Mathematical Optimization in Data Science and Machine Learning/Decision-Making

HKUST-GZ, March 3, 2023

Yinyu Ye
Stanford University and CUHKSZ (Sabbatical Leave)
Today’s Talk

1. Accelerated Second-Order Methods and Applications

2. Online Linear Programming Algorithms and Applications

3. Mixed Integer Linear Programming Solver and Applications

4. Equitable Covering & Partition and Applications
I. Early Complexity Analyses for Nonconvex Optimization

\[ \min f(x), x \in X \text{ in } \mathbb{R}^n, \]

- where \( f \) is nonconvex and twice-differentiable,
  \[ g_k = \nabla f(x_k), H_k = \nabla^2 f(x_k) \]

- Goal: find \( x_k \) such that:
  \[ \| g_k \| \leq \epsilon \quad \text{(primary, first-order condition)} \]
  \[ \lambda_{\min}(H_k) \geq -\sqrt{\epsilon} \quad \text{(secondary, second-order condition)} \]

- First-order methods typically need \( O(n^2\epsilon^{-2}) \) operations
- Second-order methods typically need \( O(n^3\epsilon^{-1.5}) \) operations
- New? Yes, HSODM: a single-loop method with \( O(n^2\epsilon^{-1.75}) \) operations
Application I: HSODM for Policy Optimization in Reinforcement Learning

• Consider policy optimization of linearized objective in reinforcement learning

\[
\max_{\theta \in \mathbb{R}^d} L(\theta) := L(\pi_\theta),
\]

\[
\theta_{k+1} = \theta_k + \alpha_k \cdot M_k \nabla \eta(\theta_k),
\]

• \( M_k \) is usually a preconditioning matrix.

• The Natural Policy Gradient (NPG) method (Kakade, 2001) uses the Fisher information matrix where \( M_k \) is the inverse of

\[
F_k(\theta) = \mathbb{E}_{\rho_{\theta_k},\pi_{\theta_k}} \left[ \nabla \log \pi_{\theta_k}(s, a) \nabla \log \pi_{\theta_k}(s, a)^T \right]
\]

• Based on KL divergence, TRPO (Schulman et al. 2015) uses KL divergence in the constraint:

\[
\max_{\theta} \nabla L_{\theta_k}(\theta_k)^T (\theta - \theta_k)
\]

\[
\text{s.t. } \mathbb{E}_{s \sim \rho_{\theta_k}} \left[ D_{KL}(\pi_{\theta_k}(\cdot | s); \pi_{\theta}(\cdot | s)) \right] \leq \delta.
\]

\[\text{Homogeneous NPG: Apply HSODM!}\]
Preliminary Results: HSODM for Policy Optimization in RL

• A comparison of Homogeneous NPG and Trust-region Policy Optimization (Schultz, 2015)

[Graphs showing performance comparisons for HalfCheetah-v2, Hopper-v2, Walker2d-v2, and Ant-v2]

• HSODM provides significant improvements over TRPO

• Ongoing: second-order information of L?

• Further reduce the computation cost per step
Dimension Reduced Second-Order Method (DRSOM)

- Motivation from Multi-Directional FOM and Subspace Method, such as CG and ADAM, DRSOM applies the trust-region method in low dimensional subspace.
- This results in a low-dimensional quadratic sub-minimization problem:
- Typically, DRSOM adopts two directions \( d = -\alpha^1 \nabla f(x_k) + \alpha^2 d_k \)

  where \( g_k = \nabla f(x_k), H_k = \nabla^2 f(x^k), d_k = x_k - x_{k-1} \)
- Then we solve a 2-d quadratic minimization problem:

  \[
  \min m_k^\alpha (\alpha) := f(x_k) + (c_k)^T \alpha + \frac{1}{2} \alpha^T Q_k \alpha \\
  \|\alpha\|_{G_k} \leq \Delta_k
  \]

  \[
  G_k = \begin{bmatrix} g_k^T g_k & -g_k^T d_k \\ -g_k^T d_k & d_k^T d_k \end{bmatrix}, 
  Q_k = \begin{bmatrix} g_k^T H_k g_k & -g_k^T H_k d_k \\ -g_k^T H_k d_k & d_k^T H_k d_k \end{bmatrix}, 
  c_k = \begin{bmatrix} -\|g_k\|^2 \\ g_k^T d_k \end{bmatrix}
  \]
Test Results: HSODM and DRSOM + HSODM

CUTEst example

- **GD** and **LBFGS** both use a Line-search (Hager-Zhang)
- **DRSOM** uses 2-D subspace
- HSODM and DRSOM + HSODM are much better!
- **DRSOM** can also benefit from the homogenized system
Application II: Neural Networks and Deep Learning

To use DRSOM in machine learning problems

• We apply the mini-batch strategy to a vanilla DRSOM
• Use Automatic Differentiation to compute gradients
• Train ResNet18/Resnet34 Model with CIFAR 10
• Set Adam with initial learning rate 1e-3
Preliminary Results: Neural Networks and Deep Learning

Training and test results for ResNet18 with DRSOM and Adam

Pros

• DRSOM has rapid convergence (30 epochs)
• DRSOM needs little tuning

Cons

• DRSOM may over-fit the models
• Running time can benefit from Interpolation
• Single direction DRSOM is also good

Good potential to be a standard optimizer for deep learning!

Training and test results for ResNet34 with DRSOM and Adam (https://arxiv.org/abs/2208.00208)
Application III: Sensor Network Location (SNL)

- Localization
  - Given partial pair-wise measured distance values
  - Given some anchors’ positions
  - Find locations of all other sensors that fit the measured distance values
    This is also called graph realization on a fixed dimension Euclidean space
A Unit Disk Ad-Hoc Network
Mathematical Formulation of Sensor Network Location (SNL)

• Consider Sensor Network Location (SNL)

\[ N_x = \{(i, j) : \|x_i - x_j\| = d_{ij} \leq r_d\}, \quad N_a = \{(i, k) : \|x_i - a_k\| = d_{ik} \leq r_d\} \]

where \( r_d \) is a fixed parameter known as the radio range. The SNL problem considers the following QCQP feasibility problem,

\[
\begin{align*}
\|x_i - x_j\|^2 &= d_{ij}^2, \forall (i, j) \in N_x \\
\|x_i - a_k\|^2 &= d_{ik}^2, \forall (i, k) \in N_a
\end{align*}
\]

• Alternatively, one can solve SNL by the nonconvex nonlinear least square (NLS) problem

\[
\min_X \sum_{(i<j,j) \in N_x} (\|x_i - x_j\|^2 - d_{ij}^2)^2 + \sum_{(k,j) \in N_a} (\|a_k - x_j\|^2 - d_{k,j}^2)^2.
\]
Semidefinite Programming Relaxation

**Step 1: Linearization**

\[
\| x_i - x_j \|^2 = x_j^T x_i - 2 x_j^T x_i + x_i^T x_j \\
Y_{ii} \quad Y_{ij} \quad Y_{ij}
\]

\[
\| a_k - x_j \|^2 = a_k^T a_k - 2 a_k^T x_j + x_j^T x_j \\
Y_{ji}
\]

Tighten: \( Y = X^T X, \ X=[x_1, \ldots, x_n] \)

**Step 2: Relax**

\[
Y \geq X^T X \iff Z = \begin{bmatrix}
I & X \\
X^T & Y
\end{bmatrix} \geq PSD
\]

This is a conic linear program which is a convex optimization problem, but \( O(n^{3.5} \log(\varepsilon^{-1})) \)

Biswas and Y 2004, So and Y 2005
Sensor Network Location (SNL) I

- Graphical results using SDP relaxation (Biswas&Y 2004, SO&Y 2007) to initialize the NLS
- $n = 80$, $m = 5$ (anchors), radio range = 0.5, degree = 25, noise factor = 0.05
- Both Gradient Descent and DRSOM can find good solutions!
• Graphical results without SDP relaxation
• DRSOM can still converge to optimal solutions
Sensor Network Location, Large-Scale Instances I

- Test large SNL instances (terminate at 3,000s and $|g_k| \leq 1e^{-5}$)

- Compare GD, CG, and DRSOM. (GD and CG use Hager-Zhang Linesearch)

| $n$ | $m$  | $|E|$ | $t$               |
|-----|------|-------|-------------------|
|     |      |       | CG    | DRSOM | GD    |
| 500 | 50   | 2.2e+04 | 1.7e+01 | 1.1e+01 | 2.3e+01 |
| 1000| 80   | 4.6e+04 | 7.3e+01 | 3.9e+01 | 1.8e+02 |
| 2000| 120  | 9.4e+04 | 2.5e+02 | 1.4e+02 | 1.1e+03 |
| 3000| 150  | 1.4e+05 | 6.5e+02 | 1.4e+02 | -       |
| 4000| 400  | 1.8e+05 | 1.3e+03 | 5.0e+02 | -       |
| 6000| 600  | 2.7e+05 | 2.0e+03 | 1.1e+03 | -       |
| 10000| 1000 | 4.5e+05 | -     | 2.2e+03 | -       |

Table 2: Running time of CG, DRSOM, and GD on SNL instances of different problem size, $|E|$ denotes the number of QCQP constraints. “-” means the algorithm exceeds 3,000s.

- DRSOM has the best running time (benefits of 2$^{nd}$ order info and interpolation!)
Sensor Network Location, Large-Scale Instances II

• Graphical results with 10,000 nodes and 1000 anchors (no noise) **within 3,000 seconds**

• GD with Line-search and Hager-Zhang CG both timeout

• DRSOM can converge to \( |g_k| \leq 1e^{-5} \) in 2,200s
Sensor Network Online Tracking, 2D and 3D
Topi 2. Online Linear Programming

1. Online learning theory and algorithm research (Agrawal et al. 2010, 14, Li & Y 2022)

What is an online learning problem?

- **Traditional machine learning problem:** There are a lot of (training) data, find the best model (examples: regression model, tree model)
  - Existing data → Best model

- **Online learning:** Data generation and learning occur simultaneously, influenced by decisions (e.g., multi-armed bandit problem)
  - Data → Decision → Feedback → Algorithm → Potential model

Need to learn and optimize一边一边.
Linear Programming and LP Giants won Nobel Prize...

\[
\max \sum \pi_j x_j \\
\text{s.t. } \sum a_j x_j \leq b, \\
0 \leq x_j \leq 1 \quad \forall \, j = 1, \ldots, n
\]
Online Auction Example

- There is a fixed selling period or number of buyers; and there is a fixed inventory of goods
- Customers come and require a bundle of goods and make a bid
- Decision: To sell or not to sell to each individual customer on the fly?
- Objective: Maximize the revenue.

<table>
<thead>
<tr>
<th>Bid #</th>
<th>$100</th>
<th>$30</th>
<th>....</th>
<th>...</th>
<th>...</th>
<th>Inventory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision</td>
<td>x1</td>
<td>x2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pants</td>
<td>1</td>
<td>0</td>
<td>....</td>
<td>...</td>
<td>...</td>
<td>100</td>
</tr>
<tr>
<td>Shoes</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>T-Shirts</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>500</td>
</tr>
<tr>
<td>Jackets</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>Hats</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>1000</td>
</tr>
</tbody>
</table>
Price Mechanism for Online Auction

- Learn and compute itemized optimal prices
- Use the prices to price each bid
- Accept if it is an over bid, and reject otherwise

<table>
<thead>
<tr>
<th>Bid #</th>
<th>$100</th>
<th>$30</th>
<th>....</th>
<th>...</th>
<th>...</th>
<th>Inventory</th>
<th>Price?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decision</td>
<td>x1</td>
<td>x2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pants</td>
<td>1</td>
<td>0</td>
<td>....</td>
<td>...</td>
<td>...</td>
<td>100</td>
<td>45</td>
</tr>
<tr>
<td>Shoes</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>45</td>
</tr>
<tr>
<td>T-Shirts</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>500</td>
<td>10</td>
</tr>
<tr>
<td>Jackets</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>200</td>
<td>55</td>
</tr>
<tr>
<td>Hats</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>1000</td>
<td>15</td>
</tr>
</tbody>
</table>
Jon Stewart Is Retiring, and it's Going to Be (Kind of) Okay

When the news broke Tuesday night that longtime Daily Show host Jon Stewart would be leaving his post in the coming months, the level of trauma on the Internet was palpable. Some expected topics arose, within hours -- minutes, even -- of the announcement trickling out. Why would Stewart leave now? What's his plan? Who should replace him? Could the next Daily Show host be a woman? (Of course.) Is this an elaborate ruse for Stewart to take over the NBC Nightly News? (Of course not).

The public conversation over the past two days has been so Stewart-centric that the retirement news effectively pushed NBC anchor Brian Williams's suspension off of social media's front pages. Part of that is the shock; we knew the other shoe was about to drop with (on?) Williams, but Stewart's departure was known only to Comedy Central brass before it was revealed to his studio audience. Part of it is how meme-worthy the parallels between the two hosts truly are ("fake newsman speaks truth, real newsman spins lies," some post on your Twitter timeline probably read). Breaking at
Revenues generated by different methods

- Total Revenue for impressions in T2 by Greedy and OLP with different allocation risk functions
Greedy exhausts budget of many advertisers early.

Log penalty keeps advertisers in budget but it is very conservative.

Exponential penalty keeps advertisers in budget until almost the end of the timeframe.

---

**# of Out-of-Budget Advertisers**

![Graph showing the number of out-of-budget advertisers over served impressions for different strategies: Greedy, Fixed Dual, Log, and Exponential.]
# Detailed Performances

<table>
<thead>
<tr>
<th>Allocation algorithm</th>
<th>Total Revenue</th>
<th>Improvement over greedy</th>
<th>Mid flight oob</th>
<th>Final oob</th>
</tr>
</thead>
<tbody>
<tr>
<td>Greedy</td>
<td>$1829.94</td>
<td>-</td>
<td>366</td>
<td>467</td>
</tr>
<tr>
<td>Fixed dual</td>
<td>$1986.67</td>
<td>8.5%</td>
<td>192</td>
<td>452</td>
</tr>
<tr>
<td>Log</td>
<td>$1915.72</td>
<td>4.6%</td>
<td>5</td>
<td>71</td>
</tr>
<tr>
<td>Exponential</td>
<td>$2043.21</td>
<td>11.6%</td>
<td>7</td>
<td>476</td>
</tr>
</tbody>
</table>

oob: out of budget

https://arxiv.org/abs/1407.5710
阿里巴巴在2019年云栖大会上提到在智能履行决策上使用OLP的算法
3.3 MCKP-Allocation

We adopt the primal-dual framework proposed by [2] to solve the problem defined in Equation 5. Let $\alpha$ and $\beta_j$ be the associated dual variables respectively. After obtaining the dual variables, we can solve the problem in an online fashion. Precisely, according to the principle of the primal-dual framework, we have the following allocation rule:

$$x_{ij} = \begin{cases} 
1, & \text{where } j = \arg \max_i (v_{ij} - \alpha c_j) \\
0, & \text{otherwise}
\end{cases} \quad (9)$$
Application V: The Online Algorithm can be Extended to Bandits with Knapsack (BwK) Applications

• For the previous problem, the decision maker first wait and observe the customer order/arm and then decide whether to accept/play it or not.

• An alternative setting is that the decision maker first decides which order/arm (s)he may accept/play, and then receive a random resource consumption vector $a_j$ and yield a random reward $\pi_j$ of the pulled arm.

• Known as the Bandits with Knapsacks, and it is a tradeoff exploration v.s. exploitation.
The decision variable $x_j$ represents the total-times of pulling the j-th arm.

We have developed a two-phase algorithm

- **Phase I**: Distinguish the optimal super-basic variables/arms from the optimal non-basic variables/arms with as fewer number of plays as possible
- **Phase II**: Use the arms in the optimal face to exhaust the resource through an adaptive procedure and achieve fairness

The algorithm achieves a problem dependent regret that bears a logarithmic dependence on the horizon $T$. Also, it identifies a number of LP-related parameters as the bottleneck or condition-numbers for the problem

- Minimum non-zero reduced cost
- Minimum singular-values of the optimal basis matrix.

**First algorithm** to achieve the $O(\log T)$ regret bound [Li, Sun & Y 2021 ICML] (https://proceedings.mlr.press/v139/li21s.html)

$$\begin{align*}
\max & \quad \sum \pi_j x_j \\
\text{s.t.} & \quad \sum_j a_j x_j \leq b, \quad x_j \geq 0 \quad \forall \ j = 1, \ldots, J
\end{align*}$$
Topic 3: Mixed Integer Linear Programming Solver

Application VI: Unit Commitment and Power Grid Optimization

COPT, Cardinal Operations 2022
Unit Commitment Problem

- Electricity is generated from units (various generators)
- Transmitted safely and stably through power grids
- Consumed at minimum (reasonable) price

*Optimization has its role to play*

\[
\text{minimize } \text{Cost of electricity} \\
\text{subject to } \text{Safety and Stability} \\
\text{Adaptivity to various units}
\]

Unit commitment problem dispatches the units safely and stably at minimum cost
Case Study: Sichuan Thermal-Hydro Hybrid Model

• A UC problem from real-life background (Sichuan Province)

• With 20 thermal and 230 hydro units

• Hydro units involve no decision (binary variables)

**Hardness**

• Costs are piecewise in generated power

• All the units are coupled by the Load balancing constraint

• A much larger and harder MILP model, but

Better Modeling + Algorithm Makes it Easier!
Successively Implemented in a Much Larger Region

- A much larger UC problem with security constraint
- With much more (millions of) constraints and variables
- More than 1000 units of Thermal, Hydro and New energy
- Consider interaction between regions and time periods

Huge size + Various business logic + Complicated coupling constraints

- Intractable without exploring structure
- Accurate and succinct model helps
- Domain specific algorithms matter a lot
- ML/AI has a big role to play

Model, Algorithm and ML/AI together make it tractable
Application VII: Beijing Public Transport Intelligent Urban Bus Operations Management with Mixed Fleet Types and Charging Schedule

Kickoff 2022.8
Peak Reduction due to Smart Charging and Discharging

<table>
<thead>
<tr>
<th></th>
<th>Standard</th>
<th>Low PGE</th>
<th>Linear Progr.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Fleet ($)</td>
<td>97,678</td>
<td>83,695</td>
<td>65,349</td>
</tr>
<tr>
<td>Mean Cost / Mile</td>
<td>0.068</td>
<td>0.044</td>
<td>0.0054</td>
</tr>
<tr>
<td>Increase in Peak</td>
<td>5.1%</td>
<td>1.4%</td>
<td>-0.25%</td>
</tr>
</tbody>
</table>
Background: Decision Intelligence in the case of Beijing Public Transport

More efficient and intelligent decision-making to achieve 14th Five-Year Plan goals

Beijing Public Transport Line 7 is selected as the Key Pilot Unit of the intelligent transformation of Beijing Public Transport

北京市“十四五规划”目标

加快构建“综合、绿色、智能、安全”的立体化现代化城市交通系统 加快建立科学、高效的“城市智能运行决策管理体系”

- 到2025年，中心城区绿色出行比例提高至76.5%
- 全面推进智慧城市建设，重点发展智慧交通
- 围绕轨道交通优化地面公交线网，减少长距离、长时间运行线路，提高车辆利用率

- 高水平推动城市交通的数字转型和智慧升级，形成城市交通整体解决方案
- 加快建设公共交通网络化智能调度体系，让公交出行越来越可靠，时间有保证

最大化工作效率 最小化总体运营成本

新能源车购车选型、车线匹配、能源布局、保养计划

运行优化、求解器、机器学习等智能决策技术
Intelligent Transformation
Empowered by Cardinal Operations

Beijing Public Transport’s total operational costs reached 6.65 billion Yuan in 2020, of which fuels, electricity, maintenance, repair and other indirect costs accounted for over 90%. Preliminary analysis shows various potential use cases for optimization in cost reduction.

Beijing Public Transport, in partner with Cardinal Operations, aims to build an innovative integrated system for smart operations in urban public transportation operations, and explore larger markets in the future.
Application VIII: Beijing-Shanghai High-speed Railway Scheduling Optimization

COPT, Cardinal Operations 2022
Background

• China High-speed Railway has been committed to providing high-quality transportation services to passengers, and the formulation of train scheduling is a key link in the operation.

• At present, train scheduling is based on human experience, which becomes increasingly difficult to handle the growing network. Therefore, both industry and academia are seeking ways to automate train scheduling.

• The train scheduling problem can be divided into Train Timetabling Problem (TTP) and Train Platforming Problem (TPP).

Optimization Model:

- **Objective:** maximize the number of trains placed in the train scheduling, thereby maximizing operating revenue;
- **Constraints:** describe the running behavior of trains and prevent train collisions;

The project mainly solves **TTP for Beijing-Shanghai High-speed Railway** and **TPP at Beijingnan Railway Station.**

- **Beijing-Shanghai High-speed Railway** is the busiest high-speed railway with the largest number of passengers in China. It is 1,318 km in total and passes 29 stations.

- **Beijingnan Railway Station** is the largest railway station in Beijing, with the largest area and the largest number of trains.

Both problems are challenging scheduling tasks, which can be formulated as Mixed Integer Programming (MIP).
Numerical Results: TTP for Beijing-Shanghai

• We solve the TTP for Beijing-Shanghai high-speed railway using Cardinal Optimizer (COPT).
• COPT is the first fully independently developed mathematical programming solver in China with strong solving ability of MIP problem. It also has excellent performance in solving this problem.
• The result is presented in the following figure. We only need about 1000 seconds to schedule 584 train in two directions.
Numerical Results: TPP at Beijingnan Station

- We solve the TPP at Beijingnan Railway Station using Cardinal Optimizer (COPT).
- Considering the connection pairs and ensuring the feasibility, we solve the model within 2 hours, which is much less than manual scheduling.
- The result is presented in the following table, including time nodes about occupation at boundaries and tracks.

<table>
<thead>
<tr>
<th>列车编号</th>
<th>前序车站</th>
<th>进站路径</th>
<th>出站路径</th>
<th>后序车站</th>
<th>进入站界时间</th>
<th>进入站界时间</th>
<th>离开站界时间</th>
<th>离开站界时间</th>
</tr>
</thead>
<tbody>
<tr>
<td>361</td>
<td>站界B10</td>
<td>站线XIV</td>
<td>站界89</td>
<td>站线10XIV</td>
<td>12:00</td>
<td>12:06</td>
<td>12:10</td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>庆坊</td>
<td>站界88</td>
<td>站线16:8</td>
<td>站界89</td>
<td>11:57</td>
<td>12:02</td>
<td>12:17</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td>站界B10</td>
<td>站线11</td>
<td>站界89</td>
<td>站线13:11</td>
<td>12:06</td>
<td>12:13</td>
<td>12:17</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>站界B10</td>
<td>站线7:17</td>
<td>站界89</td>
<td>站线10:40</td>
<td>12:14</td>
<td>12:29</td>
<td>12:34</td>
<td></td>
</tr>
<tr>
<td>251</td>
<td>站界B10</td>
<td>站线8</td>
<td>站界89</td>
<td>站线16:8</td>
<td>12:17</td>
<td>12:27</td>
<td>12:32</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>站界B10</td>
<td>站线7:17</td>
<td>站线17</td>
<td>站界87</td>
<td>12:19</td>
<td>12:23</td>
<td>12:25</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>站界B8</td>
<td>站线13:11</td>
<td>站界11</td>
<td>站界89</td>
<td>12:25</td>
<td>12:29</td>
<td>12:44</td>
<td></td>
</tr>
<tr>
<td>233</td>
<td>站界B10</td>
<td>站线17</td>
<td>站界89</td>
<td>站线7:17</td>
<td>12:29</td>
<td>12:44</td>
<td>12:48</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>庆坊</td>
<td>站界88</td>
<td>站线8:16</td>
<td>站界89</td>
<td>12:33</td>
<td>12:42</td>
<td>12:48</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>站界B10</td>
<td>站线16</td>
<td>站界89</td>
<td>站线8:16</td>
<td>12:42</td>
<td>12:57</td>
<td>13:01</td>
<td></td>
</tr>
<tr>
<td>572</td>
<td>庆坊</td>
<td>站界88</td>
<td>站线5:19</td>
<td>站界89</td>
<td>12:43</td>
<td>12:48</td>
<td>13:03</td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>站界B8</td>
<td>站线6:18</td>
<td>站线8:18</td>
<td>站界87</td>
<td>12:47</td>
<td>12:52</td>
<td>12:57</td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>站界B8</td>
<td>站线15:9</td>
<td>站线9</td>
<td>站线8:18</td>
<td>12:51</td>
<td>13:07</td>
<td>13:12</td>
<td></td>
</tr>
<tr>
<td>225</td>
<td>站界B10</td>
<td>站线18</td>
<td>站界89</td>
<td>站线6:18</td>
<td>12:57</td>
<td>13:12</td>
<td>13:17</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>站界B10</td>
<td>站线17</td>
<td>站界89</td>
<td>站线7:17</td>
<td>12:59</td>
<td>13:01</td>
<td>13:05</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>站界B8</td>
<td>站线13:11</td>
<td>站线11</td>
<td>站界87</td>
<td>12:56</td>
<td>13:00</td>
<td>13:15</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>站界B8</td>
<td>站线8:16</td>
<td>站线16</td>
<td>站界87</td>
<td>13:31</td>
<td>13:35</td>
<td>13:40</td>
<td></td>
</tr>
<tr>
<td>229</td>
<td>站界B10</td>
<td>站线16</td>
<td>站界89</td>
<td>站线8:16</td>
<td>13:36</td>
<td>13:51</td>
<td>13:55</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>站界B10</td>
<td>站线16</td>
<td>站界89</td>
<td>站线8:16</td>
<td>13:40</td>
<td>13:55</td>
<td>13:59</td>
<td></td>
</tr>
</tbody>
</table>
Problem Statement: Divide-Conquer

$n$ points are scattered inside a convex polygon $P$ (in 2D) with $m$ vertices. Does there exist a partition of $P$ into $n$ sub-regions satisfying the following:

- Each sub-region is a convex polygon
- Each sub-region contains one point
- All sub-regions have equal area
In the *Voronoi Diagram*, we satisfy the first two properties (each sub-region is convex and contains one point), but the sub-regions have different areas.
Our Result

Not only such an equitable partition always exists, but also we can find it exactly in running time $O(Nn \log N)$, where $N = m + n$. 

![Diagram of equitable partition](image)
Application VIII: Wireless Tower - Resource Allocation
## Preliminary Test Result—Effectiveness

### Based on Valid Commercial Network Model Optimization Effect Test Verification

<table>
<thead>
<tr>
<th></th>
<th>小区数</th>
<th>时段</th>
<th>区域平均负载</th>
<th>区域平均吞吐率 (Mb/S)</th>
<th>高负载小区负载</th>
<th>高负载小区吞吐率 (Mb/S)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>优化前</strong></td>
<td>27</td>
<td>中午及晚共6小时</td>
<td>31%</td>
<td>5.3</td>
<td>68%</td>
<td>2.3</td>
</tr>
<tr>
<td><strong>优化后</strong></td>
<td></td>
<td></td>
<td>30%</td>
<td>6.12(提升15%)</td>
<td>66%</td>
<td>2.8(提升22%)</td>
</tr>
<tr>
<td><strong>优化前</strong></td>
<td></td>
<td>晚7时话务高峰</td>
<td>37%</td>
<td>3.9</td>
<td>77%</td>
<td>1.6</td>
</tr>
<tr>
<td><strong>优化后</strong></td>
<td></td>
<td></td>
<td>33%</td>
<td>5.2(提升33%)</td>
<td>68%</td>
<td>2.1(提升32%)</td>
</tr>
</tbody>
</table>

### Impact Analysis

#### Optimized Before
- **Load**: 27
- **Peak Time**: 7 PM
- **Average Load**: 37%
- **Average Throughput**: 3.9 Mb/S
- **High Load Area Load**: 77%
- **High Load Area Throughput**: 1.6 Mb/S

#### Optimized After
- **Load**: 30%
- **Peak Time**: 7 PM
- **Average Load**: 33%
- **Average Throughput**: 5.2 Mb/S (33% Improvement)
- **High Load Area Load**: 68%
- **High Load Area Throughput**: 2.1 Mb/S (32% Improvement)
Application VV: Street View Application
Map-Making
Overall Takeaways

Second-Order Derivative information matters and better to integrate FOM and SOM on nonlinear optimization!

It is possible to make online decisions for quantitative decision models with performance guarantees close to that of the offline decision making with complete information.

Mixed Integer LP solvers benefit real economy.

Decomposition (Divide and Conquer) helps solving large-scale decision problems.

• THANK YOU