Vol.2013-MPS-92 No.10
2013/2/27

ロックアップ期間による制約を考慮した確率的バンディット問題

小宮山 純平[1]　佐藤 一誠[1]　中川 裕志[1]

概要：バンディット問題は，複数のアーム（選択肢）から最も報酬の高いものを探す問題であり，探索と活用のトレードオフの代表的なモデルの 1 つである．近年において，情報推薦，最適経路探索，最適化，モデル選択などの分野への応用を動機として，バンディット問題は機械学習やオペレーション・リサーチの分野において注目を浴びている．本稿はロックアップ期間（選択するアームを変更できない期間）の制約を考慮したバンディット問題を提案し，どのような方策を取れば良いかを調べる．既存の多くの有益なアルゴリズムがロックアップ期間を含めた場合に自然に拡張可能であることを示し，その regret（性能）を評価する．この regret がロックアップ期間の最大の大きさに依存することを示す．さらに，ロックアップ期間が大きい場合に regret を減らすことができる Balancing and Recommendation（BaR）メタアルゴリズムを提案する．また，計算機実験の結果を示し，理論的な結果と比較し考察する．

キーワード：バンディット問題，多腕バンディット，逐次最適化，確率的最適化

Multi-armed bandit problem with restricted selections

Junpei Komiyama[1]　Issei Sato[1]　Hiroshi Nakagawa[1]

Abstract: The multi-armed bandit problem, which is widely used to model sequential decision making, has recently attracted much attention, which has led to proposals for a wide range of appricatios, such as recommendation, adaptive routing, optimization, tree search, and model selection. In actual applications, the choice of the forecaster is often restricted. This paper aims to model this restriction. The results of both theoretical analysis and empirical evaluation are presented.

Keywords: Multi-armed Bandit Problem, Sequencial Optimization, Stochastic Optimization

1. はじめに

多腕バンディット問題（バンディット問題）は，逐次最適化問題の最も代表的なモデルの 1 つである．この問題が最初に提唱されたのは 1950 年代まで遡る [1]．近年において，その単純さと応用範囲の広さから再び注目が集まり，研究が加速している．

多腕バンディット問題という奇妙な単語の由来は，英語のバンディットマシン（アームを持つスロットマシン）である．いずれかのアームを選択すると，対応するマシンに応じた確率的な報酬が得られる．複数のアームの中から最良のものを選ぶのが問題の目的である．プレイヤーが全てのアームに関する完全情報を持っているなら，常に期待報酬の最も高いものを選べば良い．しかし，アームの報酬がどうなっているかは最初には分からないので，逐次得られる報酬情報を通じて動的に最適なアームを探ってゆく必要が出てくる．バンディット問題におけるテーマは探索と活用のトレードオフ（*Exploration-Exploitation* tradeoff）である．現在の時点で最も良さそうなアームを活用したいが，一見最適でないアームを探索してゆくと，実はそちらのほうが最適であるという可能性を見逃さないようにできる．近年において，バンディット問題の多くの応用・拡張

[1] 東京大学
The university of Tokyo, Japan

表 1　バンディット問題とその応用例の対応関係
Table 1　Applications of bandit problem

	アーム	報酬
A/B テスト	デザイン	ユーザの反応
診療	医療オプション	患者の術後経過

が提案されている．例としては，モデル比較，凸最適化，多クラス分類問題，モンテカルロ木検索などがあげられるであろう．一方，基本となるバンディット問題そのものも研究の関心として大きい．

バンディット問題では，アームを毎ラウンド自由に選択できることを仮定している．しかし，現実におけるシステムでは，自由にアームを選択できない期間が発生することがしばしば起こり得る．例えば，次のような例を考えてみると良いであろう．

例 1.（A/B テスト）複数のウェブページの新デザイン差分を用意し，どれが最もユーザの反応（リンクのクリック有無など）が良いかを一部のユーザでサンプリングしてサイトの更新の効果測定を行いたい．1 人のユーザに対して 1 つのデザインだけしか見せることはできないという点が，バンディット問題における設定に対応している（表 1）．ウェブコンテンツの動的切り替えには多くの制約が発生する．たとえば，バナー広告は広告契約者との都合から一定期間表示し続ける必要があるし，複数のデザインを切り替えるときには技術的制約からある程度以上の頻度で切り替えできないことが考えられる．

例 2.（診療）診療の問題は，バンディット問題の動機付けとして初期に提案され，近年に至るまで多くの応用がある．考慮される複数の医療オプションの中から，患者の受ける利益（報酬）を最大化させるものを選ぶ問題は，バンディットの設定そのものである（表 1）．多くの場合，医師のオペレーションや医薬品の在庫量などの都合から，同じオプションを一定期間選びつづける必要性があることが考えられるが，このような制約下における最適なオプション選択がしたい．

これらの問題に共通する，バンディットのアームの変更への制限をロックアップという形でモデル化して調べることが本稿の目的である．

本稿の構成は以下のようになる．

- 2 と 3 節において問題の定式化を行う．2 節で確率的バンディット問題について説明した後，3 節にてロックアップ期間による制約を導入する．
- 確率的バンディットにおけるアルゴリズムは，アームの制約が入った場合にどのように振る舞えば良いかが自明ではない．4 節では，多くのアルゴリズムをロックアップによる制約があった場合に自然に拡張する方法を示す．また，拡張されたアルゴリズムの regret が $O(\log(T) + L_{max})$（T はラウンド数，L_{max} は最も長いロックアップ期間）になることを示す．
- 前節で示した regret は L_{max} が小さい場合は最適だが，$L_{max} > \log T$ では前者の項が問題となる．5 節では，L_{max} が大きい場合に有効なメタアルゴリズムを提案する．
- 6 節では，これらの成果を数値的に検証するために行った計算機実験の結果について報告する．
- 最後に，7 節で本稿の内容についてまとめる．

初期条件：アームの数 K，ラウンド数 T
各ラウンド $t \in \{1, ..., T\}$ において，
(1) 予測者はアーム I_t を選択する
(2) 予測者は報酬 $X(t) \sim \nu_{I_t}$ を受け取る

図 1　確率的バンディット問題
Fig. 1　Stochastic bandit problem

2. バンディット問題

本稿では確率的バンディット問題を扱う．確率的バンディット問題では，報酬が一定の確率分布から選ばれるものと仮定する．アームの数を K，ラウンド数を T とする．各アームは $[0, 1]$ に値を取る定常の報酬確率分布 $\{\nu_1, ..., \nu_K\}$ を持つと仮定する[*1]．

バンディット問題の枠組みを図 1 に示す．各ラウンド t において，予測者はアーム I_t を選択し，そのアームの確率分布から独立に引かれる報酬 $X(t)$ を受け取る．予測者の目的は累積報酬の最大化であり，これがどの程度達成できているかは，以下で定義される regret（累積 regret）の期待値によって評価される．

$$\mathbf{R}[T] = \mu_* T - \sum_{i=1}^{K} \mu_i T_i(T). \qquad (1)$$

ここで，μ_i はアーム i の報酬の期待値（$= \mathbb{E}[\nu_i]$）であり，$T_i(T)$ はラウンド T までにアーム i が選択された回数である．i^* を最も報酬の期待値が高いアーム，そして $\mu_* = \mu_{i^*}$ と定義する．また，i^* を最適なアーム，それ以外のアームを非最適なアームと呼ぶことにする．最適なアームの期待報酬と，アーム i の期待報酬の差を $\Delta_i = \mu_* - \mu_i$ と表記すると，regret はアルゴリズムが非最適なアーム i を選んだときに Δ_i だけ上昇する．この regret を最小化することがアルゴリズムにとっての目標となる．

2.1 バンディット問題のアルゴリズム

確率的バンディット問題において最も知られているアルゴリズムは，おそらく UCB [2] であろう．UCB アルゴリズムに従う予測者は，各ラウンド $t \in \{1, ..., T\}$ において，次の信頼上界（Upper Confidence Bound）を最大化するアームを選択する

[*1] $[0, 1]$ での結果は $[a, b]$（$a, b \in \mathcal{R}$）へ容易に拡張可能である．

$$\hat{X}_i(t) + \sqrt{\frac{a \log t}{T_i(t)}} \qquad (2)$$

ここで，$\hat{X}_i(t)$ はラウンド t 開始時の経験期待値（それまでにアーム i を引いたときに得た報酬の平均）であり，a はパラメータ定数である．

探索と活用の言葉で述べると，最初の項（経験期待値）は活用，2番目の項は未知の真の期待値と経験期待値のずれの上界を見積もるという意味で探索に対応している．パラメータ a がこの探索と活用のバランスを決めており，a が大きくなればなるほど探索の割合が大きくなる．活用を大きく取ると非最適なアームを探索するラウンド数を減らすことができるが，低確率で最適なアームの経験期待値が低い事象が起きることがあり，その場合に非最適なアームを最適であると誤認してしまうリスクがある．一方，探索を大きくとると非最適なアームを多く探索しなければいけないが，真の最適なアームが実際に経験期待値が最大であるリスクを減らすことができる．これがバンディット問題における探索と活用のトレードオフである．本稿では UCB-E [3]，KL-UCB [4], MOSS [5]，UCB-Tuned [2] などのアルゴリズムを実験的に比較したが，これらは UCB と同じく信頼上界を利用している．

一方で，確率的な選択を入れた ϵ_n-greedy アルゴリズムについても紹介する．ϵ_n-greedy アルゴリズムに従う予測者は，各ラウンド $t \in \{1, ..., T\}$ において，確率 $\epsilon_t = \min\{1, cK/d^2t\}$[*2]でランダムにすべてのアームを等確率で選択し，確率 $1 - \epsilon_t$ で，これまで選択されたアームの中で経験期待値の最も高いアーム $\arg\max_i \hat{X}_i(t)$ を選択する．

これらのアルゴリズムは regret の期待値が $O(\log T)$ であり，これは定数倍のオーダーまで最適である．$\sum 1/t = log(t)$ であるので，1 ラウンドあたりの非最適アームの探索は最低でも $1/t$ オーダーで必要である．ロックアップ制約の導入によってアームが自由に選択できなくなったとき，この探索と活用のバランスをどの程度保てるかが本稿のテーマである．

2.2 先行研究

本稿の提案するロックアップ制約のある確率的バンディット問題を説明する前に，確率的バンディット問題における制約を扱った先行研究を概説する．最も多く研究されてきたモデルは，スイッチコストを持つバンディット問題である．これはアームの切り替えに一定のコストがかかる状況下で，アーム選択の変更回数を抑えた最適なアルゴリズムを探すものである．より詳細には，[6] などを参照されたい．コミットのあるバンディット問題 [7] は，ある期間まで自由にアームを選択をすることができ，探索が十分だと予測者が判断したら，その時点であるアームにコミットするというものである．予測者の regret は探索の期間の長さと，コミットしたアームが最適なアームかどうかに依存する．

スイッチコストのあるバンディット問題の場合はスイッチコストを払うことによってアームを変更できるし，コミットのあるバンディット問題の場合は探索期間を予測者の判断で伸ばすことができる．つまり，制約期間を含めた最適化の問題である．一方，本稿のロックアップ制約は，外部からアームを変更できない期間という制約が与えられ，ロックアップ中はアームを変更できないため，制約が与えられた条件での最適化の問題である．

[*2] c と d はアルゴリズムのパラメータである．

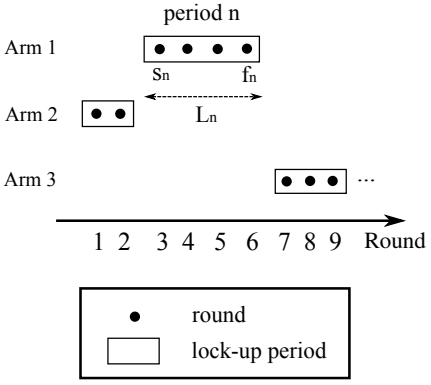

図 2 ロックアップ制約．黒い点がラウンド，四角がロックアップ期間を表す．

Fig. 2 Lock-up bandit. A black dots represent rounds and rectangles represent lock-up periods.

初期条件：アームの数 K，ラウンド数 T，ロックアップ期間 $\{L_1, ..., L_N\}$
各ラウンド $t \in \{1, ..., T\}$ において，
(1) そのラウンドがいずれかのロックアップ期間の開始ラウンドなら（$\exists n s_n = t$），予測者はアーム I_t を選択する．そうでないなら，予測者は直前のラウンドに選択したものと同じアーム I_t を選択する．
(2) 予測者は報酬 $X(t) \sim \nu_{I_t}$ を受け取る

図 3 ロックアップによる制約のあるバンディット問題

Fig. 3 Bandit problem with lock-up periods

3. ロックアップ期間のあるバンディット問題

節 2 で定式化した確率的バンディット問題に，ロックアップによる制約を導入する（図 2）．ラウンド $\{1, ..., T\}$ を，ロックアップ期間 $\{L_1, ..., L_N\}$ に分割する．各期間の初めと終わりのラウンドを $\{(s_1, f_1), ..., (s_N, f_N)\}$ と表記する．ラウンドを分割しているため，$s_1 = 1$，$f_n + 1 = s_{n+1}$，$f_N = T$ である．ロックアップによる制約を考慮したバンディット問題は図 3 のように進行する．ロックアップ制約のあるバンディット問題では，各ロックアップ期間の最初のみアームを選択することができ，ロックアップ期間中は前のラウンドと同じアームを選択しなければならない．

regret については通常のバンディット問題と同様である．

3.1 ラウンドとロックアップ期間の表記について

本稿では，ラウンド数を表す添字として $t \in \{1, ..., T\}$，ロックアップ期間を表す添字として $n \in \{1, ..., N\}$ を使用する．また，添字 $i \in \{1, ..., K\}$ をアームの番号を表すために使用する．予測者は各ロックアップ期間 n の最初にアームを選択し，期間中は単一のアームを選び続ける．この単一のアームを I_n と定義することができる．アーム i がラウンド T まで（ロックアップ期間 $L_1, ..., L_N$ に選択された回数 T_i は以下のように表すことができる．

$$T_i(L_1, ..., L_N) = \sum_{n=1}^{N} L_n \mathbb{I}_{I_n=i}. \tag{3}$$

ここで，$\mathbb{I}_{\mathcal{E}}$ は事象 $\mathcal{E}$ が真なら 1，偽なら 0 である指示関数である．

4. 既存のアルゴリズムのロックアップを考慮した拡張

小節 2.1 において，確率的バンディット問題のアルゴリズムを説明した．これらのアルゴリズムは毎ラウンド自由にアームを選択できることを前提としており，アームの選択に制限が加えられたときにどのように振る舞えばいいかが自明ではない．本節では，UCB や ϵ_n-greedy をはじめとした多くのアルゴリズムの，ロックアップ制約があるバンディット問題への自然な拡張を紹介する．これらのアルゴリズムは，

1. 毎ラウンド，アームごとの目的関数を最大化するアームを選択する．ここで，
2. 目的関数はそれぞれのアーム i を選択した数 $T_i(t)$，経験期待値 $\hat{X}_i(t)$，経験分散 $\hat{V}_i(t)$ など，アームごとに独立した量の関数である

という特徴を持つ．ロックアップを考慮した場合は，アームを選択できないラウンドが存在するが，これらのアルゴリズムは以下のようにして自然に拡張できる．各ラウンドにおいて，アームを選択できる場合，これまでと同様にアームごとの目的関数を最大化するアームを選択し，アームは選択できない場合もアームを選択できた場合と同様に処理する（i.e. アーム選択数，経験期待値などの計算を更新する．）．この方法によってロックアップ制約のあるバンディット問題に拡張されたアルゴリズムを，′ を末尾につけて記述する．（例：UCB → UCB′）

定理 4.1. （UCB′ の regret 上界）ロックアップ制約ありのバンディット問題において，探索の大きさを決めるパラメータを $a = 2$ に設定した *UCB′* に従う予測者の *regret* の期待値について，以下が成立する．

初期条件：アームの数 K，ロックアップ期間 $\{L_1, ..., L_N\}$, 推薦セットの数 $0 \leq N_r < N$, ベースアルゴリズム $\mathcal{A}$
各ロックアップ期間 $n \in \{1, ..., N\}$ の開始時に，
(1) ロックアップ期間 n が推薦セットに入っている（$= n \in \{(1), ..., (N_r)\}$）場合，$\mathcal{A}$ から推薦アームを受け取り，それをこの期間の選択アーム I_n とする．この期間における報酬は $\mathcal{A}$ にフィードバックしない．
(2) そうでない場合，$\mathcal{A}$ にアームを選択させ，それをこの期間の選択アーム I_n とする．この期間における報酬を毎ラウンド $\mathcal{A}$ にフィードバックする．

図 4 BaR メタアルゴリズム
Fig. 4 BaR meta-algorithm

$$\mathbb{E}[\mathbf{R}(L_1, ..., L_N)] \leq \sum_{i \neq i^*} \left\{ \frac{8 \log T}{\Delta_i} + L_{max} \Delta_i \left(1 + \frac{\pi^2}{3}\right) \right\}. \tag{4}$$

つまり，もとのアルゴリズムの探索と活用のバランス（$O(\log T)$）が，L_{max} に比例した程度崩れることになる．

5. BaR メタアルゴリズム

5.1 ロックアップ制約と regret

ロックアップ期間 n の開始時に非最適なアーム i を選択してしまったとすると，そのアームを L_n 回選びつづけることになるため，regret は $\Delta_i L_n$ 増加することになる．ベースとなるアルゴリズムが探索と活用のバランスを最適にしていても，ロックアップ期間で急激にバランスが崩れてしまうことなる．この理由から，非最適なアームを長いロックアップ期間の始めに選択するのを防止したい．特定のラウンドでの regret を最小化する方法として，本稿では推薦アーム（経験期待値最大のアーム）を使用する．推薦アームの 1 ラウンドの regret を単純 regret と定義する[*3]．

5.2 BaR メタアルゴリズム

今回提案する BaR（図 4）は，大きいロックアップ期間において推薦アームを利用し，それ以外の期間についてはベースとなるバンディットアルゴリズムを実行する．

[BaR, $\mathcal{A}$] の regret は，$\mathcal{A}$ の単純 regret と累積 regret を用いて次のように表せる．

$$\mathbf{R}(L_1, ..., L_N) = \mathbf{R}_{base}(L_1, ..., L_N \setminus L_{(1)}, ..., L_{(N_r)}) + \sum_{n=1}^{N_r} L_{(n)} \mathbf{r}_{base}([L_{n'} | n' < (n), L_{n'} \notin \{L_{(1)}, ..., L_{(N_r)}\}]). \tag{5}$$

ここで，累積 regret 部分（$\mathbf{R}_{base}$）はベースアルゴリズム $\mathcal{A}$ を，現在のロックアップ制約の問題から推薦セットを除いたバンディット問題で走らせた問題の regret である．また，単純 regret 部分（$\mathbf{r}_{base}$）は，推薦セットのそれぞれ

[*3] 紙面の都合で詳細を省くが，単純 regret に対する詳細は [8] を参照されたい．

に対してのベースアルゴリズムの単純 regret の和である．ただし，ここでの単純 regret は累積 regret と同様に推薦セットを除いた環境での和である．例として以下の設定を考える．$T = 100$，推薦セットを $\{(1), (2)\} = \{50, 100\}$ とする．この場合は累積 regret 部分はベースアルゴリズム $\mathcal{A}$ をロックアップ期間 $L_1, ..., L_{49}, L_{51}, ..., L_{99}$ （L_{50}, L_{100} を除いたもの）で走らせたときの regret であり，単純 regret 部分は，ベースアルゴリズムの L_{50} の $L_1, ..., L_{49}$ の後での単純 regret と L_{100} の $L_1, ..., L_{49}, L_{51}, ..., L_{99}$ の後での単純 regret の和である．

このように，累積 regret 部分から任意のロックアップ期間を取り除くことができ[*4]，また取り除いたロックアップ期間では最適なアームを高確率で選択するため，大きいロックアップ期間で regret が増加するリスクを大きく減らすことができる．単純 regret については研究が比較的新しいものであり未知の部分が多いが，あるラウンドでの単純 regret はそれまでのラウンドでの累積 regret の量に対して多項式オーダで減少することが知られている [8]．

6. 計算機実験

提案したモデルおよびアルゴリズムについて経験的に解析するため，2 セットの計算機実験を行った．

- 1 つ目の実験セット（図 5（実験 1 と 2））は，節 4 で提案された従来のアルゴリズムの自然な拡張における，regret とロックアップ制約の大きさの関係を調べた．
- 2 つ目の実験セット（図 5（実験 3 と 4））は，節 5 で提案されたメタアルゴリズム（BaR）についての事前・事後分析を行った．

複数のアルゴリズムの間の優劣を比較することは今回の目的ではない．ϵ_n-greedy は今回の実験において最も regret が少なかったが，これは経験的に良いパラメータを利用したためである[*5]．

6.1 実験設定

実験のすべてのアームの報酬分布は全てベルヌーイ分布とした．すべての実験は 10 アームのバンディット問題であり，それぞれのアームの期待値は $\{\mu_1, ..., \mu_{10}\} = (0.1, 0.05, 0.05, 0.05, 0.02, 0.02, 0.02, 0.01, 0.01, 0.01)$ とした. また，ラウンド数 $T = 10000$ とした．

ベースとなるアルゴリズムは UCB-E [2]（パラメータ $a = 2\log T, 1/2\log T$），ϵ_n-greedy [2]（パラメータ $(c, d) = (0.15, 0.1)$）, KL-UCB [4]（パラメータ $c = 0$）, MOSS [5] と UCB-Tuned [2]（参考論文と同様設定）である．

最初の実験セット（実験 1 と 2）では，ロックアップ期間の最大サイズ S[*6] と regret の関係を見た．すべての実験の各 S の値において，regret の値は 10000 回の試行の平均値である．それぞれの試行において，ロックアップ期間は以下のようにランダムに生成した．すべてのロックアップ期間の合計ラウンド数が T に到達するまで，新しいロックアップ期間を追加していく．新しいロックアップ期間はサイズ $\{1, ..., S\}$ の中から，(1) 等確率で（実験 1）(2) サイズに反比例した確率で（実験 2）生成した．全体の期間のサイズが T となるように，最後のロックアップ期間のサイズを調整した．

2 つ目の実験セット（実験 3 と 4）では，ラウンドと regret の関係を調べた．設定は同一であり，実験 3 と 4 で異なるアルゴリズム（UCB-E′ と ϵ_n-greedy′）を調べた．UCB-E は経験的にほぼ UCB と同じ挙動を示すアルゴリズムであり，UCB の累積 regret の保証に加えて単純 regret にいくらかの保証があるため，今回利用した．Regret の値は 10000 回の試行の平均値である．ロックアップ期間は，試行ごとに以下の手続きでランダムに生成された．最初の 2000 ラウンドは，ロックアップを設定しない (i.e. $L_1, ..., L_{2000} = 1$). ラウンド 2001 から 10000 までは，全ラウンド数が 10000 に達するまで，サイズ $\{1, ..., 1000\}$ のいずれかのロックアップ期間をサイズに反比例する確率で生成する．BaR で推薦セットに含めるロックアップ期間は，長さ 400 以上の全ての期間とした．さらに，ロックアップ期間がどの程度探索と活用のバランスを妨げているかを見るために，ロックアップ期間のない通常の確率的バンディット問題の regret も比較した．

6.2 実験結果と考察

1 つ目の実験セット（実験 1 と 2）の結果を見ると，すべてのアルゴリズムにおいて，最大ロックアップ期間サイズ S と regret が比例しているのが分かる．ここで，実験 1 と 2 では，サイズの大きいロックアップ期間の割合が大きく異なる[*7]にも関わらず，2 つの実験の図は良い類似を見せている．これは，ロックアップによる制約がその最大サイズに依存することを示している．2 つ目の実験セット（実験 3 と 4）では BaR メタアルゴリズムの効果を検証した．両方の元アルゴリズムにおいて，BaR を利用することによって有意に regret を減少させられたことが分かる[*8]．ラウンド数が多いと探索の割合は少ない（$O(\log T/T)$ オーダー）ため，探索側に大きく振れてしまったときの修復は

[*4] L_{max} を減らすことができる
[*5] ϵ_n-greedy は regret の理論的な保障の範囲が限定されていることや，パラメータの事前推定が困難である問題があり，実用的に広く使われているのは UCB のような信頼上界を利用するアルゴリズムのほうである．
[*6] S はここに述べる手続きで生成されうるロックアップ期間の最大サイズであり，実際の試行ごとのロックアップ期間の最大サイズは $L_{max} \leq S$ である
[*7] 実験 1 では，$\{1, ..., S\}$ の全ての大きさのロックアップ期間が均等に生成されるが，実験 2 ではサイズに反比例して大きいロックアップ期間が生成されにくくなる
[*8] 紙面の問題で省いたが，他のアルゴリズム（KL-UCB，MOSS，UCB-Tuned）でも同様に regret を減少させることができた．

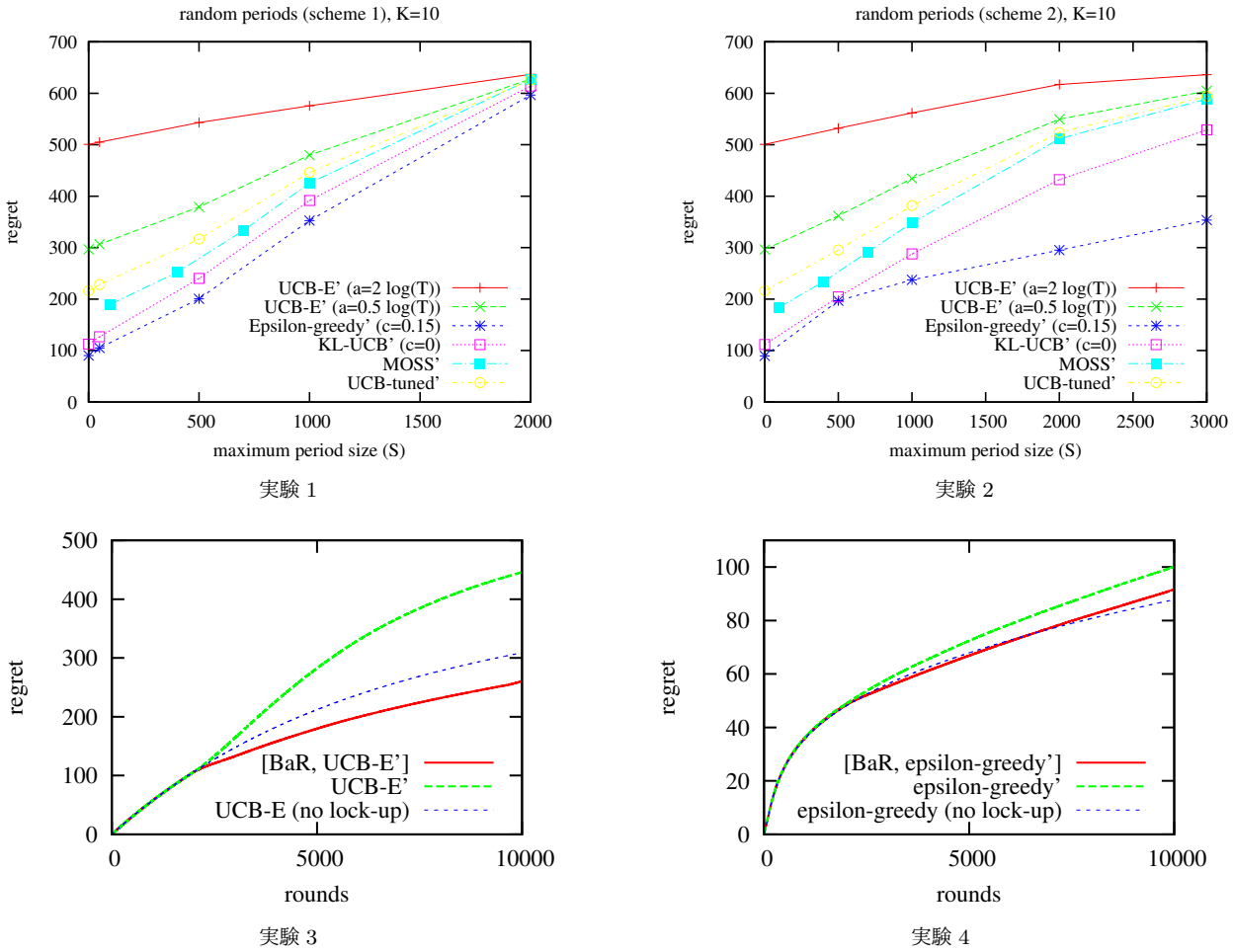

図 5 計算機実験の結果．実験セット 1（実験 1,2）はランダムに生成されるロックアップ期間の最大サイズ S と regret の関係を，実験セット 2（実験 3,4）はラウンド数と regret の関係について，BaR 適用前後の比較を示す．

Fig. 5 Experimental results. Experiments 1 and 2 show regret as function of maximum period size. Experiments 3 and 4 show regret before/after application of BaR.

難しい．逆に活用がやや過剰になってしまっても，その後のラウンドで探索を増やしてバランスすることができる．そのため大きいロックアップ期間で活用を増やすことはかなり有効に働く．これが BaR が regret を減らすことができた理由だと考えられる．ベースアルゴリズムの regret が多い UCB-E′ のほうが ϵ_n-greedy′ より探索が過剰なため，BaR による regret 減少が大きい．

7. おわりに

本稿では，バンディット問題における外部からの制約のモデルを提案した．良いアルゴリズムは探索と活用を最適にバランスしているが，外部からの制限によってこのバランスが崩れるときに，どのように最適な状態から離れないようにすれば良いかを提案した．自由アームを選べないような制約が入った場合でも，制約が小さければこれらのアルゴリズムは十分使えると言えるであろう．

参考文献

[1] Robbins, H.: Some aspects of the sequential design of experiments, *Bulletin of the AMS*, Vol. 58, pp. 527–535 (1952).
[2] Auer, P., Cesa-bianchi, N. and Fischer, P.: Finite-time Analysis of the Multiarmed Bandit Problem, *Machine Learning*, Vol. 47, pp. 235–256 (2002).
[3] Audibert, J.-Y., Bubeck, S. and Munos, R.: Best Arm Identification in Multi-Armed Bandits, *COLT* (2010).
[4] Garivier, A. and Cappé, O.: The KL-UCB Algorithm for Bounded Stochastic Bandits and Beyond, *Computing Research Repository* (2011).
[5] Audibert, J.-Y. and Bubeck, S.: Minimax policies for adversarial and stochastic bandits, *COLT* (2009).
[6] Jun, T.: A survey on the bandit problem with switching costs, *De Economist*, Vol. 152, No. 4, pp. 513–541 (2004).
[7] Bui, L., Johari, R. and Mannor, S.: Committing Bandits, *NIPS*, pp. 1557–1565 (2011).
[8] Bubeck, S., Munos, R. and Stoltz, G.: Pure Exploration for Multi-Armed Bandit Problems, *Computing Research Repository*, Vol. abs/0802.2 (2008).