工学部 情報工学科

2020年度 情報工学総合演習 レポート

演習題目：課題B
多腕バンディット問題

グループ　G3
学籍番号　18TI018
氏名　ハララノフ・ヴァレリ
演習実施日　自： 2020 年 12 月 04 日
至： 2021 年 01 月 08 日
レポート提出日　2021 年 01 月 12 日
担当教員　重原，内田，山田，松永，島田，菅野

Abstract

多腕バンディット問題とは，限られた資源を複数の (相互排他的) 選択肢に割り当て，その報酬を最大化することを目的とする問題である．多腕バンディット問題の典型的な例は，それぞれ当たる確率が異なるスロットマシンが複数台与えられたときに，報酬が最大となるように遊ぶマシンを選ぶことである．本演習ではこのような状況を仮定し，シミュレーションし，それに対しいくつかの戦略を試行し，それぞれの結果及び性能を評価した．本レポートは以下の構成からなる．まず 節 1 で多腕バンディット問題を定式化し， 節 2 で本演習で用いたそれぞれの戦略に関する簡単な説明をする． 節 3 ではそれぞれの戦略に対し施した実験の設定及び結果を示し，簡単な考察を行う．続いて 節 4 では各戦略のパラメータの最適値を求める実験について説明し，節 5 では一つの戦略に対しその性能が異なる実験環境においてどう変わっていくのかについて考察する．最後に 節 6 で未知のスロットマシンに対し当たる確率が最大となるものを特定する実験について考察し，節 7 で結果をまとめ結論を述べる．

1　多腕バンディット問題の定式化

賭博師がカジノに行って，スロットマシンで儲けようとする状況を想像しよう．カジノには M 台のスロットマシンがあり，かつそれぞれの設定が異なり，当たる確率が異なるとする．通常ならスロットマシンを動かす度にお金がかかり，当たり方も複数パターンあるが，ここで扱うのは無料のスロットマシンで，当たれば \$1 がもらえ，当たらなければ何ももらえないものとする．賭博師は毎晩カジノに行き， N 回マシンを動かす (普通のカジノであれば一週間で自己破綻するだろうが，我々が扱うカジノはなんと無料で遊べるのでその心配はない)．ここではスロットマシンを選んで，それを動かしてみるという作業のことを **試行** と呼ぶ．しかし，各試行においてどのようにスロットマシンを選ぶかも決めなければならない．スロットマシンに 1 から M までの適当な番号を割り当てると， n 回目に選ぶスロットマシンの番号を与える関数

$$f : [0, N] \to [0, M] \tag{1}$$

のことを以降 **戦略** と呼ぶ．

賭博師はカジノに行く前に戦略を決め，遊んでいる間は常にその戦略を用いるが，お酒もかなり飲むため，朝になるとどのマシンを選んだかやどのマシンからどの試行で報酬がもらえたか等が記憶に残らず，どんな戦略を使ってどれぐらい儲かったかしか分からない．賭博師が儲かるためには当たる確率が最も高いスロットマシンを見つけた上，そのマシンだけを動かせば良いが，カジノが毎晩設定を変えるので，毎晩最適なスロットマシンを探さなければならない．したがって賭博師の目標は，スロットマシンの設定を知らない状態からスタートし，一晩で得る報酬を最大にする戦略を見つけることである．

残念ながら，日本では無料のカジノが存在しないのと賭博自体が違法なので，実際にカジノに行って色々実験することができないが，我々は現代技術の最高峰であるコンピュータが使えるので，賭博師がカジノで遊ぶ状況をコンピュータ上に再現して実験を行う．具体的に，賭博師がカジノで一晩遊ぶ (泥酔して帰るまでマシンを N 回動かす) 流れを一つの **シミュレーション** とし，これから説明する 4 つの戦略に対しシミュレーションを実行する．

2　使用した戦略の概要

2.1　Random 戦略

この戦略では毎回等確率でスロットマシンを選ぶ．すなわち，ランダム変数 X が $[0, M] \subset \mathbb{N}$ 上に一様分布している時，戦略が

$$f(n) = X \tag{2}$$

の形を取る． Random 戦略は特に賢い戦略だとはいい難いが，カジノにいる多くの人が恐らくそれを使用しているため，本演習では他の戦略の比較基準として用いる．

2.2　ϵ-greedy 戦略

この戦略ではスロットマシンをランダムに選択することと，過去の情報を根拠に選択することのバランスを取るのが目的である．まず「過去の情報」というのを定式化しよう． n 回目の試行までに k 番目のスロットマシンを選択した回数を $T_k(n)$ とおき，その選択において合計で得られた報酬 (\$1 か \$0) を $R_k(n)$ とおくと， n 回目の試行までの k 番目のスロットマシンの 1 回あたりの平均報酬が

$$\hat{\mu}_k(n) = \frac{R_k(n)}{T_k(n)} \tag{3}$$

として定義できる．必然的に当たる確率が一番高いスロットマシンの平均報酬が一番高くなるので，あらゆる k の中で最大の $\hat{\mu}_k(n)$ に相当するマシンを選べば良いが，$\hat{\mu}_k(n)$ 自体の情報はあらゆるマシンを何回か試してみないと分からない．

したがって，適当にマシンを試す (**探索する**) か，頭を使ってマシンを選ぶ (過去の情報を**活用する**) かを決めるのに何かしらの材料が必要となる． ϵ-greedy 戦略ではこの選択を確率が ϵ のベルヌーイ試行の結果によって行う．すなわち，確率 ϵ のベルヌーイ分布に従う確率変数 E ，及び $[0, M] \subset \mathbb{N}$ 上の一様分布に従う確率 Y が与えられたとき， ϵ-greedy 戦略は以下で与えられる．

$$f(n) = \begin{cases} \underset{k\in[0,M]}{\operatorname{argmax}}\, \hat{\mu}_k(n), & E = 0, \\ Y, & E = 1 \end{cases} \tag{4}$$

2.3 Softmax 戦略

Softmax 戦略は ϵ-greedy 戦略において探索と活用を融合したものだと考えることができる． Softmax 関数 [1] というのは任意の重みから確率分布を生成する関数なので， それぞれのスロットマシンの平均報酬 $\hat{\mu}_k(n)$ を重みとし， n 回目の試行においてランダム変数 X_n が確率質量関数

$$\rho(X_n = k) = \frac{e^{\frac{\hat{\mu}_k(n)}{\tau}}}{\sum_{i=1}^{M} e^{\frac{\hat{\mu}_i(n)}{\tau}}} \tag{5}$$

で与えられる確率分布に従う時， Softmax 戦略は以下で与えられる．

$$f(n) = X_n \tag{6}$$

ただし，τ は温度と呼ばれ，その値を変えるとそれぞれのマシンに相当する確率の格差の度合いを調整することができる [2].

2.4 Upper Confidence Bound (UCB) 戦略

報酬が最も高くなるのは当たる確率が最も高いスロットマシンを選ぶときであるが，どのマシンが最も高いのかが最初分からないというのが多腕バンディット問題の肝である．その情報を得ながらも「賢い」選択をしようとするのが ϵ-greedy 戦略と Softmax 戦略であるが，この 2 つには厄介な問題が存在する．というのは，各マシンを十分に試し，それぞれの当たる確率の推定値 $\hat{\mu}_k(n)$ が十分に正確になった後も探索を続けることである．この現象は **後悔** *(regret)* と呼ばれ，前述の戦略は後悔が比較的大きいと考えられる．したがって後悔を減らし，情報を十分に得終わったら賢い選択のみをしていく戦略が必要であるが，それこそが UCB 戦略である．

UCB 戦略では平均報酬 $\hat{\mu}_k(n)$ に補正項

$$C_k(n) = \sqrt{\frac{\alpha \log_e(n)}{1 + T_k(n)}} \tag{7}$$

を足し，その合計

$$F_k(n) = \hat{\mu}_k(n) + C_k(n) \tag{8}$$

が最も高いマシンを常に選ぶ．これにより戦略は以下のように定義される．

$$f(n) = \underset{k\in[0,M]}{\operatorname{argmax}}\, F_k(n) \tag{9}$$

ただし，ここで $\hat{\mu}_k(0) = 0$ であり，また $F_k(n)$ が複数の k に対し同値になる場合は任意の候補を選択しても一般性が失われないことに注意する．

3 それぞれの戦略に対する実験 (課題 1–4)

本演習では上記で説明した 4 つ戦略 — Random, ϵ-greedy, Softmax, UCB — に対し，以下の条件を設定しシミュレーションによる実験を行った[*1]．まず 5 つのスロットマシンを設定し，それぞれの確率を $p_1 = 0.3$, $p_2 = 0.1$, $p_3 = 0.9$, $p_4 = 0.7$, $p_5 = 0.5$ といた．続いて，各戦略を $N = 500$ 回の試行で一つのシミュレーションを実行し，各試行において以下で定義される平均報酬 $\bar{R}(n)$ 及び平均正答率 $\mathrm{CDR}(n)$ を記録した．

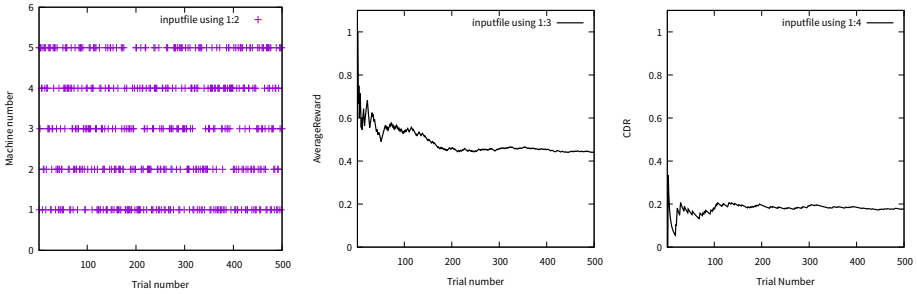
[*1] 本実験では乱数生成にメルセンヌ・ツイスタを用い，シミュレーション等に関するプログラムコードを全て [4] にまとめた．

$$\bar{R}(n) = \frac{1}{n}\sum_{k=1}^{M} R_k(n) \tag{10}$$

$$\mathrm{CDR}(n) = \frac{1}{n} T_\kappa(n) \tag{11}$$

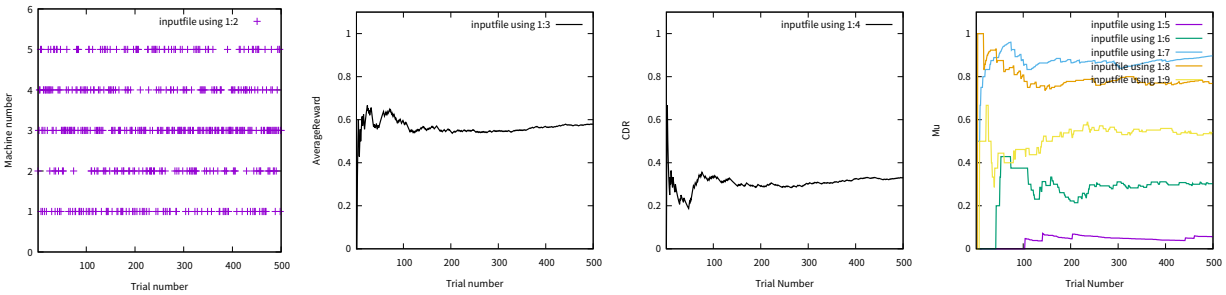
ただし，κ は当たり確率が最も高い (今回においては 3 番目の) マシンを表す．さらに，ϵ-greedy 戦略では各試行でマシンごとの平均報酬 $\hat{\mu}_k(n)$ ， Softmax 戦略では各マシンの選択確率 ρ_k， UCB 戦略では各マシンの評価値 $F_k(n)$ を記録した．また，ϵ-greedy 戦略では $\epsilon = 0.8$， Softmax 戦略では $\tau = 0.5$， UCB 戦略では $\alpha = 1.0$ と設定した．

それぞれの戦略に対するシミュレーション結果を Figs. 1 to 4 に示した．



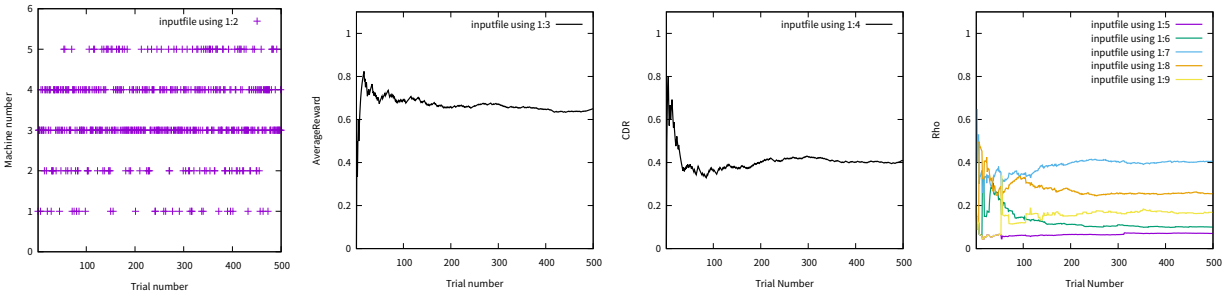
(a) 選択されたマシン　(b) 平均報酬　(c) 平均正答率

図 1: Random 戦略の実験結果



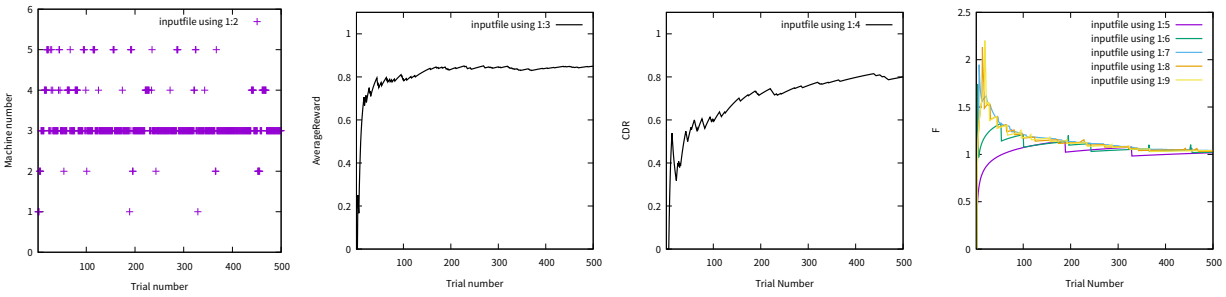
(a) 選択されたマシン　(b) 平均報酬　(c) 平均正答率　(d) 各マシンの平均報酬 $\hat{\mu}_k(n)$

図 2: ϵ-greedy 戦略の実験結果



(a) 選択されたマシン　(b) 平均報酬　(c) 平均正答率　(d) 各マシンの選択確率 $\rho_k(n)$

図 3: Softmax 戦略の実験結果



(a) 選択されたマシン　(b) 平均報酬　(c) 平均正答率　(d) 各マシンの良さ $F_k(n)$

図 4: UCB 戦略に対する結果

以降平均正答率を指標に各戦略を評価する．Random 戦略ではどのマシンも等確率で選択されえるため，平均正答率が想定通りおおよそ $0.2 = \frac{1}{5}$ となる． ϵ-greedy 戦略ではそれが少し改善され，おおよそ 0.3 まで伸びるが，それ以上にはならない．なぜなら，最も当たるマシン ($p_3 = 0.9$) を見つけても， ϵ が定数なため探索が常に行われる．情報の活用は確率 0.8 のベルヌーイ試行なので，活用する時に必ず最適なマシンが選択されると仮定しても選択確率は

$$0.2\epsilon + (1 - \epsilon) = 0.36 \tag{12}$$

となり，実験結果と概ね一致する．続いて， Softmax 戦略では更に正答率が 0.4 台まで伸び，想定通り $\rho_3(n)$ と同じ値となる．ここで温度 τ を変えれば更に高い正答率が得られるが，最初からマシン間の選択確率の格差が広がることから，収束までに時間がかかると考えられる．最後に UCB 戦略に注目すると，平均正答率が 0.8 まで伸びているという驚くほどの改善が見られる．節 2 で説明した通り，補正項の影響が時間とともに減少していくので，最終的には最適なマシンのみが選択されるはずである．本結果では $N = 500$ 回までしか示していないが UCB 戦略に対し $N = 10000$ までの実験をしてみたところ，正答率が確かに 1.0 に収束する現象が観測できた．

4　戦略ごとの最適化及び比較 (課題 5)

前節では各戦略に対し適当なパラメータを指定し実験を行った．本節では様々なパラメータを試し，各戦略の性能が最も向上するパラメータを見つけた上で戦略同士の比較を行う．しかし，その前に試行ごとのランダム性への依存をなくさなければならない．というのは，前節のように実験を行うと n 回目の試行における報酬がスロットマシンのランダム性に依存しており，シミュレーションごとに違う結果が得られる可能性があるため，シミュレーションを複数回行い，シミュレーションに対し n 回目の試行の平均を考えなければならない．したがって，本節では以下のように実験環境とパラメータの評価指標を設定する．

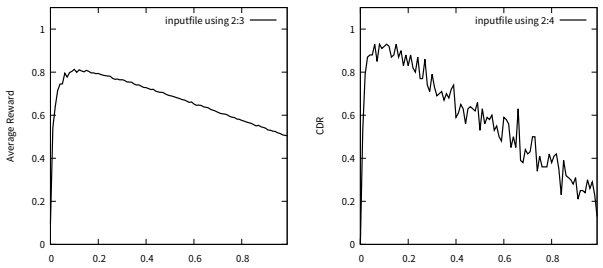
$N = 500$ の試行から成るシミュレーションを $L = 100$ 回行い，これを **正規シミュレーション** と呼ぶ．正規シミュレーション s に対し以下のように正規平均報酬と正規平均正答率を定義する．

$$\bar{R}^{(s)}(n) = \frac{1}{nL} \sum_{l=1}^{L} \sum_{k=1}^{M} R_k^{(s)}(n) \tag{13}$$

$$\mathrm{CDR}^{(s)}(n) = \frac{1}{L} T_\kappa^{(s)}(n) \tag{14}$$

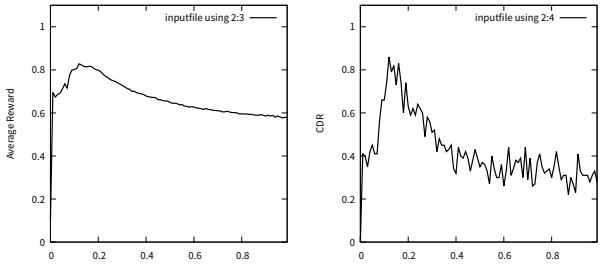
Random を除き各戦略に対し，区間 $P = [0, 1] \subset \mathbb{R}$ から等間隔で $S = 100$ 個の値を取り，戦略のパラメータをその値にした時の正規シミュレーションを実行し，$\bar{R}^{(s)}(N)$ 及び $\mathrm{CDR}^{(s)}(N)$ をパラメータの評価指標として設定する．

上記の方法に基づいて各戦略のパラメータに対する実験結果を Figs. 5 to 7 に示した．



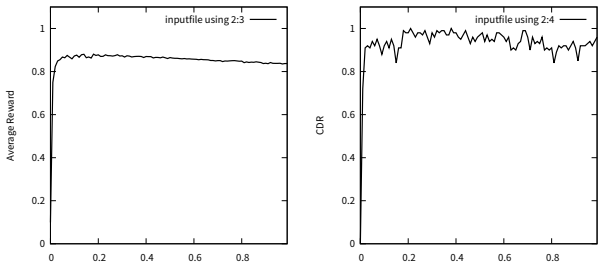
(a) 正規平均報酬 (b) 正規平均正答率

図 5: ϵ-greedy 戦略に対するパラメータ評価指標



(a) 正規平均報酬 (b) 正規平均正答率

図 6: Softmax 戦略に対するパラメータ評価指標



(a) 正規平均報酬 (b) 正規平均正答率

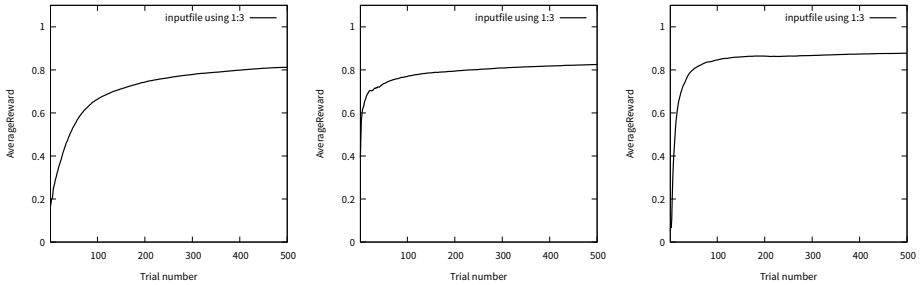
図 7: UCB 戦略に対するパラメータ評価指標

本演習では正規平均報酬が最大となるパラメータ値を最適値とし，各戦略において得られた最適値を表 1 に示した．

表 1: 各戦略パラメータの最適値

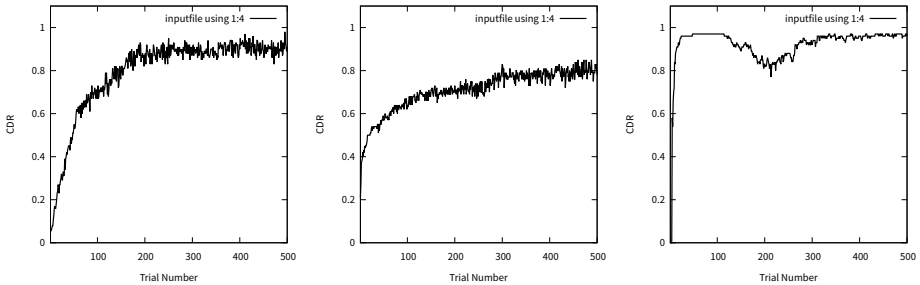
戦略	パラメータ	最適値
ϵ-greedy	ϵ	0.10
Softmax	τ	0.12
UCB	α	0.18

以上の最適値を用いて，それぞれの戦略で改めて正規シミュレーションを行い，試行ごとの正規平均報酬及び正規平均正答率を Figs. 8 and 9 に示した．



(a) ϵ-greedy (b) Softmax (c) UCB

図 8: 各戦略の正規平均報酬



(a) ϵ-greedy (b) Softmax (c) UCB

図 9: 各戦略の正規平均正答率

この結果から全体的に言えることは，どの戦略であっても探索よりも活用に重みをかけるといずれ高い報酬と正答率に収束するが，収束値が最も近いのはやはり UCB 戦略である．その理由は前節で説明した通り， UCB と違って他の戦略は非決定性であり，その期待値も完璧ではなく，最適なマシン以外のマシンも大きな確率で選択されうることからどうしても UCB と比べて性能が落ちてしまうと言える．

5 実験環境の変化に対する評価 (課題 6)

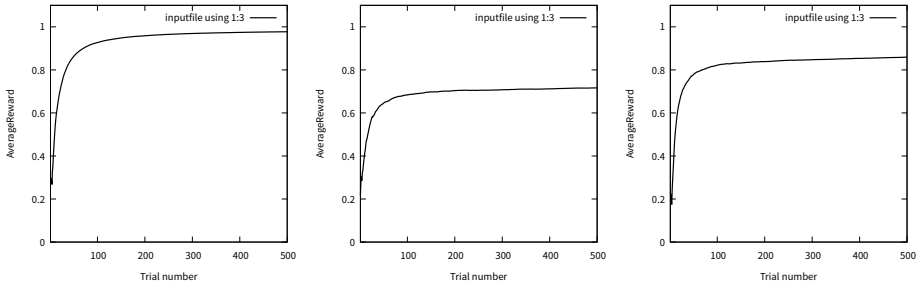
今までの実験ではずっと同じスロットマシンを用いてきた．特に，この設定には一つのマシンが飛び抜けて当たる確率が高く，戦略によらず最終的にそのマシンをずっと選べば最適報酬がもらえるという前提が成り立っていた．しかし，スロットマシンの台数や設定を変えることで，良い戦略でも当然結果が変わる．本節ではこの変化を確かめるために行った実験について説明する．

本演習で扱った戦略の中で一番性能が良い UCB 戦略を選び，前節で求めた最適なパラメータ $\alpha = 0.12$ を用い，表 2 に示した 3 つの異なるスロットマシン設定で正規シミュレーションを行った．

表 2: 調査した 3 つのスロットマシン設定

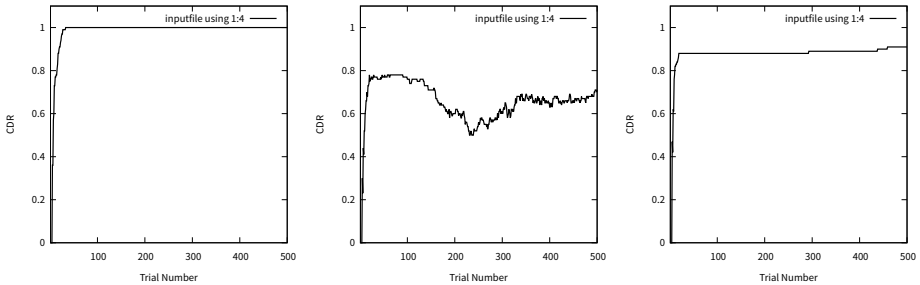
設定	S_1	S_2	S_3
スロットマシンの確率	$p_1^{(1)} = 0.30$	$p_1^{(2)} = 0.30$	$p_1^{(3)} = 0.30$
	$p_2^{(1)} = 0.10$	$p_2^{(2)} = 0.10$	$p_2^{(3)} = 0.30$
	$p_3^{(1)} = 0.99$	$p_3^{(2)} = 0.75$	$p_3^{(3)} = 0.90$
	$p_4^{(1)} = 0.70$	$p_4^{(2)} = 0.70$	$p_4^{(3)} = 0.70$
	$p_5^{(1)} = 0.50$	$p_5^{(2)} = 0.50$	$p_5^{(3)} = 0.50$
			$\vdots$
			$p_{20}^{(3)} = 0.50$

本実験においても正規平均報酬及び正規平均正答率を求め， Figs. 10 and 11 に示した．



(a) S_1 (b) S_2 (c) S_3

図 10: 各設定における正規平均報酬



(a) S_1 (b) S_2 (c) S_3

図 11: 各設定における正規平均正答率

どの設定においても正規平均報酬はいずれ最適値に収束するが，設定 S_2 における正答率 (図 11b) には一見不思議な現象が起きている．平均報酬が最適値に収束しているにもかかわらず正答率が 1 に近づくどころか，かなりのノイズを示しながら 0.6-0.7 台に留まっている．この背景には設定にのマシン 3 と 4 の当たる確率が非常に近い値になっていることが影響していると考えられる． 2 つ (以上の) マシンが近い確率を持っているとき，それぞれの期待値に有意な差が出るのに比較的多くの試行が必要である．というのは，試行が足りないと例えば 4 番目のマシンの期待値が最初に大きく，頻繁に選ばれることになり， 3 番目のマシンがそれに勝つまでには時間がかかる．ここで少し極端な話をすると，仮に $\alpha = 0$ の場合であれば補正項の影響が一切無く，最初に 4 番目のマシンの期待値が勝てば UCB 戦略は完全に騙され，最適である 3 番目のマシンに収束しない．他方，$\alpha = \infty$ であるとすれば，それぞれのマシンの良さが全て等価になり， Random 戦略と同じ結果が得られると考えられる．すなわち， α は UCB 戦略において探索と活用の対比を表すものであり，それを変えてみると上記の現象が改善されると考えられる．

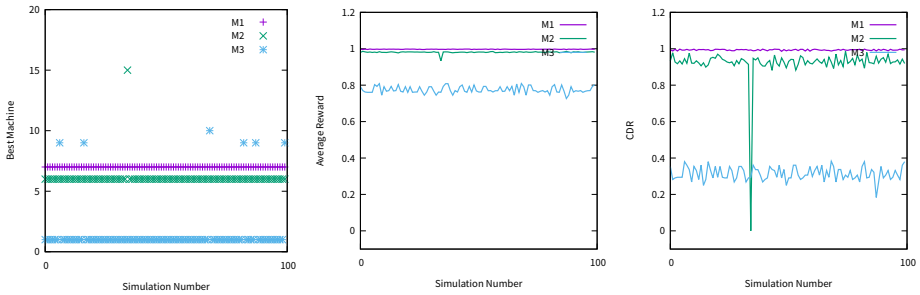
6 未知のスロットマシンから最適なマシンを求める実験

最後に，未知のスロットマシンに対し UCB 戦略を用いて最適なマシンを求めた実験について簡単に述べる．本実験ではライブラリ として実装された 3 パターンのスロットマシン環境に対し各環境において， $\alpha = 0.18$ をパラメータとし UCB 戦略で $N = 10000$ 回のシミュレーションを 100 回行った．なお，このライブラリの都合上平均正答率が得られるのは 1 実行において 1 回のみであるため， 100 回のシミュレーションを実現するのにプログラムを 100 回実行したことに注意する．また，最適なマシンとしては各シミュレーションにおいて平均報酬 $\hat{\mu}_k(n)$ が最大となるマシンを選択した．

それぞれの環境に対し以下の情報が既知だと仮定した．

1. M_1 - マシンが 10 台存在し，その中で一つのマシンの当たる確率が他を大幅に上回る．
2. M_2 - マシンが 20 台存在し，その中で最適なマシンと 2 番目に良いマシンの当たる確率の差が小さい．
3. M_3 - マシンが 10 台存在し，各マシンの当たる確率が時間とともに変わっていく．

この仮定の下で実験を行い，その結果を図 12 に示した．



(a) 最適なマシン．環境 M_1 においては 7 台目， M_2 においては 6 台目， M_3 においては 1 台目が最適となった．

(b) 平均報酬

(c) 平均正答率

図 12: 各環境における実験結果

環境 M_1 では特に面白い結果が現れなく， UCB 戦略が通常通り最適なマシンに収束したと考えられる．環境 M_2 では 1 回のみ収束値から離れ，正答率がほぼ 0 になる現象が発生しているが，平均報酬は僅かに変わる． M_2 では最適なマシンとその次に良いマシンの当たる確率が小さいため，この現象ではランダム性により UCB 戦略が 2 番目に良い戦略に収束していると考えられる．しかし面白いのが M_3 である．この環境でスロットマシンは時間 (厳密には試行回数) につれて当たる確率が変わっていくので，正答率が 0.3 前後になることは容易に想定できる．一方平均報酬を見ると 0.8 前後に収まっていることから，正答率が低くても得られる報酬が高いことが分かる．

ここで UCB が最も多く収束した 1 台目のマシンを考える．まず， シミュレーション内の $N = 10000$ 回のうち，本の数回だけで UCB が収束し，残りの回では常に $a = 1$ 台目のマシンが選択されると仮定する．この仮定は厳密には正確でないが，概ねの流れを十分に簡略化したものだと考えられる． a が最適なマシンのときの当たる確率を x，最適なマシンでないときの確率を y とすると，以下の等式を立てることができる．

$$0.3x + 0.7y = 0.8 \tag{15}$$
$$0.3x = 0.8 - 0.7y \tag{16}$$
$$x = \frac{0.8}{0.3} - \frac{0.7}{0.3}y \tag{17}$$
$$x = 2.66 - 2.33y \tag{18}$$

ここで $0 \leq x, y \leq 1$ という制約が働くので，不等式

$$2.66 - 2.33y < 1 \tag{19}$$
$$1.66 < 2.33y \tag{20}$$
$$0.7 < y \tag{21}$$

を解くと， y の下限が 0.7 程である条件が得られる．つまり，マシンの当たる確率が変わったとしても，その変動が比較的少ないことがわかる．

上述の通りこのモデルは当然厳密なものではなく，他のマシンの影響や a の当たる確率の厳密な変動を非常に簡略化したものであるが，環境 M_3 の概ねの様子を十分に解説できたものだと考えられる．

7　結論

本演習では多腕バンディット問題に対する 3 つの戦略 — ϵ-greedy, Softmax 及び UCB — を用い，問題を解くモデルを実装した．それぞれの戦略の特徴を述べた上でいくつかの実験を行い， UCB 戦略が最も性能が高いことを確認した．また，UCB 戦略に関しスロットマシンの設定が変わると性能がどのように変わるかについて実験し，考察した．最後に未知にのスロットマシンに対し UCB 戦略を施し，様々な環境で UCB がどのような性能を示すかを実験し，それらについて簡単に解説した．

感想

面白かったです．理論的な内容は結構気に入ってやりましたが，やはり実装はめんどくさく，正直なところ Python でやりたかったです (最初に聞いておくべきだったかもしれませんが).

それから，もう少し厳密な定義等があった方が，レポートを書く時に楽だと思います．というのは，多腕バンディット問題を数学的に定式化して，「戦略」というのも定式化して，多腕バンディット問題で何を求めるのが目的なのかをはっきり定めた方が良いと思いました．今回の演習ではなんとなく「報酬が高くなるアルゴリズム」というのが目的なのは分かるのですが，レポートに起こす時には厳密に書きたい (というか書いた方が良いと思う) ので，課題の説明にそういう定義とかも載っていると便利だと思います．

逆に良い点を書くと，課題全体の流れがはっきりしていて，無駄な分岐がなかったのがすごくやりやすかったです．課題 C はともかく， A でも理論を実装してみるというのが目的ですが， A の方は一個一個の課題のつながりがあまりなく，レポートを書くときも論文っぽくするのがすごく気持ち悪かったのですが， B は非常にやりやすかったです．問題を定義して，何個か方法を試して，良い方法を見つけて，最後に実践してみる，という流れを文章に起こすのがとても楽でした．それから，レポート自体の書式 (章立て云々) に自由があるのも (同じく課題 A に比べて) すごく楽でした．こういう比較的重い演習をする時はきちんと論文みたいに色々書きたいので，無理に課題らしい形式を強制しないで，内容を守った上で好きな形式で書かせてくれるのはすごくやりやすいです．馬鹿馬鹿しいかもしれませんがこういう文章も一つ一つ作品だと思って書くので，自慢できる作品が書けると非常にありがたいです．

参考文献

[1] Softmax function, Wikipedia, https://en.wikipedia.org/wiki/Softmax_function (2020-12-30 参照).
[2] Zain Sarwar, Why should we use temperature in softmax, StackOverflow, https://stackoverflow.com/a/63471046 (2020-12-30 参照).
[3] S. Roberts, The Upper Confidence Bound (UCB) Algorithm, https://towardsdatascience.com/the-upper-confidence-bound-ucb-bandit-algorithm-c05c2bf4c13f (2020-12-30 参照).
[4] V. Haralanov, 情報工学総合演習課題 B, https://github.com/Stealthmate/sougou-b .