Solving Multi-armed Bandit Problems Based on Simulation Optimization

Li Qingxuan, Liu Weizhi, Yuan Yuhe, Wong Manyu

Department of Industrial & Systems Engineering
National University of Singapore

November 7, 2014

Overview

Multi-armed Bandit

- Origin: How to play slot machines wisely in casino?

Figure 1: Slot Machines in Casino

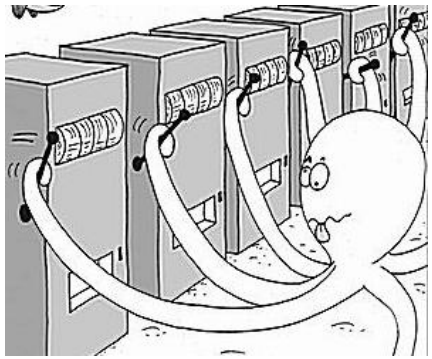


Figure 2: Multi-armed Bandit

- **Task: Maximize your total reward R with at most N trials given K slot machines to play, whose reward distribution is different.**
- Application: online advertising (A/B testing), clinical trials, adaptive routing, dynamic allocation, etc.

Problem Formulation

- Assumption: To simplify the problem without loss of generality, we assume **the reward of slot machine (also called Arm)** i **follows Bernoulli distribution with success rate** $p_i, i = 0, 1, \cdots, K-1$**.**
- Challenges:
  - Exploration vs. Exploitation Dilemma
    "Bandit problems embody in essential form a conflict evident in all human action: choosing actions which yield immediate reward (exploit current best) vs. choosing actions whose benefit will come only later (explore unknown solution)." (P.Whittle 1980)
  - Value of history information
    How to analyze history reward to evaluate each arm?
  - Confidence of optimal arm
    How confident do you believe the estimated optimal arm is the true optimal?

Notations

Table 1: Notations

Notation	Definition	Remarks
p_i	true success rate for Arm i	$p_i \in [0, 1]$
$\tilde{X}_i$	average reward of Arm i	$\tilde{X}_i \sim N(\mu_i, \frac{\sigma_i^2}{N_i})$
$\vec{r}_i$	current reward history for Arm i	$\vec{r}_i = [X_{i1}, X_{i2}, \cdots, X_{it_i}]$
R_i	current total reward for Arm i	$R_i = \vec{r}_i e^T$
N_i	current total number of draw for Arm i	the length of $\vec{r}_i$
$\hat{N}_i$	estimated next total number of draw for Arm i	please refer to OCBA
μ_i	sample mean of $\vec{r}_i$	$\mu_i = \frac{1}{N_i} R_i$
σ_i^2	sample variance of $\vec{r}_i$	$\sigma_i^2 = \frac{1}{N_i - 1} \sum_{k=1}^{t_i} (X_{ik} - \mu_i)^2$
$\tilde{p}_i$	sample point of success rates	$\tilde{p}_i \sim \text{Beta}(R_i + 1, N_i - R_i + 1)$
τ	exploration weight for ucb value	please refer to UCB

Exploration/Exploitation Dilemma - UCB

- Main Idea: Automatic balance exploration/exploitation dilemma based on ucb value.
- Procedure:
  - Step 1: Draw each Arm $i(i = 0, 1, \cdots, K-1)$ once and update $\vec{r}_i$ and N_i.
  - Step 2: Update ucb value for each Arm i

$$ucb_i = \frac{R_i}{N_i} + \tau\sqrt{\frac{2\log\sum_{i=0}^{K-1} N_i}{N_i}} \tag{1}$$

  - Step 3: Select Arm $b = \arg\max_i ucb_i$ to draw, and update $\vec{r}_b$ and N_b.
  - Step 4: Return to Step 2 until total budget run out.

History Information - Bayes Bandit

- Main Idea:
  - Generate posterior distribution $f(\theta|x)$ given prior distribution $f(\theta)$ and history information x

$$f(\theta|x) = \frac{f(x|\theta)f(\theta)}{\int_Y f(y|\theta)f(\theta)dy} \tag{2}$$

  - In our problem, we have to estimate the distribution of success rate $\tilde{p}_i$ for each arm.
  - The conjugate prior distribution of likelihood function $f(x|\theta)$ could guarantee the same formulation between prior and posterior distribution.
  - Binomial likelihood function has conjugate prior distriubtion Beta$(\alpha + 1, \beta + 1)$, where α, β are the success and failure times.
  - Beta$(1, 1)$ is just the Uniform$(0, 1)$

History Information - Bayes Bandit

- Procedure:
  - Step 1: Sample success rates $\tilde{p}_0, \cdots, \tilde{p}_{K-1}$ for each arm from its prior distribution Beta$(R_i + 1, N_i - R_i + 1)$.
  - Step 2: Select Arm $b = \arg\max_i \tilde{p}_i$ to draw, and update $\vec{r}_b$ and N_b.
  - Step 3: Return to Step 1 until total budget run out.

Confidence of Optimal Arm - OCBA

- Main Idea: Maximize the probability of correct selection based on wise allocation rules.

$$APCS = 1 - \sum_{i=0, i\neq b}^{K-1} P\{\tilde{X}_b < \tilde{X}_i\} \quad = 1 - \sum_{i=0, i\neq b}^{K-1} \Phi\left(\frac{\mu_i - \mu_b}{\sqrt{\frac{\sigma_b^2}{N_b} + \frac{\sigma_i^2}{N_i}}}\right) \qquad (3)$$

Confidence of Optimal Arm - OCBA

- Procedure:
  - Step 1: Draw each Arm i for n_0 times and update $\vec{r}_i$ and N_i.
  - Step 2: Calculate the sample mean μ_i and sample variance σ_i^2 based on $\vec{r}_i$ for each Arm i. Then find current best Arm $b = \arg\max_i \mu_i$.
  - Step 3: Increase the computing budget by 1 and compute the next allocation rule

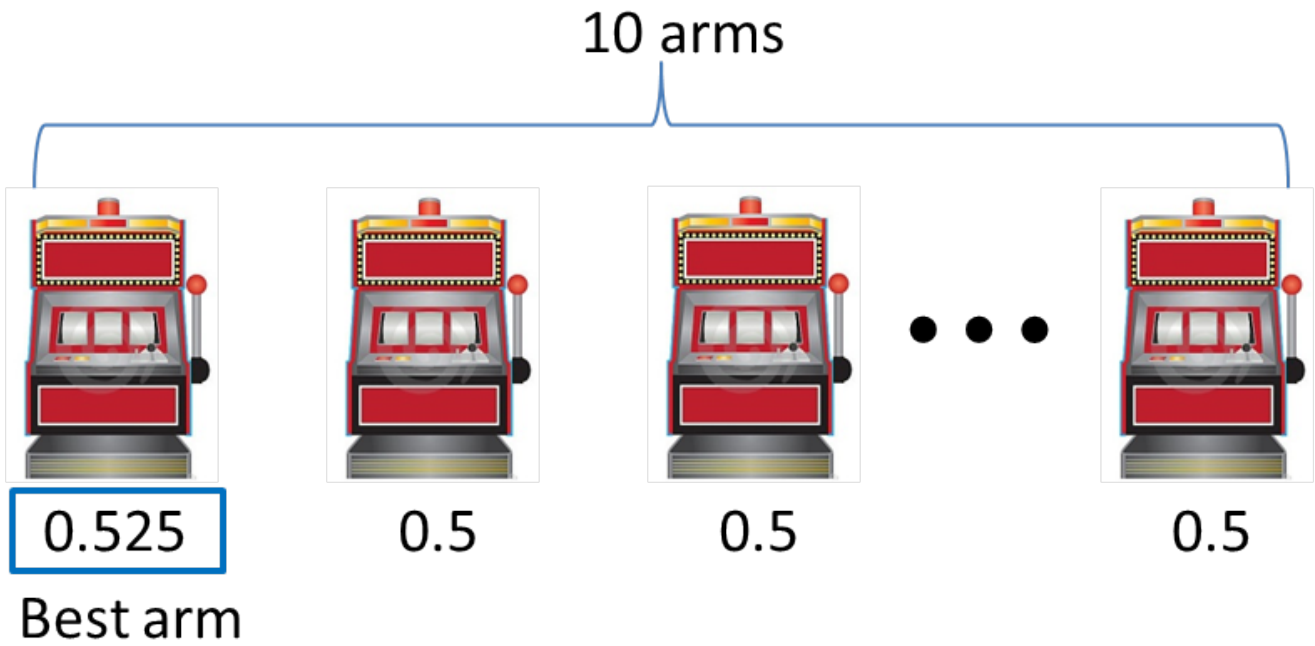
$$\frac{\hat{N}_i}{\hat{N}_j} = \left(\frac{\sigma_i(\mu_b - \mu_j)}{\sigma_j(\mu_b - \mu_i)}\right)^2, i \neq j \neq b \quad \hat{N}_b = \sigma_b \sqrt{\sum_{i=0, i\neq b}^{K-1} \left(\frac{\hat{N}_i}{\sigma_i}\right)^2} \tag{4}$$

$$\sum_{i=0}^{K-1} \hat{N}_i = \sum_{i=0}^{K-1} N_i + 1 \tag{5}$$

  - Step 4: Select Arm $k = \arg\max_i(\hat{N}_i - N_i)$ to draw and update $\vec{r}_k$ and N_k.
  - Step 5: Return to Step 2 until total budget run out.

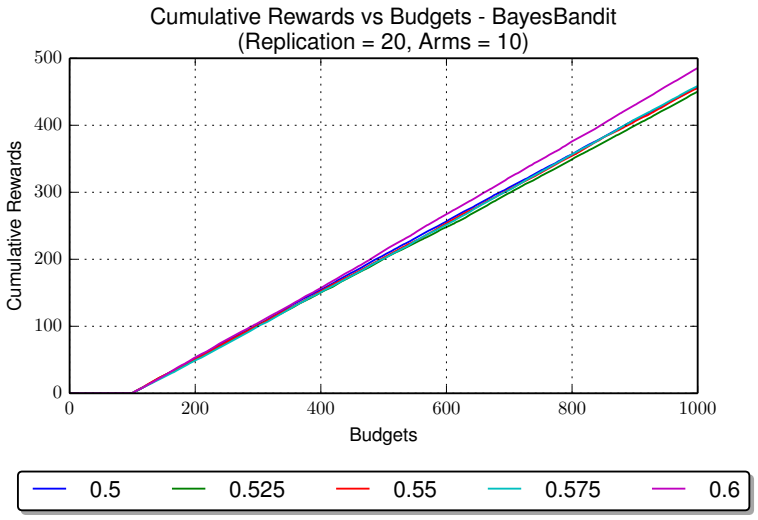
Main Experiments

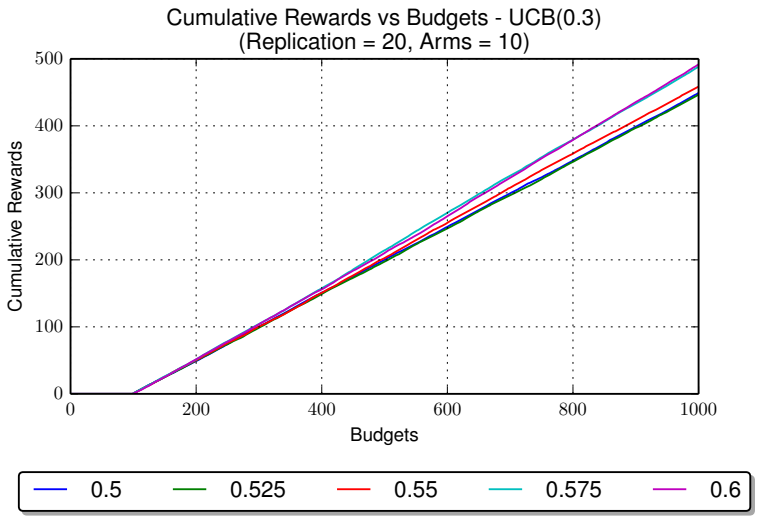
- Change of the Best Arm Winning Probability
- Change in Budget
- Change in Initial Budget (OCBA)
- Comparison of Efficiency

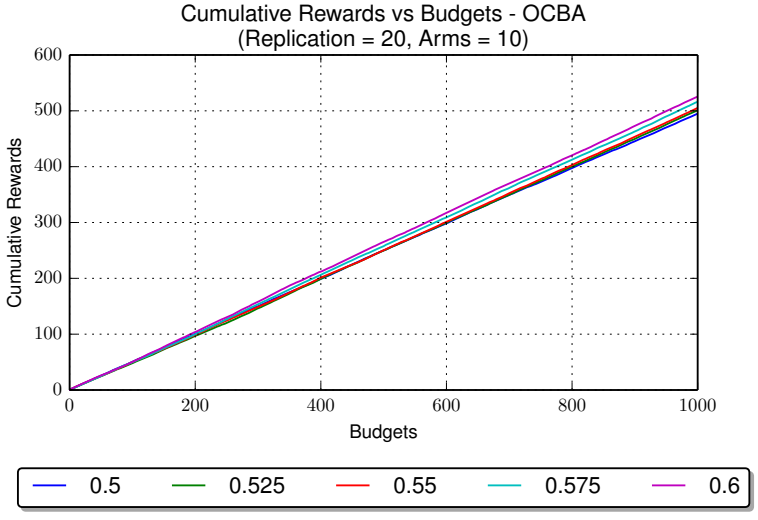


- Variation of the Best Arm Winning Probability while keeping the Winning Probability of all other arms at 0.5

Change of the Best Arm Winning Probability

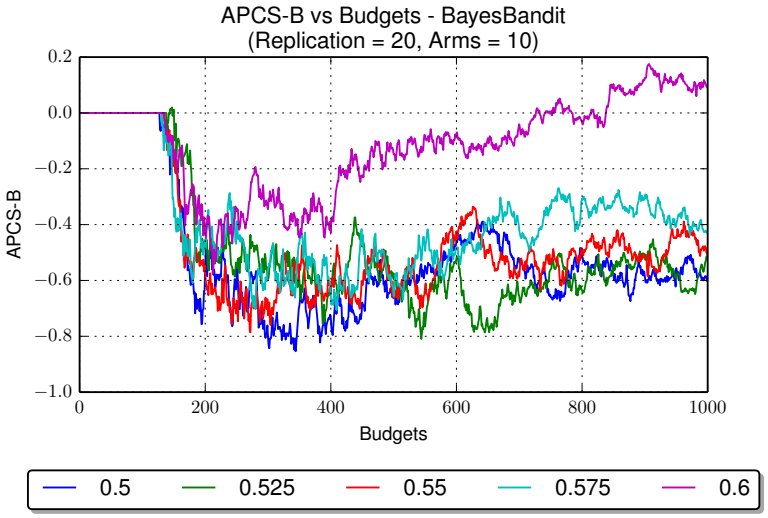


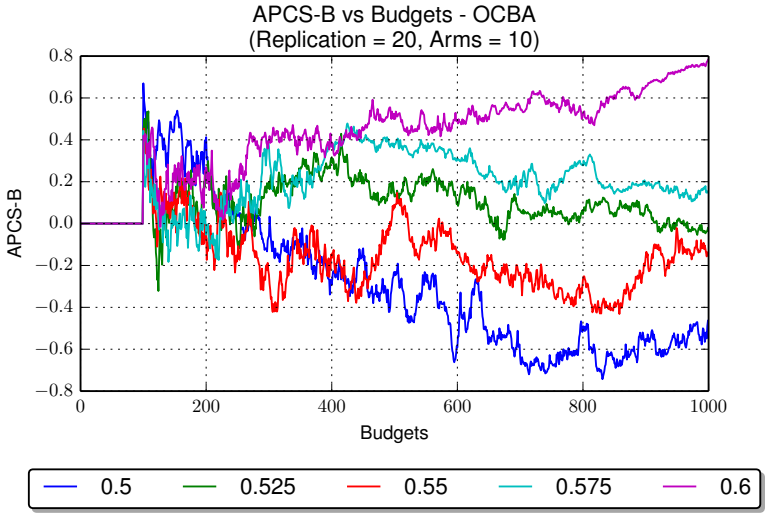


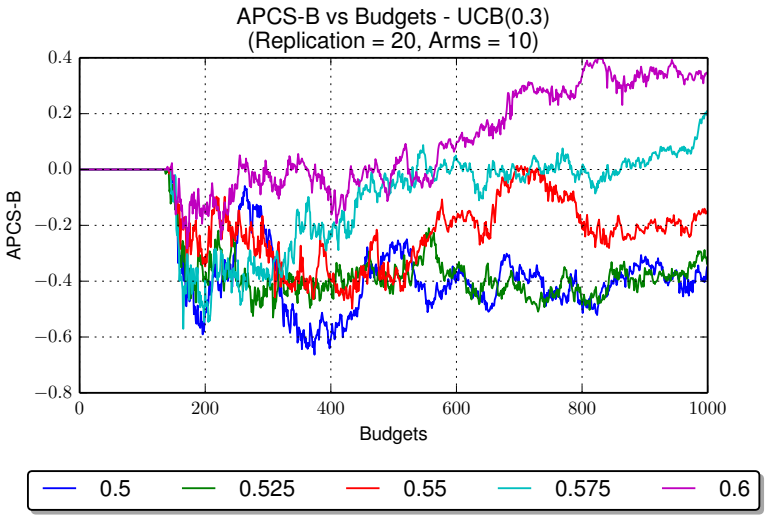


- Difference in cumulative reward increases as time goes on.
- Least obvious in OCBA.
- Most obvious in the UCB case. (Small Critical Difference)

Change of the Best Arm Winning Probability

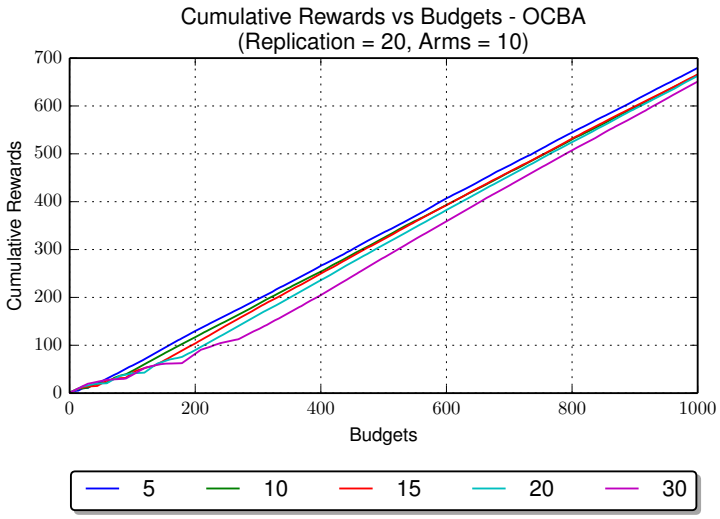


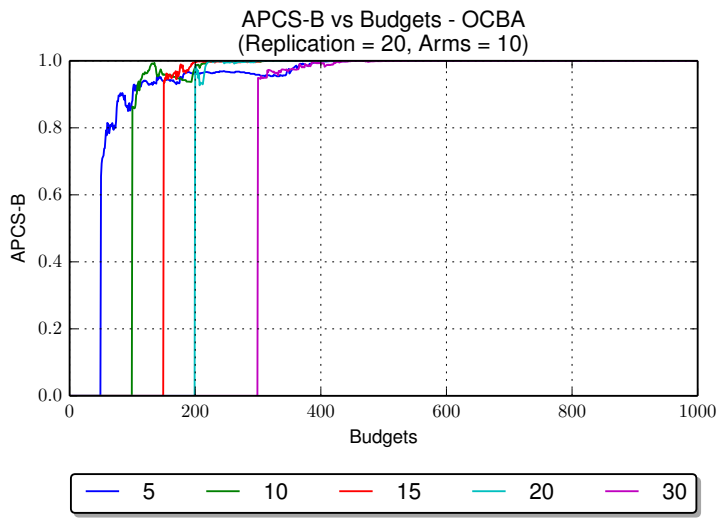




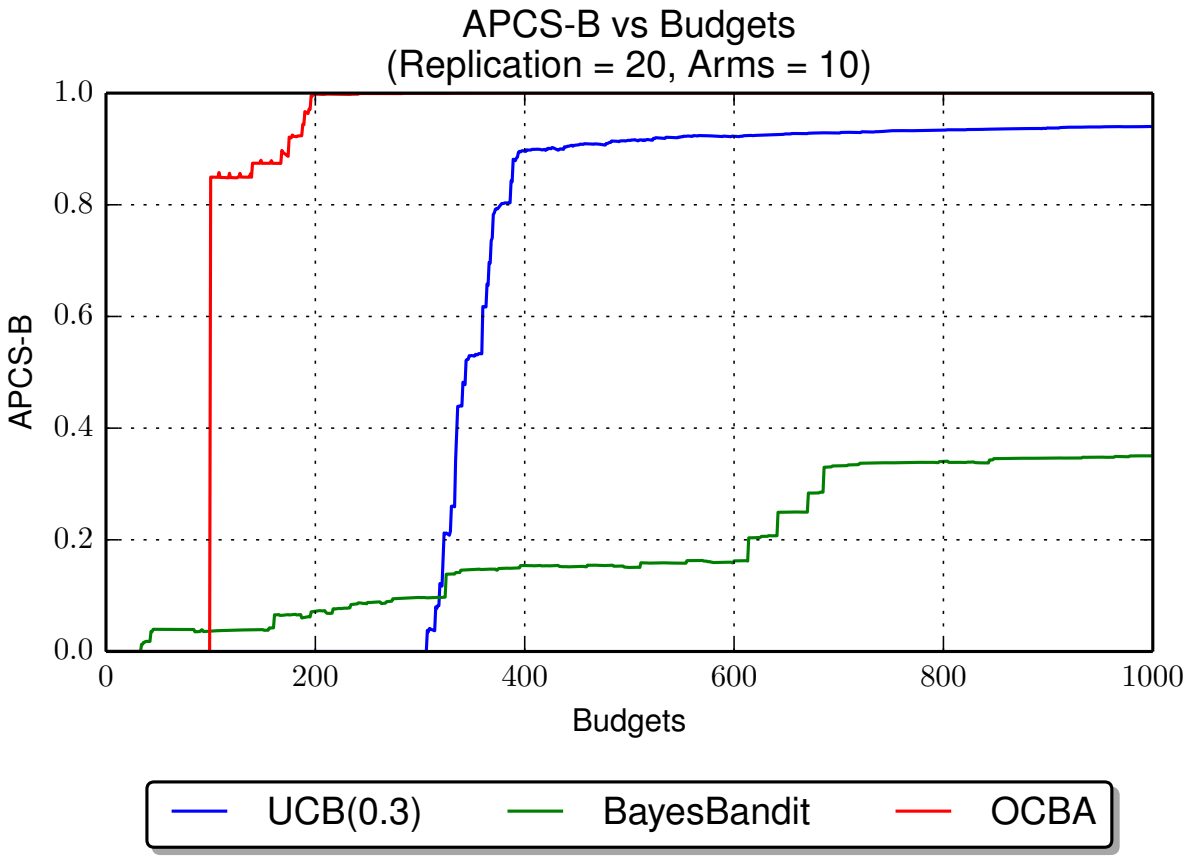
- Not very useful for UCB and BayesBandit.
- OCBA performance is a lot better.

Change in Initial Budget (OCBA)

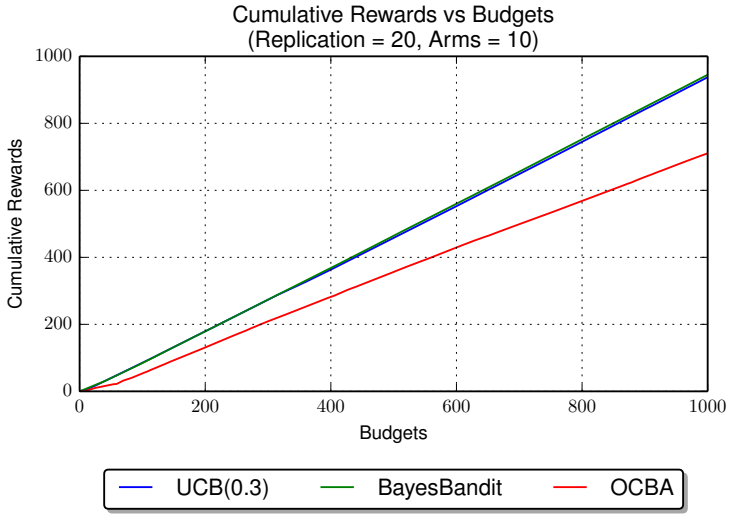


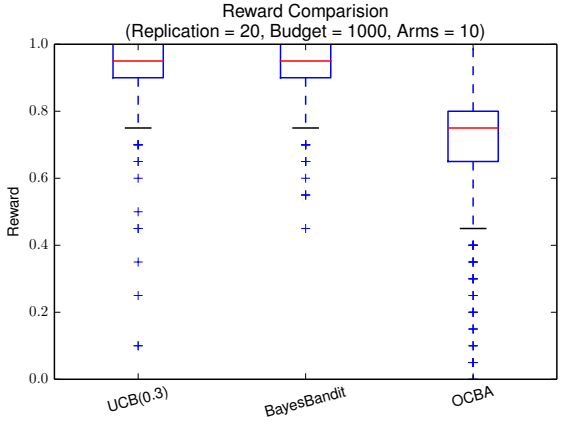


- Regardless of the initial budget allocation, the different experiments converge.
- The convergence towards a APCS-B of 1 increases as the initial budget increases, saturating at around an initial budget of 15.

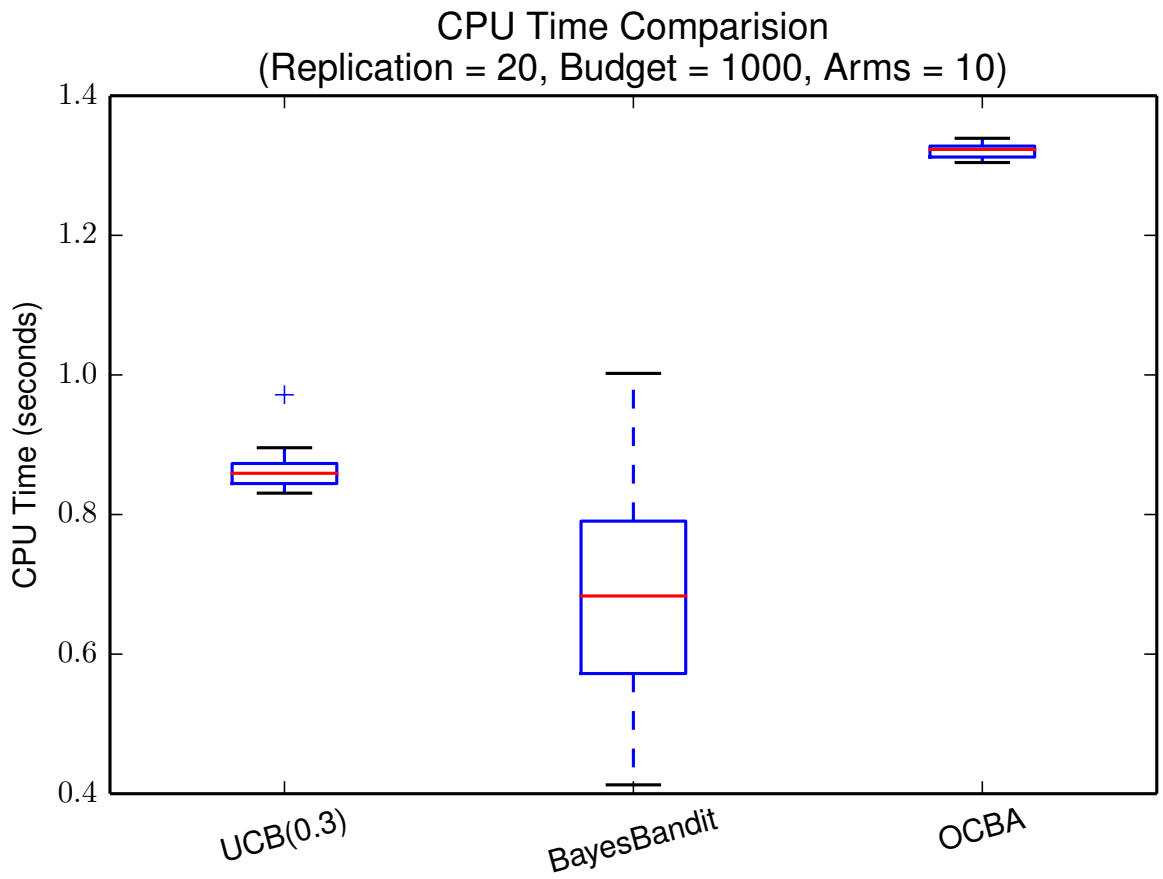


- UCB and OCBA show promising convergence
- OCBA is faster at obtaining the desired result





- The performance of UCB and BayesBandit are largely superior to that of OCBA.
- This shows that the focus of OCBA is not on obtaining the highest profit.



- Though OCBA seems to be more complex, the difference is still counted as negligible for simple experiments. This difference may be more significant when dealing with more complicated rewards distributions.

Summary of the different Algorithms

- OCBA converges the fastest in terms of finding the Probability of Correct Selection, but UCB provides the largest reward in the long run.
- A suggestion is to perform the OCBA algorithm in the initial warm-up stages and switch over to the UCB after.
- While the BayesBandit case shows an overall good performance, prior information about the distribution needs to be known, and its effectiveness may decrease if we use the wrong distribution.

Thanks!
Q&A