Online Stochastic Matching with Unequal Probabilities

Aranyak Mehta  Google
Bo Waggoner  Harvard
Morteza Zadimoghaddam  Google

SODA 2015
Outline

● Problem and motivation
● Prior work, our main result
● Key idea: Adaptivity
● Ideas behind algorithm/analysis
Motivation: Search ads

advertisers

search queries

Time
Motivation: Search ads

Simplified problem:
- display one ad per query
- have estimate of click probabilities
- advertisers pay $1 if click, $0 if no click
- advertisers have budget for one click per day

How to assign ads?
Online Stochastic Matching

[Mehta and Panigrahi, 2012]
Online Stochastic Matching

[Mehta and Panigrahi, 2012]
Online Stochastic Matching

[Mehta and Panigrahi, 2012]

Pr[ searcher clicks if we show this ad ]

fixed, offline vertices

arrivals

$p_{11}$

$p_{31}$

$p_{41}$

Time
Online Stochastic Matching

[Mehta and Panigrahi, 2012]

fixed, offline vertices

online arrivals

Assign to vertex 3!

Alg

$p_{31}$
Online Stochastic Matching

[Mehta and Panigrahi, 2012]

With prob $p_{31}$: match succeeds

With prob $1 - p_{31}$: match fails

fixed, offline vertices

Alg

Time
Online Stochastic Matching

[Mehta and Panigrahi, 2012]

- fixed, offline vertices
- match succeeded
- online arrivals
- cannot be matched again
Online Stochastic Matching

[Mehta and Panigrahi, 2012]

- fixed, offline vertices
- match failed
- online arrivals
- may be matched again later
- disappears (cannot re-try)

Alg
Measuring algorithm performance

Alg’s performance = # successes

fixed, offline vertices

online arrivals

Alg
Measuring algorithm performance

Alg’s performance = $E[\# \text{ successes }]$
Measuring algorithm performance

Alg’s performance = $E[\# \text{ successes }]$

Opt’s performance = size of max weighted assignment, budget 1
Measuring algorithm performance

Competitive ratio =

\[
\min \frac{\text{Alg}}{\text{Opt}}
\]

over all input instances.

(Note: Opt is a bit funky … not achievable even with foreknowledge of instance.)
Prior Work

- **Online Matching with Stochastic Rewards**
  Mehta, Panigrahi, FOCS 2012.
  - Greedy = 0.5.
  - Opt

  - For case where all $p$ are equal and vanishing:
    Alg $\geq$ 0.567.

Open: anything better than Greedy for unequal $p$
This work

For unequal, vanishing edge probabilities:

\[ \text{Alg} \geq 0.534 \]
This work

For unequal, vanishing edge probabilities:

So what? 0.534

algorithmic ideas to beat Greedy

Alg

Opt
Outline

● Problem and motivation

● Prior work, our main result

● Key idea: Adaptivity
● Ideas behind algorithm/analysis
Adaptive: sees whether or not assignment succeeds
Our Approach

1. Start with an optimal non-adaptive alg that is straightforward to analyze

2. Add a small amount of adaptivity (second choices)

3. Analysis remains tractable by limiting amount of adaptivity
An optimal non-adaptive algorithm

- MP-2012: nonadaptive algs have upper bound of 0.5
- How to achieve 0.5? (Previously unknown.) Seems nonobvious.
Maximize *marginal expected gain*

offline vertices

online arrivals

Assign first arrival to vertex with largest \( p_{i1} \)
Maximize *marginal expected gain*

Assign next arrival to
\[
\max \Pr[ i \text{ available } ] p_{i2}
\]
Maximize *marginal expected gain*

Assign next arrival to maximize

\[
\Pr[\text{ i available } ] p_{i2}
\]

Offline vertices:

- 0.1
- 0.3
- 0.2

Calculations:

- \((1 - 0.4) \times 0.3 = 0.18\)
- \((1) \times 0.2 = 0.2\)
NonAdaptive

**Theorem:** NonAdaptive has a competitive ratio of 0.5 for the general online stochastic matching problem.

Does not require vanishing probabilities.
Why do we like NonAdaptive?

- On a given instance, an arrival has the same “first choice” every time (regardless of previous realizations)
- Algorithm tracks/uses competitive ratio (probabilities of success)
Add Adaptivity (but not too much)

Proposed SemiAdaptive:

Assign next arrival to max $\Pr[i \text{ available}] \ p_{ij}$ unless already taken, in which case assign to second-highest.
Why do we like SemiAdaptive?

- On a given instance, an arrival has the same first and second choices every time (regardless of previous realizations)

- Algorithm tracks/uses competitive ratio (probabilities of success)

These allow us to analyze SemiAdaptive -- almost…
(Analysis?) Roadblock

- **Want:** when first-choice is not available, get measurable benefit by assigning to second choice
  → giving improvement over NonAdaptive’s 0.5
(Analysis?) Roadblock

- **Want**: when first-choice is not available, get measurable benefit by assigning to second choice
  → giving improvement over NonAdaptive’s 0.5

- **Problem**: correlation between availability of first and second choice. Perhaps when first choice is not available, most likely second choice is not available either.
  → cannot guarantee improvement over NonAdaptive
(Analysis?) Roadblock

- **Want:** when first-choice is not available, get measurable benefit by assigning to second choice
  → giving improvement over NonAdaptive’s 0.5

- **Problem:** correlation between availability of first and second choice. Perhaps when first choice is not available, most likely second choice is not available either.
  → cannot guarantee improvement over NonAdaptive

- **Fix:** introduce independence / even less adaptivity.
  (no time to say more! sorry!)
RECAP

Online stochastic matching problem:
- edges succeed probabilistically
- maximize expected number of successes
- input instance chosen adversarially

New here:
- edge probabilities may be unequal
RECAP

Results:
- optimal 0.5-competitive NonAdaptive
- 0.534-competitive SemiAdaptive (with tweak) for vanishing probabilities

Key idea:
- control adaptivity to control analysis
Future Work

Everything about Online Stochastic Matching:

- **Vanishing probabilities:**
  - Equal: 0.567 ... ? ... 0.62
  - Unequal: 0.534 ... ? ... 0.62

- **Large probabilities:**
  - Equal: 0.53 ... ? ... 0.62
  - Unequal: 0.5 ... ? ... 0.62
Future Work

Everything about Online Stochastic Matching:
● **Vanishing probabilities:**
  ○ Equal: 0.567 … ? … 0.62
  ○ Unequal: 0.534 … ? … 0.62

● **Large probabilities:**
  ○ Equal: 0.53 … ? … 0.62
  ○ Unequal: 0.5 … ? … 0.62

Thanks!
Additional slides
Final Algorithm “SemiAdaptive”

Assign next arrival to max $\Pr[i \text{ available}] \ p_{ij}$ unless already taken,* in which case assign to second-highest.

* “it would have already been taken by a previous first-choice”

(key point: even less adaptive, more independence)
Ideas behind analysis

Pr[ available ]

Either first choice is the same as Opt’s...
Ideas behind analysis

Either first choice is the same as Opt’s...

...or both first and second choice would give at least as much “gain” as Opt’s.
Ideas behind analysis

Very good because gains “compound”.

Good because we get “second-choice gains”.

Either first choice is the same as Opt’s...

...or both first and second choice would give at least as much “gain” as Opt’s.
Note: Can only get $1 - \frac{1}{e} \approx 0.632$ even with knowledge of instance

Weighted matching: $1$

$$\mathbb{E}[\text{# of matches}] = 1 - \text{Pr[ all fail ]}$$
$$= 1 - (1 - \frac{1}{n})^n$$
$$\rightarrow 1 - \frac{1}{e}$$
Example of defining Opt

Alg's performance = \( E[\text{size of matching}] \)

Opt's performance = size of max weighted assignment, budget 1

Example:

- Fixed, offline vertices
- Online arrivals

Opt gets 1
- 1/2
- 2/3
- 1/4

Opt gets 1/2
- 1/4
- 1/4

Opt gets 1/2