Real-time Bidding by Reinforcement Learning in Display Advertising

3/2/22

Shuhui Zhu,
Cheriton School of Computer Science

UNIVERSITY OF WATERLOO

[1] Cai, Han, et al. "Real-time bidding by reinforcement learning in display advertising." Proceedings of the Tenth ACM International Conference on Web Search and Data Mining. 2017.

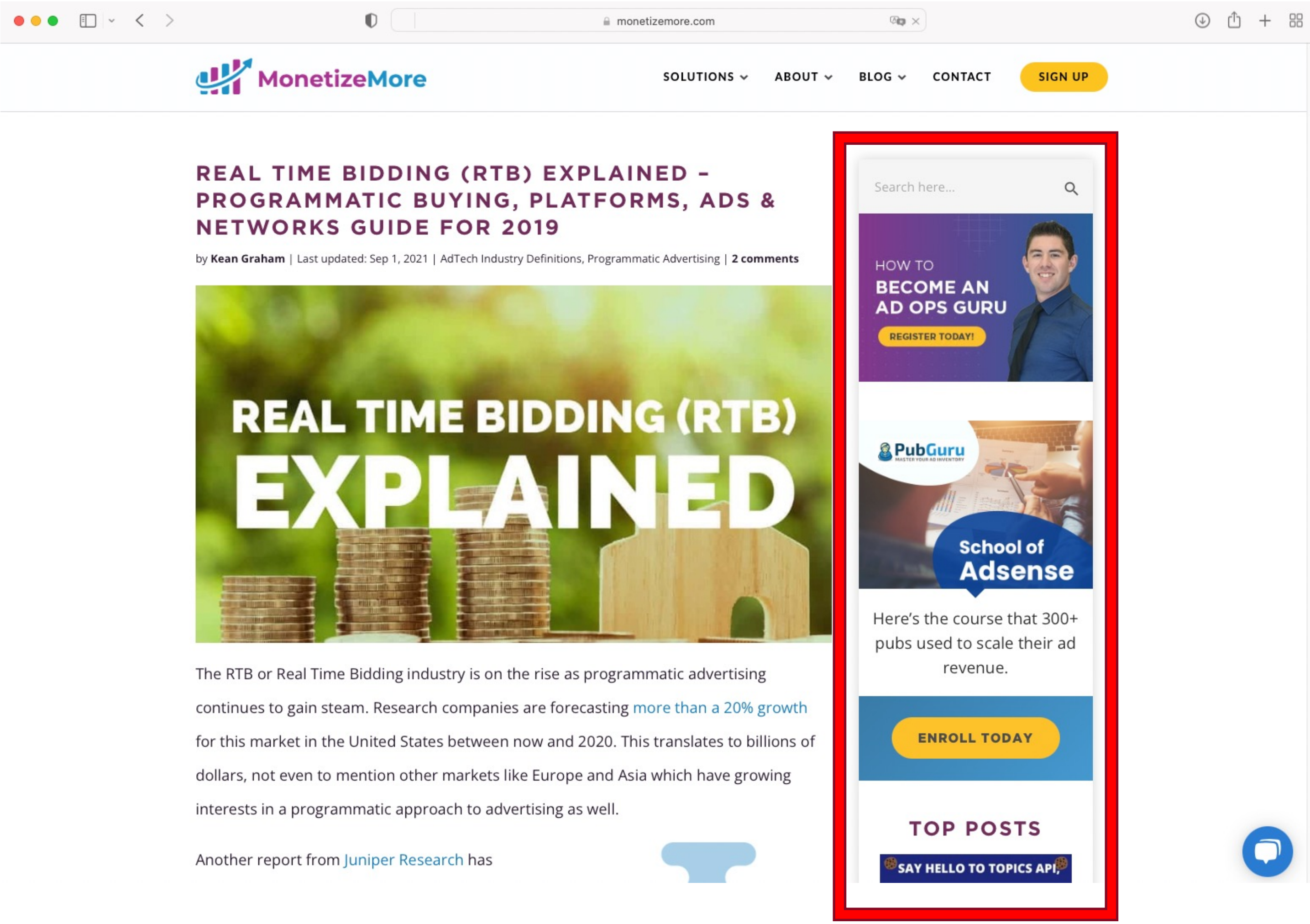
CONTENT

- Introduction
- Problem and Formulation
- Dynamic Programming Solution
- Handling Scale Issues
- Experimental Results
- Conclusions

INTRODUCTION

What is real-time bidding and how does it work?

Background: How does real-time bidding work?

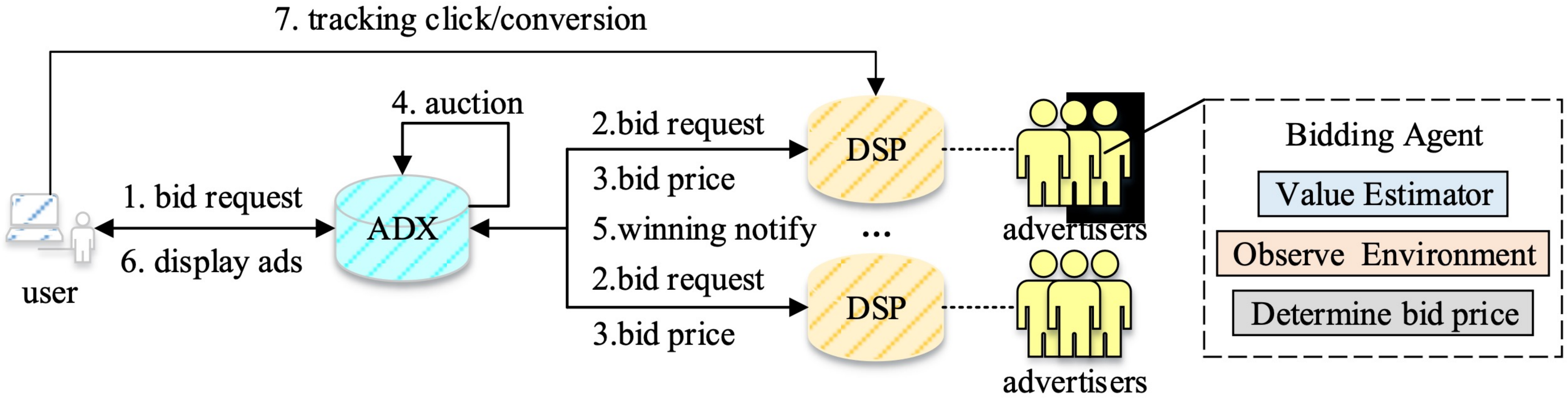


Decide within 200 ms!

Background: How does real-time bidding work?

- DSP: Demand-side platform.
  - Allows an advertiser to buy ad space and manage their ads. (e.g. Google Ads)
- ADX: Ad exchange.
  - Digital marketplace that enables advertisers and publishers to buy and sell advertising space, often through real-time auctions.
- User: Costumer who visits the website
- Advertiser: Bidding agent who bid for the ad space

Background: How does real-time bidding work?



[2] Liu, Mengjuan, et al. "Bid Optimization using Maximum Entropy Reinforcement Learning." arXiv preprint arXiv:2110.05032 (2021).

PROBLEM AND FORMULATION

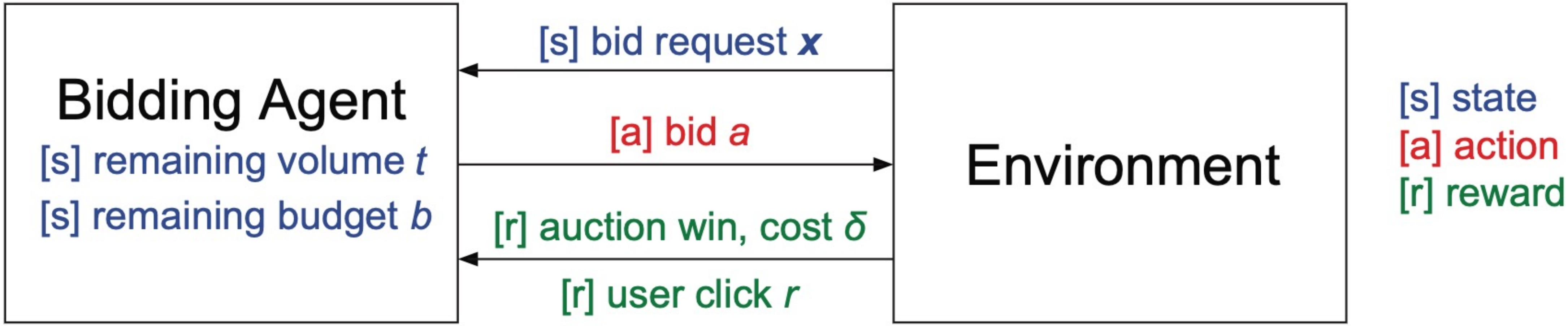
What is the problem and how to build up our solution?

Problem description

- **Goal**: From the agent side, the goal is to decide an optimal bidding price that can maximize the total clicks or revenue corresponding to the ad.
- **Problem**: RTB market is highly dynamic and static bid optimization may not work well.
- **Solution**: In this paper, consider bidding as a sequential decision and formulate it as a reinforcement learning to bid (RLB) problem.

Markov Decision Process

- **Environment**: the whole market and Internet users
- **Bidding Agent**: advertiser



Reinforcement Learning Setup

- This paper proposes a model-based solution for real-time bidding strategy.
  - Solve the value function
  - Extract the optimal policy
- **Initialization**: Agent is initialized with a total budget B and total number of auctions T for each episode.
- **State**: state s is composed of $s = (t, b, x_t)$
  - Remaining auction number $t \in \{0, \ldots, T\}$
  - Unspent budget $b \in \{0, \ldots, B\}$
  - Feature vector x_t
- **Action**: bidding price
- **Reward**: predicted click-through rate (pCTR)

Formulation of RTB

- At each timestep, for current state s $= (t, b, x_t)$, the agent receives an auction x, and determines its bid price a.
  - When bidding price a ≥ market price δ, the agent wins the auction and pays δ, remaining budget changes to b-δ, the expected reward is measured by pCTR network $\theta(x)$.
  - When bidding price a $<$ market price δ, the agent loses the auction, remaining budget remains b, the expected reward is zero.
- Repeat until the end of episode, i.e. t = 0

Notations

Notation	Description
$\boldsymbol{x}$	The feature vector that represents a bid request.
$\boldsymbol{X}$	The whole feature vector space.
$p_x(\boldsymbol{x})$	The probability density function of $\boldsymbol{x}$.
$\theta(\boldsymbol{x})$	The predicted CTR (pCTR) if winning the auction of $\boldsymbol{x}$.
$m(\delta, \boldsymbol{x})$	The p.d.f. of market price δ given $\boldsymbol{x}$.
$m(\delta)$	The p.d.f. of market price δ.
$V(t, b, \boldsymbol{x})$	The expected total reward with starting state $(t, b, \boldsymbol{x})$, taking the optimal policy.
$V(t, b)$	The expected total reward with starting state (t, b), taking the optimal policy.
$a(t, b, \boldsymbol{x})$	The optimal action in state $(t, b, \boldsymbol{x})$.

Functionals

- Transition probabilities and reward function:

$$\boxed{\begin{aligned}\mu\Big(a,(t,b,\boldsymbol{x}_t),(t-1,b-\delta,\boldsymbol{x}_{t-1})\Big) &= p_x(\boldsymbol{x}_{t-1})m(\delta,\boldsymbol{x}_t),\\ \mu\Big(a,(t,b,\boldsymbol{x}_t),(t-1,b,\boldsymbol{x}_{t-1})\Big) &= p_x(\boldsymbol{x}_{t-1})\sum_{\delta=a+1}^{\infty} m(\delta,\boldsymbol{x}_t),\end{aligned}}$$

$$\boxed{\begin{aligned}r\Big(a,(t,b,\boldsymbol{x}_t),(t-1,b-\delta,\boldsymbol{x}_{t-1})\Big) &= \theta(\boldsymbol{x}_t),\\ r\Big(a,(t,b,\boldsymbol{x}_t),(t-1,b,\boldsymbol{x}_{t-1})\Big) &= 0,\end{aligned}} \tag{1}$$

Value function and optimal policy

- The optimization is based on the value function, i.e the expected sum of rewards upon starting in state s and obeying policy π. It satisfies the Bellman equation with discount factor $\gamma = 1$.

$$V^{\pi}(s) = \sum_{s' \in \boldsymbol{S}} \mu(\pi(s), s, s')\Big(r(\pi(s), s, s') + V^{\pi}(s')\Big)$$

- The optimal policy is computed as:

$$\pi^{*}(s) = \operatorname*{argmax}_{a \in \boldsymbol{A}_s} \left\{ \sum_{s' \in \boldsymbol{S}} \mu(a, s, s')\Big(r(a, s, s') + V^{*}(s')\Big) \right\}$$

DYNAMIC PROGRAMMING SOLUTION

How to solve this model?

How to solve the optimal policy?

- Plugging the transition function and reward function into Bellman equation:

$$
\begin{aligned}
V(t,b,\boldsymbol{x}) = & \boxed{\max_{0\le a\le b}\Bigg\{\sum_{\delta=0}^{a}\int_{\boldsymbol{X}} m(\delta,\boldsymbol{x})p_x(\boldsymbol{x}_{t-1})\cdot \Big(\theta(\boldsymbol{x})+V(t-1,b-\delta,\boldsymbol{x}_{t-1})\Big)\,d\boldsymbol{x}_{t-1}} + \\
& \boxed{\sum_{\delta=a+1}^{\infty}\int_{\boldsymbol{X}} m(\delta,\boldsymbol{x})p_x(\boldsymbol{x}_{t-1})V(t-1,b,\boldsymbol{x}_{t-1})\,d\boldsymbol{x}_{t-1}\Bigg\}} \\
= & \max_{0\le a\le b}\Bigg\{\sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x})+V(t-1,b-\delta)\Big) + \\
& \sum_{\delta=a+1}^{\infty} m(\delta,\boldsymbol{x})V(t-1,b)\Bigg\}, \qquad (4)
\end{aligned}
$$

Win auction, a ≥ δ

Lose auction, a < δ

$$
\begin{aligned}
a(t,b,\boldsymbol{x}) = \operatorname*{argmax}_{0\le a\le b}\Bigg\{ & \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x})+V(t-1,b-\delta)\Big) + \\
& \sum_{\delta=a+1}^{\infty} m(\delta,\boldsymbol{x})V(t-1,b)\Bigg\}, \qquad (5)
\end{aligned}
$$

UNIVERSITY OF WATERLOO

How to solve the optimal policy?

- Consider situations where we do not observe feature vector x, and the value function V(t,b) is derived by marginalizing out x:

$$
\begin{aligned}
V(t,b) = \int_{\boldsymbol{X}} p_x(\boldsymbol{x}) \max_{0\le a\le b} \Big\{ \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x}) + V(t-1,b-\delta)\Big) \\
+ \sum_{\delta=a+1}^{\infty} m(\delta,\boldsymbol{x}) V(t-1,b)\Big\} \, d\boldsymbol{x} \\
= \max_{0\le a\le b} \Big\{ \sum_{\delta=0}^{a} \int_{\boldsymbol{X}} p_x(\boldsymbol{x}) m(\delta,\boldsymbol{x})\theta(\boldsymbol{x})\, d\boldsymbol{x} + \sum_{\delta=0}^{a} V(t-1,b-\delta) \cdot \\
\int_{\boldsymbol{X}} p_x(\boldsymbol{x}) m(\delta,\boldsymbol{x})\, d\boldsymbol{x} + V(t-1,b) \sum_{\delta=a+1}^{\infty} \int_{\boldsymbol{X}} p_x(\boldsymbol{x}) m(\delta,\boldsymbol{x})\, d\boldsymbol{x} \Big\} \\
= \max_{0\le a\le b} \Big\{ \sum_{\delta=0}^{a} \int_{\boldsymbol{X}} p_x(\boldsymbol{x}) m(\delta,\boldsymbol{x})\theta(\boldsymbol{x})\, d\boldsymbol{x} + \qquad (6) \\
\sum_{\delta=0}^{a} m(\delta) V(t-1,b-\delta) + V(t-1,b) \sum_{\delta=a+1}^{\infty} m(\delta) \Big\}.
\end{aligned}
$$

How to solve the optimal policy?

- To simplify the value function, consider an approximation $m(\delta, x) \approx m(\delta)$, then we have:

$$\int_{\boldsymbol{X}} p_x(\boldsymbol{x}) m(\delta, \boldsymbol{x}) \theta(\boldsymbol{x})\, d\boldsymbol{x} \approx m(\delta) \int_{\boldsymbol{X}} p_x(\boldsymbol{x}) \theta(\boldsymbol{x})\, d\boldsymbol{x} = m(\delta)\theta_{\text{avg}} \ , \tag{7}$$

- Plug equation (7) to value function:

$$V(t,b) \approx \max_{0 \le a \le b} \Big\{ \sum_{\delta=0}^{a} m(\delta)\theta_{\text{avg}} + \sum_{\delta=0}^{a} m(\delta) V(t-1, b-\delta) + \sum_{\delta=a+1}^{\infty} m(\delta) V(t-1, b) \Big\}. \tag{8}$$

How to solve the optimal policy?

- Extract the optimal policy from the value function:

$$
\begin{aligned}
a(t,b,\boldsymbol{x}) &= \operatorname*{argmax}_{0\le a\le b}\Big\{\sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x})+V(t-1,b-\delta)\Big)- \\
&\qquad \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})V(t-1,b)\Big\} \\
&= \operatorname*{argmax}_{0\le a\le b}\Big\{\sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x})+\boxed{V(t-1,b-\delta)}-V(t-1,b)\Big)\Big\} \\
&\equiv \operatorname*{argmax}_{0\le a\le b}\Big\{\sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\boxed{g(\delta)}\Big\},
\end{aligned}
\qquad (9)
$$

monotonically increases w.r.t. b, thus monotonically decreases w.r.t. δ

monotonically decreases w.r.t. δ

$$
a(t,b,\boldsymbol{x}) = \begin{cases} b & \text{if } g(b)\ge 0 \\ A \quad g(A)\ge 0 \text{ and } g(A+1)<0 & \text{if } g(b)<0 \end{cases}. \qquad (10)
$$

Algorithm and pseudocode

Algorithm 1 Reinforcement Learning to Bid

Input: p.d.f. of market price $m(\delta)$, average CTR θ_{avg}, episode length T, budget B

Output: value function $V(t, b)$

initialize $V(0, b) = 0$

for $t = 1, 2, \cdots, T-1$ **do**

  for $b = 0, 1, \cdots, B$ **do**

    enumerate $a_{t,b}$ from 0 to $\min(\delta_{\max}, b)$ and set $V(t, b)$ via Eq. (8)

  end for

end for

Input: CTR estimator $\theta(\boldsymbol{x})$, value function $V(t, b)$, current state $(t_c, b_c, \boldsymbol{x}_c)$

Output: optimal bid price a_c in current state

calculate the pCTR for the current bid request: $\theta_c = \theta(\boldsymbol{x}_c)$

for $\delta = 0, 1, \cdots, \min(\delta_{\max}, b_c)$ **do**

  if $\theta_c + V(t_c - 1, b_c - \delta) - V(t_c - 1, b_c) \geq 0$ **then**

    $a_c \leftarrow \delta$

  end if

end for

HANDLING LARGE-SCALE ISSUES

How to handle the real-world large-scale problem?

Large-Scale Problem

- In the algorithm, we update the value function V(t,b) by two nested loops
  - Time complexity O(TB)
  - Space complexity O(TB)
- Problem: For large-scale T and B, the algorithm can not be applied
- Solution:
  - Use parameterized models (Neural Networks) to fit the value function on small data scale, i.e., $\{0, \ldots, T_0\} \times \{0, \ldots, B_0\}$ where $T_0 < \mathrm{T}$ and $\mathrm{B}_0 < \mathrm{B}$.
  - Map the unseen states to acquainted states.
    - Implicit – segmentation model (RLB-NN-Seg)
    - Explicit – state mapping model (RLB-NN-MapD or RLB-NN-MapA)

Large-Scale Problem

- In equation (9), we expect the prediction error of θ(x) + V (t – 1,b–δ)–V(t–1,b) in the training data to be low in comparison to the average pCTR $\theta_{\{avg\}}$.

$$
\begin{aligned}
a(t,b,\boldsymbol{x}) &= \underset{0\le a\le b}{\operatorname{argmax}} \Big\{ \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x}) + V(t-1,b-\delta)\Big) - \\
&\qquad \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})V(t-1,b)\Big\} \\
&= \underset{0\le a\le b}{\operatorname{argmax}} \Big\{ \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x})\Big(\theta(\boldsymbol{x}) + V(t-1,b-\delta) - V(t-1,b)\Big)\Big\} \\
&\equiv \underset{0\le a\le b}{\operatorname{argmax}} \Big\{ \sum_{\delta=0}^{a} m(\delta,\boldsymbol{x}) g(\delta)\Big\}, \qquad (9)
\end{aligned}
$$

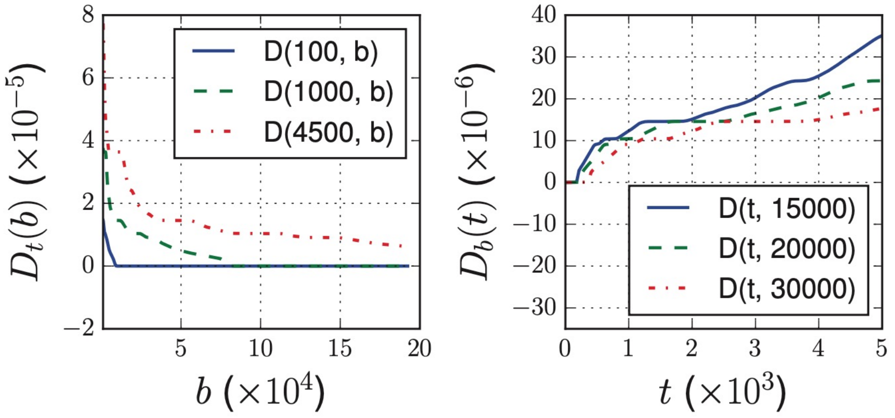
Solution to Large-Scale Problem -- State Mapping Models

- Introduce a new function D(t,b) and use it to replace the role of V(t,b):

$$D(t,b) = V(t,b+1) - V(t,b)$$

$$V(t-1,b-\delta) - V(t-1,b) = -\sum_{\delta'=1}^{\delta} D(t-1,b-\delta').$$

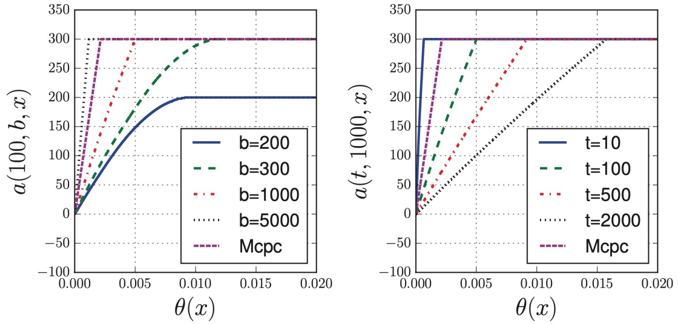
- Take fully connected neural network as a non-linear approximator for D(t,b).



UNIVERSITY OF WATERLOO

Solution to Large-Scale Problem -- State Mapping Models

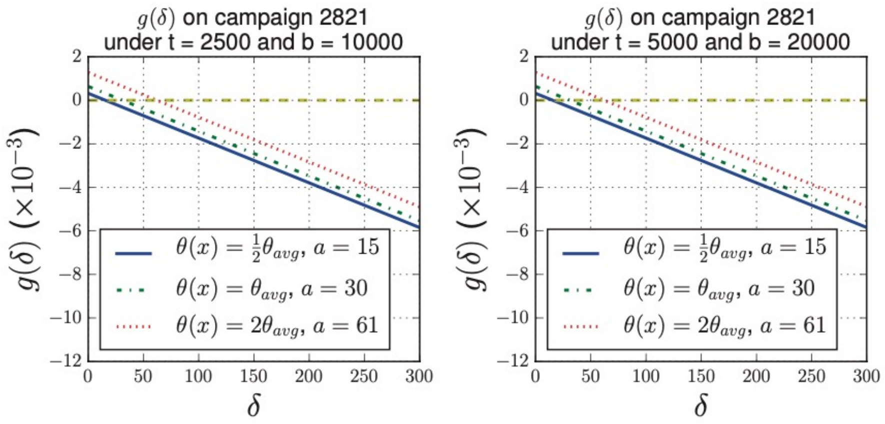
- Segmentation Model:
  - Divide the large episode into several small episodes with length T_0. Within each large episode we allocate the remaining budget to the remaining small episodes.
- State Mapping Models:
  - RLB-NN-MapD: For unseen states $t > T_0$ and $b > B_0$, there should be some points $\{(t', b')\}$ where $t' \leq T_0$ and $b' \leq B_0$ such that $D(t', b') = D(t, b)$. This model is to combine NNs with the mapping of D(t,b).
  - RLB-NN-MapA: Similarly, a(t, b, x) decreases w.r.t. t and increases w.r.t. b, which is consistent with intuitions. This model is to combine NNs with the mapping of a(t, b, x) .



UNIVERSITY OF WATERLOO

Solution to Large-Scale Problem -- State Mapping Models

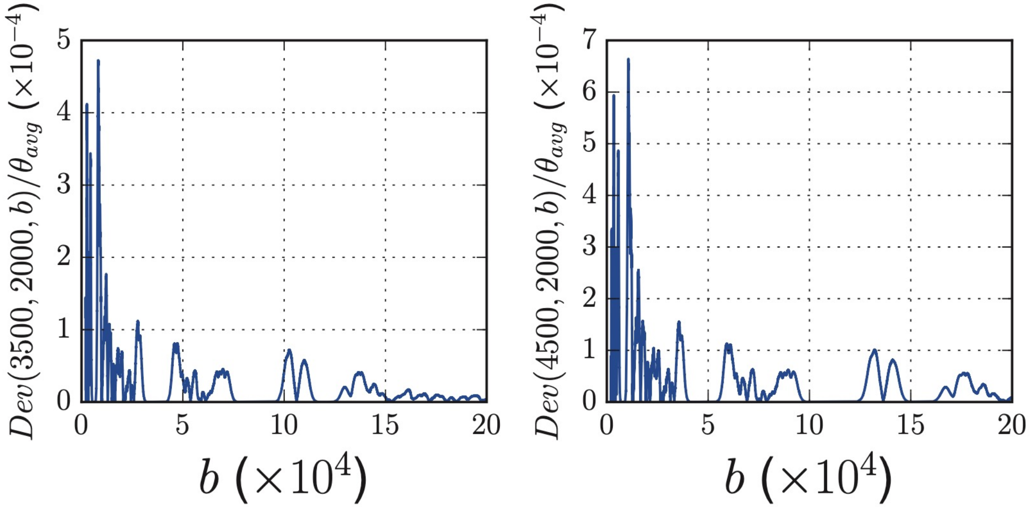
- Therefore, consider a simple case that b/t represents the budget condition, we can map the unseen states to acquainted states via two linear mapping forms:
  - (i) map a(t, b, x) where $t > T_0$ to $a(T_0, \frac{b}{t} \times T_0, x)$.
  - (ii) map D(t, b) where $t > T_0$ to D$(T_0, \frac{b}{t} \times T_0)$.
  - Intuition: From the view of practical bidding, when the remaining number of auctions are large and the budget situation is similar, given the same bid request, the agent should give a similar bid price



UNIVERSITY OF WATERLOO

Solution to Large-Scale Problem -- State Mapping Models

- Derivations of the simple linear mapping method:
  - Denote $\mathrm{Dev}(\mathrm{t}, \mathrm{T_0}, \mathrm{b}) = \left| D(t, b) - D(T_0, \frac{b}{t} \times T_0) \right|$, the deviations are low enough



EXPERIMENTAL RESULTS

How is the model performance over datasets?

Experiment Setup

- Evaluation metrics:
  - Number of acquired clicks
  - Win rate
  - Cost per mille impressions (CPM)
  - Effective cost per click (eCPC)
- Budget constraints:
  - $B = CPM_{train} \times 10^{-3} \times T \times c_0$, where c_0 acts as the budget constraints parameter
  - Run the evaluation with c_0 = 1/32, 1/16, 1/8, 1/4, 1/2
- Episode length
  - For the large-scale evaluation, we set the episode length T as 100,000
  - For the small-scale evaluation, we set the episode length T as 1,000
  - Also run a set of evaluations with episode length T = 200,400,600,800,1000

UNIVERSITY OF WATERLOO

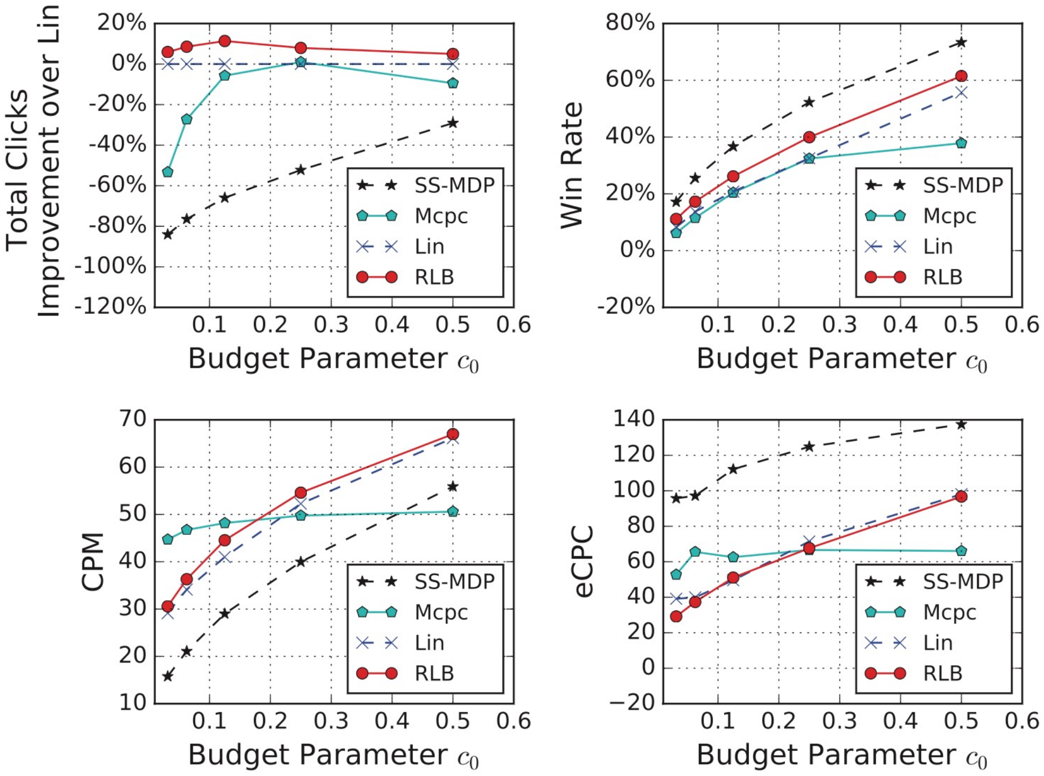
Illustration of other methods

- SS-MDP: Considering the bid landscape but ignoring the feature vector of bid request when giving the bid price. Although we regard this model as the state-of-the-art, it is proposed to work on keyword-level bidding in sponsored search, which makes it not fine-grained enough to compare with RTB display advertising strategies.

- Mcpc: gives its bidding strategy as $a_{\{Mcpc\}(t,b,x)} = CPC \times \theta(x)$, which matches some advertisers' requirement of maximum CPC (cost per click).
  - Does not adjust its strategy when the budget condition changes

- Lin: Linear bidding strategy w.r.t. the pCTR: $a_{\{Lin\}\,(t,b,x)} = b_0 \frac{\theta(x)}{\theta_{\{avg\}}}$ where b_0 is the basic bid price and is tuned using the training data [18]. This is the most widely used model in industry.

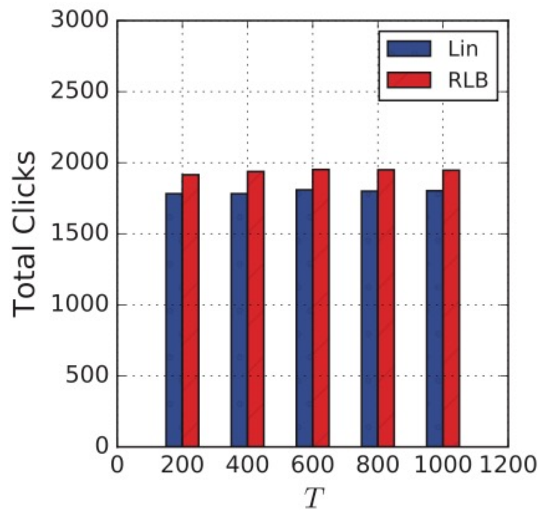
Small-Scale Evaluations

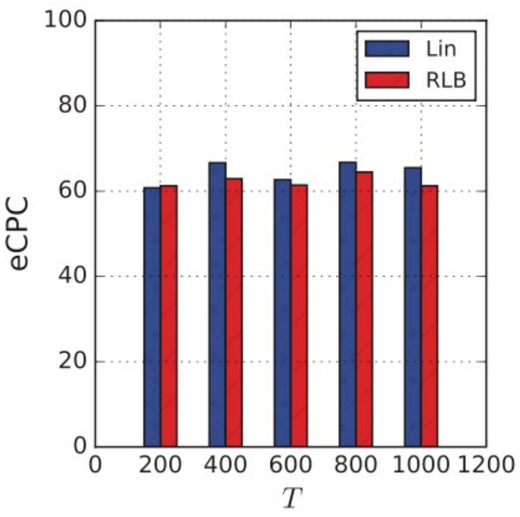


- In the comparison on total clicks, RLB performs the best under every budget condition.
  - Verifying the effectiveness of the derived algorithm for optimizing attained clicks

- In the comparison on win rate, RLB can generate a higher number of clicks with comparable CPM and eCPC against Lin.
  - RLB effectively spends the budget according to the market situation, which is unaware of by Lin

Small-Scale Evaluations





- Compared to Lin, RLB can attain more clicks with similar eCPC.

UNIVERSITY OF WATERLOO

Small-Scale Evaluations

Table 2: Click improvement of RLB over Lin for each campaign under $T = 10^3$ and different budget conditions.

iPinYou	1/32	1/16	1/8	1/4	1/2
1458	4.66%	3.96%	3.25%	0.21%	1.02%
2259	114.29%	35.29%	9.09%	32.56%	22.22%
2261	25.00%	6.25%	-3.70%	6.82%	0.00%
2821	20.00%	11.86%	27.27%	29.36%	12.97%
2997	23.81%	54.55%	85.26%	13.04%	3.18%
3358	2.42%	3.30%	0.87%	3.02%	0.40%
3386	8.47%	22.47%	13.24%	14.57%	13.40%
3427	7.58%	10.04%	12.28%	6.88%	5.34%
3476	-4.68%	-3.79%	2.50%	5.43%	0.72%
Average	22.39%	15.99%	16.67%	12.43%	6.58%
YOYI	3.89%	2.26%	7.41%	3.48%	1.71%

Table 3: Detailed AUC and clicks ($T = 10^3$ and $c_0 = 1/16$).

iPinYou	AUC of $\theta(\boldsymbol{x})$	SS-MDP	Mcpc	Lin	RLB
1458	97.73%	42	405	455	473
2259	67.90%	13	11	17	23
2261	62.16%	16	12	16	17
2821	62.95%	49	38	59	66
2997	60.44%	116	82	77	119
3358	97.58%	15	144	212	219
3386	77.96%	24	56	89	109
3427	97.41%	20	178	279	307
3476	95.84%	38	103	211	203
Average	80.00%	37	114	157	170
YOYI	87.79%	120	196	265	271

UNIVERSITY OF WATERLOO

Large-Scale Evaluations

Table 4: Approximation performance of the neural network.

	iPinYou	YOYI
RMSE ($\times 10^{-6}$)	0.998	1.263
RMSE / θ_{avg} ($\times 10^{-4}$)	9.404	11.954

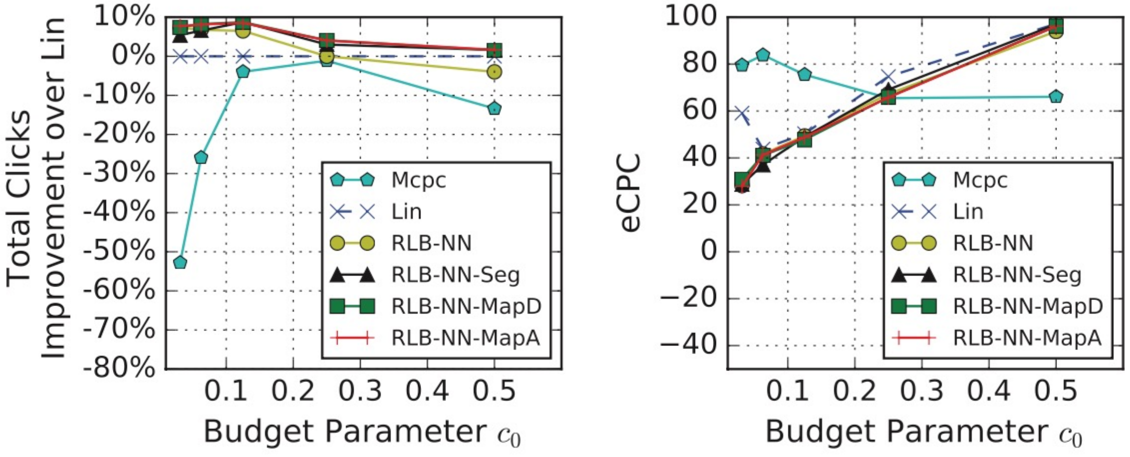


Figure 9: Overall performance on iPinYou under $T = 10^5$ and different budget conditions.

UNIVERSITY OF WATERLOO

Online Deployment and A/B Test

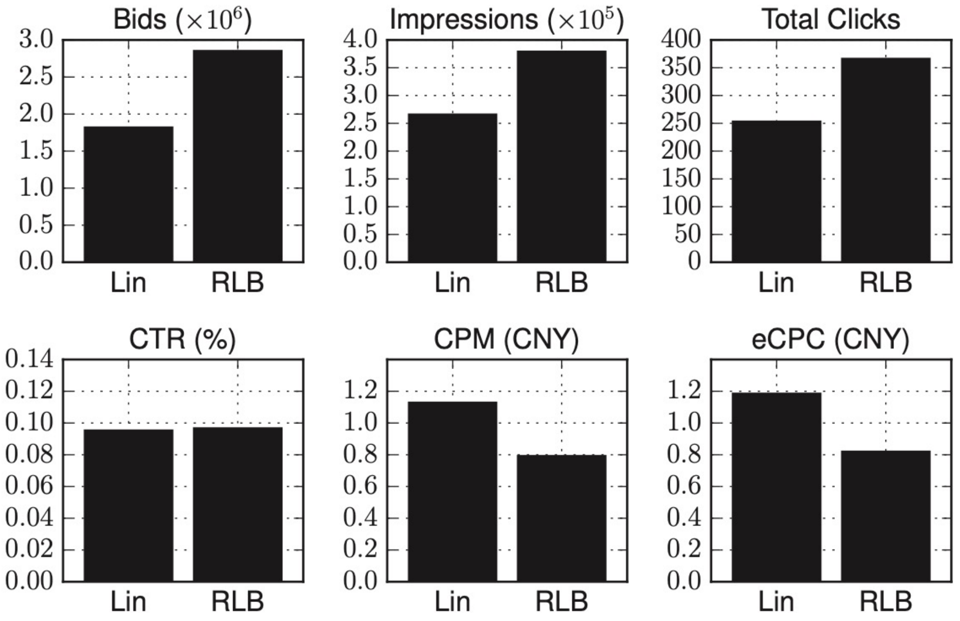


Figure 10: Online A/B testing results.

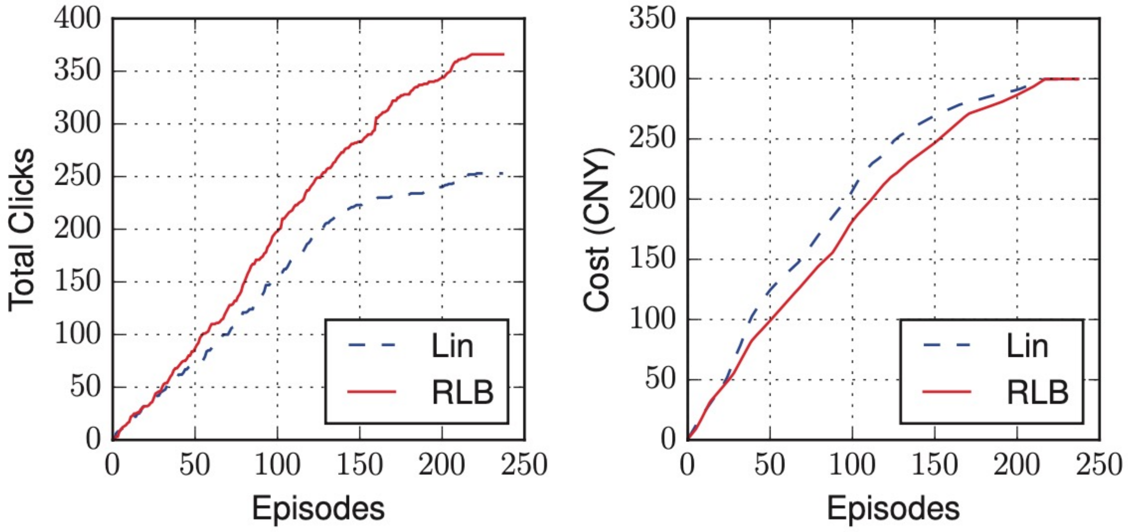


Figure 11: Total clicks and cost increase over episodes.

CONCLUSIONS

What are the conclusions?

Conclusions

- This paper proposed a model-based reinforcement learning model (RLB) for learning the bidding strategy in RTB display advertising so that the campaign budget can be dynamically allocated across all the available impressions on the basis of both the immediate and future rewards.

- The scalability problem from the large real-world auction volume and campaign budget is well handled by state value approximation using neural networks.

- Empirical study demonstrated the superior performance and high efficiency of RLB compared to state-of-the-art methods.

Future Work

- Future steps claimed by the authors:
  - Investigate **model-free approaches** such as Q-learning and policy gradient methods to unify utility estimation, bid landscape forecasting and bid optimization into a single optimization framework and handle the highly dynamic environment
  - Compare RLB solutions with the explicit budget pacing techniques

- Discussion on future extensions:
  - Continuous budget and bidding price
  - Non-stationary environments
  - Non-myoptic strategy

Some related work – recommended readings

- Continuous bidding price strategy using Soft Actor-Critic algorithm

  [2] Liu, Mengjuan, et al. "Bid Optimization using Maximum Entropy Reinforcement Learning." arXiv preprint arXiv:2110.05032 (2021).

- The long-term effect of impressions

  [3] Hausknecht, Matthew, and Peter Stone. "Deep recurrent q-learning for partially observable mdps." 2015 aaai fall symposium series. 2015.

- Non-stationary environments

  [4] Li, Zhuoshu, et al. "Faster Policy Adaptation in Environments with Exogeneity: A State Augmentation Approach." AAMAS. 2018.

- LIN – the most widely used bidding strategy in industry

  [5] Perlich, Claudia, et al. "Bid optimizing and inventory scoring in targeted online advertising." Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining. 2012.