1F3-3

予算制限バンディットアルゴリズム LAKUBE の探索率設定方法の提案

新美 真 *1　伊藤 孝行 *2
Niimi Makoto　Ito Takayuki

*1名古屋工業大学工学部情報工学科
Department of Computer Science, Nagoya Institute of Technology

*2名古屋工業大学大学院産業戦略工学専攻
School of Techno-Business Administration, Nagoya Institute of Technology

We focus on the budget-limited multi-armed bandit(BL-MAB) problems. In BL-MAB problems, the agent's actions are costly and constrained by a fixed budget. LAKUBE is BL-MAB algorithm for highly budget-constrained situation. LAKUBE has parameter K_α that limits the number of arms of exploration. But, K_α need to set optimal value. We propose new BL-MAB algorithm seKUBE. seKUBE decides to the number of arms of exploration. In our experiments, we compared the existing bandit algorithm with our proposed bandit algorithm.

1. はじめに

多腕バンディット（Multi-armed bandit, 以降 MAB）問題とは，複数台あるスロットマシン（以降アームと呼ぶ）をプレイするギャンブラーを模した問題である．アームから得られる報酬は，それぞれ独立で適当な確率分布に従うと仮定する．本研究では MAB 問題の拡張の一つである予算制限多腕バンディット（Budget-limited Multi-armed-bandit, 以降 BL-MAB）問題を取り扱う．BL-MAB 問題の制約としてコスト及び予算が存在する．アームをプレイする時にコスト分の予算を消費しなければならない．

本研究では，BL-MAB 問題のアルゴリズムの一つである LAKUBE[1] に注目する．LAKUBE は，KUBE[2] を発展させたアルゴリズムで，予算が小さい条件下において良い性能を発揮する．LAKUBE は，長期間のデータ収集を目的としたワイヤレスセンサネットワークへの応用が期待されている [5][6][7].

LAKUBE アルゴリズムでは探索するアーム数を，パラメータ K_α によって制限し，その後活用を行う．探索とはアームをプレイし得られる報酬の大きいアームを推定すること，活用とは探索で得られた情報をもとにアームをプレイすることをいう．既存研究ではパラメータ K_α の設定方法については課題としており，予算ごとに適切な値を与えなければならない．本研究では，探索を最適停止問題とみなすことで探索するアーム数を決定する手法 seKUBE を提案する．

本論文の構成を以下に示す．第 2 章は既存の BLMAB 問題のアルゴリズムについて述べる．第 3 章では提案手法である seKUBE について言及する．第 4 章では実験設定及び結果について述べる．最後に本論文をまとめる．

2. 予算制限バンディットアルゴリズム

2.1 KUBE

Knapsack based Upper confidence Bound exploration and Exploitation（以降 KUBE と呼ぶ）は，活用と同時に探索を行う予算制限バンディットアルゴリズムである [2] ．KUBE を Algorithm1 に記述する．各行の先頭の数字は step を表す．

Algorithm 1 The KUBE Algorithm

1: $t = 1; B_t = B$;
2: **while** pulling is feasible **do**
3: **if** $B_t < \min_i c_i$ **then**
4: STOP { pulling is not feasible}
5: **end if**
6: **if** $t \leq K$ **then**
7: Initial phase: play arm $i(t) = t$;
8: **else**
9: use density-ordered greedy to calculate $M^*(B_t) = \{m^*_{i,t}\}$, the solution of Equation 1;
10: randomly pull $i(t)$ with $P(i(t) = i) = \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}}$;
11: **end if**
12: update the estimated upper bound of arm $i(t)$;
13: $B_{t+1} = B_t - c_{i(t)}; t = t + 1$;
14: **end while**

KUBE はそれぞれのタイムステップ t で，アームがプレイ可能かどうかを確認する (steps 3-4)．もしアームがプレイ可能である場合，KUBE は探索時にすべてのアームを一度だけプレイする (steps 6-7)．活用時にプレイされるアームは式 (1) を満たすアームである．

$$\max \sum_{i=1}^{K} m_{i,t} \left(\hat{\mu}_{i,n_{i,t}} + \sqrt{\frac{2 \ln t}{n_{i,t}}} \right)$$
$$\text{s.t.} \quad \sum_{i=1}^{K} m_{i,t} c_i \leq B_t, \forall i, t : m_{i,t} \text{integer} \qquad (1)$$

ここで，$m_{i,t}$ は式 (1) を満たすアームのプレイ回数，$\hat{\mu}_{i,n_i}$ はアーム i がプレイして得られた報酬の平均から求められた推定報酬，及び $n_{i,t}$ はタイムステップ t までにアーム i をプレイした回数を表す．

連絡先: 伊藤孝行，名古屋工業大学，愛知県名古屋市昭和区御器所町，052-735-7407，ito.takayuki@nitech.ac.jp

エージェントの目標は残りの予算 B_t に対応した式 (1) を満たす定数 $\{m_{i,t}\}_{i\in K}$ を見つけることである．ここで K はアーム数を表す．本問題は NP-hard であるため，貪欲法を用いてアームの組み合わせを求めている．アーム i の期待報酬密度に基づく評価値は式 $\frac{\hat{\mu}_{i,n_{i,t}}}{c_i} + \frac{\sqrt{\frac{2\ln t}{n_{i,t}}}}{c_i}$ より求められる．ここで KUBE の式 (1) を満たす最大となるアームの組み合わせの解を $M^*(B_t) = \{m^*_{i,t}\}$ とする．$\{m^*_{i,t}\}$ を用いて KUBE はランダムにプレイするアームを選択する．プレイされるアームの確率は式 $P(i(t)=i) = \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}}$ に従う．プレイされた後，選択されたアームの評価値と残りの予算 B_t を更新する（steps 12-13）．

2.2 LAKUBE

LAKUBE は予算が小さいときに良い性能を発揮する予算制限バンディットアルゴリズムである [1]．LAKUBE を Algorithm2 に記述する．

Algorithm 2 The LAKUBE Algorithm

1: $t=1; B_t = B; 1 \le K_\alpha \le K$;
2: **while** pulling is feasible **do**
3: **if** $B_t < \min_i c_i$ **then**
4: STOP { pulling is not feasible}
5: **end if**
6: **if** $t \le K_\alpha$ **then**
7: Initial phase: play arm $i(t) = t$;
8: **else**
9: use density-ordered greedy to calculate $M^*(B_t) = \{m^*_{i,t}\}$, the solution of Equation 1;
10: randomly pull $i(t)$ with $P(i(t)=i) = \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}}$;
11: **end if**
12: update the estimated upper bound of arm $i(t)$;
13: $B_{t+1} = B_t - c_{i(t)}; t = t+1$;
14: **end while**

LAKUBE では，事前に探索するアーム数 K_α を与えることで，探索時に実行するアーム数を制限する．K_α の値の範囲は，$1 \le K_\alpha \le K$ である．最適な K_α を与えることで，予算が小さい場合であってもアームを活用することができる．しかし，その値を決定する方法は今後の課題となっている．活用時は探索時にプレイされたアームについて KUBE と同様に選択する．

3. 提案手法

seKUBE

seKUBE は，探索を最適停止問題ととらえることで探索時に実行するアーム数を減らし，予算が小さいときにも損失が小さくなることを目標としたアルゴリズムである．seKUBE を Algorithm3 に記述する．

F, K_s, $r(t)$ はそれぞれ探索フラグ，閾値，およびタイムステップ t での報酬を表す．K_s は事前に設定されるパラメータである．

既存手法と異なる点は，探索の打ち止め規則が存在する点である (steps 8-10)．K_s までのアームの中で，コスト当たりの報酬が最も大きい値よりも大きい値となるアームがプレイされた時，探索を打ち止める．探索の打ち切りにより予算が小さい時でも損失が小さくなることを目的としている．

Algorithm 3 The seKUBE Algorithm

1: $t=1; B_t = B; F =$ **true**;
2: set threshold value to K_s;
3: **while** pulling is feasible **do**
4: **if** $B_t < \min_i c_i$ **then**
5: STOP { pulling is not feasible}
6: **end if**
7: **if** $t \le K$ **and** F **then**
8: Initial phase: play arm $i(t) = t$;
9: **if** $\frac{r(i(t))}{c(i(t))} \ge \max\{\frac{r(i(x))}{c(i(x))}, x = 1, 2, \cdots, K_s\}$ **then**
10: $F =$ **false**;
11: **end if**
12: **else**
13: use density-ordered greedy to calculate $M^*(B_t) = \{m^*_{i,t}\}$, the solution of Equation 1;
14: randomly pull $i(t)$ with $P(i(t)=i) = \frac{m^*_{i,t}}{\sum_{k=1}^{K} m^*_{k,t}}$;
15: **end if**
16: update the estimated upper bound of arm $i(t)$;
17: $B_{t+1} = B_t - c_{i(t)}; t = t+1$;
18: **end while**

探索を最適停止問題ととらえ，探索時に実行するアーム数を決定する理由について述べる．LAKUBE では，K_α を事前に設定しなければならない．しかし，単純に K_α で打ち切ることは，もっとも報酬が得られるアームが実行されなくなる可能性を生み出す．したがって，より多くの報酬が得られるアームを活用でプレイできるように探索を停止しなければならない．最適停止問題の一種である秘書問題では，応募者の中から誰を採用するのかについて研究されており，問題の設定や採用したい条件によって最適な方針が求められている [8]．その最適な方針に従うことによって，最適なアームがプレイされる確率をなるべく大きくすることが期待される．したがって本研究では，探索を最適停止問題ととらえた探索打ち切り規則を採用した．

4. 評価実験

4.1 実験設定

既存手法である KUBE および LAKUBE と，提案手法である seKUBE の評価実験の設定について述べる．本論文の実験設定は，Tran らの実験設定に従う [2]．アーム数 K は 100 とし，アームの報酬確率分布は，切断正規分布を用いる．切断正規分布の平均，分散，および定義域は以下の通りである．

- 平均 $\mu_i = [10, 20]$
- 分散 $\delta_i = \frac{\mu_i}{2}$
- 定義域 $[0, 2\mu_i]$

平均 μ は，与えられた $[10, 20]$ の範囲からランダムに選ばれる．分散および定義域は平均値が与えられることで求められる．アームのコストは，[1，10] の範囲からそれぞれのアームに対してランダムに設定する．

LAKUBE のパラメータ K_α は，最も結果が小さくなる値を採用する．

seKUBE のパラメータ K_s は，

- Case 1：K/e
- Case 2：$\sqrt{K}$

の 2 つに場合分けをする．Case 1 の値は，最善の選択肢を選ぶ確率が最も大きくなる値である [3]．Case 2 の値は，最善の選択肢でなくともなるべく良い選択肢を選ぶことが可能な値である [4]．

4.2 実験結果

KUBE，LAKUBE，および seKUBE を比較した結果について述べる．実験結果の評価軸として，縦軸には損失率，横軸には予算を用いる．損失率は式 (2) から求める．

$$1 - \frac{total\_reward}{total\_reward^*} \tag{2}$$

$total\_reward$ はあるアルゴリズムによってアームを選択した時のアームの平均報酬の合計，$total\_reward^*$ は最も得られる平均報酬が大きいアームを選択した時の平均報酬の合計を表す．損失率が小さいほど，平均報酬が大きいアームを選択できていると言える．

実験結果の全体を図 1，予算 5500 以降について拡大したグラフを図 2 に示す．実験結果で示す損失率は 100 回試行した平均を用いている．

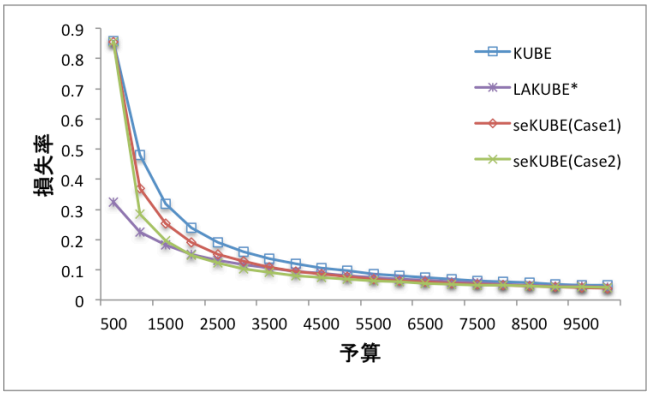

図 1: KUBE，LAKUBE，および seKUBE の損失率の比較：全体図

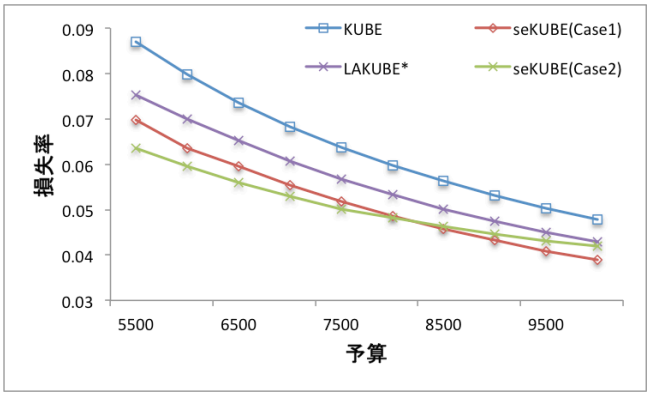

図 2: KUBE，LAKUBE，および seKUBE の損失率の比較：拡大図

本実験の結果をまとめると以下の通りである．

- 予算が特に小さい時には 4 手法の中で LAKUBE が最も損失率が小さい．

予算	KUBE	LAKUBE	seKUBE Case1	seKUBE Case2
500	0.856552447	**0.324531912**	0.855695687	0.854992899
1000	0.480411395	0.224908523	0.368648984	0.285200844
1500	0.319726049	0.183745956	0.252374878	0.196767704
2000	0.239967418	0.150799581	0.190455551	**0.147261175**
2500	0.192179764	0.130577678	0.152443495	0.121979604
3000	0.160324725	0.116665919	0.127810652	0.104187753
3500	0.137143598	0.104452408	0.108874421	0.090971242
4000	0.120068968	0.094810631	0.094955732	0.080699647
4500	0.106791385	0.087680347	0.084537749	0.073274129
5000	0.096167939	0.081509321	0.077078386	0.067740959
5500	0.086979121	0.075195259	0.069767896	0.063498949
6000	0.079734725	0.07001883	0.063589747	0.059546941
6500	0.073602121	0.065170771	0.059497681	0.055897531
7000	0.068341313	0.060624054	0.055327918	0.052877602
7500	0.063785468	0.056684205	0.05171439	0.050124907
8000	0.059797501	0.053236899	0.048552553	0.048214918
8500	0.056280001	0.050195355	0.045762697	0.046295502
9000	0.053153334	0.047491223	**0.043282825**	0.044589799
9500	0.050356649	0.045073015	0.040850542	0.043063439
10000	0.047838001	0.042894732	0.038864265	0.041895477

表 1: 各提案手法の損失率の比較

- 予算が 2000 以上のとき，seKUBE(Case 2) が最も損失率が小さくなる．
- 予算が 9000 以上のとき，seKUBE(Case 1) が最も損失率が小さくなる．

提案手法である seKUBE(Case 1) および seKUBE(Case 2) は，既存手法である KUBE と比較するとどの予算の場合も損失率が小さいことが分かる．提案手法が LAKUBE の元手法である KUBE よりも改善されていることが確認できる．

提案手法と LAKUBE を比較すると，予算が小さいとき損失率に差があることが分かる．特に顕著に表れているのは，予算が 500 の時である．提案手法の損失率が約 0.85 であるのに対して，LAKUBE は約 0.3 と 0.5 以上の差ができてしまった．原因としては，seKUBE の探索打ち切り規則では探索の時点で予算が不足してしまうことが想定される．seKUBE では一定数のアームをプレイした後に，その時点までの最も報酬を得られるアームよりも良いアームがプレイされなければ探索を終了しない．予算が小さいときには，より良いアームを発見する前に予算が尽きてしまうためである．

しかし，予算が 2000 以上になると，提案手法の方が LAKUBE よりも損失率が小さいことが分かる．最適停止問題を解くことによる探索打ち止め規則により，報酬の大きいアームを含むように探索し，活用が不要なアームをプレイせずに済んだためだと想定される．

seKUBE の Case 1 と Case 2 を比較すると，Case 2 の方が予算が小さいとき損失率が小さくなっている．Case 1 が Case 2 よりも損失率が小さくなるのは，予算が 9000 以上のときである．予算が十分大きいときには，アームを探索する数を増やしたほうが良い結果になることが分かる．

次に箱ひげ図を用いて既存手法と提案手法の比較を行なう．図 3，図 4，図 5 および図 6 はそれぞれ，KUBE の箱ひげ図，LAKUBE の箱ひげ図，seKUBE(Case 1) の箱ひげ図および seKUBE(Case 2) の箱ひげ図を表す．

図 3 の KUBE の箱ひげ図から，すべてのアームを探索する KUBE は試行ごとの差が小さいことが分かる．図 4 の LAKUBE の箱ひげ図から，中央値がかならずしも下がっているわけではなく，不安定であることが見られる．事前に探索

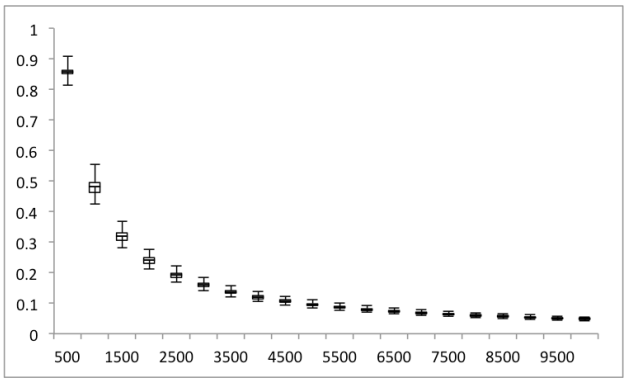

図 3: KUBE の損失率

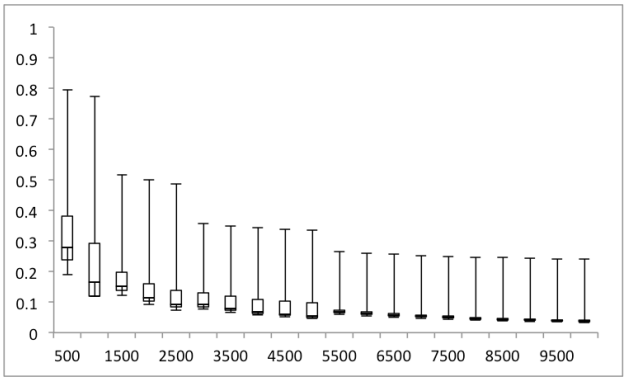

図 4: LAKUBE の損失率

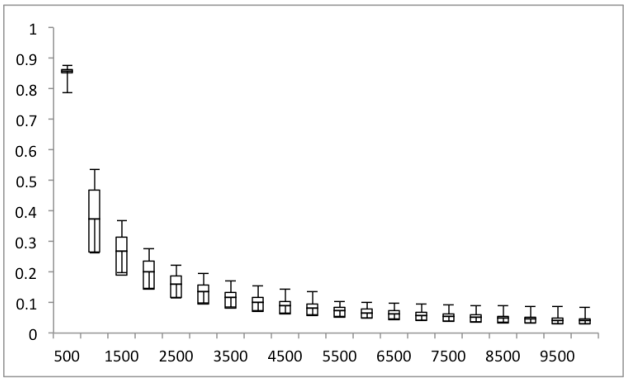

図 5: seKUBE(Case 1) の損失率

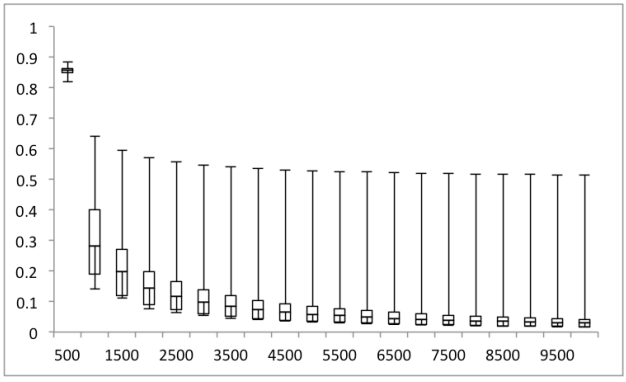

図 6: seKUBE(Case 2) の損失率

するアーム数を決めてしまうことから，必ずしも良いアームを選択できていないためだと想定される．図 5 の seKUBE(Case 1) の箱ひげ図から，KUBE よりも試行ごとの差があるものの損失率としては小さくなっていることが分かる．また，最悪の場合であっても他の試行との差が小さいことが分かる．図 6 の seKUBE(Case 2) の箱ひげ図から，最悪の場合の損失率がかなり大きいことが分かる．しかし，LAKUBE とは異なり中央値は予算が増えるごとに減少している．

箱ひげ図を用いた比較から，LAKUBE では損失率の中央値が必ずしも小さくなっていないのに対して，提案手法である seKUBE では試行ごとに多少の差はあるものの，安定して中央値が小さくなっていることが確認された．

5. まとめ

本研究では，最適停止問題による探索の打ち止め規則を応用した seKUBE を提案した．提案手法では予算が非常に小さいとき，LAKUBE の損失率よりも大きくなってしまった．しかし，KUBE と比較すると提案手法の方が改善されており，他の予算設定では LAKUBE よりも良い結果を得ることができた．

参考文献

[1] Kadono, Yoshiaki, and Naoki Fukuta. ”LAKUBE: An Improved Multi-Armed Bandit Algorithm for Strongly Budget-Constrained Conditions on Collecting Large-Scale Sensor Network Data.” PRICAI 2014: Trends in Artificial Intelligence. Springer International Publishing, pp.1089-1095, 2014.

[2] Tran-Thanh, Long, et al:”Knapsack Based Optimal Policies for Budget-Limited Multi-Armed Bandits.” AAAI-12, pp.1134-1140, 2012.

[3] Ferguson, T.S., Who solved the secretary problem?, Statistical Science, pp.282-289, 1989.

[4] J. N. Bearden. ”A new secretary problem with rank-based selection and cardinal payoffs.” Journal of Mathematical Psychology, volume 50, pp.58-59, 2006.

[5] Tran-Thanh, Long, Alex Rogers, and Nicholas R. Jennings. ”Long-term information collection with energy harvesting wireless sensors: a multi-armed bandit based approach.” Autonomous Agents and Multi-Agent Systems 25.2 (2012): 352-394.

[6] Rogers, Alex, Daniel D. Corkill, and Nicholas R. Jennings. ”Agent technologies for sensor networks.” IEEE Intelligent Systems 24.2, pp.13-17, (2009).

[7] Romer, Kay, and Friedemann Mattern. ”The design space of wireless sensor networks.” Wireless Communications, IEEE 11.6, pp.54-61, 2004.

[8] 玉置光司. ”秘書問題の最近の展開.” 数理解析研究所講究録 1263 pp131-150, 2002.