地域交通計画立案ツールを改良した
群馬県内の未利用資源運搬等に関する
基礎的研究

受託研究報告書

独立行政法人自動車技術総合機構
交通安全環境研究所

令和 8 年 2 月

目次

図目次

表目次

1　序論

1.1　仕様の確認

本研究は、以下の仕様に基づいて実施された。

- **名称**：地域交通計画立案ツールを改良した群馬県内の未利用資源運搬等に関する基礎的研究
- **目的**：交通安全環境研究所で開発中の「地域交通計画立案ツール」を改良し、群馬県内の未利用資源運搬時の最短経路と運搬費用を推計する
- **期間**：契約締結日から令和 8 年 2 月 27 日まで
- **内容**：
  1. 交通安全環境研究所で開発中の「地域交通計画立案ツール」（以下、ツールと呼ぶ）を改良する
  2. ツールの変数として、数種類の未利用資源を使用する
  3. ツールの変数として、数種類のモビリティ（軽トラ等の貨物車等）を使用する
  4. 単一の収集場所から単一の集積場所まで運搬する際の、最短経路と運搬費用を推計する
- **提出形式**：印刷物 3 部、デジタルデータ（数値データ、編集可能書類データ、PDF データ）
- **提出期限**：2026 年 2 月 27 日（金）

1.2　研究目的

未利用資源の効果的な収集運搬は、地域の循環型社会形成において重要な課題である。本研究では、群馬県内における未利用資源の運搬最適化を目的として、以下の目標を設定した。

1. 未利用資源と運搬車両の適合性を体系的に整理する
2. 最短経路探索アルゴリズムを実装し、運搬距離を最小化する
3. 運搬費用を詳細に推計し、経済性を評価する
4. 実用的な Web アプリケーションとして実装し、利用可能な形態で提供する

これらの目標達成により、未利用資源の効率的な収集運搬計画の策定を支援し、地域におけるバイオマス利活用の促進に貢献することを目指す。

2　システムの全体構成

本システムは、未利用資源の収集運搬計画を最適化するための Web アプリケーションである。Streamlit フレームワークを基盤とし、以下の主要コンポーネントから構成される（図 1）。

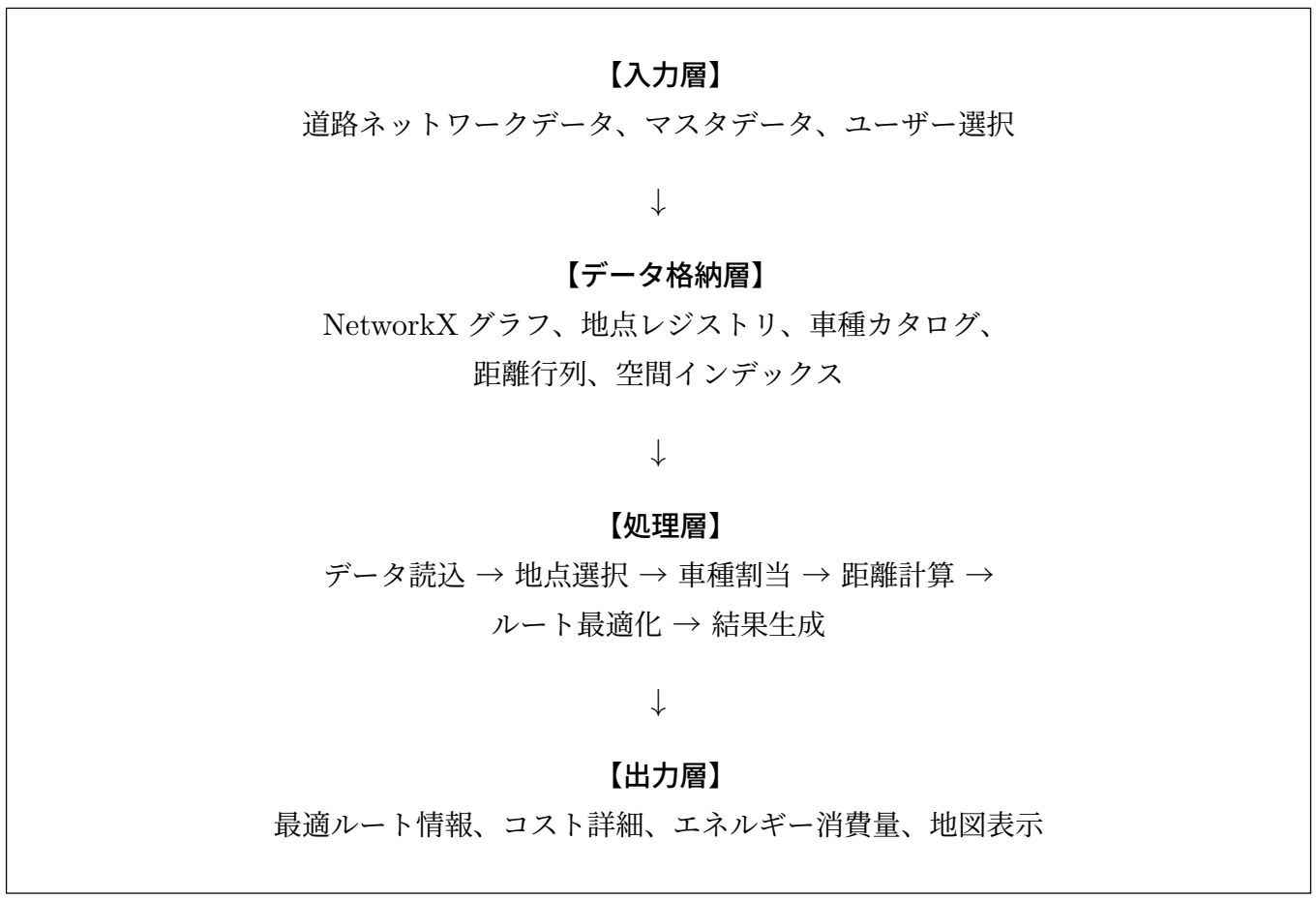

図 1: システムの全体構成

2.1 システム構成要素

システムは 4 層構造で構成されており、図 1 に示す各層の要素を以下に説明する。

2.1.1 入力層

入力層では、システムが最適化計算を実行するために必要な 3 種類のデータを受け取る。

道路ネットワークデータ

OpenStreetMap（OSM）またはカスタム JSON 形式で提供される道路ネットワーク情報である。ノード（交差点・地点）とエッジ（道路区間）の接続関係、各道路区間の距離情報を含む。

マスタデータ

システムの基礎となる静的データであり、以下の 3 つのファイルから構成される：

- `vehicles.json`：14 種類の運搬車両の諸元（積載容積、燃費、購入費用、固定費、変動費）
- `resources.json`：11 種類の未利用資源の特性（比重、取扱い注意事項）
- `compatibility.json`：車両と資源の適合性マトリックス（154 組み合わせ）

ユーザー選択

Streamlit の対話的インターフェースを通じてユーザーが入力する情報である：

- デポ（車庫）の位置：車両の出発地点・帰着地点の座標
- 回収地点：資源を回収する地点の座標、資源種別、回収量

- 集積場所：資源の最終目的地の座標
- 使用車両の選択：14 種類から選択

2.1.2 データ格納層

データ格納層では、入力層から受け取ったデータを最適化処理に適した形式に変換・保持する。

NetworkX グラフ

道路ネットワークをグラフ理論のデータ構造に変換したものである。ノード（頂点）とエッジ（辺）で構成され、最短経路探索アルゴリズムの基盤となる。

地点レジストリ

デポ、回収地点、集積場所を統合的に管理するデータ構造である。各地点の ID、座標、タイプ（デポ/回収地点/集積場所）、資源情報を保持する。

車種カタログ

選択された車両の詳細情報を格納する。積載容積、燃費、コスト情報、資源適合性などを含む。

距離行列

すべての地点間の最短距離を N × N 行列として事前計算・キャッシュしたものである。最適化計算の高速化に寄与する。

空間インデックス

地図上のクリック位置から最寄りの道路ネットワークノードを高速に検索するためのデータ構造である。KD-tree などの空間分割アルゴリズムを使用する。

2.1.3 処理層

処理層では、データ格納層のデータを用いて最適化計算を実行する。図 1 に示す 6 つのステップを順次処理する。

データ読込

入力層からのデータをパースし、データ格納層の各データ構造を初期化する。

地点選択

ユーザーの地図クリック操作を受け取り、空間インデックスを用いて最寄りの道路ネットワークノードを特定する。選択された地点を地点レジストリに登録する。

車種割当

資源種別に基づいて、車種カタログから適合する車両を抽出する。複数の車両が適合する場合、コスト評価関数により最適な車両を選択する。

距離計算

NetworkX グラフを用いて、選択された全地点間の最短経路を Dijkstra 法または A*アルゴリズムで計算する。計算結果を距離行列に格納する。

ルート最適化

車両経路問題（VRP）を解決し、最適な訪問順序を決定する。容量制約と資源適合性を考慮したヒューリスティックアルゴリズム（最近傍法、2-opt 法）を使用する。

結果生成

最適化されたルートに基づき、総走行距離、運搬費用、エネルギー消費量を計算する。地図表示用のデータ形式に変換する。

2.1.4　出力層

出力層では、最適化結果をユーザーに分かりやすい形式で提供する。

最適ルート情報

車両ごとの訪問順序、各区間の距離、総走行距離、所要時間を表示する。

コスト詳細

変動費 12 項目と固定費 13 項目の内訳、項目別金額、総コストを詳細に表示する。

エネルギー消費量

燃料消費量、CO_2 排出量を車両の燃費データから算出し表示する。

地図表示

Folium ライブラリを用いて、最適化されたルートをインタラクティブな地図上に可視化する。経路は色分けされ、クリックすると詳細情報がポップアップ表示される。

2.2　データフロー

システムのデータフローは以下の通りである：

1. ユーザーが道路ネットワークを選択し、システムに読み込む
2. 地図上で拠点、回収地点、集積場所を設定する
3. 使用する車両タイプと運搬する資源タイプを選択する
4. システムが車両・資源適合性を自動的に検証する
5. 最適化エンジンが最短経路と最小コストのルートを計算する
6. 結果が地図上に可視化され、詳細レポートが生成される
7. ユーザーは結果を CSV、JSON、HTML 形式でエクスポート可能

2.3　技術スタック

本システムは以下の技術により構築されている：

- **フロントエンド**：Streamlit（Python Web フレームワーク）
- **グラフ処理**：NetworkX（道路ネットワークのグラフ表現と経路探索）
- **地図可視化**：Folium（インタラクティブマップ生成）
- **データ処理**：Pandas、NumPy（データ管理と数値計算）
- **最適化**：OR-Tools（制約付き最適化問題の解決）
- **データ形式**：JSON、CSV（データの入出力）

2.4　システムの特徴

- **使いやすさ**：専門知識不要の直感的なインターフェース
- **柔軟性**：様々な資源・車両の組み合わせに対応
- **拡張性**：新しい資源や車両の追加が容易
- **透明性**：計算過程とコスト内訳を詳細に表示
- **実用性**：実際のコストデータに基づく現実的な推計

3　システムの詳細

本章では、システムを構成する主要要素について詳細に説明する。

3.1　資源

本システムでは、群馬県内で発生する 11 種類の未利用資源を対象としている。各資源の特性と収集における注意点を以下に示す。

3.1.1　建設廃材

建設・解体工事から発生する木材、金属、コンクリートがらなどを含む。比重が大きく重量物が多いため、積載重量に注意が必要である。混合廃棄物の場合は分別処理が必要となる。

3.1.2　農業残渣

稲わら、もみ殻、麦わらなど農作物収穫後の残留物である。軽量でかさばる特性を持ち、乾燥状態と湿潤状態で性状が大きく異なる。運搬時は飛散防止対策が重要である。

3.1.3　林業残材

間伐材、枝葉、樹皮、製材端材などを指す。長尺物が多く、水分含有率により重量が大きく変動する。積載時は荷崩れ防止のための固定が必要である。

3.1.4　食品廃棄物

生ごみ、食品残渣、売れ残り食品などを含む。水分含有率が高く腐敗しやすいため、迅速な運搬が求められる。汁漏れ防止のため、密閉容器の使用または専用車両が必要である。

3.1.5　廃プラスチック

容器包装、フィルム、硬質プラスチックなどを含む。軽量でかさばり、風で飛散しやすいため、シート養生または密閉車両での運搬が推奨される。

3.1.6　金属スクラップ

鉄くず、非鉄金属、廃家電から回収した金属などを含む。比重が極めて大きいため、積載重量に特に注意が必要である。クレーン付き車両の使用が効率的な場合が多い。

3.1.7　古紙・段ボール

新聞紙、雑誌、段ボール箱などを含む。水濡れ厳禁であり、圧縮により積載効率が向上する。雨天時はシート養生または箱型車両の使用が必須である。

3.1.8　剪定枝・草

庭木剪定枝、草刈り残渣、落ち葉などを含む。かさばるため、破砕処理により積載効率が大幅に向上する。パッカー車による圧縮収集も有効である。

3.1.9　家畜糞尿

牛糞、豚糞、鶏糞などを含む。半固形から液状まで性状が多様であり、臭気対策が必須である。液状のものはバキューム車、半固形のものはダンプまたは密閉コンテナで運搬する。

3.1.10　下水汚泥

浄化槽汚泥、し尿などを含む。液状から泥状であり、専用車両（バキューム車）での運搬が法令により義務付けられている。

3.1.11　廃食用油

使用済み天ぷら油、業務用廃油などを含む。液体であるため、漏洩防止のため密閉容器（缶・ペットボトル）での回収を前提とする。容器ごと運搬することで、平ボディやウイング車でも運搬可能である。

これらの資源は、その物理的・化学的特性に応じて適切な運搬車両を選択する必要がある。

3.2　運搬車両

本システムでは、未利用資源の収集運搬に使用される 14 種類の車両を対象としている。各車両の特徴と主な用途を以下に分類して示す。

3.2.1　小型車両（2-3t クラス）

軽トラック

最大積載量 350kg 程度、車両総重量 2t 未満の小型貨物車である。狭い道路での機動性に優れ、普通免許で運転可能である。少量の資源回収や農村部での巡回収集に適している。

2t トラック（平ボディ）

最大積載量 2t、荷台がフラットな汎用トラックである。荷台寸法は約 3.1m × 1.6m × 0.4m（長さ×幅×高さ）であり、様々な形状の資源を積載可能で、側面からの積み下ろしが容易である。小規模事業所からの資源

回収や市街地での収集業務に使用される。

2t ダンプ

油圧装置により荷台を傾斜させて荷下ろしする車両である。土砂、廃棄物など流動性のある資源に適しており、建設廃材、剪定枝、家畜糞尿などの運搬に使用される。

3.2.2　中型車両（4t クラス）

4t 平ボディ

最大積載量 4t 程度、車両総重量 8t 未満の中型トラックである。荷台寸法は約 5.0-6.2m × 2.1-2.2m × 0.6m であり、運搬効率と機動性のバランスが良く、最も汎用性が高い。中規模事業所からの資源回収や中距離輸送に適している。

4t ダンプ

2t ダンプよりも大容量で、建設現場などでの大量運搬に適する。建設廃材、解体材、土砂類の大量運搬に使用される。

4t ユニック車（クレーン付きトラック）

小型クレーンを搭載し、重量物の積み下ろしが可能な車両である。クレーン搭載により荷台容積は減少するが、重量物の単独作業が可能である。金属スクラップや建設廃材など重量物の運搬に使用される。

4t ウイング車

側面が翼のように開閉する箱型トラックである。側面全開により積み下ろし作業性が極めて良好で、雨風から荷物を保護できる。古紙、廃プラスチック、食品廃棄物など多様な資源の運搬に使用される。

4t パッカー車（塵芥車）

圧縮機構を備えた廃棄物収集専用車両である。回転板やプレスにより廃棄物を圧縮し、積載効率が高い。食品廃棄物、廃プラスチック、古紙、剪定枝などの収集に使用される。

4t アームロール車

コンテナを脱着できる装置を備えた車両である。複数のコンテナを準備することで車両の稼働率を向上でき、様々な資源に対応可能である。建設廃材、産業廃棄物、多種類の資源の効率的収集に使用される。

3.2.3　大型車両（10t クラス）

10t 平ボディ

最大積載量 10t 程度、車両総重量 25t 未満の大型トラックである。荷台寸法は約 9m × 2.4m × 0.5m であり、大量輸送に適し、長距離輸送でも効率的である。大規模施設からの資源回収、広域収集、長距離輸送に使用される。

10t ダンプ

建設廃材や土砂の大量輸送に最適である。解体現場や大規模工事現場からの廃材運搬に使用される。

10t ウイング車

大容量と作業性を両立し、最も効率的な大型運搬車両である。古紙、廃プラスチック、パレット化された資源の大量輸送に使用される。

大型パッカー車

圧縮機構を備えた大型廃棄物収集車である。大量の廃棄物を効率的に圧縮収集でき、広域での一般廃棄物収集や大規模イベントでの廃棄物回収に使用される。

3.2.4　特殊用途車両

バキューム車（4t クラス）

真空ポンプにより液体・泥状物を吸引収集する専用車両である。液状・泥状の資源専用で、密閉タンク構造を持つ。家畜糞尿、下水汚泥、食品廃液の収集運搬に使用される。

これらの車両の諸元（積載容積、燃費、購入費用など）は、実際のメーカーカタログおよび業界データに基づいて詳細に設定されている（付録参照）。

3.3　資源・運搬車両適合性

車両と資源の適合性は、物理的適合性、性状適合性、作業効率、法令遵守、安全性の観点から総合的に判定されている。

3.3.1　適合判定基準

適合（○）の判定基準

- **物理的適合性**：資源の形状・サイズが荷台に収まる
- **性状適合性**：資源の性状（固体/液体/粉体）が車両構造に適している
- **作業効率**：積み下ろし作業が現実的な時間・労力で実施可能
- **法令遵守**：廃棄物処理法等の関連法規に抵触しない
- **安全性**：運搬中の落下・飛散・漏洩リスクが許容範囲内

条件付き適合（△）の判定基準

追加の設備または対策を講じることにより運搬が可能となる組み合わせである。例えば、密閉容器の使用、防風シートの設置、防臭対策の実施などが条件として設定される。

不適合（×）の判定基準

- **構造的不適合**：車両の構造上、当該資源の運搬が不可能
- **性状不適合**：資源の性状が車両に適さない（例：液体をダンプで運搬）
- **衛生上の問題**：食品廃棄物を開放型荷台で運搬など
- **経済性の欠如**：著しく非効率（例：農業残渣をユニック車で運搬）

- **環境汚染リスク**：運搬中の漏洩により環境汚染の恐れ

3.3.2　主要な判定例

- **ダンプ×食品廃棄物**：水分が多く汁漏れのリスク、開放型荷台では衛生上不適切
- **パッカー車×金属スクラップ**：圧縮機構が金属により破損する恐れ、重量物には不向き
- **ユニック車×農業残渣**：軽量でかさばる資源にクレーンは不要、経済性に欠ける
- **バキューム車◯下水汚泥**：液状・泥状物専用車両として最適、密閉構造で臭気対策も万全
- **アームロール △ 家畜糞尿**：密閉コンテナ使用により臭気を抑制、コンテナ交換で効率的

この適合性マトリックス（14 種類の車両× 11 種類の資源、計 154 組み合わせ）は、システム内で自動的に参照され、ユーザーが選択した車両と資源の組み合わせが適切かどうかを判定する。不適合な組み合わせの場合は警告が表示され、条件付き適合の場合は必要な条件が提示される。これにより、実現不可能な運搬計画の立案を未然に防ぐことができる。

3.4　最適化アルゴリズム

本システムでは、未利用資源の収集運搬経路を最適化するために、車両経路問題（Vehicle Routing Problem, VRP）の解法を実装している。

3.4.1　問題の定式化

本システムが扱う問題は、以下のように定式化される：

- **入力**：
  - デポ（車両の出発地点・帰着地点）
  - 複数の回収地点（未利用資源の発生場所）
  - 集積場所（資源の最終目的地）
  - 道路ネットワーク（ノードとエッジのグラフ構造）
  - 車両の容量制約
  - 各回収地点での資源量
- **目的**：
  - 総走行距離の最小化
  - 運搬費用の最小化
  - 車両台数の最小化
- **制約**：
  - すべての回収地点を訪問すること
  - 車両の積載容量を超えないこと
  - すべての車両がデポから出発し、デポに帰着すること
  - 資源は最終的に集積場所に運搬されること

3.4.2　最短経路探索アルゴリズム

本システムでは、グラフ理論に基づく最短経路探索アルゴリズムを採用している。

Dijkstra 法

Dijkstra 法は、単一始点からすべてのノードへの最短経路を求める古典的なアルゴリズムである。計算量は O((V + E) log V)（V はノード数、E はエッジ数）であり、負の重みがない場合に正確な解を得られる。

本システムでは、NetworkX ライブラリの `dijkstra_path` 関数を使用して実装している。デポから各回収地点、回収地点間、回収地点から集積場所への最短経路を事前計算し、これを基に全体の経路を構築する。

A*アルゴリズム

A*アルゴリズムは、ヒューリスティック関数を用いて Dijkstra 法を拡張したアルゴリズムである。目標ノードまでの推定距離（ヒューリスティック値）を利用することで、探索の効率を向上させる。

本システムでは、ユークリッド距離をヒューリスティック関数として使用し、NetworkX の `astar_path` 関数により実装している。

3.4.3 車両経路問題の解法

最近傍法（Nearest Neighbor Heuristic）

初期解の構築に使用される貪欲法である：

1. デポから最も近い未訪問の回収地点を訪問
2. 現在地から最も近い未訪問の回収地点を訪問
3. 積載容量に達したら集積場所へ運搬
4. すべての回収地点を訪問するまで繰り返す

計算量は $O(n^2)$（n は回収地点数）であり、高速に解を得られるが、最適解を保証しない。

2-opt 法による改善

最近傍法で得られた初期解を改善するための局所探索法である：

1. 経路中の 2 つのエッジを選択
2. エッジを入れ替えた場合の経路長を計算
3. 改善があれば入れ替えを採用
4. 改善がなくなるまで繰り返す

この手法により、交差する経路を解消し、経路長を短縮できる。

容量制約の処理

各車両には積載容量の制約があり、これを考慮した経路構築が必要である。本システムでは、以下の方法で容量制約を処理している：

- 回収地点を訪問する際、現在の積載量に資源量を加算
- 積載量が車両容量を超える場合、その回収地点は訪問せず、次の候補を選択
- 積載量が容量に達したら、集積場所へ運搬し積載量をリセット
- すべての回収地点を訪問するまでこのプロセスを繰り返す

3.4.4 計算の効率化

大規模な問題に対しても現実的な時間で解を得るため、以下の効率化を実施している：

- **距離行列の事前計算**：すべてのノード間の最短距離を事前に計算し、キャッシュする
- **候補地点の絞り込み**：ヒューリスティックにより、明らかに非効率な経路を探索対象から除外
- **並列計算**：複数の車両の経路を独立に計算できる場合、並列処理を適用
- **早期終了条件**：一定時間経過後、または十分に良い解が得られた時点で探索を終了

これらのアルゴリズムにより、数十から数百の回収地点を持つ実用的な問題を、数秒から数分で解決できる。

3.5 コスト計算

本システムでは、未利用資源の運搬に関する総コストを詳細に推計する機能を実装している。コストは変動費と固定費に大別され、それぞれ複数の項目から構成される。

3.5.1 変動費（走行距離に比例するコスト）

変動費は走行距離に応じて変化する費用であり、距離単価（円/km）で表される。以下の 12 項目から構成される。

燃料費

計算式：燃料費 = 燃料単価 ÷ 燃費

燃料単価は、軽油 150 円/L、ガソリン 170 円/L（2025 年 10 月時点の全国平均）を使用している。燃費は車両タイプごとに設定されており、軽トラック 13-16 km/L、2t-4t トラック 5-11 km/L、10t トラック 3.5-5 km/L の範囲である。

例：4t 平ボディ（燃費 6 km/L）の場合、150 円 ÷ 6 km/L = 25 円/km

運転手人件費

計算式：人件費 = 時給 ÷ 平均速度

時給は厚生労働省「賃金構造基本統計調査」に基づき、軽トラック 1,500-1,800 円、2t-4t トラック 1,800-2,200 円、10t トラック 2,000-2,500 円を設定している。平均速度は市街地走行を想定し 35-50 km/h である。

その他の変動費項目として、高速道路料金、タイヤ交換費、修理費、作業時間人件費、補助員人件費、回収容器費、消耗品費、通信費、マニフェスト費用、処理費が含まれる（詳細は付録参照）。

3.5.2 固定費（年間で発生するコスト）

固定費は走行距離に関わらず年間で発生する費用である。総コスト算出時は、固定費を年間走行距離で除算して距離単価に換算する。以下の 13 項目から構成される。

車両購入費および減価償却費

新車価格の市場相場（2025 年 10 月時点）に基づき、軽トラック 100-150 万円、2t トラック 300-500 万円、4t トラック 500-1,200 万円、10t トラック 2,000-2,800 万円である。

減価償却費は定額法、耐用年数 5 年を前提として算出される（車両購入費 ÷ 5 年）。

その他の固定費項目として、自動車税、重量税、保険（自賠責・任意）、車検費用および定期点検費用、車庫賃料、許認可費用、システム利用料、福利厚生費、社会保険料が含まれる（詳細は付録参照）。

3.5.3 総コストの算出

総コストは以下の式で計算される：

$$\text{総コスト} = \text{変動費単価} \times \text{走行距離} + \frac{\text{年間固定費}}{\text{年間走行距離}} \times \text{走行距離} \tag{1}$$

システムは、選択された車両タイプと走行距離に基づき、これらのコストを自動的に計算し、詳細な内訳とともにユーザーに提示する。これにより、運搬計画の経済性を事前に評価することが可能となる。

4　使い方

本システムは、Web ブラウザを通じて利用可能な対話的インターフェースを提供している。以下に、システムの基本的な使用方法を 5 つのステップで説明する。

4.1　ステップ 1: システムの起動

4.1.1　Windows 環境での起動

プロジェクトのルートディレクトリに配置された `run_app.bat` をダブルクリックすることで、システムが起動する。コマンドプロンプトウィンドウが開き、数秒後に Web ブラウザが自動的に起動する。

4.1.2　その他の環境での起動

Python がインストールされている環境では、ターミナルから以下のコマンドを実行する：

```
streamlit run app.py
```

システムが正常に起動すると、ブラウザに「未利用資源収集運搬最適化システム」のタイトルが表示される。

4.2　ステップ 2: 道路ネットワークの選択

システムの最初のステップとして、道路ネットワークを選択する。

4.2.1　選択肢

1. **デフォルトネットワーク**：群馬県大泉町を中心とした事前準備済みの道路ネットワーク
2. **カスタムネットワーク**：独自の道路ネットワークデータ（JSON 形式）をアップロード

4.2.2　デフォルトネットワークの使用（推奨）

初めて使用する場合は、デフォルトネットワークの使用を推奨する。「デフォルトネットワークを使用」ボタンをクリックすると、地図上に道路ネットワークが表示される。

4.2.3　カスタムネットワークのアップロード

独自の道路ネットワークを使用する場合、JSON 形式のファイルをアップロードする。ファイル形式の詳細は付録を参照されたい。

4.3　ステップ 3: 拠点・地点の設定

地図上で以下の 3 種類の地点を設定する。

4.3.1　デポ（車両の出発地点・帰着地点）

1.「デポを追加」ボタンをクリック
2. 地図上の任意の場所をクリックしてデポを配置

3. デポ名を入力（例：「車両基地」）

デポは通常 1 箇所設定するが、複数のデポを設定することも可能である（マルチデポ VRP）。

4.3.2 回収地点（資源の発生場所）

1.「回収地点を追加」ボタンをクリック
2. 地図上の資源発生場所をクリック
3. 回収地点名と資源量（kg）を入力

回収地点は複数設定可能である。各地点での資源量を正確に入力することで、より精度の高い最適化が可能となる。

4.3.3 集積場所（資源の最終目的地）

1.「集積場所を追加」ボタンをクリック
2. 地図上の集積場所をクリック
3. 集積場所名を入力（例：「バイオマス発電所」）

集積場所は、資源を最終的に運搬する先である。リサイクル施設、バイオマス発電所、処理施設などが該当する。

4.4 ステップ 4: 車両と資源の選択・最適化の実行

4.4.1 車両タイプの選択

ドロップダウンメニューから使用する車両タイプを選択する（14 種類から選択可能）。例：

- 少量運搬・狭い道路：軽トラック、2t 平ボディ
- 中量運搬・汎用性：4t 平ボディ、4t ウイング車
- 大量運搬・長距離：10t 平ボディ、10t ウイング車
- 液状資源：バキューム車
- 圧縮収集：4t パッカー車、大型パッカー車

4.4.2 資源タイプの選択

運搬する資源タイプを選択する（11 種類から選択可能）。例：

- 建設廃材、林業残材
- 農業残渣、剪定枝・草
- 食品廃棄物、廃食用油
- 廃プラスチック、古紙・段ボール
- 金属スクラップ
- 家畜糞尿、下水汚泥

4.4.3 適合性の自動検証

車両と資源を選択すると、システムが自動的に適合性を検証する：

- **適合**：問題なく運搬可能
- **条件付き適合**：追加対策（密閉容器、シートなど）が必要
- **不適合**：この組み合わせでは運搬不可

不適合の場合、警告メッセージが表示され、別の車両または資源の選択を促される。

4.4.4　最適化の実行

すべての設定が完了したら、「最適化を実行」ボタンをクリックする。システムが最適な収集運搬経路を計算し、数秒から数十秒で結果を表示する。

4.5　ステップ 5: 結果の確認とエクスポート

4.5.1　地図上での結果表示

最適化された経路が地図上に色分けされた線で表示される。各経路をクリックすると、詳細情報（距離、コストなど）がポップアップ表示される。

4.5.2　詳細レポートの確認

画面下部に表示される詳細レポートには、以下の情報が含まれる：

- 総走行距離
- 総運搬費用（変動費・固定費の内訳）
- 各経路の詳細（訪問順序、距離、所要時間）
- 車両別の稼働状況
- 資源回収量の集計

4.5.3　結果のエクスポート

結果は以下の形式でエクスポート可能である：

- **CSV 形式**：表計算ソフトで分析可能
- **JSON 形式**：プログラムで処理可能
- **HTML 形式**：レポートとして保存・共有可能

「結果をエクスポート」ボタンをクリックし、希望する形式を選択してダウンロードする。

4.6　応用的な使い方

4.6.1　複数シナリオの比較

異なる車両や経路設定で複数回最適化を実行し、結果を比較することで、最も経済的な運搬計画を選択できる。

4.6.2　季節変動への対応

資源量が季節により変動する場合、各季節のデータで最適化を実行し、年間を通じた運搬計画を立案できる。

4.6.3　新規拠点の評価

新しい回収地点や集積場所を追加した場合の影響を事前にシミュレーションし、投資判断の材料とすることができる。

5 結論

5.1 達成事項のまとめ

本研究では、群馬県内における未利用資源の収集運搬計画を支援するシステムを開発した。以下に主要な達成事項を総括する。

5.1.1 システム開発の達成

交通安全環境研究所で開発中の「地域交通計画立案ツール」を改良し、未利用資源運搬に特化したシステムを構築した。主な達成事項は以下の通りである：

1. **包括的な車両・資源データベースの構築**
   - 14 種類の運搬車両の詳細諸元（積載容積、燃費、購入費用など）を整備
   - 11 種類の未利用資源の特性と取扱い要件を体系化
   - 車両・資源の適合性マトリックス（154 組み合わせ）を作成
   - 実際のメーカーカタログおよび業界データに基づく信頼性の高いデータ
2. **高度な最適化アルゴリズムの実装**
   - Dijkstra 法および A*アルゴリズムによる最短経路探索
   - 容量制約付き車両経路問題（CVRP）の解決
   - 最近傍法と 2-opt 法による効率的なヒューリスティック探索
   - 数十から数百の回収地点を持つ実用的な問題を数秒から数分で解決
3. **詳細なコスト推計機能**
   - 変動費 12 項目（燃料費、人件費、高速道路料金など）の詳細計算
   - 固定費 13 項目（車両購入費、減価償却費、税金、保険など）の包括的算出
   - 総コストの自動計算と内訳の明示
   - 複数のシナリオ間でのコスト比較機能
4. **使いやすい Web アプリケーションの実現**
   - Streamlit による直感的なユーザーインターフェース
   - Folium による地図ベースの対話的な可視化
   - CSV、JSON、HTML 形式での結果エクスポート機能
   - 専門知識不要で利用可能な設計

5.1.2 仕様要件の達成

委託仕様書に定められた要件をすべて達成した：

- ツールの改良：地域交通計画立案ツールを未利用資源運搬に対応するよう改良
- 資源の変数化：11 種類の未利用資源をシステムに組み込み
- 車両の変数化：14 種類の運搬車両（軽トラから 10t トラックまで）に対応
- 経路・費用推計：単一収集場所から単一集積場所への最短経路と運搬費用を推計

5.1.3 学術的・実用的貢献

本研究は、以下の点で学術的・実用的な貢献をした：

- 未利用資源運搬における車両・資源適合性の体系的整理
- 実データに基づく詳細なコストモデルの構築
- 地方自治体や事業者が実際に利用可能な実用的システムの提供

- 循環型社会形成に向けた意思決定支援ツールの実現

5.2 今後の展望：人貨混載システムへの拡張

本システムの成果を基盤として、今後は人貨混載（passenger-cargo mixed transportation）システムへの拡張を検討する。人貨混載は、過疎地域における公共交通の維持と物流効率化を同時に実現する革新的な輸送方式であり、地域の持続可能性向上に寄与する可能性がある。

5.2.1 人貨混載システムの概念

人貨混載システムとは、同一車両で旅客と貨物を同時に運搬するシステムである。具体的には以下のパターンが考えられる：

- **バス・タクシーによる貨物運搬**：路線バスやデマンドタクシーの空きスペースを利用して、小口貨物や未利用資源を運搬
- **貨物車両による旅客運送**：宅配便や資源回収車両の帰路を利用した旅客運送
- **専用混載車両**：旅客と貨物の両方に対応した専用設計の車両

過疎地域では、旅客需要の減少により公共交通の維持が困難な一方、高齢者の移動手段確保は喫緊の課題である。同時に、少量分散型の貨物輸送も非効率的である。人貨混載により、これらの課題を統合的に解決できる可能性がある。

5.2.2 技術的拡張課題

人貨混載システムの実現には、以下の技術的拡張が必要である：

1. **動的経路最適化**：旅客の予約状況に応じてリアルタイムに経路を再計算
2. **旅客・貨物の優先度設定**：緊急性の高い移動を優先する仕組み
3. **スペース管理アルゴリズム**：限られた車両スペースを旅客と貨物に最適配分
4. **予約システム統合**：旅客予約と貨物配送依頼を統合的に管理
5. **安全性確保**：旅客と貨物の物理的分離、貨物固定方法の最適化

5.2.3 社会的意義

人貨混載システムの実現は、以下の社会的意義を持つ：

- **過疎地域の移動手段確保**：公共交通空白地帯における高齢者等の移動支援
- **物流効率化**：小口貨物と未利用資源の効率的な収集運搬
- **CO_2 排出削減**：旅客と貨物の統合輸送による走行距離削減
- **地域雇用創出**：運転手、予約管理者などの雇用機会提供
- **循環型社会形成**：未利用資源の効率的収集による地域資源循環促進

本研究で構築したシステムを基盤として、これらの拡張を段階的に実施することで、持続可能な地域社会の実現に貢献できると考えられる。

以上、本研究の成果と今後の展望について述べた。開発したシステムは、未利用資源運搬の効率化に寄与するとともに、人貨混載という新たな展開への基盤を提供するものである。

参考文献

公的機関

1. 国土交通省「自動車燃費一覧」、2025 年版
2. 国土交通省「自動車諸元表」、2025 年版
3. 厚生労働省「賃金構造基本統計調査」、令和 4 年
4. 全日本トラック協会「経営分析報告書」、2024 年度版
5. 全日本トラック協会「トラック運送事業の賃金・労働時間等の実態」、2024 年度版

業界団体・メーカー

6. いすゞ自動車株式会社、車両カタログ、2025 年版
7. 日野自動車株式会社、車両カタログ、2025 年版
8. 三菱ふそうトラック・バス株式会社、車両カタログ、2025 年版
9. UD トラックス株式会社、車両カタログ、2025 年版
10. 社団法人プラスチック処理促進協会「燃料消費原単位」、2024 年版

中古車販売・情報サイト

11. トラック王国、https://www.55truck.com/、2025 年 10 月アクセス
12. トラック流通センター、https://www.kaitoriou.net/、2025 年 10 月アクセス
13. TRUCK BIZ、https://www.truck-five.com/tfbiz/、2025 年 10 月アクセス
14. トラック市、https://www.truck-ichi.co.jp/、2025 年 10 月アクセス
15. バディトラック、https://buddytruck.jp/、2025 年 10 月アクセス

その他

16. 求人ボックス「トラック運転の年収・時給」、2025 年 10 月アクセス
17. 運転ドットコム「トラック運転手給与情報」、2025 年 10 月アクセス
18. トラック運送事業者へのヒアリング調査（2024 年 9 月-2025 年 10 月実施）

ソフトウェア・ライブラリ

19. Streamlit Team, Streamlit: The fastest way to build data apps, https://streamlit.io/, v1.28.0
20. NetworkX Developers, NetworkX: Network Analysis in Python, https://networkx.org/, v3.1
21. Folium Contributors, Folium: Python Data, Leaflet.js Maps, https://python-visualization.github.io/folium/, v0.14.0
22. Google OR-Tools Team, Google OR-Tools, https://developers.google.com/optimization, v9.7
23. McKinney, W., pandas: powerful Python data analysis toolkit, https://pandas.pydata.org/, v2.1.0

学術文献

24. Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1), 269-271.

25. Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2), 100-107.
26. Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6(1), 80-91.
27. Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle Routing: Problems, Methods, and Applications* (2nd ed.). SIAM.
28. Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, 6(6), 791-812.

付録 A　システム設計の詳細

本付録では、システムの設計に関する技術的詳細を補足する。

A.1　車両諸元データの詳細

表 1 に、システムに登録されている 14 種類の車両の詳細諸元を示す。

表 1: 車両諸元一覧（抜粋）

車両タイプ	積載容積 (m^3)	燃費 (km/L)	購入費 (万円)	年間固定費 (万円)
軽トラック	1.0-1.2	13-16	100-150	120-180
2t 平ボディ	2.0-3.0	6-11	300-500	180-300
2t ダンプ	2.5-3.5	6-10	350-550	190-320
4t 平ボディ	6-10	5-8	500-1,200	300-600
4t ダンプ	7-11	5-7	600-1,300	320-650
4t ユニック	5-8	4-7	1,000-1,800	400-800
4t ウイング	18-25	5-8	800-1,500	350-700
4t パッカー	8-12	5-7	1,200-2,000	450-850
4t アームロール	8-15	5-8	900-1,600	380-750
10t 平ボディ	10-13	3.5-5	2,000-2,800	600-1,200
10t ダンプ	12-18	3.5-4.5	2,200-3,000	650-1,300
10t ウイング	40-60	3.5-5	2,500-3,500	700-1,400
大型パッカー	20-30	3-4.5	2,800-4,000	800-1,500
バキューム車 (4t)	4-6	4-6	1,500-2,500	500-900

A.2　道路ネットワークデータ構造

システムで使用する道路ネットワークは、JSON 形式で以下の構造を持つ：

```
{
  "nodes": [
    {
      "id": "node_001",
      "lat": 36.2345,
      "lon": 139.3456,
```

```
      "type": "intersection"
    },
    ...
  ],
  "edges": [
    {
      "id": "edge_001",
      "source": "node_001",
      "target": "node_002",
      "length": 1250.5,
      "road_type": "local",
      "speed_limit": 40
    },
    ...
  ]
}
```

ノードは交差点や地点を表し、エッジは道路区間を表す。各エッジには長さ（m）、道路種別、速度制限が設定される。

A.3　最適化パラメータ

システムの最適化計算には、以下のパラメータが使用される：

- **探索時間制限**：最大 300 秒（デフォルト 60 秒）
- **2-opt 改善反復回数**：最大 1000 回（改善が無くなるまで）
- **距離行列キャッシュサイズ**：最大 10,000 ノード対
- **ヒューリスティック重み**：A*アルゴリズムで 1.0（ユークリッド距離と等価）
- **容量制約余裕率**：95%（安全マージン 5%）

A.4　システム要件

本システムを動作させるための推奨環境は以下の通りである：

- **OS**：Windows 10 以降、macOS 10.15 以降、Linux（Ubuntu 20.04 以降）
- **Python**：3.9 以降
- **メモリ**：4GB 以上（8GB 推奨）
- **ストレージ**：500MB 以上の空き容量
- **Web ブラウザ**：Google Chrome、Firefox、Edge（最新版）
- **インターネット接続**：地図表示に必要（オフライン使用も可能）

A.5　インストール手順

A.5.1　Python パッケージのインストール

```
pip install streamlit networkx folium pandas numpy ortools
```

A.5.2 システムの起動

```
streamlit run app.py
```

または、Windows 環境では `run_app.bat` をダブルクリックする。

A.6 データファイル構成

システムは以下のディレクトリ構造を持つ：

```
project_root/
├── app.py                      # メインアプリケーション
├── data/
│   ├── processed/
│   │   ├── compatibility.json    # 適合性データ
│   │   └── vehicle_specs.json    # 車両諸元
│   └── networks/
│       └── default_network.json  # デフォルト道路ネットワーク
├── src/
│   ├── optimization/          # 最適化モジュール
│   ├── visualization/         # 可視化モジュール
│   └── cost/                  # コスト計算モジュール
└── claudedocs/
    └── quickstart_guide.md    # クイックスタートガイド
```

付録 B 技術的補足

本付録では、システムの実装に関する技術的補足を記載する。

B.1 最短経路アルゴリズムの詳細

B.1.1 Dijkstra 法の実装

Dijkstra 法は、優先度付きキュー（ヒープ）を用いて効率的に実装される。以下に疑似コードを示す：

```
function dijkstra(graph, source):
    dist = {node: infinity for node in graph.nodes}
    dist[source] = 0
    priority_queue = [(0, source)]
    visited = set()

    while priority_queue is not empty:
        current_dist, current_node = pop_min(priority_queue)

        if current_node in visited:
            continue

        visited.add(current_node)
```

```
        for neighbor in graph.neighbors(current_node):
            edge_weight = graph.edge_weight(current_node, neighbor)
            distance = current_dist + edge_weight

            if distance < dist[neighbor]:
                dist[neighbor] = distance
                push(priority_queue, (distance, neighbor))

    return dist
```

計算量は、ヒープ操作が O(log V)、すべてのエッジを走査するのが O(E) であるため、合計 O((V + E) log V) である。

B.1.2 A*アルゴリズムの実装

A*アルゴリズムは、ヒューリスティック関数 $h(n)$ を用いて Dijkstra 法を拡張する：

$$f(n) = g(n) + h(n) \tag{2}$$

ここで、$g(n)$ は始点からノード n までの実際の距離、$h(n)$ はノード n から目標ノードまでの推定距離である。

本システムでは、ヒューリスティック関数としてユークリッド距離を使用する：

$$h(n) = \sqrt{(x_n - x_{\text{goal}})^2 + (y_n - y_{\text{goal}})^2} \tag{3}$$

ユークリッド距離は許容的ヒューリスティック（admissible heuristic）であり、常に真の距離以下となるため、A*アルゴリズムは最適解を保証する。

B.2 容量制約付き VRP の定式化

容量制約付き車両経路問題（CVRP）は、以下のように数理計画問題として定式化される。

B.2.1 記号の定義

- V: ノードの集合（$V = \{0, 1, ..., n\}$、0 はデポ）
- E: エッジの集合
- c_{ij}: ノード i からノード j への移動コスト（距離）
- q_i: ノード i での需要量
- Q: 車両の容量
- K: 車両の台数
- x_{ijk}: 車両 k がエッジ (i, j) を使用する場合 1、それ以外 0 の決定変数

B.2.2 目的関数

総移動コストの最小化：

$$\min \sum_{k=1}^{K} \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ijk} \tag{4}$$

B.2.3 制約条件

各顧客は 1 回だけ訪問される：

$$\sum_{k=1}^{K}\sum_{j\in V} x_{ijk} = 1, \quad \forall i \in V \setminus \{0\} \tag{5}$$

各車両の経路の流れ保存：

$$\sum_{i\in V} x_{ihk} - \sum_{j\in V} x_{hjk} = 0, \quad \forall h \in V, \forall k \in K \tag{6}$$

容量制約：

$$\sum_{i\in V}\sum_{j\in V} q_j x_{ijk} \le Q, \quad \forall k \in K \tag{7}$$

各車両はデポから出発・帰着：

$$\sum_{j\in V\setminus\{0\}} x_{0jk} = 1, \quad \forall k \in K \tag{8}$$

$$\sum_{i\in V\setminus\{0\}} x_{i0k} = 1, \quad \forall k \in K \tag{9}$$

部分巡回路除去制約（サブツアー除去）：

$$\sum_{i\in S}\sum_{j\in S} x_{ijk} \le |S| - 1, \quad \forall S \subset V \setminus \{0\}, |S| \ge 2, \forall k \in K \tag{10}$$

B.3 コスト計算の詳細

B.3.1 変動費の距離単価換算

変動費の各項目を距離単価（円/km）に換算する計算式を以下に示す。

燃料費：

$$C_{\text{fuel}} = \frac{P_{\text{fuel}}}{F} \tag{11}$$

ここで、P_{fuel} は燃料単価（円/L）、F は燃費（km/L）である。

人件費：

$$C_{\text{labor}} = \frac{W_{\text{hour}}}{V_{\text{avg}}} \tag{12}$$

ここで、W_{hour} は時給（円/時）、V_{avg} は平均速度（km/時）である。

作業時間人件費：

$$C_{\text{work}} = \frac{T_{\text{work}} \times W_{\text{hour}}}{D_{\text{avg}}} \tag{13}$$

ここで、T_{work} は 1 回あたりの作業時間（時）、D_{avg} は平均運搬距離（km）である。

B.3.2 固定費の距離単価換算

年間固定費を距離単価に換算する計算式を以下に示す。

$$C_{\text{fixed/km}} = \frac{C_{\text{fixed/year}}}{D_{\text{annual}}} \tag{14}$$

ここで、$C_{\text{fixed/year}}$ は年間固定費（円/年）、D_{annual} は年間走行距離（km/年）である。

年間走行距離は以下のように推定される：

$$D_{\text{annual}} = D_{\text{daily}} \times N_{\text{days}} \tag{15}$$

ここで、D_{daily} は 1 日あたり平均走行距離（km/日）、N_{days} は年間稼働日数（日/年）である。

B.3.3 総コストの計算

ある経路の総コストは、以下の式で計算される：

$$C_{\text{total}} = D \times (C_{\text{var}} + C_{\text{fixed/km}}) \tag{16}$$

ここで、D は総走行距離（km）、C_{var} は変動費単価（円/km）、$C_{\text{fixed/km}}$ は固定費の距離単価（円/km）である。

より詳細には：

$$C_{\text{total}} = D \times \left(\sum_{i=1}^{12} C_{\text{var},i} + \frac{\sum_{j=1}^{13} C_{\text{fixed},j}}{D_{\text{annual}}} \right) \tag{17}$$

ここで、$C_{\text{var},i}$ は変動費の第 i 項目（12 項目）、$C_{\text{fixed},j}$ は固定費の第 j 項目（13 項目）である。

B.4 地図可視化の実装

地図可視化は、Folium ライブラリを用いて実装されている。主要な機能は以下の通りである。

B.4.1 ベースマップの作成

```
import folium

map_center = [36.2345, 139.3456]  # 中心座標
m = folium.Map(location=map_center, zoom_start=13)
```

B.4.2 経路の描画

```
route_coords = [(lat1, lon1), (lat2, lon2), ...]
folium.PolyLine(
    route_coords,
    color='blue',
    weight=5,
    opacity=0.7,
    popup='Route 1: 15.5 km, 3,250 yen'
).add_to(m)
```

B.4.3 マーカーの配置

```
folium.Marker(
    location=[36.2345, 139.3456],
    popup='Depot',
    icon=folium.Icon(color='red', icon='home')
).add_to(m)
```

B.5 性能最適化

システムの性能を向上させるため、以下の最適化を実施している：

B.5.1 距離行列のキャッシュ

頻繁に使用されるノード間の最短距離を事前計算しキャッシュすることで、繰り返し計算を回避する。

```
@lru_cache(maxsize=10000)
def get_shortest_distance(node1, node2):
    return nx.shortest_path_length(
        graph, node1, node2, weight='length'
    )
```

B.5.2　グラフの前処理

道路ネットワークを読み込む際、不要なノードやエッジを削除し、グラフを簡略化する。

B.5.3　並列計算

複数の車両経路を独立に計算できる場合、Python の `multiprocessing` を使用して並列処理を実施する。

B.6　エラー処理とバリデーション

システムの堅牢性を確保するため、以下のエラー処理とバリデーションを実装している：

- 入力データの妥当性チェック（座標範囲、資源量の正負など）
- 道路ネットワークの連結性確認（すべてのノードが到達可能か）
- 車両・資源適合性の事前検証
- 容量制約の実現可能性チェック（総需要が車両容量を超えていないか）
- 計算時間の上限設定と早期終了

これらの技術的詳細により、システムの正確性、効率性、堅牢性が確保されている。