Model-based Constrained MDP for Budget Allocation in Sequential Incentive Marketing

Shuai Xiao*
Ant Financial Services Group
Shanghai, China
shuai.xsh@antfin.com

Le Guo*
Ant Financial Services Group
Beijing, China
guole.gl@antfin.com

Zaifan Jiang*
Ant Financial Services Group
Beijing, China
zaifan.jzf@antfin.com

Lei Lv
Ant Financial Services Group
Beijing, China
lvlei.ll@antfin.com

Yuanbo Chen
Ant Financial Services Group
Beijing, China
yuanbo.cyb@antfin.com

Jun Zhu
Ant Financial Services Group
Beijing, China
elizhu.zj@antfin.com

Shuang Yang†
Ant Financial Services Group
San Mateo, CA
shuang.yang@antfin.com

ABSTRACT

Sequential incentive marketing is an important approach for online businesses to acquire customers, increase loyalty and boost sales. How to effectively allocate the incentives so as to maximize the return (e.g., business objectives) under the budget constraint, however, is less studied in the literature. This problem is technically challenging due to the facts that 1) the allocation strategy has to be learned using historically logged data, which is counterfactual in nature, and 2) both the optimality and feasibility (i.e., that cost cannot exceed budget) needs to be assessed before being deployed to online systems. In this paper, we formulate the problem as a constrained Markov decision process (CMDP). To solve the CMDP problem with logged counterfactual data, we propose an efficient learning algorithm which combines bisection search and model-based planning. First, the CMDP is converted into its dual using Lagrangian relaxation, which is proved to be monotonic with respect to the dual variable. Furthermore, we show that the dual problem can be solved by policy learning, with the optimal dual variable being found efficiently via bisection search (i.e., by taking advantage of the monotonicity). Lastly, we show that model-based planing can be used to effectively accelerate the joint optimization process without retraining the policy for every dual variable. Empirical results on synthetic and real marketing datasets confirm the effectiveness of our methods.

*Authors contributed equally to this research.
†Corresponding author.

CIKM '19, November 3–7, 2019, Beijing, China

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358031

CCS CONCEPTS

• **Computing methodologies** → **Reinforcement learning**; **Sequential decision making**; • **Applied computing** → *Electronic commerce*;

KEYWORDS

Marketing Campaign, Reinforcement Learning, Recommendation, Constrained Resource Allocation

ACM Reference Format:
Shuai Xiao, Le Guo, Zaifan Jiang, Lei Lv, Yuanbo Chen, Jun Zhu, and Shuang Yang. 2019. Model-based Constrained MDP for Budget Allocation in Sequential Incentive Marketing. In *The 28th ACM International Conference on Information and Knowledge Management (CIKM'19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3358031

1 INTRODUCTION

Marketing with a form of incentives such as monetary prizes is a common approach especially in today's online internet industry. For example, in a typical online promotion, the owner of the campaign offers prizes such as coupons to encourage its target customers for certain favorable actions such as clicks or conversions. This campaign can be run only once such as in online advertising, or it can be run repetitively for multiple times throughout the lifecycle of a customer. The latter is becoming more important as companies are increasingly seeing the opportunities not only to acquire customers but also to increase their loyalty and boost sales. This problem is called sequential incentive marketing. Figure 1 shows a real-world example where our problem emerges from. During one of market campaigning activities, Alipay repetitively sends red envelopes (coupons) with different amount of money to its users for a couple of days. Each red envelope incurs certain cost if consumed. The objective of the platform is to maximize the user engagement (e.g., the total times that people consume the red envelopes) under a global budget constraint through personalized sequential incentive allocation.

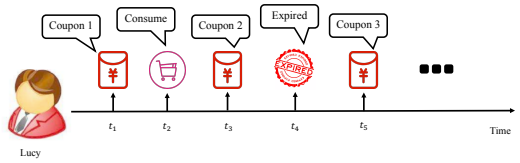


Figure 1: The illustrating example for constrained incentive allocation problem. During one of market campaigning activities, Alipay repetitively sends red envelopes with different amount of money to its users for a couple of days. Each red envelope incurs certain cost if consumed. The objective of the platform is to maximize the user engagement (e.g., the total times that users consume the red envelopes) under a global budget constraint through personalized sequential incentive allocation.

Sequential incentive marketing poses unique technical challenges. One of the difficulties of optimizing such problems is that only the feedback of the chosen recommendations (bandit feedback) is observed when multiple potential items for recommendation exist. Previous works [15, 24, 25] have studied learning from logged bandit feedback with the help of counterfactual policy optimization without constraints. Another challenge is that both the optimality and feasibility (i.e., that cost cannot exceed budget) needs to be assessed before being deployed to online systems. In the industrial setting, the allocation strategy should be learned and verified in an off-policy manner from logged data because on-policy learning of such strategies has uncontrollable risks as the budget can't be reverted once dispensed. Therefore, vanilla on-policy algorithms are not suitable here since they rely on realtime interaction with the industrial environment to collect feedback of the current policy. Furthermore, as the amount of data (i.e., billions of users and hundreds of items) is huge and the recommender system often include neural networks as modules, batch-training of such systems is usually necessary.

In this paper, we focus on sequential incentive recommendations with a global constraint, as is often the case in real-world industrial settings. For this problem, users are repetitively recommended certain items from a candidate set. Each item incurs a cost if consumed by users. The objective is to maximize expected rewards such as CVR while the cost doesn't exceed the global budget. To solve this problem, we first formulate it as constrained Markov decision process (CMDP) where MDP describes the repetitive recommendation process for each user. This sequential incentive allocation problem with constraint has been less studied before. Previous works [15, 24, 25] solve contextual bandit problems from logged bandit feedback without constraints. Lopez et al. [16] target at constrained bandit problem without considering the sequential allocation scenario where consecutive allocations have inter-dependence. Achiam et al. [1] extend trust region optimization method [23] to solve CMDP for high-dimensional control, which falls into the category of on-policy reinforcement learning. The optimization relies on on-line data collection from the interactions between the policy to optimize and the environment, which is inapplicable in our real systems.

To solve the CMDP problem with logged off-line data, we propose an efficient learning algorithm which combines bisection search and model-based planning. Firstly, the primary CMDP is converted into its Lagrangian dual problem. The Lagrangian is formed by adding the budget constraint multiplying by a Lagrangian multiplier (also called dual variable) to the original objective function. We prove that the cost of incentive allocation decreases monotonically as the dual variable increases. Therefore the optimal dual variable for the dual problem can be identified efficiently through bisection search. In the learning process, the policy would have to be retrained for every value of the dual variable, which could be extremely time-consuming. To alleviate the heavy computational cost, model-based planning is also employed which enables one-pass policy training during the whole learning process. The transformation from primary CMDP to the dual problem also makes batch-training of CMDP possible, which is important as the amount of training data is very large and allocation systems often include neural networks as modules.

The primary contributions of our work are the following. 1). For real system, the allocation strategy has to be learned from logged off-line data and verified before applied to online system. We propose a novel formulation to sequential incentive allocations problem which allows for strategy learning and verifying from logged off-line data. This formulation also allows for batch-training which is vital for deep neural networks and large-scale datasets. 2). Efficient learning algorithm is devised for the proposed formulation based on theoretical findings. To accelerate to the learning process, bisection search is used based on the theoretical finding that the cost of incentive allocation is monotonic to the dual variable. 3). Model-based planning is used for policy updating so that the policy can be trained only once during the dual variable searching process.

2 RELATED WORK

2.1 Batch Learning from Bandit Feedback

In real recommendation system, only feedback of executed actions is observed. Learning from logged feedback data, also called bandit problem, naturally belongs to counterfactual inference. It's necessary to consider the counterfactual risk when evaluating models and minimizing counterfactual risk becomes a reasonable objective because of incomplete observations. Most methods employ importance sampling-based estimators to calculate the counterfactual risk of new policies. Alekk et al. [2] propose a online learning algorithm for contextual bandit problem through iterative collections of feedback from real systems. As the amount of data in industry is huge, deep neural networks are often embedded as estimation modules for recommendation system which renders the batch-learning of such systems necessary. Thorsten et al. introduce Counterfactual Risk Minimization(CRM) principle and propose a series of works [24–26] performing batch learning from logged bandit feedback. SNIPS [26] is proposed to solve the propensity overfitting problem of CRM through self-normalization. Recently, they propose a deep learning based model called BanditNet [15], and convert the objective into a constrained optimization problem to allow the training neural network on a large amount of bandit data using stochastic gradient descent (SGD) optimization. Lopez et al. [16] introduce structured reward and HSIC to alleviate data collection bias, and use binary

search to find a deterministic policy which satisfies the budget constraint. All these methods don't consider sequential allocation setting and focus on contextual bandits problem where actions are independent. In this paper, we focus on sequential allocation problem where sequential actions are correlated.

2.2 Counterfactual Policy Evaluation

For industrial applications, evaluating new policies before applying to online systems is necessary to ensure the safety. For bandit problems or sequential decision problems where partial feedbacks are observed, counterfactual policy evaluation is usually employed to assess the expectation of newly-developed policies. Such evaluations are based importance-sampling where logged feedback data from an old policy π_b is served as a proxy to evaluate a new policy π [21] in the following equation:

$$E_\pi(f) = \int \pi * f(x)dx = \int \pi_b \frac{\pi}{\pi_b} * f(x)dx = E_{\pi_b}(\frac{\pi}{\pi_b} * f)$$

where f is the reward function.

To reduce the variance of vanilla importance sampling methods, Jiang et al. [14] extend doubly robust (DR) estimator for contextual bandits [7] to reinforcement learning. They combine value estimation with importance sampling to alleviate the problem of high variance. Thomas et al. [27] propose two methods to reduce the variance of DR at the cost of introducing a bias. Mehrdad et al. [8] reduce the evaluation variance further by directly minimizing the variance of the doubly robust estimator.

2.3 Constrained Policy Optimization

Sequential allocation problems with constraints are mostly formulated as the constrained Markov Decision Process(CMDP) [3]. Optimal policies for finite CMDP problem with known dynamics and finite states can be solved by linear programming. However, learning methods for high-dimensional control are lacking [1]. Achiam et al. [1] extend trust region optimization methods [23] to solve CMDP for continuous actions. Those methods rely on on-policy data collection from environment which is inapplicable in industrial settings as the policy isn't allowed to explore and learn from scratch in real system due to the unbearable cost. Di et al. [30] formulate budget constrained bidding as a CMDP problem and the Lagrangian multiplier are actions. They treat the constraint as parts of the environment, and search optimal Lagrangian multiplier sequences from logged data where lots of samples exist that the budget is completely consumed. This method is not applicable in the case where logged data doesn't have experience that the budget are consumed completely.

3 DUAL METHOD FOR CMDP

3.1 CMDP Formulation

We formulate the sequential cost-effective incentive allocation problem with budget constaints as the Constrained Markov Decision Process (CMDP), which can be represented as a $(S, A, P, R, C, \mu, \gamma,$ b) tuple:

- **S**: The state space describing the user context, such as the user features.
- **A**: The action space containing candidate items for allocation.
- **P**: A probability distribution: $S * A \rightarrow S$, describing the dynamic transition from current state to the next one after taking action a.
- **R**: $S \times A \rightarrow \mathbb{R}$ is the reward function which maps states and actions to certain real number.
- **C**: $S \times A \rightarrow \mathbb{R}$ is the cost function corresponding to the budget consumed.
- μ is the initial state distribution for state : S_0.
- $\gamma \in [0, 1]$ is the discount factor for future rewards, which means how important intermediate rewards are. If $\gamma = 1$, then rewards are all equally important.
- **b** is the global budget constraint, which the cost can't exceed.

To learn the CMDP problem, logged feedback data is collected from an old behavior policy π_b that interacted with the real system in the past. We assume π_b is stationary for simplicity. The logged data D are lists of tuples with six elements, consisting of observed state s_i , action $a_i \sim \pi_b(*|s_i)$, the propensity p_i defined as $\pi_b(a_i|s_i)$, the observed reward r_i, the cost c_i and the next state s_{i+1}.

$$D = [(s_0, a_0, p_0, r_0, c_0, s_0), ..., (s_n, a_n, p_n, r_n, c_n, s_{n+1})]$$

The goal of learning the CMDP problem is to find a policy $\pi(a|s)$ from D that maximizes an objective function, $J(\pi)$, which is usually a cumulative discounted reward, $J(\pi) = E_{\tau\sim\pi}[\sum_{t=0}^{T} \gamma^t R(s_t, a_t)]$ while the cost $J_C(\pi)$ doesn't exceed the budget constraint b where $J_C(\pi) = E_{\tau\sim\pi}[\sum_{t=0}^{T} \gamma^t C(s_t, a_t)]$. Here $\tau \sim \pi$ is shorthand for indicating that the distribution over trajectories depends on π: $s_0 \sim \mu, a_t \sim \pi, s_{t+1} \sim P$. To summarize, the CMDP can be formulated as the following equation:

$$\begin{aligned} p^* = \max_{\pi} \quad & J(\pi) \\ s.t. \quad & J_C(\pi) \leq b \end{aligned} \tag{1}$$

or equivalent equation:

$$\begin{aligned} p^* = \min_{\pi} \quad & -J(\pi) \\ s.t. \quad & J_C(\pi) \leq b \end{aligned} \tag{2}$$

3.2 Lagrangian Dual problem for CMDP

To efficiently solve the CMDP problem in Equation 2, we firstly convert the primary problem into a Lagrangian dual problem by adding the constraint term to the original objective function to form the Lagrangian of CMDP as follows:

$$L(\pi, \lambda) = -J(\pi) + \lambda(J_C(\pi) - b)$$

where λ is the Lagrangian multiplier.

Then the Lagrangian dual problem of CMDP can be formulated as the follows:

$$\begin{aligned} d^* = \max_{\lambda} \quad & g(\lambda) \\ s.t. \quad & \lambda \geq 0 \end{aligned} \tag{3}$$

where $g(\lambda) = \min_\pi -J(\pi) + \lambda(J_C(\pi) - b)$ is called Lagrangian dual function. Here we refer π as the primary variable and λ as the dual variable. For any $\lambda \geq 0$ and feasible π, $p^* \geq g(\lambda)$ always holds which means that $g(\lambda)$ is a lower bound of p^*.

In the following, we first prove that the solution of the Lagrangian dual problem of CMDP in Equation 3 exists. Then the learning algorithm of the Lagrangian dual problem of CMDP is given. Before proceeding into the proof of existence of the Lagrangian dual problem, we need to give basic definitions and assumptions that our proof relies on.

Defination 1. *$\pi_l(s) = \arg\max_a C(s,a)$: The lowest-cost policy always chooses actions with minimum cost at every state. $\pi_h(s) = \arg\max_a C(s,a)$: The highest-cost policy always chooses actions with maximum cost at every state.*

Assumption 1. *Budget constraint under the lowest-cost policy π_l is strictly feasible: $J_C(\pi_l) < b$.*

Assumption 2. *The cost exceeds the budget constraint under the highest-cost policy π_h: $J_C(\pi_h) > b$.*

Proposition 1 (Existence of Solutions). *When constraint satisfies assumption 1 and 2, then the optimal value for dual variable λ in Equation 3 always exists and is positive, $\lambda^* > 0$.*

Proof. Under the assumption 1, a policy that doesn't exceed the budget exists and therefore the primary and dual problem always have a strictly feasible solution. Under the assumption 2, it's obvious that there always exists a positive optimal Lagrangian multiplier λ which penalizes the budget constraint violation. If the cost of high-cost policy doesn't exceed the budget, then the CMDP problem degrades into MDP without constraints and the optimal dual variable is always zero. In conclusion, based on the two natural assumptions which hold in practical problems, we have the above proposition. □

3.3 Solving the Dual Problem

After proving the existence of solutions of the primary and dual problem, we iteratively improve the lower bound by optimizing the dual problem instead. To solve the dual problem in equation 3, dual ascent method [4] can be used, where primary variable and dual variable can be optimized alternatively via gradient ascent. In practice, the computational cost of dual ascent method is extremely high as for each dual variable λ, we need to learn the optimal policy π and for each π, we need to run counterfactual policy evaluation (CPE) to compute the sub-gradient of λ. To reduce the heavy computational cost, we develop an novel algorithm which can efficiently identify optimal dual variable λ based on theoretical deductions. We first derive that the cost of incentive allocation is monotonic to the Lagrangian multiplier λ. Then a faster bisection method is proposed to determine the optimal dual variable λ^*.

Theorem 1. *Let Lagrangian $L(\pi,\lambda) = -J(\pi) + \lambda(J_C(\pi) - b)$, and $\pi_\lambda = \arg\max_\pi L(\pi,\lambda)$. If $\lambda_a > \lambda_b$, then $J_C(\pi_{\lambda_a}) \le J_C(\pi_{\lambda_b})$. That's to say, $J_C(\pi)$ is monotonic with λ.*

Proof. Because π_{λ_a} and π_{λ_b} is the minimizer of Lagrangian dual function, therefore

$$-J(\pi_{\lambda_a}) + \lambda_a(J_C(\pi_{\lambda_a}) - b) \le -J(\pi_{\lambda_b}) + \lambda_a(J_C(\pi_{\lambda_b}) - b)$$

$$-J(\pi_{\lambda_b}) + \lambda_b(J_C(\pi_{\lambda_b}) - b) \le -J(\pi_{\lambda_a}) + \lambda_b(J_C(\pi_{\lambda_a}) - b)$$

Adding the two inequalities

$$\begin{aligned} &-J(\pi_{\lambda_a}) - J(\pi_{\lambda_b}) + \lambda_a(J_C(\pi_{\lambda_a}) - b) + \lambda_b(J_C(\pi_{\lambda_b}) - b) \\ \le &-J(\pi_{\lambda_b}) - J(\pi_{\lambda_a}) + \lambda_a(J_C(\pi_{\lambda_b}) - b) + \lambda_b(J_C(\pi_{\lambda_a}) - b) \\ &\Downarrow \\ &(\lambda_a - \lambda_b)(J_C(\pi_{\lambda_a}) - J_C(\pi_{\lambda_b})) \le 0 \end{aligned}$$

because $\lambda_a > \lambda_b$, which leads to the following conclusion:

$$J_C(\pi_{\lambda_a}) \le J_C(\pi_{\lambda_b})$$

Proof finishes. □

With the result that the cost of incentive allocation is monotonic with dual variable λ, the dual problem can be solved much faster than dual ascent method [4]. We firstly design a mechanism to identify optimal dual variable based on bi-section search, which empirically converges faster than dual ascent method, consequently reducing computational time greatly. The dual ascent usually has near-linear convergence rate while the bisection search can have exponential convergence rate. We firstly compute the lower bound λ_l and upper bound λ_u of dual variable λ. Then we learn the policy π_{λ_m} from logged off-line data by fixing the dual variable λ_m where $\lambda_m = \frac{\lambda_l+\lambda_u}{2}$ is the middle point lying between lower and upper bound. Then the pseudo sub-gradient of λ at the middle point λ_m is obtained through running counterfactual policy evaluation (CPE) over logged data to compute $J_C - b$, which indicates the direction of next move for dual vaiable λ. In summary, the details of dual problem learning are given in Algorithm 1.

Algorithm 1 Dual Method Learning Framework for Constrained Markov Decision Processes

Require: Initial policy π_0, logged training data D_{train}, logged evaluation data D_{val}.
Require: Lower bound λ_l and upped bound λ_u of dual variable λ.
Require: Policy learning algorithm *PG*, and Counterfactual policy evaluation algorithm *CPE*.
while $\lambda_l < \lambda_u$ **do**
  Identify middle point of dual variable $\lambda_m = \frac{\lambda_l+\lambda_u}{2}$
  Learning policy π_{λ_m} using algorithm *PG* from logged data D_{train} with fixed λ_m.
  Run *CPE* to get counterfactual cost J_C = *CPE* (D_{val}, π_{λ_m}).
  Compute pseudo sub-gradient of λ to determine the search direction.
  if $J_C \le b$ and $|J_C - b| < \delta$ **then**
    break
  end if
  if $J_C < b$ **then**
    $\lambda_l = \lambda_m$
  else
    $\lambda_u = \lambda_m$
  end if
end while
return Optimal dual variable λ_m and policy π_{λ_m}.

The policy learning *PG* in Algorithm 1 will be introduced in detail in Section 4.1. To circumvent retraining the policy for each

dual variable λ, a model-based planning algorithm is proposed to accelerate the learning process in Section 4.2. To make the main idea of the paper coherent, we move the details of counterfactual policy evaluation (CPE) and identification of upper bound λ_u of dual variable into Appendix A and B.

4 POLICY LEARNING WITH DUAL VARIABLE

For dual method learning in Algorithm 1, we need to update policy for each λ. In this section, we firstly derive a variant of DQN as policy learning methods and then add entropy regularizer to improve both exploration and robustness in challenging decision-making tasks [9]. For the above approach, the policy has to be re-trained for each each λ, which is very time-consuming. To alleviate this problem, a model-based approach is firstly proposed to accelerate the training process. We'll give the details of policy learning and model-based acceleration in the following.

4.1 Policy Learning

As in our case that the cost is incurred for each action taken which means that the cost and reward have the same distribution depending on policy π, so the cost can be subsumed into the reward as shown in the following equation.

$$\begin{aligned} L(\pi, \lambda) &= -J(\pi) + \lambda(J_C(\pi)) \\ &= -\mathbb{E}_{\tau\sim\pi(\tau)}[r(\tau)] + \lambda\big(\mathbb{E}_{\tau\sim\pi(\tau)}[c(\tau)] - b\big) \\ &= -\mathbb{E}_{\tau\sim\pi(\tau)}\left[\sum_{i=1}^{T}\underbrace{(r(s_t,a_t) - \lambda c(s_t,a_t))}_{\text{new reward}}\right] - \lambda b \end{aligned} \tag{4}$$

As a result, we can use the new reward $rc(s,a) = r(s,a) - \lambda c(s,a)$ to learn a DQN policy [18, 28] given λ. One benefit of this reformulation is that the new reward has reasonable interpretation that the policy taking actions with high rewards and low costs is desired. Another benefit is that this reformulation makes the batch-learning of DQN easier as the cost term disappears so we can sampling a batch of samples $\{(s_i, a_i, p_i, r_i, c_i, s_{i+1})\}$ for training as done in traditional DQN.

To improve both exploration and robustness in challenging decision-making tasks [9], we can also add entropy regularizer to the above reward-reshaped DQN. The entropy regularizer makes the policy to take actions as diverse as possible. In this way, we get a more stochastic DQN, which falls into the category of Soft-Q based algorithms [11, 22]. The objective function after adding an entropy regularizer to the policy becomes:

$$\begin{aligned} J_\rho(\pi) \quad &:= \mathbb{E}_{\tau\sim\pi(\tau)}[rc(\tau) - \rho g(\tau)] \\ \text{where} \quad & rc(\tau) = r(\tau) - \lambda c(\tau) \\ & g(\tau) = \sum_{i=0}^{T}\gamma^i \log\pi(a_i|s_i) \end{aligned} \tag{5}$$

The optimal policy for entropy-regularized policy in equation 5 can be derived in a similar way to PCL [19] and is given directly as bellow:

$$\begin{aligned} Q^*(s,a) &= rc(s,a) + \gamma V^*(s') \\ V^*(s) &= \rho \log \sum_a \exp\{\frac{Q^*(s,a)}{\rho}\} \\ \pi^*(a|s) &= \exp\{\frac{Q^*(s,a) - V^*(s)}{\rho}\} \end{aligned} \tag{6}$$

From above equation, if ρ is close to zero, then the stochastic DQN degrades into original DQN without entropy regularizer, where the action with the largest Q value is taken. If ρ is infinitely large, then all actions have the same probability to be taken. The details of policy learning are summarized in Algorithm 2.

Algorithm 2 Policy Learning with Dual Variable

Require: Dual variable λ, logged training data D_{train}.
1: Calculate reshaped reward $rc(s,a) = r(s,a) - \lambda c(s,a)$.
2: Policy learning using batch SGD with loss 4 or 5.
3: Prob(s,a) = $\pi_\lambda(s,a)$ during inference. ▷ Only used in model inference
4: **return** Optimal policy π_λ

4.2 Model-based Acceleration

Even using bi-section search for λ, it's very time-consuming for re-training policy for multiple times. To alleviate the re-training problem, we can turn to the idea of model predictive control (MPC). If we have the dynamic model of the environment, we can use tree search based planning algorithm by expanding the full search tree from the current state to derive the optimal policy. To this end, we first learn the environment model from logged data and then derive optimal policy through tree-search based model planning.

Firstly, the cost and reward model, and state transition model are learned from logged data:

$$\begin{aligned} \text{Transition model:} \quad & T(s_t, a_t) \rightarrow s_{t+1} \\ \text{Reward model:} \quad & R(s_t, a_t) \rightarrow r_t \\ \text{Cost model:} \quad & C(s_t, a_t) \rightarrow c_t \end{aligned}$$

Lots of ways exist to model transition dynamics, like neural network [20], gaussian processes [6] and latent dynamics [10, 12, 29]. Here we use a feed-forward deep neural network with a regression loss to learn the dynamics. The reward model, typically a CTR prediction problem, can be learned with logistic regression [17], logistic regression with GBDT [13] or Wide&Deep model [5]. In our paper, Wide&Deep model is used as its flexibility suits our high-dimensional user feature. The cost model is learned with Wide&Deep model with a regression loss.

With the learned model, we can perform model predictive control (MPC) to obtain policy π. In this work, we simply traverse the search tree over a finite horizon H to calculate the Q values using the learned dynamics. Note that for higher dimensional action spaces and longer horizons, Monte Carlo sampling may be insufficient, which we leave for future work. The overall procedure of model-based approach is presented in Algorithm 3.

This combination of learned model dynamics model and model predictive planning is beneficial in that the model is trained only

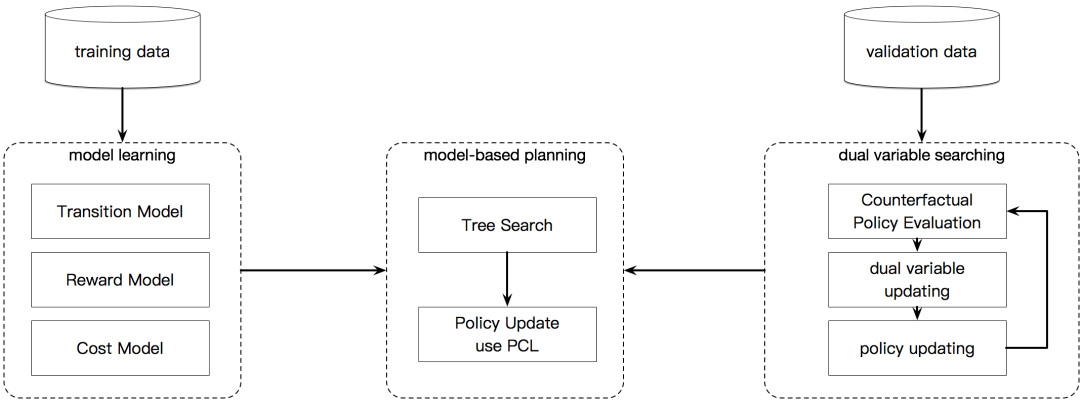


Figure 2: The workflow of the proposed model-based acceleration. The main process shown in the right column is a closed loop for using dual ascent to solve Lagrangian dual problem for constrained MDP which contains dual variable identification and policy learning. The sub-gradient of the Lagrangian multiplier (dual variable) is calculated using counterfactual policy evaluation. The middle column corresponds to model-based acceleration and the right column shows the data flow for learning environment model.

Algorithm 3 Policy Learning with Dual Variable using Model-based Planning Acceleration

Require: Dual variable λ, entropy regularizer coefficient ρ, planning horizon h.
Require: Transition model $T(s_t, a_t)$, reward model $R(s_t, a_t)$, cost model $C(s_t, a_t)$.
1: Set $Q(s, a) = 0$ for any s, a.
2: **for** i = 1 ... N **do**
3: Traverse local tree with depth h from state s using learned dynamic T.
4: Calculate $Q(s, a), V(s)$ using equation 6.
5: Calculate the policy $\pi^*(a|s) = \exp\{\frac{Q^*(s,a)-V^*(s)}{\rho}\}$
6: **end for**
7: **return** Policy π^*

once. By simply changing the λ, we can accomplish a variety of policy validation effectively, without a need for λ-specific retraining.

The overall workflow of the model-based acceleration is shown the Figure 2. The main process shown in the right column is a closed loop for using dual ascent to solve Lagrangian dual problem for constrained MDP which contains dual variable identification and policy learning. The sub-gradient of the Lagrangian multiplier (dual variable) is calculated using counterfactual policy evaluation. The middle column corresponds to model-based acceleration and the right column shows the data flow for learning environment model.

5 EXPERIMENTAL RESULTS

We conduct experiments on synthetic and real-world data. To testify the soundness of the proposed method, synthetic data which satisfies the three assumptions is simulated. For real-world data, its characteristics are firstly analyzed to check the validation of the assumptions and then the performances of the proposed method are demonstrated.

5.1 Baselines

To show the benefits of the proposed approach, several alternatives are used to demonstrate the advantages and disadvantages of different approaches, which are briefly introduced as follows:

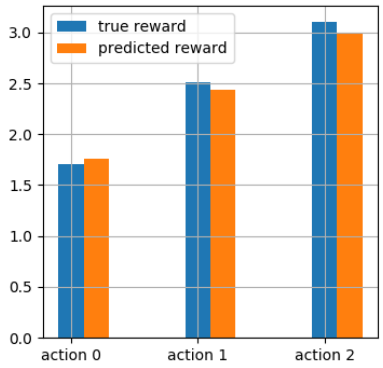
- Random: Randomness is a simple and effective strategy to allocate incentives to customers when recommendation system works with a cold start. The policy allocate all items to customers at an uniform probability.
- Constrained contextual bandit: Constrained contextual bandit constitutes one-step Constrained MDP where correlations between steps are not considered.
- Constrained MDP: Constrained Markov decision process is solved with the proposed fast bisection search with and without model-based acceleration.

5.2 Synthetic data

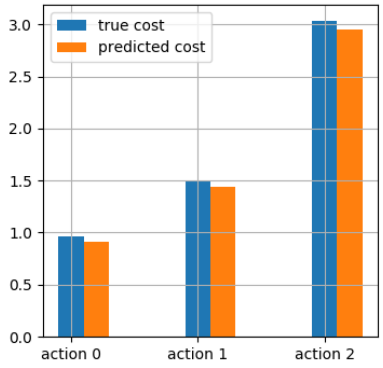
Dataset Description We generate a two-step sequential contextual bandit dataset and actions are selected with a uniform distribution. All states in the first step are 0, and next states are $i + 1$ when taking action $i, i = 0, ...n$. The reward for state s action i is sampled from a gaussian distribution with mean reward $r[s, i]$ and variance v. The cost for state s and action i is sampled from a gaussian distribution with mean cost $c[s, i]$ and variance 0.1. We set $r[s, i] = r[0, s] + (i - s) * \beta_r$, $c[s, i] = r[0, s] + (i - s) * \beta_c$. Therefore, rewards $r[s, 0]...r[s, n]$ and costs $c[s, 0]...c[s, n]$ are in ascending order.

Result For the model-based approach, the reward and cost model should be learned at first. The true average reward and the predicted reward for hold-out validation data from the reward model are shown in Figure 3a. The reward model can predict the rewards for different users accurately with an acceptable error. The true average cost and the predicted cost for hold-out validation data from the cost model are shown in Figure 3b. The prediction accuracy for costs is very high. Note that there is a trade-off between speeding up of algorithm through model-based planning and the performance of

the dual CMDP problem. If the prediction accuracy of model-based planning doesn't reach the acceptable criterion, we can roll back to use the original model without model-based acceleration.



(a) The prediction accuracy for the reward model. Average ground-truth and predicted reward over the holdout validation dataset are shown.



(b) The prediction accuracy for the cost model. Average ground-truth and predicted cost over the holdout validation dataset are shown.

Figure 3: The prediction accuracy of the reward and cost model in the model-based approach.

For constrained MDP, we compare the performance and computational complexity for the proposed fast bisection search and model-based acceleration learning algorithms. The performances of the two learning algorithms are shown in Figure 4. The two learning algorithms produce almost the same rewards under different budgets while the model-based acceleration doesn't need to re-train the policy for each λ. The computational cost of bi-section search is $O(n)$ times as much as model-based acceleration where n is the search times for λ. Except for explicitly stated, the performance for constrained MDP is for the model-based acceleration in the later discussion.

The performances of constrained contextual bandit and constrained MDP approaches are shown in Figure 5. The rewards and

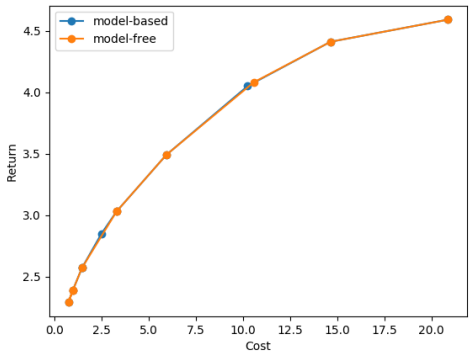


Figure 4: The performance of constrained MDP solved with the proposed fast bisection search and model-based acceleration learning algorithms. The "model-free" represents the bisection search and "model-based" represents model-based acceleration.

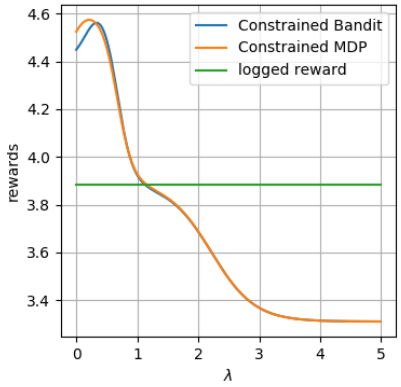
costs are computed over the holdout validation dataset using the models learned under different λ. The cost curve for the dual problem of constrained MDP in Figure 5b decreases monotonically as the λ increases. This fact empirically verifies the Theorem 1. The reward curves for the dual problem of constrained MDP in Figure 5a almost decreases with λ. This is because that the costs decrease with λ, so the rewards decrease together with the costs as stated in Assumption 3. As we can see, the reward curves for constrained bandit and MDP are almost overlapped. Therefore, for a fixed budget b, the λ for constrained bandit is smaller than constrained MDP and the reward for constrained MDP is larger than that of constrained bandit. The Table 1 shows the performances of different approaches under a fixed budget $b = 2.9295$ on the synthetic dataset. The reward of constrained bandit has a 13.31% increase over the random approach. The proposed constrained MDP has an 16.11% improvement over the random approach.

Table 1: Performances of different approaches under a fixed budget on the synthetic dataset.

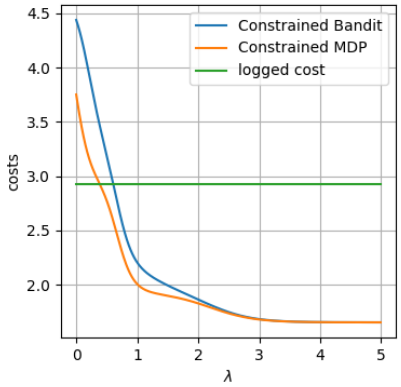
Algorithm	cost	reward
Random	2.9295	3.8836
Constrained contextual bandit	2.9295	4.4006(**+13.31%**)
Constrained MDP	2.9295	4.5093(**+16.11%**)

5.3 Real-world Dataset

Dataset Description The Dataset comes from Alipay which is one of China's largest payment platforms owned by Ant Financial. During one of market campaigning activities, Alipay repetitively sends red envelopes to its users for a couple of days. The purpose is to improve users' engagement and activeness. This real world data is used to testify our algorithm. In the dataset, each record contains user features, the amount of red envelope. We take the amount of red envelope as action and whether the user is active in the next



(a) Reward curves of different approaches regarding to λ. Rewards are computed over the holdout validation dataset using the models learned under different λ.



(b) Cost curves of different approaches regarding to λ. Costs are computed over the holdout validation dataset using the models learned under different λ.

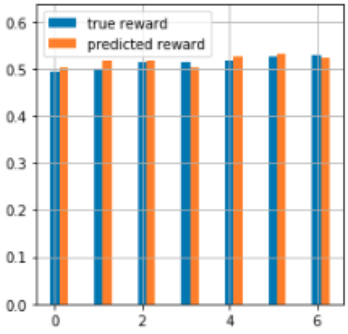
Figure 5: The performances of constrained contextual bandit and constrained MDP approaches.

day as reward. The amount of red envelop is also the cost incurred by that action. This application obeys the three assumptions. Especifically, for Assumption 3, the more money in the red envelope, the higher probability users will be active in the next day.

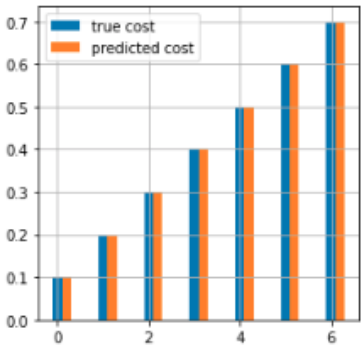
Results Figure 6 shows the accuracy of learning the model dynamics, which include the reward model and cost model. As shown, the predicting errors of reward and cost model are relevantly very small in our case. Therefore, the error introduced by model-based planning is small enough which won't deteriorate the performance of model-based acceleration algorithm too much.

Figure 7a and Figure 7b describes relationships between reward(cost) and lambda. The reward decreases as λ increases. We can see cost decreases when λ increases which means the Theorem 1 hold for this real dataset.

The Table 2 shows the performances of different approaches under a fixed budget $b = 0.532$ on the real dataset. The reward of



(a) Prediction accuracy of reward



(b) Prediction accuracy of cost

Figure 6: The prediction accuracy for the cost model. Average ground-truth and predicted cost over the holdout validation real dataset are shown.

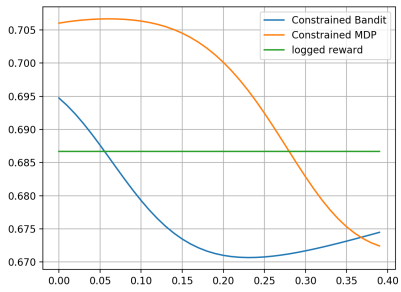
constrained bandit has a 8.97% increase over the random approach. The proposed constrained MDP outperforms alternatives over a notable margin, 13.61% improvement over the random approach. From another point of view, we evaluate the cost by fixing the campaigning reward. By setting the target reward to 0.687, constrained MDP can achieve the goal with a cost 9.40% lower than the random approach which constrained contextual bandit needs more budget.

Table 2: Performances of different approaches under a fixed budget on the real-world dataset.

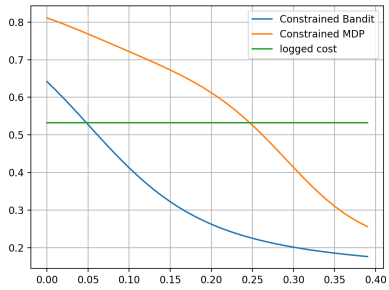
Algorithm	cost	reward
Random	0.532	0.687
Fixed budget		
Constrained contextual bandit	0.532	0.688(**+0.145%**)
Constrained MDP	0.532	0.693(**+0.873%**)
Fixed reward		
Constrained contextual bandit	0.503 (**-5.46%**)	0.687
Constrained MDP	0.482 (**-9.40%**)	0.687

6 CONCLUSION

This paper presents an efficient solution framework for sequential incentives allocation with the budget constraint which employs



(a) Reward curves of constrained MDP approach regarding to λ. Rewards are computed over the holdout validation dataset using the models learned under different λ.



(b) Cost curve of constrained MDP approach regarding to λ. Costs are computed over the holdout validation dataset using the models learned under different λ.

Figure 7: The performances of constrained MDP approach regarding to λ.

bisection search and model-based planning to solve the CMDP problem with logged counterfactual data. Empirical results on synthetic and real industrial data show its superior performances compared to alternatives. For future work, we plan to explore ways to jointly optimize the primary variable π and dual variable λ as in Equation 3, which can converge faster to optimal solutions without model-based planning. Another interesting direction is to use model-based approach to improve data-efficiency in reinforcement learning as collecting data from real systems is often costly or restricted due to practical concerns.

REFERENCES

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. 2017. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 22–31.

[2] Alekh Agarwal, Daniel Hsu, Satyen Kale, John Langford, Lihong Li, and Robert Schapire. 2014. Taming the monster: A fast and simple algorithm for contextual bandits. In *International Conference on Machine Learning*. 1638–1646.

[3] Eitan Altman. 1999. *Constrained Markov decision processes*. Vol. 7. CRC Press.

[4] Stephen Boyd and Lieven Vandenberghe. 2004. *Convex optimization*. Cambridge university press.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. ACM, 7–10.

[6] Marc Deisenroth and Carl E Rasmussen. 2011. PILCO: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*. 465–472.

[7] Miroslav Dudík, Dumitru Erhan, John Langford, Lihong Li, et al. 2014. Doubly robust policy evaluation and optimization. *Statist. Sci.* 29, 4 (2014), 485–511.

[8] Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. 2018. More Robust Doubly Robust Off-policy Evaluation. In *International Conference on Machine Learning*. 1446–1455.

[9] Peter Geibel and Fritz Wysotzki. 2005. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research* 24 (2005), 81–108.

[10] David Ha and Jürgen Schmidhuber. 2018. World models. *arXiv preprint arXiv:1803.10122* (2018).

[11] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. 2017. Reinforcement learning with deep energy-based policies. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1352–1361.

[12] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2018. Learning latent dynamics for planning from pixels. *arXiv preprint arXiv:1811.04551* (2018).

[13] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. ACM, 1–9.

[14] Nan Jiang and Lihong Li. 2016. Doubly Robust Off-policy Value Evaluation for Reinforcement Learning. In *International Conference on Machine Learning*. 652–661.

[15] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. 2018. Deep learning with logged bandit feedback. (2018).

[16] Romain Lopez, Chenchen Li, Xiang Yan, Junwu Xiong, Michael I Jordan, Yuan Qi, and Le Song. 2019. Cost-Effective Incentive Allocation via Structured Counterfactual Inference. *arXiv preprint arXiv:1902.02495* (2019).

[17] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1222–1230.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529.

[19] Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. 2017. Bridging the gap between value and policy based reinforcement learning. In *Advances in Neural Information Processing Systems*. 2775–2785.

[20] Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. 2018. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 7559–7566.

[21] Doina Precup. 2000. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series* (2000), 80.

[22] John Schulman, Xi Chen, and Pieter Abbeel. 2017. Equivalence between policy gradients and soft q-learning. *arXiv preprint arXiv:1704.06440* (2017).

[23] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. 2015. Trust region policy optimization. In *International Conference on Machine Learning*. 1889–1897.

[24] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *Journal of Machine Learning Research* 16, 1 (2015), 1731–1755.

[25] Adith Swaminathan and Thorsten Joachims. 2015. Counterfactual risk minimization: Learning from logged bandit feedback. In *International Conference on Machine Learning*. 814–823.

[26] Adith Swaminathan and Thorsten Joachims. 2015. The self-normalized estimator for counterfactual learning. In *advances in neural information processing systems*. 3231–3239.

[27] Philip Thomas and Emma Brunskill. 2016. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*. 2139–2148.

[28] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. 2015. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581* (2015).

[29] Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. 2015. Embed to control: A locally linear latent dynamics model for control from raw images. In *Advances in neural information processing systems*. 2746–2754.

[30] Di Wu, Xiujun Chen, Xun Yang, Hao Wang, Qing Tan, Xiaoxun Zhang, Jian Xu, and Kun Gai. 2018. Budget constrained bidding by model-free reinforcement learning in display advertising. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 1443–1451.

A COUNTERFACTUAL POLICY EVALUATION (CPE)

The details of Counterfactual Policy Evaluation is given in Algorithm 4.

Algorithm 4 Counterfactual Policy Evaluation

Require: Policy π to evaluate, logged evaluation data D_{val}.

Initial empty samples set Φ.
for $(s, a, p, r, c) \in D_{val}$ **do**
    $p_{new} = \pi(s, a)$
    Add transformed samples $((s, a, p, c, p_{new}))$ into samples set Φ.
end for
Evaluated cost C= Doubly Robust(Φ) ▷ evaluate cost use doubly robust.
return Evaluated cost C.

B UPPER BOUND FOR λ

When using bisection search, we can use assumption 3 to calculate upper bound for dual variable λ.

Assumption 3. *Monotonicity between reward and cost at every state, when $c(s, a_0) > c(s, a_1)$, then $r(s, a_0) > r(s, a_1)$*

This assumption presents the simple idea that the larger reward you get, the higher cost you pay. This idea holds in most promotion and advertising scenarios.

The details of deriving upper bound are given as follows.

The basic idea to find an upper bound for λ is the optimal policy for λ is π_l in definition 1. Assume the reward and cost is in ascending order for action $a_0, a_1, ...$, define reward and action for action a_i is r_i, c_i, so $r_i < r_j, \forall i < j$ and $c_i < c_j, \forall i < j$, define upper bound of λ as λ_u, for π_l, we need $\arg\max_i r_i - \lambda_u * c_i$ is 0. which means

$$r_0 - \lambda_u * c_0 < r_i - \lambda_u * c_i, \quad \forall i = 1, ..., n \tag{7}$$

so

$$\frac{r_0 - r_i}{c_0 - c_i} < \lambda_u, \quad \forall i = 1, ..., n \tag{8}$$

The details of find upper bound of λ step is given in Algorithm 5

Algorithm 5 Upper Bound for Lambda

```
procedure LambdaUpperBound(D_val)
    ▷ calculate mean reward and cost for each action
    action_reward_sum = Counter()
    action_cost_sum = Counter()
    action_count = Counter()
    for (s, a, p, r, c) ∈ D_val do
        action_reward_sum[a] += r
        action_cost_sum[a] += c
        action_count[a] += 1
    end for
    action_reward = Dict()
    action_cost = Dict()
    action_list = List()
    for a, n ∈ action_count do
        action_list.append(a)
        action_reward[a] = action_reward_sum[a]/n
        action_cost[a] = action_cost_sum[a]/n
    end for
    ▷ search upper bound for λ
    λ = 0
    r_0 = action_reward[0]
    c_0 = action_cost[0]
    for i ∈ range(1, len(action_list)) do
        a = action_list[i]
        r_i = action_reward[a]
        c_i = action_cost[a]
        value = (r_0 − r_i)/(c_0 − c_i) + 1.
        if value < λ then
            λ = value
        end if
    end for
end procedure
```