DOI:10.7544/issn1000-1239.2019.20180019
56(3):643-654,2019

一种自适应的多臂赌博机算法

章晓芳[1,2] 周 倩[1] 梁 斌[1] 徐 进[1]
[1](苏州大学计算机科学与技术学院 江苏苏州 215006)
[2](计算机软件新技术国家重点实验室(南京大学) 南京 210023)
(xfzhang@suda.edu.cn)

An Adaptive Algorithm in Multi-Armed Bandit Problem

Zhang Xiaofang[1,2], Zhou Qian[1], Liang Bin[1], and Xu Jin[1]
[1](*School of Computer Science and Technology, Soochow University, Suzhou, Jiangsu* 215006)
[2](*State Key Laboratory for Novel Software Technology (Nanjing University), Nanjing* 210023)

Abstract As an important ongoing field in machine learning, reinforcement learning has received extensive attention in recent years. The multi-armed bandit (MAB) problem is a typical problem of the exploration and exploitation dilemma in reinforcement learning. As a classical MAB problem, the stochastic multi-armed bandit (SMAB) problem is the base of many new MAB problems. To solve the problems of insufficient use of information and poor generalization ability in existing MAB methods, this paper presents an adaptive SMAB algorithm to balance exploration and exploitation based on the chosen number of arm with minimal estimation, namely CNAME in short. CNAME makes use of the chosen times and the estimations of an action at the same time, so that an action is chosen according to the exploration probability, which is updated adaptively. In order to control the decline rate of exploration probability, the parameter w is introduced to adjust the influence degree of feedback during the selection process. Furthermore, CNAME does not depend on contextual information, hence it has better generalization ability. The upper bound of CNAME's regret is theoretically proved and analyzed. Our experimental results in different scenarios show that CNAME can yield greater reward and smaller regret with high efficiency than commonly used methods. In addition, its generalization ability is very strong.

Key words reinforcement learning; multi-armed bandit; exploration and exploitation; adaptation; contextual

摘 要 多臂赌博机问题是强化学习中研究探索和利用两者平衡的经典问题,其中,随机多臂赌博机问题是最经典的一类多臂赌博机问题,是众多新型多臂赌博机问题的基础.针对现有多臂赌博机算法未能充分使用环境反馈信息以及泛化能力较弱的问题,提出一种自适应的多臂赌博机算法.该算法利用当前估计值最小的动作被选择的次数来调整探索和利用的概率(chosen number of arm with minimal estimation, CNAME),有效缓解了探索和利用不平衡的问题.同时,该算法不依赖于上下文信息,在不同场景的多臂赌博机问题中有更好的泛化能力.通过理论分析给出了该算法的悔值(regret)上界,并通过不同场景的实验结果表明:CNAME 算法可以高效地获得较高的奖赏和较低的悔值,并且具有更好的泛化能力.

收稿日期:2018-01-09;**修回日期**:2018-08-27
基金项目:国家自然科学基金项目(61772263,61772014,61572375);苏州市科技发展计划基金项目(SYG201807)
This work was supported by the National Natural Science Foundation of China (61772263, 61772014, 61572375) and the Suzhou Science and Technology Development Project (SYG201807).

关键词 强化学习;多臂赌博机;探索和利用;自适应;上下文相关

中图法分类号 TP181

强化学习(reinforcement learning, RL)是智能体从环境状态到动作映射的学习,以使动作从环境中获得的累积奖赏值最大[1-3].强化学习通过试错与环境交互获得策略的改进,这就导致一个问题:哪一种试错策略可以产生最有效的学习.因此强化学习面临探索(exploration)和利用(exploitation)的两难问题:长期来看,探索可能获得更高的累积奖赏,帮助智能体收敛到最优策略;而利用总是最大化短期奖赏,但可能收敛到次优解上[4-5].多臂赌博机(multi-armed bandit, MAB)问题是强化学习中研究探索与利用平衡的经典问题.

随机多臂赌博机(stochastic multi-armed bandit, SMAB)问题是一类经典的 MAB 问题,该问题最早由 Robbins 提出[6].一个 MAB 问题中包括 K 个臂,玩家每次只能选择其中一个臂,并获得相应的奖赏.n 轮游戏后,玩家的目标是最大化奖赏的期望值.其中,玩家选择某个臂之后只能知道该臂的奖赏,且各个臂的奖赏是相互独立的,服从某种未知的分布.为了达到目标,玩家需要在获得奖赏的同时去学习该奖赏的分布.因此,每个时间步,玩家都必须在以下两者做出选择:利用,选择目前为止已知奖赏最高的臂;探索,尝试其他未来奖赏可能更高的臂.这样,在游戏过程中玩家面临着探索与利用间的平衡问题.

作为一个序列化的决策问题,MAB 被应用于很多实际场景中,最早的应用就是 Robbins 提出的诊治试验[6].诊治试验中各个患者的治疗方案对应 MAB 问题中的各个臂.诊治试验的目标是最小化患者的健康损失.近来,MAB 问题有了更多广泛的应用,比如推荐算法[7-9]、众包机制[10-12]、搜索引擎关键字选择[13]、网络信道选择[14]等.除了 SMAB 问题,MAB 问题还存在多个变种,例如 Markov 赌博机(Markovian bandit)[15],对抗式赌博机(adversarial bandit)[15]和上下文相关的赌博机(contextual bandit)[16]等.在上下文相关的赌博机问题中,玩家选择一个臂之后,除了获得相应的奖赏之外,还可以获得该臂的上下文信息.本文将研究 SMAB 问题,提出一种自适应的 MAB 算法,并将其推广到上下文相关的场景中.

目前已有很多策略来平衡 MAB 问题中探索与利用的难题.Watkins 首次引入 ε-贪心(ε-greedy)算法[17],ε-greedy 算法以一个很小的概率 ε 进行探索,以 1-ε 的概率利用.但是 ε-greedy 算法将在所有臂中均等选择进行探索,没有利用选择臂之后获得的信息,比如各个臂的平均奖赏和选择次数.软最大化(SoftMax)算法最早由 Luce 提出[18],SoftMax 算法是基于当前已知臂的平均奖赏来对探索和利用的程度进行折中.然而 SoftMax 算法只考虑了平均奖赏,这种较为单一的信息可能导致错误的概率分配,比如某个臂第一次被选择,随机得到一个很低的奖赏,SoftMax 算法根据这一次的奖赏为该臂分配了很低的选择概率,但该臂有可能是初期奖赏低而实际平均奖赏很高的臂.置信上界(upper-confidence-bound, UCB)动作选择方法[19]则根据各个臂已知的平均奖赏和选择次数直接计算出要选择的臂,不利用随机性来选择臂.使用 UCB 算法时必须在初期轮流选择所有臂各一次,那么当实验次数小于等于臂的数量时,难以使用 UCB 方法,因此 UCB 方法泛化能力较弱.LinUCB 算法[20-21]是经典的上下文相关的 MAB 算法,利用上下文信息最大化奖赏的期望值.然而 LinUCB 算法只适用于上下文可用的场景中.

本文针对上述随机多臂赌博机算法未能充分使用反馈信息以及泛化能力较弱的问题,提出了一种自适应的多臂赌博机算法,利用当前估计值最小的动作被选择的次数(chosen number of arm with minimal estimation, CNAME)来调整探索和利用的概率,记为 CNAME 算法.本文的主要贡献有3个方面:

1) 提出了一种自适应的随机多臂赌博机算法 CNAME,其探索概率由一个简单分式给出,该分式由当前平均奖赏最小的臂被选择的次数和参数 w 构成,充分利用了选择臂之后的信息.该算法不依赖上下文信息,因此在各个应用场景中具有更强的推广能力和泛化能力.

2) 通过理论分析给出了 CNAME 算法的悔值(regret)上界,并与其他算法的悔值上界进行对比,为 CNAME 算法提供了有力的效果保证.

3) 通过实验研究给出了 CNAME 算法中关键参数的参考取值范围.在无上下文的随机数据集和内容分发网络(content distribution network, CDN)[22]数据集以及带有上下文的雅虎公司 R6A 数据集上的实验结果表明:利用估计值最小的动作被选择的次数能有效平衡探索和利用,并且在上下文相关的场景中依然表现良好.

1 背景和相关工作

1.1 随机多臂赌博机问题

随机多臂赌博机问题,又称为 K-臂赌博机问题,K 即为臂的数量,下文把各个臂统称为动作. K-臂赌博机的目标是最大化获得的累积奖赏. 若时刻 t 动作 i 被选择,则获得随机奖赏 $X_{i,t}$. 各个动作的奖赏相互独立且服从均值为 $\boldsymbol{\mu}=[\mu_1,\mu_2,\cdots,\mu_K]$ 的某种分布,μ_i 是动作 i 的真实值. 假设已知各个动作的真实值,只要一直选择有最高真实值的动作便可以获得最大的累积奖赏,这种情况下多臂赌博机问题自然得解. 然而,事实上随机多臂赌博机问题可以通过获得的奖赏来估计各个动作的值,但动作的真实值是未知的.

悔值是 K-臂赌博机的一种典型的度量标准[15]. 在动作真实值未知的情况下,任何算法都不可能每次都选到奖赏最高的动作,悔值就是实际得到的奖赏与期望得到的最大奖赏之间的差值. 给定 n 次操作后的悔值定义为

$$R_n=n\mu^*-\sum_{i=1}^{K}E[N_n(i)]\mu_i,$$
$$\mu^*=\max_{1\leqslant j\leqslant K}\mu_j, \tag{1}$$

其中,$N_n(i)$是前 n 次操作中动作 i 被选择的次数. 悔值越小,则该算法产生的策略越接近最优策略.

1.2 估计值函数

动作的真实值是选择动作后期望得到的平均奖赏,因此一种估计动作值的简单方法是计算选择动作之后实际获得的平均奖赏. 假设第 n 次操作之后,动作 i 已经被选择 $N_n(i)$ 次,得到的奖赏分别是 r_1,r_2,…,$r_{N_n(i)}$,这时动作 i 的估计值为

$$Q_{n+1}(i)=\frac{r_1+r_2+\cdots+r_{N_n(i)}}{N_n(i)}. \tag{2}$$

$Q_{n+1}(i)$是到第 n 次操作之后的动作 i 的估计值,即动作 i 实际获得的平均奖赏. 根据式(2),$Q_{n+1}(i)$的更新需要记录每次选择动作 i 之后获得的奖赏,为了减小需要的存储空间,估计值的更新可以转化为增量式的更新:

$$Q_{n+1}(i)=Q_n(i)+\frac{1}{N_n(i)}[r_{N_n(i)}-Q_n(i)]. \tag{3}$$

根据式(3),估计值的更新只需要存储立即奖赏和上一时刻的估计值. 若 $N_n(i)=0$,$Q_n(i)$为初始值;当 $N_n(i)\to\infty$时,根据大数定理可知 $Q_n(i)$收敛到动作 i 的真实值. 这种方法叫作抽样平均(sample average)[23],每个估计值都是奖赏的简单平均值. 因此,通过以上方法得到动作的估计值后,可以根据估计值作出动作选择.

1.3 动作选择方法

本节主要介绍已有的 3 类经典随机多臂赌博机算法.

1.3.1 ε-greedy 方法及其变种

最简单的动作选择方法是一直选择估计值最高的动作,即贪心动作 a^*,这种方法称为贪心(greedy)方法[1]. 贪心方法利用当前知识最大化立即奖赏,没有探索的过程.

一个简单的代替贪心的办法是在大多数时间内采用 a^*,但是以一个很小的概率 ε 进行探索,即与估计动作值无关地随机均匀地选择一个动作,该方法为 ε-greedy[17],ε 位于开区间(0,1)上.

ε-greedy 算法的一个变种是 ε-优先(ε-first)算法[24],ε-first 算法在实验初期先做完所有的探索工作. 例如对于一个给定实验次数为 n 的实验,在最初的 εn 次实验中随机选择动作,即纯探索阶段,在余下的$(1-\varepsilon)n$ 次实验中,选择平均奖赏最高的动作,即纯利用阶段.

ε-greedy 算法的另一个变种是 ε-递减(ε-decreasing)算法[25]. ε-decreasing 算法用一个递减的概率来接近最优策略,递减变量 ε_t 由 $\varepsilon_t=\min\{1,\varepsilon_0/t\}$给出,其中 t 是时间步,初始参数 $\varepsilon_0>0$. 此外,Cesa-Bianchi 等人[25]提出了混合贪心(greedy mix)算法,用递减因子 $1/\ln t$ 代替了 ε-decreasing 算法的递减因子 $1/t$. 类似地,Strens[26] 提出了 Least Taken 算法,Vermorel 等人[27]则在 Least Taken 算法的基础上添加了初始参数 ε_0.

1.3.2 SoftMax 方法及其变种

尽管 ε-greedy 算法是一类有效的方法,但是 ε-greedy 算法在探索的时候是在所有的动作中均等选择. 这就意味着 ε-greedy 算法选择下一个动作时有可能选到最差的动作. 为了避免这种情况,SoftMax 方法将探索概率更改为与估计值相关的一个分级函数:贪心动作仍然具有最高的选择概率,其他动作则根据其估计值进行分级并分配权重. SoftMax 算法广泛使用 Gibbs 或 Boltzmann 分布,时刻 t 选择动作 i 的概率为

$$\frac{e^{Q_t(i)/\tau}}{\sum_{j=1}^{K}e^{Q_t(j)/\tau}}, \tag{4}$$

其中,τ 是温度(temperature)参数[28]. 高温可以使

得各动作的选择概率趋于相等,低温则使具有不同估计值的动作的选择概率趋于不同. 当 $\tau \to 0$ 时,SoftMax 算法的作用与 greedy 方法一致.

与 ε-greedy 算法相似,SoftMax 算法可以变形为递减的软最大化(decreasing SoftMax)算法[25],decreasing SoftMax 算法的温度随时间步的增加而递减,即 $\tau_t=\tau_0/t$. 类似地,可以用递减因子 $1/\ln t$ 代替 decreasing SoftMax 算法的递减因子 $1/t$[25]. 作为一个更复杂的 SoftMax 算法变种,Exp3 算法的关键在于利用了累计奖赏与动作选择概率的比值[25].

1.3.3 UCB 方法

由于动作值的估计具有不确定性,因此,需要在动作选择过程中引入探索. 然而,非贪心动作中有接近贪心的动作,也有很差的动作. 如果能根据非贪心动作成为最优动作的可能性进行探索,探索的效果往往会更好.

基于上述的思想,UCB 算法动作选择方法同时考虑非贪心动作估计值对最优动作估计值的接近程度和估计值的不确定性:

$$A_t=\arg\max_i\left[Q_t(i)+c\sqrt{\frac{\ln t}{N_t(i)}}\right], \tag{5}$$

其中,t 是时间步,A_t 是时刻 t 应该被选择的动作,$N_t(i)$ 表示动作 i 在时间步 t 之前已经被选择的次数,参数 $c>0$ 控制探索的程度. 若 $N_t(i)=0$,则把动作 i 看作最优动作 A_t. 参数 $c=\sqrt{2}$ 时,为 UCB1 算法[19]. 本文后续对比实验中采用的是 UCB1 算法.

雅虎公司的科学家在 UCB 算法的基础上引入了上下文信息,提出了 LinUCB 算法[20],并应用于雅虎公司的新闻推荐中. LinUCB 算法假设一个玩家选择一个动作之后,其奖赏和上下文信息成线性关系. 于是学习过程就变成:用玩家和动作的上下文信息预估奖赏及其置信区间,选择置信区间上界最大的动作,观察奖赏后更新线性关系的参数,以此达到最大化奖赏期望值的目的.

1.3.4 已有算法存在的问题

基于上述讨论可知:ε-greedy 算法是一个简单有效的方法,其存在的问题是在所有动作中均等探索,没有利用选择动作后获得的反馈信息. SoftMax 算法仅基于当前已知动作的估计值对各动作的选择概率进行分级,因此 SoftMax 算法容易被误导. ε-greedy 算法和 SoftMax 算法都仅需要设置一个参数,计算相对简单. 与上述 2 种方法不同的是,UCB 动作选择方法充分利用了动作的估计值和被选择次数,每次都根据已有信息直接计算出要选择的动作,其计算开销相对较大. 另外,UCB 动作选择方法必须在实验初期轮流选择所有动作各一次,因此当实验次数小于等于动作的数量时,UCB 动作选择方法将不适用. LinUCB 算法则只适用于上下文信息可用的场景中,实际应用中存在着大量没有先验信息的环境,此时 LinUCB 算法的使用将受限.

2 算法描述

本文针对已有的随机多臂赌博机算法存在的不足,提出了一种自适应的随机多臂赌博机算法(CNAME).

2.1 CNAME 算法描述

探索概率的变化将直接影响探索和利用的平衡. 针对该问题,本文提出一种新的探索概率计算方法:

$$p=w/(w+m_t^2),$$

其中,m_t 是当前平均奖赏最小的动作被选择的次数,参数 $w>0$ 用来控制探索概率衰减的速率,起到宏观调控的作用. w 和 m_t 共同控制探索概率. 该计算方法可以充分利用选择动作之后的信息,根据环境自适应地调整探索概率,平衡探索和利用. 本文 3.1 节将通过实验讨论参数 w 对 CNAME 算法效果的影响及其参考取值范围.

CNAME 算法以 $w/(w+m_t^2)$ 的概率选择目前最少被选择的动作,即探索;或者选择当前估计值最大的动作,即利用. 具体如算法 1 所示:

算法 1. CNAME 算法.

输入:动作集合;

输出:估计值.

① 设置参数 w;

② for 每个动作 $i\in[1,2,\cdots,K]$ do

③ $Q_0(i)=0$;

④ $N_0(i)=0$;

⑤ end for

⑥ for 每个时间步 $t\leqslant n$ do

⑦ $m_t=N_t(\arg\min_i Q_t(i))$;

⑧ $p=\dfrac{w}{w+m_t^2}$;

⑨ 在开区间(0,1)上产生一个随机数 x;

⑩ $a_t\leftarrow\begin{cases}\arg\max_i Q_t(i), & x>p;\\ \arg\min_i N_t(i), & \text{otherwise};\end{cases}$

⑪ 获得立即奖赏 $X_{a_t,t}$;

⑫ $N_t(a_t)=N_{t-1}(a_t)+1$;

⑬ $Q_t(a_t)=Q_{t-1}(a_t)+\frac{1}{N_{t-1}(a_t)}[X_{a_t,t}-Q_{t-1}(a_t)]$;

⑭ end for

根据算法 1,步骤①首先设置 CNAME 算法的关键参数 w,参数 w 影响着探索概率衰减的速率.步骤②~⑤初始化动作的估计值和被选择的次数,即 $Q_0(i)=0$,$N_0(i)=0$.这里也可以使用乐观初始值,比如 $Q_0(i)=1$,乐观初始值在实验初期可以临时地促进探索[1].步骤⑥~⑩为动作选择的过程,n 是实验次数,m_t 是时刻 t 估计值最小的动作已经被选择的次数,p 是当前的探索概率.步骤⑩在选择动作时,同时考虑了实际获得的平均奖赏和动作选择的次数.步骤⑪是选择动作 a_t 后获得立即奖赏 $X_{a_t,t}$,该奖赏服从某种概率分布.步骤⑫⑬对动作被选择的次数和估计值进行更新,其中,估计值更新方法采用的是增量式抽样平均方法.

2.2 CNAME 算法分析

在任何一个特定环境下,探索的效果更优还是利用的效果更优取决于估计的精确值、环境不确定性和剩余操作次数等复杂的具体情况[1].

CNAME 算法充分利用了用户反馈,即动作估计值和动作选择的次数,自适应地根据实际情况来调整探索概率.同时,通过参数 w 控制探索概率下降的速率,并根据 m_t 来改变探索概率.具体从 3 个方面分析:

1) 当参数 w 的取值较大时,m_t 对探索概率的影响较小;当参数 w 的取值较小时,m_t 对探索概率的影响则较大.由于不同环境中 m_t 的实际影响也不同,本文设计参数 w 来削弱或者增强 m_t 在调整探索概率时起到的作用.

2) 每当 m_t 增加一次,说明估计值最小的动作又被选择了一次,估计值最小的动作是对整个累积奖赏贡献最少的动作,应尽量减少该动作被选择的次数.随着 m_t 的增加,探索概率降低,贪心动作的选择概率增加,这样有助于提高实际获得的累积奖赏.

3) CNAME 算法在探索时选取的是当前最少被选择的动作,这样就保证每个动作都有被选择到的机会,同时避免了随机选择动作的盲目性.

2.3 CNAME 算法的悔值上界证明

基于 2.1~2.2 节的算法描述和分析,本节将通过证明给出 CNAME 算法的悔值上界.悔值用来评价算法的好坏,悔值上界越小,说明该算法的效果越好.

Lai 等人[29]发现,对于均值为单一参数的奖赏分布,所有算法都满足:

$$E[N_n(i)]\geqslant\frac{\ln n}{D(p_i\parallel p^*)},\tag{6}$$

其中,$D(p_i\parallel p^*)=\int_X p_i(x)\ln p_i(x)/p^*(x)\mathrm{d}x$ 是 p_i 和 p^* 之间的 KL 散度,p_i 指任意次优动作 i 在随机变量 X 上的概率分布,p^* 指最优动作在随机变量 X 上的概率分布,$x\in X$.KL 散度可度量 p_i 和 p^* 的接近程度,2 个概率分布越接近,KL 散度值越小,则式(1)中的悔值也越小,p_i 越接近最优策略 p^*.

对于某个算法 A,根据式(1)(6),若已知算法中 p_i 的上界,便可得到算法 A 的悔值上界.为了方便证明,给出已知的 2 个不等式:

引理 1. Chernoff-Hoeffding 不等式.

设 $X_1,X_2,\cdots,X_n$ 为[0,1]之间的随机变量,且 $E[X_t\mid X_1,X_2,\cdots,X_{t-1}]=\mu$,$S_n=X_1+X_2+\cdots+X_n$.那么对于所有的 $a\geqslant 0$ 都有:

$$P\{S_n\geqslant n\mu+a\}\leqslant\exp\left(-\frac{2a^2}{n}\right);$$

$$P\{S_n\leqslant n\mu-a\}\leqslant\exp\left(-\frac{2a^2}{n}\right).$$

引理 2. Bernstein 不等式.

设 $X_1,X_2,\cdots,X_n$ 为[0,1]之间的随机变量,且 $\sum_{t=1}^{n}Var[X_t\mid X_{t-1},\cdots,X_2,X_1]=\sigma^2$,$S_n=X_1+X_2+\cdots+X_n$,那么对于所有的 $a\geqslant 0$ 都有:

$$P\{S_n\geqslant E[S_n]+a\}\leqslant\exp\left(-\frac{a^2/2}{\sigma^2+a/2}\right).$$

定理 1. 第 n 次操作时,CNAME 算法的次优动作选择概率 p_i 不超过:

$$\frac{w}{(w+1)K}+\frac{wn-2}{Kw}\exp\left(-\frac{wn-2}{10Kw}\right)+\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right).$$

证明. 首先,对于所有的动作 $1\leqslant i\leqslant K$,定义 $X_{i,n}$ 为动作 i 在第 n 次操作得到的立即奖赏,显然 $E[X_{i,n}]=\mu_i$.另外,拥有 μ^* 的动作叫作最优动作.类似地,N_n^* 表示最优动作在前 n 次操作被选择的次数,$\overline{X}_n^*$ 表示最优动作在第 n 次操作时的平均奖赏.最优动作值与次优动作值的差值为 $\Delta_i=\mu^*-\mu_i$.动作 i 在第 n 次操作时的平均奖赏定义为

$$\overline{X}_{i,n}=\frac{1}{n}\sum_{t=1}^{n}X_{i,t}.$$

令前 n 次操作中动作 i 被选择次数的期望为：$E[N_n(i)]=\sum_{t=1}^{n}P\{a_t=i\}$，其中 $P\{a_t=i\}$ 为时刻 t 动作 i 的选择概率.

令 ε_n 表示第 n 次操作的探索概率，即

$$\varepsilon_n=\frac{w}{w+m_n^2},$$

其中，$w>0$. 同时，令 $x_0=\frac{1}{2K}\sum_{t=1}^{n}\varepsilon_t$. 则动作 i 在第 n 次操作的选择概率满足：

$$P\{a_n=i\}\leqslant\frac{\varepsilon_n}{K}+\left(1-\frac{\varepsilon_n}{K}\right)P\{\overline{X}_{i,N_{n-1}(i)}\geqslant\overline{X}^{*}_{N^{*}_{n-1}}\}. \tag{7}$$

根据式(7)，要求解 $P\{a_t=i\}$，只需要求出概率 $P\{\overline{X}_{i,N_{n-1}(i)}\geqslant\overline{X}^{*}_{N^{*}_{n-1}}\}$ 即可. 已知：

$$P\{\overline{X}_{i,N_{n-1}(i)}\geqslant\overline{X}^{*}_{N^{*}_{n-1}}\}\leqslant$$

$$P\left\{\overline{X}_{i,N_{n-1}(i)}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}+P\left\{\overline{X}^{*}_{N^{*}_{n-1}}\leqslant\mu^{*}-\frac{\Delta_i}{2}\right\},$$

令 $N_n^R(i)$ 为动作 i 在前 n 次操作中随机被选到的次数，结合 Chernoff-Hoeffding 不等式(引理 1)，可得：

$$P\left\{\overline{X}_{i,N_{n-1}(i)}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}=$$

$$\sum_{t=1}^{n}P\left\{N_n(i)=t\wedge\overline{X}_{i,t}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}=$$

$$\sum_{t=1}^{n}P\left\{N_n(i)=t\mid\overline{X}_{i,t}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}\times$$

$$P\left\{\overline{X}_{i,t}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}\leqslant$$

$$\sum_{t=1}^{n}P\left\{N_n(i)=t\mid\overline{X}_{i,t}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}\times\exp\left(\frac{\Delta_i^2t}{2}\right)\leqslant$$

$$\sum_{t=1}^{x_0}P\left\{N_n(i)=t\mid\overline{X}_{i,t}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}+$$

$$\frac{2}{\Delta_i^2}\exp\left(\frac{\Delta_i^2x_0}{2}\right)\leqslant\sum_{t=1}^{\lfloor x_0\rfloor}P\left\{N_n^R(i)\leqslant t\mid\right.$$

$$\left.\overline{X}_{i,t}\geqslant\mu_i+\frac{\Delta_i}{2}\right\}+\frac{2}{\Delta_i^2}\exp\left(\frac{\Delta_i^2\lfloor x_0\rfloor}{2}\right)\leqslant$$

$$x_0\times P\left\{N_n^R(i)\leqslant x_0\right\}+\frac{2}{\Delta_i^2}\exp\left(\frac{\Delta_i^2\lfloor x_0\rfloor}{2}\right). \tag{8}$$

根据式(7)(8)，求解 $P\{a_n=i\}$ 的问题可以转化为求解 x_0 和 $P\{N_n^R(i)\leqslant x_0\}$ 的问题. 其中，$P\{N_n^R(i)\leqslant x_0\}$ 的求解过程为：

已知 $E[N_n^R(i)]=\frac{1}{K}\sum_{t=1}^{n}\varepsilon_t$，而且 $N_n^R(i)$ 的方差

$$\mathrm{Var}[N_n^R(i)]=\sum_{t=1}^{n}\frac{\varepsilon_t}{K}\left(1-\frac{\varepsilon_t}{K}\right)\leqslant\frac{1}{K}\sum_{t=1}^{n}\varepsilon_t.$$

根据 Bernstein 不等式(引理 2)可以得到：

$$P\{N_n^R(i)\leqslant x_0\}\leqslant\exp\left(-\frac{x_0}{5}\right). \tag{9}$$

求解 x_0，具体过程为

$$x_0=\frac{1}{2K}\sum_{t=1}^{n}\varepsilon_t=\frac{1}{2K}\sum_{t=1}^{n}\frac{w}{w+m_t^2}\geqslant$$

$$\frac{1}{2K}\sum_{t=1}^{n}\frac{w}{w+t^2}\geqslant\frac{1}{2K}\sum_{t=1}^{n}\left(1-\frac{t^2}{w+t^2}\right)\geqslant$$

$$\frac{1}{2K}\left(n-\frac{1}{w}\sum_{t=1}^{n}\frac{1}{t^2}\right)\geqslant\frac{1}{2K}\left(n-\frac{1}{w}-\frac{1}{w}\sum_{t=2}^{n}\frac{1}{t^2}\right)\geqslant$$

$$\frac{1}{2K}\left(n-\frac{2}{w}\right)\geqslant\frac{wn-2}{2Kw}. \tag{10}$$

当 $n\geqslant K$ 时，结合式(7)～(10)可得：

$$P\{a_n=i\}\leqslant\frac{\varepsilon_n}{K}+2\left(x_0\exp\left(-\frac{x_0}{5}\right)+\right.$$

$$\left.\frac{2}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2\lfloor x_0\rfloor}{2}\right)\right)\leqslant\frac{w}{(w+m_n^2)K}+\frac{wn-2}{Kw}\times$$

$$\exp\left(-\frac{wn-2}{10Kw}\right)+\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right)\leqslant$$

$$\frac{w}{(w+1)K}+\frac{wn-2}{Kw}\exp\left(-\frac{wn-2}{10Kw}\right)+$$

$$\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right).$$

那么对于 CNAME 算法，$w>0$，若 $n\geqslant K$，那么在第 n 次操作时动作 i 的选择概率 p_i 有：

$$p_i\leqslant\frac{w}{(w+1)K}+\frac{wn-2}{Kw}\exp\left(-\frac{wn-2}{10Kw}\right)+$$

$$\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right). \tag{11}$$

证毕.

定理 2. 第 n 次操作时，CNAME 算法的悔值满足：

$$R_n\leqslant n\mu^{*}-\sum_{i=1}^{K}(\mu_i\ln n)\Big/D\left(\frac{w}{(w+1)K}+\frac{wn-2}{Kw}\times\right.$$

$$\left.\exp\left(-\frac{wn-2}{10Kw}\right)+\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right)\Big\|p^{*}\right).$$

证明. 结合式(6)(11)可得到 $E[N_n(i)]$ 的下界为

$$E[N_n(i)]\geqslant(\ln n)\Big/\left(D\left(\frac{w}{(w+1)K}+\frac{wn-2}{Kw}\times\right.\right.$$

$$\left.\left.\exp\left(-\frac{wn-2}{10Kw}\right)+\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right)\Big\|p^{*}\right)\right). \tag{12}$$

最后，根据式(1)(12)可得出 CNAME 算法的悔值上界：

$$R_n\leqslant n\mu^{*}-\sum_{i=1}^{K}(\mu_i\ln n)\Big/D\left(\frac{w}{(w+1)K}+\frac{wn-2}{Kw}\times\right.$$

$$\left.\exp\left(-\frac{wn-2}{10Kw}\right)+\frac{4}{\Delta_i^2}\exp\left(-\frac{\Delta_i^2(wn-2)}{4Kw}\right)\Big\|p^{*}\right).$$

证毕.

值得注意的是,不同于其他算法的悔值上界,例如 ε-decreasing 算法的悔值上界 $O(\log n)$和 Exp3 算法的悔值上界 $O(\sqrt{Kn\log K})$,本节给出了 CNAME 算法的次优动作选择概率的上界,基于此可以得到某一时刻动作选择概率的分布,进而得到该时刻的悔值上界,即瞬时的悔值上界. 此时,若要进一步知道某个时刻的悔值,只需要知道最优动作与次优动作之间的差值.

3 实验部分

本节通过 4 组实验来分别研究 CNAME 算法中参数 w 的影响以及 CNAME 算法与其他算法的效果比较. 在对比实验中,使用 3 个数据集:服从正态分布的随机数据集、内容分发网络数据集[22]和带有上下文信息的雅虎公司 R6A 数据集①.

3.1 实验 1:CNAME 算法中参数 w 的影响

本节的实验对象是一组有 1 000 个随机产生的 K-臂赌博机的任务,其中 $K=10$. 具体实验设置如下:

1) 每个任务的真实值 $\boldsymbol{\mu}=[\mu_1,\mu_2,\cdots,\mu_K]$服从均值为 0、方差为 1 的正态分布.

2) 每个动作 i 的奖赏服从一个均值为 μ_i、方差为 1 的正态分布.

3) 1 000 个 K-臂赌博机任务通过 1 000 次重复随机选择 $\boldsymbol{\mu}$ 产生,选取 1 000 个不同的 K-臂赌博机任务是为了克服任务和环境本身的随机性.

在上述实验条件下取不同的 w 值,分别记录 1 000 个随机任务的平均奖赏和平均每轮的悔值.

首先,在闭区间[0.01,100]以 10 的倍率分别取 5 组不同的 w 值,图 1 给出了相应的平均奖赏和平均每轮的悔值.

从图 1 可以看出:当 $w\in[1,100]$时,随着 w 的减小,平均奖赏呈增加趋势,而平均每轮的悔值呈下降趋势. 然而当 $w=0.1$ 和 $w=0.01$ 时获得的平均奖赏却小于 $w=1$ 时的平均奖赏,即平均奖赏并不是随着 w 的减小而无限增加的. 同时,平均每轮的悔值的下降也存在下限. 基于图 1 的数据,可以猜测,$w=1$ 是 CNAME 算法效果的分界线,$w<1$ 或 $w>1$ 时,效果均下降. 较之其他 w 的取值,当 w 取值为 0.1 和 1 时,CNAME 算法的平均奖赏较高,平均每轮的悔值较低.

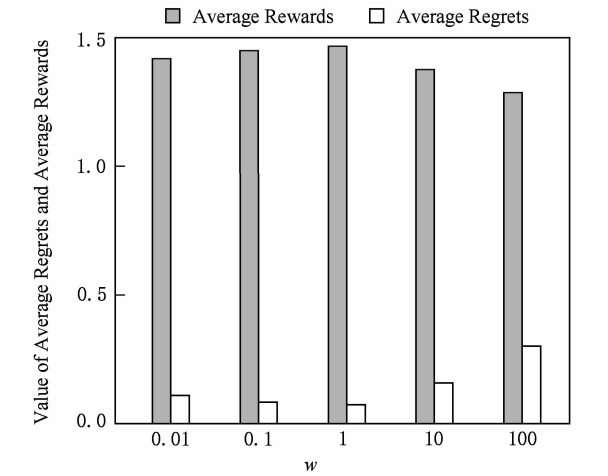

Fig. 1 Average rewards and average regrets of algorithm CNAME with different values of parameter w

图 1 不同 w 值下 CNAME 算法的平均奖赏和平均悔值

为进一步研究当 $w\in[0.1,1]$时,CNAME 算法的性能,下面以 0.1 为间隔分别取 10 组不同的 w 值展开实验. 表 1 给出了对应的实验结果数据,加粗部分是效果最好的参数及结果.

Table 1 Average Rewards and Average Regrets of CNAME ($w\in[0.1,1]$)

表 1 CNAME 算法的平均奖赏和平均悔值($w\in[0.1,1]$)

w	Average Rewards	Average Regrets
0.1	1.451	0.085
0.2	1.443	0.089
0.3	1.453	0.081
0.4	1.448	0.087
0.5	1.457	0.079
0.6	1.446	0.088
0.7	1.469	0.074
0.8	1.472	0.071
0.9	**1.478**	**0.068**
1.0	1.468	0.075

Note: The best result is highlighted in bold.

基于图 1 和表 1 的数据,可以确认当 $w\in[0.1,1]$时,CNAME 算法能够获得较高的平均奖赏和较低的平均悔值. 此外,根据表 1 可知,当 w 取值较为接近 1 时,比如 $w=0.9$ 时,获得的平均奖赏更大,平均每轮的悔值更小,即 CNAME 算法的效果更好.

仅以平均奖赏为例对图 1 和表 1 的实验结果进行分析,主要有 2 个原因:

1) 当 w 的取值过小时,探索概率的衰减速率过快,导致没有足够的探索,无法得到最高的累积奖赏.

① https://webscope.sandbox.yahoo.com

初始时刻 $m_0=0$,CNAME 算法探索的概率为 1,此时接近随机策略. 一定时间步之后,当 $m_t\neq 1$ 时,探索概率开始衰减,w 越小,衰减的速率越高. 例如当 $w=1$,$m_t=1$ 时,探索概率为 0.5;当 $m_t=2$ 时,探索概率降为 0.2,算法效果近似于 ε-decreasing. 而当 $w=0.01$,$m_t=1$ 时,探索概率为 0.009 9;当 $m_t=2$ 时,探索概率衰减到 0.002 5,这时算法效果类似于 greedy,可能会忽略更优的动作,从而不能获得最高的平均奖赏. 因此,w 取值过小,平均奖赏反而会有所下降.

2) 当 w 的取值较大时,探索概率的衰减速率过慢,导致过度探索,平均奖赏相对较小.

例如当 $w=100$,$m_t=1$ 时,探索概率为 0.990;$m_t=2$,探索概率为 0.962. m_t 的增加对探索概率的影响过小,不能及时地减少探索概率,导致了过多的探索,使得平均奖赏较小. 因此 w 取值过大,平均奖赏相对较小.

综上所述,参数 w 应根据实际环境适当取值,不宜过大也不宜过小. 根据实验结果,该环境下 CNAME 算法的参数 w 取闭区间[0.1,1]中的值较好.

3.2 实验 2:随机数据集上的比较

本节基于随机数据集,比较 CNAME 算法和其他 3 类经典算法及其变种的算法性能.

与实验 1 相似,本实验同样采用随机产生的服从正态分布的数据集:1 000 个随机产生的多臂赌博机任务. 具体地,$K=10$,$n=2\,000$;$\boldsymbol{\mu}=[\mu_1,\mu_2,\cdots,\mu_K]$服从均值为 0、方差为 1 的正态分布;每个动作 i 的奖赏服从一个均值为 μ_i、方差为 1 的正态分布. 根据实验 1 的结果,本次实验中,参数 $w=0.95$.

在上述实验条件下,分别记录各个算法的平均奖赏和平均每轮的悔值. 表 2 列出了实验中 6 种算法的参数设置、对应 1 000 个任务在 2 000 个时间步下的平均奖赏和平均每轮的悔值. 各个算法均采用的是该实验条件下效果较好的参数,表 2 中加粗部分是效果最好的结果.

Table 2 Parameters and Experimental Results of Six Algorithms

表 2 6 种算法的参数设置及实验结果

Algorithms	Parameters	Average Rewards	Average Regrets
ε-greedy	$\varepsilon=0.1$	1.338	0.185
ε-decreasing	$\varepsilon_0=10$	1.378	0.143
SoftMax	$\tau=0.2$	1.470	0.081
decreasing SoftMax	$\tau_0=20$	1.478	0.060
UCB1	$c=\sqrt{2}$	1.419	0.088
CNAME	$w=0.95$	**1.491**	**0.054**

Note: The best result is highlighted in bold.

从表 2 可以看出,该实验条件下,CNAME 算法获得的平均奖赏最高,平均每轮的悔值最低. 变种 ε-decreasing 算法的效果明显优于 ε-greedy 算法的效果;类似地,变种 decreasing SoftMax 算法的效果优于 SoftMax 算法.

为了简洁清晰地表示这几种算法的对比效果,图 2 和图 3 仅分别展示了 ε-decreasing,decreasing SoftMax,UCB1,CNAME 这 4 种算法运行至 1 000 个时间步时的实验数据. 图 2 是 CNAME 算法与其他 3 种算法平均奖赏的结果比较. 图 3 为 CNAME 算法与其他 3 种算法的悔值的比较,同时给出了 CNAME 算法平均悔值的上界(upper bound). 基于 2.3 节得出的悔值上界,upper bound 是平均每轮的悔值上界. 由图 3 可知,随着时间步的增加,upper bound 的值保持平稳,可见 CNAME 算法非常稳定. 另一方面,CNAME 算法各个时间步的平均悔值均在 upper bound 之下,符合 2.3 节的理论结论.

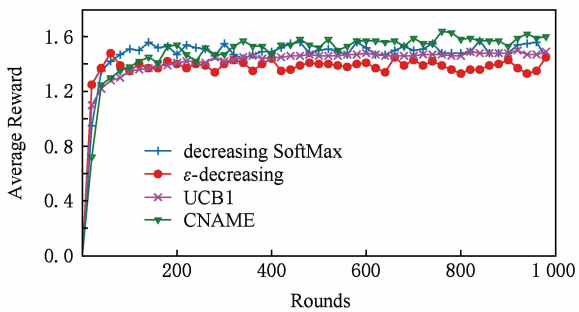

Fig. 2 Average rewards of CNAME and the other three algorithms

图 2 CNAME 算法与其他 3 种算法的平均奖赏

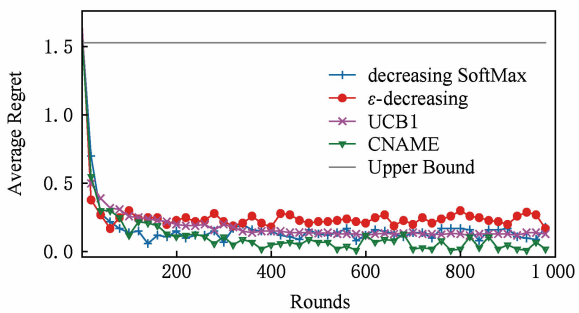

Fig. 3 Average regrets of CNAME and the other three algorithms

图 3 CNAME 算法与其他 3 种算法的平均每轮的悔值

表 2、图 2 和图 3 中的数据均是 1 000 个随机任务的平均值. 结合图表,本文从 3 方面进行比较:

1) CNAME 算法和 ε-greedy 算法及其变种

CNAME 算法的收敛速度略低于 ε-decreasing 算法,但其平均奖赏明显高于 ε-greedy 算法和 ε-decreasing 算法,平均每轮悔值也明显低于 ε-greedy 算法和 ε-decreasing 算法. CNAME 算法的效果与 ε-greedy 算法和 ε-decreasing 算法相比有了很大的提高.

这是因为 CNAME 算法利用了用户反馈,虽然收敛速度有所减小,但效果比探索概率固定不变的 ε-greedy 算法好很多. 同理,ε-decreasing 算法的探索概率虽然是随着时间降低的,但是没有利用选择动作之后的信息,因此其效果比 CNAME 算法差.

2) CNAME 算法和 SoftMax 算法及其变种

CNAME 算法的收敛速度略低于 decreasing SoftMax 算法,但其平均奖赏高于 SoftMax 算法和 decreasing SoftMax 算法,且平均每轮的悔值较低. 因此,CNAME 算法的效果与 SoftMax 算法和 decreasing SoftMax 算法相比有所提高.

这是因为与 SoftMax 算法相比,CNAME 算法除了考虑动作估计值,还考虑了动作被选择的次数,比 SoftMax 算法和 decreasing SoftMax 算法利用的信息多,虽然收敛速度略有降低,但不易被前期的估计值误导. 因此 CNAME 算法最终获得的平均奖赏和平均每轮悔值均优于 SoftMax 算法和 decreasing SoftMax 算法.

3) CNAME 算法和 UCB1 算法

CNAME 算法的收敛速度与 UCB1 算法相当,但其平均奖赏高于 UCB1 算法,且平均每轮的悔值低于 UCB1 算法. 值得注意的是,通过分析实验数据的变化,我们发现:CNAME 算法的数据波动较小,而 UCB1 算法在实验初期的数据会出现尖峰.

这是因为 CNAME 算法和 UCB1 算法均充分利用了选择动作之后的信息,两者收敛速度相当. 但是 UCB1 算法在初期的前 K 次操作中,先将每个动作轮流选择一次,在第 $K+1$ 次操作时选择贪心动作,因此 UCB1 算法会在第 $K+1$ 操作时出现尖峰. 而 CNAME 算法在第 $K+1$ 次操作时,依然根据当前的探索概率选择动作,因此 CNAME 算法的数据不会出现尖峰.

综上所述,CNAME 算法是一种有效的多臂赌博机算法.

3.3 实验 3:内容分发网络数据集上的比较

实验 3 采用的数据集对应于真实环境中可用资源冗余情况下的数据检索问题. 例如在内容分发网络中,agent 必须通过一个可用资源冗余的网络来检索数据. 假设每次检索 agent 只选择一个资源,直到检索到数据,agent 的目标是最小化检索的累积延时. 该问题可转换为随机多臂赌博机问题进行处理[27],3 项具体说明如下:

1) 实验 3 用超过 700 个学校的主页作为资源,每一个主页对应多臂赌博机中的一个臂(动作).

2) 这些主页约每隔 10 min 被检索一次,共记录了 10 d 的检索延迟信息(约 1 300 条),单位为 ms.

3) 每一个延时对应一个负奖赏,因此,平均延时越小,对应算法越有效.

上述延时数据来源于实际问题,数据范围广、波动大,而且隐含了实际环境中的各种干扰和影响. 本文分别利用各个算法对该数据集进行处理,更能体现各个算法性能的差异和各自的稳定性.

表 3 是 6 种算法平均每轮的检索延时,这些数据是 1 000 次模拟的平均结果. 考虑到实际环境的随机性,每次模拟中延时的顺序都是随机的.

Table 3 Average Latency of Six Algorithms

表 3 6 种算法的平均延时

Algorithms	Parameters	Average Latency/ms
ε-greedy	$\varepsilon=0.05$	427
	$\varepsilon=0.1$	494
	$\varepsilon=0.15$	475
ε-decreasing	$\varepsilon_0=5$	424
	$\varepsilon_0=10$	397
	$\varepsilon_0=15$	389
SoftMax	$\tau=0.1$	411
	$\tau=0.15$	410
	$\tau=0.2$	409
decreasing SoftMax	$\tau_0=15$	401
	$\tau_0=20$	392
	$\tau_0=25$	388
UCB1	$c=\sqrt{2}$	373
CNAME	$w=0.1$	**372**
	$w=0.5$	374
	$w=1$	376

Note: The best result is highlighted in bold.

从表 3 可以看出,使用 ε-greedy,ε-decreasing,SoftMax,decreasing SoftMax 这 4 种算法得到的平均延时明显高于 UCB1 和 CNAME 算法. 这主要是

因为关于延时的网络数据通常波动很大,而且延时数据的取值范围很广,可能是 10 ms,也可能是 1 000 ms. 对于这样的数据,需要非常小心地探索,因为尝试一个动作有可能得到很小的延时,但也有可能得到一个突增的延时. 这种情况下,UCB1 算法和 CNAME 算法的优势很明显,这 2 种算法同时根据动作的估计值和被选择的次数自适应地调整探索和利用,充分利用了反馈信息.

虽然 UCB1 算法和 CNAME 算法最后得到的平均延时相当,但 CNAME 算法只需要约 2 min 就可以完成计算,而 UCB1 算法则需要约 6 h. 这主要是因为 UCB1 算法相对复杂,计算负担较大,所以需要较长的计算时间;而 CNAME 算法的计算则相对简单,计算负担较小,极大地节约了计算时间.

可见,CNAME 算法可以更加高效地平衡随机多臂赌博机问题中的探索和利用.

3.4 实验 4:雅虎公司 R6A 数据集上的比较

在线内容推荐是交互式机器学习问题的重要内容,需要在探索和利用之间进行有效的平衡. 实验 4 将推荐系统建模为多臂赌博机模型. 雅虎公司 R6A 数据集包含 45 811 883 个用户访问雅虎公司首页 Today 模块的点击日志. 对于每次访问,用户和每个候选文章都有关联的 6 维的特征向量.

本节对比了 CNAME 算法与其他 3 种算法在雅虎公司 R6A 数据集上的推荐效果,实验结果如图 4 所示,纵坐标表示累计奖赏,对应推荐系统中的点击次数. 其中,Random 算法作为基准算法,每次为用户随机推荐物品,LinUCB 算法为上下文相关的多臂赌博机算法.

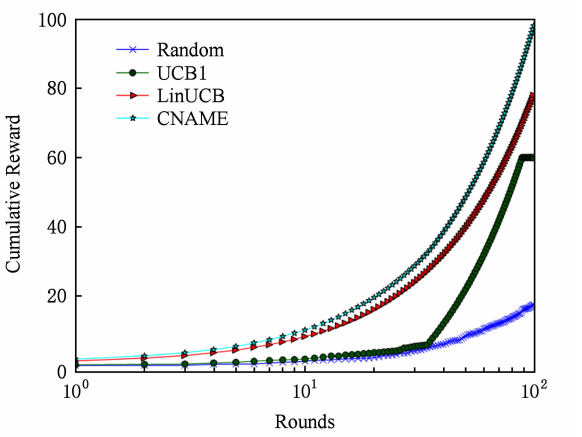

Fig. 4 Cumulative rewards of CNAME and the other three algorithms

图 4 CNAME 算法与其他 3 种算法的累计奖赏

从图 4 可以看出,Random 算法的累计奖赏最低,这是因为 Random 算法只是随机产生推荐,没有利用用户反馈信息. CNAME 算法获得的累计奖赏明显高于 UCB1 算法. UCB1 算法在初期将各个物品轮流推荐给用户,在物品数量较大的情况下不能及时地学习到用户的兴趣. 因此,UCB1 算法在初期获得的累计奖赏接近 Random 算法,后期获得的累计奖赏则快速增加.

LinUCB 算法作为经典的上下文相关的多臂赌博机算法,主要结合数据集中的 6 维特征向量进行推荐,在 100 个时间步后获得的累计奖赏为 78. 由于 LinUCB 算法依赖用户和物品的特征向量,而不同推荐系统中的特征向量维度和含义不同,LinUCB 算法无法方便地推广到不同的应用场景. CNAME 算法基于探索概率,在利用用户反馈进行推荐的同时,探索可能为用户带来惊喜的物品. 因此 CNAME 算法获得的累计奖赏很高且稳定增长,100 个时间步后获得的累计奖赏高达 98,累计奖赏高于基于上下文信息的 LinUCB 算法. CNAME 算法在推荐的过程中,一方面根据用户反馈,自适应地调整探索概率,另一方面不依赖上下文信息,可以方便地应用到各个场景中.

4 总 结

本文提出了一种自适应的随机多臂赌博机算法,即 CNAME 算法. 该算法充分利用选择动作之后获得的信息,同时考虑动作的估计值和被选择的次数,简便高效地解决了随机多臂赌博机中探索和利用的平衡问题. CNAME 算法根据当前估计值最小的动作(即实际获得的平均奖赏最小的动作)已经被选择的次数 m_i 和参数 w 共同控制探索的概率,自适应地根据实际环境调整探索和利用的程度. 同时,CNAME 算法在探索时选择目前最少被选择的动作,避免了随机选择动作的盲目性.

一方面,本文通过理论分析和证明给出了 CNAME 算法的悔值上界,为 CNAME 算法的性能提供了有力的理论支持. 另一方面,本文通过 4 组实验讨论了 CNAME 算法的关键参数 w 的影响及参考取值范围并分别在随机数据集,内容分发网络数据集和带有上下文信息的雅虎公司 R6A 数据集上与其他 3 类经典算法及其变种进行了对比实验. 实验表明 CNAME 算法与 ε-greedy 算法和 ε-decreasing 算法相比有了很大的提高,效果比 SoftMax 算法和 decreasing SoftMax 算法略好,比 UCB1 算法更高效.

此外,CNAME 算法的泛化能力较强,在雅虎公司 R6A 数据集上获得了与 LinUCB 算法相当甚至更好的效果.综上所述,CNAME 算法是一种效果稳定、高效且泛化能力强的多臂赌博机算法.

未来工作尝试将 CNAME 算法应用到更加复杂的多臂赌博机问题中,比如带有预算的多臂赌博机问题[30]、定性的多臂赌博机问题等[31];同时,探讨 CNAME 算法解决复杂问题的可能性,尝试提出有实际应用的新型多臂赌博机问题.此外,考虑与深度学习相结合,生成深度网络多臂赌博机,以解决更加复杂的问题.

参 考 文 献

[1] Sutton R, Barto G. Reinforcement Learning: An Introduction [M]. Cambridge, MA: MIT Press, 1998: 1-20

[2] Gao Yang, Zhou Ruyi, Wang Hao, et al. Study on an average reward reinforcement learning algorithm [J]. Chinese Journal of Computers, 2007, 30(8): 1372-1378 (in Chinese)
(高阳,周如益,王皓,等. 平均奖赏强化学习算法研究[J]. 计算机学报, 2007, 30(8): 1372-1378)

[3] Zhao Fengfei, Qin Zheng. A multi-motive reinforcement learning framework [J]. Journal of Computer Research and Development, 2013, 50(2): 240-247 (in Chinese)
(赵凤飞,覃征. 一种多动机强化学习框架[J]. 计算机研究与发展, 2013, 50(2): 240-247)

[4] Liu Quan, Fu Qiming, Yang Xudong, et al. A scalable parallel reinforcement learning method based on intelligent scheduling [J]. Journal of Computer Research and Development, 2013, 50(4): 843-851 (in Chinese)
(刘全,傅启明,杨旭东,等. 一种基于智能调度的可扩展并行强化学习方法[J]. 计算机研究与发展, 2013, 50(4): 843-851)

[5] Liu Zhibin, Zeng Xiaoqin, Liu Huiyi, et al. A heuristic two-layer reinforcement learning algorithm based on BP neural networks [J]. Journal of Computer Research and Development, 2015, 52(3): 579-587 (in Chinese)
(刘智斌,曾晓勤,刘惠义,等. 基于 BP 神经网络的双层启发式强化学习方法[J]. 计算机研究与发展, 2015, 52(3): 579-587)

[6] Robbins H. Some aspects of the sequential design of experiments [J]. American Mathematical Society, 1952, 58(5): 527-535

[7] Devanur N R, Kakade S M. The price of truthfulness for pay-per-click auctions [C] //Proc of ACM Conf on Electronic Commerce. New York: ACM, 2009: 99-106

[8] Cheng Shi, Mao Luhong, Chang Peng. Multi-armed bandit recommender algorithm with matrix factorization [J]. Journal of Chinese Computer Systems, 2017, 38(12): 2754-2758 (in Chinese)
(成石,毛陆虹,常鹏. 融合矩阵分解的多臂赌博机推荐算法[J]. 小型微型计算机系统, 2017, 38(12): 2754-2758)

[9] Yu Yonghong, Gao Yang, Wang Hao, et al. Integrating user social status and matrix factorization for item recommendation [J]. Journal of Computer Research and Development, 2018, 55(1): 113-124 (in Chinese)
(余永红,高阳,王皓,等. 融合用户社会地位和矩阵分解的推荐算法[J]. 计算机研究与发展, 2018, 55(1): 113-124)

[10] Jain S, Gujar S, Zoeter O, et al. A quality assuring multi-armed bandit crowdsourcing mechanism with incentive compatible learning [C] //Proc of the 13th Int Conf on Autonomous Agents and Multi-Agent Systems. New York: ACM, 2014: 1609-1610

[11] Jain S, Narayanaswamy B, Narahari Y. A multi-armed bandit incentive mechanism for crowdsourcing demand response in smart grids [C] //Proc of the 28th AAAI Conf on Artificial Intelligence and the 26th Innovative Applications of Artificial Intelligence Conf. Menlo Park, CA: AAAI, 2014: 721-727

[12] Sun Xinxin. Research on task assignment technology in crowdsourcing environment [D]. Yangzhou: Yangzhou University, 2016 (in Chinese)
(孙信昕. 众包环境下的任务分配技术研究[D]. 扬州:扬州大学, 2016)

[13] Zhou Baojian. Research on search engine keyword optimal selection strategy based on multi-armed bandit [D]. Harbin: Harbin Institute of Technology, 2014 (in Chinese)
(周保健. 基于多臂赌博机的搜索引擎关键字最优选择策略研究[D]. 哈尔滨:哈尔滨工业大学, 2014)

[14] Chen Hongcui. Research on channel selection mechanism based on multi-armed bandit in cognitive network [D]. Chongqing: Chongqing University of Posts and Telecommunications, 2016 (in Chinese)
(陈红翠. 认知网络中基于赌博机模型的信道选择机制研究[D]. 重庆:重庆邮电大学, 2016)

[15] Bubeck S, Cesabianchi N. Regret analysis of stochastic and nonstochastic multi-armed bandit problems [J]. Foundations and Trends in Machine Learning, 2012, 5(1): 1-22

[16] Slivkins A. Contextual bandits with similarity information [J]. Journal of Machine Learning Research, 2014, 15(1): 2533-2568

[17] Watkins C J C H. Learning from delayed rewards [J]. Robotics & Autonomous Systems, 1989, 15(4): 233-235

[18] Luce R D. Individual choice behavior [J]. American Economic Review, 1959, 67(1): 1-15

[19] Auer P, Cesa-Bianchi N, Fischer P. Finite-time analysis of the multiarmed bandit problem [J]. Machine Learning, 2002, 47(2/3): 235-256

[20] Li Lihong, Chu Wei, Langford J, et al. A contextual-bandit approach to personalized news article recommendation [C] //Proc of ACM Int Conf on World Wide Web. New York: ACM, 2010: 661-670

[21] Chu Hongmin, Lin Hsuan Tien. Can active learning experience be transferred? [C] //Proc of IEEE Int Conf on Data Mining. Piscataway, NJ: IEEE, 2017: 841-846

[22] Krishnamurthy B, Wills C, Zhang Yin. On the use and performance of content distribution networks [C] //Proc of the 1st ACM SIGCOMM Internet Measurement Workshop. New York: ACM, 2001: 169-182

[23] Thathachar M A L, Sastry P S. A new approach to the design of reinforcement schemes for learning automata [J]. IEEE Transactions on Systems Man & Cybernetics, 1985, SMC-15(1): 168-175

[24] Even-Dar E, Mannor S, Mansour Y. PAC bounds for multi-armed bandit and Markov decision processes [C] //Proc of the 15th Annual Conf on Computational Learning Theory (COLT). Berlin: Springer, 2002: 255-270

[25] Cesa-Bianchi N, Fischer P. Finite-time regret bounds for the multi-armed bandit problem [C] //Proc of the Int Conf on Machine Learning. New York: ACM, 1998: 100-108

[26] Strens M. Learning Cooperation and feedback in pattern recognition [D]. London: Physics Department, King's College London, 1999

[27] Vermorel J, Mohri M. Multi-armed bandit algorithms and empirical evaluation [C] //Proc of the European Conf on Machine Learning. Berlin: Springer, 2005: 437-448

[28] Kirkpatrick B S, Gelatt C, Vecchi D. Optimization by simulated annealing [J]. Science, 1983, 220(4598): 671-680

[29] Lai T L, Robbins H. Asymptotically efficient adaptive allocation rules [J]. Advances in Applied Mathematics, 1985, 6(1): 4-22

[30] Xia Yingce, Qin Tao, Ding Wenkui, et al. Finite budget analysis of multi-armed bandit problems [J]. Neurocomputing, 2017, 258: 13-29

[31] Szörényi B, Busa-Fekete R, Weng P. Qualitative multi-armed bandits: A quantile-based approach [C] //Proc of the 32nd Int Conf on Machine Learning. New York: ACM, 2015: 1660-1668

Zhang Xiaofang, born in 1980. PhD, associate professor, member of CCF. Her main research interests include reinforcement learning, machine learning, and software testing and analysis.

Zhou Qian, born in 1992. Master. Her main research interests include reinforcement learning, and deep reinforcement learning. (20154227029@stu.suda.edu.cn)

Liang Bin, born in 1993. Master. His main research interests include sentiment analysis, natural language processing, and deep learning. (bliang@stu.suda.edu.cn)

Xu Jin, born in 1992. Master. His main research interests include reinforcement learning, deep learning and deep reinforcement learning. (20154227016@stu.suda.edu.cn)