組合せ最適化

厳密解法とヒューリスティクスによる
複雑な意思決定の最適化

M1 名嘉山結月

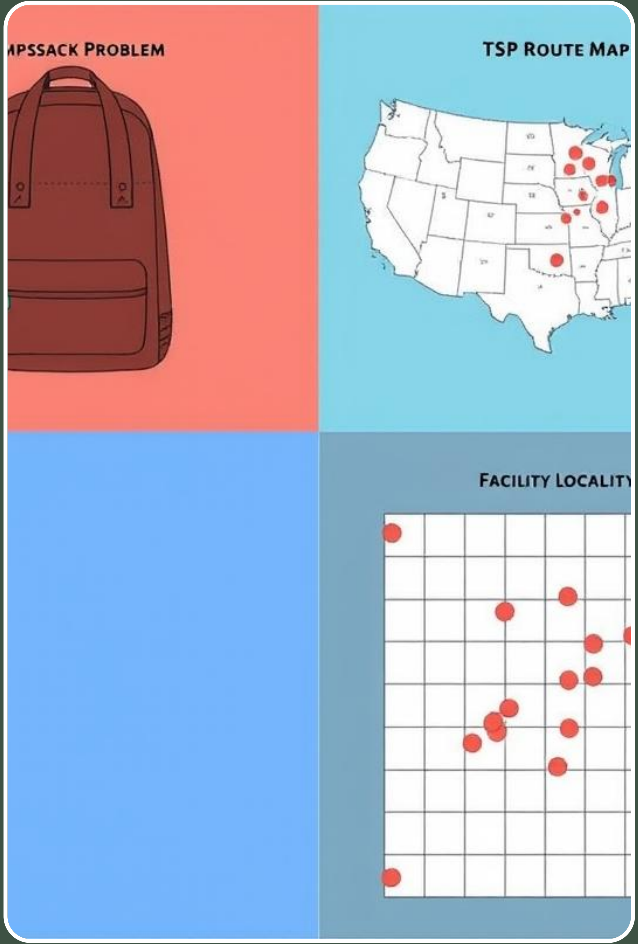
組み合わせ最適化とは「変数が離散的な値をとる最適化問題」

- **目的**：有限な選択肢から、特定の制約を満たしつつ最も良い組み合わせを選ぶ
- **特徴**：連続値ではなく、**離散値** (0か1、整数など) を決定変数とする
- **具体例**：配送ルート計画 (VRP) 、施設配置、ダイヤ編成

最適化問題の基本形：

$$\min f(x) \quad \text{subject to} \quad x \in X, x_i \in \{0,1\}$$

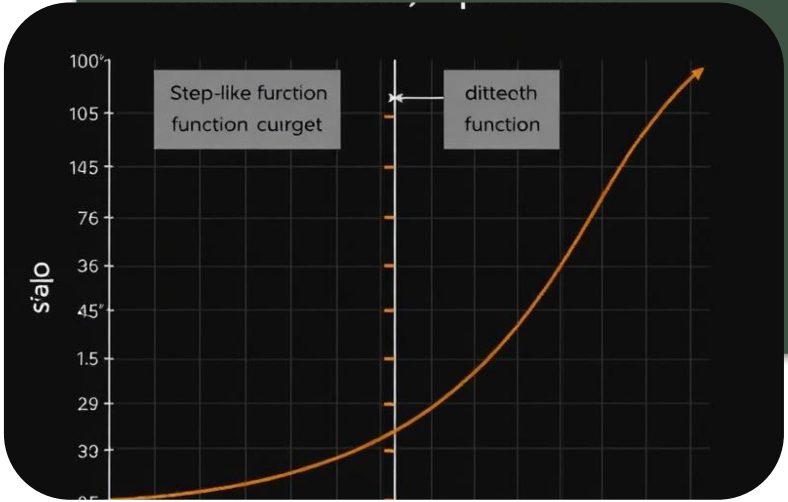
(x_i は「選ぶ/選ばない」などの離散的な意思決定)



代表的な組合せ最適化問題

- ナップサック問題：容量制限内で価値最大化（$$\sum w_i x_i \le W, \max \sum v_i x_i$$）
- 巡回セールスマン問題（TSP）：全都市を1回ずつ巡り最短経路を求める（$$\min \sum c_{ij} x_{ij}$$）
- 施設配置問題：顧客への距離最小化で施設数を固定（p-medianモデル）

整数計画と0-1計画の基礎



整数計画問題（IP）は、変数が**整数値のみを取る最適化問題**。

0-1計画問題は、変数が0または1しか取れない特殊なIPで、選択・非選択を表す。

「局所最適解」が「大域的最適解」とは限らない

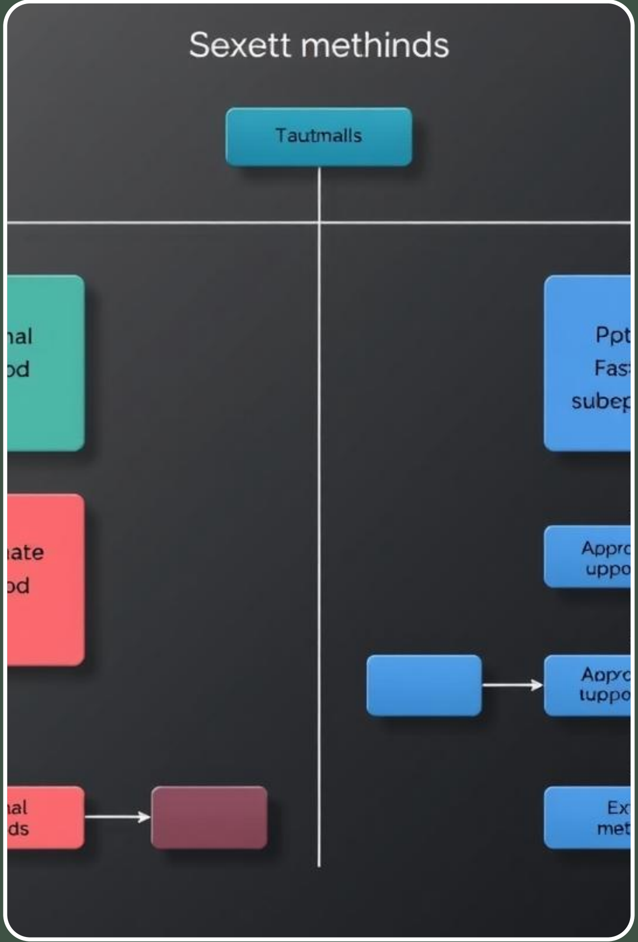
離散的な決定が必要な場面で活用
例) 現実の設備配置やスケジューリング等

組合せ爆発

- 直面する壁：問題サイズが大きくなると、候補数が指数関数的に増大する
- NP困難性：すべての候補を計算機で調べる（全列挙）ことは実質不可能
- 戦略の必要性：「賢く探索する」アルゴリズムが不可欠
- 巡回セールスマン問題（TSP）の候補数：O(n!) (都市数 n = 20 で約243京通り（天文学的数字）に到達)

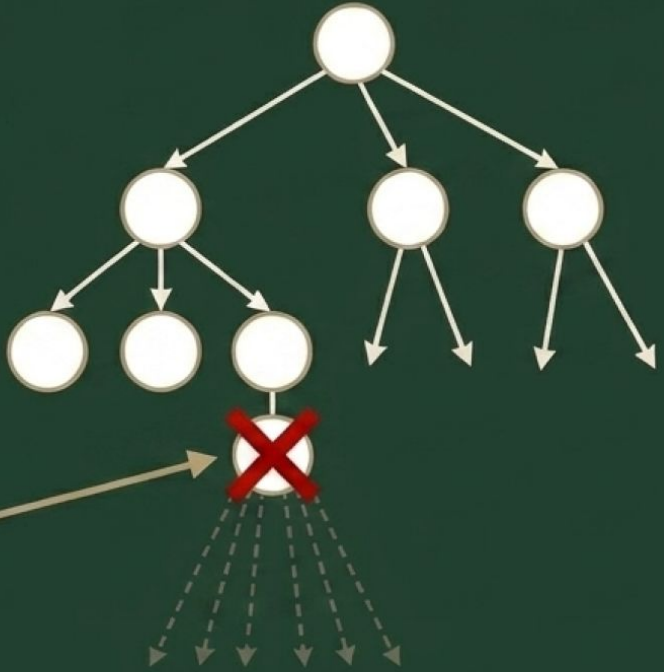
厳密解法と近似解法

- 厳密解法:
  数学的に**最適解を保証** する手法。
  計算時間がかかる場合がある
  小規模問題に適用可能
- 近似解法:
  十分良い解を**短時間で得る** 手法。
  最適保証はできない。
  大規模問題に適用



厳密解法①：分枝限定法（Branch and Bound）

- **方針**：探索空間を分割（分枝）し、不要な部分を理論的に切り捨てる（限定）
- **仕組み**：緩和問題から得た「下界」と、暫定解の「上界」を比較して枝刈り
- **メリット**：全列挙を避けつつ、最適解を完全に保証できる



枝刈り（Pruning）の条件：

$$Z_{\text{lower\_bound}} \geq Z_{\text{upper\_bound}} \Rightarrow \text{探索不要}$$

（部分問題の理論上の最小値が、すでに見つけた暫定解より悪ければ、その先は探さない）

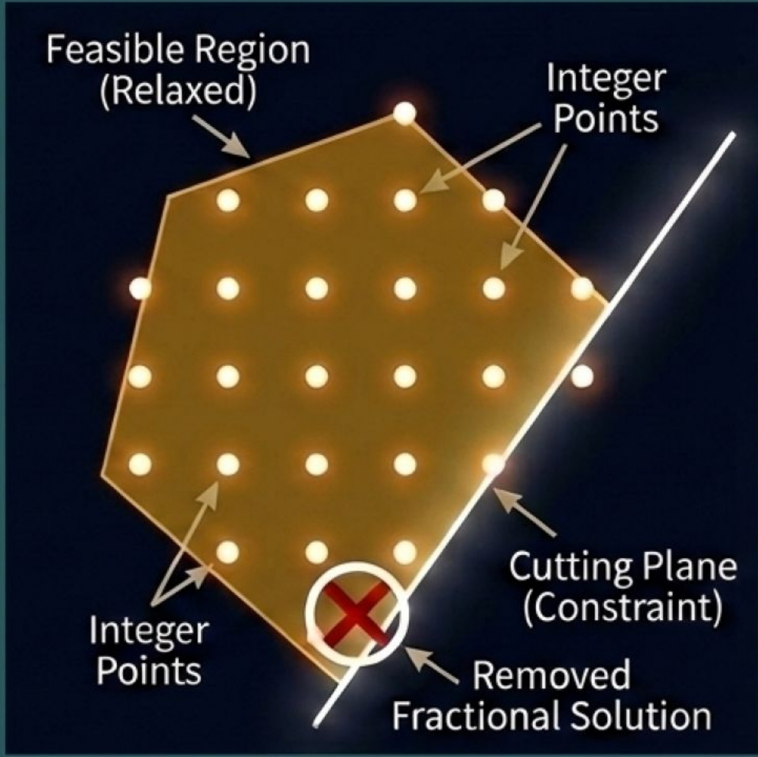
厳密解法②：切除平面法（Cutting Plane Method）

- 方針：整数条件を一時的に無視して解き、後から不要な小数を排除する
- 仕組み：整数解は残しつつ、小数解だけを切り落とす「制約（カット）」を追加
- メリット：線形計画ソルバーの高速な計算力を活用できる

代表的なカット（TSPの部分巡回除去制約）：

$$\sum_{i\in S}\sum_{j\in S} x_{ij} \leq |S| - 1 \quad \forall S \subset V, |S| \geq 2$$

（意味をなさない「小さなループ」を排除する数式を後から追加する）



厳密解法の意義と限界

解空間を体系的に探索し、理論的に「最適解」であることを証明可能。

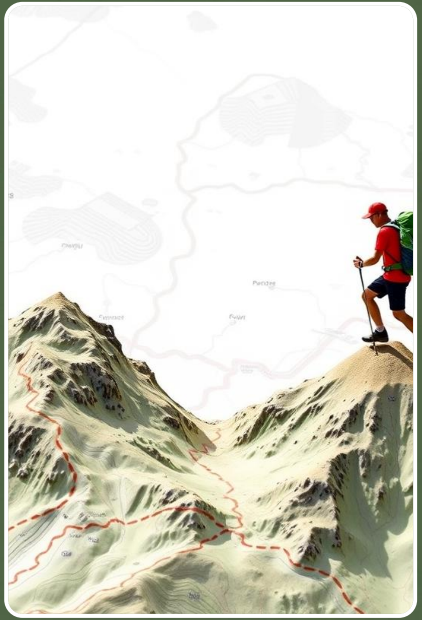
しかし、多くの組合せ最適化問題はNP困難であり、変数が増えると計算時間が現実的な範囲を超えてしまう。

そのため、大規模な現実問題では、最適解に近い「近似解」を迅速に見つける近似解法が不可欠。

ヒューリスティクス：最適性よりも実用的な速度を

方針：最適解の保証を捨て、「十分に良い解」を高速に探す

- 近傍探索 (Local Search) : 初期解から少しずつ解を変化させ、改善を繰り返す
- 初期解の設定：単純構成法と構成ヒューリスティクス法



近傍探索における局所最適解の定義：

$$f(x^*) \leq f(x)\ \forall x \in N(x^*)$$

(近傍 $N(x^*)$ の中のどの解よりも、現在の解 x^* の方が優れている状態)

近傍探索が陥る「局所最適解の罠」

- 課題：目の前の改善だけを追いかけると、本物の山頂に辿り着けない
- 局所最適解：その周辺エリアの中では一番良いが、全体での最高峰ではない解
- 限界：単純な近傍探索は、ここで探索がストップしてしまう

大域的最適解（Global Optimum）との関係：

$$f(x_{\text{global}}) \leq f(x_{\text{local}})$$

（局所最適解は、必ずしも真の最適解とは一致しない）

罠から脱出する「メタヒューリスティクス」

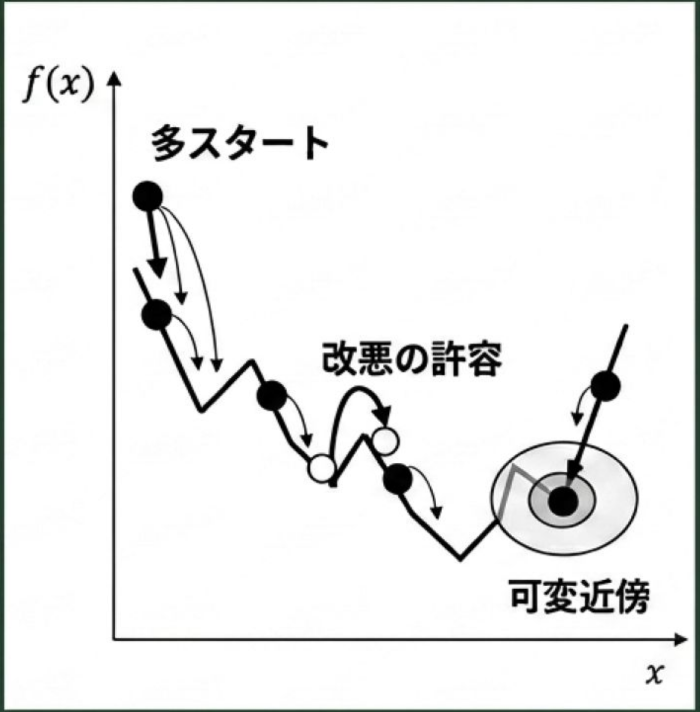
メタヒューリスティクス：局所探索を制御し、罠を抜け出すための上位戦略

- 手法1：多スタート法 → 初期解を何度も変えて登り直す
- 手法2：改悪の許容（焼きなまし法など）→ 確率的に一時的な「下り坂」を許す

$$P(\text{accept}) = \exp\left(\frac{f(x_{\text{current}}) - f(x_{\text{new}})}{T}\right)$$

（温度 T が高い初期は、改悪解でも受け入れて広く探索する）

- 手法3：可変近傍探索 → 探す範囲（近傍）の大きさを動的に変える



解法戦略の使い分け（マトリクス）

Trade-off: Optimality ⇔ Computational Time

	厳密解法 (Exact Methods)	ヒューリスティクス (Heuristics)
最適性の保証	完全保証（真の最適解）	保証なし（局所最適の可能性）
計算時間	膨大（問題サイズにより指数的）	高速（実用的な時間で完了）
適用シーン	小～中規模、インフラ設計、ベンチマーク	大規模、日々の運用、リアルタイム配車
代表的手法	分枝限定法、切除平面法	局所探索、メタヒューリスティクス

What is Combinatorial Optimization?

- **Problems requiring the selection of specific combinations from a finite set.**
- **Examples include Shortest Path, Traveling Salesman Problem (TSP), and Delivery Routing.**

自動運転オンデマンド交通の最適サービス設計

高尾駅周辺を対象に、自動運転オンデマンド交通の「対象領域・拠点配置・車両台数」の最適化問題を定義する。

CONTEXT & SCOPE

【目的】

・端末交通としての自動運転オンデマンド交通の最適なサービス設計の分析

【対象領域】

・高尾駅周辺3km圏内

3つの主要課題

【対象ゾーン】どのエリアをサービス範囲に含めるか？

×

【拠点配置】どこに車両拠点を配置するか？

×

【フリート規模】どの程度の車両台数で成立するか？

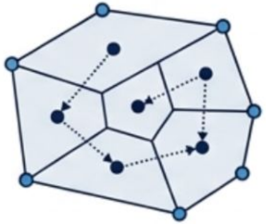
データ構造とネットワークモデル

現実の需要と空間を、ゾーン（需要）と拠点（供給）のネットワークとしてモデル化する。

東京PTデータ
（パーソントリップデータ）

500mメッシュ需要データ

ゾーン代表点（空間重心）



（供給ノード: Supply）
拠点 j

[移動距離・時間コスト]

（需要ノード: Demand）
ゾーン i

・拠点（Hub）：車両の待機・発着地点
・ゾーン（Zone）：発生する移動需要の集約点
・コスト定義：拠点とゾーン間の移動距離・時間をネットワーク近似により算出

定式化（1）変数定義と目的関数

総サービス時間を最小化するため、拠点選択とゾーン割当の0-1整数計画問題として定式化する。

[決定変数 (Decision Variables)]

$x_j \in \{0,1\}$ ：拠点 j を選択するか

$y_{ij} \in \{0,1\}$ ：ゾーン i を拠点 j が担当するか

[目的関数 (Objective Function)]

総サービス時間の最小化：

$$\min \sum_i \sum_j (demand_i \times service\_time_{ij} \times y_{ij})$$

※ $service\_time_{ij}$ = 往復時間 + バッファ + 駅接続コスト

[システム制約 (System Constraint)]

(1) カバー率制約：全体の需要に対して一定割合（α）以上をカバーする

$$\sum_i covered\_demand_i \geq \alpha \times total\_demand$$

定式化（2）物理・運用上の制約条件

各ゾーンへの単一拠点割当に加え、近接制約と容量制約によって現実的な配車を担保する。

配置・割当制約（Location & Assignment）

(3) 拠点数制約：選択する全拠点数は P 個に固定する。

$$\sum_j x_j = P$$

(4) 割当制約：各ゾーンは選ばれた1つの拠点にのみ割り当てる。（※論理的排他制約）

運用制約（Operational Bounds）

(5) 近接制約 [本研究の新規性]：ゾーン i は、距離が近い上位 k 個の拠点（集合 $N_k(i)$）にのみ割当可能とする。

$$y_{ij} = 0 \quad (\forall i, \forall j \notin N_k(i))$$

(6) 容量制約：各拠点のサービス時間合計は、保有する車両フリートの総容量を超えない。

$$\sum_i \mathrm{service\_time}_{ij} \times y_{ij} \leq \mathrm{fleet} \times \mathrm{capacity} \ (\forall j)$$

最適化アプローチ

SciPyを用いたMILPソルバーにより、全候補の組み合わせから厳密最適解を探索する。

[全候補拠点の抽出] → [MILP構築] (scipy.optimize.milp) → [制約評価] (カバー率・公平性・容量) → [目的関数最小解の特定]

[アプローチの特徴]

- 問題クラス：0-1混合整数線形計画問題（MILP）
- 探索方針：全候補拠点の組み合わせを網羅的に評価し、大域的最適解を保証
- 実装環境：Python（SciPyモジュール）環境による最適化モデルの構築と実行

評価指標とパラメータ設定

ネットワーク距離から推定したサービス時間と、普及率を考慮した有効需要を用いて解を評価する。

1. ネットワーク距離推定 (Spatial Metrics)

- distance_km：拠点〜ゾーン間の空間距離
- travel_time_min：ネットワーク構造から走行時間を換算

2. サービスコスト定義 (Cost Metrics)

- service_time：往復時間 + バッファ + 駅接続コスト（総所要時間）

3. 需要モデル (Demand Model)

- effective_demand：潜在需要 × adoption_rate（普及率 20%と仮定）

最適解の結果とパフォーマンス

高尾駅を含む3拠点での最適運用において、容量制約が上限に達しシステム稼働率が最大化された。

[設定条件 (Input Parameters)]

- 拠点数 $P = 3$
- 近接上限 $k = 8$
- カバー率制約 $\alpha \geq 0.5$
- 公平性制約 $\beta \geq 0.3$

[最適配置構造 (Optimal Hub Topology)]

- Hub 1: 高尾駅 (メインハブ))
- Hub 2: Candidate_6
- Hub 3: Candidate_8

[主要KPI (Performance Dashboard)]

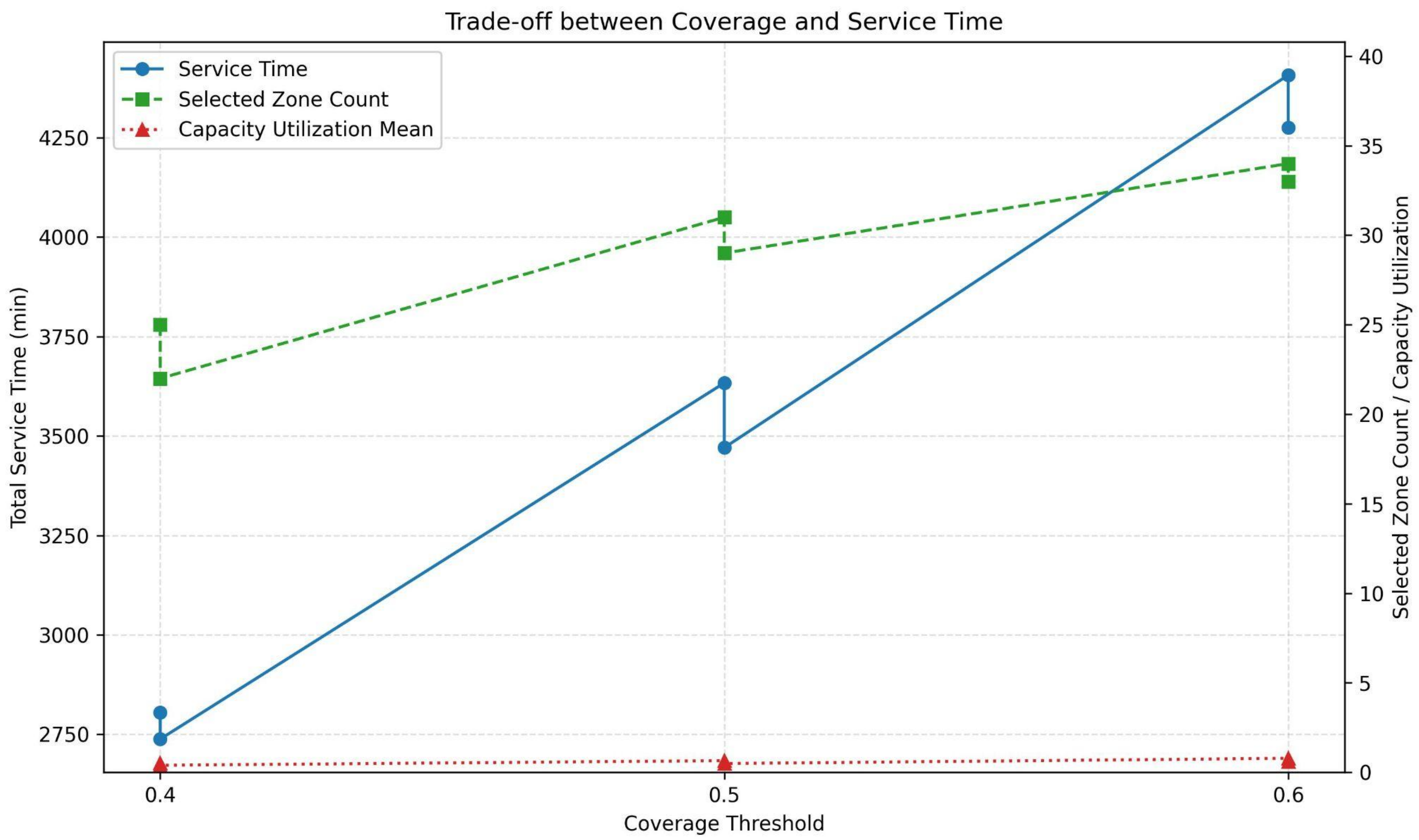
- カバー率（Coverage）：約 0.50（制約下限で均衡）
- 総サービス時間：約 3600 分

稼働率（Capacity Utilization）：約 1.0

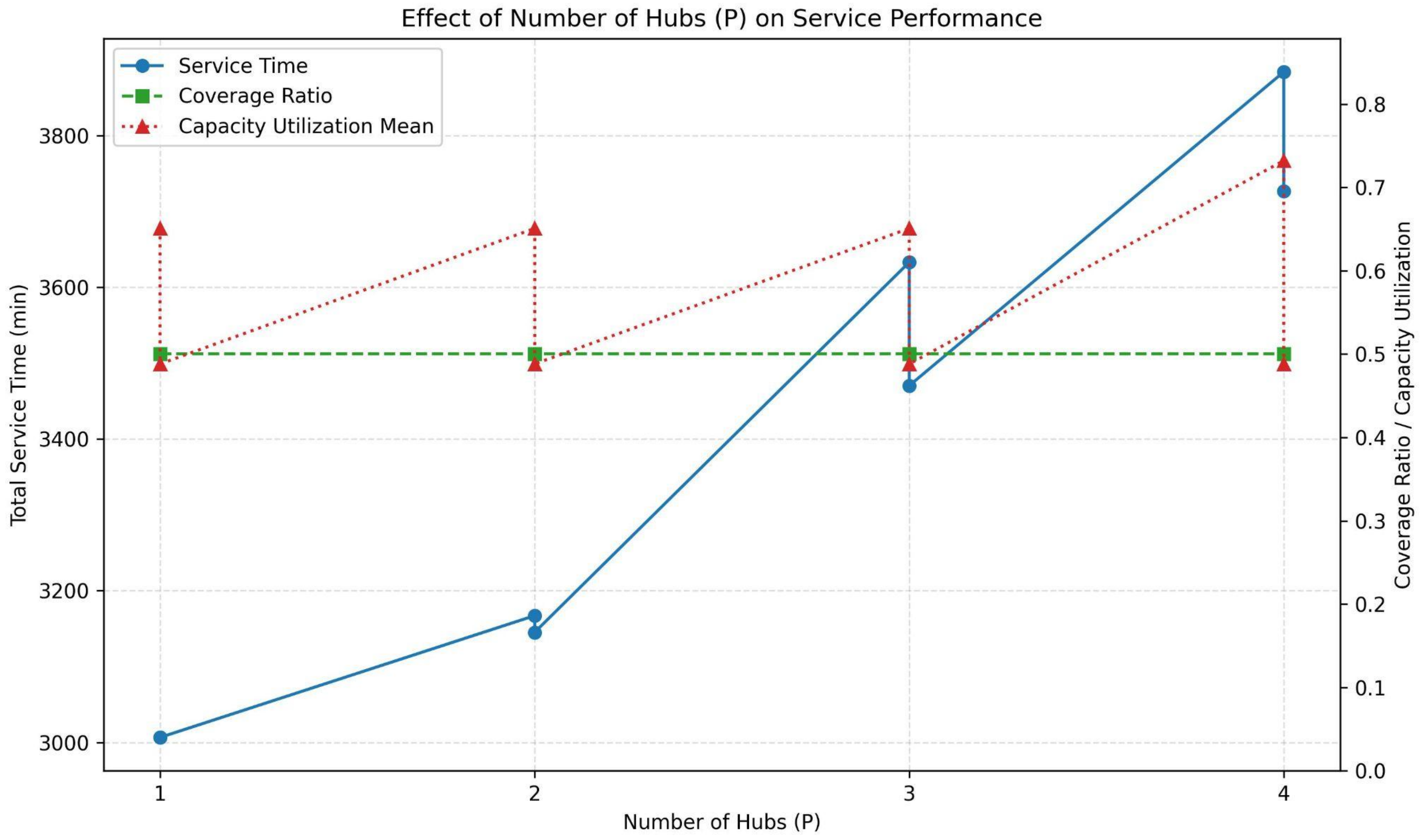
（容量制約が強くバインディング）

Summary of Key Cases

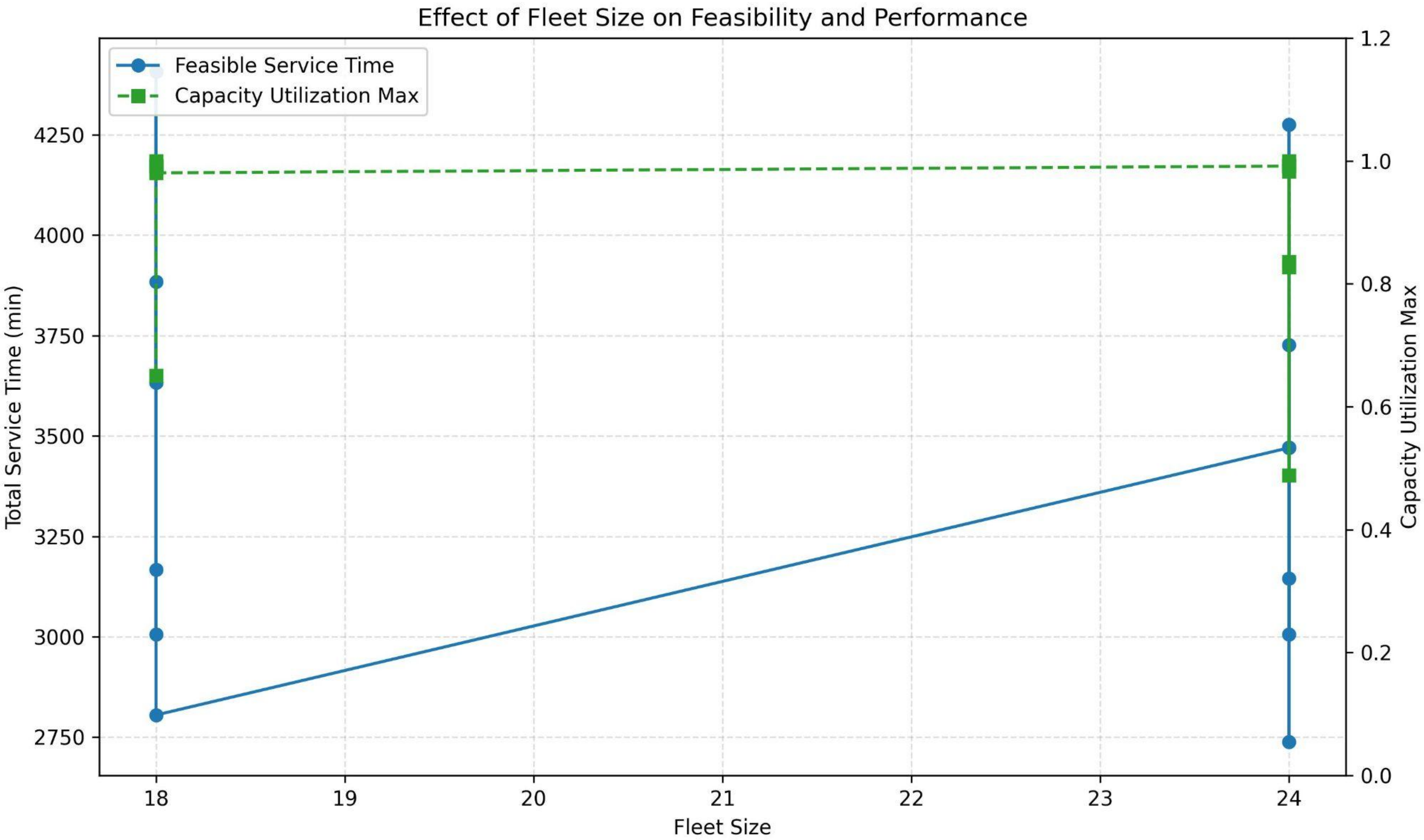
case_name	P	coverage_threshold	total_service_time_mi	coverage_ratio	capacity_utilization_mear
P3_k8	3	0.5	3633.2	0.5	0.65
cov40_k8	3	0.4	2805.24	0.4	0.52
cov50_k8	3	0.5	3633.2	0.5	0.65
cov60_k8	3	0.6	4407.11	0.6	0.78
beta00_k8	3	0.5	3633.2	0.5	0.65
beta05_k8	3	0.5	3633.2	0.5	0.65
beta10_k8	3	0.5	3633.2	0.5	0.65



Trade-off between Coverage and Service Time
Service Time
Selected Zone Count
Capacity Utilization Mean
Total Service Time (min)
4250
4000
3750
3500
3250
3000
2750
Selected Zone Count / Capacity Utilization
40
35
30
25
20
15
10
5
0
0.4
0.5
0.6
Coverage Threshold



Effect of Number of Hubs (P) on Service Performance
Service Time
Coverage Ratio
Capacity Utilization Mean
Total Service Time (min)
Coverage Ratio / Capacity Utilization
Number of Hubs (P)
3000
3200
3400
3600
3800
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
1
2
3
4


Effect of Fleet Size on Feasibility and Performance
Feasible Service Time
Capacity Utilization Max
Total Service Time (min)
Capacity Utilization Max
Fleet Size
4250
4000
3750
3500
3250
3000
2750
1.2
1.0
0.8
0.6
0.4
0.2
0.0
18
19
20
21
22
23
24

考察：空間構造の違いと問題の支配的要因

サービス成立性は容量制約とカバー率のトレードオフに依存し、都市と地方で支配的な制約が異なる。

【最適化結果に対する考察】

- 拠点数を増やしても、総サービス時間の劇的な効率改善は見られない。
- フリート（車両台数）と容量制約が、システム全体の限界を決定づけている。

都市部（例: 高尾）

- 需要特性: 需要密度が高い
- 支配的制約: 「容量制約」がバインディングする
- 問題構造: いかに限られた車両リソースで溢れる需要を捌くか（Fleet Optimization)

地方部（例: 伊方）

- 需要特性: 需要が広域に分散
- 支配的制約: 「拠点配置（距離）」がバインディングする
- 問題構造: いかに遠方の需要を効率的にカバーする拠点網を敷くか (Topology Optimization)

結論：地域の空間構造に応じて、オンデマンド交通の最適化モデルは力点が根底から変化する。

今後の展望

①端末交通としてのNW表現
→実際の道路NWの反映、経路計算や混雑制約の導入など

②拠点割当の厳密化
→各ゾーンを最寄り拠点に限定（k=1）

③拠点配置の現実化
→駅・学校・公共施設を候補とした現実の立地条件の反映

④地方（伊方）への展開
→需要密度が分散、空間構造の制約が支配的な土地への応用

LLM活用

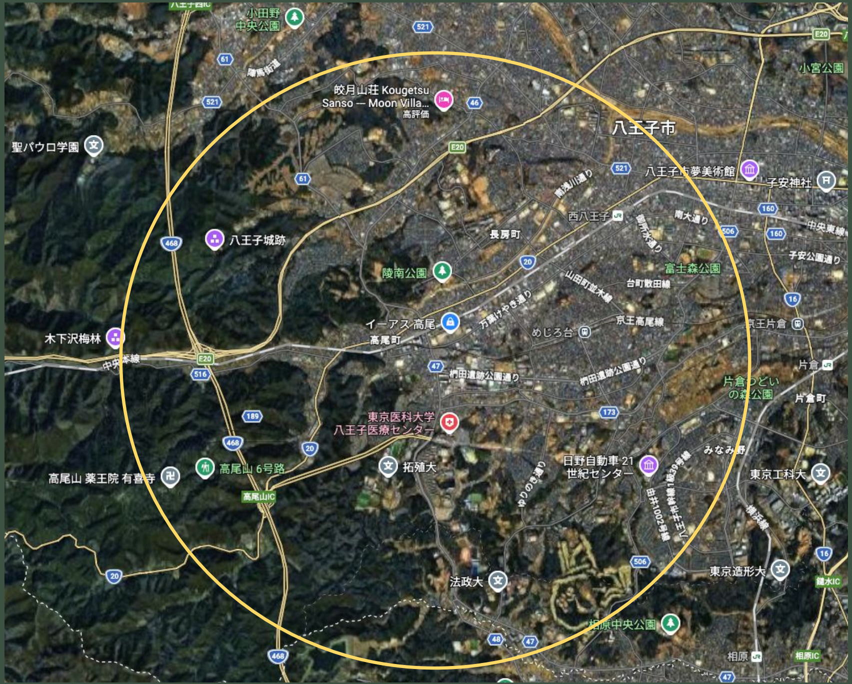
スライド生成を複数のLLMで行い、比較した。

①教科書説明→AI Slide(Google slide プラグイン)
②演習課題→NotebookLM

後者の方が出力の質は高いが、編集の柔軟性が低い
最適なツール・活用法を模索していきたい

（本資料は後ほど体裁を整えて再アップロードします）

対象地域とデータ構造



対象：高尾駅周辺3km圏内

データ：2019年度PT調査データと500mメッシュ需要データの統合

ネットワーク：ゾーン（需要）と拠点（供給）の双方向接続