Designing Approximately Optimal Search on Matching Platforms

Alexander Wei (UC Berkeley)

with Nicole Immorlica (Microsoft)
Brendan Lucier (Microsoft)
Vahideh Manshadi (Yale)

Online Matching Platforms

Online Matching Platforms

tinder Uber upwork™ formerly oDesk DOORDASH airbnb

Online Matching Platforms

tinder Uber upwork™ formerly oDesk DOORDASH airbnb

Challenges:

- Must organize matching at a massive scale
- Agents have unknown, heterogeneous preferences

Online Matching Platforms

tinder Uber upwork™ formerly oDesk DOORDASH airbnb

Challenges:

- Must organize matching at a **massive scale**
- Agents have **unknown**, **heterogeneous** preferences

Solution? Personalized **match recommendations**

- Good match recommendations help agents find better matches faster
- But also allow for individual preferences to be expressed

Online Matching Platforms

Challenges:

- Must organize matching at a **massive scale**
- Agents have **unknown**, **heterogeneous** preferences

Solution? Personalized **match recommendations**

- Good match recommendations help agents find better matches faster
- But also allow for individual preferences to be expressed

Our focus: How should the platform **design** match recommendations?

Match Recommendations as Search

For market participants, finding a match is often a **search process**:

- agents **“meet”** potential partners in sequence
- matching requires mutual want between agents

Match Recommendations as Search

For market participants, finding a match is often a **search process**:

- agents **“meet”** potential partners in sequence
- matching requires mutual want between agents

$$\text{match recommendations} \iff \text{platform-guided search}$$

Match Recommendations as Search

For market participants, finding a match is often a **search process**:

- agents **"meet"** potential partners in sequence
- matching requires mutual want between agents

match recommendations $\iff$ platform-guided search

Our focus′: How should the platform **design search**?

Match Recommendations as Search

For market participants, finding a match is often a **search process**:

- agents **“meet”** potential partners in sequence
- matching requires mutual want between agents

$$\text{match recommendations} \iff \text{platform-guided search}$$

Our focus′: How should the platform **design search**? (**Who meets whom**?)

Match Recommendations as Search

For market participants, finding a match is often a **search process**:

- agents **"meet"** potential partners in sequence
- matching requires mutual want between agents

$$\text{match recommendations} \iff \textbf{platform-guided search}$$

Our focus′: How should the platform **design search**? (**Who meets whom**?)

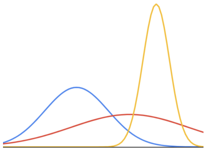
The platform's search design should be:

- informed by data, but also account for uncertainty
- robust to user adaptation in response to the design

Our Model: Data-driven Search Design

Model:

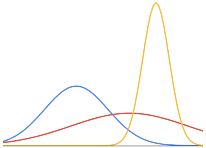
- Platform has **distributional knowledge** of preferences via **agent types**



Our Model: Data-driven Search Design

Model:

- Platform has **distributional knowledge** of preferences via **agent types**

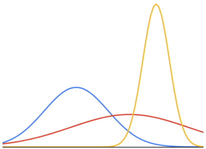


$\Longrightarrow$

Our Model: Data-driven Search Design

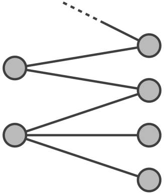
Model:

- Platform has **distributional knowledge** of preferences via **agent types**
- Search design = rates at which agents of different types “meet”



$\Longrightarrow$

$\Longrightarrow$



Our Model: Data-driven Search Design

Model:

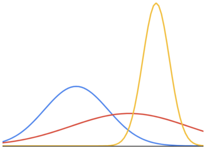
- Platform has distributional knowledge of preferences via agent types
- Search design = rates at which agents of different types "meet"
- Agents strategically accept/reject potential matches to maximize $\mathbb{E}$(utility)

Our Model: Data-driven Search Design

Model:

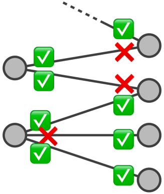
- Platform has **distributional knowledge** of preferences via **agent types**
- Search design = rates at which agents of different types “meet”
- Agents **strategically accept/reject** potential matches to maximize $\mathbb{E}$(utility)

Platform’s objective: Maximize **social welfare** generated by the market



$\Longrightarrow$

$\Longrightarrow$



Our Model: Data-driven Search Design

Model:

- Platform has **distributional knowledge** of preferences via **agent types**
- Search design = rates at which agents of different types "meet"
- Agents **strategically accept/reject** potential matches to maximize $\mathbb{E}$(utility)

Platform's objective: Maximize **social welfare** generated by the market

Challenges:

- Must handle **uncertainty** about agent preferences
- Agents can strategically **hold out** for better matches
    - Leads to congestion and cannibalization in the market

Our Model: Algorithmic Results and Takeaways

Develop efficient 4-approximation algorithm for the search design problem:

- algorithm attains 1/4 of the first-best welfare
- (approximation necessary—even 1.01-approximation is NP-hard)

Our Model: Algorithmic Results and Takeaways

Develop efficient 4-approximation algorithm for the search design problem:

- algorithm attains 1/4 of the first-best welfare
- (approximation necessary—even 1.01-approximation is NP-hard)

How does it work? Restrict meetings to submarkets (of types) with simple structure

Our Model: Algorithmic Results and Takeaways

Develop **efficient 4-approximation algorithm** for the search design problem:

- algorithm attains 1/4 of the **first-best welfare**
- (**approximation necessary**—even 1.01-approximation is NP-hard)

How does it work? Restrict meetings to submarkets (of types) with simple structure

Takeaway: Through careful search design, platform can induce equilibrium outcome with almost (socially) optimal welfare

Related Work

Search and matching. Long history of search in the context of matching markets: Burdett and Coles (*QJE*, 1997), Shimer and Smith (*Econometrica*, 2000), Adachi (*JET*, 2003), ... But these works assume *random meeting* of agents and do not consider the problem of designing search

Online matching platforms. Rios et al. (EC 2021) empirically study assortment optimization for online dating; Kanoria and Saban (*Manag. Sci.*, 2021) study online marketplace design with search frictions; Shi (EC 2020) studies matchmaking strategies on two-sided matching platforms; Halaburda et al. (*Manag. Sci.*, 2018) model the effect of restricting choice; Ashlagi et al. (WINE 2020) study two-sided assortment optimization with a multinomial logit choice function; ...

Model

Agents, Types, and Dynamics

Continuum model—agents have infinitesimal mass

Each agent has one of finitely many **types** $\boldsymbol{\theta} \in \boldsymbol{\Theta}$

Types divided into two sides: $\Theta = \mathcal{M} \sqcup \mathcal{W}$

Continuum model—agents have infinitesimal mass

Each agent has one of finitely many **types $\boldsymbol{\theta} \in \boldsymbol{\Theta}$**

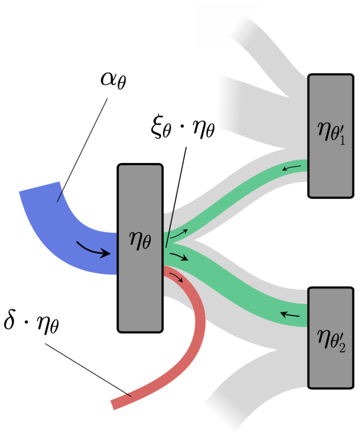
Types divided into two sides: $\Theta = \mathcal{M} \sqcup \mathcal{W}$

Agents of type θ enter at exogenous **arrival rate $\boldsymbol{\alpha_\theta}$**

Departure is endogenous—agents either leave:

(a) **matched**—by entering into a mutually agreed upon match, or

(b) **unmatched**—by experiencing a "life event" at an exogenous (per-agent) rate of $\boldsymbol{\delta\, dt}$



Agents, Types, and Dynamics

Continuum model—agents have infinitesimal mass

Each agent has one of finitely many types $\theta \in \Theta$
Types divided into two sides: $\Theta = \mathcal{M} \sqcup \mathcal{W}$

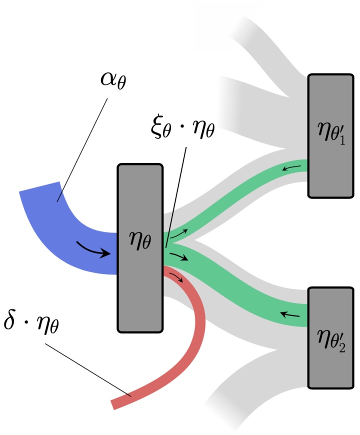
Agents of type θ enter at exogenous arrival rate α_θ

Departure is endogenous—agents either leave:

(a) matched—by entering into a mutually agreed upon match, or

(b) unmatched—by experiencing a “life event” at an exogenous (per-agent) rate of $\delta \, dt$

Each type has a stationary population mass η_θ from balancing inflows against outflows



Preferences and Utilities

Agents have random, symmmetric, cardinal utilities determined by their types:

- each pair of types $(m, w) \in \mathcal{M} \times \mathcal{W}$ has continuous utility distribution F_{mw}
- each pair of type m and w agents has shared utility for matching $\sim F_{mw}$
- utility for each pair of agents is drawn independently

Preferences and Utilities

Agents have random, symmmetric, cardinal utilities determined by their types:

- each pair of types $(m, w) \in \mathcal{M} \times \mathcal{W}$ has continuous utility distribution F_{mw}
- each pair of type m and w agents has shared utility for matching $\sim F_{mw}$
- utility for each pair of agents is drawn independently

Randomness represents idiosyncratic compatibility, unknown to platform

- For each pair of agents, only revealed when they meet

Preferences and Utilities

Agents have random, symmmetric, cardinal utilities determined by their types:

- each pair of types $(m, w) \in \mathcal{M} \times \mathcal{W}$ has continuous utility distribution F_{mw}
- each pair of type m and w agents has shared utility for matching $\sim F_{mw}$
- utility for each pair of agents is drawn independently

Randomness represents idiosyncratic compatibility, unknown to platform

- For each pair of agents, only revealed when they meet

Conversely, assume platform knows distributions F_{mw} for all $(m, w) \in \mathcal{M} \times \mathcal{W}$

- Platform's data $\implies$ distributional knowledge

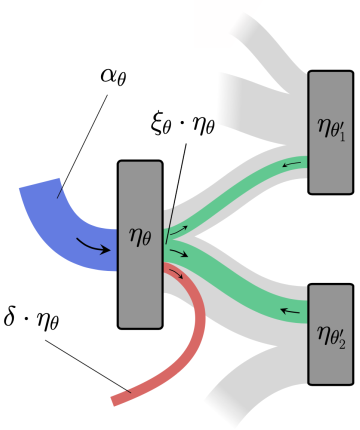
Directed Search

Platform sets rates at which pairs of types meet

- Type θ agents meet type θ' agents according to Poisson process of **rate $\lambda_\theta(\theta')$**

Upon pair of type m and w agents meeting, each sees utility $u \sim F_{mw}$, then **accepts**/**rejects** match

- Match (and leave) with utility u iff both accept



Directed Search

Platform sets rates at which pairs of types meet

- Type θ agents meet type θ' agents according to Poisson process of **rate $\lambda_\theta(\theta')$**

Upon pair of type m and w agents meeting, each sees utility $u \sim F_{mw}$, then **accepts**/**rejects** match

- Match (and leave) with utility u iff both accept

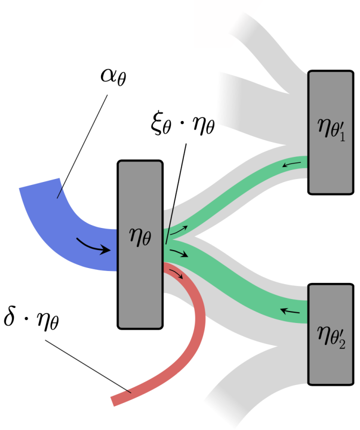
Rates $\lambda_\theta(\theta')$ subject to feasibility constraints:

(a) **capacity** constraint: ≤ 1 meeting / unit time

$$\sum_{\theta'} \lambda_\theta(\theta') \leq 1$$

(b) **flow** constraint: equal #s of any two types meet

$$\eta_m \lambda_m(w) = \eta_w \lambda_w(m)$$



The Design Problem

The Platform's Design Problem

Platform sets rates subject to feasibility constraints (capacity + flow)

Feasible choice of rates $\implies$ game among agents

- **Structural result:** unique Nash equilibrium always exists!

Platform optimizes over Nash equilibria of the induced games that are sustainable in stationary equilibrium

The Platform's Design Problem

Platform sets rates subject to feasibility constraints (capacity + flow)

Feasible choice of rates $\implies$ game among agents

- **Structural result:** unique Nash equilibrium always exists!

Platform optimizes over Nash equilibria of the induced games that are sustainable in stationary equilibrium

Platform's objective: maximize social welfare in stationary equilibrium

The Optimization Program

The resulting platform optimization problem:

$$\max_{\lambda_\theta, \tau_\theta, \xi_\theta, \eta_\theta} \quad 2 \cdot \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}} \left(\eta_m \lambda_m(w) \int_{\max(\tau_m, \tau_w)} u \, dF_{mw} \right)$$

such that

$$\alpha_\theta = (\delta + \xi_\theta)\eta_\theta \qquad \text{(stationarity)}$$

$$\eta_m \lambda_m(w) = \eta_w \lambda_w(m) \qquad \text{(flow)}$$

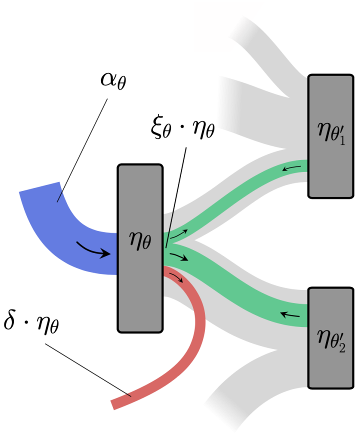
$$1 \geq \sum_{\theta'} \lambda_\theta(\theta') \qquad \text{(capacity)}$$

$$\xi_\theta = \sum_{\theta'} \left(\lambda_\theta(\theta') \int_{\max(\tau_\theta, \tau_{\theta'})}^{\infty} dF_{\theta\theta'} \right) \qquad (\xi_\theta \text{ defn})$$

$$\tau_\theta = \frac{1}{\delta + \xi_\theta} \sum_{\theta'} \left(\lambda_\theta(\theta') \int_{\max(\tau_\theta, \tau_{\theta'})}^{\infty} u \, dF_{\theta\theta'} \right)$$

(agents' best response)

$$\lambda_m(w), \lambda_w(m) \geq 0. \qquad \text{(nonnegativity of rates)}$$



The resulting platform optimization problem:

$$\max_{\lambda_\theta, \tau_\theta, \xi_\theta, \eta_\theta} \quad 2 \cdot \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}} \left(\eta_m \lambda_m(w) \int_{\max(\tau_m, \tau_w)} u \, dF_{mw} \right)$$

such that

$$\alpha_\theta = (\delta + \xi_\theta)\eta_\theta \qquad \text{(stationarity)}$$

$$\eta_m \lambda_m(w) = \eta_w \lambda_w(m) \qquad \text{(flow)}$$

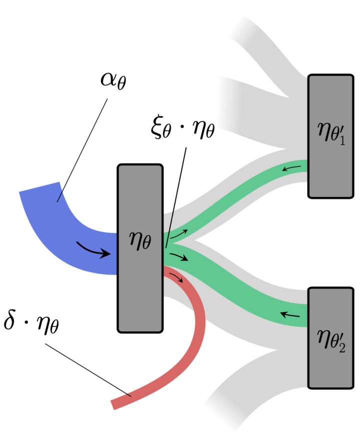
$$1 \geq \sum_{\theta'} \lambda_\theta(\theta') \qquad \text{(capacity)}$$

$$\xi_\theta = \sum_{\theta'} \left(\lambda_\theta(\theta') \int_{\max(\tau_\theta, \tau_{\theta'})}^{\infty} dF_{\theta\theta'} \right) \qquad (\xi_\theta \text{ defn})$$

$$\tau_\theta = \frac{1}{\delta + \xi_\theta} \sum_{\theta'} \left(\lambda_\theta(\theta') \int_{\max(\tau_\theta, \tau_{\theta'})}^{\infty} u \, dF_{\theta\theta'} \right)$$

(agents' best response)

$$\lambda_m(w), \lambda_w(m) \geq 0. \qquad \text{(nonnegativity of rates)}$$



Approximation Algorithm

Algorithmic Result

Theorem

The platform can efficiently find a 4-approximately optimal search design.

Welfare of our construction **4-approximates** welfare of the **first-best** outcome

- First best = "planned market" where platform can choose agent strategies

Algorithmic Result

Theorem

The platform can efficiently find a 4-approximately optimal search design.

Welfare of our construction **4-approximates** welfare of the **first-best** outcome

- First best = "planned market" where platform can choose agent strategies

Algorithm sketch: Construct approximately optimal search design in two phases:

1. **Solve for the first best.** Exactly computing the first-best outcome turns out to be a computationally tractable problem
2. **Approximate first best via star-shaped submarkets.** Divide market into submarkets based on first best; then reintroduce incentives

Phase I: Finding the First Best

1. Relax agent best response constraint
2. Rewrite optimization problem in terms of a “cutoff” for each pair of types
3. Messy initial optimization problem reduces to a generalized assignment problem!

$$
\begin{aligned}
\max_{\lambda_\theta, \tau_\theta, \xi_\theta, \eta_\theta} \quad & 2 \cdot \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}} \left(\eta_m \lambda_m(w) \int_{\max(\tau_m, \tau_w)} u \, dF_{mw} \right) \\
\text{such that} \quad & \alpha_\theta = (\delta + \xi_\theta)\eta_\theta \\
& \eta_m \lambda_m(w) = \eta_w \lambda_w(m) \\
& 1 \geq \sum_{\theta'} \lambda_\theta(\theta') \\
& \xi_\theta = \sum_{\theta'} \left(\lambda_\theta(\theta') \int_{\max(\tau_\theta, \tau_{\theta'})}^{\infty} dF_{\theta\theta'} \right) \\
& \cancel{\tau_\theta = \frac{1}{\delta + \xi_\theta} \sum_{\theta'} \left(\lambda_\theta(\theta') \int_{\max(\tau_\theta, \tau_{\theta'})}^{\infty} u \, dF_{\theta\theta'} \right)} \\
& \lambda_m(w), \lambda_w(m) \geq 0.
\end{aligned}
$$

Phase I: Finding the First Best

1. Relax agent best response constraint
2. Rewrite optimization problem in terms of a "cutoff" for each pair of types
3. Messy initial optimization problem reduces to a generalized assignment problem!

$$
\begin{aligned}
\max_{\beta_{mw}} \quad & \sum_{m\in\mathcal{M}} \sum_{w\in\mathcal{W}} \rho_{mw} \cdot \beta_{mw} \\
\text{such that} \quad & \sum_{w\in\mathcal{W}} \beta_{mw} \leq \alpha_m \quad \forall m \in \mathcal{M} \\
& \sum_{m\in\mathcal{M}} \beta_{mw} \leq \alpha_w \quad \forall w \in \mathcal{W},
\end{aligned}
$$

Phase I: Finding the First Best

1. Relax agent best response constraint
2. Rewrite optimization problem in terms of a "cutoff" for each pair of types
3. Messy initial optimization problem reduces to a generalized assignment problem!

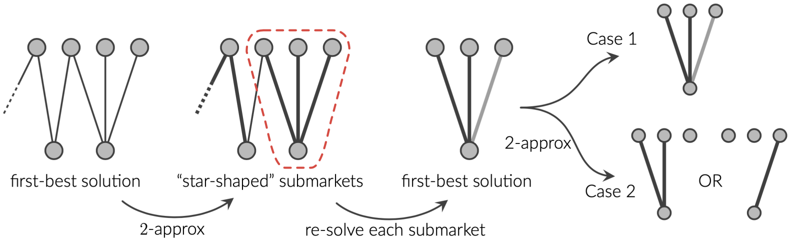
$$
\begin{aligned}
\max_{\beta_{mw}} \quad & \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}} \rho_{mw} \cdot \beta_{mw} \\
\text{such that} \quad & \sum_{w \in \mathcal{W}} \beta_{mw} \leq \alpha_m \quad \forall m \in \mathcal{M} \\
& \sum_{m \in \mathcal{M}} \beta_{mw} \leq \alpha_w \quad \forall w \in \mathcal{W},
\end{aligned}
$$

Phase I: Finding the First Best

1. Relax agent best response constraint
2. Rewrite optimization problem in terms of a "cutoff" for each pair of types
3. Messy initial optimization problem reduces to a **generalized assignment problem**!
   - Optimal solution has the structure of a **tree over agent types**

$$
\begin{aligned}
\max_{\beta_{mw}} \quad & \sum_{m \in \mathcal{M}} \sum_{w \in \mathcal{W}} \rho_{mw} \cdot \beta_{mw} \\
\text{such that} \quad & \sum_{w \in \mathcal{W}} \beta_{mw} \leq \alpha_m \quad \forall m \in \mathcal{M} \\
& \sum_{m \in \mathcal{M}} \beta_{mw} \leq \alpha_w \quad \forall w \in \mathcal{W},
\end{aligned}
$$

Phase II: Approximation via Star-shaped Submarkets



1. 2-approximate (tree-shaped) first-best solution with *star*-shaped submarkets
   - Akin to Lenstra et al. (*Math. Program.*, 1990), Banerjee et al. (WWW 2017)
2. Re-solve for the first-best solution in each submarket
3. Adjust the new first-best in each submarket into a stationary equilibrium outcome while losing at most a 2-factor in welfare

Conclusion

Summary + Takeaways

In this work, we:

- Connected match recommendations to search design for matching markets
- Investigated the challenges of making match recommendations when facing incomplete knowledge of preferences and strategic agents
- Developed an efficient algorithm to find an approximately optimal search design for general preference distributions

Summary + Takeaways

In this work, we:

- Connected match recommendations to search design for matching markets
- Investigated the challenges of making match recommendations when facing incomplete knowledge of preferences and strategic agents
- Developed an efficient algorithm to find an approximately optimal search design for general preference distributions

Takeaway: Through careful search design—which can involve limiting agents' choice, the platform can induce equilibrium outcome with almost (socially) optimal welfare