MS&E338 Reinforcement Learning Lecture 12 - May 9, 2018

Bandit Learning and Heuristic Exploration strategies

Lecturer: Ben Van Roy *Scribe: Arjun Parthipan, Anand Venkatesan*

1 Recap

Last lecture we were focusing on Multi Armed Bandit Problem where we considered Beta Bernoulli Bandits. We considered tossing k coins with probability of heads as $p_1, p_2, ..., p_k$ where we get a reward of \$1 for heads and \$0 for tails. The reward is given by,

$$r_t = \begin{cases} 1 & \text{with probability } p_{x_t} \\ 0 & \text{with probability } 1 - p_{x_t} \end{cases}$$

We try to maximize the expected reward i.e, $max\Big[E\sum_{t=1}^{T} r_t\Big]$ where T is a very long time horizon. We discussed about the greedy approach in the last lecture and we observed that this method gets into a mode where it never selects one choice. We also talked about epsilon-greedy and boltzmann exploration methods and their limitations.

2 Gittin's indices

We assume discounted rewards for this section. The Gittins index is a measure of the reward that can be achieved by a random process bearing a termination state. Considering the bandit problem, given the belief about each coin, Gittin's index is an index or a number based on information of each coin. It returns a scalar value and the coin which results in highest Gittin's index may be chosen.

Consider the coin can be sold at any instant of time. Let there be some bias in the coin and consider that the distribution of the coin is not known. Let \$X be the price of the coin when it is sold. There comes an inherent question as to when to sell the coin. Intuitively, if X is large, the action is "sell now". If X is small, the action is "never sell". But if X takes values in between, the action will be "sell in future depending on the coin toss". This optimal stopping problem can be solved by value iteration.

$$V(s) = max(x, E[r + \gamma V(s')])$$

where s is the state.

$$V_x(\alpha_k, \beta_k) = max\Big(x, \frac{\alpha_k}{\alpha_k + \beta_k} + \gamma\Big(\frac{\alpha_k}{\alpha_k + \beta_k} V_x(\alpha_{k+1}, \beta_k) + \frac{\beta_k}{\alpha_k + \beta_k} V_x(\alpha_k, \beta_{k+1})\Big)\Big)$$

The Gittin's index X* is given by:

$$X^*(\alpha, \beta) = min\{X : V_X(\alpha, \beta) = X\}$$

2.1 Gittin's index theorem

Gittin's index theorem states that the following algorithm is optimal:

Select the coin k that maximizes $X^*(\alpha_k, \beta_k)$

The Gittin's index theorem is irrelevant for complex problem, but it may be used as a sanity check for newly designed bandit or RL algorithms.

The expected discounted reward of a coin with parameters (α, β) over infinite time is given by:

$$E\Big[\sum_{t=1}^{\infty} \gamma^{t-1} r_t\Big] = \frac{1}{1-\gamma}\frac{\alpha}{\alpha+\beta}.$$

It can be seen from the definition that the Gittin's index needs to always be bigger. Yet, when $\alpha + \beta$ is already large, they are approximately equal. We have

$$X^*(\alpha, \beta) \geq \frac{1}{1-\gamma}\frac{\alpha}{\alpha+\beta}$$
$$As\ \alpha+\beta \to \infty, \quad X^*(\alpha, \beta) \to \frac{1}{1-\gamma}\frac{\alpha}{\alpha+\beta}.$$

Note that the greedy policy simply uses the estimate $\frac{1}{1-\gamma}\frac{\alpha}{\alpha+\beta}$. As uncertainty increases Gittin's index, we see that Gittin's index adds an "exploration bonus."

(Note: $E[r] = E[E[r|p_k]] = E[p_k] = \frac{\alpha_k}{\alpha_k+\beta_k}$)

3 Upper Confidence Bounds

Motivated by the idea of Gittin's indices, the concept of Upper Confidence Bounds (UCB) is built. Given the information about the coin, we have the error bars and the expectation of each coin. UCB selects the coin with the largest upper confidence. This avoids cases where it has no chance of being optimal. Let UCB

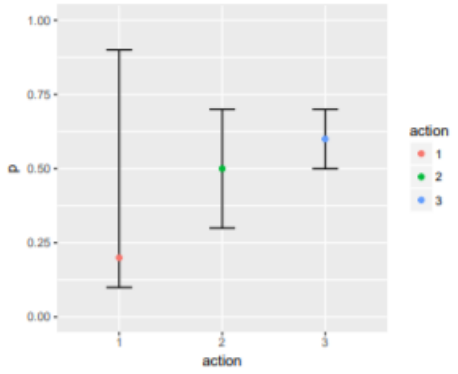

Figure 1: Estimated parameters with confidence intervals in UCB method

of the coin k be u(k). Some typical upper confidence bounds are

$$u(k) = quantile_{\alpha_k\beta_k}(1 - \frac{1}{t}),$$
$$u(k) \approx E[p_k] + \frac{\log t}{\sqrt{\alpha_k + \beta_k}} \quad \text{or} \quad u(k) \approx E[p_k] + \frac{\log t}{\sqrt{n_k}}.$$

where $\alpha_k + \beta_k$ or n_k is the number of times the coin k has been tried. While this method works well for the Beta-Bernoulli bandit, it is often computationally cumbersome for more complex problems.

4 Thompson Sampling

The Thompson sampling algorithm is similar at an abstract level to the UCB algorithm and it is computationally less expensive. In Thompson sampling, instead of working with the posterior expected value of the mean rewards or confidence sets, we sample $\hat{\theta}_t$ from the posterior distribution and then similar to the greedy algorithm, choose the arm corresponding to the highest estimated mean reward. In other words,

- Sample $\hat{\theta} \sim \mathbb{P}[\theta = \cdot | H_{t-1}]$
- Apply $x_t = \arg\max_{x \in X} \mathbb{E}[r_t | \theta = \hat{\theta}, x_t = x]$
- Observe y_t, and $r_t = r(y_t)$

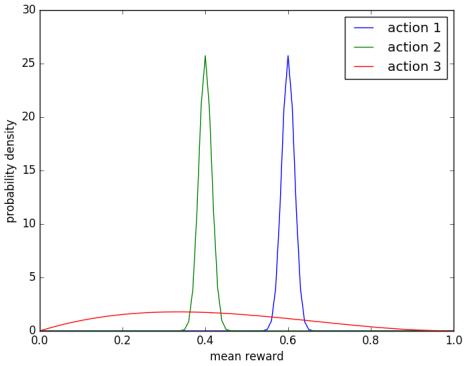

Figure 2: Probability density functions over mean rewards

Consider three actions with mean rewards $\theta \in R^3$. Each time an action k is selected, a reward of 1 is generated with probability θ_k. Otherwise, a reward of 0 is generated. The mean rewards are not known to the agent and the mean rewards can be represented as posterior distributions. These posterior distributions are represented by the probability density functions plotted in Figure 1. Action 1 is likely to be selected by the greedy algorithm since the uncertainty around this expected mean reward is small. Action 2 is avoided as it is unlikely that $\theta_2 > \theta_1$. However, there is some chance that $\theta_3 > \theta_1$ due to which the agent needs to try action 3. The purely greedy algorithm is unlikely to ever choose action 3 and the $\epsilon-$greedy algorithm wastes resources by spending a fair amount of exploratory effort on actions that have almost no chance of being optimal. Thompson sampling on the other hand allocates exploration effort more intelligently.

5 Analysis of Regret

The performance of algorithms can be studied and compared using plots of regret. The regret of an algorithm over time t is the difference between the mean reward of an optimal action and the action selected by the algorithm.

$$Regret = E\left[\sum_{t=1}^{T} {r_t}^*\right] - E\left[\sum_{t=1}^{T} r_t\right]$$

$$r_t{}^* = \begin{cases} 1 & \text{with probability } \max_k p_k \\ 0 & \text{with probability } 1 - \max_k p_k \end{cases}$$

The objective is to maximize expected cumulative reward over a long time horizon. For Thompson sampling, the mean cumulative reward converges to a fixed slope line over time, and the per period regret converges to 0 as shown in the figure.

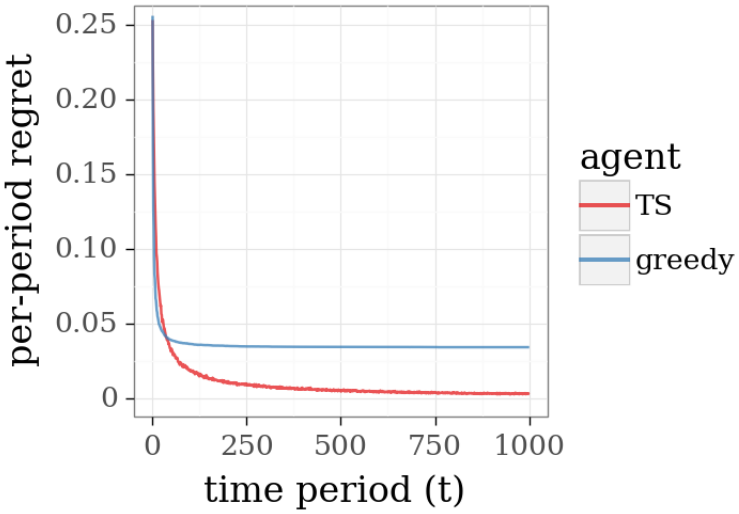

Figure 3: Per period regret from applying greedy and Thompson sampling algorithms to the three armed Bernoulli bandit

6 Active Learning problem

This is also called the online decision problem. The model is an extension of supervised learning. At each time t, a supervised learning algorithm fits a model to historical data pairs generating an estimate $\hat{\theta}$ of model parameters.The resulting model can then be used to predict the reward $r_t = r(y_t)$ from applying action x_t. The optimization algorithm selects the action x_t that maximizes expected reward assuming that $\theta = \hat{\theta}$. This action is then applied to the system to observe the outcome y_t.

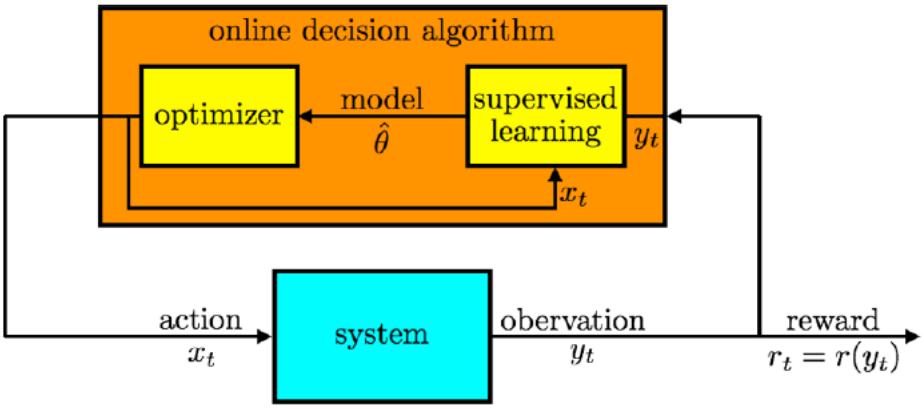

Figure 4: The overview of the active learning setting

7 Shortest path

In the shortest path problem, we want to travel from a source node to the destination node of a graph G = (V, E). For this transition, we experience some delay which is the sum of the random delays experienced at each edge that has been crossed along the path. The set of possible paths from the source to the destination are the set of actions in the reinforcement learning setting. The delays observed along the path are the observations and the sum of the delays is the loss that is incurred. The delays experienced are represented as θ_e with $e \in E$. The objective is to

$$\min_{v_1=s,v_2,.....,v_n=d} \theta_{(v_1,v_2)} + \theta_{(v_2,v_3)} + ... + \theta_{(v_{n-1},v_n)}$$

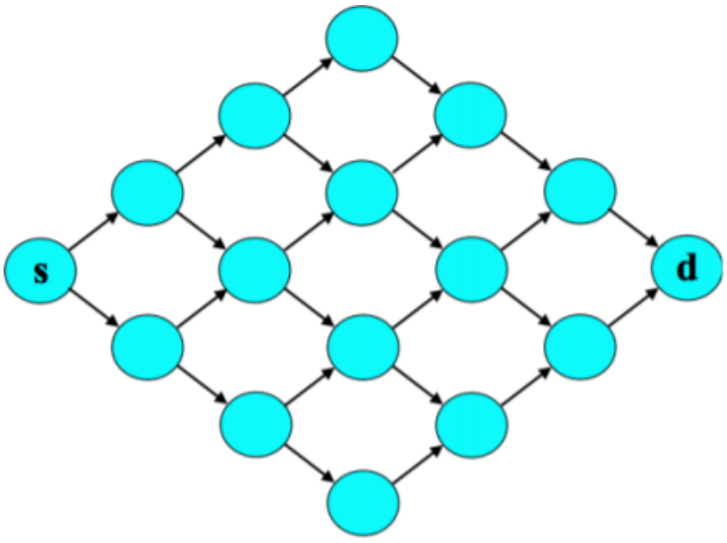

Figure 5: Binomial bridge with six stages

Given a graph $G = (V, E, v_s, v_d)$ where $v_s, v_d \in V$, we have that

$$\begin{aligned}
\textit{Mean delay} &: \theta_e \textit{ for } e \in E, \\
\textit{Action} &: x_t = (e_1, \ldots, e_n), \textit{a path from } v_s \textit{ to } v_d, \\
\textit{Observed Delay} &: y_t = (y_{t,e_1}, y_{t,e_2}, ..., y_{t,e_n}) \\
\textit{Reward} &: r_t = -(y_{t,e_1} + y_{t,e_2} + ... + y_{t,e_n}) \\
\textit{Distribution of } \theta_e &: \ln(\theta_e) \sim N(\ln(\mu) - \frac{\sigma^2}{2}, \sigma^2) \\
\textit{Distribution of } y_{t,e}|\theta_e &: \text{lognormal with mean } \theta_e.
\end{aligned}$$

$\forall e \in X_t$ we have to update distribution for θ_e. Since we have a lognormal prior and multiplicative lognormal noise, we will get a simple update.

8 References

[1] Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband and Zheng Wen (2018), A Tutorial on Thompson Sampling, Foundations and Trends in Machine Learning: Vol. xx, No. xx, pp 118. DOI: 10.1561/XXXXXXXXX.