exploration and exploitation

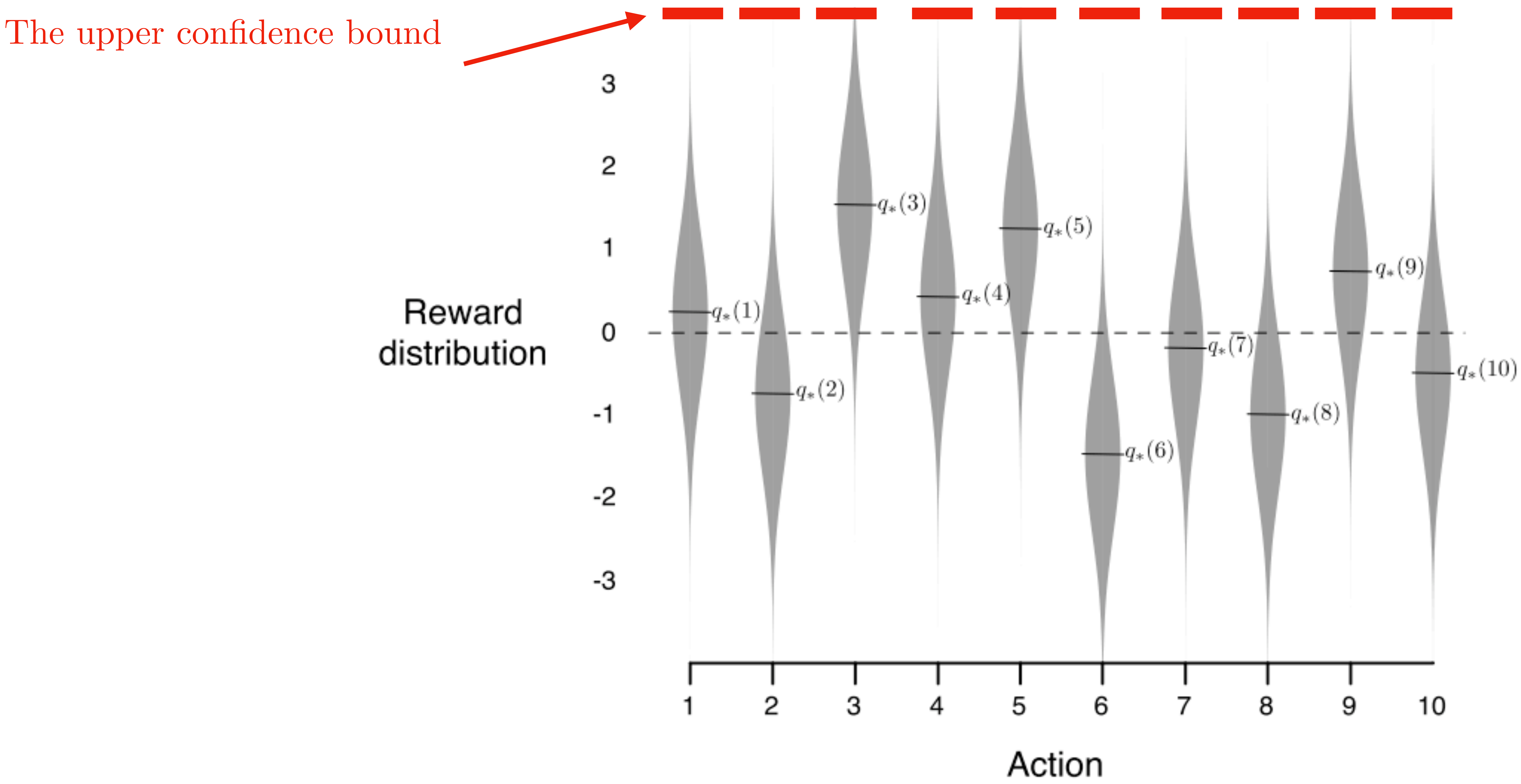
- "Optimism in the face of uncertainty." Naturally trades off choosing actions:

— With a high estimated reward

— We are uncertain about (have no tried many times)

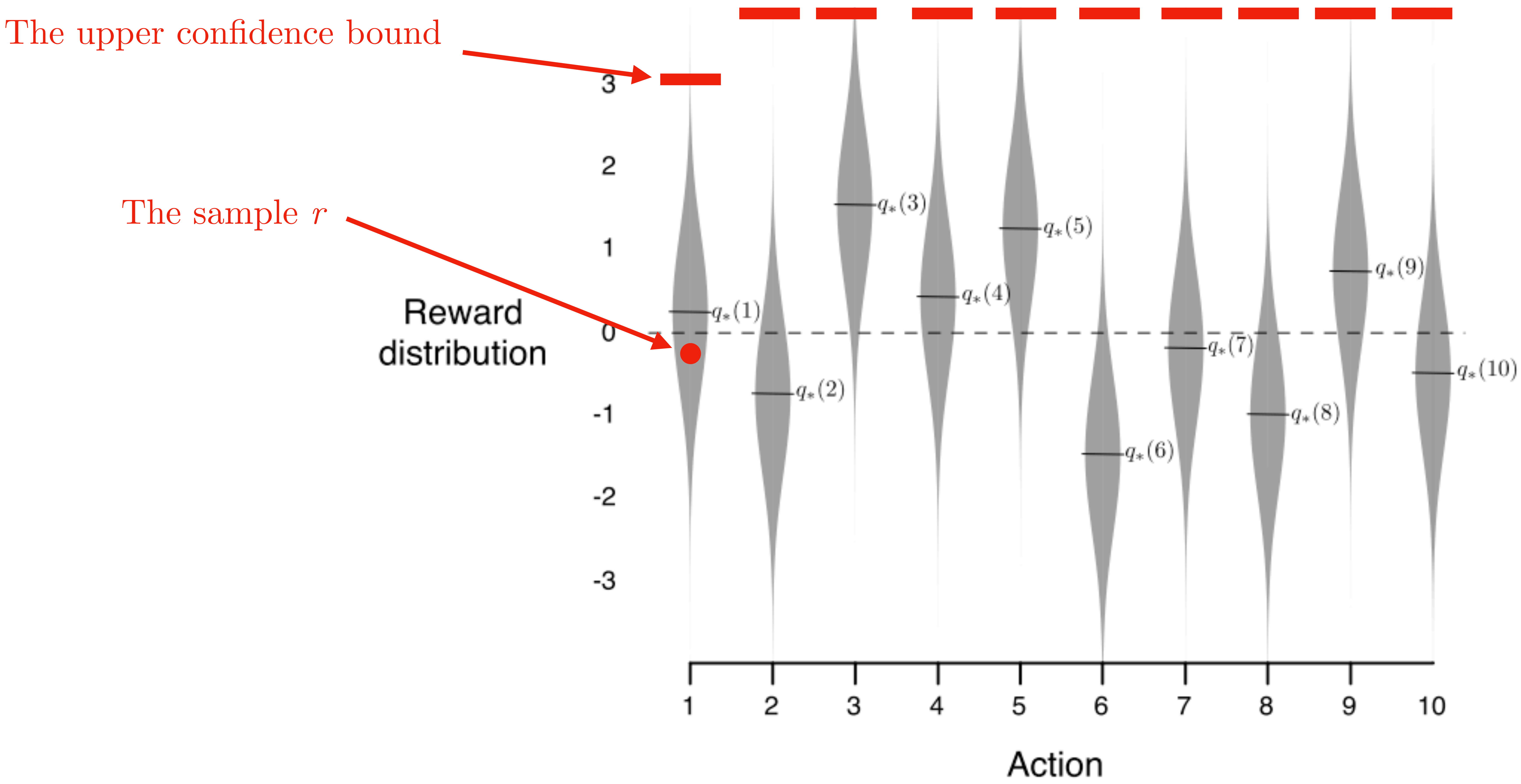
A Smart Algorithm:
Upper Confidence Bound (UCB) Algorithm

At t=0, $n_k = 0$ for all k and the upper confidence bounds are infinite.



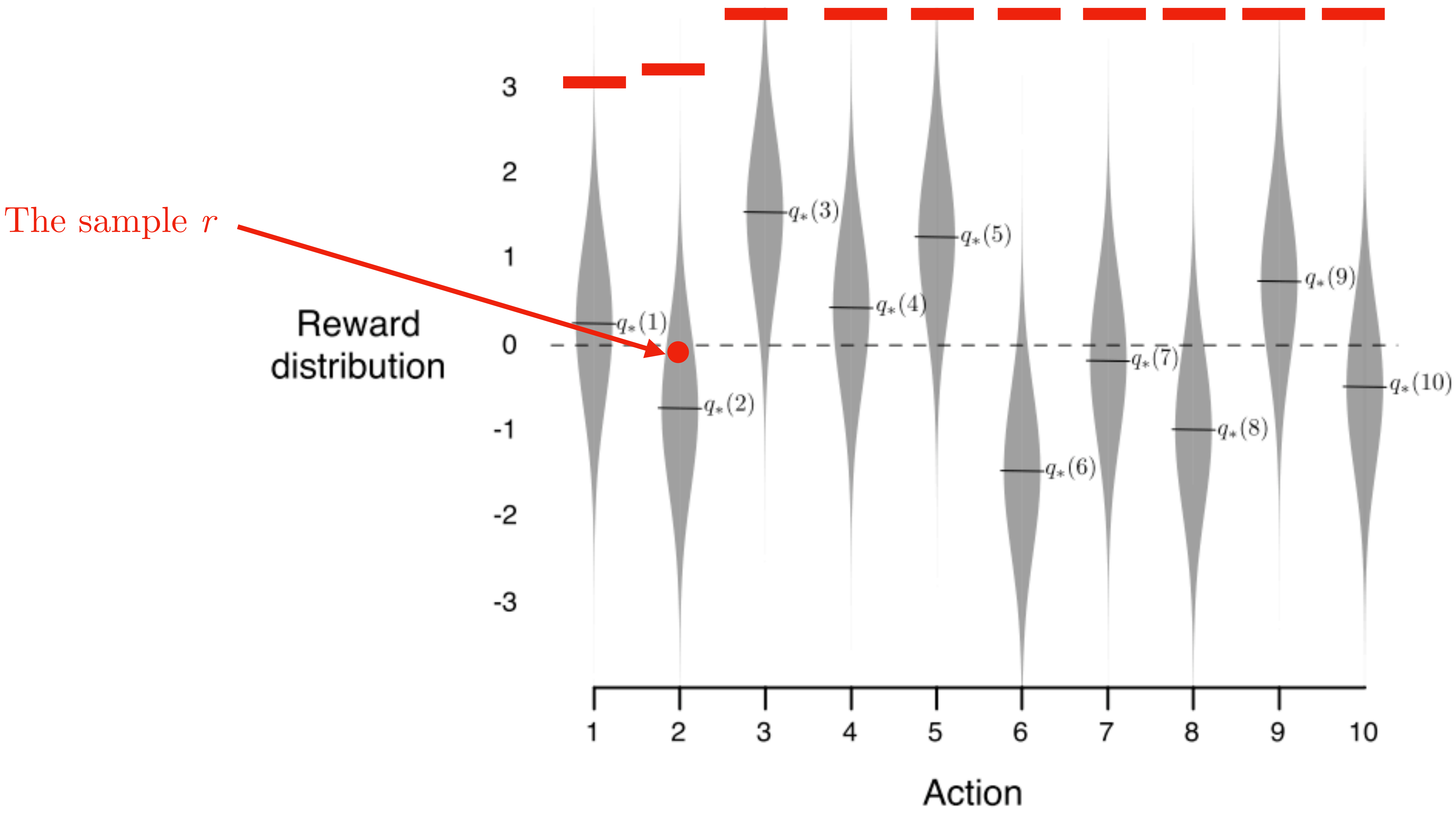
A Smart Algorithm:
Upper Confidence Bound (UCB) Algorithm

We pull the first arm



A Smart Algorithm:
Upper Confidence Bound (UCB) Algorithm

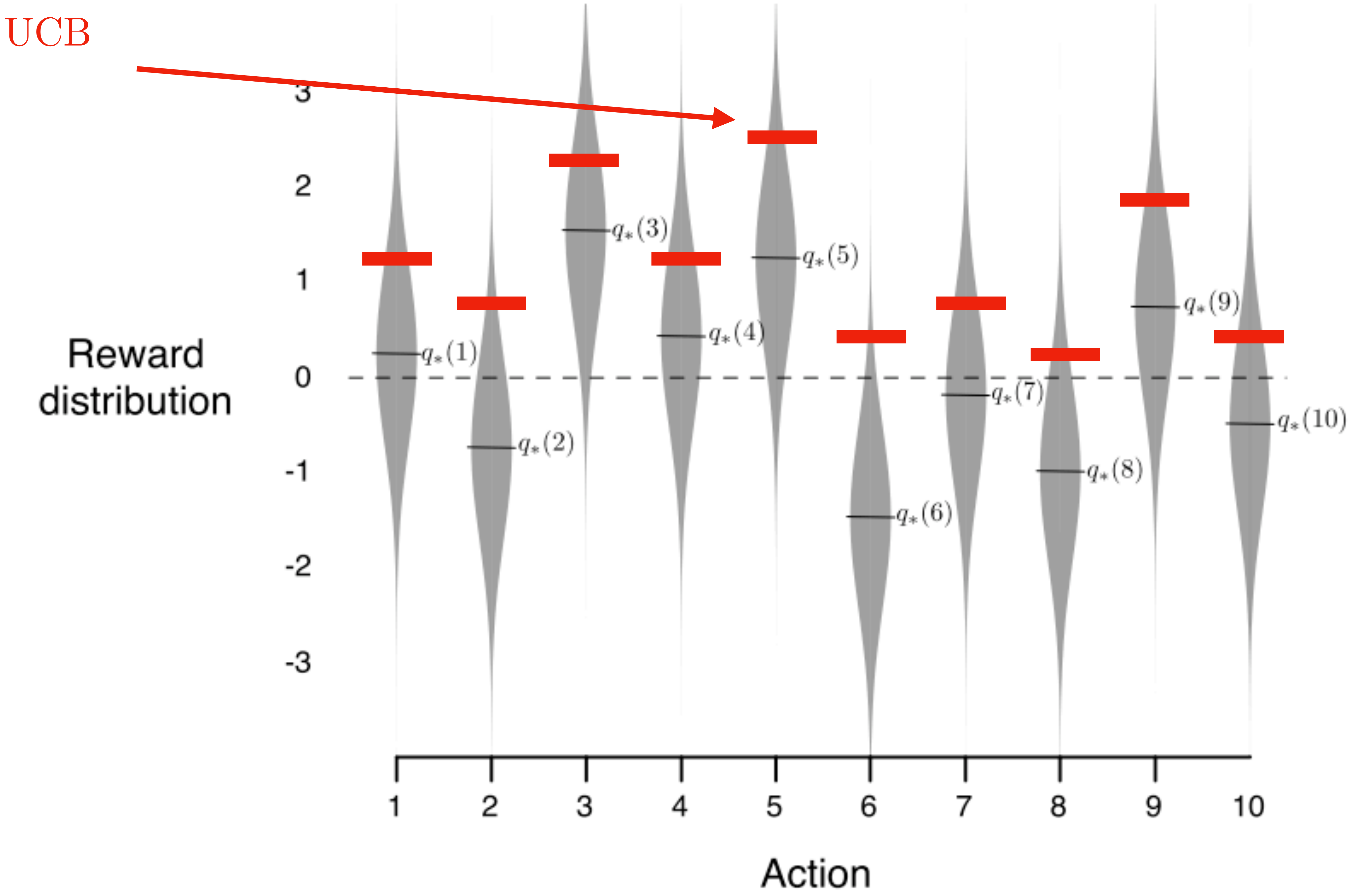
We pull the second arm



A Smart Algorithm: Upper Confidence Bound (UCB) Algorithm

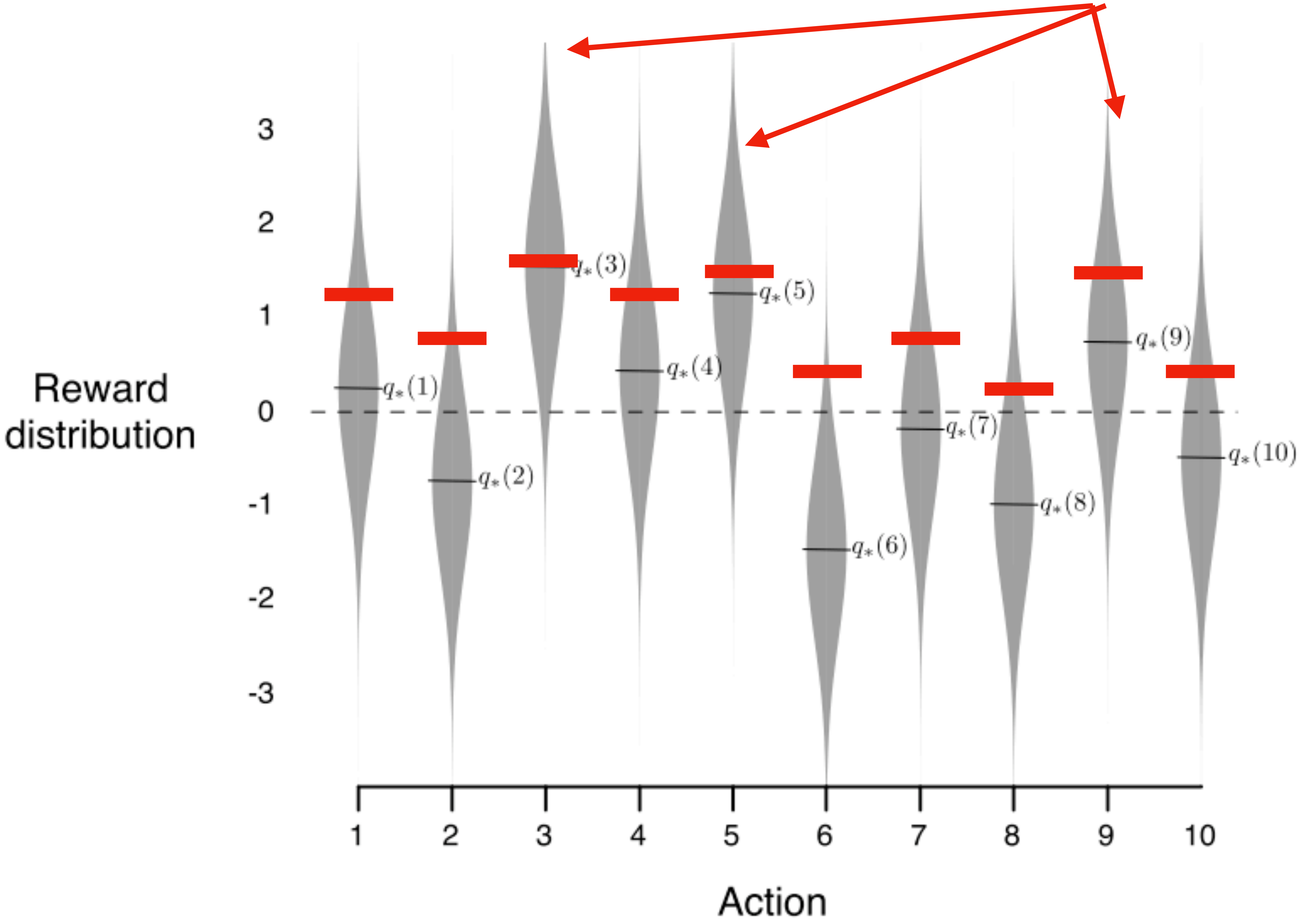
- By now, each arm has been pulled a couple times, and we have a more reasonable UCB.
- Note that these UCB's are *optimistic*, and allow us to focus our actions on promising actions in the same way as planning heuristics (e.g. A*).

Action with the best UCB



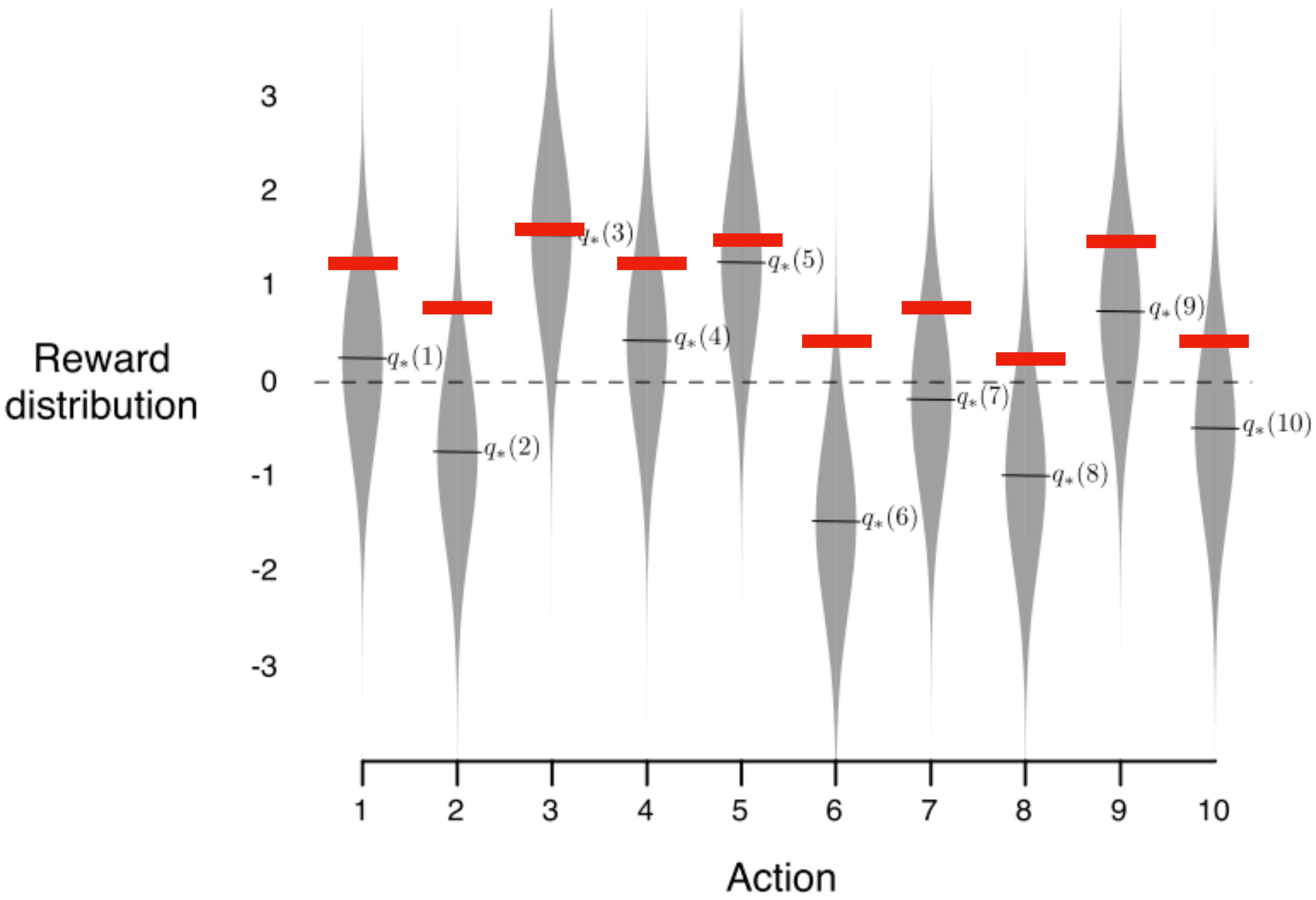
A Smart Algorithm:
Upper Confidence Bound (UCB) Algorithm

Only need to pull these three arms!
Will effectively stop exploring once all UCB's are less than r_3

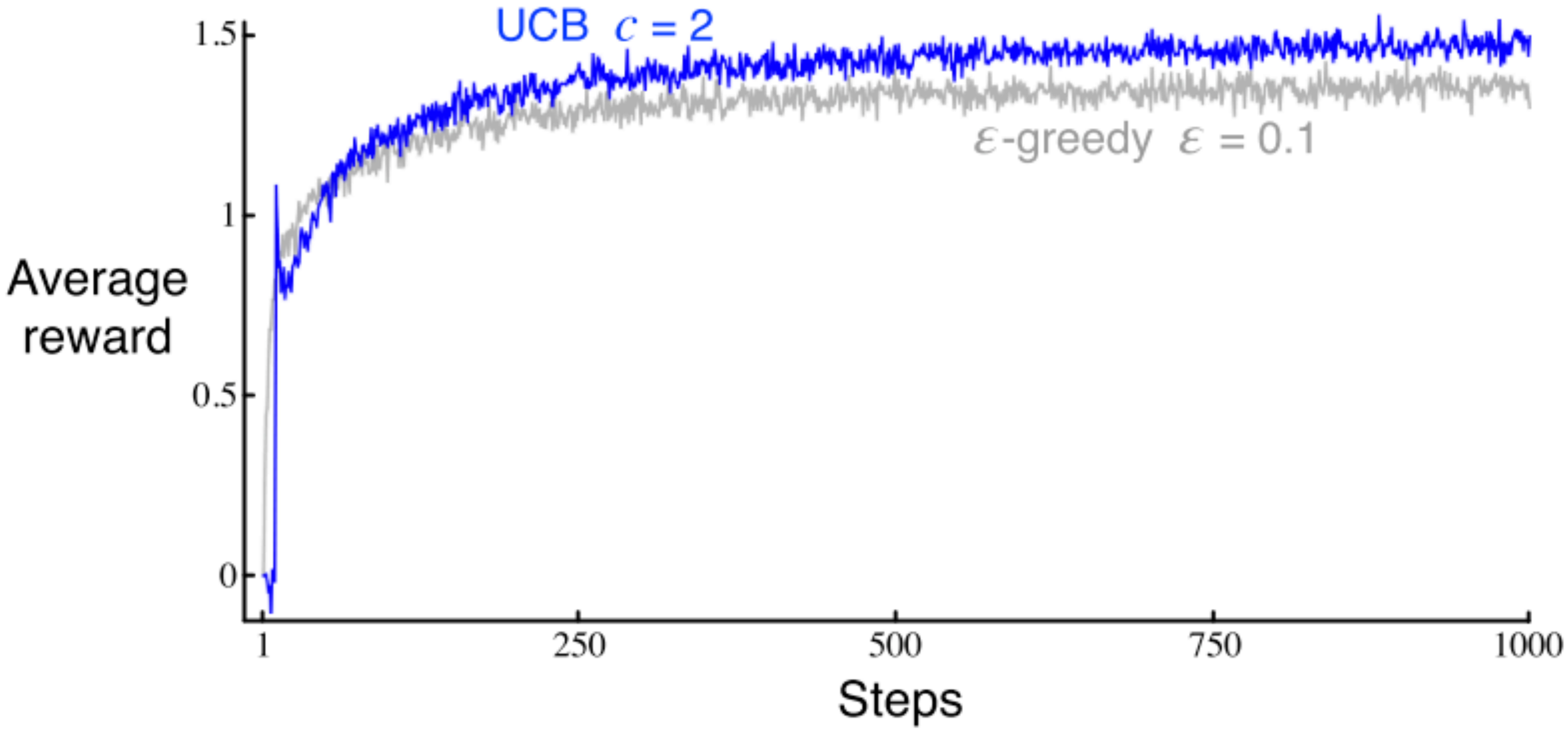


A Smart Algorithm:
Upper Confidence Bound (UCB) Algorithm

- Focus **exploration** on more promising arms, evident by the better arms having tighter UCB'
- Naturally transitions from exploration to **exploitation**



UCB outperforms ε-greedy in practice (and in theory)



Contextual bandits are a powerful variation of the classic bandit problem

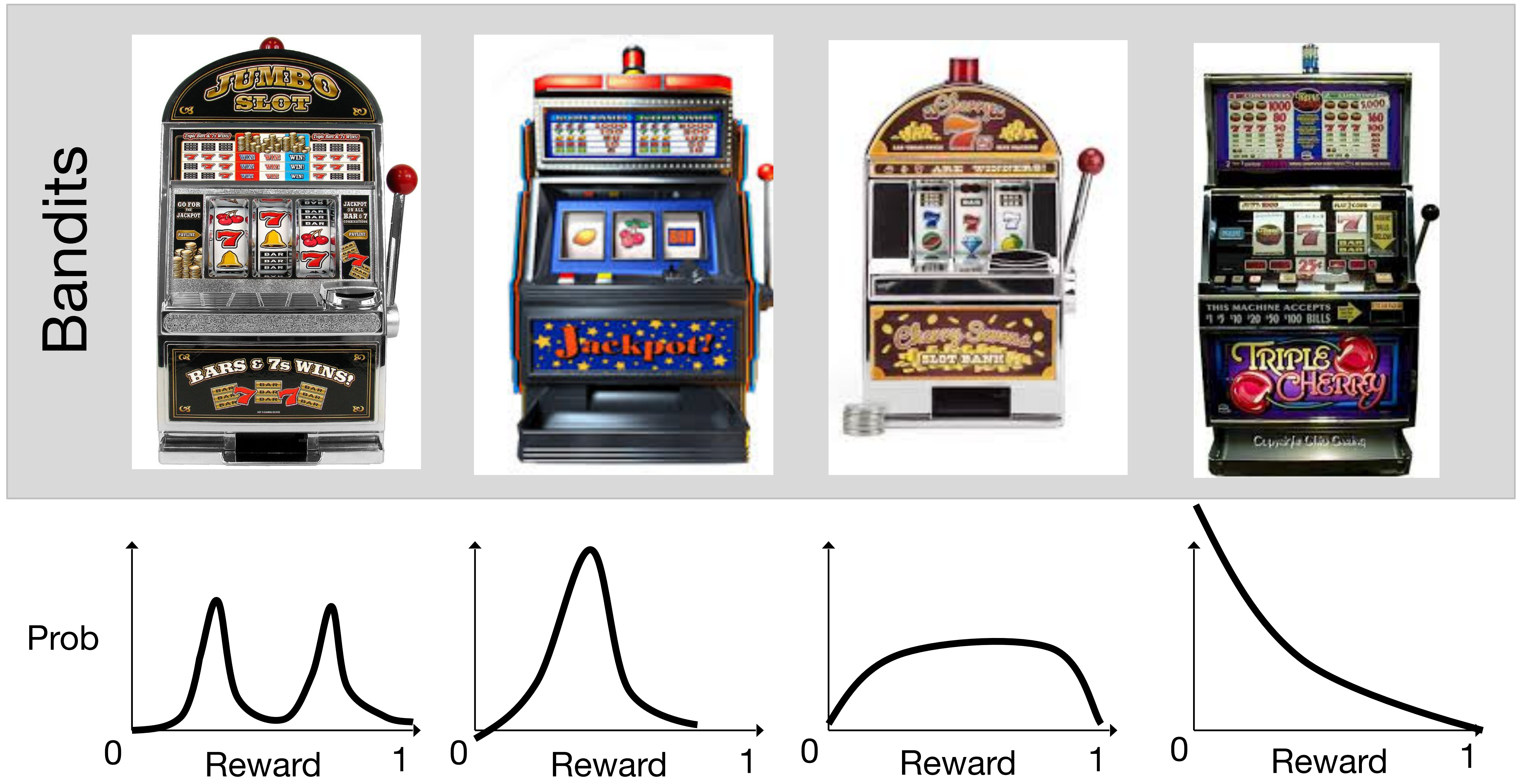
Example: Advertising – which ad to show user?

Additional context: User history, type of ad, IP address, time of day

Reward: Does user click the ad?

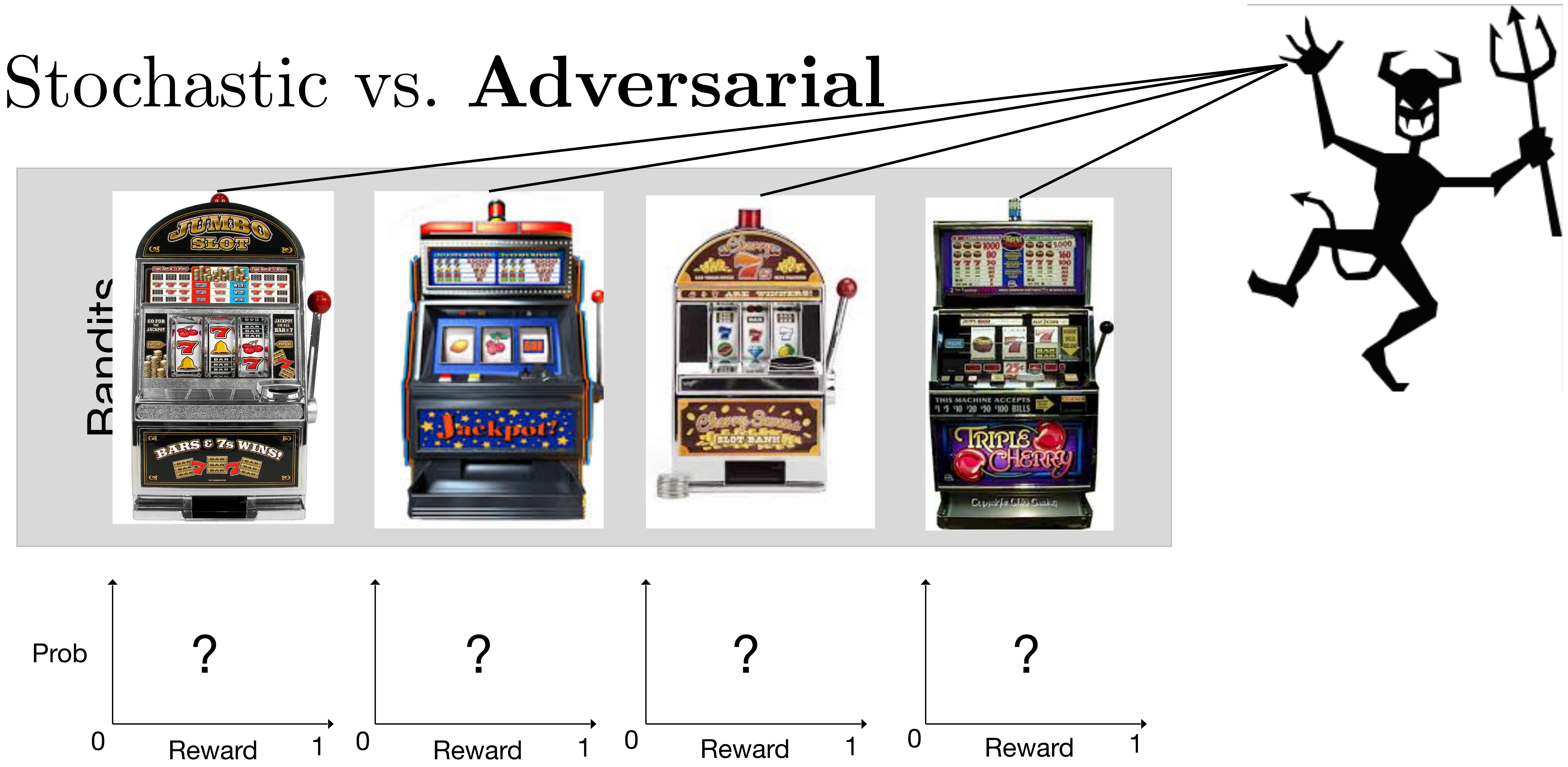
Another variation: Adversarial rewards

Stochastic vs. Adversarial



- Stochastic bandits assume reward drawn i.i.d. from some bounded distribution

Another variation: Adversarial rewards



- Adversarial bandits only assume reward is bounded, samples not drawn IID from some distribution
- Surprisingly, we can prove strong guarantees in this setting, too (EXP3 algorithm)

Recap

The multi-armed bandit problem is reinforcement learning with __ state(s)

1

Recap

What is the fundamental trade-off in bandit (and reinforcement learning) problems?

Exploration vs. exploitation

Recap

Reward is equivalent to _________

Negative cost

Recap

The ε-greedy algorithm randomly explores with probability

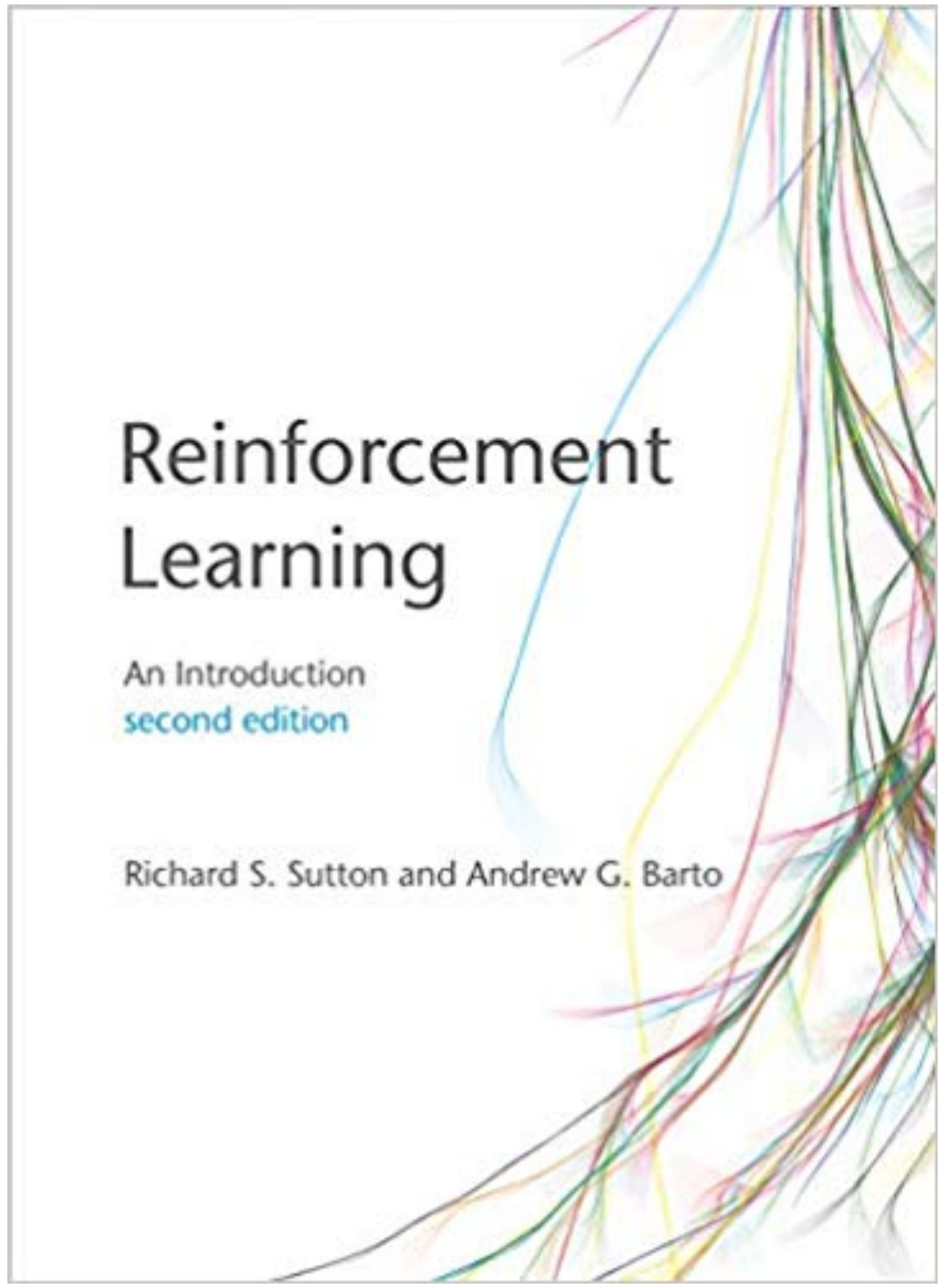
$\boldsymbol{\varepsilon}$

Recap

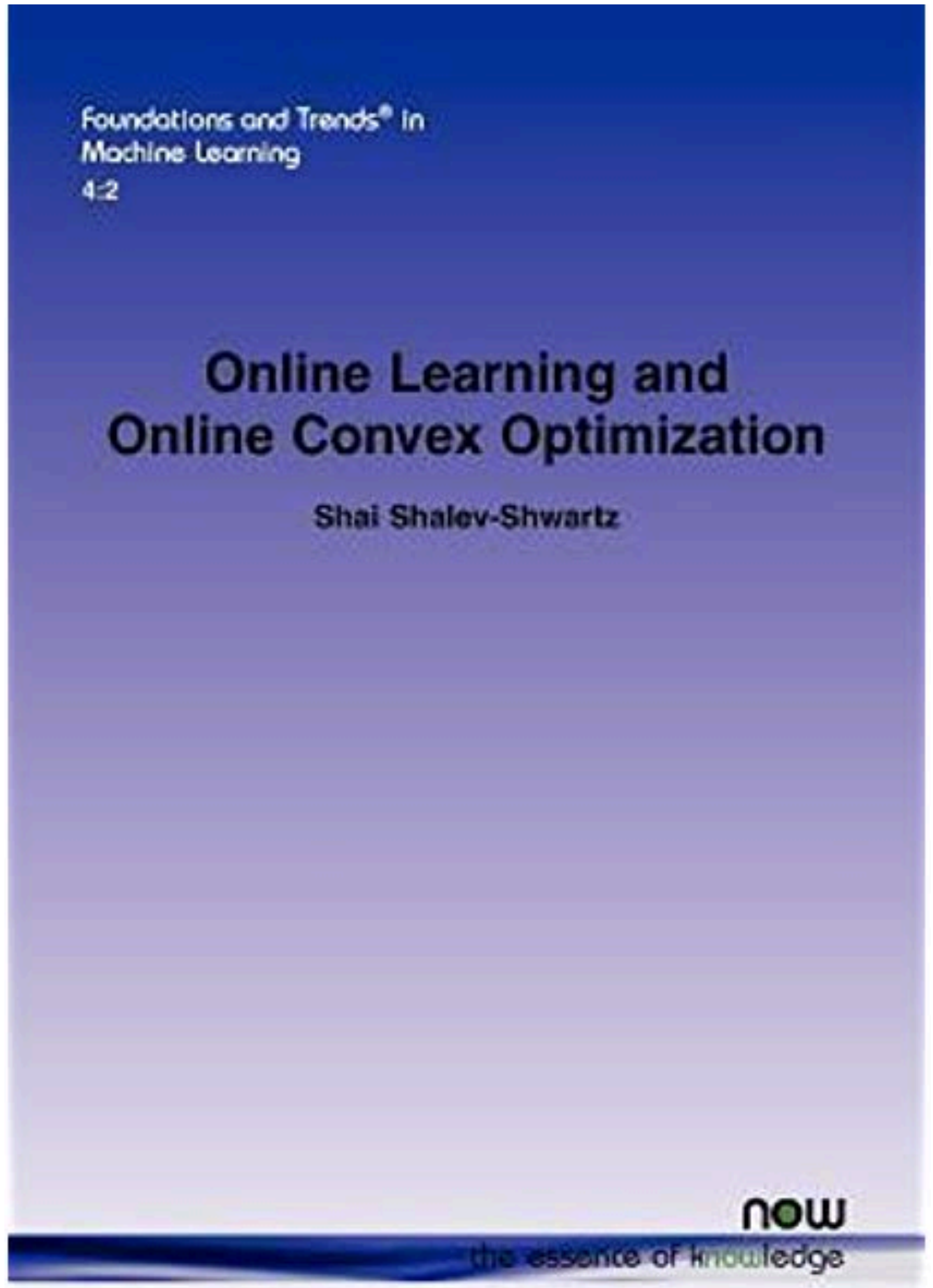
The UCB algorithm chooses actions according to the estimated reward, plus a bonus (i.e. confidence interval) which decreases with respect to _____

The number of times we try that action.

Further Reading



Chapter 2: Multi-armed bandits



Advanced reading:
Connection to online convex optimization

Preview of the next lecture

- **Imitation learning** — what if we have help from a (human) expert?