Lecture 12: Fast Reinforcement Learning

Emma Brunskill

CS234 Reinforcement Learning

Winter 2025

- With some slides from or derived from David Silver, Examples new

Refresh Your Understanding: Multi-armed Bandits

$N_t(a)$ = # pulls of action a at or before t

$\hat{Q}_t(a)$ = empirical estimate of expected reward for arm a

- Select all that are true:

T 1. Algorithms that minimize regret also maximize reward

T 2. Up to variations in constants, ignoring δ, UCB selects the arm with $\arg\max_a \hat{Q}_t(a) + f(\sqrt{\frac{1}{N_t(a)}})$ (where $f(x)$ indicates a function of x)

T 3. UCB still would likely learn to pull the optimal arm more than other arms if we instead used $\arg\max_a \hat{Q}_t(a) + \sqrt{\frac{1}{\sqrt{N_t(a)}} \log(t/\delta)}$

T 4. UCB uses $\arg\max_a \hat{Q}_t(a) + b$ where b is a bonus term. Consider $b = 5$. This will make the algorithm optimistic with respect to the empirical rewards but it may still cause such an algorithm to suffer linear regret.

T 5. A k-armed multi-armed bandit is like a single state MDP with k actions

6. Not Sure

pulls = taking an action

arms = actions

Refresh Your Understanding: Multi-armed Bandits Solution

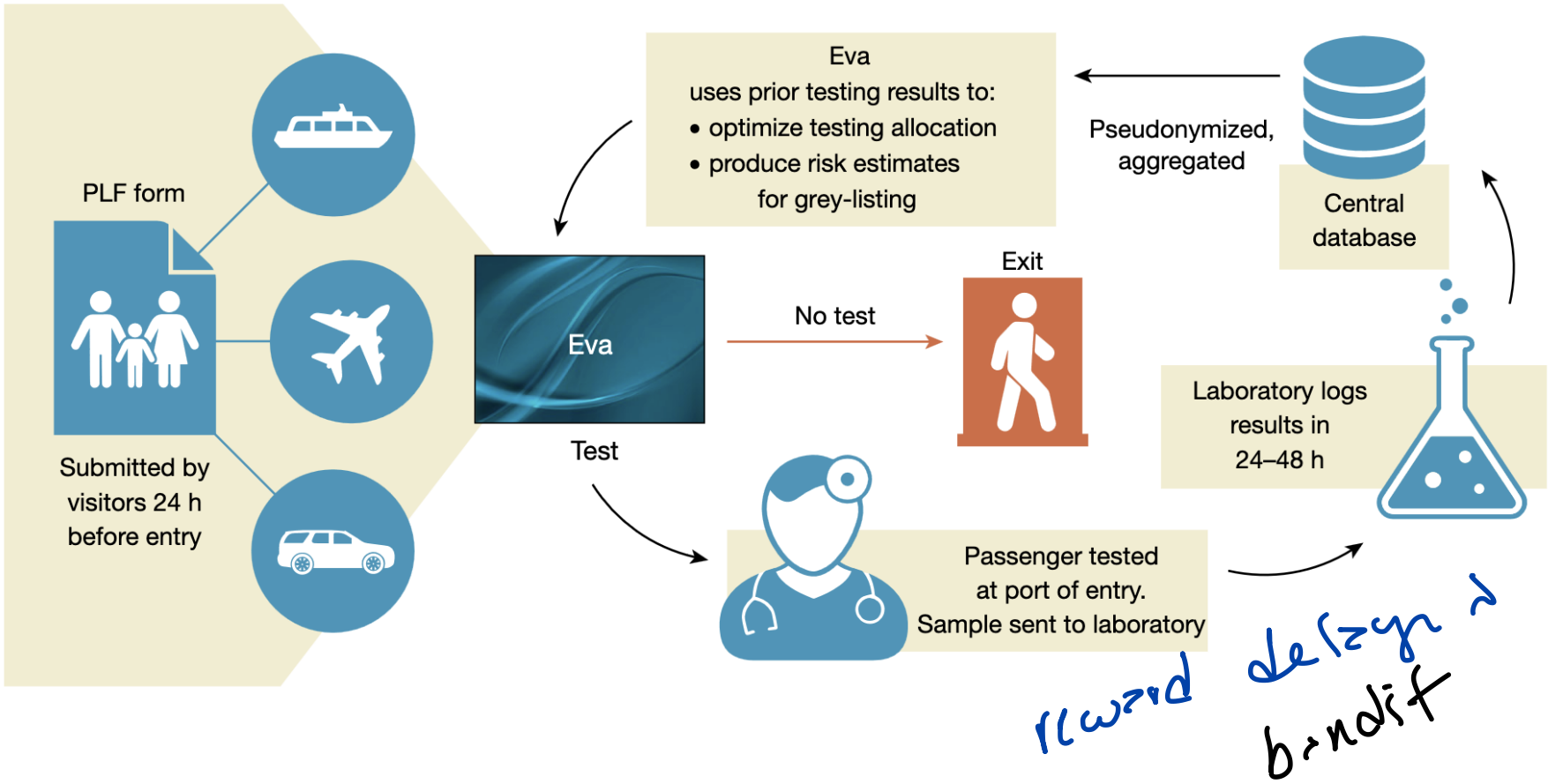
- Select all that are true:
  1. Algorithms that minimize regret also maximize reward
  2. Up to variations in constants, ignoring δ, UCB selects the arm with $\arg\max_a \hat{Q}_t(a) + f(\sqrt{\frac{1}{N_t(a)}})$ (where $f(x)$ indicates a function of x)
  3. UCB still would likely learn to pull the optimal arm more than other arms if we instead used $\arg\max_a \hat{Q}_t(a) + \sqrt{\frac{1}{\sqrt{N_t(a)}} \log(t/\delta)}$
  4. UCB uses $\arg\max_a \hat{Q}_t(a) + b$ where b is a bonus term. Consider $b = 5$. This will make the algorithm optimistic with respect to the empirical rewards but it may still cause such an algorithm to suffer linear regret.
  5. A k-armed multi-armed bandit is like a single state MDP with k actions
  6. Not Sure

  Solutions: True. True. True. True. True

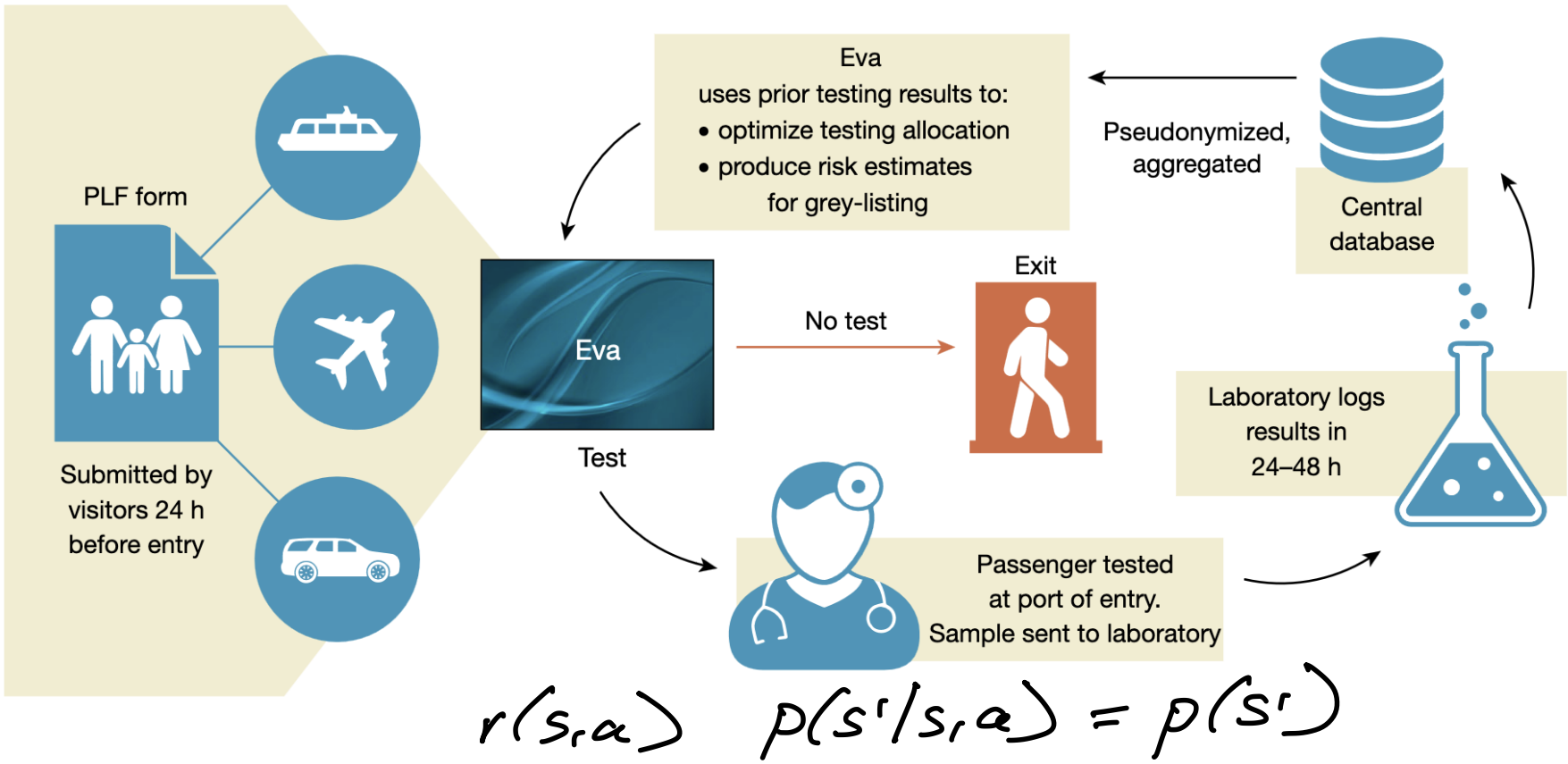
Where We are

- Last time: Bandits and regret and UCB (fast learning)
- This time: Bandits, regret and bayesian bandits (fast learning)
- Next time: Bayesian learning and MDPs (fast learning)

Deciding Who To Test for Covid. Bastani et al. Nature 2001



Deciding Who To Test for Covid. Bastani et al. Nature 2001



$r(s,a)$ $p(s'|s,a) = p(s')$

- A *nonstationary, contextual, batched bandit problem with delayed feedback and constraints*

Today

- Bandits and regret
- Bandits and Probably Approximately Correct
- Bayesian bandits
- Thompson sampling
- Bayesian Regret

Multiarmed Bandits Notation Recap

- Multi-armed bandit is a tuple of $(\mathcal{A}, \mathcal{R})$
- $\mathcal{A}$: known set of m actions (arms)
- $\mathcal{R}^a(r) = \mathbb{P}[r \mid a]$ is an unknown probability distribution over rewards
- At each step t the agent selects an action $a_t \in \mathcal{A}$
- The environment generates a reward $r_t \sim \mathcal{R}^{a_t}$
- Goal: Maximize cumulative reward $\sum_{\tau=1}^{t} r_\tau$
- **Regret** is the opportunity loss for one step

$$l_t = \mathbb{E}[V^* - Q(a_t)]$$

- **Total Regret** is the total opportunity loss

$$L_t = \mathbb{E}[\sum_{\tau=1}^{t} V^* - Q(a_\tau)]$$

- Maximize cumulative reward $\iff$ minimize total regret

Optimism Under Uncertainty: Upper Confidence Bound (UCB) Algorithm

- UCB1 algorithm

$$a_t = \arg\max_{a \in \mathcal{A}} \left[\hat{Q}(a) + \sqrt{\frac{2 \log \frac{1}{\delta}}{N_t(a)}} \right]$$

δ prob that this UCB holds

Regret Bound for UCB Multi-armed Bandit Sketch (See Section 7.1 of Bandit Algorithms book)

Done on Board

Thm 7.1. Consider UCB on a stochastic k armed bandit with rewards in $(0,1)$. For any horizon n, if $\delta = 1/n^2$ then

$$\text{Regret}_n \leq 3\sum_{i=1}^{K} \Delta_i + \sum_{i:\Delta_i>0} \frac{16 \log n}{\Delta_i}$$

$\Delta_i = Q(a^*) - Q(a_i)$ $\hat{Q}$ = empirical, no hat = true

Assume w/out loss of generality $a^* = a_1$

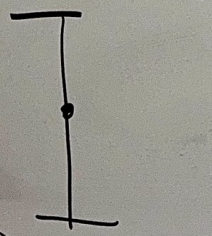
(1) $$\text{Regret}_n = \sum_{i=1}^{K} \Delta_i E[N_t(a_i)]$$

Good event

$$G_i = \left\{ Q(a_1) < \min_{t \in n} UCB_i(t, \delta) \right\} \cap \quad (*)$$

$$(**) \quad \left\{ \hat{Q}_{U_i}(a_i) + \sqrt{\frac{2}{U_i} \log \frac{1}{\delta}} < Q_t(a_i) \right\}$$

U_i = some number of pulls for arm i



(2) a_i $a_i \neq a_1$

(3)

$$E[N_t(a_i)] = E[1(G_i = T) N_t(a_i)] + E[1(G_i^c = T) N_t(a_i)]$$

$$\leq u_i + P(G_i^c = T)\, n \qquad \text{(iv)}$$

u_i: need to prove

$P(G_i^c = T)\, n$: next look at this

Proof by Contradiction: Assume $N_t(a_i) > u_i$ and $G_i = 1$ (good event holds)

$N_{t-1}(a_i) = u_i$

(4) $UCB_i(t-1, \delta) = \hat{Q}_{t-1}(a_i) + \sqrt{\frac{2\log(1/\delta)}{N_{t-1}(a_i)}}$ defn UCB

$= \sqrt{\frac{2\log 1/\delta}{u_i}}$

$< Q(a_i)$ by **

$< UCB_i(t-1, \delta)$ by *

Contradiction because UCB pulls argmax_a UCB

Goal upper bound $P(G_i^c = T)$

1st consider *

$$Q(a_1) \geq \min_{t \in n} UCB_1(t, \delta) = \bigcup_{s \in n} \left\{ Q(a_1) \geq \hat{Q}_s(a_1) + \sqrt{\frac{2 \log \frac{1}{\delta}}{s}} \right\}$$

union bound

$$P\left(Q(a_1) \geq \min_{t \in n} UCB_1(t, \delta)\right) \leq \sum_{s=1}^{n} P\left(Q(a_1) \geq \hat{Q}_s(a_1) + \sqrt{\frac{2 \log \frac{1}{\delta}}{s}}\right)$$

$$\leq n\delta$$

$$P(G_i^c = T_i) \leq n\delta + \exp\left(\frac{-u_i c^2 \Delta_i^2}{2}\right)$$

sub. back to iv

$$E[N_t(a_i)] \leq u_i + n\left(n\delta + \exp\left(\frac{-u_i c^2 \Delta_i^2}{2}\right)\right) \quad \text{(eq. vi)}$$

Now choose u_i such that Eqn v holds

$$(1-c)\Delta_i \geq \sqrt{\frac{2\log 1/\delta}{u_i}}$$

$$(1-c)^2 \Delta_i^2 \geq \frac{2\log 1/\delta}{u_i}$$

$$u_i \geq \left\lceil \frac{2\log 1/\delta}{(1-c)^2 \Delta_i^2} \right\rceil$$

$\lceil \cdot \rceil$ ceiling

$$E[N_t(a_i)] \leq \left\lceil \frac{2 \log 1/\delta}{(1-c)^2 \Delta_i^2} \right\rceil + 1 + n^{1 - 2c^2/(1+c^2)}$$

recall $\delta = 1/n^2$

$c = .5$

$$E[N_t(a_i)] \leq 3 + \frac{\log n}{\Delta_i^2}$$

$$(1-c)\Delta_i \geq \sqrt{\frac{2\ln(1/\delta)}{v_i}}$$

$$(1-c)^2\Delta_i^2 \geq \frac{2\ln 1/\delta}{v_i}$$

$$v_i \geq \frac{2\ln 1/\delta}{(1-c)^2\Delta_i^2}$$ for finite Δ_i always po

say $\delta = 1/n^2$

$$= v_i + n(n\delta + \exp(-v_i c^2\Delta_i^2/2))$$

$$= v_i + 1 + n\exp(-v_i c^2\Delta_i^2/2)$$

$$= \frac{2\ln n^2}{(1-c)^2\Delta_i^2} + 1 + n\exp\left(\frac{-2\ln n^2}{2(1-c)^2\Delta_i^2}\right)$$

$$= \quad " \quad + 1 + n$$

$$n\exp\left(-\frac{2\log(1/\delta)}{2(1-c)^2\Delta_i^2}c^2\Delta_i^2\right)$$

$$= n\exp\left(\frac{c^2}{(1-c)^2}\log\delta\right)$$

$$= n\exp\left(\log\delta^{c^2/(1-c)^2}\right)$$

$$= n\,\delta^{c^2/(1-c)^2}$$

$$= n\cdot\left(\frac{1}{n^2}\right)^{c^2/(1-c)^2}$$

$$= n\cdot(n^{-2})^{c^2/(1-c)^2}$$

$$= n\cdot n^{-2c^2/(1-c)^2}$$

$$= n^{1-2c^2/(1-c)^2}$$

Additional intermediary steps for last part of derivation

choose $c = 1/2$ then $n^{1-(2\cdot 1/4\ /\ 1/4)} = n^{-1}$

$$\frac{2\log(n^2)}{1/4\,\Delta_i^2} = \frac{8\log n^2}{\Delta_i^2} = \frac{\log n}{\Delta_i^2}$$

Simpler Optimism

- UCB, an optimism under uncertainty approach has sublinear regret bounds
- Do we need to formally model uncertainty to get the right form of optimism?

Optimistic Initialization with Greedy Bandit Algorithms

- Simple and practical idea: initialize $\hat{Q}(s, a)$ to high value
- Update action value by incremental Monte-Carlo evaluation
- Starting with $N(a) > 0$

$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)}(r_t - \hat{Q}_{t-1})$$

Optimistic Initialization with Greedy Bandit Algorithms

- Simple idea: initialize $\hat{Q}(s, a)$ to high value
- Update action value by incremental Monte-Carlo evaluation
- Starting with $N(a) > 0$

$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)}(r_t - \hat{Q}_{t-1})$$

- Encourages systematic exploration early on
- But can still lock onto suboptimal action
- Depends on how high initialize Q
- Optional check your understanding: What is the downside to initializing Q too high?
- Optional check your understanding: Is this trivial to do with function approximation? Why or why not?

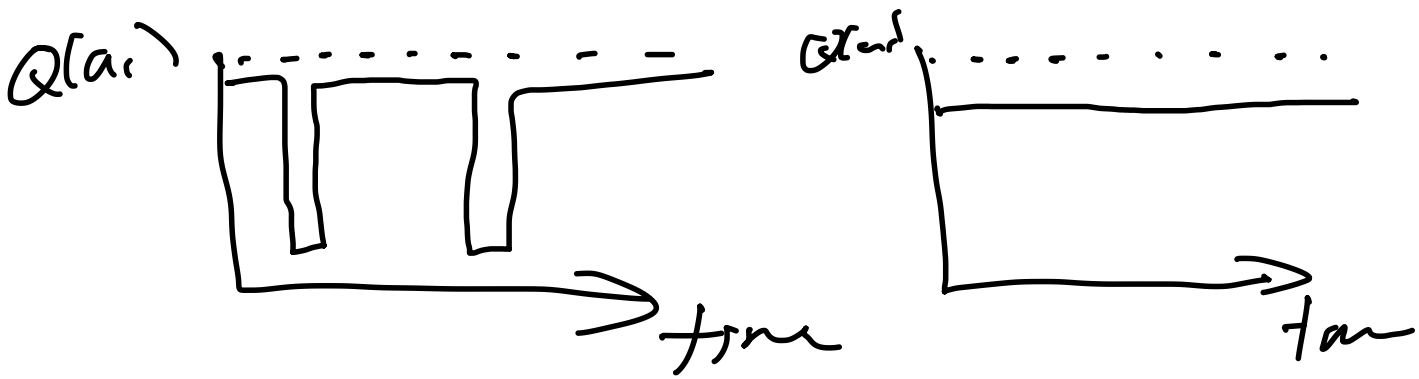
Optimistic Initialization with Greedy Bandit Algorithms

- Simple idea: initialize Q(a) to high value
- Update action value by incremental Monte-Carlo evaluation
- Starting with $N(a) > 0$

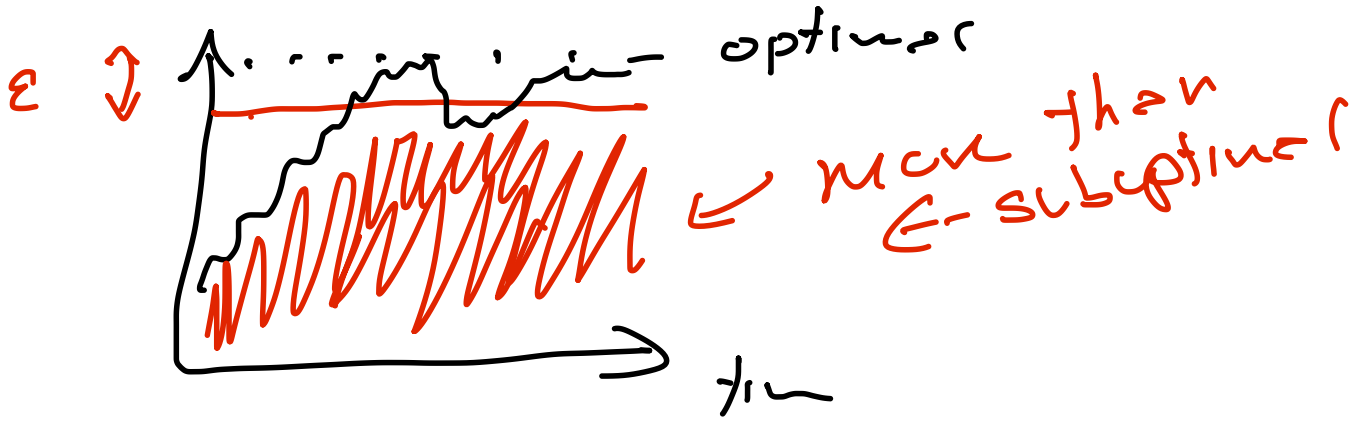
$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)}(r_t - \hat{Q}_{t-1})$$

- Will turn out that if carefully choose the initialization value, can get good performance
- Under a new measure for evaluating algorithms

Framework: Regret



- Theoretical regret bounds specify how regret grows with T
- Could be making lots of little mistakes or infrequent large ones
- May care about bounding the number of non-small errors



Framework: Probably Approximately Correct

- Theoretical regret bounds specify how regret grows with T
- Could be making lots of little mistakes or infrequent large ones
- May care about bounding the number of non-small errors
- More formally, probably approximately correct (PAC) algorithms
  - on each time step, choose an action a
  - whose value is ϵ-optimal: $Q(a) \geq Q(a^*) - \epsilon$
  - with probability at least $1 - \delta$
  - on all but a polynomial number of time steps
- Polynomial in the problem parameters (#actions, ϵ, δ, etc)

Probably Approximately Correct Algorithms

- Theoretical regret bounds specify how regret grows with T
- Could be making lots of little mistakes or infrequent large ones
- May care about bounding the number of non-small errors
- More formally, probably approximately correct (PAC) algorithms
  - on each time step, choose an action a
  - whose value is ϵ-optimal: $Q(a) \geq Q(a^*) - \epsilon$
  - with probability at least $1 - \delta$
  - on all but a polynomial number of time steps
- Polynomial in the problem parameters (#actions, ϵ, δ, etc)
- Most PAC algorithms based on optimism or Thompson sampling
- Some PAC algorithms using optimism simply initialize all values to a (specific to the problem) high value

Toy Example: Probably Approximately Correct and Regret

- Surgery: $\phi_1 = .95$ / Taping: $\phi_2 = .9$ / Nothing: $\phi_3 = .1$
- Let $\epsilon = 0.05$
- O = Optimism, TS = Thompson Sampling: W/in $\epsilon = \mathbb{I}(Q(a_t) \geq Q(a^*) - \epsilon)$

O	Optimal	O Regret	O W/in ϵ
a^1	a^1	0	
a^2	a^1	0.05	
a^3	a^1	0.85	
a^1	a^1	0	
a^2	a^1	0.05	

PAC

PAC

only weak

Greedy Bandit Algorithms vs Optimistic Initialization

- **Greedy**: Linear total regret
- **Constant ϵ-greedy**: Linear total regret
- **Decaying ϵ-greedy**: Sublinear regret but schedule for decaying ϵ requires knowledge of gaps, which are unknown
- **Optimistic initialization**: Sublinear regret if initialize values sufficiently optimistically, else linear regret

Today

- Bandits and Probably Approximately Correct
- **Bayesian Bandits**
- Thompson Sampling
- Bayesian Regret