

# Project Milestone

● Graded

## Group

Chris Lapop Salazar

Eric Peterson

Michelle Nhung Le


 [View or edit group](#)

## Total Points

20 / 20 pts

## Question 1

### Project Outline

 20 / 20 pts

✓ - 0 pts Correct

- 2 pts Went over page limit

- 1 pt Didn't use NeurIPS formatting

💬 [R1] Nice work implementing the initial environment setup and a modification of Thompson sampling. This does seem like a fairly simple extension of ideas from class, though. I'm more excited about the more advanced setting you're considering in phase 2. Make sure to keep careful records of your experiments to avoid repeating work, and don't fret too much if you aren't able to achieve close-to-optimal performance: we're more concerned about the quality of your thinking and experimentation process than the performance of your final policy. Good luck!

[R2]

Great work, seems like you have made substantial progress in implementing the environment - it's a large coding feat so definitely appreciate the hard work here. I think the pseudocode for your algorithm is detailed, and it's great that you already have some initial experiment plots. Given the large nature of the project, focus on getting each stage of experiments done. Then, implement as much as you can given your time constraint. As far as next steps go, I would simply reiterate the previous points I made on the project proposal, and otherwise good progress and best of luck for the final report!

---

# CS/STAT 184(0) Project Milestone

---

**Christopher Lapop Salazar**  
**Eric Peterson**  
**Michelle Nhung Le**  
clapopsalazar@college.harvard.edu  
peterson@g.harvard.edu  
michellele@college.harvard.edu

## Abstract

1        We are working on a model of hobby card-collecting. This includes purchasing  
2        box sets and pulling the cards within the box sets, each with its own market values  
3        and probability of being pulled. The model will later pivot toward the objective of  
4        specific collection of cards using these boxes.

## 5    1    Project Phases

6        We begin by re-articulating our problem formulation before describing the progress we have made  
7        so far on its implementation. For brevity, we won't discuss the more ambitious extensions that we  
8        mentioned in our proposal but haven't started working on yet.

### 9    1.1    Phase 1: Unknown box pull rates, minimize regret with different box choices

10       This phase will focus on a generalization of the Bernoulli Bandit. Instead of outputting 0 or 1  
11       drawn from a Bernoulli distribution as we learned in class, a pull outputs a card, which is represented  
12       as a one-hot vector drawn from a categorical distribution. The distributions of the different arms are  
13       not known by the RL agent ahead of time, so exploration is required. In order to quantify the regret,  
14       we will try to maximize the value of the cards drawn during the time horizon where the "optimal  
15       pack" is simply the pack with the highest expected market value. In later phases, we will consider  
16       optimization for more nuanced objective functions based on trying to build a desired hand rather than  
17       just maximizing the market value.

### 18    1.2    Phase 2: Maximize probability of completing a specific deck of cards within set budget

19       Phase 2 introduces selling individual cards at market value, which allows the agent to extend the  
20       time horizon of the model by selling the cards for more card pulls. This marks a change from an MAB  
21       problem towards an MDP problem, as the current budget and cards drawn so far are used to make  
22       decisions. The state space encodes the cards that have been pulled so far and the remaining budget  
23       for buying cards. The action space consists of the arms to pull from Phases 1-2, and additionally the  
24       option to sell cards to a merchant to recoup money for more card packs.

25       Actions that result in acquiring needed cards could generate rewards, while actions that don't result  
26       in acquiring needed cards would generate either no reward or smaller rewards if they give high-value  
27       cards that can be sold to buy more card packs. The distributions of cards granted by the different  
28       packs are encoded in transition probabilities of the MDP. The state space consists of all possible  
29       combinations of cards, which is quite high-dimensional, so we will look into different exploration  
30       and policy gradient methods for learning the MDP and finding an optimal policy. An additional

ability to purchase cards at market value price is being considered, which would allow guaranteed (but possibly overly costly) steps towards completing the specific deck of cards.

We can optimize card collection utility by applying Proximal Policy Optimization. Environment settings are:

**State Space:**  $s_t = \mathbf{x}_t$ , where  $\mathbf{x}_t \in \mathbb{N}^N$  is current collection (count of each card) ( $N$  is total number of unique cards in possession)

**Action Space:**

- Buy box:  $\mathcal{A}_{\text{buy}} = \{a_{\text{box}}(k) \mid k \in [K]\}$
- Sell card:  $\mathcal{A}_{\text{sell}} = \{a_{\text{sell}}(i) \mid i \in [N]\}$
- Direct purchase:  $\mathcal{A}_{\text{purchase}} = \{a_{\text{purchase}}(i) \mid i \in [N]\}$

**Reward Function:**

$$R(s_t, a_t, s_{t+1}) = \begin{cases} R_{\text{complete}} & \text{if collection completed} \\ R_{\text{progress}} \cdot \Delta\text{completion} + R_{\text{value}} \cdot \text{value}(c_t) & \text{if closer to target} \\ R_{\text{value}} \cdot \text{value}(c_t) & \text{otherwise} \end{cases}$$

where

- $\Delta\text{completion}$  measures progress toward target collection
- $\text{value}(c_t)$  is associated market value of acquired card

## 2 Phase 1

### 2.1 Environment

We wrote an environment class called PACKPOOL from scratch, which will also serve as a core for the environment in the later phases of our project. The class contains num\_packs "packs" objects, which correspond to the boxes with randomly-sampled cards. In order to make the environment correspond to some realistic card-collecting scenarios, there are 3 sub-categories of cards ("common," "uncommon," and "foil") which each have different ranges of values. The class stores the lists of cards that each box can possibly contain, the probabilities of drawing them, and each of their individual values. The PACKPOOL class also includes methods to open packs (effectively, sample according to the card probabilities within each respective pack) and calculate the values of sets of cards.

### 2.2 Algorithm

The problem of trying to maximize the value of collected cards by opening packs with a priori unknown card distributions is conceptually similar to the multi-armed bandit problem that we studied in class. The most effective approach we learned for this kind of problem was Thompson sampling, but in order to apply this algorithm to our problem we need to extend it to deal with an important distinction from Bernoulli bandits: the arms now output a vector drawn from a multinomial distribution, with the reward being a weighted sum of the categories drawn (here, the categories are the cards and the weights are their individual values).

Since the essence of the Thompson sampling algorithm is to maintain Bayesian priors on the arm distributions and sample them according to which ones seem most promising given all existing information, this suggests that we can straightforwardly adapt it if we have a conjugate prior for the multinomial distribution and some way of ranking arms to compute the  $\arg \max_k$  over boxes. It turns out that the Dirichlet distribution  $\text{Dir}(\alpha_k)$  is a conjugate prior of  $\text{Mult}(K, \mathbf{p})$  and has a similar Bayesian update rule to that of the Beta distribution, which is just to increment  $\alpha_k^j$  by one each time box  $k$  outputs card  $j$ .

58 Finally, we define the reward of a box pull  $\mathbf{n}_t$  as the total value of all the cards drawn, i.e.  $r_t := \mathbf{n}_t \cdot \mathbf{v}$ ,  
 59 where  $\mathbf{v} \in \mathbb{R}^J$  is a vector containing all of the individual card values. Then, the regret is just the  
 60 cumulative sum of differences between the expected value of the pulled boxes and the box with the  
 61 highest expected value.

62 In summary, our new algorithm can be written as:

---

**Algorithm 1** Modified Thompson Sampling for Boxes

---

**Require:** Number of boxes  $K$ , number of cards  $J$ , horizon  $T \geq K$

---

```

1: Initialize  $\alpha_k^{(j)} = 1 \in \mathbb{R}^{J,K}$ 
2: for  $t = 0, 1, \dots, T-1$  do
3:   for  $k = 1, 2, \dots, K$  do
4:     Sample  $\hat{\theta}_k \sim \text{Dir}(\alpha_k)$ 
5:     Compute reward map  $\hat{r}_k = r(\hat{\theta}_k)$ , where  $r$  is the function that maps the hand to its value,
        $\mathbb{R}^J \rightarrow \mathbb{R}$ 
6:   end for
7:   Open box  $a_t = \arg \max_{k \in \{1, \dots, K\}} \hat{r}_k$ , where ties are broken uniformly at random
8:   Update the belief for box  $a_t$  using the drawn cards  $\mathbf{n}_t$ :

```

$$(\alpha_{a_t}^{(1)}, \alpha_{a_t}^{(2)}, \dots, \alpha_{a_t}^{(J)}) \leftarrow (\alpha_{a_t}^{(1)} + n_t^{(1)}, \alpha_{a_t}^{(2)} + n_t^{(2)}, \dots, \alpha_{a_t}^{(J)} + n_t^{(J)})$$

```

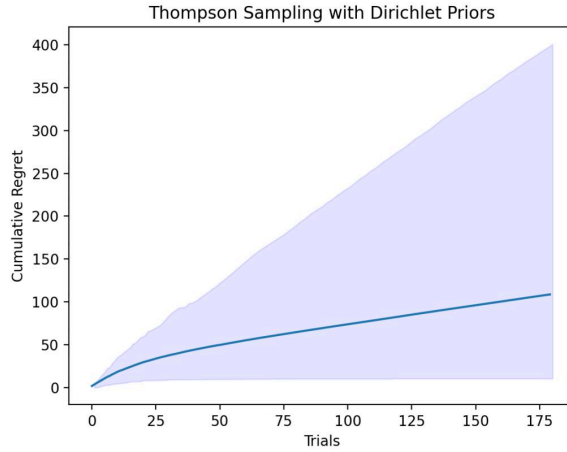
9: end for

```

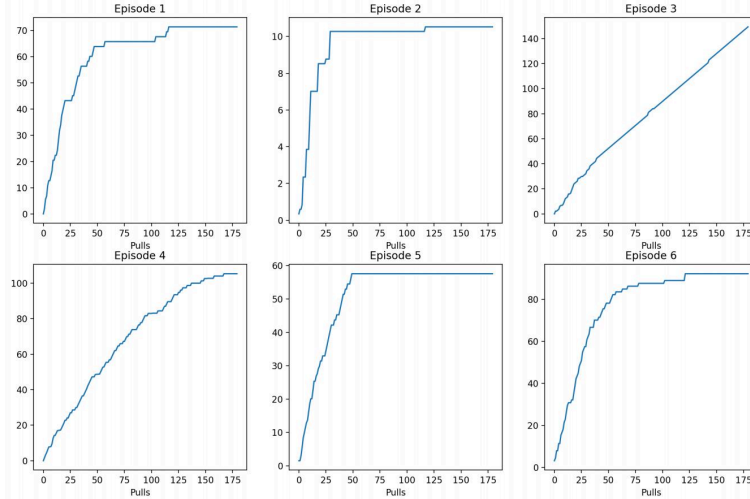
---

### 63 2.3 Data generation

64 We ran our algorithm on the PACKPOOL class for 100 episodes and a horizon of  $T = 180$  box pulls.  
 65 The mean regret with 95 percent confidence intervals is plotted below:



66 The algorithm is evidently able to achieve sublinear regret, although with a fairly large variance.  
 67 Plotting the first 6 episodes suggests that algorithm's behavior is fairly "streaky:" it is often able to  
 68 quickly identify the optimal box and almost completely stops exploring (Episodes 1, 2, 5 and 6) but  
 69 occasionally stops exploring with a high-confidence prior on the incorrect optimal box (Episode 3):  
 70



71

## 72 2.4 Possible extensions of phase 1

73 Having explored an algorithm for maximizing the reward of box pulls within a finite time horizon, it  
 74 would be interesting to see whether the Bayesian approach we used to learn the box distributions can  
 75 be incorporated into the next phase of our project, where we try to assemble a given "deck" of cards  
 76 within a finite time horizon (which is more complex than simply trying to maximize reward, and has  
 77 some similarities to the "coupon collector" problem. The large variance in the long-term regret of  
 78 our data also suggests that our algorithm might be too greedy, and perhaps updating the Dirichlet  
 79 parameters by  $1 - \epsilon$  could improve the performance. Analyzing the performance of the modified  
 80 Thompson sampling algorithm theoretically might also shed some light on this behavior and could be  
 81 instructive, especially considering that we didn't analyze the original Thompson sampling algorithm  
 82 in very much theoretical depth in class.

## 83 3 Phase 2

### 84 3.1 Environment

85 In approaching Phase 2, the environment makes use of the `PACKPOOL` class with modifications  
 86 to the `open_pack_list` function to provide the specific cards produced from pack openings rather  
 87 than only producing the card values. Most notably, Phase 2 would involve the creation of a class  
 88 `DefinedCollection` whose reward function is built to provide a full reward in cases where a pulled  
 89 card is found and then removed in a living `target_list` and partial rewards based on cards' values above  
 90 a certain threshold. The number of pack openings will also be modified, based on the subtraction of a  
 91 consistent pack price from a budget, allowing cards to contribute toward opening an additional pack.

### 92 3.2 Algorithm

93 The problem of maximizing probability to complete the specified collection initially led to us  
 94 considering Proximal Policy Evaluation in the Project Proposal. However, UCB-VI is being highly  
 95 considered due to how sparse the rewards provided from the packs may be, allowing UCB-VI's  
 96 exploration to shine in making sure card packs are not only chosen with confidence but are selected  
 97 without neglecting the potential of less explored packs.