Lecture 12: Fast Reinforcement Learning

Emma Brunskill

CS234 Reinforcement Learning

Spring 2024

With some slides from or derived from David Silver, Examples new
Select all that are true:

1. Algorithms that minimize regret also maximize reward
2. Up to variations in constants, ignoring $\delta$, UCB selects the arm with $\arg\max_a \hat{Q}_t(a) + \sqrt{\frac{1}{N_t(a)} \log(f(\delta))}$
3. Over an infinite trajectory, UCB will sample all arms an infinite number of times
4. UCB still would likely learn to pull the optimal arm more than other arms if we instead used $\arg\max_a \hat{Q}_t(a) + \sqrt{\frac{1}{\sqrt{N_t(a)}} \log(t/\delta)}$
5. 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.
6. A $k$-armed multi-armed bandit is like a single state MDP with $k$ actions
7. Not Sure
Select all that are true:

1. Algorithms that minimize regret also maximize reward \( \checkmark \)
2. Up to variations in constants, ignoring \( \delta \), UCB selects the arm with
   \[ \arg \max_a \hat{Q}_t(a) + \sqrt{\frac{1}{N_t(a)} \log(f(/\delta))} \]
3. Over an infinite trajectory, UCB will sample all arms an infinite number of times \( \checkmark \)
4. UCB still would likely learn to pull the optimal arm more than other
   arms if we instead used \( \arg \max_a \hat{Q}_t(a) + \sqrt{\frac{1}{\sqrt{N_t(a)}} \log(t/\delta)} \)
5. 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. \( \xmark \)
6. A \( k \)-armed multi-armed bandit is like a single state MDP with \( k \) actions \( \checkmark \)
7. Not Sure

Solutions: (1) True. (2) False (\( \log t \) is missing) (3) True (4) True (5) True (6) True
Where We are

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

Eva uses prior testing results to:
- optimize testing allocation
- produce risk estimates for grey-listing

Pseudonymized, aggregated
Central database

Laboratory logs results in 24–48 h

PLF form
Submitted by visitors 24 h before entry

Test

Eva

No test

Exit

Passenger tested at port of entry. Sample sent to laboratory

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

- 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 \(\sum_{t=1}^t r_t\)
- **Regret** is the opportunity loss for one step

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

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

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

- Maximize cumulative reward \(\iff\) minimize total regret
Last time saw UCB, an optimism under uncertainty approach, which has sublinear regret bounds.

Do we need to formally model uncertainty to get the right form of optimism?
Simple and practical idea: initialize $\hat{Q}(s, a)$ to high value
Update action value by incremental Monte-Carlo evaluation
Starting with $N(a) > 0$

$$\hat{Q}_t(a_t) = \hat{Q}_{t-1} + \frac{1}{N_t(a_t)} (r_t - \hat{Q}_{t-1})$$
Simple and practical idea: initialize $\hat{Q}(s, a)$ to high value
Update action value by incremental Monte-Carlo evaluation
Starting with $N(a) > 0$

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

Encourages systematic exploration early on
But can still lock onto suboptimal action
Depends on how high initialize $Q$
Check your understanding: What is the downside to initializing $Q$ too high?
Check your understanding: Is this trivial to do with function approximation? Why or why not?
Optimistic Initialization with Greedy Bandit Algorithms

- Simple and practical 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
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
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 $\epsilon$-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, $\epsilon$, $\delta$, etc)
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 $\epsilon$-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, $\epsilon$, $\delta$, 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: \( W/in \)

\[ \epsilon = \mathbb{I}(Q(a_t) \geq Q(a^*) - \epsilon) \]

<table>
<thead>
<tr>
<th>O</th>
<th>Optimal</th>
<th>O Regret</th>
<th>O W/in $\epsilon$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a^1$</td>
<td>$a^1$</td>
<td>0</td>
<td>\text{E optimal}</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$a^1$</td>
<td>0.05</td>
<td>\text{E optimal}</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$a^1$</td>
<td>0.85</td>
<td>\text{E optimal}</td>
</tr>
<tr>
<td>$a^1$</td>
<td>$a^1$</td>
<td>0</td>
<td>\text{E optimal}</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$a^1$</td>
<td>0.05</td>
<td>\text{E optimal}</td>
</tr>
</tbody>
</table>
Greedy Bandit Algorithms vs Optimistic Initialization

- **Greedy**: Linear total regret
- **Constant $\epsilon$-greedy**: Linear total regret
- **Decaying $\epsilon$-greedy**: Sublinear regret but schedule for decaying $\epsilon$ 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
Bayesian Bandits

- So far we have made no assumptions about the reward distribution $\mathcal{R}$
  - Except bounds on rewards
- **Bayesian bandits** exploit prior knowledge of rewards, $p[\mathcal{R}]$
In Bayesian view, we start with a prior over the unknown parameters:
- Here the unknown distribution over the rewards for each arm.

Given observations / data about that parameter, update our uncertainty over the unknown parameters using Bayes Rule.
In Bayesian view, we start with a prior over the unknown parameters.
- Here the unknown distribution over the rewards for each arm.

Given observations/data about that parameter, update our uncertainty over the unknown parameters using Bayes Rule.

For example, let the reward of arm $i$ be a probability distribution that depends on parameter $\phi_i$.

- Initial prior over $\phi_i$ is $p(\phi_i)$.
- Pull arm $i$ and observe reward $r_{i1}$.
- Use Bays rule to update estimate over $\phi_i$: \[ \hat{\phi}_i \]
In Bayesian view, we start with a prior over the unknown parameters. Here the unknown distribution over the rewards for each arm.

Given observations/data about that parameter, update our uncertainty over the unknown parameters using Bayes Rule.

For example, let the reward of arm $i$ be a probability distribution that depends on parameter $\phi_i$.

Initial prior over $\phi_i$ is $p(\phi_i)$.

Pull arm $i$ and observe reward $r_{i1}$.

Use Bayes rule to update estimate over $\phi_i$:

$$p(\phi_i|r_{i1}) = \frac{p(r_{i1}|\phi_i)p(\phi_i)}{p(r_{i1})} = \frac{p(r_{i1}|\phi_i)p(\phi_i)}{\int_{\phi_i} p(r_{i1}|\phi_i)p(\phi_i)d\phi_i}$$
In Bayesian view, we start with a prior over the unknown parameters.

Give observations/data about that parameter, update our uncertainty over the unknown parameters using Bayes Rule:

$$p(\phi_i | r_{i1}) = \frac{p(r_{i1} | \phi_i)p(\phi_i)}{\int_{\phi_i} p(r_{i1} | \phi_i)p(\phi_i) d\phi_i}$$

In general computing this update may be tricky to do exactly with no additional structure on the form of the prior and data likelihood.
In Bayesian view, we start with a prior over the unknown parameters.

Give observations / data about that parameter, update our uncertainty over the unknown parameters using Bayes Rule:

$$p(\phi_i | r_{i1}) = \frac{p(r_{i1} | \phi_i)p(\phi_i)}{\int_{\phi_i} p(r_{i1} | \phi_i)p(\phi_i) d\phi_i}$$

In general computing this update may be tricky.

But sometimes can be done analytically.

If the parametric representation of the prior and posterior is the same, the prior and model are called **conjugate**.

For example, exponential families have conjugate priors.
Consider a bandit problem where the reward of an arm is a binary outcome 0, 1, sampled from a Bernoulli with parameter $\theta$

- E.g. Advertisement click through rate, patient treatment success/fails, ...

The Beta distribution $\text{Beta}(\alpha, \beta)$ is conjugate for the Bernoulli distribution

$$p(\theta|\alpha, \beta) = \theta^{\alpha-1}(1 - \theta)^{\beta-1} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}$$

where $\Gamma(x)$ is the Gamma family
Consider a bandit problem where the reward of an arm is a binary outcome 0, 1, sampled from a Bernoulli with parameter $\theta$
  
  E.g. Advertisement click through rate, patient treatment success/fails, ...

The Beta distribution $Beta(\alpha, \beta)$ is conjugate for the Bernoulli distribution

$$p(\theta|\alpha, \beta) = \theta^{\alpha-1}(1-\theta)^{\beta-1}\frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}$$

where $\Gamma(x)$ is the Gamma family

Assume the prior over $\theta$ is $Beta(\alpha, \beta)$ as above

Then after observed a reward $r \in \{0, 1\}$ then updated posterior over $\theta$ is $Beta(r + \alpha, 1 - r + \beta)$
- Maintain distribution over reward parameters
- Use this to inform action selection
Bayesian Bandits Overview

- So far we have made no assumptions about the reward distribution $\mathcal{R}$
  - Except bounds on rewards
- **Bayesian bandits** exploit prior knowledge of rewards, $p[\mathcal{R}]$
- They compute posterior distribution of rewards $p[\mathcal{R} | h_t]$, where $h_t = (a_1, r_1, \ldots, a_{t-1}, r_{t-1})$
- Use posterior to guide exploration
  - Upper confidence bounds (Bayesian UCB)
  - Probability matching (Thompson Sampling)
- Better performance if prior knowledge is accurate
Bandits and Probably Approximately Correct
Bayesian Bandits
Thompson Sampling
Bayesian Regret
Assume have a parametric distribution over rewards for each arm

**Probability matching** selects action $a$ according to probability that $a$ is the optimal action

$$
\pi(a \mid h_t) = \mathbb{P}[Q(a) > Q(a'), \forall a' \neq a \mid h_t]
$$

Probability matching is often optimistic in the face of uncertainty

- Uncertain actions have higher probability of being max

Can be difficult to compute probability that an action is optimal analytically from posterior

Somewhat incredibly, a simple approach implements probability matching
Thompson Sampling

1: Initialize prior over each arm $a$, $p(R_a)$
2: for iteration = 1, 2, ... do
3: For each arm $a$ sample a reward distribution $R_a$ from posterior
4: Compute action-value function $Q(a) = \mathbb{E}[R_a]$
5: $a_t = \arg \max_{a \in A} Q(a)$
6: Observe reward $r$
7: Update posterior $p(R_a)$ using Bayes Rule
8: end for
Thompson sampling implements probability matching

- Thompson sampling:

\[ \pi(a \mid h_t) = \mathbb{P}[Q(a) > Q(a'), \forall a' \neq a \mid h_t] \]
\[ = \mathbb{E}_{R|h_t} \left[ 1(a = \arg\max_{a \in \mathcal{A}} Q(a)) \right] \]
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
- Surgery: $\theta_1 = 0.95$ / Taping: $\theta_2 = 0.9$ / Nothing: $\theta_3 = 0.1$
- Thompson sampling:
- Place a prior over each arm’s parameter. Here choose Beta(1,1) (Uniform)
  - Sample a Bernoulli parameter given current prior over each arm Beta(1,1), Beta(1,1), Beta(1,1):
Toy Example: Ways to Treat Broken Toes, Thompson Sampling\(^1\)

- True (unknown) Bernoulli parameters for each arm/action
  - Surgery: \( \theta_1 = 0.95 \) / Taping: \( \theta_2 = 0.9 \) / Nothing: \( \theta_3 = 0.1 \)
  - Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose Beta(1,1)
    1. Sample a Bernoulli parameter given current prior over each arm
       - Beta(1,1), Beta(1,1), Beta(1,1): 0.3 0.5 0.6
    2. Select \( a = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = \)
        Do nothing \( a_3 \)

---

\(^1\)Note: This is a made up example. This is not the actual expected efficacies of the various treatment options for a broken toe
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
- Surgery: $\theta_1 = 0.95$ / Taping: $\theta_2 = 0.9$ / Nothing: $\theta_3 = 0.1$
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose $\theta_i \sim \text{Beta}(1,1)$
  1. Per arm, sample a Bernoulli $\theta$ given prior: 0.3 0.5 0.6
  2. Select $a_t = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = 3$
  3. Observe the patient outcome's outcome: 0
  4. Update the posterior over the $Q(a_t) = Q(a^3)$ value for the arm pulled
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
  - Surgery: \( \theta_1 = 0.95 \) / Taping: \( \theta_2 = 0.9 \) / Nothing: \( \theta_3 = 0.1 \)
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose \( \theta_i \sim \text{Beta}(1,1) \)
  1. Sample a Bernoulli parameter given current prior over each arm
     \( \text{Beta}(1,1), \text{Beta}(1,1), \text{Beta}(1,1) : 0.3 \ 0.5 \ 0.6 \)
  2. Select \( a_t = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = 3 \)
  3. Observe the patient outcome’s outcome: 0
  4. Update the posterior over the \( Q(a_t) = Q(a^1) \) value for the arm pulled
     - \( \text{Beta}(c_1, c_2) \) is the conjugate distribution for Bernoulli
     - If observe 1, \( c_1 + 1 \) else if observe 0 \( c_2 + 1 \)
  5. New posterior over Q value for arm pulled is:
  6. New posterior \( p(Q(a^3)) = p(\theta(a_3) = \text{Beta}(1,2)) \)
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
- Surgery: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose $\theta_i \sim \text{Beta}(1,1)$
  1. Sample a Bernoulli parameter given current prior over each arm
     $\text{Beta}(1,1), \text{Beta}(1,1), \text{Beta}(1,1)$: 0.3 0.5 0.6
  2. Select $a_t = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = 1$
  3. Observe the patient outcome’s outcome: 0
  4. New posterior $p(Q(a^1)) = p(\theta(a_1) = \text{Beta}(1,2)$
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
- Surgery: $\theta_1 = 0.95$ / Taping: $\theta_2 = 0.9$ / Nothing: $\theta_3 = 0.1$
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose $\theta_i \sim \text{Beta}(1,1)$
  - Sample a Bernoulli parameter given current prior over each arm
    - Beta(1,1), Beta(1,1), Beta(1,2): 0.7, 0.5, 0.3
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
- Surgery: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose $\theta_i \sim \text{Beta}(1,1)$
  1. Sample a Bernoulli parameter given current prior over each arm
     Beta(1,1), Beta(1,1), Beta(1,2): 0.7, 0.5, 0.3
  2. Select $a_t = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = 1$
  3. Observe the patient outcome’s outcome: 1
  4. New posterior $p(Q(a^1)) = p(\theta(a_1)) = \text{Beta}(2,1)$
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
  - Surgery: \( \theta_1 = 0.95 \) / Taping: \( \theta_2 = 0.9 \) / Nothing: \( \theta_3 = 0.1 \)
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose \( \theta_i \sim \text{Beta}(1,1) \)
    1. Sample a Bernoulli parameter given current prior over each arm
       Beta(2,1), Beta(1,1), Beta(1,2): 0.71, 0.65, 0.1
    2. Select \( a_t = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = 1 \)
    3. Observe the patient outcome’s outcome: 1
    4. New posterior \( p(Q(a^1)) = p(\theta(a_1) = \text{Beta}(3,1)) \)
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- True (unknown) Bernoulli parameters for each arm/action
  - Surgery: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Thompson sampling:
  - Place a prior over each arm’s parameter. Here choose $\theta_i \sim \text{Beta}(1,1)$
    - 1. Sample a Bernoulli parameter given current prior over each arm
        Beta(2,1), Beta(1,1), Beta(1,2): 0.75, 0.45, 0.4
    - 2. Select $a_t = \arg \max_{a \in A} Q(a) = \arg \max_{a \in A} \theta(a) = 1$
    - 3. Observe the patient outcome’s outcome: 1
    - 4. New posterior $p(Q(a^1)) = p(\theta(a_1) = \text{Beta}(4,1)$

![Graph showing Beta distribution]
Toy Example: Ways to Treat Broken Toes, Thompson Sampling vs Optimism

- Surgery: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- How does the sequence of arm pulls compare in this example so far?

<table>
<thead>
<tr>
<th>Optimism</th>
<th>TS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a^1$</td>
<td>$a^3$</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$a^1$</td>
</tr>
<tr>
<td>$a^3$</td>
<td>$a^1$</td>
</tr>
<tr>
<td>$a^1$</td>
<td>$a^1$</td>
</tr>
<tr>
<td>$a^2$</td>
<td>$a^1$</td>
</tr>
</tbody>
</table>
Now we will see how Thompson sampling works in general, and what it is doing.
Today

- Bandits and Probably Approximately Correct
- Bayesian Bandits
- Thompson Sampling
- Bayesian Regret
Framework: Regret and Bayesian Regret

- How do we evaluate performance in the Bayesian setting?
- Frequentist regret assumes a true (unknown) set of parameters

\[
\text{Regret}(\mathcal{A}, T; \theta) = \mathbb{E}_\tau \left[ \sum_{t=1}^{T} Q(a^*) - Q(a_t) \mid \theta \right]
\]

where \( \mathbb{E}_\tau \) denotes an expectation with respect to the history of actions taken and rewards observed given an algorithm \( \mathcal{A} \).

- Bayesian regret assumes there is a prior over parameters

\[
\text{BayesRegret}(\mathcal{A}, T; \theta) = \mathbb{E}_{\theta \sim p_{\theta, \tau}} \left[ \sum_{t=1}^{T} Q(a^*) - Q(a_t) \mid \theta \right]
\]
How do we evaluate performance in the Bayesian setting?

Frequentist regret assumes a true (unknown) set of parameters

$$\text{Regret}(A, T; \theta) = \mathbb{E}_\tau \left[ \sum_{t=1}^{T} Q(a^*) - Q(a_t) \middle| \theta \right] \leq \mathbb{E}_\tau \left[ \sum_{t=1}^{T} U_t(a_t) - Q(a_t) \middle| \theta \right]$$

where $\mathbb{E}_\tau$ denotes an expectation with respect to the history of actions taken and rewards observed given an algorithm $A$ (under event that $U_t$ is an upper bound).
Thompson sampling implements probability matching

- Frequentist bounds for standard* Thompson sampling do not* (last checked) match best bounds for frequentist algorithms
- Empirically Thompson sampling can be effective, especially in contextual multi-armed bandits
Thompson Sampling for News Article Recommendation (Chapelle and Li, 2010)

- Contextual bandit: input context which impacts reward of each arm, context sampled iid each step
- Arms = articles
- Reward = click (+1) on article ($Q(a)$=click through rate)
Consider an online news website with thousands of people logging on each second. Frequently a new person will come online before we see whether the last person has clicked (or not). Select all that are true:

1. Thompson sampling would be better than optimism here, because optimism algorithms are deterministic and would select the same action until we get feedback (click or not).
2. Optimism algorithms would be better than TS here, because they have stronger regret bounds for this setting.
3. Thompson sampling could cause much worse performance than optimism if the initial prior is very misleading.
4. Not sure
Check Your Understanding: Thompson Sampling and Optimism Solutions

Consider an online news website with thousands of people logging on each second. Frequently a new person will come online before we see whether the last person has clicked (or not). Select all that are true:

1. Thompson sampling would be better than optimism here, because optimism algorithms are deterministic and would select the same action until we get feedback (click or not).
2. Optimism algorithms would be better than TS here, because they have stronger regret bounds for this setting.
3. Thompson sampling could cause much worse performance than optimism if the initial prior is very misleading.
4. Not sure

Solution: (1) T (2) F (3) T. Consider prior Beta(100,1) for a Bernoulli arm with parameter 0.1. Then the prior puts large weight on high values of theta for a long time.
Optimal Policy for Bayesian Bandits?

- Thompson Sampling often works well, but is it optimal?
- Given prior, and known horizon, could compute decision policy that would maximize expected rewards given the available horizon
- Computational challenge: naively this would create a decision policy that is a function of the history to the next arm to pull
Thompson Sampling often works well, but is it optimal?

Given prior, and known horizon, could compute decision policy that would maximize expected rewards given the available horizon.

Computational challenge: naively this would create a decision policy that is a function of the history to the next arm to pull.

**Index policy**: a decision policy that computes a "real-valued index for each arm and plays the arm with the largest index," using statistics only from that arm and the horizon (definition from Lattimore and Svespari 2019 Bandit Algorithms).

**Gittins index**: optimal policy for maximizing expected discounted reward in a Bayesian multi-armed bandit.
Today

- Bandits and Probably Approximately Correct
- Bayesian Bandits
- Thompson Sampling
- Bayesian Regret
What You Should Understand

- Understand how multi-armed bandits relate to MDPs
- Be able to define regret and PAC
- Be able to prove why UCB bandit algorithm has sublinear regret
- Understand (be able to give an example) why e-greedy and greedy and pessimism can result in linear regret
- Be able to implement Thompson Sampling for Bernoulli or Gaussian rewards
- Be able to implement UCB bandit algorithm
Where We are

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