OT

Andrew Bai

1 The Economic Problem

1.1 Setting

A **monopolist** designs a menu of products to sell to a heterogeneous population of consumers.

Players:

- **Firm**: Chooses which products to offer and at what prices
- **Consumers**: Each has private willingness-to-pay (WTP) for product attributes
- **Goal**: Maximize profit subject to individual rationality (IR) and incentive compatibility (IC)

Example:

- Products: Software bundles with features {basic, analytics, collaboration, API access}
- Consumers: Companies with different valuations for each feature
- Firm must offer a menu (e.g., "Basic $10", "Pro $25", "Enterprise $50") without knowing each consumer's exact WTP

1.2 Mathematical Primitives

1.2.1 Consumer Side

Consumer types: Each consumer i is characterized by their WTP vector

$$v_i \in \mathbb{R}^d$$

where d is the number of attributes (features).

Interpretation: $v_i[j]$ = consumer i's WTP for attribute j

Population: We model consumers as draws from a distribution

$$\mu \in \mathcal{P}(\mathbb{R}^d)$$

where $\mathcal{P}(\mathbb{R}^d)$ is the space of probability measures on $\mathbb{R}^d$.

For computation, we observe a finite sample:

$$\{v_1, v_2, \ldots, v_N\} \sim \mu$$

1.2.2 Product Side

Bundles: A product is a vector of attributes

$$b \in \{0,1\}^d$$

where $b[j] = 1$ if feature j is included, 0 otherwise.

Prices: Each bundle has a price

$$p \in \mathbb{R}_+$$

Products in joint space: A product is the PAIR

$$x = (b, p) \in \{0,1\}^d \times \mathbb{R}_+$$

This is the key conceptual shift: **price is not separate from the product – it's part of the product definition**.

1.2.3 Menu

A **menu** M is a finite collection of products:

$$M = \{x_1, x_2, \ldots, x_L\} \tag{1}$$
$$= \{(b_1, p_1), (b_2, p_2), \ldots, (b_L, p_L)\} \tag{2}$$

where L is the menu size.

1.3 Consumer Preferences

Utility: Consumer with type v derives utility from product $x = (b, p)$:

$$u(v, x) = v \cdot b - p$$

This is **quasilinear utility**:

- Positive component: $v \cdot b$ (value from features)
- Negative component: p (cost of purchase)

Outside option: Consumers can choose not to buy, giving utility 0.
Choice: Faced with menu M, consumer v chooses:

$$x^*(v) = \text{argmax}_{x \in M \cup \{\emptyset\}} u(v, x)$$

where $\emptyset$ denotes the outside option with $u(v, \emptyset) = 0$.

1.4 Firm's Problem

Cost function: Producing bundle b costs

$$C(b) = c_1 \cdot |b| + c_2 \cdot |b|^\alpha$$

where $|b|$ = number of features in bundle (typically $\alpha \geq 1$ for convex costs).
Profit: If consumer v chooses product $x = (b, p)$, firm earns

$$\pi(v, x) = \begin{cases} p - C(b) & \text{if } u(v, x) \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Total profit: Expected profit over consumer distribution μ:

$$\Pi(M) = \int \pi(v, x^*(v)) d\mu(v)$$

Firm's problem:

$$\max_M \quad \Pi(M) \tag{3}$$
$$\text{subject to:} \quad \text{IC: } x^*(v) = \text{argmax}_{x \in M} u(v, x) \quad \forall v \text{ (incentive compatibility)} \tag{4}$$
$$\text{IR: } u(v, x^*(v)) \geq 0 \quad \forall v \text{ (individual rationality)} \tag{5}$$

2 Optimal Transport Formulation

2.1 Why This Is Optimal Transport

The key insight is that the firm's problem is equivalent to **choosing a discrete target measure** in the joint (b, p) space.

Source measure: Consumer types distributed according to μ

Target measure: Menu represented as

$$\nu = \sum_{m=1}^{L} w_m \cdot \delta_{x_m}$$

where:

- $x_m = (b_m, p_m)$ is the m-th product (an atom in joint space)
- w_m is the mass assigned to product m (market share)
- δ_{x_m} is a Dirac mass at x_m

Transport plan: The assignment of consumers to products is a coupling

$$\gamma \in \Pi(\mu, \nu)$$

where $\Pi(\mu, \nu)$ is the set of joint distributions with marginals μ and ν.

IC constraint in OT language: The transport plan γ must be optimal with respect to the utility cost:

$$\gamma = \text{argmin}_{\pi \in \Pi(\mu,\nu)} \int \int c(v, x) d\pi(v, x)$$

where the **cost function** is:

$$c(v, x) = -(v \cdot b - p) \quad \text{(negative utility)}$$

This enforces that consumers choose utility-maximizing products!

2.2 The Monge-Kantorovich Formulation

Primal problem (Monge-Kantorovich):

$$\max_{\nu} \quad \int \max_{x \in \text{supp}(\nu)} \{v \cdot b - p\} d\mu(v) - \int C(b) d\nu(b, p) \tag{6}$$

$$\text{subject to:} \quad \nu = \sum_{m=1}^{L} w_m \delta_{(b_m, p_m)} \tag{7}$$

$$w_m \geq 0, \sum_m w_m = 1 \tag{8}$$

$$b_m \in \{0, 1\}^d, p_m \in \mathbb{R}_+ \tag{9}$$

Equivalently (since we're maximizing profit):

$$\max_{(b_1,p_1),\ldots,(b_L,p_L)} \sum_{m=1}^{L} w_m(b, p) \cdot [p_m - C(b_m)]$$

where:

$$w_m(b, p) = \mu(\{v : m \in \text{argmax}_j (v \cdot b_j - p_j)\})$$

This is the **semi-discrete optimal transport problem**: continuous source (μ), discrete target (finite support).

2.3 Kantorovich Dual

The **dual problem** provides important structure:

$$\max_{\varphi,\psi} \quad \int \varphi(v) d\mu(v) + \sum_{m=1}^{L} \psi_m w_m \tag{10}$$

$$\text{subject to:} \quad \varphi(v) + \psi_m \leq v \cdot b_m - p_m \quad \forall v, \forall m \tag{11}$$

Dual variables:

- $\varphi(v)$: Consumer surplus (utility of optimal choice)
- ψ_m: Shadow price of product m

Complementary slackness: At optimum,

$$\varphi(v) + \psi_m = v \cdot b_m - p_m \iff \text{consumer } v \text{ is assigned to product } m$$

Key insight: The dual variables encode:

- $\varphi(v) = \max_m \{v \cdot b_m - p_m\}$: IC constraint (utility-maximizing choice)
- $\varphi(v) \geq 0$: IR constraint (participation)

So the IC/IR constraints from mechanism design are **automatically encoded** in the OT dual structure!

3 Computational Algorithm

3.1 Semi-Discrete OT via Lloyd's Algorithm

The **Lloyd's algorithm** (generalized k-means) solves semi-discrete OT by iteratively:

1. **E-step**: Assign source points to nearest target atom
2. **M-step**: Update target atoms to minimize within-cluster cost

For our problem:

3.1.1 E-step: Consumer Assignment

Given current menu $\{x_1, \ldots, x_L\}$, assign each consumer:

$$m^*(v) = \text{argmax}_{m \in \{1,\ldots,L\}} u(v, x_m) \tag{12}$$

$$= \text{argmax}_m (v \cdot b_m - p_m) \tag{13}$$

This is the **OT assignment** – minimum cost transport.
Output: Assignment function $m^* : \text{supp}(\mu) \to \{1, \ldots, L\} \cup \{\emptyset\}$

3.1.2 M-step: Atom Update (Voronoi Centroid)

For each product m, update (b_m, p_m) to maximize profit from assigned consumers:

$$(b_m^*, p_m^*) = \text{argmax}_{b,p} \sum_{v:m^*(v)=m} [p - C(b)] \cdot \mathbb{1}\{v \cdot b \geq p\}$$

This is the **Voronoi centroid** in joint (b, p) space.
Key observation: This optimization decomposes!

3.2 Decomposition of the M-Step

Problem: Given assigned consumers $V_m = \{v : m^*(v) = m\}$, find optimal (b, p).
Decomposition:
Step 1: Fix p, optimize b
For fixed price p, the optimal bundle is:

$$b^*(p) = \text{argmax}_{b \in \{0,1\}^d} |\{v \in V_m : v \cdot b \geq p\}| \cdot (p - C(b))$$

This is a **discrete optimization** over 2^d bundles.
Algorithm: Enumerate all bundles, count demand at price p, compute profit.
Step 2: Fix b, optimize p
For fixed bundle b, the optimal price is found by **threshold pricing**:

$$p^*(b) = \text{argmax}_{p \geq C(b)} |\{v \in V_m : v \cdot b \geq p\}| \cdot (p - C(b))$$

Key insight: Demand is monotone decreasing in p, so optimal p is at a consumer's valuation threshold.
Algorithm:

1. Compute valuations: $T_i = v_i \cdot b$ for $i \in V_m$
2. Sort in descending order: $T_{(1)} \geq T_{(2)} \geq \cdots \geq T_{(n)}$
3. For each k, price $p = T_{(k)}$ gives k buyers at margin $T_{(k)} - C(b)$
4. Choose k^* that maximizes $k \cdot (T_{(k)} - C(b))$

Step 3: Coordinate ascent
Alternate between fixing b and fixing p until convergence.
In practice: For small d (e.g., $d = 4$), we enumerate all 2^d bundles and use threshold pricing for each, finding the global optimum directly.

3.3 Complete Algorithm

3.4 Subroutine: Joint (b, p) Optimization

Complexity: $\mathcal{O}(2^d \cdot |V_m| \cdot \log |V_m|)$ per atom update.
For $d = 4$: only 16 bundles to check, very fast.

Algorithm 1 Semi-Discrete OT for Menu Design

Input: Consumer WTP sample $V = \{v_1, \ldots, v_N\} \sim \mu$, Menu size L, Max iterations T
Output: Optimal menu $M^* = \{(b_1^*, p_1^*), \ldots, (b_L^*, p_L^*)\}$

1. Initialize:
for $m = 1$ to L **do**
  Sample or heuristically choose $(b_m^{(0)}, p_m^{(0)})$
end for

2. Main Loop:
for $t = 1$ to T **do**

  // E-step: Assign consumers (OT assignment)
  for each consumer i **do**
    $m_i = \text{argmax}_m (v_i \cdot b_m^{(t-1)} - p_m^{(t-1)})$
    if $\max_m (v_i \cdot b_m^{(t-1)} - p_m^{(t-1)}) < 0$ **then**
      $m_i = \emptyset$ *// outside option*
    end if
  end for

  // M-step: Update atoms (Voronoi centroids)
  for each product m **do**
    $V_m = \{v_i : m_i = m\}$ *// assigned consumers*
    if $|V_m| > 0$ **then**
      $(b_m^{(t)}, p_m^{(t)}) = \text{argmax}_{b,p} \sum_{v \in V_m} [p - C(b)] \cdot \mathbb{1}\{v \cdot b \geq p\}$
    else
      $(b_m^{(t)}, p_m^{(t)}) = (b_m^{(t-1)}, p_m^{(t-1)})$
    end if
  end for

  // Check convergence
  if $\|\text{atoms}^{(t)} - \text{atoms}^{(t-1)}\| < \varepsilon$ **then**
    break
  end if
end for

return $M^* = \{(b_m^{(t)}, p_m^{(t)})\}_{m=1}^{L}$

Algorithm 2 UpdateAtom

```
Input: Assigned consumers V_m, cost function C
Output: Optimal (b*, p*)

best_profit ← −∞
best_b ← None
best_p ← None

for each bundle b ∈ {0,1}^d do
  // Compute valuations
  T ← {v · b : v ∈ V_m}

  // Sort descending
  T_sorted ← sort(T, descending=True)

  // Try each threshold
  for k = 1 to |T_sorted| do
    p ← T_sorted[k]
    if p < C(b) then
      break
    end if

    // Profit: k buyers at margin (p − C(b))
    profit ← k · (p − C(b))

    if profit > best_profit then
      best_profit ← profit
      best_b ← b
      best_p ← p
    end if
  end for
end for

return (best_b, best_p)
```

Algorithm 3 Multi-Start Lloyd's

```
Input: V, L, restarts R

best_profit ← −∞
best_menu ← None

for r = 1 to R do
   // Random or heuristic initialization
   M_0 ← Initialize(V, L, seed=r)

   // Run Lloyd's from this init
   M_r ← Lloyd(V, L, init=M_0)

   // Evaluate
   profit_r ← EvaluateProfit(M_r, V)

   if profit_r > best_profit then
      best_profit ← profit_r
      best_menu ← M_r
   end if
end for

return best_menu
```

3.5 Multi-Start Strategy

Because the problem is **non-convex** (discrete bundles), we use multi-start:

Typical choice: $R = 10$–20 restarts.

4 Theoretical Properties

4.1 Convergence of Lloyd's Algorithm

Theorem 1 (Lloyd's Convergence)**.** *The Lloyd's algorithm converges to a local optimum in finite iterations.*

Work in Progress!

Note: Global optimum NOT guaranteed (NP-hard problem).

4.2 Optimality Conditions (Kantorovich Dual)

At a local optimum, the **complementary slackness conditions** hold:

$$\varphi(v) + \psi_m = v \cdot b_m - p_m \quad \Longleftrightarrow \quad \text{consumer } v \text{ assigned to product } m \tag{14}$$

$$\varphi(v) + \psi_m \geq v \cdot b_m - p_m \quad \Longleftrightarrow \quad \text{otherwise} \tag{15}$$

where:

- $\varphi(v) = \max_m \{v \cdot b_m - p_m\}$: consumer surplus

- $\psi_m = -p_m$: shadow price (in standard OT normalization)

Interpretation:

- If consumer v is assigned to product m, they get exactly $\varphi(v)$ utility
- No consumer would strictly prefer any other product
- This encodes both IC (utility maximization) and IR ($\varphi(v) \geq 0$)

5 Summary

5.1 The Mathematical Framework

Problem: Design menu $M = \{(b_1, p_1), \ldots, (b_L, p_L)\}$ to maximize profit
OT Formulation:

- **Source**: Consumer distribution $\mu \in \mathcal{P}(\mathbb{R}^d)$
- **Target**: Discrete measure $\nu = \sum_m w_m \delta_{(b_m, p_m)}$
- **Cost**: $c(v, (b, p)) = -(v \cdot b - p) + C(b)$
- **Objective**: $\max_\nu \int \varphi(v) d\mu(v) - \sum_m C(b_m) w_m$

Algorithm: Semi-discrete OT via Lloyd's

- E-step: Assign consumers to utility-maximizing products
- M-step: Update each atom (b_m, p_m) to maximize profit from assigned consumers
- Multi-start: Run from multiple initializations

Theory:

- Dual potentials encode IC/IR
- Lloyd's converges to local optimum
- Complementary slackness at convergence

Extensions:

- Column generation for better bundle search
- Continuous relaxation for smooth optimization
- Dual diagnostics for optimality verification

6 Notation Reference

Sets:

- $\mathbb{R}^d$: d-dimensional Euclidean space
- $\mathbb{R}_+$: Non-negative reals
- $\{0, 1\}^d$: Binary vectors (bundles)
- $\mathcal{P}(X)$: Probability measures on X

Consumer side:

- $v \in \mathbb{R}^d$: WTP vector (type)
- $\mu \in \mathcal{P}(\mathbb{R}^d)$: Type distribution
- $V = \{v_1, \ldots, v_N\}$: Sample from μ

Product side:

- $b \in \{0, 1\}^d$: Bundle (features)
- $p \in \mathbb{R}_+$: Price
- $x = (b, p)$: Product atom
- $M = \{x_1, \ldots, x_L\}$: Menu

Functions:

- $u(v, x) = v \cdot b - p$: Utility
- $C(b)$: Production cost
- $\pi(v, x)$: Profit from consumer v buying x

OT objects:

- $\nu = \sum_m w_m \delta_{x_m}$: Target measure
- $\gamma \in \Pi(\mu, \nu)$: Transport plan
- $c(v, x)$: Transport cost
- $\varphi(v)$: Consumer surplus (dual potential)
- ψ_m: Shadow price (dual potential)

Algorithm:

- $m^*(v)$: Assignment function
- V_m: Consumers assigned to product m
- T: Max iterations
- R: Number of restarts