Lecture 10: Fast Reinforcement Learning [1]

Emma Brunskill

CS234 Reinforcement Learning

[1] With many slides from or derived from David Silver, Examples new

Refresh Your Knowledge. Policy Gradient

- Policy gradient algorithms change the policy parameters using gradient descent on the mean squared Bellman error
  1. True
  2. False.
  3. Not sure
- Select all that are true
  1. In tabular MDPs the number of deterministic policies is smaller than the number of possible value functions
  2. Policy gradient algorithms are very robust to choices of step size
  3. Baselines are functions of state and actions and do not change the bias of the value function
  4. Not sure

Refresh Your Knowledge. Policy Gradient Answers

- Policy gradient algorithms change the policy parameters using gradient descent on the mean squared Bellman error
  1. True
  2. False.
  3. Not sure
- Select all that are true
  1. In tabular MDPs the number of deterministic policies is smaller than the number of possible value functions
  2. Policy gradient algorithms are very robust to choices of step size
  3. Baselines are functions of state and actions and do not change the bias of the value function
  4. Not sure

Solution: They do gradient ascent on the value function. In tabular MDPs the number of deterministic policies is smaller than the number of value functions. Policy gradient algorithms are not very robust to step size choice. Baselines are only functions of state.

Class Structure

- Last time: Policy Gradient
- **This time: Fast Learning**
- Next time: Fast Learning

Up Till Now

- Discussed optimization, generalization, delayed consequences

Teach Computers to Help Us

Computational Efficiency and Sample Efficiency

Computational Efficiency	Sample Efficiency

Algorithms Seen So Far

- How many steps did it take for DQN to learn a good policy for pong?

Evaluation Criteria

- How do we evaluate how "good" an algorithm is?
- If converges?
- If converges to optimal policy?
- How quickly reaches optimal policy?
- Mistakes make along the way?
- Will introduce different measures to evaluate RL algorithms

Settings, Frameworks & Approaches

- Over next couple lectures will consider 2 settings, multiple frameworks, and approaches
- Settings: Bandits (single decisions), MDPs
- Frameworks: evaluation criteria for formally assessing the quality of a RL algorithm
- Approaches: Classes of algorithms for achieving particular evaluation criteria in a certain set
- Note: We will see that some approaches can achieve multiple frameworks in multiple settings

Today

- Setting: Introduction to multi-armed bandits & Approach: greedy methods
- Framework: Regret
- Approach: ϵ-greedy methods
- Approach: Optimism under uncertainty
- Framework: Bayesian regret
- Approach: Probability matching / Thompson sampling

Multiarmed Bandits

- 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 $\sum_{\tau=1}^{t} r_\tau$

Toy Example: Ways to Treat Broken Toes[1]

- Consider deciding how to best treat patients with broken toes
- Imagine have 3 possible options: (1) surgery (2) buddy taping the broken toe with another toe, (3) do nothing
- Outcome measure / reward is binary variable: whether the toe has healed (+1) or not healed (0) after 6 weeks, as assessed by x-ray

[1] Note:This is a made up example. This is not the actual expected efficacies of the various treatment options for a broken toe

Check Your Understanding: Bandit Toes [1]

- Consider deciding how to best treat patients with broken toes
- Imagine have 3 common options: (1) surgery (2) buddy taping the broken toe with another toe (3) doing nothing
- Outcome measure is binary variable: whether the toe has healed (+1) or not (0) after 6 weeks, as assessed by x-ray
- Model as a multi-armed bandit with 3 arms, where each arm is a Bernoulli variable with an unknown parameter θ_i
- Select all that are true
  1. Pulling an arm / taking an action corresponds to whether the toe has healed or not
  2. A multi-armed bandit is a better fit to this problem than a MDP because treating each patient involves multiple decisions
  3. After treating a patient, if $\theta_i \neq 0$ and $\theta_i \neq 1 \ \forall i$ sometimes a patient's toe will heal and sometimes it may not
  4. Not sure

Check Your Understanding: Bandit Toes Solution [1]

- Consider deciding how to best treat patients with broken toes
- Imagine have 3 common options: (1) surgery (2) buddy taping the broken toe with another toe (3) doing nothing
- Outcome measure is binary variable: whether the toe has healed (+1) or not (0) after 6 weeks, as assessed by x-ray
- Model as a multi-armed bandit with 3 arms, where each arm is a Bernoulli variable with an unknown parameter θ_i
- Select all that are true
  1. Pulling an arm / taking an action corresponds to whether the toe has healed or not
  2. A multi-armed bandit is a better fit to this problem than a MDP because treating each patient involves multiple decisions
  3. After treating a patient, if $\theta_i \neq 0$ and $\theta_i \neq 1$ $\forall i$ sometimes a patient's toe will heal and sometimes it may not
  4. Not sure

Solution: 3 is true. Pulling an arm corresponds to treating a patient. A MAB is a better fit than a MDP, because actions correspond to treating a patient, and the treatment of one patient does not

Greedy Algorithm

- We consider algorithms that estimate $\hat{Q}_t(a) \approx Q(a) = \mathbb{E}[R(a)]$
- Estimate the value of each action by Monte-Carlo evaluation

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{t=1}^{T} r_t \mathbb{1}(a_t = a)$$

- The **greedy** algorithm selects the action with highest value

$$a_t^* = \arg\max_{a \in \mathcal{A}} \hat{Q}_t(a)$$

Toy Example: Ways to Treat Broken Toes[1]

- Imagine true (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$

[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, Greedy[1]

- Imagine true (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- Greedy
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get 0, $\hat{Q}(a^1) = 0$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get +1, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$
  2. What is the probability of greedy selecting each arm next? Assume ties are split uniformly.

[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, Greedy[2]

- Imagine true (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- Greedy
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get 0, $\hat{Q}(a^1) = 0$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get +1, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$
  2. Will the greedy algorithm ever find the best arm in this case?

[2]Note:This is a made up example. This is not the actual expected efficacies of the various treatment options for a broken toe

Greedy Algorithm

- We consider algorithms that estimate $\hat{Q}_t(a) \approx Q(a) = \mathbb{E}[R(a)]$
- Estimate the value of each action by Monte-Carlo evaluation

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{t=1}^{T} r_t \mathbb{1}(a_t = a)$$

- The **greedy** algorithm selects the action with highest value

$$a_t^* = \arg\max_{a \in \mathcal{A}} \hat{Q}_t(a)$$

- **Greedy can lock onto suboptimal action, forever**

Today

- Setting: Introduction to multi-armed bandits & Approach: greedy methods
- **Framework: Regret**
- Approach: ϵ-greedy methods
- Approach: Optimism under uncertainty
- Framework: Bayesian regret
- Approach: Probability matching / Thompson sampling

Assessing the Performance of Algorithms

- How do we evaluate the quality of a RL (or bandit) algorithm?
- So far: computational complexity, convergence, convergence to a fixed point, & empirical performance performance
- Today: introduce a formal measure of how well a RL/bandit algorithm will do in any environment, compared to optimal

Regret

- **Action-value** is the mean reward for action a

$$Q(a) = \mathbb{E}[r \mid a]$$

- **Optimal value** V^*

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$$

- **Regret** is the opportunity loss for one step

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

Regret

- **Action-value** is the mean reward for action a

$$Q(a) = \mathbb{E}[r \mid a]$$

- **Optimal value** V^*

$$V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$$

- **Regret** is the opportunity loss for one step

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

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

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

- Maximize cumulative reward $\Longleftrightarrow$ minimize total regret

Evaluating Regret

- **Count** $N_t(a)$ is expected number of selections for action a
- **Gap** Δ_a is the difference in value between action a and optimal action a^*, $\Delta_i = V^* - Q(a_i)$
- Regret is a function of gaps and counts

$$
\begin{aligned}
L_t &= \mathbb{E}\left[\sum_{\tau=1}^{t} V^* - Q(a_\tau)\right] \\
&= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)](V^* - Q(a)) \\
&= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)]\Delta_a
\end{aligned}
$$

- A good algorithm ensures small counts for large gap,s but gaps are not known

Toy Example: Ways to Treat Broken Toes, Optimism, Assessing Regret of Greedy

- True (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- Greedy

Action	Optimal Action	Observed Reward	Regret
a^1	a^1	0	
a^2	a^1	1	
a^3	a^1	0	
a^2	a^1	1	
a^2	a^1	0	

Toy Example: Ways to Treat Broken Toes, Optimism, Assessing Regret of Greedy

- True (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- Greedy

Action	Optimal Action	Observed Reward	Regret
a^1	a^1	0	0
a^2	a^1	1	0.05
a^3	a^1	0	0.85
a^2	a^1	1	0.05
a^2	a^1	0	0.05

- Regret for greedy methods can be **linear** in the number of decisions made (timestep)

Toy Example: Ways to Treat Broken Toes, Optimism, Assessing Regret of Greedy

- Greedy

Action	Optimal Action	Observed Reward	Regret
a^1	a^1	0	0
a^2	a^1	1	0.05
a^3	a^1	0	0.85
a^2	a^1	1	0.05
a^2	a^1	0	0.05

- **Note: in real settings we cannot evaluate the regret because it requires knowledge of the expected reward of the true best action.**
- Instead we can prove an upper bound on the potential regret of an algorithm in **any bandit** problem

Today

- Setting: Introduction to multi-armed bandits & Approach: greedy methods
- Framework: Regret
- **Approach: ϵ-greedy methods**
- Approach: Optimism under uncertainty
- Framework: Bayesian regret
- Approach: Probability matching / Thompson sampling

ϵ-Greedy Algorithm

- The ϵ-**greedy** algorithm proceeds as follows:
  - With probability $1 - \epsilon$ select $a_t = \arg\max_{a \in \mathcal{A}} \hat{Q}_t(a)$
  - With probability ϵ select a random action
- Always will be making a sub-optimal decision ϵ fraction of the time
- Already used this in prior homeworks

Toy Example: Ways to Treat Broken Toes, ϵ-Greedy[1]

- Imagine true (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- ϵ-greedy
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get $+1$, $\hat{Q}(a^1) = 1$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get $+1$, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$
  2. Let $\epsilon = 0.1$
  3. What is the probability ϵ-greedy will pull each arm next? Assume ties are split uniformly.

[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, Optimism, Assessing Regret of Greedy

- True (unknown) Bernoulli reward parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- UCB1 (Auer, Cesa-Bianchi, Fischer 2002)

Action	Optimal Action	Regret
a^1	a^1	
a^2	a^1	
a^3	a^1	
a^1	a^1	
a^2	a^1	

- Will ϵ-greedy ever select a^3 again? If ϵ is fixed, how many times will each arm be selected?

Recall: Bandit Regret

- **Count** $N_t(a)$ is expected number of selections for action a
- **Gap** Δ_a is the difference in value between action a and optimal action a^*, $\Delta_i = V^* - Q(a_i)$
- Regret is a function of gaps and counts

$$\begin{aligned} L_t &= \mathbb{E}\left[\sum_{\tau=1}^{t} V^* - Q(a_\tau)\right] \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)](V^* - Q(a)) \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)]\Delta_a \end{aligned}$$

- A good algorithm ensures small counts for large gap, but gaps are not known

Check Your Understanding: ϵ-greedy Bandit Regret

- **Count** $N_t(a)$ is expected number of selections for action a
- **Gap** Δ_a is the difference in value between action a and optimal action a^*, $\Delta_i = V^* - Q(a_i)$
- Regret is a function of gaps and counts

$$L_t = \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)] \Delta_a$$

- Informally an algorithm has linear regret if it takes a non-optimal action a constant fraction of the time
- Select all
  1. $\epsilon = 0.1$ ϵ-greedy can have linear regret
  2. $\epsilon = 0$ ϵ-greedy can have linear regret
  3. Not sure

Check Your Understanding: ϵ-greedy Bandit Regret Answer

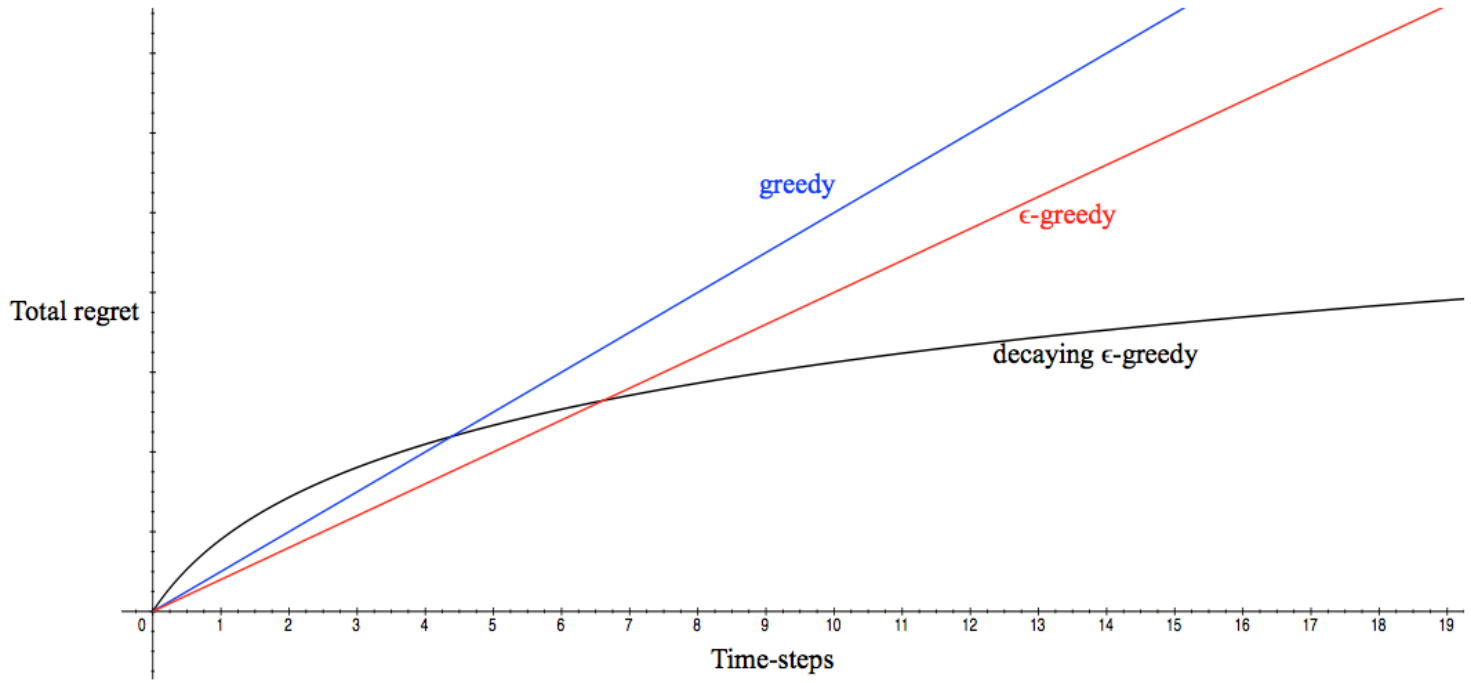
- **Count** $N_t(a)$ is expected number of selections for action a
- **Gap** Δ_a is the difference in value between action a and optimal action a^*, $\Delta_i = V^* - Q(a_i)$
- Regret is a function of gaps and counts

$$L_t = \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)] \Delta_a$$

- Informally an algorithm has linear regret if it takes a non-optimal action a constant fraction of the time
- Select all
  1. $\epsilon = 0.1$ ϵ-greedy can have linear regret
  2. $\epsilon = 0$ ϵ-greedy can have linear regret
  3. Not sure

Both can have linear regret.

"Good": Sublinear or below regret



- **Explore forever**: have linear total regret
- **Explore never**: have linear total regret
- Is it possible to achieve sublinear (in the time steps/number of decisions made) regret?

Types of Regret bounds

- **Problem independent**: Bound how regret grows as a function of T, the total number of time steps the algorithm operates for
- **Problem dependent**: Bound regret as a function of the number of times we pull each arm and the gap between the reward for the pulled arm a^*

Lower Bound

- Use lower bound to determine how hard this problem is
- The performance of any algorithm is determined by similarity between optimal arm and other arms
- Hard problems have similar looking arms with different means
- This is described formally by the gap Δ_a and the similarity in distributions $D_{KL}(\mathcal{R}^a \| \mathcal{R}^{a^*})$
- Theorem (Lai and Robbins): Asymptotic total regret is at least logarithmic in number of steps

$$\lim_{t \to \infty} L_t \geq \log t \sum_{a | \Delta_a > 0} \frac{\Delta_a}{D_{KL}(\mathcal{R}^a \| \mathcal{R}^{a^*})}$$

- Promising in that lower bound is sublinear

Today

- Setting: Introduction to multi-armed bandits & Approach: greedy methods
- Framework: Regret
- Approach: ϵ-greedy methods
- **Approach: Optimism under uncertainty**
- Framework: Bayesian regret
- Approach: Probability matching / Thompson sampling

Approach: Optimism in the Face of Uncertainty

- Choose actions that that might have a high value
- Why?
- Two outcomes:

Approach: Optimism in the Face of Uncertainty

- Choose actions that that might have a high value
- Why?
- Two outcomes:
  - Getting high reward: if the arm really has a high mean reward
  - Learn something: if the arm really has a lower mean reward, pulling it will (in expectation) reduce its average reward and the uncertainty over its value

Upper Confidence Bounds

- Estimate an upper confidence $U_t(a)$ for each action value, such that $Q(a) \leq U_t(a)$ with high probability
- This depends on the number of times $N_t(a)$ action a has been selected
- Select action maximizing Upper Confidence Bound (UCB)

$$a_t = \arg\max_{a \in \mathcal{A}} [U_t(a)]$$

Hoeffding's Inequality

- Theorem (Hoeffding's Inequality): Let $X_1, \ldots, X_n$ be i.i.d. random variables in $[0, 1]$, and let $\bar{X}_n = \frac{1}{n} \sum_{\tau=1}^{n} X_\tau$ be the sample mean. Then

$$\mathbb{P}\left[\mathbb{E}\left[X\right] > \bar{X}_n + u\right] \leq \exp(-2nu^2)$$

UCB Bandit Regret

- This leads to the UCB1 algorithm

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

Toy Example: Ways to Treat Broken Toes, Thompson Sampling[1]

- True (unknown) parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- Optimism under uncertainty, UCB1 (Auer, Cesa-Bianchi, Fischer 2002)
  1. Sample each arm once

[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, Optimism[1]

- True (unknown) parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- UCB1 (Auer, Cesa-Bianchi, Fischer 2002)
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get $+1$, $\hat{Q}(a^1) = 1$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get $+1$, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$

[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, Optimism[1]

- True (unknown) parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- UCB1 (Auer, Cesa-Bianchi, Fischer 2002)
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get $+1$, $\hat{Q}(a^1) = 1$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get $+1$, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$
  2. Set $t = 3$, Compute upper confidence bound on each action

$$UCB(a) = \hat{Q}(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$$

[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, Optimism[1]

- True (unknown) parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- UCB1 (Auer, Cesa-Bianchi, Fischer 2002)
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get $+1$, $\hat{Q}(a^1) = 1$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get $+1$, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$
  2. Set $t = 3$, Compute upper confidence bound on each action

$$UCB(a) = \hat{Q}(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$$

  3. $t = 3$, Select action $a_t = \arg\max_a UCB(a)$,
  4. Observe reward 1
  5. Compute upper confidence bound on each action

[1] Note: This is a made up example. This is not the actual expected efficacies of the

Toy Example: Ways to Treat Broken Toes, Optimism[1]

- True (unknown) parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- UCB1 (Auer, Cesa-Bianchi, Fischer 2002)
  1. Sample each arm once
     - Take action a^1 ($r \sim$Bernoulli(0.95)), get $+1$, $\hat{Q}(a^1) = 1$
     - Take action a^2 ($r \sim$Bernoulli(0.90)), get $+1$, $\hat{Q}(a^2) = 1$
     - Take action a^3 ($r \sim$Bernoulli(0.1)), get 0, $\hat{Q}(a^3) = 0$
  2. Set $t = 3$, Compute upper confidence bound on each action

$$UCB(a) = \hat{Q}(a) + \sqrt{\frac{2 \log t}{N_t(a)}}$$

  3. $t = t + 1$, Select action $a_t = \arg\max_a UCB(a)$,
  4. Observe reward 1
  5. Compute upper confidence bound on each action

[1] Note:This is a made up example. This is not the actual expected efficacies of the

Toy Example: Ways to Treat Broken Toes, Optimism, Assessing Regret

- True (unknown) parameters for each arm (action) are
  - surgery: $Q(a^1) = \theta_1 = .95$
  - buddy taping: $Q(a^2) = \theta_2 = .9$
  - doing nothing: $Q(a^3) = \theta_3 = .1$
- UCB1 (Auer, Cesa-Bianchi, Fischer 2002)

Action	Optimal Action	Regret
a^1	a^1	
a^2	a^1	
a^3	a^1	
a^1	a^1	
a^2	a^1	

High Probability Regret Bound for UCB Multi-armed Bandit

- Any sub-optimal arm $a \neq a^*$ is pulled by UCB at most $\mathbb{E}N_T(a) \leq C' \frac{\log T}{\Delta_a^2} + \frac{\pi^2}{3} + 1$.
  So the regret of UCB is bounded by $\sum_a \Delta_a \mathbb{E}N_T(a) \leq \sum_a C' \frac{\log T}{\Delta_a} + |A|(\frac{\pi^2}{3} + 1)$.
  (Arm means $\in [0, 1]$)

$$P\left(|Q(a) - \hat{Q}_t(a)| \geq \sqrt{\frac{Clogt}{N_t(a)}}\right) \leq \frac{\delta}{T} \tag{1}$$

High Probability Regret Bound for UCB Multi-armed Bandit

- Any sub-optimal arm $a \neq a^*$ is pulled by UCB at most $\mathbb{E}N_T(a) \leq C' \frac{\log T}{\Delta_a^2} + \frac{\pi^2}{3} + 1$.

  So the regret of UCB is bounded by $\sum_a \Delta_a \mathbb{E}N_T(a) \leq \sum_a C' \frac{\log T}{\Delta_a} + |A|(\frac{\pi^2}{3} + 1)$.
  (Arm means $\in [0, 1]$)

$$P\left(|Q(a) - \hat{Q}_t(a)| \geq \sqrt{\frac{Clogt}{N_t(a)}}\right) \leq \frac{\delta}{T} \tag{2}$$

$$Q(a) - \sqrt{\frac{Clogt}{N_t(a)}} \leq \hat{Q}_t(a) \leq Q(a) + \sqrt{\frac{Clogt}{N_t(a)}} \tag{3}$$

High Probability Regret Bound for UCB Multi-armed Bandit

- Any sub-optimal arm $a \neq a^*$ is pulled by UCB at most $\mathbb{E}N_T(a) \leq C' \frac{\log T}{\Delta_a^2} + \frac{\pi^2}{3} + 1$.
  So the regret of UCB is bounded by $\sum_a \Delta_a \mathbb{E}N_T(a) \leq \sum_a C' \frac{\log T}{\Delta_a} + |A|(\frac{\pi^2}{3} + 1)$.
  (Arm means $\in [0, 1]$)

$$Q(a) - \sqrt{\frac{Clogt}{N_t(a)}} \leq \hat{Q}_t(a) \leq Q(a) + \sqrt{\frac{Clogt}{N_t(a)}} \tag{4}$$

$$\hat{Q}_t(a) + \sqrt{\frac{Clogt}{N_t(a)}} \geq \hat{Q}_t(a^*) + \sqrt{\frac{Clogt}{N_t(a^*)}} \geq Q(a^*) \tag{5}$$

$$\tag{6}$$

High Probability Regret Bound for UCB Multi-armed Bandit

- Any sub-optimal arm $a \neq a^*$ is pulled by UCB at most $\mathbb{E}N_T(a) \leq C' \frac{\log T}{\Delta_a^2} + \frac{\pi^2}{3} + 1$.
  So the regret of UCB is bounded by $\sum_a \Delta_a \mathbb{E}N_T(a) \leq \sum_a C' \frac{\log T}{\Delta_a} + |A|(\frac{\pi^2}{3} + 1)$.
  (Arm means $\in [0, 1]$)

$$Q(a) - \sqrt{\frac{Clogt}{N_t(a)}} \leq \hat{Q}_t(a) \leq Q(a) + \sqrt{\frac{Clogt}{N_t(a)}} \tag{7}$$

$$\hat{Q}_t(a) + \sqrt{\frac{Clogt}{N_t(a)}} \geq \hat{Q}_t(a^*) + \sqrt{\frac{Clogt}{N_t(a^*)}} \geq Q(a^*) \tag{8}$$

$$Q(a) + 2\sqrt{\frac{Clogt}{N_t(a)}} \geq Q(a^*) \tag{9}$$

$$2\sqrt{\frac{Clogt}{N_t(a)}} \geq Q(a^*) - Q(a) = \Delta_a \tag{10}$$

$$N_t(a) \leq \frac{2C \log t}{\Delta_a^2} \tag{11}$$

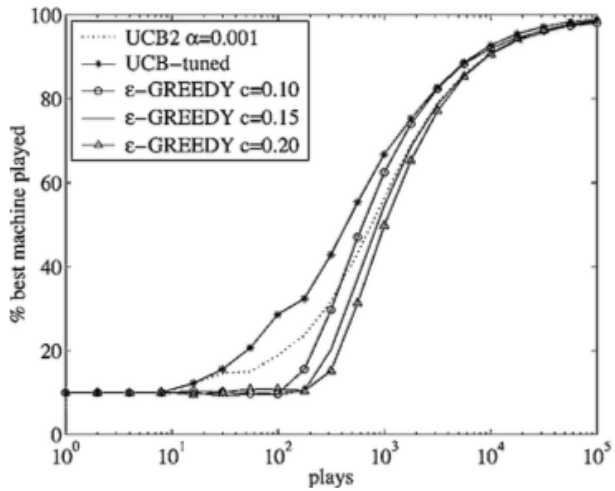
UCB Bandit Regret

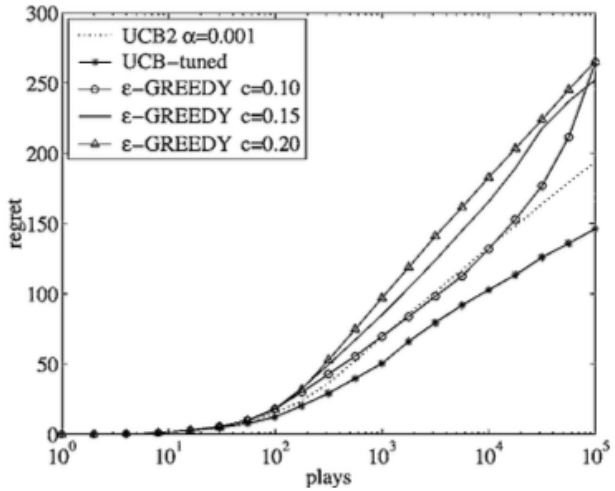
- This leads to the UCB1 algorithm

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

- Theorem: The UCB algorithm achieves logarithmic asymptotic total regret

$$\lim_{t \to \infty} L_t \leq 8 \log t \sum_{a | \Delta_a > 0} \Delta_a$$





Check Your Understanding

- An alternative would be to always select the arm with the highest lower bound
- Why can this yield linear regret?
- Consider a two arm case for simplicity

Today

- Setting: Introduction to multi-armed bandits & Approach: greedy methods
- Framework: Regret
- Approach: ϵ-greedy methods
- Approach: Optimism under uncertainty
- Note: bandits are a simpler place to see these ideas, but these ideas will extend to MDPs
- Next time: more fast learning

Lecture 11: Fast Reinforcement Learning [1]

Emma Brunskill

CS234 Reinforcement Learning

[1] With many slides from or derived from David Silver, Examples new

Refresh Your Understanding: Multi-armed Bandits

- Select all that are true:
  1. Up to slide variations in constants, UCB selects the arm with $\arg\max_a \hat{Q}_t(a) + \sqrt{\frac{1}{N_t(a)} \log(1/\delta)}$
  2. Over an infinite trajectory, UCB will sample all arms an infinite number of times
  3. UCB still would 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. Algorithms that minimize regret also maximize reward
  6. Not Sure

Refresh Your Understanding: Multi-armed Bandits Solution

- Select all that are true:
  1. Up to slide variations in constants, UCB selects the arm with $\arg\max_a \hat{Q}_t(a) + \sqrt{\frac{1}{N_t(a)} \log(1/\delta)}$
  2. Over an infinite trajectory, UCB will sample all arms an infinite number of times
  3. UCB still would 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. Algorithms that minimize regret also maximize reward
  6. Not Sure
- Solutions: (1) False (log t is missing) (2) True (3) True (4) True (5) True

Where We are

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

Recall Motivation

- Fast learning is important when our decisions impact the world

Today

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

Settings, Frameworks & Approaches

- Over next couple lectures will consider 2 settings, multiple frameworks, and approaches
- Settings: Bandits (single decisions), MDPs
- Frameworks: evaluation criteria for formally assessing the quality of a RL algorithm
- Approaches: Classes of algorithms for achieving particular evaluation criteria in a certain set
- Note: We will see that some approaches can achieve multiple frameworks in multiple settings

Multiarmed Bandits 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 $\Longleftrightarrow$ minimize total regret

Simpler Optimism

- 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?

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 $N(a) > 0$

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

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 $N(a) > 0$

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

- 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

Framework: Probably Approximately Correct

- Theoretical regret bounds specify how regret grows with T

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

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) results state that the algorithm will 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 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	TS	Optimal	O Regret	O W/in ϵ	TS Regret	TS W/in ϵ
a^1	a^3	a^1	0		0.85	
a^2	a^1	a^1	0.05		0	
a^3	a^1	a^1	0.85		0	
a^1	a^1	a^1	0		0	
a^2	a^1	a^1	0.05		0	

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

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) results state that the algorithm will 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 steps
- Polynomial in the problem parameters (#actions, ϵ, δ, etc)
- Most PAC algorithms based on optimism or Thompson sampling
- PAC approaches can be relevant to MDPs as well

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
- Check your understanding: why does fixed ϵ-greedy have linear regret? (Encourage you to do a proof sketch)

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, $p[\mathcal{R}]$
- They compute posterior distribution of rewards $p[\mathcal{R} \mid h_t]$, 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

Short Refresher / Review on Bayesian Inference

- 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

Short Refresher / Review on Bayesian Inference

- 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 ϕ_i
- Initial prior over ϕ_i is $p(\phi_i)$
- Pull arm i and observe reward r_{i1}
- Use Bays rule to update estimate over ϕ_i:

Short Refresher / Review on Bayesian Inference

- 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 ϕ_i
- Initial prior over ϕ_i is $p(\phi_i)$
- Pull arm i and observe reward r_{i1}
- Use Bays rule to update estimate over ϕ_i:

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

Short Refresher / Review on Bayesian Inference II

- 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

Short Refresher / Review on Bayesian Inference: Conjugate

- 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 b e 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

Short Refresher / Review on Bayesian Inference: Bernoulli

- Consider a bandit problem where the reward of an arm is a binary outcome 0, 1, sampled from a Bernoulli with parameter θ
  - 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

Short Refresher / Review on Bayesian Inference: Bernoulli

- Consider a bandit problem where the reward of an arm is a binary outcome 0, 1, sampled from a Bernoulli with parameter θ
  - 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 θ is $Beta(\alpha, \beta)$ as above
- Then after observed a reward $r \in \{0, 1\}$ then updated posterior over θ is $Beta(r + \alpha, 1 - r + \beta)$

Bayesian Inference for Decision Making

- Maintain distribution over reward parameters
- Use this to inform action selection

Today

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

Thompson Sampling

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

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 Beta(1,1) (Uniform)
  1. 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: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- 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 $a = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) =$

[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: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Thompson sampling:
- Place a prior over each arm's parameter. Here choose $\theta_i \sim$Beta(1,1)
  1. Per arm, sample a Bernoulli θ given prior: 0.3 0.5 0.6
  2. Select $a_t = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) = 3$
  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: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Thompson sampling:
- Place a prior over each arm's parameter. Here choose $\theta_i \sim$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 $a_t = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) = 3$
  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
     - 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 $p(Q(a^3)) = p(\theta(a_3) = Beta(1, 2)$

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 $\theta_i \sim$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 $a_t = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) = 1$
  3. Observe the patient outcome's outcome: 0
  4. New posterior $p(Q(a^1)) = p(\theta(a_1) = Beta(1, 2)$

Toy Example: Ways to Treat Broken Toes, Thompson Sampling

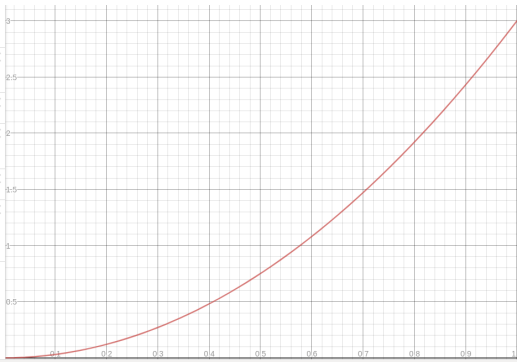
- True (unknown) Bernoulli parameters for each arm/action
- Surgery: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Thompson sampling:
- Place a prior over each arm's parameter. Here choose $\theta_i \sim$Beta(1,1)
  1. Sample a Bernoulli parameter given current prior over each arm
     Beta(1,1), Beta(1,1), Beta(1,2): 0.7, 0.5, 0.3

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 $\theta_i \sim$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 $a_t = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) = 1$
  3. Observe the patient outcome's outcome: 1
  4. New posterior $p(Q(a^1)) = p(\theta(a_1) = Beta(2, 1)$

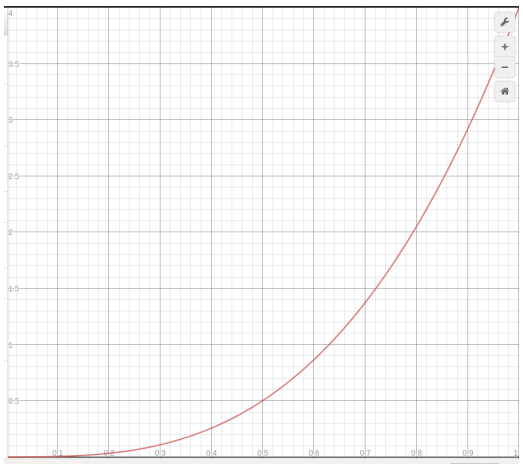
Toy Example: Ways to Treat Broken Toes, Thompson Sampling

- 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 $\theta_i \sim$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 $a_t = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) = 1$
  3. Observe the patient outcome's outcome: 1
  4. New posterior $p(Q(a^1)) = p(\theta(a_1) = Beta(3, 1)$



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 $\theta_i \sim$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 $a_t = \arg\max_{a \in A} Q(a) = \arg\max_{ainA} \theta(a) = 1$
  3. Observe the patient outcome's outcome: 1
  4. New posterior $p(Q(a^1)) = p(\theta(a_1) = Beta(4, 1)$



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?

Optimism	TS	Optimal	Regret Optimism	Regret TS
a^1	a^3			
a^2	a^1			
a^3	a^1			
a^1	a^1			
a^2	a^1			

Toy Example: Ways to Treat Broken Toes, Thompson Sampling vs Optimism

- Surgery: $\theta_1 = .95$ / Taping: $\theta_2 = .9$ / Nothing: $\theta_3 = .1$
- Incurred regret?

Optimism	TS	Optimal	Regret Optimism	Regret TS
a^1	a^3	a^1	0	0
a^2	a^1	a^1	0.05	
a^3	a^1	a^1	0.85	
a^1	a^1	a^1	0	
a^2	a^1	a^1	0.05	

On to General Setting for Thompson Sampling

- 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

Probability Matching

- 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 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, $p(\mathcal{R}_a)$
2: **for** iteration$=1, 2, \ldots$ **do**
3: For each arm a **sample** a reward distribution $\mathcal{R}_a$ from posterior
4: Compute action-value function $Q(a) = \mathbb{E}[\mathcal{R}_a]$
5: $a_t = \arg\max_{a \in \mathcal{A}} Q(a)$
6: Observe reward r
7: Update posterior $p(\mathcal{R}_a|r)$ using Bayes Rule
8: **end for**

Thompson sampling implements probability matching

- Thompson sampling:

$$\begin{aligned}\pi(a \mid h_t) &= \mathbb{P}[Q(a) > Q(a'), \forall a' \neq a \mid h_t] \\ &= \mathbb{E}_{\mathcal{R}|h_t}\left[\mathbb{1}(a = \arg\max_{a \in \mathcal{A}} Q(a))\right]\end{aligned}$$

Framework: Regret and Bayesian Regret

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

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

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

$$BayesRegret(\mathcal{A}, T; \theta) =$$

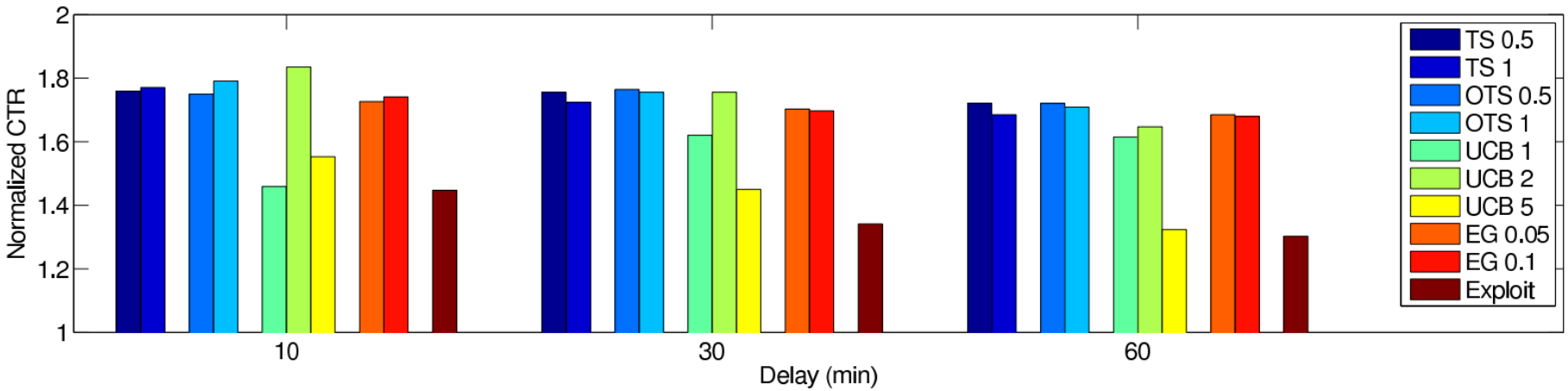
$$\mathbb{E}_{\theta \sim p_\theta, \tau} \left[\sum_{t=1}^{T} Q(a^*) - Q(a_t) | \theta \right] \leq \mathbb{E}_{\theta \sim p_\theta, \tau} \left[\sum_{t=1}^{T} U_t(a_t) - Q(a_t) | \theta \right]$$

Thompson sampling implements probability matching

- Thompson sampling(1929) achieves Lai and Robbins lower bound
- Bounds for optimism are tighter than for Thompson sampling
- But empirically Thompson sampling can be extremely effective

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)



Check Your Understanding: Thompson Sampling and Optimism

- 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
  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
  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.

Today

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

Where We are

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

Bayesian Regret Bounds for Thompson Sampling

- Regret(UCB,T)

$$BayesRegret(TS, T) = E_{\theta \sim p_\theta}\left[\sum_{t=1}^{T} f^*(a^*) - f^*(a_t)\right]$$

- Posterior sampling has the same (ignoring constants) regret bounds as UCB

Lecture 12: Fast RL Part III[1]

Emma Brunskill

CS234 Reinforcement Learning

[1] With a few slides derived from David Silver

Refresh Your Knowledge Fast RL Part II

- The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right figure). Select all that are true.
  1. Sample 3 params: 0.1,0.5,0.3. These are more likely to come from the Beta(1,2) distribution than Beta(1,1).
  2. Sample 3 params: 0.2,0.5,0.8. These are more likely to come from the Beta(1,1) distribution than Beta(1,2).
  3. It is impossible that the true Bernoulli parameter is 0 if the prior is Beta(1,1).
  4. Not sure
- The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right). The true parameters are arm 1 $\theta_1 = 0.4$ & arm 2 $\theta_2 = 0.6$. Thompson sampling = TS
  1. TS could sample $\theta = 0.5$ (arm 1) and $\theta = 0.55$ (arm 2).
  2. For the sampled thetas (0.5,0.55) TS is optimistic with respect to the true arm parameters for all arms.
  3. For the sampled thetas (0.5,0.55) TS will choose the true optimal arm for this round.
  4. Not sure

Refresh Your Knowledge Fast RL Part II Solution

- The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right figure). Select all that are true.
  1. Sample 3 params: 0.1,0.5,0.3. These are more likely to come from the Beta(1,2) distribution than Beta(1,1). (true)
  2. Sample 3 params: 0.2,0.5,0.8. These are more likely to come from the Beta(1,1) distribution than Beta(1,2). (true)
  3. It is impossible that the true Bernoulli parameter is 0 if the prior is Beta(1,1). (false)
  4. Not sure
- The prior over arm 1 is Beta(1,2) (left) and arm 2 is a Beta(1,1) (right). The true parameters are arm 1 $\theta_1 = 0.4$ & arm 2 $\theta_2 = 0.6$. Thompson sampling = TS
  1. TS could sample $\theta = 0.5$ (arm 1) and $\theta = 0.55$ (arm 2). (true)
  2. For the sampled thetas (0.5,0.55) TS is optimistic with respect to the true arm parameters for all arms. (false)
  3. For the sampled thetas (0.5,0.55) TS will choose the true optimal arm for this round. (true)
  4. Not sure

Class Structure

- Last time: Fast Learning (Bayesian bandits to MDPs)
- **This time: Fast Learning III (MDPs)**
- Next time: Batch RL

Settings, Frameworks & Approaches

- Over these 3 lectures will consider 2 settings, multiple frameworks, and approaches
- Settings: Bandits (single decisions), MDPs
- Frameworks: evaluation criteria for formally assessing the quality of a RL algorithm. So far seen empirical evaluations, asymptotic convergence, regret, probably approximately correct
- Approaches: Classes of algorithms for achieving particular evaluation criteria in a certain set. So far for exploration seen: greedy, $\epsilon-$greedy, optimism, Thompson sampling, for multi-armed bandits

Table of Contents

Fast RL in Markov Decision Processes

- Very similar set of frameworks and approaches are relevant for fast learning in reinforcement learning
- Frameworks
  - Regret
  - Bayesian regret
  - Probably approximately correct (PAC)
- Approaches
  - Optimism under uncertainty
  - Probability matching / Thompson sampling
- Framework: Probably approximately correct

Fast RL in Markov Decision Processes

- Montezuma's revenge
- https://www.youtube.com/watch?v=ToSe_CUG0F4

Model-Based Interval Estimation with Exploration Bonus (MBIE-EB)

(Strehl and Littman, J of Computer & Sciences 2008)

Given ϵ, δ, m

$\beta = \frac{1}{1-\gamma}\sqrt{0.5 \ln(2|S||A|m/\delta)}$

$n_{sas}(s,a,s') = 0,\ \forall s \in S,\ a \in A,\ s' \in S$

$rc(s,a) = 0,\ n_{sa}(s,a) = 0,\ \tilde{Q}(s,a) = 1/(1-\gamma),\ \forall\ s \in S,\ a \in A$

$t = 0,\ s_t = s_{init}$

loop

  $a_t = \arg\max_{a \in \mathcal{A}} \tilde{Q}(s_t, a)$

  Observe reward r_t and state s_{t+1}

  $n_{sa}(s_t,a_t) = n_{sa}(s_t,a_t) + 1,\ n_{sas}(s_t,a_t,s_{t+1}) = n_{sas}(s_t,a_t,s_{t+1}) + 1$

  $rc(s_t,a_t) = \frac{rc(s_t,a_t)(n_{sa}(s_t,a_t)-1)+r_t}{n_{sa}(s_t,a_t)}$

  $\hat{R}(s_t,a_t) = rc(s_t,a_t)$ and $\hat{T}(s'|s_t,a_t) = \frac{n_{sas}(s_t,a_t,s')}{n_{sa}(s_t,a_t)},\ \forall s' \in S$

  while not converged **do**

    $\tilde{Q}(s,a) = \hat{R}(s,a) + \gamma \sum_{s'} \hat{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a) + \frac{\beta}{\sqrt{n_{sa}(s,a)}},\ \forall\ s \in S,\ a \in A$

  end while

end loop

Framework: PAC for MDPs

- For a given ϵ and δ, A RL algorithm $\mathcal{A}$ is PAC if on all but N steps, the action selected by algorithm $\mathcal{A}$ on time step t, a_t, is ϵ-close to the optimal action, where N is a polynomial function of $(|S|, |A|, \gamma, \epsilon, \delta)$
- Is this true for all algorithms?

MBIE-EB is a PAC RL Algorithm

Theorem 2. *Suppose that ϵ and δ are two real numbers between 0 and 1 and $M = \langle S, A, T, \mathcal{R}, \gamma \rangle$ is any MDP. There exists an input $m = m(\frac{1}{\epsilon}, \frac{1}{\delta})$, satisfying $m(\frac{1}{\epsilon}, \frac{1}{\delta}) = O(\frac{|S|}{\epsilon^2(1-\gamma)^4} + \frac{1}{\epsilon^2(1-\gamma)^4} \ln \frac{|S||A|}{\epsilon(1-\gamma)\delta})$, and $\beta = (1/(1-\gamma))\sqrt{\ln(2|S||A|m/\delta)/2}$ such that if MBIE-EB is executed on MDP M, then the following holds. Let $\mathcal{A}_t$ denote MBIE-EB's policy at time t and s_t denote the state at time t. With probability at least $1-\delta$, $V_M^{\mathcal{A}_t}(s_t) \geqslant V_M^*(s_t) - \epsilon$ is true for all but $O(\frac{|S||A|}{\epsilon^3(1-\gamma)^6}(|S| + \ln \frac{|S||A|}{\epsilon(1-\gamma)\delta}) \ln \frac{1}{\delta} \ln \frac{1}{\epsilon(1-\gamma)})$ timesteps t.*

A Sufficient Set of Conditions to Make a RL Algorithm PAC

- Strehl, A. L., Li, L., & Littman, M. L. (2006). Incremental model-based learners with formal learning-time guarantees. In Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence (pp. 485-493)

A Sufficient Set of Conditions to Make a RL Algorithm PAC

1. Optimism
2. Accuracy
3. Bounded learning complexity: number of updates of the state-action Q values, and number of times visit a (s,a) pair for which don't have an accurate estimate of its reward and/or dynamics model.

- Note: the above assumed a tabular domain (finite state and action space). But these ideas relate back to the ideas we saw in UCB, and also are relevant later for function approximation.

Table of Contents

Refresher: Bayesian Bandits

- **Bayesian bandits** exploit prior knowledge of rewards, $p[\mathcal{R}]$
- They compute posterior distribution of rewards $p[\mathcal{R} \mid h_t]$, 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

Refresher: Bernoulli Bandits

- Consider a bandit problem where the reward of an arm is a binary outcome $\{0, 1\}$ sampled from a Bernoulli with parameter θ
  - E.g. Advertisement click through rate, patient treatment succeeds/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 function.
- Assume the prior over θ is a $Beta(\alpha, \beta)$ as above
- Then after observed a reward $r \in \{0, 1\}$ then updated posterior over θ is $Beta(r + \alpha, 1 - r + \beta)$

Thompson Sampling for Bandits

1: Initialize prior over each arm a, $p(\mathcal{R}_a)$
2: **loop**
3: For each arm a **sample** a reward distribution $\mathcal{R}_a$ from posterior
4: Compute action-value function $Q(a) = \mathbb{E}[\mathcal{R}_a]$
5: $a_t = \arg\max_{a \in \mathcal{A}} Q(a)$
6: Observe reward r
7: Update posterior $p(\mathcal{R}_a|r)$ using Bayes law
8: **end loop**

Bayesian Model-Based RL

- Maintain posterior distribution over **MDP** models
- Estimate both transition and rewards, $p[\mathcal{P}, \mathcal{R} \mid h_t]$, where $h_t = (s_1, a_1, r_1, \ldots, s_t)$ is the history
- Use posterior to guide exploration
  - Upper confidence bounds (Bayesian UCB)
  - Probability matching (Thompson sampling)

Thompson Sampling: Model-Based RL

- Thompson sampling implements probability matching

$$\pi(s, a \mid h_t) = \mathbb{P}[Q(s, a) \geq Q(s, a'), \forall a' \neq a \mid h_t]$$
$$= \mathbb{E}_{\mathcal{P},\mathcal{R}|h_t}\left[\mathbb{1}(a = \arg\max_{a \in \mathcal{A}} Q(s, a))\right]$$

- Use Bayes law to compute posterior distribution $p[\mathcal{P}, \mathcal{R} \mid h_t]$
- **Sample** an MDP $\mathcal{P}, \mathcal{R}$ from posterior
- Solve MDP using favorite planning algorithm to get $Q^*(s, a)$
- Select optimal action for sample MDP, $a_t = \arg\max_{a \in \mathcal{A}} Q^*(s_t, a)$

Thompson Sampling for MDPs

1: Initialize prior over the dynamics and reward models for each (s, a), $p(\mathcal{R}_{as})$, $p(\mathcal{T}(s'|s, a))$
2: Initialize state s_0
3: **loop**
4: Sample a MDP $\mathcal{M}$: for each (s, a) pair, sample a dynamics model $\mathcal{T}(s'|s, a)$ and reward model $\mathcal{R}(s, a)$
5: Compute $Q^*_{\mathcal{M}}$, optimal value for MDP $\mathcal{M}$
6: $a_t = \arg\max_{a \in \mathcal{A}} Q^*_{\mathcal{M}}(s_t, a)$
7: Observe reward r_t and next state s_{t+1}
8: Update posterior $p(\mathcal{R}_{a_t s_t}|r_t)$, $p(\mathcal{T}(s'|s_t, a_t)|s_{t+1})$ using Bayes rule
9: $t = t + 1$
10: **end loop**

Check Your Understanding: Fast RL III

- Strategic exploration in MDPs (select all):
  1. Doesn't really matter because the distribution of data is independent of the policy followed
  2. Can involve using optimism with respect to both the possible dynamics and reward models in order to compute an optimistic Q function
  3. Is known as PAC if the number of time steps on which a less than near optimal decision is made is guaranteed to be less than an exponential function of the problem domain parameters (state space cardinality, etc).
  4. Not sure
- In Thompson sampling for MDPs:
  1. TS samples the reward model parameters and could use the empirical average for the dynamics model parameters and obtain the same performance
  2. Must perform MDP planning everytime the posterior is updated
  3. Has the same computational cost each step as Q-learning
  4. Not sure

Check Your Understanding: Fast RL III Solutions

- Strategic exploration in MDPs (select all):
  1. Doesn't really matter because the distribution of data is independent of the policy followed (False)
  2. Can involve using optimism with respect to both the possible dynamics and reward models in order to compute an optimistic Q function (True)
  3. Is known as PAC if the number of time steps on which a less than near optimal decision is made is guaranteed to be less than an exponential function of the problem domain parameters (state space cardinality, etc). (false)
  4. Not sure
- In Thompson sampling for MDPs:
  1. TS samples the reward model parameters and could use the empirical average for the dynamics model parameters and obtain the same performance (false)
  2. Must perform MDP planning everytime the posterior is updated (True in shown algorithm, could imagine alternatives)
  3. Has the same computational cost each step as Q-learning (False)
  4. Not sure

Table of Contents

Generalization and Strategic Exploration

- Active area of ongoing research: combine generalization & strategic exploration
- Many approaches are grounded by principles outlined here
  - Optimism under uncertainty
  - Thompson sampling

Generalization and Optimism

- Recall MBIE-EB algorithm for finite state and action domains
- What needs to be modified for continuous / extremely large state and/or action spaces?

Model-Based Interval Estimation with Exploration Bonus (MBIE-EB)

(Strehl and Littman, J of Computer & Sciences 2008)

1: Given ϵ, δ, m
2: $\beta = \frac{1}{1-\gamma}\sqrt{0.5 \ln(2|S||A|m/\delta)}$
3: $n_{sas}(s,a,s') = 0,\ \forall s \in S,\ a \in A,\ s' \in S$
4: $rc(s,a) = 0,\ n_{sa}(s,a) = 0,\ \tilde{Q}(s,a) = 1/(1-\gamma),\ \forall\ s \in S,\ a \in A$
5: $t = 0,\ s_t = s_{init}$
6: **loop**
7: $\quad a_t = \arg\max_{a \in \mathcal{A}} \tilde{Q}(s_t, a)$
8: $\quad$ Observe reward r_t and state s_{t+1}
9: $\quad n_{sa}(s_t,a_t) = n_{sa}(s_t,a_t) + 1,\ n_{sas}(s_t,a_t,s_{t+1}) = n_{sas}(s_t,a_t,s_{t+1}) + 1$
10: $\quad rc(s_t,a_t) = \frac{rc(s_t,a_t)(n_{sa}(s_t,a_t)-1)+r_t}{n_{sa}(s_t,a_t)}$
11: $\quad \hat{R}(s_t,a_t) = rc(s_t,a_t)$ and $\hat{T}(s'|s_t,a_t) = \frac{n_{sas}(s_t,a_t,s')}{n_{sa}(s_t,a_t)},\ \forall s' \in S$
12: $\quad$ **while** not converged **do**
13: $\quad\quad \tilde{Q}(s,a) = \hat{R}(s,a) + \gamma \sum_{s'} \hat{T}(s'|s,a) \max_{a'} \tilde{Q}(s',a) + \frac{\beta}{\sqrt{n_{sa}(s,a)}},\ \forall\ s \in S,\ a \in A$
14: $\quad$ **end while**
15: **end loop**

Generalization and Optimism

- Recall MBIE-EB algorithm for finite state and action domains
- What needs to be modified for continuous / extremely large state and/or action spaces?
- Estimating uncertainty
  - Counts of (s,a) and (s,a,s') tuples are not useful if we expect only to encounter any state once
- Computing a policy
  - Model-based planning will fail
- So far, model-free approaches have generally had more success than model-based approaches for extremely large domains
  - Building good transition models to predict pixels is challenging

Recall: Value Function Approximation with Control

- For Q-learning use a TD target $r + \gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{w})$ which leverages the max of the current function approximation value

$$\Delta \boldsymbol{w} = \alpha(r(s) + \gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{w}) - \hat{Q}(s, a; \boldsymbol{w}))\nabla_{\boldsymbol{w}} \hat{Q}(s, a; \boldsymbol{w})$$

Recall: Value Function Approximation with Control

- For Q-learning use a TD target $r + \gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{w})$ which leverages the max of the current function approximation value

$$\Delta \boldsymbol{w} = \alpha(r(s)+r_{bonus}(s,a)+\gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{w})-\hat{Q}(s, a; \boldsymbol{w}))\nabla_{\boldsymbol{w}} \hat{Q}(s, a; \boldsymbol{w})$$

Recall: Value Function Approximation with Control

- For Q-learning use a TD target $r + \gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{w})$ which leverages the max of the current function approximation value

$$\Delta \boldsymbol{w} = \alpha(r(s) + r_{bonus}(s, a) + \gamma \max_{a'} \hat{Q}(s', a'; \boldsymbol{w}) - \hat{Q}(s, a; \boldsymbol{w}))\nabla_{\boldsymbol{w}} \hat{Q}(s, a; \boldsymbol{w})$$

- $r_{bonus}(s, a)$ should reflect uncertainty about future reward from (s, a)
- Approaches for deep RL that make an estimate of visits / density of visits include: Bellemare et al. NIPS 2016; Ostrovski et al. ICML 2017; Tang et al. NIPS 2017
- Note: bonus terms are computed at time of visit. During episodic replay can become outdated.

Benefits of Strategic Exploration: Montezuma's revenge

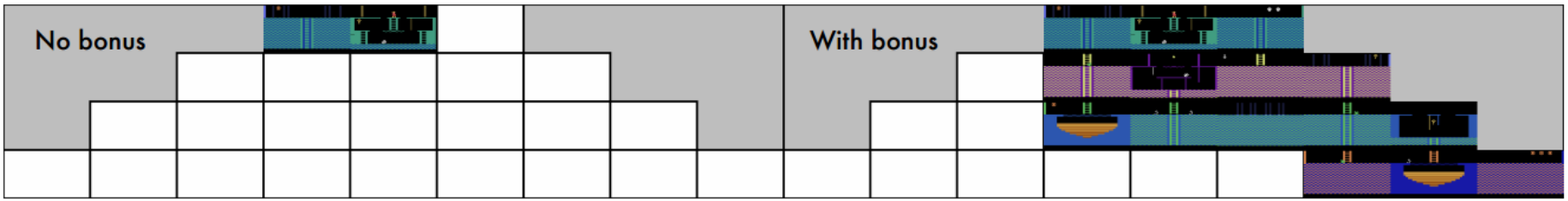


Figure 3: "Known world" of a DQN agent trained for 50 million frames with (**right**) and without (**left**) count-based exploration bonuses, in MONTEZUMA'S REVENGE.

Figure: Bellemare et al. "Unifying Count-Based Exploration and Intrinsic Motivation"

- Enormously better than standard DQN with ϵ-greedy approach

Generalization and Strategic Exploration: Thompson Sampling

- Leveraging Bayesian perspective has also inspired some approaches
- One approach: Thompson sampling over representation & parameters (Mandel, Liu, Brunskill, Popovic IJCAI 2016)

Generalization and Strategic Exploration: Thompson Sampling

- For scaling up to very large domains, again useful to consider model-free approaches
- Non-trivial: would like to be able to sample from a posterior over possible Q^*
- Bootstrapped DQN (Osband et al. NIPS 2016)
  - Train C DQN agents using bootstrapped samples
  - When acting, choose action with highest Q value over any of the C agents
  - Some performance gain, not as effective as reward bonus approaches

Generalization and Strategic Exploration: Thompson Sampling

- Leveraging Bayesian perspective has also inspired some approaches
- One approach: Thompson sampling over representation & parameters (Mandel, Liu, Brunskill, Popovic IJCAI 2016)
- For scaling up to very large domains, again useful to consider model-free approaches
- Non-trivial: would like to be able to sample from a posterior over possible Q^*
- Bootstrapped DQN (Osband et al. NIPS 2016)
- Efficient Exploration through Bayesian Deep Q-Networks (Azizzadenesheli, Anandkumar, NeurIPS workshop 2017)
  - Use deep neural network
  - On last layer use Bayesian linear regression
  - Be optimistic with respect to the resulting posterior
  - Very simple, empirically much better than just doing linear regression on last layer or bootstrapped DQN, not as good as reward bonuses in some cases

Theoretical Results

- We discussed regret bounds for bandits, & PAC bounds for tabular MDPs

Theoretical Results

- We discussed regret bounds for bandits, & PAC bounds for tabular MDPs
- Now exist tight (in dominant term) minimax results for regret and PAC for tabular MDPs
  - Azar, Mohammad Gheshlaghi, Ian Osband, and Rémi Munos. Minimax regret bounds for reinforcement learning. ICML 2017 (regret)
  - Dann, C., Li, L., Wei, W., and Brunskill, E. Policy certificates: Towards accountable reinforcement learning. ICML 2019 (PAC)
- Also exist instance-dependence bounds for tabular MDPs. For example:
  - Zanette (your CA) and Brunskill. Tighter problem-dependent regret bounds in reinforcement learning without domain knowledge using value function bounds. ICML 2019
  - Simchowitz, Max, and Kevin Jamieson. Non-asymptotic gap-dependent regret bounds for tabular MDPs. NeurIPS 2019.

Theoretical Results: Function Approximation & RL

- Do there exist strong theoretical bounds for RL with function approximation?
- Active area of recent work
  - Jin, Yang, Wang, and Jordan. "Provably efficient reinforcement learning with linear function approximation." COLT 2020.
  - Many others, including our work (lead by Andrea Zanette), and Mengdi Wang's lab.

Table of Contents

Summary: What You Are Expected to Know

- Define the tension of exploration and exploitation in RL and why this does not arise in supervised or unsupervised learning
- Be able to define and compare different criteria for "good" performance (empirical, convergence, asymptotic, regret, PAC)
- Be able to map algorithms discussed in detail in class to the performance criteria they satisfy
- Understand the UCB proof sketch

Class Structure

- Last time: Fast Learning (Bayesian bandits to MDPs)
- **This time: Fast Learning III (MDPs)**
- Next time: Batch RL

Resampling in Coordinated Exploration

- Concurrent PAC RL. Guo and Brunskill. AAAI 2015
- Coordinated Exploration in Concurrent Reinforcement Learning. Dimakopoulou and Van Roy. ICML 2018
- https://www.youtube.com/watch?v=xjGK-wm0PkI&feature=youtu.be