Hello everyone, my name is Zilin Huang and I am a senior in ACMS. I am glad that I was chosen as one of the participants for this quarter's SPA Directed Reading Program, under Zhaoqi Li's guidance. Our project focuses on adaptive experimental design, that is, designing experiments and adapting its procedures along with the change of observed data. One particular instance is multi-armed bandits.

Let's focus on the case of gambling. Suppose there are a bunch of slot machines in front of us, and in the very ideal case, assume no cost occurs when playing with these machines, and we would only win or earn no dollars during each round. Under this setting, we need to decide the best strategy in order to win the largest amount of dollars possible. What would these strategies look like? Now, if we take a statistical approach, and simplify the mechanism behind slot machines as a bunch of probability distributions of dollars, we can find a way to really estimate the total dollars we could earn, and this idea is called "stochastic bandit". The main objective of it is that the player conducts the action of playing one of the arms during each round, and try to maximize the cumulative rewards after some rounds. For example, the simplest of this "stochastic bandit" is "Bernoulli Bandit", in which each arm follows the Bernoulli distribution, and generates either one or zero dollars only in each round.

Another term related to stochastic bandit is Regret, the difference of the mean reward (or just deficiencies here) between the optimal strategy possible and the practical strategy we conduct. We want the regret to be as small as possible, which implies the reward will be as large as possible, and we mostly calculate regret by summing up by the number of rounds, as shown in this formula. The idea of regret also

depends on the concept of “suboptimality gap”, which is difference of mean reward between an arbitrary arm and optimal arm.

The first and most basic strategy is called the Explore-then-Commit (or ETC) Algorithm. Assume we play each arm for some times, and record their mean rewards respectively. During the commit stage, we play with the single arm with largest mean rewards calculated, until the game itself ends. Although it is very straightforward to implement such an algorithm, its deficiency is large as well: The regret may grow linearly, since we can not guarantee that this chosen arm will still generate the optimal reward as we would expect. So, think about the case when we just randomly play any arm we like, and do not care how large the regret should be. That is quite similar to the case when the ETC Algorithm functions not so well.

Another useful and mostly-discussed algorithm is called Upper Confidence Bound, or UCB Algorithm. The basic idea is that after the “explore” stage, similar to the previous algorithm, we still compute the mean reward of each arm. In this case, we also define the index of “UCB” for each of these arms according to this formula here. After this process, in each round, we would play the single arm with the largest “UCB”, and then update its “UCB” value. Fortunately, the UCB Algorithm will usually generate sublinear regret (like the function of square root of x), since it will implicitly not play with arms with very low UCB, and therefore very high regret as time goes, and therefore largely reducing the overall regret. Unfortunately, the proof of formula for UCB’s regret is quite complicated, and we skip it for now. The only point we should notice here is that this whole algorithm is set under the “good event”, namely one arm’s actual sample mean does not differ from its expected mean a lot with a very high confidence.

Another interesting algorithm is called the Elimination Algorithm. Its rule is somehow different, in that in each phase (no longer rounds here), we play with each arm for some rounds, still computing their mean rewards, and ELIMINATE the arms whose mean rewards are way more smaller than that of the optimal arm. By eliminating, it means that when going to the next phase, we will not play with these arms any more. This difference is always measured by manually letting the two arms's Confidence Interval not intersect with each other, and thus comes the meaning of “too large” the difference. When there is only one arm remaining in the last, we will then begin calculating the regret for this algorithm.

The last algorithm I would like to introduce is Thompson Sampling Algorithm. Its idea is quite unique from others, in that the mean reward of each arm is no longer a number, but another probability distribution function to be estimated. (OPTIONAL) For the initial stage, we still extract some samples from each of the arms, which is then composed to another pdf. One thing to notice is that this pdf is the estimation of the actual pdf representing the actual mean of that arm, and after the initial stage in each round, we play with the arm containing the highest mean of the pdf of the mean reward. Its core idea lies in the Bayesian formula, and the formula for regret follows exactly this theorem. (OPTIONAL ENDS) One major application of Thompson Sampling is Amazon’s front-page recommendation, in which the reward represents users’ click of each of the potential categories, like ads for clothes, for books, for video games, and so on. The engineers will design algorithms that learn such actions for each customer, and tend to recommend similar pages for them in their future usage of Amazon.

That is all I would like to show for our project in Adaptive Experimental Design, thank you for your listening and you are welcome to ask any questions.