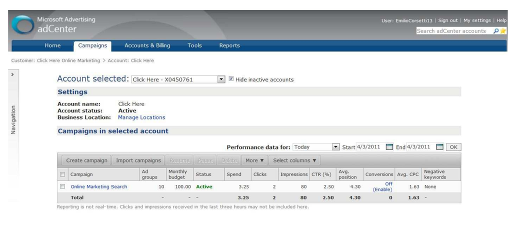
Budget Optimization for Sponsored Search: Censored Learning in MDPs

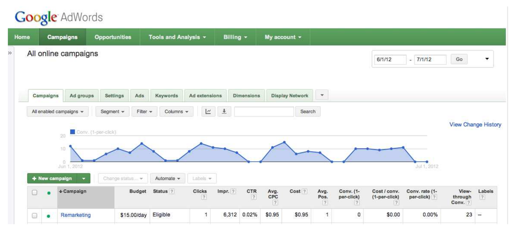
Kareem Amin and Michael Kearns
University of Pennsylvania

Peter Key and Anton Schwaighofer
Microsoft Research

Problem : Budgeted Bidding

- Advertisers for a website participate in online advertising networks: Microsoft adCenter, Google Adwords.
- *Goal*: Buy impressions (or clicks) from users searching for keywords relevant to the advertiser's website.
- Advertisers specify a budget that they wish to spend over the course of a certain time-period.





Problem : Budgeted Bidding

- Once a budget is specified, advertisers participate in a sequence of auctions.
- Each advertiser places a bid. The winner of the auction shows their ad to the user (an impression).
- This continues until the advertiser saturates its budget, or the time-period is over. When the time-period ends, the advertiser's budget is refreshed, and the next time-period begins.
- *Question:* What is a good bidding strategy?

Key Assumption

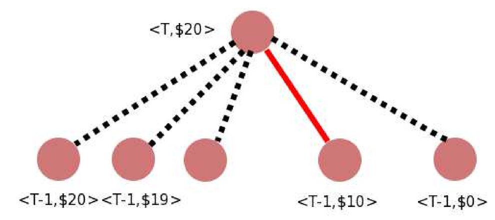
- Result of an auction *actually* depends on the strategic behavior of multiple advertisers participating in a modified second-price auction.
- *However*, from a single advertiser's perspective, there is some market price p_t^* determined by the behavior of the remaining advertisers.
- Bidding $b_t > p_t^*$ will win the impression at time t.
- *Key Assumption:* p_t^* are drawn iid from some unknown distribution D.
  - Complex strategic behavior modelled as stochastic market.
  - Common in finance.
  - Motivates a natural algorithm.
  - *Very good performance on real data (i.e. even if assumption is violated).*

Model

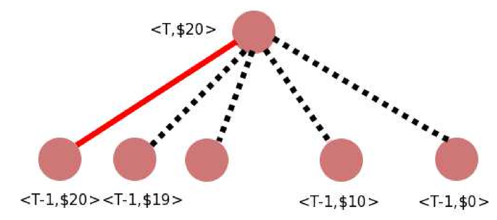
- Single keyword with unknown market price distribution D. Single Ad-Slot.
- Budget B^*.
- For time-period $u = 1, 2, \ldots$
  - Refresh Budget $B \leftarrow B^*$.
  - For auctions remaining $t = T, T-1, T-2, \ldots, 1$
    - Advertiser bids $b_{u,t}$.
    - Market draws price $p_{u,t}^* \sim D$.
    - If $b_{u,t} \geq p_{u,t}^*$ Advertiser wins impression for $p_{u,t}^*$.
      - Update Budget $B \leftarrow B - p_{u,t}^*$.
- Wrinkles easily accomodated by model/algorithm:
  - Multiple keywords; clicks instead of impressions; advertiser only charged if user clicks on ad; quality scores.
- More complicated: multiple ad-slots.

Sponsored Search MDP : SS-MDP

- Key Observation: Advertiser can be viewed as an agent in an MDP.
- States: $\langle T, B\rangle$ Time Remaining in Period, Budget Remaining in Period.
- Actions: Any bid less than B.
- Rewards: 1 if advertiser wins impression. 0 otherwise.
- Example: Bidding $\$15$ with $\$20$ and T rounds remaining.



– $p_T^* = \$10$. Reward of 1.



– $p_T^* > \$15$. Reward of 0.

- All states $\langle 1, B\rangle$ transition back to starting state $\langle T^*, B^*\rangle$ with probability 1 on any action, and next period begins.
- General Algorithms for solving MDPs (RMax, QLearning, etc...) won't exploit special structure.

Solving the SS-MDP

- Let $O(T, B)$ denote the optimal bid in state $\langle T, B\rangle$.
- Let $V(T, B)$ denote the value of the optimal strategy in state $\langle T, B\rangle$.
  - i.e. The expected number of impressions earned by the optimal strategy before returning to $\langle T^*, B^*\rangle$
- When $T = 1$:
  - $O(1, B) = B$. Go for broke, and bid remaining budget.
  - $V(1, B) = P(p \leq B)$, where $p \sim D$.
- In general:

$$V(T,B,c) = \sum_{b=1}^{c} P(p=b)[1 + V(T-1, B-b)] + P(p>c)V(T-1,B)$$

(Value of bidding c, and then playing optimally thereafter.)

$$V(T,B) = \max_c V(T,B,c) \qquad O(T,B) = \arg\max_c V(T,B,c)$$

Optimal strategy can be computed *if D is known.*

Switching Gears: Censored Learning

- If an advertiser bids b but does not win an impression, it knows that it did not bid high enough.
  - It does not observe the market price $p_{u,t}^*$ exactly, but knows $p_{u,t}^* > b$.
  - Censored Observations.
- On the other hand, when the advertiser wins an impression, the market price $p_{u,t}^*$ is charged to its budget.
  - It observes $p_{u,t}^*$ exactly.
  - Direct Observations.
- If advertiser only charged if a click occurs, two different types of censoring.

Censored Learning : Product Limit Estimator

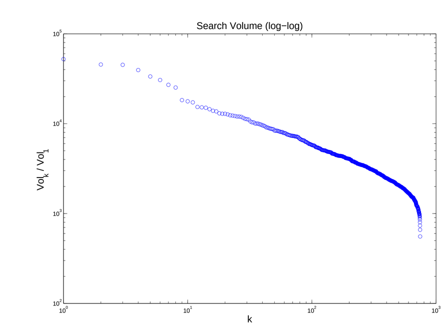
- Well-studied problem in long-term survival studies.
- Let $\{z_i\}$ be iid random variables from distribution D.
- Let $\{k_i\}$ be integers and observations $o_i = \min(z_i, k_i)$.
  - If $o_i = k_i$, o_i is censored.
- Given partially-censored observations, Kaplan and Meier give the non-parametric MLE for the CDF of D, called the Product-Limit Estimator.
- Let $D(s) = |\{o_i \mid s = o_i < k_i\}|$, the number of direct observations at s.
- Let $N(s) = |\{o_i \mid s \leq o_i, s < k_i\}|$.
- Let $S(t) = \prod_{s=1}^{t-1} 1 - \frac{D(s)}{N(s)}$.
- The MLE estimate for the CDF of D is given by $1 - S(t)$.

Putting it all Together

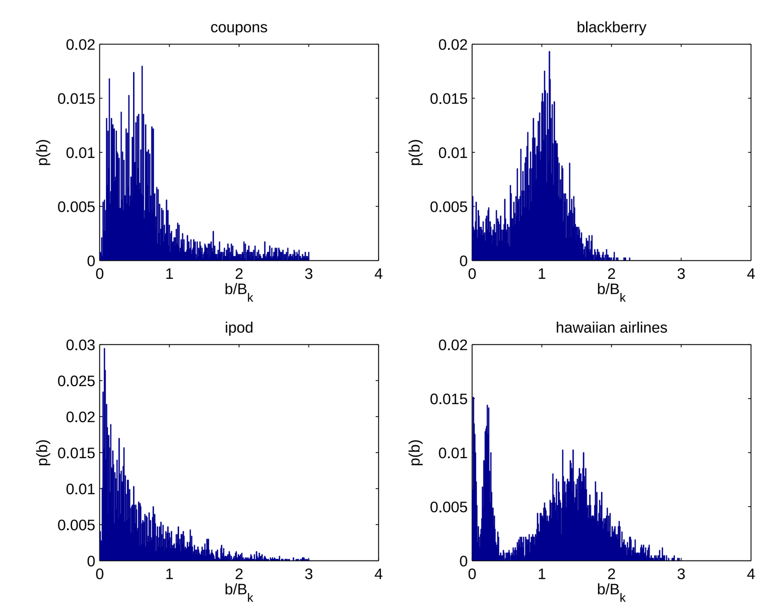
- *Algorithm Greedy Product-Limit*
- Let $O_D(T, B)$ be the optimal bid in state $\langle T, B\rangle$ of the SS-MDP, when the distribution is D.
- Algorithm will maintain an estimate $\hat{D}$ of true distribution D.
- Bid according to $O_{\hat{D}}(T, B)$.
- Receive possibly-censored observation drawn from D.
- Update $\hat{D}$ using Product-Limit Estimator
- *Natural approach, yet powerful.*

Experiments : Dataset

- Data collected from real auction history of advertisers placing bids through Microsoft's adCenter.
  - Vol_k volume of k-th most-searched keyword
  - Auction history of 100 keywords over three months.



- Two Sets of Experiments:
  - (1) Distributional. Contruct distribution from empirical distribution of bids. Simulate model.



  - (2) Sequential. Violate Modeling Assumption. Feed historical bids to algorithm sequentially.

Experiments : Results

- Summary of performance – Competitive Ratio With Optimal

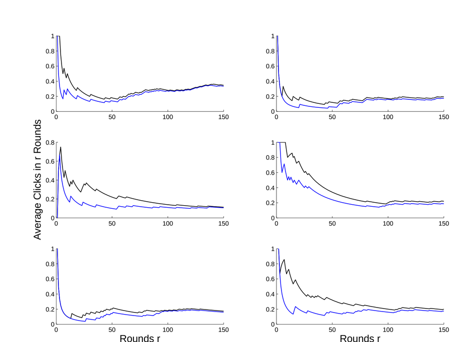
Distributional Results

Algorithm Name	Competitive Ratio	Std
Greedy Product-Limit	0.9573	0.1704
LuekerLearn	0.8448	0.1842
Fixed-Price Search	0.8352	0.1733
Q-learn	0.7484	0.1786
Budget Smoothing	0.1597	0.2418

Sequential Results

Algorithm Name	Competitive Ratio	Std
Greedy Product-Limit	0.9062	0.1166
LuekerLearn	0.8962	0.1152
Fixed-Price Search	0.6253	0.1395
Q-learn	0.5879	0.1558
Budget Smoothing	0.3105	0.3252

- *Lueker Learn* slight variant of *Greedy Product-Limit* replacing optimal control with stochastic knapsack algorithm.
- Convergence of *Greedy Product-Limit* happens on the time-scale of auctions, not periods.
- Natural time-scale of other algorithms is on the period time-scale
- Black: Offline Optimal (Sequential), Blue: Greedy Product-Limit



Experiments : More Results

- O_k the offline optimal number of clicks that can be attained after 10 periods (the entire data set).
- $A_{k,p}$ the clicks attained after a single period.
- Performance $A_{k,p}/O_k$ on sequential experiments.