$\mathcal{X}$-Armed Bandits

S´ebastien Bubeck, R´emi Munos

Gilles Stoltz, Csaba Szepesv´ari (*University of **Alberta***)

Hong Cheng
chenghong@kw.ac.kr
Advisor: Prof. Hyukjoon Lee

Computer Intelligence Network Laboratory, Dept. of CE,
KwangWoon University

CINe Lab

$\mathcal{X}$-Armed Bandits

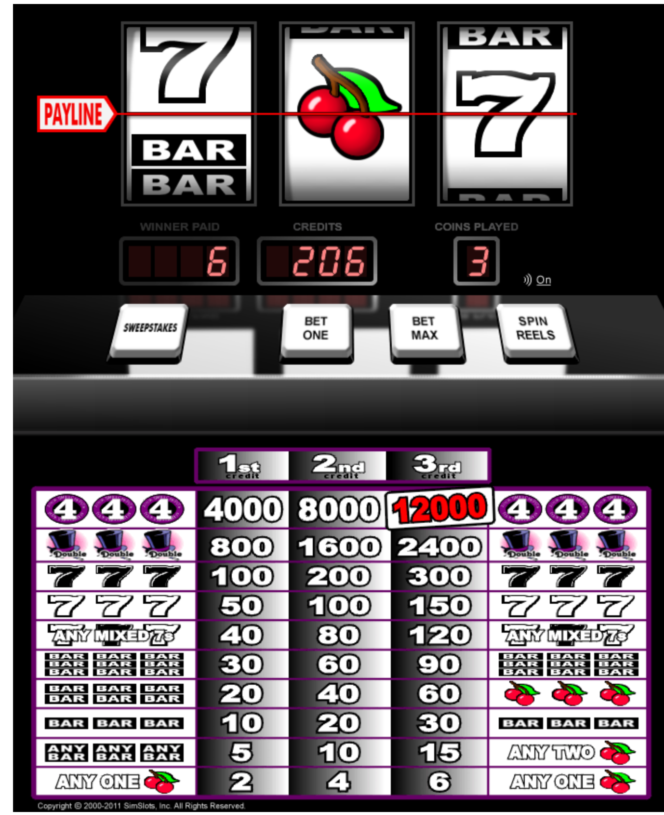
- Introduction
- Problem Setup
- The HOO Strategy

Introduction

- **What's bandit problem?**
  - The **multi-armed bandit problem** is the problem a ***gambler*** faces at a row of **slot machines** when deciding which *machines* to play, how many *times* to play each machine and in which *order* to play them.
  - When played, each machine provides a **random reward** from a distribution specific to that machine.

Introduction

- **What's bandit problem?**
  - Free Online Slot Machines
    https://www.freeslots.com/Slot7.htm
  - Old fashion Slot Machine



Introduction

▪What's bandit problem?

- The **gambler** pulls the arms of the machines one by one in a sequential manner, simultaneously learning about the machines' **payoff-distributions** and gaining actual monetary reward.
- Thus, in order to maximize his gain, the gambler must choose the next arm by taking into consideration both the urgency of gaining reward ("**exploitation**") and acquiring new information ("**exploration**").
- Maximizing the total cumulative *payoff* is **equivalent** to minimizing the (total) ***regret***, that is, **minimizing the difference** between the total cumulative payoff of the gambler and the one of another gambler who chooses the arm with the best mean-payoff in every round.

Introduction

- **The Motivation**
  - Handle the $\mathcal{X}$-armed bandits problems in a unified framework.
  - More precisely, we consider a general setting that allows us to study bandits with almost no restriction on the set of arms.
  - In particular, we allow the set of arms to be an arbitrary measurable space.

Introduction

- **Assumptions**
  - The **gambler** has some **knowledge** about the behavior of the mean-payoff function
    - in terms of its local regularity around its maxima, roughly speaking
    - since we allow **non-denumerable sets**,
  - There exists a **dissimilarity function** that constrains the behavior of the mean-payoff function
    - where ***a dissimilarity function is a measure of the discrepancy between two arms*** that is neither symmetric, nor reflexive, nor satisfies the triangle inequality.
    - In particular, the dissimilarity function is assumed to locally set a bound on the decrease of the mean-payoff function at each of its global maxima.[?]
  - The decision maker can construct a **recursive covering** of the space of arms **in such a way** that the diameters of the sets in the covering **shrink** at a known geometric rate *when measured with this dissimilarity*. [?]

Problem Setup

- **Stochastic bandit problem**
  - $\boldsymbol{B}$ is a pair $\boldsymbol{B} = (X, M)$
    - X is a **measurable space of arms**
    - M determines the **distribution of rewards** associated with each arm
    - We say that M is a ***bandit environment*** on X .
  - Formally, M is an mapping $X \rightarrow M_1(R)$,
    - $M_1(R)$ is the space of probability distributions over the reals.
    - The distribution assigned to **arm** $x \in X$ is denoted by M_x
  - We require that for each ***arm*** $\boldsymbol{x} \in X$, the distribution M_x admits a first-order moment;

Problem Setup

- **Stochastic bandit problem**
  - We then denote by $f(x)$ its expectation ("**mean payoff**"),

$$f(x) = \int y \, dM_x(y)$$

  - The **mean-payoff function** f thus defined is assumed to be measurable.
  - For simplicity, we shall also assume that all M_x have bounded supports, included in some fixed bounded interval,[3] say, the unit interval $[0, 1]$.
  - Then, f also takes bounded values, in $[0, 1]$.

Problem Setup

- **Decision Maker**
  - A **decision maker** (the **gambler** of the introduction) that interacts with a *stochastic bandit problem* $\boldsymbol{B}$ plays a game at discrete time steps according to the following rules.
  - In the first round the decision maker can select an arm $X_1 \in \boldsymbol{X}$ and receives a reward Y_1 drawn at random from M_{X_1}.
  - In round $\boldsymbol{n > 1}$ the decision maker can select an arm $X_n \in \boldsymbol{X}$ based on the information available up to time $\boldsymbol{n}$, that is, $(X_1, Y_1, \ldots, X_{n-1}, Y_{n-1})$, and receives a reward Y_n drawn from M_{X_n}, independently of $(X_1, Y_1, \ldots, X_{n-1}, Y_{n-1})$ given X_n.
  - Note that a decision maker may randomize his choice, but can only use information available up to the point in time when the choice is made.

Problem Setup

- **Decision Maker**
  - The goal of the decision maker is to **maximize** his expected cumulative reward.
  - **Equivalently**, the goal can be expressed as **minimizing** the expected cumulative **regret**, which is defined as follows.
  - Let
$$f^* = \sup_{x \in \mathcal{X}} f(x)$$
    be the best expected payoff in a single round.
  - At round n, the cumulative regret of a decision maker playing B is
$$\widehat{R}_n = n f^* - \sum_{t=1}^{n} Y_t,$$
    that is, the difference between the maximum expected payoff in n rounds and the actual total payoff.

Problem Setup

▪Pseudo-regret

- In the sequel, we shall restrict our attention to the **expected cumulative regret**, which is defined as the expectation $E[R_n]$ of the cumulative regret R_n.
- Finally, we define the cumulative ***pseudo-regret*** as

$$R_n = n f^* - \sum_{t=1}^{n} f(X_t) \, ,$$

- that is, the **actual rewards** used in the definition of the regret are replaced by the **mean-payoffs** of the arms pulled.
- Since (by the tower rule) $\mathbb{E}\left[Y_t\right] = \mathbb{E}\left[\mathbb{E}\left[Y_t | X_t\right]\right] = \mathbb{E}\left[f(X_t)\right] \, ,$
- the expected values $E[R_n]$ of the cumulative regret and $E[R_n]$ of the cumulative pseudo-regret are the same.
- Thus, we focus below on the study of the behavior of $E[R_n]$

The HOO Strategy

- The HOO strategy (cf., ***Algorithm 1***) incrementally builds an estimate of the ***mean-payoff function*** f over $\boldsymbol{X}$.
- The **core idea** is to estimate f precisely around its maxima, while estimating it loosely in other parts of the space $\boldsymbol{X}$.
- HOO maintains a **binary tree** whose nodes are associated with measurable regions of the arm-space $\boldsymbol{X}$ such that the *regions* associated with nodes **deeper** in the tree (further away from the root) represent increasingly *smaller subsets* of $\boldsymbol{X}$.

The HOO Strategy

Parameters: Two real numbers $\nu_1 > 0$ and $\rho \in (0,1)$, a sequence $(\mathcal{P}_{h,i})_{h \geqslant 0, 1 \leqslant i \leqslant 2^h}$ of subsets of $\mathcal{X}$ satisfying the conditions (1) and (2).

Auxiliary function $\textsc{Leaf}(\mathcal{T})$: outputs a leaf of $\mathcal{T}$.

Initialization: $\mathcal{T} = \{(0,1)\}$ and $B_{1,2} = B_{2,2} = +\infty$.

for $n = 1, 2, \ldots$ **do** ▷ Strategy HOO in round $n \geqslant 1$
    $(h,i) \leftarrow (0,1)$ ▷ Start at the root
    $P \leftarrow \{(h,i)\}$ ▷ P stores the path traversed in the tree
    while $(h,i) \in \mathcal{T}$ **do** ▷ Search the tree $\mathcal{T}$
        if $B_{h+1,2i-1} > B_{h+1,2i}$ **then** ▷ Select the "more promising" child
            $(h,i) \leftarrow (h+1, 2i-1)$
        else if $B_{h+1,2i-1} < B_{h+1,2i}$ **then**
            $(h,i) \leftarrow (h+1, 2i)$
        else ▷ Tie-breaking rule
            $Z \sim \mathrm{Ber}(0.5)$ ▷ e.g., choose a child at random
            $(h,i) \leftarrow (h+1, 2i-Z)$
        end if
        $P \leftarrow P \cup \{(h,i)\}$
    end while
    $(H,I) \leftarrow (h,i)$ ▷ The selected node
    Choose arm X in $\mathcal{P}_{H,I}$ and play it ▷ Arbitrary selection of an arm

The HOO Strategy

Receive corresponding reward Y

$\mathcal{T} \leftarrow \mathcal{T} \cup \{(H,I)\}$ ▷ Extend the tree

for all $(h,i) \in P$ **do** ▷ Update the statistics T and $\widehat{\mu}$ stored in the path

  $T_{h,i} \leftarrow T_{h,i} + 1$ ▷ Increment the counter of node (h,i)

  $\widehat{\mu}_{h,i} \leftarrow \left(1 - 1/T_{h,i}\right)\widehat{\mu}_{h,i} + Y/T_{h,i}$ ▷ Update the mean $\widehat{\mu}_{h,i}$ of node (h,i)

end for

for all $(h,i) \in \mathcal{T}$ **do** ▷ Update the statistics U stored in the tree

  $U_{h,i} \leftarrow \widehat{\mu}_{h,i} + \sqrt{(2\ln n)/T_{h,i}} + \nu_1\rho^h$ ▷ Update the U-value of node (h,i)

end for

$B_{H+1,2I-1} \leftarrow +\infty$ ▷ B-values of the children of the new leaf

$B_{H+1,2I} \leftarrow +\infty$

$\mathcal{T}' \leftarrow \mathcal{T}$ ▷ Local copy of the current tree $\mathcal{T}$

while $\mathcal{T}' \neq \{(0,1)\}$ **do** ▷ Backward computation of the B-values

  $(h,i) \leftarrow \textsc{Leaf}(\mathcal{T}')$ ▷ Take any remaining leaf

  $B_{h,i} \leftarrow \min\left\{U_{h,i}, \max\left\{B_{h+1,2i-1}, B_{h+1,2i}\right\}\right\}$ ▷ Backward computation

  $\mathcal{T}' \leftarrow \mathcal{T}' \setminus \left\{(h,i)\right\}$ ▷ Drop updated leaf (h,i)

end while

end for

The HOO Strategy

- At each **node** of the tree, HOO **stores** some statistics based on the information received in **previous rounds**.

- In particular, HOO keeps track of the number of times a node was traversed up to round n and the corresponding empirical average of the rewards received so far.

4: **while** $(h,i) \in \mathcal{T}$ **do**
5:     **if** $B_{h+1,2i-1} > B_{h+1,2i}$ **then**
6:         $(h,i) \leftarrow (h+1, 2i-1)$
7:     **else if** $B_{h+1,2i-1} < B_{h+1,2i}$ **then**
8:         $(h,i) \leftarrow (h+1, 2i)$
9:     **else**
10:         $Z \sim \mathrm{Ber}(0.5)$
11:         $(h,i) \leftarrow (h+1, 2i-Z)$
12:     **end if**
13:     $P \leftarrow P \cup \{(h,i)\}$
14: **end while**

  - HOO assigns an optimistic estimate (denoted by ***B***) to the maximum mean-payoff associated with each node.
  - These estimates are then used to select the next node to "play".

- This is done by traversing the tree, beginning from the root, and always following the node with the highest ***B***-value (***cf., lines 4–14 of Algorithm 1***) .

The HOO Strategy

▪ Algorithm 1

- Once a node is selected, a point in the region associated with it is chosen ***(line 16)*** and is sent to the environment.

- Based on the point selected and the received reward, the tree is updated ***(lines 18–33)***.

18: $\mathcal{T} \leftarrow \mathcal{T} \cup \{(H,I)\}$
19: **for all** $(h,i) \in P$ **do**
20: $\quad T_{h,i} \leftarrow T_{h,i} + 1$
21: $\quad \widehat{\mu}_{h,i} \leftarrow \left(1 - 1/T_{h,i}\right)\widehat{\mu}_{h,i} + Y/T_{h,i}$
22: **end for**
23: **for all** $(h,i) \in \mathcal{T}$ **do**
24: $\quad U_{h,i} \leftarrow \widehat{\mu}_{h,i} + \sqrt{(2\ln n)/T_{h,i}} + \nu_1 \rho^h$
25: **end for**
26: $B_{H+1,2I-1} \leftarrow +\infty$
27: $B_{H+1,2I} \leftarrow +\infty$
28: $\mathcal{T}' \leftarrow \mathcal{T}$
29: **while** $\mathcal{T}' \neq \{(0,1)\}$ **do**
30: $\quad (h,i) \leftarrow \textsc{Leaf}(\mathcal{T}')$
31: $\quad B_{h,i} \leftarrow \min\Big\{U_{h,i}, \max\big\{B_{h+1,2i-1}, B_{h+1,2i}\big\}\Big\}$
32: $\quad \mathcal{T}' \leftarrow \mathcal{T}' \setminus \{(h,i)\}$
33: **end while**

The HOO Strategy

▪ Algorithm 1

- In the algorithm listing the ***recursive*** computation of the ***B***-values ***(lines 28–33)*** makes a local copy of the tree.

28: $\mathcal{T}' \leftarrow \mathcal{T}$
29: **while** $\mathcal{T}' \neq \{(0,1)\}$ **do**
30: $\quad (h,i) \leftarrow \textsc{Leaf}(\mathcal{T}')$
31: $\quad B_{h,i} \leftarrow \min\Big\{U_{h,i},\, \max\{B_{h+1,2i-1}, B_{h+1,2i}\}\Big\}$
32: $\quad \mathcal{T}' \leftarrow \mathcal{T}' \setminus \{(h,i)\}$
33: **end while**

- Other **arbitrary** choices in the algorithm as shown here are how tie breaking in the node selection part is done(***lines 9 –12***)

9: **else**
10: $\quad Z \sim \mathrm{Ber}(0.5)$
11: $\quad (h,i) \leftarrow (h+1, 2i-Z)$
12: **end if**

- ***or*** how a point in the region associated with the selected node is chosen ***(line 16)***.

16: Choose arm X in $\mathcal{P}_{H,I}$ and play it

The HOO Strategy

▪Cover tree

- **The tree of coverings** which HOO needs to receive as an **input** is an ***infinite binary tree*** whose nodes are associated with subsets of X.
- The nodes in this tree are indexed by ***pairs*** of integers (h, i); node (h, i) is located at depth $h > 0$ from the root.
- The range of the second index i associated with nodes at depth h is *restricted* by $1 \leq i \leq 2^h$.
- Thus, the root node is denoted by $(0, 1)$.
- By convention, $(h+1, 2i-1)$ and $(h+1, 2i)$ are used to refer to the two children of the node (h, i).
- Let $\mathcal{P}_{h,i} \subset \mathcal{X}$ be the region associated with node (h, i).

The HOO Strategy

- **Cover tree**
  - By assumption, these regions are measurable and must satisfy the *constraints*

$$\mathcal{P}_{0,1} = \mathcal{X}\,, \tag{1}$$

$$\mathcal{P}_{h,i} = \mathcal{P}_{h+1,2i-1} \cup \mathcal{P}_{h,2i}\,, \quad \text{for all } h \geqslant 0 \text{ and } 1 \leqslant i \leqslant 2^h. \tag{2}$$

  - As a corollary, the regions $\mathcal{P}_{h,i}$ at any level ***h > 0*** cover the space ***X***,

$$\mathcal{X} = \bigcup_{i=1}^{2^h} \mathcal{P}_{h,i}\,,$$

explaining the term "tree of coverings"

The HOO Strategy

3.1 Illustrations

- ***Figure 1*** illustrates **correspondence between** the nodes of the tree constructed by the algorithm and their associated regions.
- ***Figure 2*** shows trees built by running HOO for a specific environment.

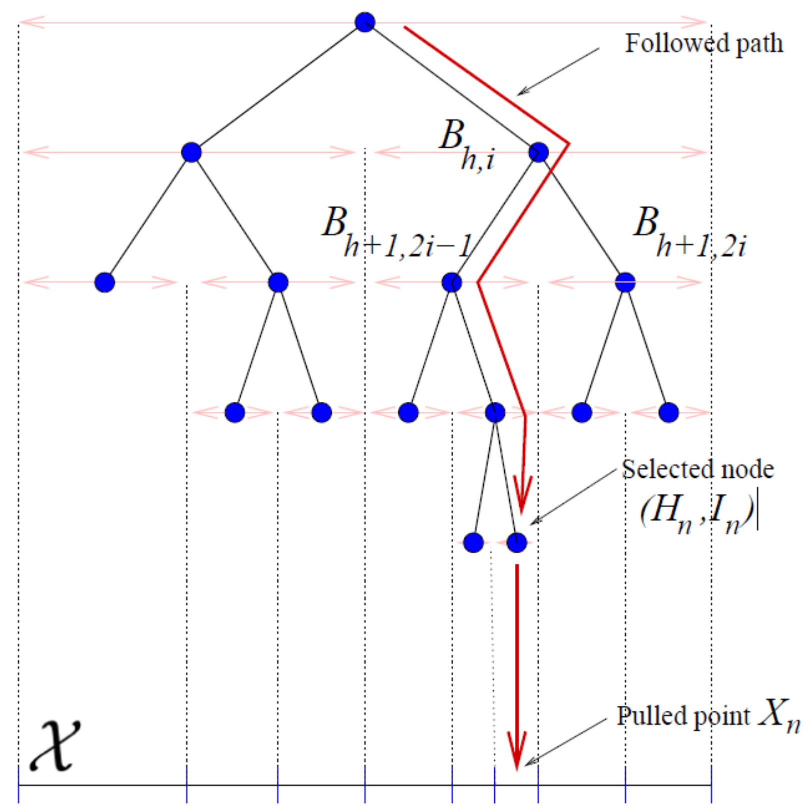


Figure 1: Illustration of the node selection procedure in round n. The tree represents $\mathcal{T}_n$. In the illustration, $B_{h+1,2i-1}(n-1) > B_{h+1,2i}(n-1)$, therefore, the selected path included the node $(h+1,2i-1)$ rather than the node $(h+1,2i)$.

The HOO Strategy- Illustrations

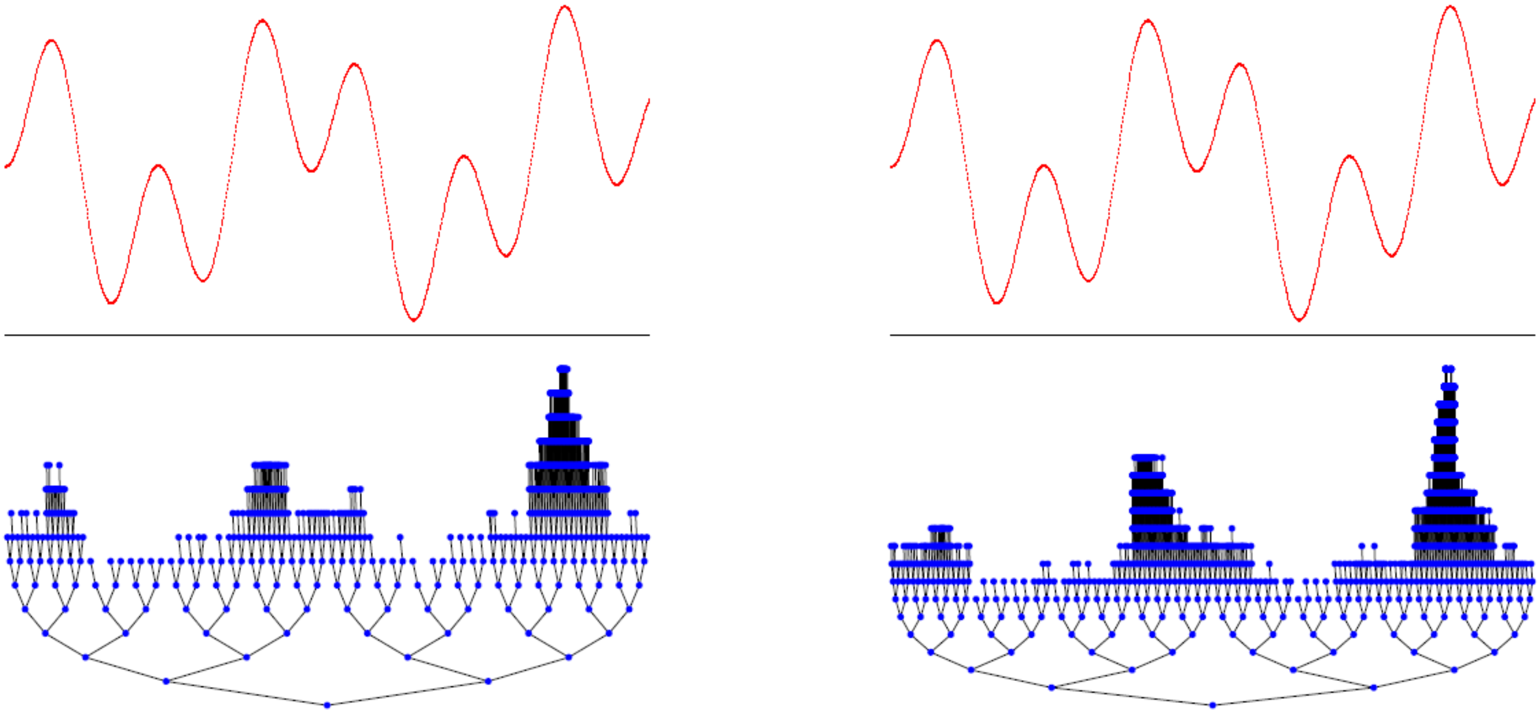


Figure 2: The trees (bottom figures) built by HOO after 1,000 (left) and 10,000 (right) rounds. The mean-payoff function (shown in the top part of the figure) is $x \in [0,1] \longmapsto 1/2\big(\sin(13x)\sin(27x)+1\big)$; the corresponding payoffs are Bernoulli-distributed. The inputs of HOO are as follows: the tree of coverings is formed by all dyadic intervals, $\nu_1 = 1$ and $\rho = 1/2$. The tie-breaking rule is to choose a child at random (as shown in the Algorithm 1), while the points in $\mathcal{X}$ to be played are chosen as the centers of the dyadic intervals. Note that the tree is extensively refined where the mean-payoff function is near-optimal, while it is much less developed in other regions.