Online Learning of Assignments

Matthew Streeter — Google, Inc
Daniel Golovin — Caltech
Andreas Krause — Caltech

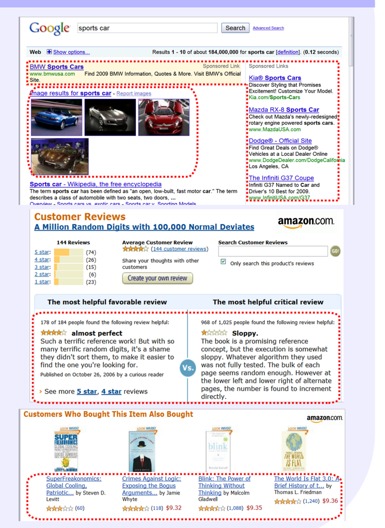
Optimizing Assignments Offline

Motivation

Assign:
- Ads to locations on a webpage
- Actuated sensors to sensing tasks

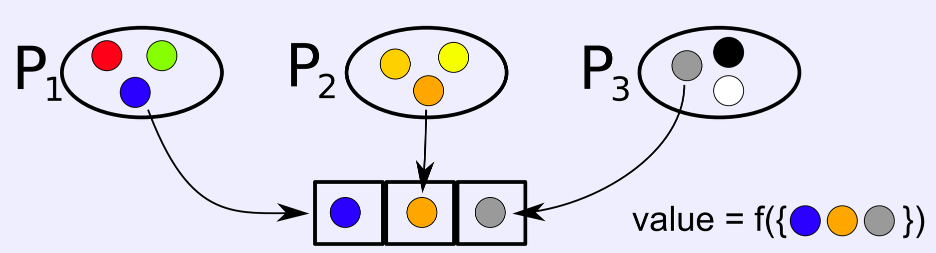
Rank (i.e., assign ranks to):
- Search results
- Information sources
- Recommendations

Optimize the whole assignment, not just sum of individual edges!
E.g., value diversity in top k results.



The Assignment Problem

K positions, and K sets of items, $P_1, P_2, \ldots, P_K$.
For each j, pick an element from P_j to put in position j.
Maximize $f(S)$, where $S \subseteq \cup_i P_i$ and $|S \cap P_i| \le 1$ for all i.



Problem is NP-hard, even for "simple" non-linear objective functions (e.g., max coverage).

Submodularity/Diminishing Returns

Function f is *submodular* if for all $S \subseteq T$ and $e \notin T$

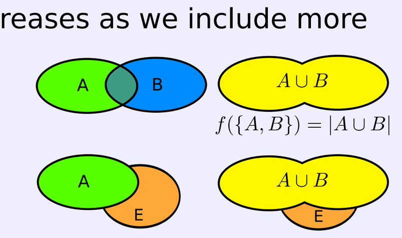
$$f(S \cup \{e\}) - f(S) \ge f(T \cup \{e\}) - f(T)$$

Submodularity = discrete diminishing returns
The marginal benefit of including *e* decreases as we include more

Example: the coverage objective (e.g., for sensor placement):

$$f(\{S_1, S_2, \ldots, S_k\}) = \left|\bigcup_i S_i\right|$$

Other examples:
- Joint Entropy: f(X) = H(X)
- Probabilistic Coverage: f(X) = Pr[e covered], given that each x in X covers e independently with some probability p(e).



E contributes less when we've already picked B.

The Locally Greedy Algorithm

For k = 1, 2, ..., K
  $s_k = \arg\max_{s \in P_k} \{f(\{s_1, \ldots, s_{k-1}\} + s)\}$
Output $\{s_1, \ldots, s_K\}$

Yields a 1/2 approximation: $f(\{s_1, \ldots, s_K\}) \ge \frac{1}{2}\mathrm{OPT}$
[Fisher *et al.*, Math Prog. Study '78]

Learning Assignments Online

Online Assignment Problem

K positions, and K sets of items, $P_1, P_2, \ldots, P_K$. An *assignment* $S \subseteq \cup_i P_i$ contains one element from each P_i.

In each round t, pick an assignment S_t
Observe payoff $f_t(S_t)$

Example: Sponsored Search Ad Allocation

Bandit Algorithms

Multiarmed Bandit Problem:
Feasible set of choices *F*.
For rounds *t* = 1, 2, 3, ...
  Pick *x(t)* in *F*
  Observe payoff $f_t(x(t))$, and nothing else.

Regret = how much better *best fixed choice* does than you.

$$R(T) = \max_{x \in F} \sum_{t=1}^{T} f_t(x) - \sum_{t=1}^{T} f_t(x(t))$$

Fact: There exist algorithms with $\mathbb{E}[R(T)] = \mathcal{O}(\sqrt{T|F|\log|F|})$
[Auer *et al.*, SIAM J. Comput. '02]

For the assignment problem, |*F*|, regret, and convergence time are all exponential in *K* ...

Online Locally Greedy

... but we can exploit submodularity to reduce the assignment problem to K smaller bandit problems.

Key idea: replace each greedy step with a bandit algorithm.
[Streeter & Golovin, NIPS '08]

In each round *t* = 1, 2, 3, ...
  For *k* = 1, 2, ..., *K*
    s_k = choice of a bandit algorithm $\mathcal{A}_k$ trying to pick $s \in P_k$ to maximize $f(\{s_1, \ldots, s_{k-1}\} + s)$
  Output $S_t = \{s_1, \ldots, s_K\}$
  Feed back $f(\{s_1, \ldots, s_k\})$ to $\mathcal{A}_k$ for each $1 \le k \le K$.

Theoretical Guarantees

α-Regret measures how much worse you are than an α-approx to the best fixed solution.

$$R_\alpha(T) \equiv \alpha \cdot \max_{S \in \mathcal{P}} \left(\sum_{t=1}^{T} f_t(S)\right) - \sum_{t=1}^{T} f_t(S_t)$$

Theorem: Online Locally Greedy with good bandit subroutines has low 1/2-regret. Specifically, $\mathbb{E}[R_{\frac{1}{2}}(T)] = \mathcal{O}(K\sqrt{T|F|\log|F|})$

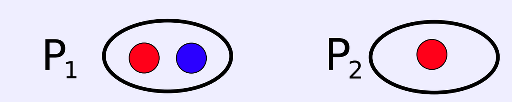
Can also get o(T) expected regret if you only observe $f_t(S_t)$.

So, Online Locally Greedy converges to a 1/2-approximation of the best fixed solution ...

The Algorithm

Worst-case for Locally Greedy

... but gets no better than a 1/2-approximation in the worst-case, because (offline) locally greedy can get stuck with OPT/2 in the worst case.



f(S) = # of distinct colors in *S*.

Locally greedy may pick ● from P_1, get stuck picking ● from P_2, and get f(S) = 1, while OPT = 2.

So, can we do better than 1/2?

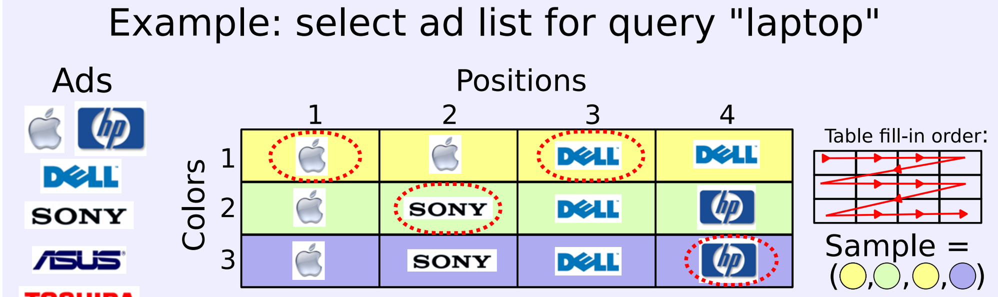
Tabular Greedy Algorithm

- Best offline approximation: (1 - 1/e) = 0.632... [Vondrak STOC '08]
- *Unconditional* matching hardness result [Mirrokni *et al.*, EC '08]
- Vondrak's algorithm seems unsuitable for our online problem.
- Our contribution: New, simpler (1 - 1/e)-approx algorithm.

Key idea: Avoid getting stuck with bad choices by building up solution gradually.

Tabular Greedy Algorithm
- C colors. K players, one per position.
- Players greedily commit 1/C probability to an ad (to maximize expected payoff) in round robin fashion over C rounds. Then all players must sample from their distributions.
- Payoff to player i is marginal benefit of its ad a_i over all ads whose play was committed to *before* a_i.

Example: select ad list for query "laptop"



- Player 2 committed 1/3 probability to picking [Apple] in the yellow round, and 1/3 prob. to picking SONY in the green and blue rounds.
- Player 2 sampled green, and thus plays SONY
- The outcome is ([Apple] , SONY , DELL , [hp])
- The payoff to player 2 is
  f([Apple] , SONY , DELL , null) - f([Apple] , null, DELL , null)

Online Tabular Greedy

Key idea: replace each greedy step with a bandit algorithm.

- Table of bandit algorithms, one per (position, color) pair.
- Each algorithm tries to maximize its payoff in the game defined by the algorithm.
- Works because selfish play leads to good approximation of global objective (i.e., game has low "price of total anarchy")

Theoretical Results

Theoretical Guarantees

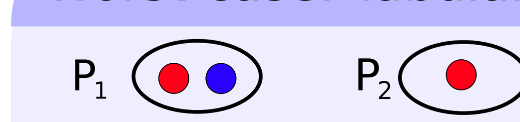
α-Regret measures how much worse you are than an α-approx to the best fixed solution.

$$R_\alpha(T) \equiv \alpha \cdot \max_{S \in \mathcal{P}} \left(\sum_{t=1}^{T} f_t(S)\right) - \sum_{t=1}^{T} f_t(S_t)$$

Theorem: Online Tabular Greedy with good bandit subroutines and a suitable number of colors has low (1-1/e)-regret. In the bandit setting, where you only observe $f_t(S_t)$
$\mathbb{E}[R_{(1-\frac{1}{e})}(T)] = \mathcal{O}\left(T^{5/6}\,\mathrm{poly}(K, |F|)\right)$

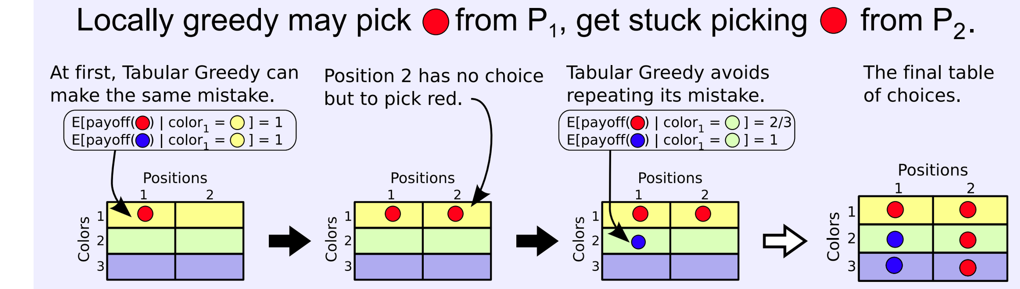
So, Online Tabular Greedy converges to a (1 - 1/e)-approximation of the best fixed solution.

Worst-case: Tabular vs Locally Greedy



f(S) = # of distinct colors in *S*.

Locally greedy may pick ● from P_1, get stuck picking ● from P_2.



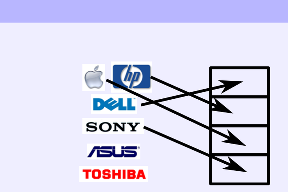
Optimal = 2. Locally greedy might get only one.
Tabular greedy gets 1*1/3+2*2/3 = 5/3 in expectation.

Subsumed Models for Ad Selection

1. Position dependent click-through-rates
   Put ad a_i in location i on round t, get reward $\sum_i \pi_{i,t}(a_i)$ for arbitrary $\pi_{i,t}$: Ads $\to [0, 1]$.
   [Edelman *et al.*, Amer. Econ. Review '07]
2. Models that value diversity
   Simple example: user t is interested in ads A_t, get reward 1 if you show at least one ad in A_t, zero reward otherwise.
   [Radlinski *et al.*, ICML '08], [Streeter & Golovin, NIPS '08]
3. Various Markovian models for users with varied interests and attention spans.

Experiments: Ad Selection

Given a user query, which list of ads should you show?



User model:
- Users have a random # of locations they'll look at, and a random set of ads they like. (drawn from an *arbitrary* joint distribution)
- Users click on one ad they like in the locations they look at, otherwise abandon results.
- Goal: Maximize # of clicks.

Experimental Results

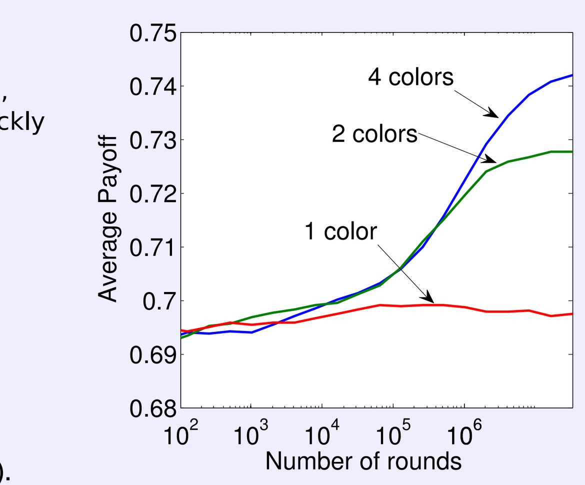
Results: Ad Selection

The Setup:
Two types of users:
  Type A looks everywhere,
  type B loses interest quickly
Two types of ads, A and B.

Pr[user likes ad]

ad type \ user type	A	B
A	0.5	0.2
B	0.2	0.5

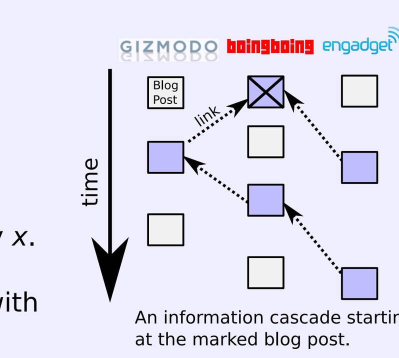
Bandit feedback: you only observe $f_t(S_t)$.



Experiments: Blog Ranking

Given limited time, which blogs should you follow?

The Setup:
- Blog = sequence of posts.
- hyperlink (*u*,*v*) means *v* influenced *u*.
- *Cascade* at *x* = all posts influenced by *x*. (influence is transitive)
- Cascade *detected* if you read a blog with a post in it.
- Possible objectives:
  1) Detect as many cascades as possible
  2) Minimize average time to detect cascades
  3) Maximize number of blogs that appear in the cascade *after* you detect it, i.e., "be one of the first to know."



An information cascade starting at the marked blog post.

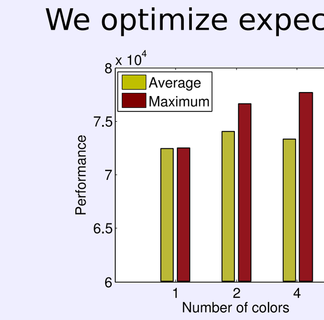
Results: Blog Ranking

We use objective #3, output lists of 5 blogs (of $\sim$ 45K), and suppose

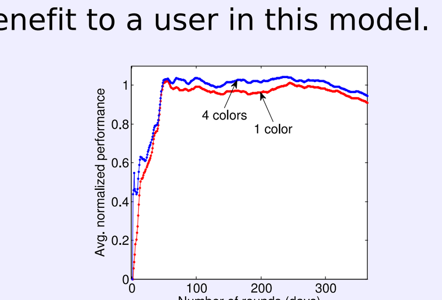
$$\mathbf{Pr}\,[\text{user reads first } k \text{ blogs}] \propto \gamma^k$$

for some $\gamma \in [0, 1]$. (Here $\gamma = 0.8$)

We optimize expected benefit to a user in this model.



Offline results: avg and max over 200 trials.



Online results: Algorithm converged within 50 rounds with full information feedback.

Conclusions

- New algorithm to learn to optimize lists and assignments.
- Theoretically optimal worst-case guarantees for monotone submodular objectives:
  - Includes a broad class of holistic quality measures.
  - Generalizes many previously studied metrics
- Empirical demonstration that Online Tabular Greedy is superior to previous approaches for some important applications.