博士論文

人の生活に浸透するモビリティサービスの設計支援手法

村本衛一

主指導教員　篠田　陽一

北陸先端科学技術大学院大学
先端科学技術専攻
(情報科学)

令和 6 年 3 月

Abstract

Transportation systems are no longer just a means of transportation but have become part of an urban system with significant social, economic, and environmental impacts, and have a major impact on people's lives, businesses, social relationships, and the living environment. A more holistic approach is required.

In this study, I leverage advances in computer science and computing power to propose new mobility service design and evaluation methods that go beyond the limits of cost-benefit analysis in transportation planning. The proposed method is a mobility service design support and evaluation method that allows multiple decision makers to derive optimal solutions from many service options, keeping in mind that new demands will arise, and new mobility will be operated in new environments. This is a method that uses an event-driven simulator interactively while accurately grasping evaluation indicators by utilizing formal specifications. This method evaluates three or more indicators simultaneously, taking into consideration not only economy and convenience, but also safety. After investigation of regional demand and traffic volume and extraction of possible operation patterns. Then, a simulator is used to reproduce these and extract the optimal solution from the balance of evaluation indicators.

Using the power of formal specification description and model checking, which are results of computer science, I examined the liveness properties of ride-sharing services. Ride-sharing services are an example of parallel systems that require liveness properties, such as dispatching rides during childbirth or accidents, or delivering medicines. It is socially unacceptable that a ride is not dispatched even after a reservation has been made. A ride-sharing system is described using linear temporal logic and Kripke structure. Using a specification description language called Maude, I showed an example of description of the ride-share system separating essential system specifications and parameters such as maps, locations of people, and vehicles. It is known that in parallel systems, the liveness property cannot be satisfied unless a fairness assumption is made. In this study, I pointed out that model checking of ride-sharing systems also requires strong fairness assumptions. If I make the assumption of fairness, the amount of calculation will explode, and even current computers may not be able to calculate it. For this problem, I adopted a divide-and-conquer approach and succeeded in speeding up model checking. It was demonstrated that model checking, which cannot be calculated on current computers due to the possibility of running out of memory, can be shortened to approximately 3 minutes and 43 seconds.

As a practical example of how to design and evaluate new mobility services by utilizing computational power using a simulator, I applied the proposed method to cases of transporting people and goods. As an example of transporting people, I evaluated a self-driving ride-sharing system service in the Osaka area. In the Osaka area, there is data on the server where employees stamp the time on their employee cards when they enter and exit buildings, making it possible to accurately reproduce the movements of around 300,000 trip a day. I defined a safety indicator that calculates the severity of an accident based on the frequency of events in which an autonomous vehicle and a person are close to each other within a certain distance, and the weight and speed of the vehicle. Using StarBED, an HPC, I conducted a complete search for an optimal solution that took into account not only the economic efficiency and convenience indicators used in existing cost-benefit analyses, but also three indicators, including a safety indicator. I also searched

for the optimal solution using the annealing method. As a result, contrary to the intuition that the optimal solution would be obtained by operating vehicles with a maximum capacity of 4 passengers per vehicle, I found that operating vehicles with 3 passengers per vehicle was optimal. In addition, since simulations can identify locations where people and vehicles are in proximity, when the location where the event occurred was visualized on a map, it has been confirmed that this location matches the location where an actual proximity event between a self-driving vehicle and a person occurred. In other words, it has been found that a more optimal solution can be obtained by redesigning a route that avoids places where dangerous events occur or by creating a new operation plan that limits the maximum speed when passing through such places. It can be said that the solution was obtained through comprehensive testing using the power of computers.

In addition, a simulator was used to utilize computing power to design and evaluate a delivery service using an automatic delivery robot at Fujisawa SST. At Fujisawa SST, many residents live by ordering, receiving, and consuming products. Assuming deliveries from multiple stores, I searched for an optimal solution using factors such as the number of robots and business hours as simulation parameters. Since there are over 1 billion parameter combinations, I proposed a method to find the optimal solution using an interactive method. I run multiple limited simulation batches and display Pareto solutions on the evaluation axes of safety, economy, and convenience to help decision makers select the best solution. As a result, even though there are over 1 billion options, many evaluators choose similar solutions as the best. Through a questionnaire, I conducted subjective evaluations regarding execution time, explainability, UI, persuasiveness, and overall understanding. As a result, the evaluation of execution time and explainability was high. Although the UI, persuasiveness, and overall understanding were evaluated favorably, it was confirmed that there is still room for improvement.

Furthermore, I defined a fourth indicator, plotted it in three-dimensional space, and proposed a method to compare Pareto solutions. In addition to safety, economy, and convenience, I defined new constraints such as “stopping time on the road” and “waiting time at the store” using descriptions of signal temporal logic (STL) and I searched for the optimal solution using an interactive method using a simulator. Mobility services include many indicators that change temporally and spatially, such as stop time while moving and waiting time at stores. By introducing the STL formula into evaluation using a simulator, the number of items that can be designed and evaluated using the proposed method has increased, making it a more practical design and evaluation method.

The proposed method is suitable for the design and evaluation of mobility services where it is desirable to formulate a hypothesis and verify it using multiple evaluation indicators, especially mobility services that require the introduction of new mobility for which driving routes and operating methods have not yet been established.

Keywords: Transportation planning, Autonomous mobility, Decision simulation, Multi-objective optimization

概 要

交通システムが単なる移動手段から、社会的、経済的、環境的な影響が大きい都市システムの一部となり人々の生活やビジネス、社会的関係、生活環境などに大きな影響を与えるため、その計画には、より総合的なアプローチが求められている。

本研究では、計算機科学と計算能力の進歩を活用して、交通計画の費用便益分析の限界を超える新しいモビリティサービスの設計と評価手法を提案する。提案手法は、新たな要求が発生し新たなモビリティが新たな環境で運用されることを念頭に、複数の意思決定者が多数のサービス選択肢から最適解を導出できるモビリティサービス設計支援・評価手法であり、形式仕様の活用で評価指標を正確に把握しつつ、事象駆動シミュレータを対話的に用いる手法である。この手法では、経済性と利便性だけでなく、安全性も考慮した 3 つ以上の指標を同時に評価する。地域の需要と交通量を調査し、運行パターンの候補を抽出する。そして、シミュレータを使用してこれらを再現し、評価指標の均衡から最適な解を抽出する。

計算機科学の成果である形式仕様記述とモデル検査の力を利用して、ライドシェアサービスの活性特性（liveness property）を検査した。ライドシェアサービスは、出産や事故時の配車、薬剤の配送など、活性特性が求められる並列システムの一例である。予約したにもかかわらず配車が行われない状況は社会的に受け入れられない。線形時相論理とクリプキ構造を使用してライドシェアシステムを記述した。Maude という仕様記述言語を用いて、本質的なシステム仕様と地図や人や車の配置などのパラメータを分離して記述する例を示した。並列システムでは、公平性の仮定を置かないと活性特性が満たされないことが知られている。本研究では、ライドシェアシステムのモデル検査においても強い公平性の仮定が必要であることを指摘した。公平性の仮定を置くと、計算量が爆発し、現在の計算機でも計算不可能な場合がある。この問題に対して、分割統治アプローチを採用し、モデル検査を高速化することに成功した。現在の計算機ではメモリ不足になる可能性があるため計算不能なモデル検査を、約 3 分 43 秒に短縮できることを例示した。

シミュレータを用いて計算機能力を活用して、新たなモビリティサービスの設計・評価を行う方法の実践として、人を運ぶ事例とモノを運ぶ事例に提案手法を適用した。人を運ぶ事例としては、大阪地区での自動運転ライドシェアシステムのサービスを題材に評価を行った。大阪地区では、社員がビルの出入りの際に社員カードで時刻を刻印するデータがあるため、1 日 30 万トリップ程度の人の動きをほぼ正確に再現できる。自動運転車両と人とがある一定の距離内に近接する事象の頻度と車両の重量及び速度から事故発生時の重篤度を算出する安全性の指標を定義した。既存の費用便益分析で用いられてきた経済性、利便性の指標のみならず、安全性の指標を含めた 3 指標を加味した最適解を HPC である StarBED を活用して全探索した。また、焼きなまし法を用いて最適解を探索した。この結果、最大積載人数の 4 名/台の車両を運行することで最適解が得られるという直感とは反して、3 人乗りの車両を運行したほうが最適であるという結果を得ている。また、シミュレーションでは、人と車が近接する場所を特定できるため、当該事象の発生場所を地図上に可視化したところ、実際の自動運転車両と人との近接事象が発生した場所と合致することが確認できている。すなわち、危険事象が発生する場所を回避する経路を再設計したり、当該場所を通過する時の最高速度を制限した新たな運行計画を作成することで、より最適な解が得られることが分かった。計算機の力を使って網羅的に検査することで得られた解であると言える。

また、シミュレータを使用して計算能力を活用し、藤沢 SST での自動配送ロボットによる配送サービスの設計と評価を行った。藤沢 SST では、多くの住民が商品を注文し受け取り消費して生活している。複数の店舗からの配送を想定し、ロボットの数や営業時間などの要素をシミュレーションパラメータとして最適解を探索した。パラメータの組み合わせは 10 億を超えるため、複数の限られたシミュレーションバッチを実行し、安全性、経済性、利便性の評価軸上にパレート解

を表示し、意思決定者がベストな解を選択していく対話的な方法で最適解を見つける手法を提案した。結果、10 億以上の選択肢があるにも関わらず多くの評価者は類似した解を最適として選ぶ結果が得られている。アンケートにより、実行時間、説明性、UI、納得性、全体把握性に関する主観評価を行った。結果、実行時間と説明性の評価は高かった。UI、納得性、全体把握性に関しては良好な評価ではあるが、まだ改善の余地もあることが確認できた。

さらに、4 つ目の指標を定義し、これを 3 次元空間にプロットし、パレート解を比較できる方法を提案した。安全性、経済性、利便性に加えて、「道路での滞留時間」と「店頭待ち時間」という新たな制約を信号時相論理（STL) の記述を用いて定義し、シミュレータを使用した対話的な手法で最適解を探索した。STL の記法により評価指標について評価者の共通理解を得た上で、既存の Optuna 方式と提案方式の比較した。結果、提案方式は評価者の戦略をよく反映しているが、網羅性に関しては既存方式が高い評価を得た。既存方式と提案方式のシミュレーション実行回数を比較すると、既存方式の方が 25 倍多くのシミュレーションを実行していたため、シミュレーションバッチの回数を増やすことで提案手法の網羅性を高められることが分かった。モビリティサービスには、移動中の停留時間や店舗での待ち時間など、時間的空間的に変化する指標が数多く存在する。シミュレータを用いた評価に STL 式を導入することにより、提案方式で設計・評価できる項目が増え、より実用的な設計・評価方法にできた。

仮説を立案し複数の評価指標により検証することが望まれているモビリティサービス、特に走行路や運行方法が確立していない新たなモビリティの投入が必要となるモビリティサービスの設計・評価に提案手法は好適である。

Keywords: 交通計画, 自律移動モビリティ, 意思決定シミュレーション, 多目的最適化

目 次

第1章　序論

1.1　旧来のモビリティサービスの計画

通勤、買い物、医療や教育施設への移動やモノの製造や製品の供給等、モビリティサービスを設計し社会に実装することは、我々の生活に欠かせない。本章では、これまでの交通計画に伝統的に用いられてきた手法の限界点と本研究でのアプローチについて説明する。

1.1.1　費用便益分析

これまで古くから、大規模なインフラ投資を伴う交通計画では、交通インフラ整備がマクロ経済への効果や各地域の人口や経済力への影響を推計するため様々なモデル [63, 71, 77] が提案されてきた。過去、様々な指標が乱立する状況で交通インフラ整備が行われてきた反省から、国土交通省では、鉄道や道路計画の費用便益分析のマニュアル案 [70] を策定してきた。つまり、大規模な投資の意思決定を行うために関連するステークホルダーの間で統一的な視点で評価する方式を定めてきており、高速鉄道ネットワーク [20]、渋滞課金システム [33]、電動市バスの運行 [43] など、さまざまな交通システムに対して費用便益分析が長く用いられている。

費用便益分析では、ある年次を基準年とし、一定期間の便益額、費用額を算定する。図 1.1 は、費用便益分析のフローの概略を示したものである。

便益については、道路整備が行われる場合と、行われない場合の交通流推計を用いて「走行時間短縮」、「走行経費減少」、「交通事故減少」の項目について、道路投資の評価手法として定着している消費者余剰を計測することにより便益を算出する。対象とする道路網の範囲において、OD 表（Origin Destination Table）、分布交通量を用いて交通流の推計し、便益計算の基礎とする。

走行時間短縮便益は、道路の整備・改良がない場合の総走行時間費用から、道路の整備・改良がある場合の総走行時間費用を減じた差として算定する。総走行時間費用は、各トリップのリンク別車種別の走行時間に時間価値原単位を乗じた値をトリップ全体で集計する。

走行経費減少便益は、道路の整備・改良が行われない場合の走行経費から、道路の整備・改良が行われる場合の走行経費を減じた差として算定する。

交通事故減少便益は、道路の整備・改良が行われない場合の交通事故による社会的損失から、道路の整備・改良が行われる場合の交通事故による社会的損失を減じた差として算定する。ここで発生事故率を基準とした算定式を用いてリンク別の交通事故の社会的損失を算定している点に注意したい。事故率は道路・沿道区分に特定の値を用いたモデル化が行われており、車種や交差点の種別や交通量の変化は加味されていない。

四段階推定法

費用便益分析で用いる交通量は、四段階推定法を用いて計算される。四段階推計法では、将来交通需要の予測にあたり、段階に分け、徐々に詳細に推計する方法である。米国で開発され、日本

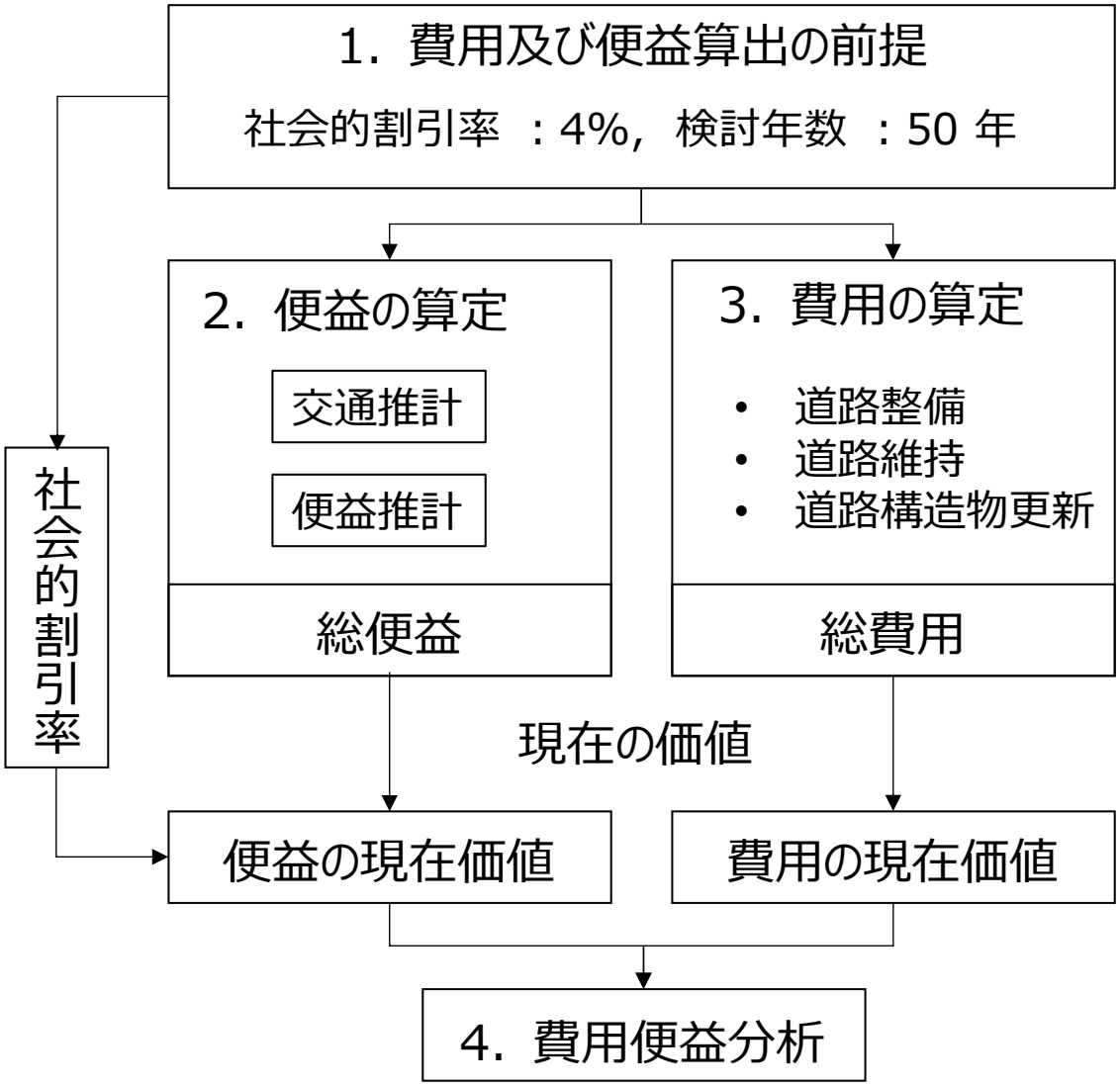

図 1.1: 費用便益分析のフロー

Fig. 1.1: Cost-benefit analysis flow

の交通計画においても利用されている。

算出対象地域を複数のゾーンに分割し、以下の４段階で予測を行う手法が用いられている。

1. 発生・集中交通量の予測
2. 分布交通量の予測
3. 分担交通量の予測
4. 配分交通量の予測

まず、各ゾーンから発生、あるいはそのゾーンを目的地として集中する交通量を予測し、ゾーン間の移動の分布と各ゾーンの発生量・集中量を整合させる。次にそれぞれのゾーン間の移動を複数の交通機関で分担する割合を定め、最後に道路ネットワークでの移動の配分を割つける。

配分交通量を算出・検討する際に用いる道路のリンクコストやＯＤ間旅行コストが決まっていないと現実的には分布交通量や分担交通量予測を行うことができず、鶏卵問題となっており、現実的には、四段階推計の途中で用いられたＯＤ間旅行コストと最後に算出されるＯＤ間旅行コストの値は一致しないという問題が生じている。また、定常状態を仮定した静的な分析であり時間軸に沿った交通状況の変化は取り扱えない課題が指摘されている [76]。

1.2 旧来の方式の限界

上に述べた費用便益分析による交通計画の方式では不十分となる要因を列挙する。

要求の多様化 地球温暖化、地域の活性化、住民の健康促進などのモビリティサービスに対する要望が多様化し、新しいテクノロジーを導入したより持続可能な交通システムの構築が望まれている。

新たなモビリティの出現 自動車業界の変革で話題になっている、CASE（Connected, Autonomous, Shared, Electric）に加えて、遠隔操作型小型車といった新たな自律モビリティ、キックボードといった新たなモビリティの実用化、自転車等の利便性向上のための技術の導入、歩行者の安全性向上のための技術が望まれている。

我が国においても、技術革新による自動車業界の変革のみならず、気候変動を食い止めるため、2050 年にカーボンニュートラルを達成すべく、公共交通の利用促進、デジタル化による物流効率化、電気自動車の導入促進等、モビリティサービスを見直す機運が高まっている [68]。また、レベル４自動運転の公道実証、電動キックボード免許なしでの利用、遠隔操作型小型車の公道での運用を実現する法改正 [66] が行われ、自律移動を行う新たなモビリティの社会実装が可能となってきている。

すなわち、費用便益分析では、交通需要の調査を起点としており、よりクリーンで省エネな交通手段への変更や自動車中心の道路利用から生活者中心の運用運用の変更といった要望に対する要望を加味できないため、既存の費用便益分析による交通計画だけでは不十分となってきている。

1.3 小型電動モビリティへの期待

ガソリン車に比べて CO_2 排出量が少なく、環境にやさしい交通手段として小型電動モビリティに対する期待が高まっている。小型モビリティは、車両本体価格が比較的安価で、燃料費も安い

ため、低コストとなる。都市部では、路上駐車による車両の渋滞や駐車場の不足が問題となるが、小型のモビリティの占有面積は少なく、比較的狭い場所での駐車が可能となるため、都市交通の混雑緩和につながると期待されている。

1.3.1 小型自律移動モビリティ

前述の遠隔操作型小型車は、前述の法施行を受け、まさに実社会への実装始まる段階となっている。遠隔操作型小型車では、運転の一義的な責任は遠隔操作者にあるものの、運行においては機体の自律移動機能の役割が大きい。機体は基本的に自律的に移動を行うが、駐車車両の回避等人による判断が必要な場合のみ、遠隔操作者が遠隔操作により介入する。このような機体の社会実装を実現するには、機体の自律移動時の安全性、遠隔制御時の安全性の確保が必要となる。小型の自律移動モビリティに関する安全基準[1]では、速度が低く質量が軽いため最大運動エネルギーが小さく衝突時に人に危害を加える確率が低い機体の安全性能を最大運動エネルギーでカテゴリ分けし、本質安全を満たせないカテゴリの機能安全の信頼性のレベルを区分けして定義している。遠隔操作時の安全性に関しては、一定以上の遅延が生じた場合、機体が安全停止することで安全性を担保する。我が国では、この考え方をもとに、一般社団法人ロボットデリバリー協会で安全基準を 2023 年 3 月に策定 [16] し、2023 年 6 月その審査を開始し、7 月に最初の合格証を発行、8 月に届出制に基づきパナソニックホールディングス株式会社が東京都と神奈川県に届出を行い公道走行を開始している [18]。

安全基準の国際標準に関しては、IEC TC125 で Autonomous Driving Cargo e-Transporter（ACeTs）の安全基準が作成されており、日本から NP : New work item Proposal (新業務項目提案) の投稿が行われ、2025 年 3 月の発行に向けて標準化作業が行われている。

先に述べた TC125 では、日本では、中型中速と呼ばれる車道を走るタイプの ACeTs の安全基準も同じ枠組みで策定作業が行われている。

かつて工場の動力が蒸気機関から電気モータに置き換えられた当初、単に工場に動力を供給する部分を電動化しただけでは生産性があがらなかった。工作機械の一つ一つを工場内で供給される回転エネルギの動力で駆動させるのではなく、旋盤や加工機器や手持ちの電動ドリルに置き換え、電子制御による自動化が進んだことで、生産能力に柔軟性が付加された結果として生産性が飛躍的に向上した[2]。現在、自動車の電動化が進んでいる。サハラ砂漠を走れるような車体を持った自動車をそのまま電動化させて、それを使いながら近所のコンビニに行っているような状況で人々が幸せで持続可能な社会が実現できるだろうか。我々は現在当たり前に存在する自動車の機能が形を変えて、人と協調する低速小型モビリティや自動配送ロボットや自動回送のパーソナルモビリティといった姿になり、街に実装されていくことで、持続的な開発目標が達成されると考える。

1.4 生活空間としての道路の活用

文献 [15] の中で、道路は、ライフラインの収容、防災や環境保全、市街地の形成、景観の形成、コミュニティの形成等のための空間を提供する。特に、道路は古来より、人々の交流やコミュニケーションを育む場であった。子供たちが遊んだり、大人が立ち話や井戸端会議を行う光景が、かつては至るところで見られたものの、モータリゼーションにより失われてしまったと指摘してい

[1] JIS B 8446-1 生活支援ロボットの安全要求事項

[2] 文献 [67] では、電動機が、工作機械に直接組み込まれたる利点に加え、工作機械の主軸の回転数をコントロールすることが容易になり、回転数と切削効率の研究を進めた事実が記述されている

る。我々も、同様の見解を持っている。この文献は道路の役割の変化として、「行きたくなる、居たくなる道路」と道路の将来像を示している。本研究では道路の役割のみならず新たなモビリティを特定エリアにサービスして導入・改善する際の設計と評価に注目し、その手法を提案する。

1.5　費用便益分析から都市システム計画へ統合的なアプローチの必要性

本研究の目標は、モビリティサービスに対する要求の多様化、新たなモビリティの出現といった変化の中、費用便益分析による交通計画の限界を計算機科学と計算機能力の進展で超えることにある。既存の手法の具体的な問題点と解決するためのアプローチを述べる。

1.5.1　費用便益分析の問題点

費用便益分析の交通計画の限界を列挙する。

安全性の指標の不足 1.1.1 で述べた通り、費用便益分析では、事故の発生は道路種別毎に定義された確率で表現される。鉄道や道路計画では費用便益分析が用いられていた。しかし、持続可能な交通システムを構成にあたっては、自律移動機能を伴う新たなモビリティを用いたサービスを設計実装する必要がある。このようなモビリティは固定起動を走るのではなく自由軌道を走行する。既存の費用便益分析の評価指標である利便性、経済性に加えて、どこで危険な事象が発生するかといった分析を可能とする安全性の指標を含めた評価が必要となる。

多様な評価の欠如 既存の費用便益分析では、便益は、「走行時間短縮」、「走行経費減少」、「交通事故減少」といった項目表現される。しかし、これらの評価だけでは、地球温暖化、地域の活性化、住民の健康促進などのモビリティサービスに対する要望をどのように満たしているか評価できない。このため、単純な費用便益に還元した指標のみならず、多くの評価指標を総合的に加味しながら新たに定義するモビリティサービスを評価する必要がある。

定義の曖昧性 費用便益分析では、各指標の解釈は自然言語で記述されているのみであり、具体的なデータの収集統計方法や数値を積算する範囲は規定されておらず、個別の交通計画のプロジェクトで費用便益分析を行う際は、具体的な指標算出の方法を個別に規定する必要があり、指標の意味するものが曖昧となりプロジェクト間の比較も困難となる課題がある。また、上に述べたように多くの評価指標を扱う場合、意思決定を行う複数のステークフォルダの間で評価指標を誤解なく正確に理解して評価する必要性が高まる。

新たなモビリティサービスの設計・評価では、費用便益分析で設計・評価できた交通計画の範囲では不十分で、より統合的な都市システム計画の一部として複数基準による評価が求められている。都市交通の持続可能性評価に関する 99 の論文を調査した文献 [40] では、2/3 の論文が指標やフレームワークの開発、モデリングやシミュレーション、複数基準決定分析 (MCDA：multiple-criteria decision analysis) に関する論文で構成されていることを示している。

1.5.2　解決に向けたアプローチ

これまで説明してきた新たなモビリティサービスに対する要求と費用便益分析の問題点の関係を図 1.2 に示す。モビリティサービスに対する要求の多様化、新たなモビリティの出現といった変化の中、費用便益分析による交通計画の限界を計算機科学と計算機能力の進展で超えることが本研究の目指すところである。

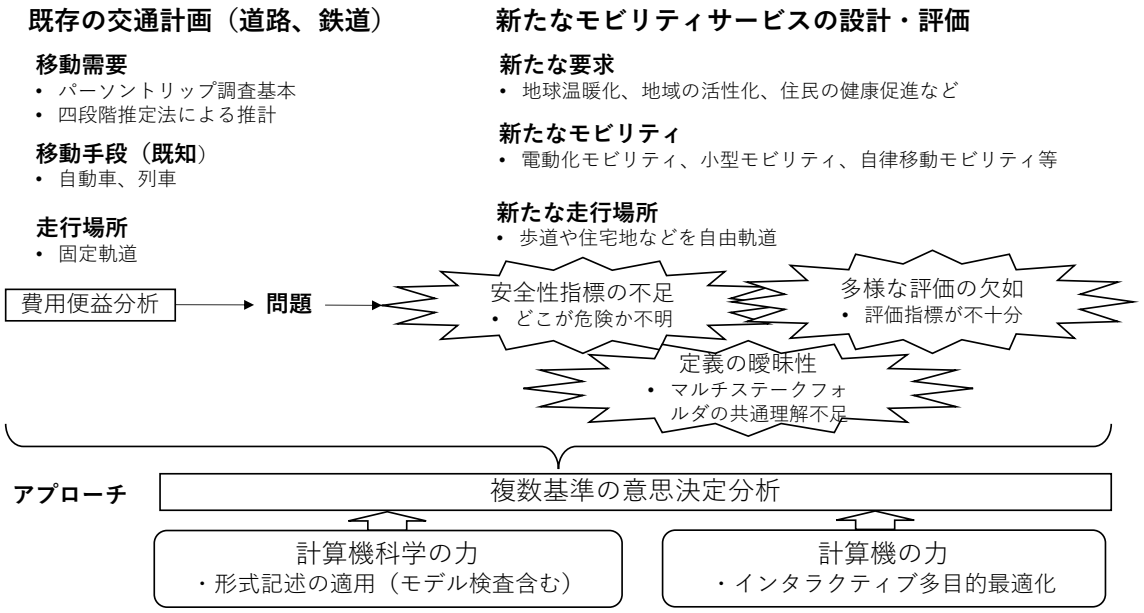

図 1.2: 費用便益分析の問題と解決に向けたアプローチ

1.6 本論文の構成

本論文は、次のように構成する。第 1 章の序論では、交通計画の既存手法である費用便益分析の限界とそれを解決するアプローチについて述べた。第 2 章で関連研究について述べる。第 3 章で、本研究の提案手法を述べる。第 4 章で、形式的記述を利用し曖昧性を排除し、モビリティサービスが目的の動作を行うか検査する試みについて述べる。第 5 章と第 6 章では、計算機能力の進展により多目的最適解探索を解決する手法について述べる。第 5 章は、人を運ぶモビリティサービスに、第 6 章は、モノを運ぶモビリティサービスに本手法を適用し、最適解探索に形式記述を用いることで評価指標の曖昧性を排除しつつ最適解をインタラクティブに探索する手法を示し、第 7 章でまとめる。

本論文の各章の関係を図 1.3 を用いて説明する。

本研究の提案は、第 3 章で示す。提案手法は、新たな要求が発生し新たなモビリティが新たな環境で運用されることを念頭に、複数の意思決定者が多数のサービス選択肢から最適解を導出できるモビリティサービス設計支援・評価手法であり、形式仕様の活用で評価指標を正確に把握しつつ、事象駆動シミュレータを対話的に用いる手法である。第 3 章では、提案手法のフローが示される。

本研究で採用する先行研究の技術と本研究の関係を第 2 章で説明する。本研究では、様々な要求に応じるため、交通計画から都市計画へ発展した統合的なアプローチを参考にしている。また本研究では、どこで危険な事象が発生するかといった分析を可能とするシミュレーション、多様な評価を考慮して評価する多目的最適化、指標の曖昧性を排除する形式仕様を採用している。これらの技術の先行研究と本研究の差分を第 2 章で説明する。

図 1.3 で提案手法が、形式記述（計算機科学応用）と計算機の力を活用する方式であることを図示している。形式記述の応用は、第 4 章と第 6 章で説明している。また、計算機の能力を活用する方式を説明する章は、第 5 章と第 6 章である。

第 4 章では、線形時相論理を用いて自動運転ライドシェアシステムの仕様をモデル検査した研究について示す。第 3 章で示す提案手法のフローの最初の段階で実施する工程にあたる。重要な社会インフラとなるモビリティサービスが満たすべき活性特性を担保する方法を説明する。

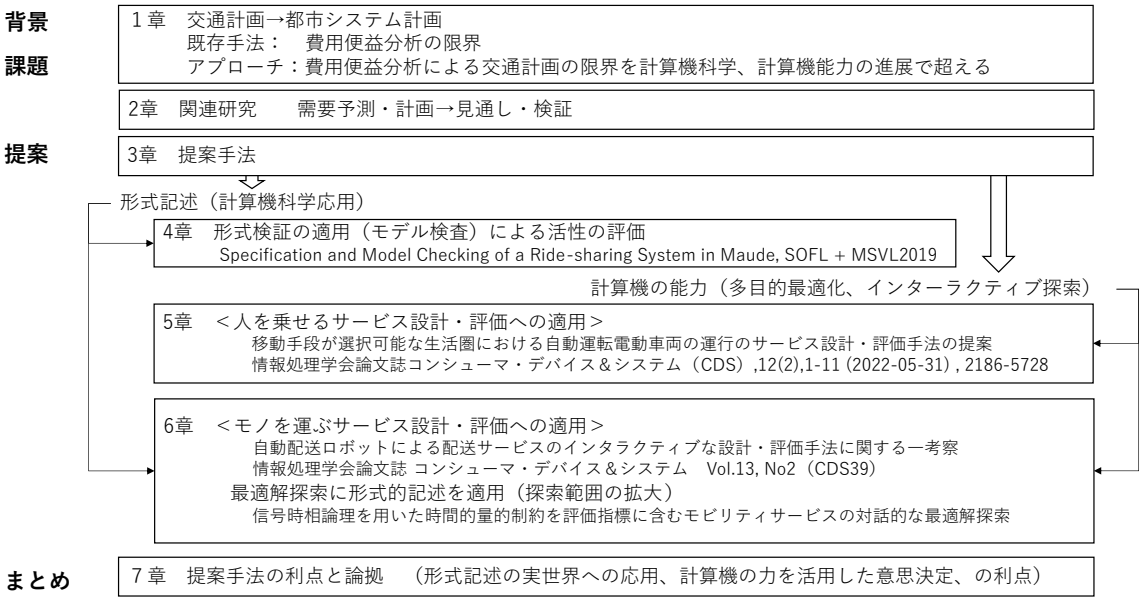

図 1.3: 本論文の構成

第 5 章では、人を運ぶモビリティサービスに提案手法を適用した例について説明する。データドリブンな分析に基づきシミュレーションで人や自動運転車両の動きを再現するため、ヒヤリハット事象の発生場所が特定できるなど安全性の評価の精度が高まることが示される。また、計算機を使った可能性の探索であるため、分析の抜け漏れ防止、人知を超えた変化点・変化量の把握が可能となり効率的な解の発見につながることが示される。

第 6 章では、モノを運ぶモビリティサービスに本手法を適用した例について説明する。計算機シミュレーションを対話的に使った可能性の探索の手法であるため、意思決定者の戦略を反映した最適解が選択可能となり、また説明性が向上することが示される。また、形式記述を用いることで、意思決定者の間で時間的空間的な変化を含む評価指標を正確に把握しながら評価する例が示される。

第 7 章では、本研究の結論を述べるとともに、本研究で示した利点と論拠を述べている章・節を提案手法のフローに対応づけて整理している。

第2章　関連研究

本章では、3 章で提案する手法の位置づけを明確にするため、関連する先行研究との関係を述べる。図 2.1 は、本研究と本章で説明する関連研究との関係を示したものである。本研究では、様々な要求に応じ、交通計画から都市計画へ発展した統合的なアプローチを参考にしている。本研究では、どこで危険な事象が発生するかといった分析を可能とするシミュレーション、多様な評価を考慮して評価する多目的最適化、指標の曖昧性を排除する形式仕様を採用している。

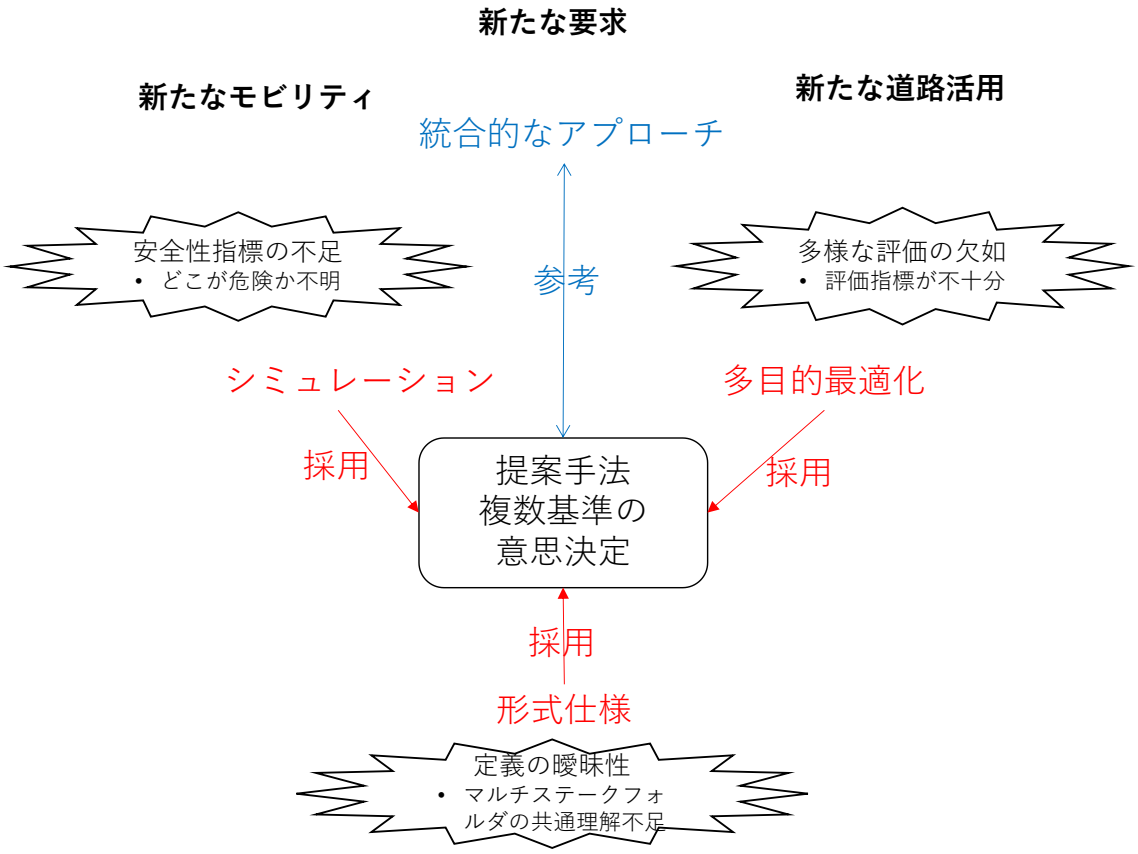

図 2.1: 本研究と関連研究の関係

Fig. 2.1: Relationship between this study and other studies

次節より、交通計画から都市計画への流れについて説明し、本提案で利用する形式記述、多目的最適化、シミュレーションを用いた交通計画の先行研究について説明する。複数の要因が関係する要因を読み解き複数のステークホルダーの間で同意を得ることは難しいことであるが、シミュレーションを用いた評価や段階的評価による先行研究について述べたのち、その解決に形式記述の適用を試みた例について説明する。

2.1 交通計画から都市計画へ

交通システムは、人々の生活やビジネス、社会的関係、環境などに大きな影響を与えるため、より総合的なアプローチが求められている。文献 [40] では、持続可能な都市交通システムに関する研究を総合的にレビューし、より持続可能な交通システムの構築の必要性を論じている。著者らは、都市交通が環境、社会、経済的に持続可能であるためには、自転車や歩行、公共交通などの持続可能な交通手段の利用を増やすことが必要であることを指摘している。

高見淳史氏は、2019 年に開催された講演「転換期の都市交通計画」の中で、きたるモビリティの大変革を賢く都市が受け入れるために今できることは何か？との問題意識から、都市交通計画のアプローチを、「予測して供給する」、交通計画から、「見通しを立てて検証」へと変革することが望まれていると主張している [14]。

米国の全国都市交通安全協会 (NACTO:National Association of City Transpotation Officials) が発表した自動運転時代の都市の姿に関わる文章 [7] では、自動運転の車両の普及で都市に駐車場が不要となる未来の姿だけでなく、道路が車両の通行や歩行者や自転車の利用にも利用されるため、道路空間の利用を変えることで、都市の持続可能性を高めることの必要性にも言及している。

このように交通計画に対して、単なる需要予測に基づき各交通機関の分担量の予測を提供するのみならず、より統合的なアプローチにより問題を定義し検証していく方法へと変革が求められる。

2.2 多目的最適化の適用

交通計画において多様な要素を考慮する必要がある。本節では、マクロ経済に及ぼす影響を考慮した例や自律移動モビリティの導入の影響を評価した例を説明する。

根津らは、高速道路や新幹線のような交通インフラ整備において、人流・物流といったフロー効果のみならず、沿線地域を成長させるストック効果を評価可能なモデルシステムの構築と構築したモデルシステムの挙動の検証を行っている [71]。このモデルの中では、交通インフラ網整備による、地域間の連結性を表現するアクセシビリティの向上や、公共投資額の変化に応じた実質 GDP(Gross Domestic Product) 等の変化を推計することができることに加え、インフレ・デフレ状況といった異なる経済情勢下で公共投資の乗数効果が異なることを考慮した変数を内在化させている。この評価の中で、交通利便性の向上、とりわけ新幹線整備が生産年齢人口割合を向上させるといった事象や新幹線整備により地域内第 1 次産業 1 人当たり生産額が増加するといった事象の関係を表現可能であるとしている。公共投資のストック効果の評価を試みたモデルであるが、域内の一人当たり GRP(Gross Regional Product) は、産業別の人口分布を用いた推計を用いて検証しており、著者自身もモデルの継続的な改善が必要と主張している。また、鉄道や高速道路整備と人口増加の関係を調査した論文 [77] や開設されたインターチェンジから周辺 1km 以内においては、それまでの都市開発の変遷に影響を受けるものの、工業団地の立地や地価の上昇の効果を示した取組がある [63]。これらの研究では、どの経路を通過させることにより安全性が変化するかといった点は考慮しておらず、利便性（フロー効果）とインフラとしてのストック効果に着目して評価している。

本研究では、利便性の指標の計算において、ストック効果の算出は対象とせず、人やモノを搬送した際に得られる便益であるフロー効果のみを対象している。また、自由軌道を走行する自律移動モビリティを想定するため安全性の評価も可能とする。

2.3 シミュレーションによる交通計画の評価

人の流れ表現する方法として、洪水のように巨視的な動きとして捉える方法 [39] と引力斥力で人の動きをとらえる微視的な方法 [38] がある。微視的な方法では、歩道上での帯状集団の生成などが観測されることが知られており、車道へのはみだしなどリスクにつながるリアルな行動を再現できる。本研究では微視的な方法を用いる。

文献 [62] は、自動運転社会下の街路空間の評価指標の作成と乗降環境を考慮した街路空間の可視化を行っている。ステーション配置の検討ための評価指標として、遅れ時間と旅行速度を用いて、乗降場の規模が大きいほど駐車・発進挙動の車両に遭遇する回数が増加するため、遅れ時間が増加する挙動をシミュレーションで示し、バス型ステーションと路上駐車型のステーションとを比較すると路上駐車型のステーションのほうが 2 倍程度遅れ時間が増加するという評価結果を示している。利便性に関係する指標を導入しシミュレータを用いてステーション配置の検討に用いている点で本研究と共通点があるが、安全性の指標に関しては考慮されておらず、経済性に関しても定性的な分析のみである点が異なる。

文献 [72] は、自律移動モビリティの配車アルゴリズムをマルチエージェント上に実装し、カリフォルニアの街で 5 分以内の待ち時間にするために必要となる車両台数を評価し、ライドシェアのほうが少ない台数で需要をカバーでき、走行距離も 26%少なくなるという分析を行っている。自律移動モビリティ同士の衝突や追従は考慮されておらず、自律移動モビリティの運行の安全性担保に関する考察は含まれていない。

文献 [65] では、マルチモーダル検索結果等の 2 次データを用いて公共交通連携及び補完向けにワンウェイ型カーシェアリングを活用するためのステーション探索方法として、駅からの距離だけなく、ステーション候補間のトリップ量について、スマートフォン位置情報から移動量を推定する手法を採用し、利用頻度が高くなると予測できるステーション位置を探索する方法を提案している。位置情報に関して実際の人の移動データを用いる点や利便性の指標としてステーション配置の評価に移動時間の差分を含めた指標を用いる点で類似しているが、同文献は人間が運転する一般のカーシェアリングを対象といるのに対して、本研究では、自律移動モビリティの運行を対象としており、安全性も評価指標に含めている点が異なる。

文献 [50] では、現実世界のトラフィックフロー最適化問題の一部の解決に量子アニーリングを適用した事例が報告されている。北京の空港と中心街を結ぶ区画にある 418 台のタクシーに限定し渋滞の解消を最適化問題に帰着させて、それを量子アニーリングマシンである D-Wave Systems を用いて計算時間 22 秒で解かせたとの報告がある。本研究では、リアルタイムの交通量最適化ではなく、最適なモビリティサービスを複数の意思決定者が概ね 1 週間程度以内で意思決定を行うことを支援することを目標としている。

d'Orey らは、タクシーのドアツードアのシェアリングサービスを実際の街に適用したシミュレーションを実施し、現在のタクシー運行モデルと比較して、タクシー相乗りシステムの完全導入により、走行キロメートルあたりの平均乗車率が 48% 増加するという結果を示している [31]。

文献 [44] では、急加速や急ハンドルが起きているなど事故につながるような複合的で重大な問題が発生するシミュレーションシナリオを進化計算を用いながら探索する手法を提案している。事故の発生だけでなく速度超過や急ハンドルが複合的に関わる状況を最も重大と考え、それらの状況に至るシミュレーション設定の探索を繰り返すことで、問題が複合的に起きるようなより重大な状況が発生するシナリオを探索している。特定の危険シーンの抽出に注力しており、モビリティサービス全体の利便性や経済性を評価するものではない。

2.3.1 自律移動モビリティの動作のシミュレーション

Terje Kristensen らは、マルチエージェント システム開発フレームワーク (JADE [25]) とエージェントベースの交通シミュレーター (SUMO [30]) を組み合わせ、交差点制御の方式が緊急車両やその他の車両の移動時間の変化を分析し、インテリジェント交通制御を行うことで、緊急車両の交差点の待ち時間を 96%削減できることを示した [42]。

Seongjin Choi らは、自律移動モビリティである自動運転車両が車線変更の際に採用するアルゴリズムを総遅延時間の短縮を評価値とした遺伝アルゴリズムで探索し、微視的交通シミュレータを用いた評価を行っている [28]。

Zhang Chi らは、自律移動モビリティが各種センサーを用いて認知・判断・操作を行う状況を試験する交通シーンシミュレータとして、RoadView [61] を提案している。

Calò, Alessandro らは、シミュレータを用いて、自律移動モビリティが衝突してしまうようなシーンの生成と回避する代替構成を探索的に同時に生成する方法を提案し、衝突を回避できる代替構成が存在しない重大な衝突が見つかる可能性を見つけている [26]。類似するシーン生成に関する研究としては、車線を守るというルールの逸脱に着目して自動運転ソフトを評価したもの [37] や、信号を厳格に守るという機能と他の機能との相互作用に着目したもの [19] がある。

これらの研究は、サイバー空間を用いて主に自動運転を制御するソフトを評価することを目的として行われており、特定のモビリティサービス全体の評価を目指すものではない。

2.4 段階的評価

輸送ルートと頻度の設定を同時に決定する設計問題に関して、文献 [59] では、上位レベルの問題として、乗客の乗り換え回数を最小に抑える混合整数非線形プログラムとして定式化し、下位レベルの問題は、容量制限のある交通機関の割り当て問題として定義し、人口蜂コロニー（ABC）アルゴリズムを上位問題に適用する提案を行っている。すべてを考慮した最適解を網羅的なシミュレーションで求めると計算時間が爆発的に増加するため、上位問題と下位問題を分けて定義し上位の解決に ABC アルゴリズムを用いて最適解を求めている。下降方向探索法は、下位問題の最適性条件を分析し、下位目的関数と上位目的関数の関係を利用している。膨大な解空間の探索を行う点で本研究と類似しているが、下位問題の定義に車両と人の接近や衝突といった安全性の指標を考慮していない点、本研究は上位問題と下位問題を合わせたサービスモデル全体の評価に Human in the Loop のアプローチを適用している点が異なる。

文献 [57] は、複数のインタラクティブな探索ベースのソフトウエア工学の研究論文を調査し、インタラクティブの頻度、提示するソリューションの数、ユーザに選択させる選択肢の種別の統計値を示し体系的な分類を示した調査論文である。調査対象の論文のうち、92%の論文が膨大なシミュレーションパラメータの組み合わせから、一定数の繰り返しのたびに実行結果をユーザに提示し、42%の論文が探索の方向性をユーザに選択させる方法を採用しているとの報告が含まれている。本研究で扱う Human in the Loop のサービス設計・評価手法は、一定数の繰り返しのたびに探索の方向性をユーザに選択させるインタラクティブな探索ベースの手法である。

文献 [22] では、ソフトウェアの次期リリースの最適解を決定する過程において、Human in the Loop でインタラクティブに探索の方向を埋め込む手法において、意思決定者の潜在的な好みを引き出す利点についても指摘している。本研究では、モビリティサービスを導入する意思決定者が、膨大なサービスの候補から最適解を探索する際に、自らが好む方向を選択し探索することで、最適解が妥当と納得できるものとなるか、をアンケートで確認する。

多目的最適化において、パレートフロント解の探索や絞り込みを効率的に行う研究がある。文献 [23] は、6 章で説明する自動配送ロボットを用いた配送サービスの評価を題材に、最適解の探索として投入するロボット台数と稼働時間に着目し、探索アルゴリズムとして NSGA-II [21] を使用するインクリメンタルな探索手法を提案している。同様に、文献 [60] では、パレートフロントの探索において、バラツキを考慮してパレートフロント解を絞り込む手法を提案している。具体的には、配送リクエストパターンやシミュレーションのランダム性に応じて評価指標の値がばらつくことを考慮し、正規分布となることを仮定し、枝刈りする手法を提案している。これらの研究は本研究の派生研究である。

2.5 形式手法と応用例

形式手法はもともとソフトウェア工学で発展してきた理論で、数理論理学に基づいて設計仕様を形式的に記述することで、システムの安全性や品質を数理的に保証できる。

2.5.1 線形時相論理を用いたモデル検査

非決定性有限オートマトンの一種であるクリプキ構造を用いて、あるシステムの振る舞いを表現し、そのモデルを検証することができる。グラフ構造のノードがシステムの到達可能状態を表し、エッジは遷移状態を表す。ラベル関数はそれぞれのノードを対応する遷移状態一組のプロパティに変換する役割を持つ。つまり、クリプキ構造（KS）K は、$\langle S, I, P, L, T \rangle$の 5 つ組である。$S$ は、状態の集合であり、$I \subseteq S$ は、初期状態であり、$P \subseteq U$ は、原子状態命題の集合である。ここで、U は、シンボルのユニバーサル集合を指す。L は、$S \rightarrow 2^P$ であるラベル付け関数、$T \subseteq S \times S$ は、完全 2 項関係である。要素 (s, s') $\in$ T は、$s \rightarrow s'$ と書かれることがあり、遷移と呼ばれる。形式的な仕様記述方法の一つに、線形時相論理 (LTL:Linear Temporal Logic) がある。線形時相論理では、◊（Eventually; 将来的にある性質が成り立つ）や □（Always; 常にある性質が成り立つ）といった時相演算子を用いることで、システムの振る舞いの時間経過に関する性質を表現できる。K のための線形時相論理の文法 ϕ は、次のように定義される。
$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \bigcirc \varphi \mid \varphi \,\mathcal{U}\, \varphi$
ここで、$p \in P$ である。$\mathcal{F}$ を K の中の全ての式 (Formula) の集合とする。$s_i \in S$ であり、$(s_i, s_{i+1}) \in T$ であるとし、実行列 π を $s_0, s_1, \ldots, s_i, s_{i+1}, \ldots$ とする。$\mathcal{P}$ を全ての実行列の集合とする。実行列 π が $\pi(0) \in I$ である計算到達列とし、$\mathcal{C}$ をすべての計算到達列の集合とする。π_i を $s_i, s_{i+1}, \ldots$ とし、π(i) を s_i とする。π が K の LTL 式 ϕ を満たすことを $K,\pi \vDash \varphi$ で表し帰納的に次のように定義する。ここで π^i は、$s_i, \ldots$、すなわち π から最初の i の要素を取り除いたものとする。

- $K,\pi \vDash \top$
- $K,\pi \vDash p$ if and only if (iff) $p \in L(\pi(0))$
- $K,\pi \vDash \neg\varphi_1$ iff $K,\pi \nvDash \varphi_1$
- $K,\pi \vDash \varphi_1 \vee \varphi_2$ iff $K,\pi \vDash \varphi_1$ or $K,\pi \vDash \varphi_2$
- $K,\pi \vDash \bigcirc \varphi_1$ iff $K,\pi^1 \vDash \varphi_1$
- $K,\pi \vDash \varphi_1 \,\mathcal{U}\, \varphi_2$ iff there exists a natural number i such that $K,\pi^i \vDash \varphi_2$ and for all natural numbers $j < i$ $K,\pi^j \vDash \varphi_1$

ここで、φ_1 と φ_2 は、LTL 式である。K の計算可能列の集合 $\mathcal{C}$ すべて要素 π について K,$\pi \vDash \varphi$ であるとき、$K \vDash \varphi$ とする。時相演算子 $\bigcirc$ と $\mathcal{U}$ は、それぞれ Next 演算子、Until 演算子と呼ばれる。これ以外の論理演算子や時相演算子は、通例通りに定義される。つまり、 以下の通り定義する。

- $\bot \triangleq \neg\top$,
- $\varphi_1 \vee \varphi_2 \triangleq \neg(\neg\varphi_1 \wedge \neg\varphi_2)$,
- $\varphi_1 \Rightarrow \varphi_2 \triangleq \neg\varphi_1 \vee \varphi_2$,
- $\Diamond\varphi \triangleq \top\mathcal{U}\varphi$,
- $\Box\varphi \triangleq \neg(\Diamond\neg\varphi)$
- $\varphi_1 \rightsquigarrow \varphi_2 \triangleq \Box(\varphi_1 \Rightarrow \Diamond\varphi_2)$

時相演算子 $\Diamond, \Box, \rightsquigarrow$ は、それぞれ、Eventually 演算子、Globally 演算子、Leads to 演算子と呼ばれる。

形式記述すべきシステム仕様の中には、状態遷移を参照する必要があるものがある。システムが特定のリソースをあるタスクに割り当てる際の仕様記述に必要となる公平性の仮定が代表的なものである [52]。

ラベル付きクリプキ構造（LKS）[27] lK は、状態遷移の参照を可能とすることができる構造である。ラベル付きクリプキ構造 lK は、6 組 $\langle lS, lI, lE, lP, lL, lT\rangle$ である。ここで、lT は状態の集合である。$lI \subseteq lS$ は初期状態の集合、$lE \subseteq U$ は、イベントの集合、$lP \subseteq U$ は、$lP \cap lE = \emptyset$ である原始状態命題である。lL は、その型が $lS \to 2^{lP}$ であるラベル付け関数であり、$lT \subseteq lS \times lE \times lS$ は、全 3 項関係である。イベント $(s, e, s') \in lT$ は、$s \to_e s'$ と記述でき、e でラベル付けされた遷移と言及されることがある。（もしくは単にラベル付き遷移 e あるいは、単に遷移 e と呼ばれる。）すべての $s, s' \in lS$ について $(s, \iota, s') \notin lT$ であるようなイベント $\iota \in lE$ が存在する。

lK に対する状態/イベント-ベースの線形時相論理 (SE-LTL) における式 (Formula) の文法は、線形時相論理のものを引き継ぎ、イベント $e \in lE$ を有するとする。つまり任意の LKS lK について、lK の SE-LTL における式 $l\varphi$ の文法は次の通りである。

$l\varphi ::= \top \mid p \mid e \mid \neg \mathrm{l}\varphi \mid \mathrm{l}\varphi \wedge \mathrm{l}\varphi \mid \bigcirc\, \mathrm{l}\varphi \mid \mathrm{l}\varphi\, \mathcal{U}\, \mathrm{l}\varphi$

ここで、$p \in lP$ であり、$l\mathcal{F}$ を lK に対する SE-LTL の全ての式の集合とする。

任意の LKS lK に対して、任意の実行列 $l\pi \in \mathcal{P}$, lK の SE-LTL の式 $l\varphi \in l\mathcal{F}$ について、$lK, l\pi \vDash \varphi$ は、帰納的に次のように定義される。

- もし、$l\varphi$ が $\top$ なら、$lK, l\pi \vDash \top$,
- もし、$l\varphi$ が $p \in lP$ で、$p \in lL(2^{\bullet} l\pi(0))$ であるとき、$lK, l\pi \vDash p$,
- もし、$l\varphi$ が $e \in lE$ で、$e = 1^{\bullet} l\pi(0)$ であるとき、$lK, l\pi \vDash e$,
- もし、$l\varphi$ が $\neg l\varphi_1$ で、$lK, l\pi \nvDash l\varphi_1$ であるとき、$lK, l\pi \vDash \neg \mathrm{l}\varphi_1$,
- もし、$l\varphi$ が $l\varphi_1 \wedge l\varphi_2$ で、$lK, l\pi \vDash l\varphi_1$ かつ $lK, l\pi \vDash l\varphi_2$ であるなら、$lK, l\pi \vDash l\varphi_1 \wedge l\varphi_2$,
- もし、$l\varphi$ が $\bigcirc\, l\varphi_1$ で、$lK, l\pi^1 \vDash l\varphi_1$ であるなら、$lK, l\pi \vDash \bigcirc l\varphi_1$,

- もし、$l\varphi$ が $l\varphi_1 \mathcal{U} l\varphi_2$ で、自然数 j ¡ i である、任意の j について $lK, \pi^j \vDash l\varphi_1$ であり、$lK, l\pi^i \vDash l\varphi_2$ である i が存在するとき、$lK, l\pi \vDash l\varphi_1 \mathcal{U} l\varphi_2$

ここで、$l\varphi_1$ と $l\varphi_2$ は、SE-LTL の式である。lK の各 $l\pi \in lC$ について、$lK, l\pi \vDash l\varphi$ であるなら、$lK \vDash l\varphi$ である。$lK, l\pi \vDash e$ は、ラベル付き遷移 e が、lK に関して、状態 $2\bullet l\pi(0)$ に適用されることを意味する。

各イベント e ∈ lE に対して、$(s, e, s') \in lT$, $s' \in lS$ の場合に限り、各 $s \in lS$ 状態に対して原子状態命題 enabled(e) $\in lL(s)$ が成り立つとする（enabel(e) の定義）。ラベル付き遷移 e は、enable(e) $\in lL(s)$ の時にのみ適用可能となる。

SE-LTL では、便利な公平性の仮定を表現することが可能である。ラベル付き遷移 e の 2 つの公平性を定義する。

弱い公平性（Week fairness） $(\Diamond \Box \text{enabled}(e)) \Rightarrow (\Box \Diamond e)$

強い公平性（Strong fairness） $(\Box \Diamond \text{enabled}(e)) \Rightarrow (\Box \Diamond e)$

これら 2 つの式を、それぞれ WF（e)、SF(e) と言及する。

$S_{\text{ees}} = \{(e, s) \mid e \in lE, s \in lS\}$ (すなわち $lE \times lS$), $I_{\text{ees}} = \{(\iota, s) \mid s \in lI\}$, $P_{\text{ees}} = lP \cup lE$, $e \in lE$ と $s \in lS$ のそれぞれについて、$L_{\text{ees}}((e, s)) = \{e\} \cup lL(s)$ とし、$T_{\text{ees}} = \{((e, s), (e', s')) \mid e, e' \in lE, s, s' \in lS, (s, e', s') \in lT\}$ であるとき、状態にイベントを埋め込んだクリプキ構造（events-embedded-in-states KS (EES-KS)）K_{ees} は $\langle S_{\text{ees}} I_{\text{ees}}, P_{\text{ees}}, L_{\text{ees}}, T_{\text{ees}} \rangle$ の 5 つ組である。

K_{ees} の状態は、lK のイベントと状態で構成される。K_{ees} のパスは、lK の無限シーケンスは $lE \times lS$ であり、K_{ees} の計算可能列である。lK のイベントは、K_{ees} の原始状態命題となる。

ラベル付きクリプキ構造 (LKS) と状態にイベントを埋め込んだクリプキ構造（ESS-KS）の間では、性質が保存される定理がある [53]。任意のラベル付きクリプキ構造 lK について、K_{ees} を状態にイベントを埋め込んだクリプキ構造 lK とすると、lK の任意の状態埋め込み時相論理式 $l\varphi$、同時に K_{ees} に対する任意の時相論理式について、$lK \models l\varphi$ であるとき、$K_{\text{ees}} \models l\varphi$ が成立する。この定理により、計算機上で利用可能でモデル検査の機能を有する仕様記述言語、例えば線形時相論理のモデル検査が可能な Maude [29] が利用可能となる。

本研究では、分析対象のシステムの挙動について、状態にイベントを埋め込んだクリプキ構造の記述、つまり、状態埋込時相論理式（SE-LTL formula）として、Maude 上で記述することで Maude のモデル検査機能を利用する。

2.5.2 線形時相論理を用いたモデル検査の事例

自律移動モビリティの形式検証に関する第 1 回ワークショップが、2017 年 9 月にイタリアのトリノで開催され、そのワークショップで発表された論文の一つで、自律移動モビリティ（AV）の行動計画の実装と形式検証に関して報告している [35]。著者らは、障害物回避、障害物選択（衝突が避けられない場合）車両回避といった行動計画に興味を持っている。この研究では、Gwendolen エージェントプログラミング言語で知的で合理的な車両の行動計画を実装している。これらの行動計画は、Java で実装された模擬環境でシミュレーションできる。検証が必要な性質を LTL で表現し、MCAPL(Model Checking Agent Programming Languages) フレームワーク内で実行される JPF(Java PathFinder) の拡張である AJPF(Agent Java PathFinde) 検証ツールを用いて、行動計画のモデル検査が実行されている。

Mitsch らは、地上を自律走行するロボットを対象に同様の研究を行っている [47]。定理照明器 KeYmaera X [36] を用いて、静止した障害物との衝突の分析に加えて、移動する障害物や動く障害物との衝突を考慮して、衝突回避機能で回避できるための安全距離やロボットに搭載されているセンサーの不完全な検知範囲を考慮した分析を実施し、衝突しない制約を規定している。本研究では、モビリティサービスを提供するシステム全体の挙動のモデル検査と評価指標の定義に形式記述を利用する。

2.5.3 信号時相論理

信号時相論理（Signal Temporal Logic: STL）[45] は、動的なシステムの状態軌道等の「信号」に対して、時間制約を含む複雑な仕様を柔軟かつコンパクトに記述することができる形式手法の一つである。STL は、LTL と比較して過去の状態についての情報を持つため、実世界の制約をより上手く表現できることが期待されており、次節で示すような分析事例が出てきている。

構文的には、STL の数式の集合 Fml は次のように定義する。

$Fml \ni \varphi ::= \top \mid \bot \mid f(\vec{x}) > 0 \mid \neg\varphi \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathcal{G}_{\mathcal{I}}\varphi \mid \mathcal{F}_{\mathcal{I}}\varphi$

ここで、f は n 項関数で、$\vec{x}$ はシンボルの組であり、$\mathcal{I}$ は、実数で示される経過時間の区間 ($\mathcal{I} \geq 0$) である。論理結合子 $\neg$ や$\vee$ は、標準的な意味合い（すなわち、”not”や”or”）で用いられている。時相演算子の $\mathcal{G}_{\mathcal{J}}\varphi$ と $\mathcal{F}_{\mathcal{J}}\varphi$ 時間の経過中のある性質の真偽を表現する。$\mathcal{G}_{\mathcal{J}}\varphi$ は「時間区間 $\mathcal{J}$ の間、φ が真である」ということを意味する（Globally）。$\mathcal{F}_{\mathcal{J}}\varphi$ は、「時間区間 $\mathcal{J}$ の間に、いつか φ が真となる」ことを意味する。双方とも時間区間 $\mathcal{J}$ の間に関しての記述である。

STL 式は、時間に伴い変化する信号の様々な性質を記述することができる。例えば、STL 式 $\mathcal{G}_{[0,\infty]}(x - 300 > 0 \rightarrow \mathcal{F}_{[0,3]}(10 - y > 0))$ は、「x が 300 より大きくなったとき、いつでも 3 秒以内に y が 10 より小さくなる」ということを意味する。

2.5.4 信号時相論理を用いた分析の事例

Raman らは、STL を用いて要求仕様を記述することで、離散時間サイバーフィジカルシステムのモデル予測制御 (Model predictive control) のため安全性等の性質を記述する方法を提案し、建物のエネルギーと屋内気候制御の生成への実験的な適用結果を報告している [56]。

金島らは、信号時相論理式を用いて、online pickup and delivery 問題の制約を表現し、最適な配送スケジュールを求める方法を提案している [64]。配達時間短縮といった問題の制約を信号時相論理で表現し最適化問題として定式化している。本研究では時間的かつ空間的な評価指標を定式化する際に信号時相論理による記述を導入しており、時間的かつ空間的な制約を超えた解も含めた最適解をインタラクティブに求める点が異なる。

佐藤らは、時相論理仕様に対する反例を見つける最適化ベースの方法を産業用ガスタービンシステムのパラメータ合成問題に適用し、複雑な要件を信号時相論理式で表現した上で，要件を満足しない入力パターンや運転条件をシミュレータと最適化ソルバーにより探索している [58]。「しきい値を超える量も時間も短い方がよい」といった量と時間の両方に関する要件を満たす、ガスタービンの負荷遮断時における燃料制御パラメータを探索している。具体的には、システムが要求仕様をどのぐらい違反しているか実数値で表す関数（目的関数）を定義し、目的関数の値がより小さくなる方向にパラメータを繰り返し修正する手順で最適解を探索している。本研究では、モビリティサービスの評価に量と時間の両方に関する要件を満たす指標を記述する際に信号時相論

理を用いて、意思決定者の間での指標の正確な理解を得たうえでインターラクティブに最適解を探索してる。

第3章　提案手法

本研究で提案するモビリティサービスの設計・評価手法について説明する。

3.1　想定する適用範囲

本研究では、新たなモビリティサービスを特定の地域に導入・改善することを想定する。特に人々が行う生活圏に小型自律電動モビリティを用いたサービスを導入することを想定する。小型自律電動モビリティに着目する理由は、1.3 で説明した通り、時代の要請にあう新たなサービスを提供する可能性を持っているためである。

3.1.1　既存手法との想定の差分

既存手法である費用便益分析と提案する手法の想定の差を表 3.1 で説明する。道路の設計・評価を中心とする交通計画に用いている費用便益分析が想定している車両は自動車・トラック・バス等であり特性や性能は既知であり人の歩行とは分離された道路のレーンという固定軌道を走行することを想定している。本研究で想定する小型自律移動モビリティは様々な特性が想定され、自由軌道を走行し人混在環境での運用が想定されている。自動車の運行や運転に関するルールは既に定められており信号・踏切等のインフラの役割も定まっている。新たなモビリティである小型自律移動モビリティについては住宅地等の人混在の環境での運用が期待されており、ルールやインフラの役割もこれから規定されていく。

表 3.1: 提案手法の特徴

	車両・機体	軌道	環境	ルール	インフラ
既存手法	既知	固定軌道	道路 人分離	既に策定	既知 信号・踏切等
提案手法	未知 重量・加減速性能 積載量・最高速度	自由軌道	道路＋歩道 人混在	新たに策定	新たに検討

3.2　提案する設計・評価手法

図 3.1 にしたがって提案手法を説明する。

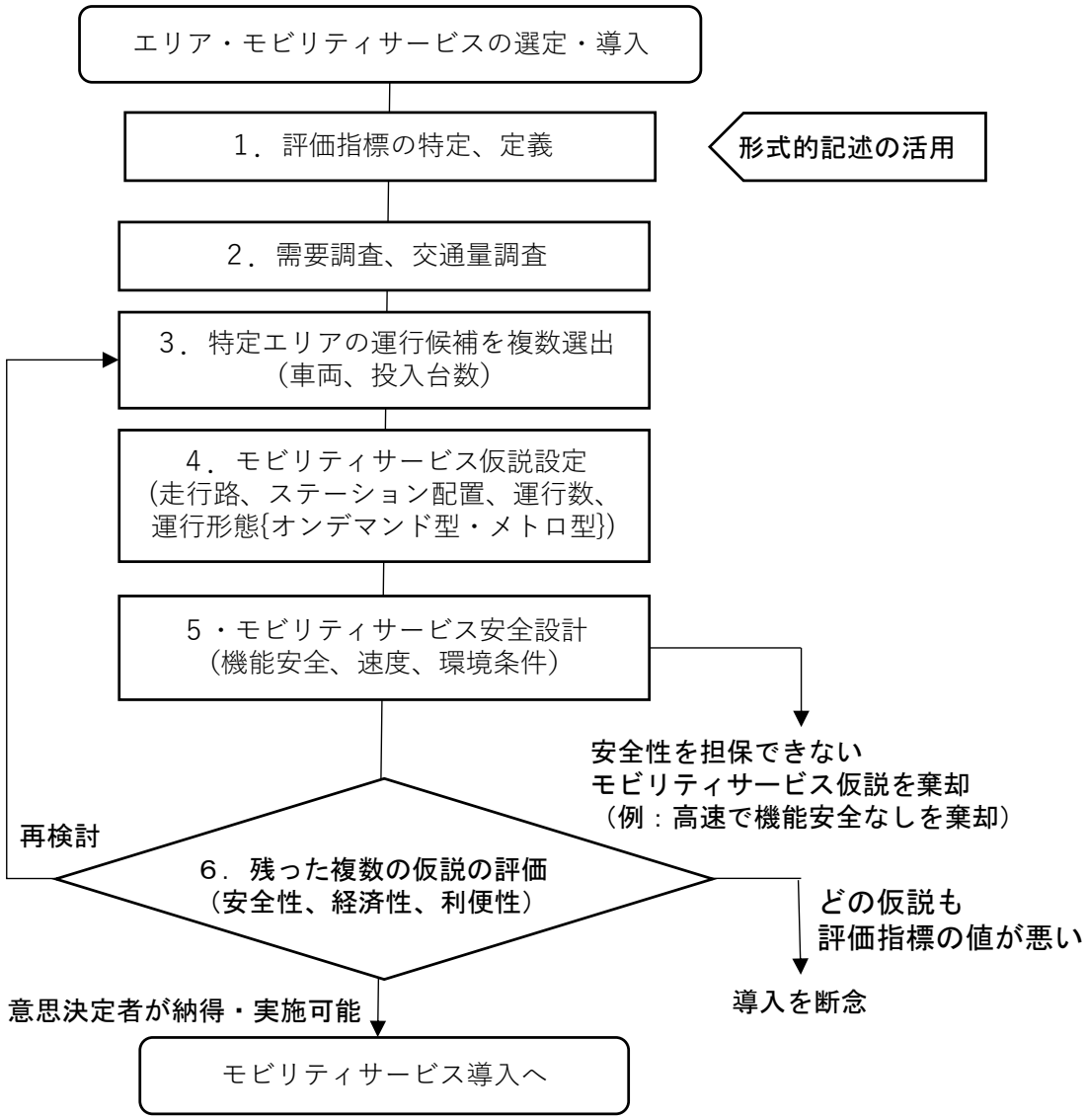

図 3.1: 提案する設計・評価の流れ

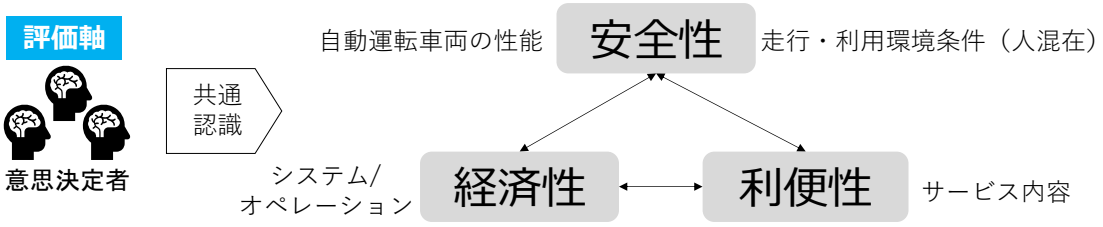

図 3.2: 意思決定で注目する評価指標

3.2.1 評価指標の特定、定義

新たなモビリティサービスを特定の地域（エリア）に導入する場合、要求される事項を指標化する。小型自律移動モビリティは、自由軌道を走行し、住宅地や歩道等を含め人混在環境を走行することが想定されるため、モビリティサービスの設計・評価にあたっては、単純な費用便益の観点のみならず安全性の観点が重要となる。すなわち、安全性、費用、便益に関する分析が重要となる。本研究では、安全性を担保した上での人等との衝突リスク (Risk)、費用は導入・運用コスト (Cost)、利便性に代表される便益 (Value) の均衡点に着目し、RCV 分析と呼ぶことがある。モビリティサービスの設計・評価においては、モビリティサービスの導入・改善の意思決定を行うステークホルダの間でこれらの指標による評価が共有されることが重要となる（図 3.2 参照）。

モビリティサービスを提供するシステムの挙動やその影響を評価する指標の定義では、曖昧性を排除することが望ましい。複数のステークホルダ間で共通理解を得るため形式的記述を活用する。

3.2.2 需要調査、交通量調査

モビリティサービスを提供する地域（エリア）での需要の幅や他の交通参加者の交通量を調査する。需要調査では、当該エリアでの住民の数を考慮してできるだけ正確に人やモノの移動の需要量を見積もる。交通量調査では、時間帯別の車や人の通行量を計測するといった方法によって交通量を得る。

3.2.3 特定エリアの運行候補を複数選出

当該エリアでサービスを提供する車両・機体の最大乗車人数、最大積載量、最高速度といた機能・性能や投入台数といった運行候補を複数パターン選定する。

3.2.4 モビリティサービス仮説設定

ステーション配置、走行路、運行数、運行形態といった導入するモビリティサービスの仮説を設定する。

3.2.5 モビリティサービス安全設計

これらのサービスを安全に実現するための安全設計を行う。この過程で不安全な運行を排除する。例えば、機能安全設計が不十分な大型車両を歩車分離のない住宅地で降雪時に高速運行させ

るといったサービス仮説はこの工程で棄却する。

3.2.6　複数の仮説の評価

主に安全性、利便性、経済性の観点から計算機シミュレーションを用いた定量評価を行い、どの仮説が妥当か判定し、モビリティサービス導入・改善の意思決定を行う。この時、新たな視点での評価が必要であることに気づいたり、妥当なモビリティサービスの仮説が存在しない場合、仮説の生成に戻って再評価を行う。つまり、意思決定に関わる複数のステークホルダは、RCV 分析による評価で共通認識をもって、モビリティサービスの導入・改善を決定する。

3.3　提案手法の利点

これまで広く用いられてきた費用便益分析と比較して、提案方式の利点は、以下の通りである。

- 安全性を含む多様な指標で評価できる
- 多様な指標であっても正確に意思決定者の間で理解できる
- 得られた気づきを反映した再評価を同じ枠組みで実行できる

第4章　形式検証（モデル検査）の適用による活性特性の評価

本章では、図 3.1 で説明した提案フローにおける最初の段階で行う「評価指標の特定、定義」の段階において、補助的に実施する形式検証について説明する。2.5 節で説明した線形時相論理を用いたモデル検査の一例であり、これから設計・導入しようとするモビリティサービスの活性特性が担保できているか確認するために実施する。本章の内容は文献 [49] として発表したものと同等の内容である。

4.1　導入

ライドシェアリングサービスは市場に登場しており、数多くのサービスが提供されている。加えて、自動運転車両（ADV: autonomous driving vechicle）が乗客を迎えに行き目的地まで送り届けるサービスが実現される新しい世界がやってこようとしている [51]。ソフトウエアにより完全自動運転される車両の制御は、非常に複雑なフォールトトレラントなリアルタイム分散システムによって実現される。本章では、そのようなシステムを自動運転ライドシェアシステムと呼び、目的とする性質を満たすか形式的手法で検証する。

このようなシステムの形式検証については、これまでもいくつかの研究が行われてきたが、システムの品質が我々の将来の生活に大きく影響する可能性が高く、より信頼性の高いシステムであるかどうか、経験と知見を蓄積する必要があり、さらなる研究の推進と知見の獲得が必要である。本章では、簡素なライドシェアリングシステムの形式仕様を作成し、システムがいくつかの望ましい特性を享受できることをモデル検査した事例について報告する。「簡素な」とは、検証対象とするシステムがフォールトトレラントやリアルタイムの能力を含まないことを意味する。システムによって提供されるサービスが利用できる小さな地図のみを考慮する。このようなシステムの形式検証に関する研究は、まだ常熟していないため、適度に簡素化した設定から始めるのが合理的である。

仕様記述言語として、Maude [29] を用い、Maude の LTL モデル検証器を形式検証のために用いる。Maude は、モデル検査の対象となるシステム仕様において帰納的に定義されたデータ構造や結合律（associativity）と交換律（commutativity）を満たす 2 項演算子の使用が認められるといった記述力に優れた特徴を有す。後者によりものの並び順が等しさに影響しない多重集合のようなもののあつまりを記述することが可能となる。そのようなもののあつまりをスープと呼ぶ。スープを扱えることで分散システムの局所的な状態変化（状態遷移）を簡潔に記述することが可能になる。Maude の記述能力は、サービスが提供される地図（あるいはエリア）を表現する際にも便利である。例えば、地図を変えるたびに新たな形式仕様を定義しなおすといったことを避けることができる。つまり、形式記述による表現において、地図の定義をパラメータとして定義できる。車の集合や乗車希望する人の集合を地図と同様にパラメータとして扱える形で定義する。このような手法で形式仕様を作成することで、いくつかの重要なデータをパラメータ化して記述で

きることを示すことが本研究の貢献の一つである。つまり、Maude を用いることで、このような一般的な形式仕様を作成することが可能であることを本章で示す。システムの活性特性（Liveness property）と安全性特性（Safety property）をモデル検査する。活性特性のモデル検査では、しばしば公平性の仮定を置くことが必要となる。これは、モデル検査の性能を劇的に悪化させたり、時には実行不能とする。本章では、分割統治（divided and conquer）アプローチ [53] を活性特性のモデル検査に適用する。これにより公平性の仮定を有する活性特性の検証が現実的に実行可能であることを示す。これも本研究の貢献の一つである。

4.1.1 システム状態の表現

システムの状態を表現する方法は複数存在する。本章では、システムの状態を名前と値の組のスープで表現する。名前はパラメータを含むことがある。この名前と値の組のことを観測可能コンポーネントと呼ぶ。すなわちシステムの状態は観測可能コンポーネントのスープで表現する。並置 2 項演算子を、スープの構成子として使用する。oc_1, oc_2, oc_3 を観測可能コンポーネントとすると、$oc_1\ oc_2\ oc_3$ でこれら 3 つの観測可能コンポーネントから構成されるスープを表す。ここで並置 2 項演算子は結合律と交換律を満たすため観測可能コンポーネントの並びの順番はスープの等しさとは無関係であることに注意されたい。

4.1.2 分割統治（divided and conquer）アプローチ

ある LKS lK の EES-KS K_{ees} と線形時相論理式 $(f_1 \wedge \ldots \wedge f_n) \Rightarrow \varphi$ を考える。ここで、各 $i = 1, \ldots, n$ に対し、f_i を公平性の仮定で、 φ は活性特性である。線形時相論理式の集合をその要素（線形時相論理式）の連言とみなすことにする。例えば、$\{f_1, \ldots, f_n\}$ は $f_1 \wedge \ldots \wedge f_n$ のことであり、$\{f_1, \ldots, f_n\} \Rightarrow \varphi$ は $(f_1 \wedge \ldots \wedge f_n) \Rightarrow \varphi$ のことである。

$qf_j,\ j = 1, 2, \ldots, m$ を線形時相論理式とする。F_1 を $\{f_1, \ldots, f_n\}$ の部分集合とする。$F_{j'+1}$, $j' = 1, 2, \ldots, m-1$ を $\{f_1, \ldots, f_n\} \cup \{qf_1, \ldots, qf_{j'}\}$ の部分集合とする。加えて、各 $j = 1, 2, \ldots, m$ に対し、EES-KS $K_{\text{ees}} \models F_j \Rightarrow qf_j$ が成立することを仮定する。これらの仮定から、 $K_{\text{ees}} \models (f_1 \wedge \ldots \wedge f_n) \Rightarrow (qf_1 \wedge \ldots \wedge qf_m)$ を導くことができる。

QF を $\{f_1, \ldots, f_n\} \cup \{qf_1, \ldots, qf_m\}$ の部分集合とする。すると、$K_{\text{ees}} \models QF \Rightarrow \varphi$ から $K_{\text{ees}} \models (f_1 \wedge \ldots \wedge f_n) \Rightarrow \varphi$ を導くことを可能とする。このため、$(f_1 \wedge \ldots \wedge f_n) \Rightarrow \varphi$ は、$F_j \Rightarrow qf_j$, $j = 1, 2, \ldots, m$ と $QF \Rightarrow \varphi$ に分割させることができる。それから、 $K_{\text{ees}} \models (f_1 \wedge \ldots \wedge f_n) \Rightarrow \varphi$ をモデル検査するには、$K_{\text{ees}} \models F_j \Rightarrow qf_j,\ j = 1, 2, \ldots, m$ と $K_{\text{ees}} \models QF \Rightarrow \varphi$ をモデル検査すれば十分であることになる。$qf_j,\ j = 1, 2, \ldots, m$ を準公平性と呼ぶことにする。このアプローチは公平性の仮定をおいた活性特性のモデル検査を行うための分割統治アプローチと呼ばれる [53]。

本章では、活性特性のモデル検査を現実的な時間で実行するために、このアプローチを採用する。

4.2 ライドシェアシステム

システムの主な構成要素は人と車であるが、システムが提供するサービスが利用できるエリア（または地図）を考慮する必要がある。エリアの中に人や車が点在する。 このようなエリアは二次元の連続空間であるが、本章では有向グラフとして扱う。通常、エリアには駅や病院などの複数のランドマークである目的地があり、これらを有向グラフの節点として扱う。ランドマークまたは

ランドマークの近くに位置する人および／または車は、ランドマークに対応する節点内に位置するものとして扱う。2 つのランドマーク lm_1、lm_2 があったとして、他のランドマークを経由せずに lm_1 から lm_2 への直接ルートがある場合、グラフには、lm_1 から lm_2 にそれぞれ対応する n_1 から n_2 までの辺を設定する。lm_1 から lm_2 への直接ルートが 2 つ以上存在する可能性があるが、n_1 から n_2 への 1 つの辺として扱う。lm_1 から lm_2 への直接ルートがあったとしても、一方通行のため、たとえば、lm_2 から lm_1 に直接行くことが許可されない場合がある。したがって、領域は有向グラフとして扱う。 エリア内には通常は循環するルートがあるため、エリアをモデル化するグラフは非循環とはしない。 (つまり、グラフは 有効非巡回グラフとしない)。実際の地図から有向グラフを取得する方法については説明せず、有向グラフは与えられるとする。実際の地図から有向グラフを取得することは現実のサービスを評価するためには重要であるが、本章ではライドシェアリングシステムを形式的に規定する方法と、システムが望ましい特性を享受しているかどうかをモデル検査する方法に焦点を当てる。ただし、複数のマップ (または有向グラフ) に対して複数の形式仕様を持たせることは望ましくない。次の章では、複数のマップに対しても 1 つの形式仕様を持つだけで十分になる方法について説明する。2 つの異なる節点 n、n' については、n から n' へのルートが常に存在すると仮定する (逆も同様)。これは 1 つのランドマークから n' へのルートが存在しないことは現実世界では考えにくいためである。したがって、n からの最短パスが常に少なくとも 1 つ存在する。n' へ (その逆も同様)。 また、各エッジには同じ重み 1 が与えられると仮定する。

人それぞれの目的地は時々変わるので、出発地と最終目的地も当然異なる。各人には 1 つ以上の中間目的地があり、その人が目的地を訪れる順路は常に同じであり、出発地と最終目的地も常に同じであると仮定する。つまり、各人は常に複数の中間目的地を一定の順序で訪れ、また同じ場所に戻ることを繰り返すとする。人が車で次の目的地 n に到着すると、人はいつもそこで車から降りて、n の次の目的地に移動するために車を届けるようシステムに依頼する。システム内の各人には、中間目的地があり、最終目的地が出発節点と同じであるような、順路の 1 つの長い移動を行うような人が複数存在するとする。

現実世界では 1 台の車に 2 人以上の乗客が乗る可能性があるが、本章では各車に最大 1 人の乗客が乗れるものとする。なお、1 人の乗客は個人のグループとして解釈できる。各車はアイドル状態、運行中、または予約済み状態のいずれかの状態を取る。アイドル状態の車は、予約されるまで現在の節点に留まり続ける。つまり、タクシーの流しの様な動作はしないとする。

アイドル状態の車 c が人 p によって予約されている場合、c の位置 n_1 が p の位置 n_2 と異なる場合には、c は n_2 に移動する。この時 n_1 から n_2 までの最短経路の 1 つを経由する。 c が p に対して運行中の場合、c は n_2 から n_3 までの最短経路の 1 つを通り、p の旅行順路中の次の目的地 n_3 に移動する。

図 4.1 は、2 人の人と 2 台の車を配置し 6 つの節点（ランドマーク）と 10 の辺を持つ地図である。車 1(car1) と 車 2(car2) は、それぞれ節点 D と節点 C に配置されている。人 1(person1) と 人 2(person2) は、それぞれ節点 A と節点 C に配置される。人 1 の旅行順路は C、F、A、人 2 の旅行順路は D、B、C である。図 4.1 に示すグラフを Map6 と呼ぶ。

4.3 形式仕様記述

システムの形式仕様では、マップ (またはエリア) をモデル化する有向グラフが仕様のパラメータとして扱われる。仕様記述で有向グラフをそのまま使用すると、モデル検証実験を行う際に、節点から節点への最短経路をその都度計算する必要があるが、これはモデル検査の性能に重大な悪

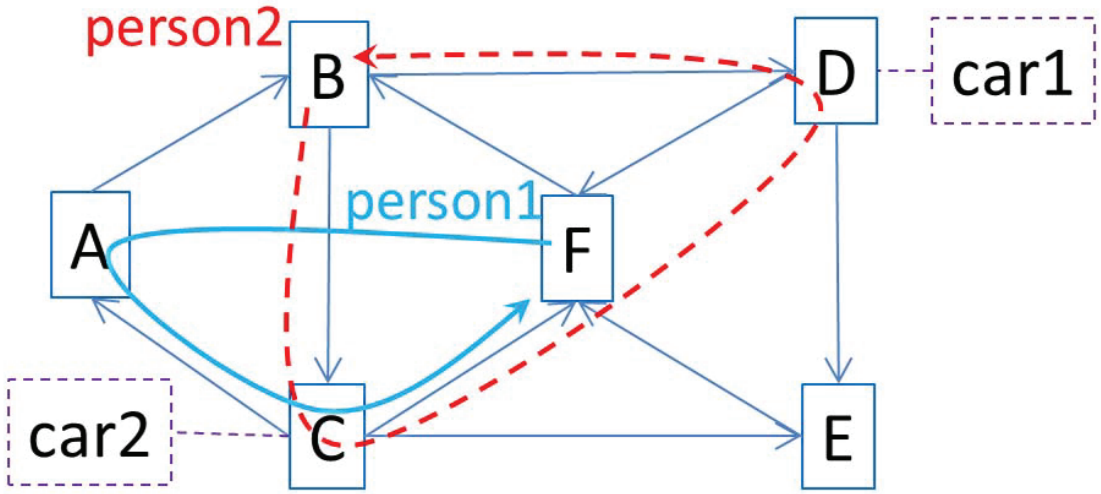

図 4.1: 人 2 名、車 2 台を配置した地図

Fig. 4.1: A map on which there are 2 persons and 2 cars

影響を与えるはずである。このため、ダイクストラのアルゴリズムを使用して、各節点の有向辺 (n, n') ごと、最短経路最短経路を事前計算しておく。前章で説明した仮定により、節点のすべての有向辺 (n, n') には少なくとも 1 つの最短経路があることに注意されたい。

たとえば、Map6 上の節点 C から節点 D までの最短経路の 1 つは、C、A、B、D である。最短経路は次のような項で表される。

```
dbentry(nc,nd,routeentry(nc | na | nb | nd | empty,3) | empty)
```

x = a, b, c, 等について、nx という記述は、節点 X = A, B, C, 等に対する節点の記述を示す。dbentry は、出発地（の節点）n と目的地（の節点）n' と経路上の節点の並びにより表現する。経路上の節点の並び routeentry(sp,l) は、 n から n' への最短経路 sp とその経路の長さ l の組である。本章では、すべての有向グラフの辺 (n, n') に対して、一つの最小経路のみを考慮する。現実的には、エリア Map6 の上でも nc から nd への最小経路は 2 つ存在するが、一つの経路のみを用いる。

有向グラフは、このような dbentry の集合として表現される。スープをそのような集合として使用する。 routedb-m6 を Map6 の dbentries のスープとする。これは次のように定義される。

```
eq routedb-m6 =
 d6nana d6nanb d6nanc d6nand d6nane d6nanf d6nbna d6nbnb d6nbnc
 d6nbnd d6nbne d6nbnf d6ncna d6ncnb d6ncnc d6ncnd d6ncne d6ncnf
 d6ndna d6ndnb d6ndnc d6ndnd d6ndne d6ndnf d6nena d6nenb d6nenc
 d6nend d6nene d6nenf d6nfna d6nfnb d6nfnc d6nfnd d6nfne d6nfnf .
```

ここで並列演算子 __ が結合的で可換的であるスープのコンストラクタとして使用している。各要素は dbentry であり、d6ncna の定義はすでに上で説明したものである。 他の dbentiry も同様に定義される。

人の ID は pid(x) で表される。x は自然数である。特別な人物 ID pidnone を用意している。これは誰でもないことを意味する。個々の人は、pidle、prequest、pwait、pride の 4 つの状

態のいずれかにあると仮定する。システムを使用したくない人、システムを使用したい人、車を待っている人、サービスを受けている人、をそれぞれ意味する。人はそれぞれ、ID、状態、現在位置、次の目的地、移動ルートを持っている。サービスを受けている人には、その人を乗せた車の ID が与えられる。例えば、初期状態の Map6 上の person1 は次の項で表される。

```
person(pid(1),pidle,nf,na,cidnone,(na | nc | nf | empty))
```

person1 の ID は `pid(1)` である。 彼/彼女は最初はアイドル状態、すなわちシステムを使用したくないとする。彼/彼女の現在の位置は `nf` で、 次の目的地を `na` で示されている 。 彼/彼女はサービスを受けていないので、`cidnone` 車 ID として、車が付与されていないことを示す特別な ID が割り当てられている。彼/彼女の旅行経路は `na | nc | nf | emply` であり 、`na`、`nc`、`nf` をこの順で繰り返し訪問することを示している。

車の ID は `cid(`x`)` で表される。x は自然数である。専用の車両 ID `cidnone` を用いている。 これは、すでに見たように、車が割り当てられていないことを示す。各車は、`cidle`、`creserved`、`cservice` の 3 つの状態のいずれかにあると仮定する。これらは、車が予約済みでも運行中でもないアイドル、予約済みである、人を乗せて運行中である、ことを意味する。各車両には ID、状態、現在位置、次の目的地が設定される。 車が運行されると、その車で運行される人の ID が割り当てられる。それ以外の場合、つまり人が割り当てられていない空車状態の際には、 `pidnone` が付与される。たとえば、初期状態の Map6 上の car1 は次の項で表される。

```
car(cid(1),cidle,nd,nd,pidnone)
```

car1 の ID は `cid(1)` である。最初は、アイドル状態である。現在位置は、`nd` であり、次の目的地が `nd` である。これは、人を割り当てる要求もないアイドル状態であり、誰も割り当てられていない `pidnone` が割り当てられている状態であるため、この車両は、`nd` に留まり続けていることを意味する。

システムのそれぞれ状態を表現するため、4 つの観測可能なコンポーネント:`(cars:` cs`)`、`(ps:` ps`)`、`(ds:` $bdes$`)` 、`(tran:` e`)` を用いる。ここで cs は、システムに参加している車両のスープ、ps は、システムに参加している人のスープ、$dbes$ は、システムによりサービスが提供されている場所の地図（あるいはエリア）を有効グラフで表現した dbentry のスープ、 e は、最近に実施されたイベント（あるいは状態遷移）を示す。それぞれの状態は、4 つの観測可能な要素のスープで示される：

```
(cars: cs) (ps: ps) (ds: bdes) (tran: e)
```

状態遷移は書き換えルールの観点で cs、ps、$bdes$ のインスタンスには関係なく記述される。つまり、cs、ps、$bdes$ は形式仕様のパラメータとみなすことができる。システムの初期状態は、Map6 が使われる。 $bdes$ は `routedb-m6` であり、 cs は、car1 と car2 の 2 台分のスープであり、図 4.1 で示される位置に配置される。ps は図 4.1 で示される位置に配置される person1 と pserson2 の 2 名分のスープで示される。e は `notran` であり ι を意味する。`im6c2p2` を Map6 での、この初期状態を示すものとして用いる。

人は自分の状態を `prequest` に変更できる。 状態が `pidle` の場合、現在位置と次の目的地は異なる。これは次の書き換えルールとして規定される。

```
crl [startreq] :
 (ps: (person(PID,pidle,N1,N2,CID,DL) PS)) (tran: T)
  =>
 (ps: (person(PID,prequest,N1,N2,CID,DL) PS)) (tran: startreq)
if not(N1 == N2) .
```

PID や N1 など大文字で書かれているものはすべて Maude の変数である。後継する状態は、観測可能なコンポーネント tran: で、この書き換えルールによってまさに変更され startreq の状態となったことを保持する。

人が prequest の状態である場合、空いている車があれば、その人はそれを予約できる。これは次の書き換えルールとして規定される。

```
crl [book] :
 (ps: (person(PID,prequest,N,N1,cidnone,DL) PS))
 (cars: (car(CID,cidle,N2,N3,pidnone) CS)) (db: RDB)
 (tran: T)
 =>
 (ps: (person(PID,pwait,N,N1,CID,DL) PS))
 (cars: (car(CID,creserved,N2,N,PID) CS)) (db: RDB)
 (tran: book(CID))
if car(CID,cidle,N2,N3,pidnone) \in
   (nearestvacant N of (car(CID,cidle,N2,N3,pidnone) CS) on RDB) .
```

dbes 上の車 c の nearestvacant n は節点 n に最も近い利用可能な空き車のスープを計算する。これは Maude が提供する強力な連想交換パターンマッチング機能が書き換えルールで使用されることで実現できる。観測可能なコンポーネントである cars: において、車のスープの中で空車である任意の車両 car(CID,creserved,N2,N,PID) が選択される。ここで、結合性と可換性のため、要素が列挙される順序は無関係であることに注意いただきたい。この条件では、任意に選択した空き車が、節点 N に最も近い位置にある利用可能な空き車の 1 つであるかどうかをチェックしている。この書き換え規則に当てはまる場合、その人のステータスは pwait に変わる。車はその人のために予約されており、その状態は car(CID,creserved,N2,N,PID) と変更される。後続する状態は、観測可能なコンポーネント tran: は、book(CID) に変更される。CID が book のパラメータとして使われている理由は、車の観点からの公平性の仮定を表現するためである。

車が人によって予約されており、その人がいる節点とは異なる節点に存在する場合、その車は人の位置に近づく。これは、次の書き換えルールとして規定される。

```
crl [rsvmove] :
 (cars: (car(CID,creserved,N,N1,PID) CS) ) (db: RDB) (tran: T)
 =>
 (cars: (car(CID,creserved,nexthop(N,N1,RDB),N1,PID) CS))
 (db: RDB ) (tran: rsvmove(CID))
if not (N == N1) .
```

nexthop(N,N1,RDB) は、N に現在車が居る場合の次の場所を計算する。これは、RDB が、地図上で接続されているすべての目的地に対する最短経路を維持しているので簡単に計算できる。

人が予約した車がその人のいる場所に到着すると、その人が車に乗り込み、車が運行を開始することを、次の書き換えルールとして規定する。

```
rl [ridesvc] :
 (cars: (car(CID, creserved, N, N, PID) CS))
 (ps: (person(PID, pwait, N, N1, CID, DL) PS))
 (tran: T)
 =>
 (cars: (car(CID, cservice, N, N1, PID) CS))
 (ps: (person(PID, pride, N, N1, CID, DL) PS))
 (tran: ridesvc(CID)) .
```

後続する状態では、車の次の目的地は、その人の 目的地 N1 と同じになる。

人が車でサービスを受けている場合、人も車も、目的地までの最短経路にある現在の節点の次の節点に移動する。これは、次の書き換えルールとして規定される。

```
crl [svcmove] :
 (cars: (car(CID,cservice,N,N1,PID) CS))
 (ps: (person(PID,pride,N,N1,CID,DL) PS))
 (db: RDB) (tran: T)
  =>
 (cars: (car(CID,cservice,nexthop(N,N1,RDB),N1,PID) CS))
 (ps: (person(PID,pride,nexthop(N,N1,RDB),N1,CID,DL) PS))
 (db: RDB) (tran: svcmove(CID))
if not (N == N1) .
```

nexthop(N,N1,RDB) は、RDB の中で、現在地から目的地に向けた最短経路上の次の節点を計算する。

車でサービスを受けている人が目的地に到着した場合、その人はアイドル状態となり車から降りる。これは、次の 2 つの書き換えルールとして規定する。

```
crl [getoff] :
 (cars: (car(CID,cservice,N,N,PID) CS))
 (ps: (person(PID,pride,N,N,CID,DL) PS))
 (tran: T)
 =>
 (cars: (car(CID,cidle,N,N,pidnone) CS))
 (ps: (person(PID,pidle,N,nextdest(DL,N),cidnone,DL) PS))
```

```
 (tran: getoff(CID))
if not (nextdest(DL,N) == emptynode) .

crl [getoff2] :
 (cars: (car(CID,cservice,N,N,PID) CS))
 (ps: (person(PID,pride,N,N,CID,DL) PS))
 (tran: T)
 =>
 (cars: (car(CID,cidle,N,N,pidnone) CS))
 (ps: (person(PID,pidle,N,top(DL),cidnone,DL) PS))
 (tran: getoff2(CID))
if (nextdest(DL,N) == emptynode) .
```

最初の書き換えルール `getoff` は、現在の節点がその人の最終目的地の移動ルートの途上にあり最後の節点ではない場合を扱う。 (`nextdest(DL,N)`) は、 旅行巡回先の次の目的地を表す。2 番目の書き換えルール `getoff2` は、現在の節点が人の旅行巡回先 `DL` の最後の節点である場合を扱う。 この場合は、`nextdest(DL,N)` `emptynode` を返す。つまり、旅行巡回先にはもう目的地がなく、次の旅行の目的地 `top(DL)` として、人が巡回する目的地のリストの最初の目的地を次の目的地として使用する。

これらは、EES-KS $K^{\mathrm{rsc}}_{\mathrm{ees}}$ の一部として、 S_{ees}、I_{ees}、T_{ees} としてシステムを形式化したものとなる。これらは、Maude の書き換え理論仕様として規定されている。P_{ees} と L_{ees} の部分については、次の章で説明する。

4.4 モデル検査

Map6 を含む複数のマップに対してシステムが享受すべき望ましい特性として、いくつかの安全性特性と活性特性をモデル検査した。本章では、それらの中から 2 つの安全性特性と 2 つの活性特性を取り上げる。2 つの安全特性の非形式的な説明は次の通りである。

- ダンブルブッキングがない。No Double Booking (NoDB)
- サービスを提供しない車には乗客が乗っていない。Idle Car has no Person (ICnoP)

2 つの活性特性の非形式的な説明は次の通りである。

- 要求した人はやがて乗車できる。Request Person will eventually Ride (RPweR)
- 空車はやがてサービスを提供する。Vacant Car will eventually Serve (VCweS)

2 つの安全性特性 NoDB と ICnoP を形式的に規定するため、3 つの原始命題、 `isidle(`*cid*`)`、`hasperson(`*cid*`)` 、`doublebook(`*cid*`)` をそれぞれの車 IDcar ID *cid* に対して定義する。

```
ops isidle hasperson doublebook : Cid -> Prop .
eq (cars: (car(CID,cidle,N,N1,PID) CS)) S
 |= isidle(CID) = true .
eq (cars: (car(CID,CST,N,N1,PID) CS)) S
 |= isidle(CID) = false [owise] .
ceq (cars: (car(CID,CST,N,N1,PID) CS)) S
 |= hasperson(CID) = true
if not (PID == pidnone) .
eq (cars: (car(CID,CST,N,N1,PID) CS)) S
 |= hasperson(CID) = false [owise] .
ceq (ps: (person(PID,PSTAT,N,N1,CID,DL)
          person(PID2,PSTAT2,N2,N3,CID2,DL2) PS)) S
 |= doublebook(CID) = true  if (CID == CID2) .
eq  (ps: (person(PID,PSTAT,N,N1,CID,DL)
          person(PID2,PSTAT2,N2,N3,CID2,DL2) PS)) S
 |= doublebook(CID) = false [owise] .
```

isidle(*cid*) ID が *cid* の車の状態が cidle である場合に成立する。hasperson(*cid*) ID が *cid* である車に与えられた人の ID が pidnone でない場合に成立する。doublebook(*cid*) は、車の ID として *cid* が与えられた 2 人の異なる人がいる場合に成立する。

そして、2 つの安全性特性 NoDB と ICnoP を、次の様に形式的に規定する。

```
ops nodb icnop : Cid -> Formula .
eq nodb(CID)  = ([] ~(doublebook(CID)) ) .
eq icnop(CID)  = ([] (isidle(CID) -> ~(hasperson(CID)))) .
```

ここで []_、_->_ 、 ~_は、それぞれ、□、⇒、¬ を意味する。

2 つの活性特性 RPweR and VCweS を次のように形式的に規定するために、それぞれの人 ID *pid* に対して、2 つの原始命題 reserving(*pid*) と riding(*pid*) を定義する。

```
ops reserving riding : Pid -> Prop .
ops isvacant inservice : Cid -> Prop .
eq (ps: (person(PID,prequest,N,N1,CID,DL) PS)) S
 |= reserving(PID) = true .
eq (ps: (person(PID,PSTAT,N,N1,CID,DL) PS)) S
 |= reserving(PID) = false [owise] .
eq (ps: (person(PID,pride,N,N1,CID,DL) PS)) S
 |= riding(PID) = true .
eq (ps: (person(PID,PSTAT,N,N1,CID,DL) PS)) S
```

```
 |= riding(PID) = false [owise] .
eq (cars: (car(CID,cidle,N,N1,PID) CS)) S
 |= isvacant(CID) = true .
eq (cars: (car(CID,CST,N,N1,PID) CS)) S
 |= isvacant(CID) = false [owise] .
eq (cars: (car(CID,cservice,N,N1,PID) CS)) S
 |= inservice(CID) = true .
eq (cars: (car(CID,CST,N,N1,PID) CS)) S
 |= inservice(CID) = false [owise] .
```

reserving(*pid*) は、 ID が *pid* の人の状態が prequest である場合に成立する。riding(*pid*) は ID が *pid* の人の状態が pride である場合に成立する。iscacant(*cid*) は、 ID が *cid* の車の状態が cidle である場合に成立する。inservice(*cid*) は、ID が *cid* の車の状態が cservice である場合に成立する。

そして、2 つの活性特性 RPweR と VCweS は、次のように形式的に規定する。

```
op rpwer : Pid -> Formula .
op vcwes : Cid -> Formula .
eq rpwer(PID) = (reserving(PID) |-> riding(PID)) .
eq vcwes(CID) = (isvacant(CID) |-> inservice(CID)) .
```

ここで_|->_は、 $\rightsquigarrow$ を示す。

Map6 を使用するシステムの NoDB をモデル検査するには、次を行う (reduce) だけでよい。

```
modelCheck(im6c2p2,nodb(cid(1)) /\ nodb(cid(2)))
```

ここで_/_は、 $\wedge$ を示す。Map6 を用いたシステムの ICnoP をモデル検査するには、次を行う (reduce) だけでよい。

```
modelCheck(im6c2p2,icnop(cid(1)) /\ icnop(cid(2)))
```

どちらのモデル検査も反例は見つからなかった。これにより Map6 を用いたシステムは 2 つの安全性特性を享受していることが示された。

Map6 を用いたシステムの RPweR をモデル検査するには、次を行う (reduce) だけでよい。

```
modelCheck(im6c2p2,rpwer(pid(1)) /\ rpwer(pid(2)))
```

Map6 を用いたシステムの VCweS をモデル検査するには、次を行う (reduce) だけでよい。

```
modelCheck(im6c2p2,vcwes(cid(1)) /\ vcwes(cid(2)))
```

それぞれのモデル検査は反例を見つけ出す。ただし、これをもって直ちに Map6 を用いたシステムが RPweR や VCweS の活性特性を満たさないと結論づけることはできない。並列システムや

分散システムの活性特性をモデル検査するときに、このような反例が見つかることがよくある。ライドシェアシステムもこのようなシステムといえる。なぜこのような反例が見つかるかというと、不公平なスケジュールが排除されないためである。不公平なスケジュールをすべて排除し、公平なスケジュールのみを使用する必要がある。この目的のために、公平性の仮定を使用できる。車の観点から、各トランジションの弱い公平性と強い公平性の両方を使用する。`wfair(`*cid*`)` を `wf(stratreq)`、`wf(book(`*cid*`))`、`wf(rsvmove(`*cid*`))`、`wf(ridesvc(`*cid*`))`、`wf(svcmove(`*cid*`))`、`wf(getoff(`*cid*`))`、`wf(getoff2(`*cid*`))` の連言として、`wfair` (弱い公平性の仮定の使用) を `wfair(1)` と `wfair(2)` の連言とする。ここで `wf` が WF を示すものとする。同様に `sfair`(強い公平性の仮定の使用) を定義する。

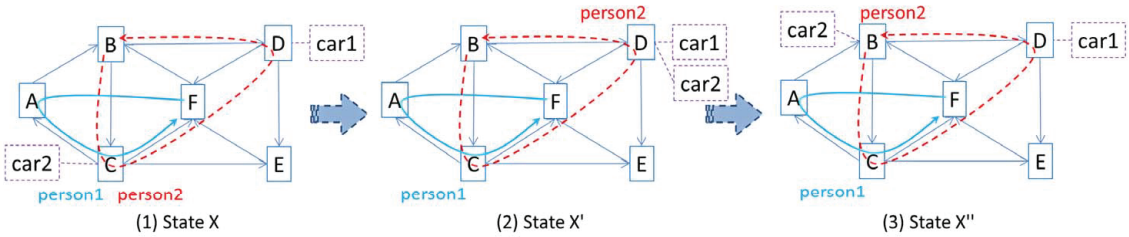

図 4.2: 弱い公平性の仮定での VCweS の反例の一部

Fig. 4.2: A fragment of a counterexample VCweS under `wfair`

Map6 を用いるシステムにおいて、弱い公平性の仮定 `wfair` の元で RPweR と VCweS のモデル検査を行うには、次を行う (reduce) だけでよい。

```
modelCheck(im6c2p2,wfair -> rpwer(pid(1)) /\ rpwer(pid(2)))
modelCheck(im6c2p2,wfair -> vcwes(cid(1)) /\ vcwes(cid(2)))
```

弱い公平性の仮定 `wfair` の元で、RPweR のモデル検査で反例は見つからない。弱い公平性の仮定 `wfair` の元で、VCweS のモデル検査についても、反例が見つからないことを期待したが、実際には反例が見つかった。図 4.2 は、見つかった反例の一部を示している。図 4.2 の状態 X は、`im6c2p2` から到達可能である。person1 と person2 は両方とも節点 C におり、car2 も節点 C にあり、car2 が person2 に最も近いものとなる。 car2 が person2 を節点 D に運ぶと、person2、car1、car2 は節点 D に配置される（図 4.2 の状態 X' 参照)。それゆえ、車 car1 と car2 が両方とも人 person2 に最も近い車となる。どちらの車が人 person2 のために選ばれてもよい。ここで、car2 が選ばれて、人 person2 を節 B に運んだと仮定する。車 car2 は、人 person2 に最も近くなり、車 car1 は近くなくなる (図 4.2 の状態 X" 参照)。つまり、`wfair` では car1 が決して選択されないシナリオが存在する。 なぜなら、たとえ車 car1 が人 person2 に最も近い車になったとしても、車 car1 が継続的に人 person2 に最も近い車であり続けるとは限らないからである。

強い公平性の仮定 `sfair` の元で、Map6 を用いるシステムの VCweS をモデル検査するには、次を行う (reduce) だけでよい。

```
modelCheck(im6c2p2,sfair -> vcwes(cid(1)) /\ vcwes(cid(2)))
```

強い公平性の仮定 `sfair` の元では、VCweS の活性特性のモデル検査の反例は見つからない。

`sfair` の下で Map6 が使用されるシステムでは VCweS が成立するが、`sfair` の下でも VCweS がシステムによって満足されない地図がいくつかあることに注意が必要である。例えば、人が決

して訪れることのない節点があり、もしある車が最初からその節点に配置されており、その車が決して人に最も近いものにならない場合、そのような車は誰にも役立つことはなく、割り当てられない。VCweS は、システムだけでなく地図や初期の車の配置とみなすことができる。特定の地図について、その地図が使用されているシステムの VCweS をモデル検査することで、そのような車がシステムまたは地図に存在するかどうかを確認できる。VCweS を検査する価値は、このような避けるべき車の配置が分かることにある。そのような車が存在する場合は、`wfair` で地図が使用されるシステムで VCweS が成立できるように、車の数を減らすか、各車の初期位置を変更する必要がある。VCweS のモデル検査の利点は、このような状況がシステムを運用する前に確認できる点にあると言える。

4.5 公平性をより低下させたモデル検査性能の改善

`sfair` での VCweS のモデル検査には、3.20GHz マイクロプロセッサ、256GB メモリ、SUSE Linux Enterprise Server を使用する SGI コンピュータで約 7.7 時間要する。Map6 にもう 1 人追加した、`sfair` の下で VCweS のモデル検査が試行すると スタックオーバーフローのため失敗する。この状況を緩和するため、公平性の仮定の下で活性特性のモデル検査に対し分割して克服するアプローチ [53] を適用する。

F_0 を

{`sf(startreq)`, `sf(book(cid(1)))`, `sf(rsvmove(cid(1)) )`,
`sf(ridesvc(cid(1)))`, `sf(svcmove(cid(1)))`, `sf(getoff(cid(1)))`,
`sf(getoff2(cid(1)))`, `sf(book(cid(2)))`, `sf(rsvmove(cid(2)))`,
`sf(ridesvc( cid(2)))`, `sf(svcmove(cid(2)))`, `sf(getoff(cid(2)))`,
`sf(getoff2(cid(2)))`}.

の集合とする。
この集合は、全ての要素の組み合わせの連元を要素として持つ式として得られる。そのため F_0 `-> vcwes(cid1)) /\ vcwes(cid(2))` は、`sfair -> vcwes(cid(1)) /\ vcwes(cid(2))` と等価である。全ての LKSs、つまりは全て EES-KSs のそれぞれの要素 e について $\mathrm{SF}(e) \Rightarrow \mathrm{WF}(e)$ である。この事実を用いて、F_1 を F_0 の全要素について弱い公平性の要素に変換して得られる要素の集合とする。例えば `sf(rsvmove(cid(1)))` は、`wf(rsvmove(cid(1)))` で置き換える。

一つの仮定を推測する。すなわち、それぞれ車の IDcid について、もし `sf(book(`cid`))` と `wf(startreq)` が成立するなら、車はやがて人により予約されるだろうと推測できる。この仮説の部分を、次のように形式的に規定する。

```
eq vcwebr(CID) = isvacant(CID) |-> reserved(CID) .
```

ここで原始命題 `reserved(`cid`)`、ID が cid である車の状態が `creserved` の時に成立する。`vcwebr` は、「空車はやがて予約される」を意味する。我々は、この仮説を次を行う (reduce) ことでモデル検査する。

```
modelCheck(im6c2p2,
  wf(startreq) /\ sf(book(cid(1))) /\ sf(book(cid(2)))
  -> vcwebr(cid(1)) /\ vcwebr(cid(2)) )
```

反例は見つからない。これは同じ SGI の実行環境で約 3 秒で実行できる。F_2 を前提内のすべての連言の集合として構成し $F_2 \subseteq F_0 \cup F_1$ とする。そこで次のモデル検査の実験を実行する。

```
modelCheck(im6c2p2,
  wf(rsvmove(cid(1))) /\ wf(rsvmove(cid(2))) /\
  wf(ridesvc(cid(1))) /\ wf(ridesvc(cid(2))) /\
  vcwebr(cid(1)) /\ vcwebr(cid(2)) -> vcwes(cid(1)) /\ vcwes(cid(2)))
```

反例は見つからない。これも同様の SGI の実行環境で約 3 秒要する。F_3 を前提内のすべての連言の集合として構成し $F_3 \subseteq F_0 \cup F_1 \cup \{$`vcwebr(cid(1))`,`vcwebr(cid(2))`$\}$ とする。事実より、$F_0 \Rightarrow F_1$ である。はじめのモデル検査の実験から、$F_0 \cup F_1 \Rightarrow F_3$ である。二つ目の実験から、$F_3 \Rightarrow$ `vcwes(cid(1)) /\ vcwes(cid(2)))` である。このため、$F_0 \Rightarrow$ `vcwes(cid(1)) /\ vcwes(cid(2)))` である。それゆえ、我々は次の行うことで反例は見つからないと言える。

```
modelCheck(im6c2p2,sfair -> vcwes(cid(1)) /\ vcwes(cid(2)))
```

これらの二つのモデル検査の実験結果から、分割して克服するアプローチを適用することで、直接的に実行すると約 7.7 時間要するモデル検査を約 6 秒でモデル検査できるといえる。

さらに VCweS が成り立つために必要な公平性に関する仮定を改良した。全ての遷移に関して、強い公平性を仮定する必要はなく、それぞれの *cid* に対して、 `sf(book(`*cid*`))`、`wf(startreq)`、`wf(rsvmove(`*cid*`))`、`wf(ridesvc(`*cid*`))` の公平性を仮定すれば十分であることを示した。`book(`*cid*`)` だけに対しては強い公平性の仮定が必要であり、その他の `startreq`、`rsvmove(`*cid*`)` 、`ridesvc(`*cid*`)` に対しては弱い公平性を仮定すれば十分である。加えて、`svcmove(`*cid*`)`、`getoff(`*cid*`)` 、`getoff2(`*cid*`)` に対しては、全く公平性の仮定を置く必要がないことを示した。

Map6 にさらに 1 人追加すると、1 つ目のサブモデル検査実験には約 5 秒、2 つ目のサブモデル検査実験には 3 分 38 秒要する。つまり、分割して克服するアプローチにより、公平性の下で活性特性のモデルの検査実験は、合計で約 3 分 43 秒で実行できる。他方、直接的な方法で同じことを実行すると、スタック オーバーフローを引き起こし、実行不可能である。

本章では、文献 [53] で示された公平性仮定の元での活性特性のモデル検査に分割統治アプローチの適用事例とは別の事例を示していることに注意いただきたい。これは、本研究をはじめとする多くの論文が、モデル検査が効果的に適用できる事例については報告しているものの、モデル検査そのものを提案していないのと似ている。 後者は報告する価値があり、前者の事例も報告する価値がある。

なお、LKS の EES-KS は、ローカライズされた公平性に関する Meseguer の研究が起源である [46]。公平性の仮定の元での活性特性のモデル検査に特化したモデル検査器が開発された。その一つは、Bae-Meseguer のモデル検査器である。[24] 我々は、このモデル検査器を 並列システム の RPweR と VCweS の活性特性のモデル検査に適用したが、我々の目的で用いることはできなかった。

4.6 本章のまとめ

本章の貢献は以下の 2 点である。

- ライドシェアリング システムの本質的な性質の検証を可能とするため、地図、車、人といった要素をパラメータ化した形式仕様の記述を示した点
- 公平性の仮定を置くと計算時間が爆発するが、分割統治アプローチを適用することで現実的な時間でモデル検査可能となる実例を示したこと

4.6.1 まとめ

Maude でライドシェアリング システムを形式的に規定し、Maude LTL モデル検査器を使用してシステムが望ましい特性を享受していることをモデル検査した。形式的にシステム仕様を記述する本章のアプローチでは、地図、車の集合、および人の集合がパラメータとして扱われるため、1 つのライドシェアリング システムについて 1 つの形式的にシステム仕様の規定を準備するだけで十分であり、これを複数のインスタンスに使用できる。このアプローチでは、豊富なデータ構造を使用する必要がある。これには、LTL モデル検査器を備えた Maude を利用することで解決した。Spin などの既存の LTL モデル検査器のほとんどは、豊富なデータ構造をサポートしていないため、少なくとも単純に本研究のアプローチを実装するために使用できるかどうかは疑問が残る。賢明な研究者やエンジニアの中には、制限された配列や整数などの限られたデータ構造セットを使用して、本研究のアプローチで使用される豊富なデータ構造の重要なエンコーディングを思いつく人もいるかもしれない。しかし、そのような自明ではないエンコーディングが豊富なデータ構造を忠実に表現していることを誰が保証するのであろうか? それには、重要なエンコーディングが豊富なデータ構造を忠実に表現していることを正式に検証する必要がある [54]。

また、公平性の下でモデルの活性特性をモデル検査することが非常に遅くなり、実行不可能になるという状況を分割統治アプローチは軽減でき、活性特性が維持される公平性の仮定を洗練できることも実証した。このアプローチにより、公平性の下で遅い活性特性のモデル検査 (約 7.7 時間) が高速化 (約 6 秒) され、実行不可能なモデル検査が実行可能かつ適度に高速になった (約 3 分 43 秒)。分割統治アプローチでは、活性特性が維持される前提を洗練させることができる。これも実証された。

2 つの安全性特性と 2 つの活性特性に加えて、関連するシステムについてさらにいくつかの安全性と活性性プロパティをモデル検査した。さらにいくつかのマップを使用してモデル検査の実験を実施し、各モデル検査実験に要する時間を測定した。

4.6.2 今後の課題

モデル検査の性能に関して我々が発見した興味深い点の 1 つは、モデル検査の実験に要する時間は地図のサイズに重大な影響を受けないことだ。これは、ダイクストラのアルゴリズムを使用して節点のすべての有向ペア (n, n') の最短経路を事前計算しているためと考えられる。

本章のライドシェアリングシステムの形式仕様では、各人は常に複数の中間目的地を特定の順序で訪問し、同じ場所に戻ることを繰り返す。これを各人は、システムに車の配達を依頼するたびに、その場で目的地を決定するように変更できる。例えば線形時相論理のモデル検査が可能な Maude、初期状態の Map6 上の person1 を表す用語は次のように修正する。

```
person(pid(1),pidle,nf,nonode,cidnone)
```

ここで nonode は person1 が訪れる目的地を持っていないことを示す。そして書き換え規則で次のように置き換える。

```
crl [startreq1] :
 (ps: (person(PID,pidle,N1,nonode,CID,DL) PS))
 (dst[PID]: (N | Ns)) (tran: T)
  =>
 (ps: (person(PID,prequest,N1,D,CID,DL) PS))
 (dst[PID]: Ns) (tran: startreq)
if not(N1 == N) .

crl [startreq2] :
 (ps: (person(PID,pidle,N1,nonode,CID,DL) PS))
 (dst[PID]: (N | Ns)) (tran: T)
  =>
 (ps: (person(PID,pidle,N1,nonode,CID,DL) PS))
 (dst[PID]: Ns) (tran: startreq)
if N1 == N .
```

ここで、(dst[PID]:...) は、PID の訪問可能な目的地の節点のスープを保持する新しい観測可能なコンポーネントである。例えば、この観測可能なコンポーネントは最初は na na nd ne と na を 2 度、nd と ne を一度訪問候補として保持してもよい。この場合、PID は na 最大二度、nd を一度と ne を 一度訪問する。 PID 非決定的に選択された場所を順次訪問する。実際の地図の取込と各人の訪問頻度を加味した訪問先の選択の実装は今後の課題である。

4.6.3 5 章、6 章の実例への適用に関する基礎検討

本章で示した形式仕様によるモデル検査は、5 章、6 章で示す人やモノを運ぶモビリティサービスに対して適用することは可能である。しかし、大規模な仕様のモデル検査を実行するとメモリ不足で実行不能となったり、現実的な時間で実行できなかったりする。具体的には、本章で示した安全性検証の計算時間は、人、車、地図のノードの数を増やすと指数関数的に増加する傾向がある。図 4.3 に示す地図を用いて変更した際に NoDB の計算時間（単位:ms）を図 4.4 に示す。X 軸は車の数である。

活性特性に関しては、人に関する公平性の仮定を必要とする RPweR は、人を増やした場合、指数的に増加し、3 名の場合では現実的な時間で結果がでた。車に関する公平性の仮定を必要とする VCweS については、上述の通り、分割統治アプローチを適用した場合、3 名の場合では現実的な時間で結果が得られたが、4 名に増やすと 1 カ月でも結果が得られなかった。

5 章、6 章の実例は、人流、物流の起点・終点となる建物やステーションの数、あるいは移動需要の量は、上で説明した人・車・地図の規模より大きい。このため本章で示した形式仕様をそのまま適用すると現実的な時間で計算結果が得られないことが予想できる。しかしながら、本章で示した実験で明らかになった知見は 5 章、6 章のシステムの設計に活用している。具体的には VCweS

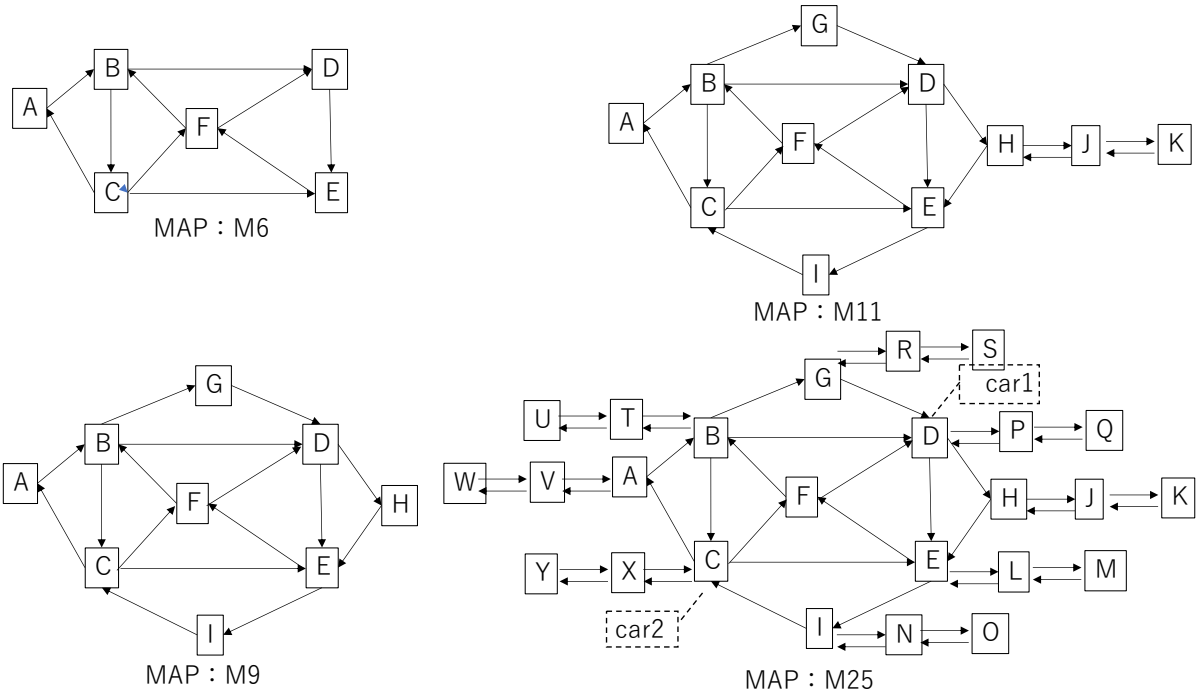

図 4.3: 計算時間の調査に用いた地図

Fig. 4.3: maps used in computation time measurement

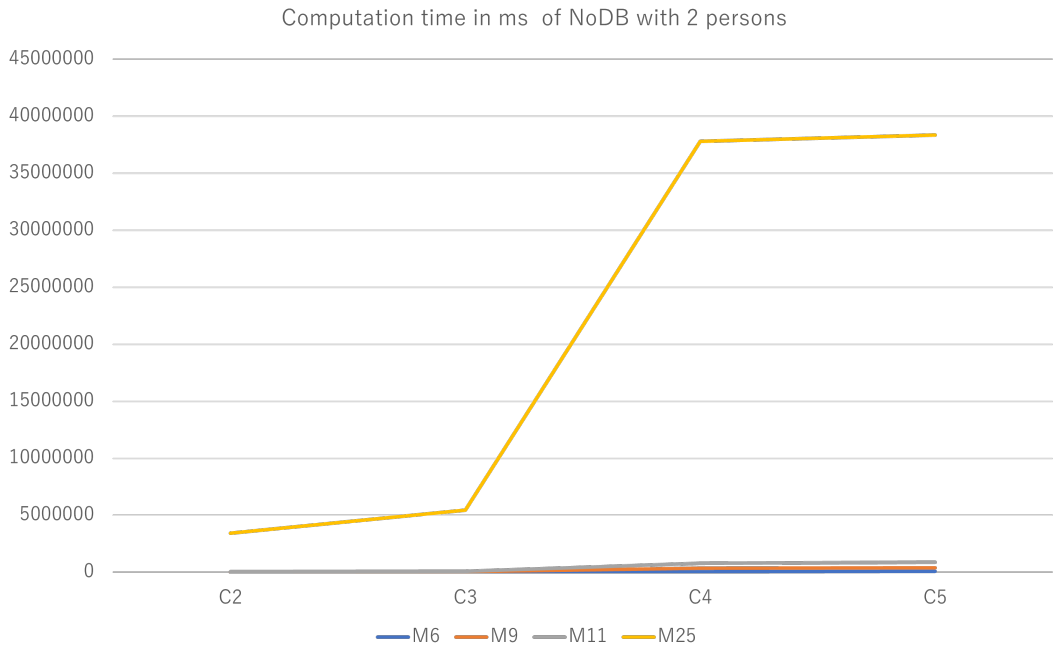

図 4.4: 地図変更時の NoDB の計算時間

Fig. 4.4: Computation time in ms of NoDB with 2 persons

を満たすためには、book の書き換え、すなわち予約割り当てに対して強い公平性の仮定をおく必要があるという知見は、5 章や 6 章の実装で用いている配車割り当てのアルゴリズムの設計時に考慮している。

第5章　人を乗せるサービス設計・評価への適用

本章では、3 章で説明した手法を人を乗せるモビリティサービスの設計・評価に適用した事例を説明する。本章の内容は文献 [73] として発表したものと同等の内容である。

5.1　導入

環境にやさしい電動車両を用いて人やモノを運ぶモビリティサービスは、持続可能で快適な生活には必須のサービスとして期待される。中でも安全性が担保された自動運転によるサービスは、低コストで利便性の高い運用を可能とする切り札として期待されている。

我々は、人の生活圏にフォーカスしたモビリティソリューションを通じて、“人”を元気に“コミュニティ”を元気に“地球”を元気にするサービスの創出を目標に活動を展開している [78]。人の生活圏では、クリーンで環境にやさしい電動車両 (EV) を用いた安全便利で安価なモビリティサービスが求められている。

人の生活圏への新たなモビリティサービスの導入・維持には、人混在環境での運用が求められるため、これまで鉄道や道路の計画・導入・維持の評価に用いられてきた費用便益分析による評価では、事故発生の可能性がある場所や頻度や事故の重篤度が特定できず、安全性の評価の観点で不十分である。

本章では、人の活動データの分析に基づき、3 章で提案した手法を大阪地区での自動配送ライドシェアシステムに適用した例を説明する。

本章の構成は次の通りである。次の節で、モビリティサービスの要求について述べる。5.3 節でモビリティサービスの導入・設計のアプローチを再度説明する。5.4 節でサービス設計・評価手法を述べる。5.5 節で、サービス設計・評価手法の実装、5.6 節で評価を述べ、5.7 節で本章のまとめを述べる。

5.2　モビリティサービスの要求

モビリティサービスの導入及び改善を行う意思決定者は、どのようなエリアにどのような条件のモビリティサービスが導入可能か判断する必要がある。意思決定者にとっては、自動運転機能を持ったモビリティを特定のエリアに導入する場合、新たなリスクと新たな利便性を特定エリアに導入することになるため、単純な費用便益の評価のみではなく、適切な判断基準・評価手法が必要となる。

人が日常生活を行うにあたり、子供を遊ばせたり、近隣の知人とのコミュニケーションを図ったりする場所であった街においても、高速で移動可能な自動車の侵入できる状況がこれまで常識とされてきた [69]。しかしながら、我々は、日常生活を行う街では、人の存在に適応し低速で走行することで人混在環境でも人に恐怖心を与えない人にやさしいモビリティのみを走行可能とし、有害な排ガスを出し、高速移動する自動車とは分離することが望ましいと考える。ここでは、この

ような街を想定し、人にやさしいモビリティとしての、ADEV を用いたサービスが満たすべき要求を記述する。

5.2.1 安全性

ADEV は、車両の外の歩行者等の人や ADEV に乗車中の人に危害を加えるような走行をしてはならない。

5.2.2 利便性

ADEV を用いて移動することによって、人は移動時間の短縮、移動中の時間を有効に活用できることで利便性を享受する。ADEV の運行は、人の移動に関し、最大限の利便性を提供する必要がある。

5.2.3 経済性

ADEV の安全運行には、ADEV の導入費用や維持費用の他にも、遠隔監視・制御のための費用、乗車場所の提示、ADEV 運行環境であることを周知するための運行路や運行の路側の表示の設置等、安全運行に必要な環境維持のための費用が必要となる。持続可能な ADEV 運行の実現には、これらの費用の削減・圧縮が必要となる。

5.3 モビリティサービスの設計・導入アプローチ

安全性、利便性、経済性の指標を向上させるための決定要因は互いに依存関係がある。その関係を考慮し均衡解を探索し、妥当なサービスを決定する。本章では、この決定要因をサービス因子と呼ぶ。サービス因子の変化は、安全性、利便性、経済性の指標に影響及ぼすため、これらの状況が変化した場合を想定し、その状況に応じた均衡解を再度探索することで最適なサービスに変更する。

サービス因子は、具体的には、表 5.1 で示す要素で構成される。主に影響がある指標欄の R、V、C はそれぞれ 5.2.1 節、5.2.2 節、5.2.3 節 で示した安全性、利便性、経済性の指標に影響があるかどうかを示す。例えば、移動需要 (人の流れ) が増えると利用価値が高まるため利便性の指標に影響するが、歩行者も増えるため運行している ADEV との事故に至るリスクも増える。他の交通参加者の交通量が増えれば安全性が低下するが、乗り換え等により移動需要が増加することにつながれば利便性の指標は良化する。運行に関係するサービス因子である速度や車両性能の加減速性能は、利用者の移動時間の短縮につながるため利便性の向上につながるが車両価格が上昇し経済性が悪化する。ブレーキやセンサーのシステムを二重化すると故障等に対する安全性は向上するが導入・維持のコストが高まるため経済性は悪化する。ADEV は、例えば突然の走行路の封鎖等、想定外の状況になれば停止することで安全性を担保する。このとき遠隔管制からの監視や介入 (制御) することにより運行の継続性を担保する。このようなオペレーションはコスト要因であるが利便性の担保には必須となる。ADEV の運行環境では、交通ルールを策定し、これを周囲の交通関係者である歩行者等に周知徹底する。この交通ルール策定・見直しや交通ルールを徹底するための標識・表示、信号等の設備投資が不十分であると事故のリスクが増大する。ADEV を運

表 5.1: 主要サービス因子

Tab. 5.1: Majour Service factor

分類	サービス因子	主に影響を受ける指標
需要	移動需要（人の流れ）	R, V
運行	コース、ステーション配置、運行速度、運行時間等	R, V, C
車両	乗車定員、センサー・ブレーキ・加速性能、二重化実装度	R, C
オペレーション	遠隔監視、遠隔制御による介入	V, C
交通量	交通参加者の種別、量	R, V
環境	標識・表示、信号等	R, C

行するサービスの設計においては、運行する環境に応じてサービス因子を適切に設定し、安全性を十分担保したうえで、利便性、経済性のバランスを取る。

5.3.1 サイバー空間を用いた最適解探索

ADEV を運行するサービスの設計では、複雑に関係したサービス因子の組合せの中から、求める最適解を探索する必要がある。実世界の ADEV の導入や運行を試行錯誤すると高コストになる。このため、サイバー空間で複雑に関連するサービス因子の関係を担保し、ADEV を運行する環境をシミュレートし、移動需要に応じた最適解を探索する。移動需要としては、現実世界の人の流れを再現する。

5.3.2 大阪地区での自動運転ライドシェアサービス

企業内の MaaS のサービス実証として、構内の社員を対象とした ADEV を用いたライドシェアサービスを展開している [79]。

大阪のパナソニック株式会社の本社エリアの敷地内に乗車定員最大 4 名の小型 EV に自動運転のための装備 (センサー、認識・判断・操作を行う ECU、機能安全系の装置、二重系ブレーキ等) を導入し ADEV とした車両 4 台を用いた。ADEV ライドシェアサービスを 2019 年 10 月に開始した。図 5.1 は、ADEV 車両の外観を示したものである。外部をセンシングする LiDAR やカメラを複数搭載した小型の ADEV を用いたサービスであり社員であれば誰でも利用できる。

社内の敷地内で一周 2.4km の周回路を最大速度 20km の社内ルールの範囲内で運行している。2021 年現在の運行は、乗降を行う 3 個所のステーションで、定期的に時刻表にしたがって運行するメトロ型運行と乗車希望に応じて出発地のステーションに迎車するオンデマンド運行の 2 つの運行形態でサービスを運用している。

5.3.3 サービス更新サイクルを実現するためのデータヒュージョン・分析基盤

前節で説明した環境で得られたデータを正規化して蓄積し分析・可視化するためのツール群を整備してきている。可視化ツールとしては、MapBox [5]、OpenStreetMap [8]、Maxar [6] の出力を利用している。図 5.2 は、人流量可視化ツールの例である。人が建物や門から出入りする際に記

図 5.1: 大阪地区での社員向けサービスの車両

Fig. 5.1: Service vehicle for employees in the Osaka area

録する入退室のデータから建物間の移動を集計しグループ化して色の濃さと線の幅で人流の多さを表現している。

グループ化することで大まかな移動の量を直観的に把握することができ、ステーションの配置の検討や、ADEV の運行ダイヤの策定に利用できる。

5.3.4 ミクロ交通シミュレータの利用

移動要望を持った人が特定の ADEV に乗車できるかどうかは、ADEV に空席があるかといった因果関係を表現する必要がある。また、人の流れが多い運行環境で ADEV を運行すると ADEV と人 (歩行者) が近接するリスクが高まる。このような因果関係を再現するシミュレータとして、ミクロ交通シミュレータである PTV Vissim [11] を採用する。PTV Vissim はロンドンオリンピックの運用モデルとして活用された実績を持ち [32]、国内外問わず交通施策の検証や研究に広く利用されている。

5.4 サービス設計・評価手法

モビリティサービスを特定エリアに導入する際の一連の工程を図 3.1 を参照して振り返る。まず、新たなモビリティサービスを特定の地域（エリア）に導入する場合、要求される事項を指標化する。次に、当該エリアの移動需要、交通量の調査を行い、車両や投入台数といった運行候補を複数パターン選定する。さらに、ステーション配置、走行路、運行数、運行形態といった導入

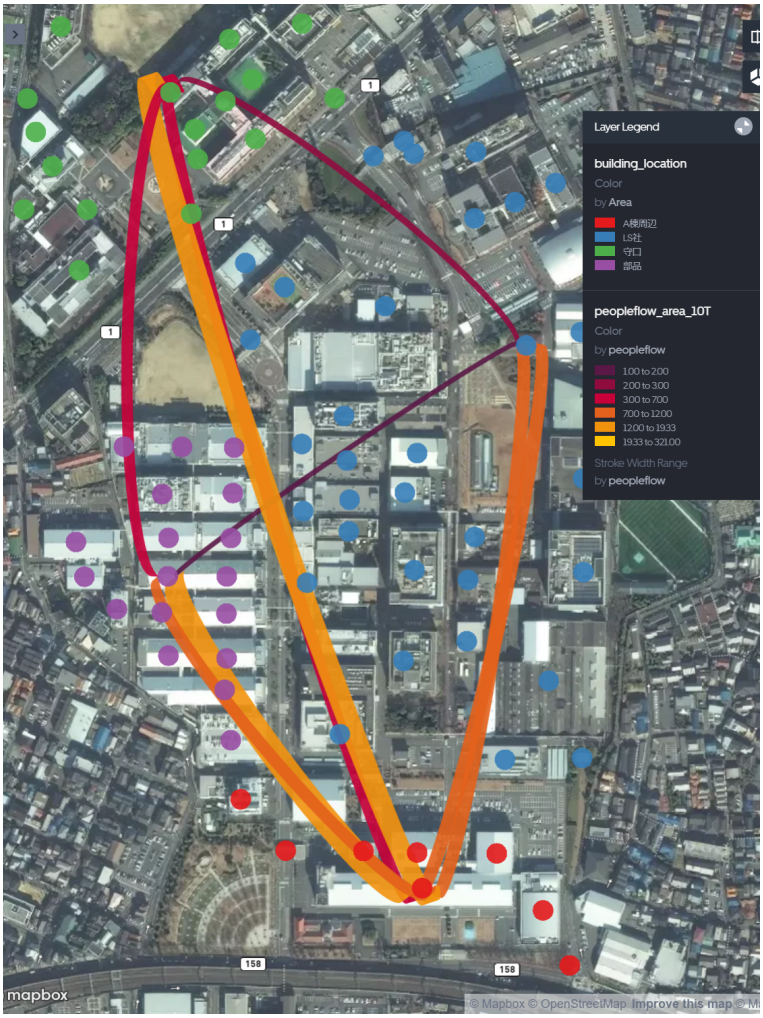

図 5.2: 人流量の可視化ツールの出力例

Fig. 5.2: Output example of human flow visualization tool

するモビリティサービスの仮説を立案する。その後、これらのサービスを安全に実現するための安全設計を行う。この過程で不安全な運行を排除する。例えば、機能安全設計が不十分な大型車両を歩車分離のない住宅地で降雪時に高速運行させるといったサービス仮説はこの工程で棄却する。この後、安全性、利便性、経済性の観点から妥当か計算機シミュレーションを用いた定量評価を行い、モビリティサービス導入の意思決定を行う。式 (1) で M は、「仮説の評価」の工程で、安全性、利便性、経済性を加味した評価指標の例である。

$$M = \alpha R + \beta V + \gamma V \tag{5.1}$$

$$R = \sum_{i=1}^{n} A(\sum_{j=1}^{m} P_j(i)) \tag{5.2}$$

$$V = \sum_{k=1}^{t} r(T_{org}(k) - T_s horten(k)) \tag{5.3}$$

$$C = \sum_{j=1}^{m} C(j) + C_{ope} + C_{env} \tag{5.4}$$

上記の式 5.1 の $\alpha\beta\gamma$ は、それぞれ安全性、利便性、経済性の指数間の重みを決める定数である。α、γ は、マイナスの数値、β はプラスの数値を設定する。α、β、γ の絶対値は、意思決定者の意向により変更し繰り返し評価・調整可能である。

5.4.1 安全性の指標

式 5.2 で定義する R は安全性の指標であり、具体的には特定の ADEVj と歩行者 i との間の近接回数を示す。$P_j(i)$ は、$ADEV_j$ と歩行者 i との距離が最高速度 Vmax × 定数の範囲に入った回数である。図 3.1 の「モビリティサービス安全設計」の段階で危険な事象は排除できる車両性能、システム性能、運行条件を設計し、安全性を担保した上で ADEV の運行を計画する。「仮説の評価」の段階では、ADEV と歩行者との近接回数からヒヤリハット事象の発生回数を算出し、これをもとにハインリッヒの法則を適用し、事故発生件数を算出する。その事故の障害の程度から損害額に金額換算した値 (円) を採用する。障害の程度は、事故時の運動エネルギーから、AIS : Abbreviated Injury Scale （簡易傷害スケール）[1] を用いて算出し、その障害の治療に必要な金額を損害額 (円) として算出する。式 5.2 の関数 A() は、この損害金額への換算を行う関数である。

5.4.2 利便性の指標

式 5.4 で定義する V は利便性の指標である。具体的には、 $T_{org}(k) - T_s horten(k)$ は、人が ADEV に乗ることで短縮できた時間 (時間) であり、有効な時間が創出できたとし工数単価 r を掛け、金額に換算している。なお、本章では r として代表的な従業員の健保等級として用いられる 4500 円/時間を採用した。

5.4.3　経済性の指標

式 5.3 で定義する C は経済性の指標である。具体的には、C(j) は、ADEVj の減価償却費、保守の費用 (日割、単位: 円) C_{ope} は、ADEV を用いたライドシェアサービスのオペレーションに必要となる運営費用 (日割、単位: 円) である。C_{env} は、同サービスを運用する環境面の維持費用（ステーション費用、白線・ガードレール・標識等の設置費用（日割、単位: 円）である。

5.4.4　評価指標を最大化する最適解の探索

意思決定者は、安全性を担保しながら、利便性と経済性の均衡点を探索する。具体的には、金額に統一された式 5.1 を最大化する ADEV の運行サービスを策定する。安全性の担保のための方針として $\sum_{j=1}^{m} P_j(i)$ が一定数以下となる条件で式 5.1 の M を最大化するサービス因子の組合せを探索する。

5.5　サービス設計・評価手法の実装

本節では、5.3.2 節で説明した大阪地区での ADEV の運行サービス対象としたサービス設計・評価手法の実装方法を説明する。

5.5.1　人流と乗車の再現方法

構内の人流データとして、構内のそれぞれの建物の入退室のデータを用いて、ミクロ交通シミュレータ PTV Vissim で人の動きを再現する。シミュレータの地図の構内の建物の出入り口に人の移動が発生する出発点や人の移動が完了する目的点を設置し、入退室データに応じた人の移動を再現する。このため、現実とシミュレータ上の人流の発生量は完全に一致し、ADEV の空席状況や構内を移動する歩行者の流れを正確に再現でき、ADEV と人の近接するリスクを評価できる。

特定の人の移動が発生した際に、図 5.3 に示す乗車判定アルゴリズムに従ってサービス車両に乗車すべく乗車ステーションに向かうか、目的点まで徒歩で移動するかを決定する。つまり、歩行者が移動元の建物のゲート 1 から出て、目的の建物のゲート 2 に移動する入退室データをもとに、ゲート 1 から出たタイミングで乗車ステーションまで歩行し、到着する ADEV を待ち、乗車して移動、降車ステーションで降車し、目的の建物のゲート 2 に移動する時間の合計 (図中の A+B+C+D) が ADEV に乗車せず最短経路で歩行して移動する時間 (図 5.3 の E) より短い場合に、人は ADEV に乗車すると判定する。この乗車判定は、人が建物の出口のゲートに出たタイミングで、最短経路で歩いて移動するか、ADEV で移動するかを判断するための情報にアクセスし判断可能であることを前提としている。また、この乗車判定のアルゴリズムは、身体的理由により歩行が困難である人や乗車のコストを支払う団体に所属しない人等、行動制約がある場合には、このアルゴリズムの対象外となる。

歩行者は横断歩道で車道を横断したり、歩道を歩く人数が多い場合、車道にはみ出して追い越したりするため、人の流れが多い交差点や通りでは、ADEV と近接するリスクが高まる。シミュレーションでは、図 5.3 に示す risk_d の車両速度に比例して長くなる円形の領域に、歩行者が入る回数をヒヤリハット事象につながる件数としてカウントする。

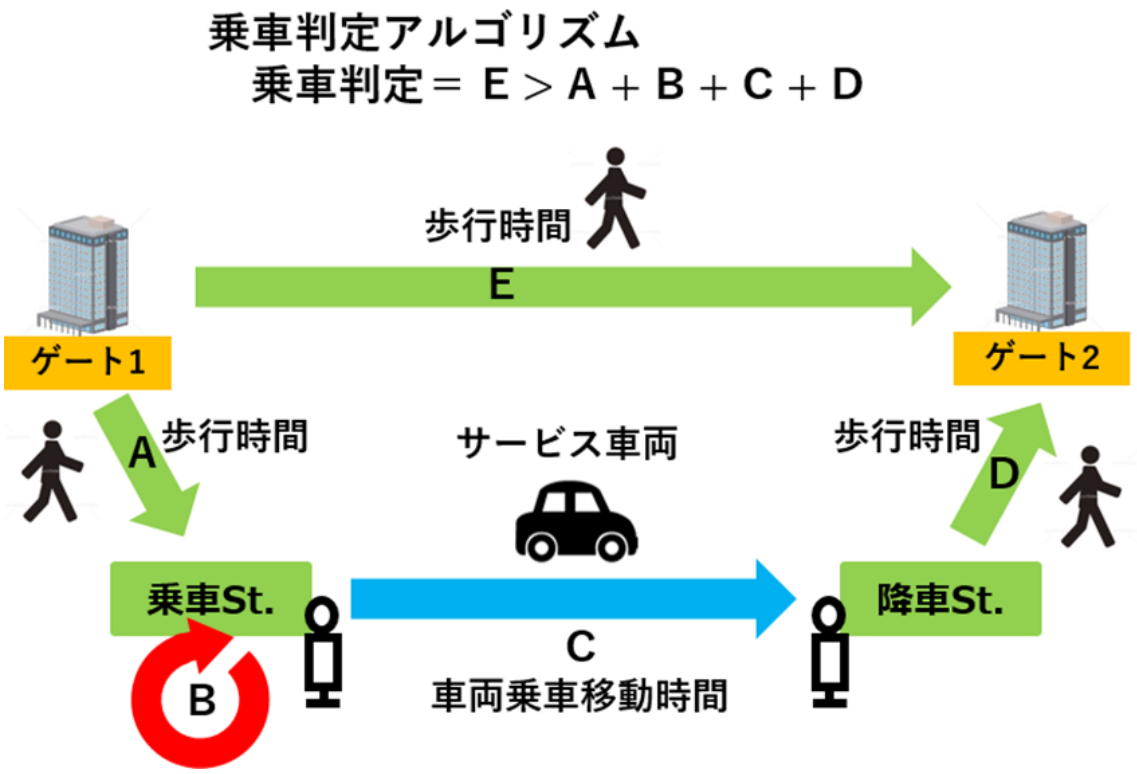

図 5.3: 乗車判定アルゴリズム

Fig. 5.3: Boarding judgment algorithm

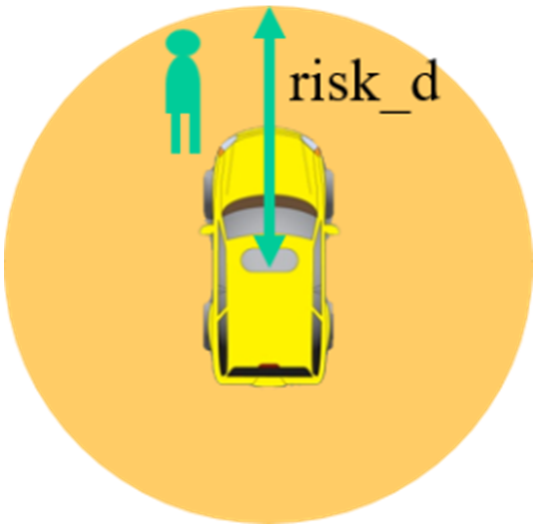

図 5.4: 安全性の指標に用いる ADEV 近傍領域

Fig. 5.4: Area near ADEV used as an index of safety

5.5.2 HPC(StarBED) を用いた全探索

ミクロ交通シミュレータは、マクロ交通シミュレータと比較して、ADEV、人といった対象物の動作を細粒度で再現するため演算時間を要する。複数のサービス因子の組合せの中から、最適な解を短期間で探索するため、複数のシミュレーションの並列実行を可能とする HPC(High Performance Computer) を用いる。HPC として、PTV Vissim がベースとしている Windows 系の OS を前提とした展開や並行実行の実績 [75] がある StarBED [48] を採用する。並行実行の自動化環境として、HTCondor [3] を StarBED に導入し、サービス因子を組み合わせたすべてのパターンのシミュレーションを実行し、実行結果から最適解を特定する。

5.5.3 最適解の探索（焼きなまし法）

HPC による並行実行による処理能力は HPC のノード数に比例するため、HPC を用いても爆発的な組み合わせパターンの増加を吸収することはできない。つまり、複数のサービス因子に関するパラメータの評価域の全域のシミュレーションを実行すると不効率である。そこで、 式 5.1 で定義した評価指標 M を最大化するサービス因子の組み合わせを、最適化手法の焼きなまし法 [41] で探索する。なお、多峰性がある最適解の探索方法としては、焼きなまし法の他に、タブー探索、ネルダーミード法等を用いることで局所最適解に陥るリスクを低減できる。

5.6 評価

これまで説明した手法を用いて、大阪地区での ADEV の運行の最適解を、HPC を用いて探索した。

5.6.1 人流と乗車の再現

ADEV の運行を想定し、表 5.1 で示した関係を加味したサービス因子から、大阪地区での運行サービスに必要な要素を抽出、パラメータ化しシミュレータに実装した。具体的には、最高速度、乗車定員、車両台数とメトロ型運行とオンデマンド運行の比率、乗車の判断を補正する時間（希望短縮時間）、運行経路パターンの 5 つのサービス因子をパラメータ化した。

評価範囲の確定

大阪地区では 20km/h に構内速度が限定されているため、最高速度は、これ以下の範囲で 6km/h、10km/h、14km/h、18km/h とした。ADEV の種別は小型の 1 種類で乗車定員数は、1 名から 4 名、投入台数は 2 台から 5 台、運行形態は、メトロ型運行とオンデマンド運行の混在比を網羅的に 10 パターンとした。乗車を判断する際に希望する短縮時間は、より短縮の移動を好む人から多少移動時間が伸びても ADEV に乗車する人の判断を模倣し評価するため、+120 秒、+60 秒、0 秒、-60 秒、-120 秒の範囲とした。また、運行経路としては、移動エリアの両端を中心に乗降ステーションを配置する運行経路と移動エリアの中央に乗降ステーションを配置する運行経路の 2 通りとした。

これらの 5 つのパラメータの組合せは、1600 通りとなる（表 5.2）。

表 5.2: サービス因子の組合せ

Tab. 5.2: Combination of service factors

分類	サービス因子	パラメータ値
需要	人流データ	2019 年 9 月 9 日分
	希望短縮時間	-120, -60, 0, 60, 120
車両	最高速度 (km/h)	6, 10, 14, 18
	乗車定員（人）	1,2,3,4
運行	投入台数（メトロ型—オンデマンド型）	2(1-1), 3(1-2), 3(2-1), 4(1-3), 4(2-2), 4(3-1), 5(1-4), 5(2-3), 5(3-2), 5(4-1)
	乗降ステーション配置	（図 5.5、図 5.6）中間ステーションあり、なし

評価指標の重みの特定

本章では、式 5.1 で説明した評価指標の重みとして、金額換算した指標をその通りの重みで評価するバランス型（α, β, γ）=(-1,1,-1) と、それぞれを相対的に安全性、利便性、経済性を 3 倍重視する条件の合計 4 条件の探索例を示す。

5.6.2 HPC の並列処理による最適解の算出（全探索）

StarBED の 30 ノードを用いて表 5.2 に示す 1600 通りのサービス因子の組合せのミュレーションを並列実行した。これらの 1600 パターンのシミュレーション実行は 1 台の PC でシーケンシャルに実行した場合、9287 時間必要となるが、30 ノードで 1 ノードあたり 4 プロセスの PTV Vissim を並列実行することで、要した時間は、81.12 時間であった。このとき使用した機材の性能は表 5.3 の通りである。

表 5.3: 使用機材の仕様（1 ノード）

Tab. 5.3: Specifications of equipment used (1 node)

CPU	Intel Xeon E5-2683 v4（2.1GHz/16core）× 2
Memory	32GB RDIMM（DDR4-2400/mb ECC）× 12
Storage	HDD 1.2TB × 1, SSD 1.6TB × 1
NIC	10GigE × 2, 1GigE × 1

5.6.3 焼きなまし法を用いた最適解の探索

5.6.1 で求めた評価指標 M を最大化するサービス因子の組合せを 5.6.1 で示した評価指標の重みを用いて焼きなまし法で探索した。実行時間を表 5.4 に、それぞれの最適解におけるサービス因子の組み合わせを表 5.5 に示す。表 5.5 の投入台数欄は、x(y-z) の形式で示しており、x は合計台数で、そのうち y、z はそれぞれメトロ型、オンデマンド型の台数である。

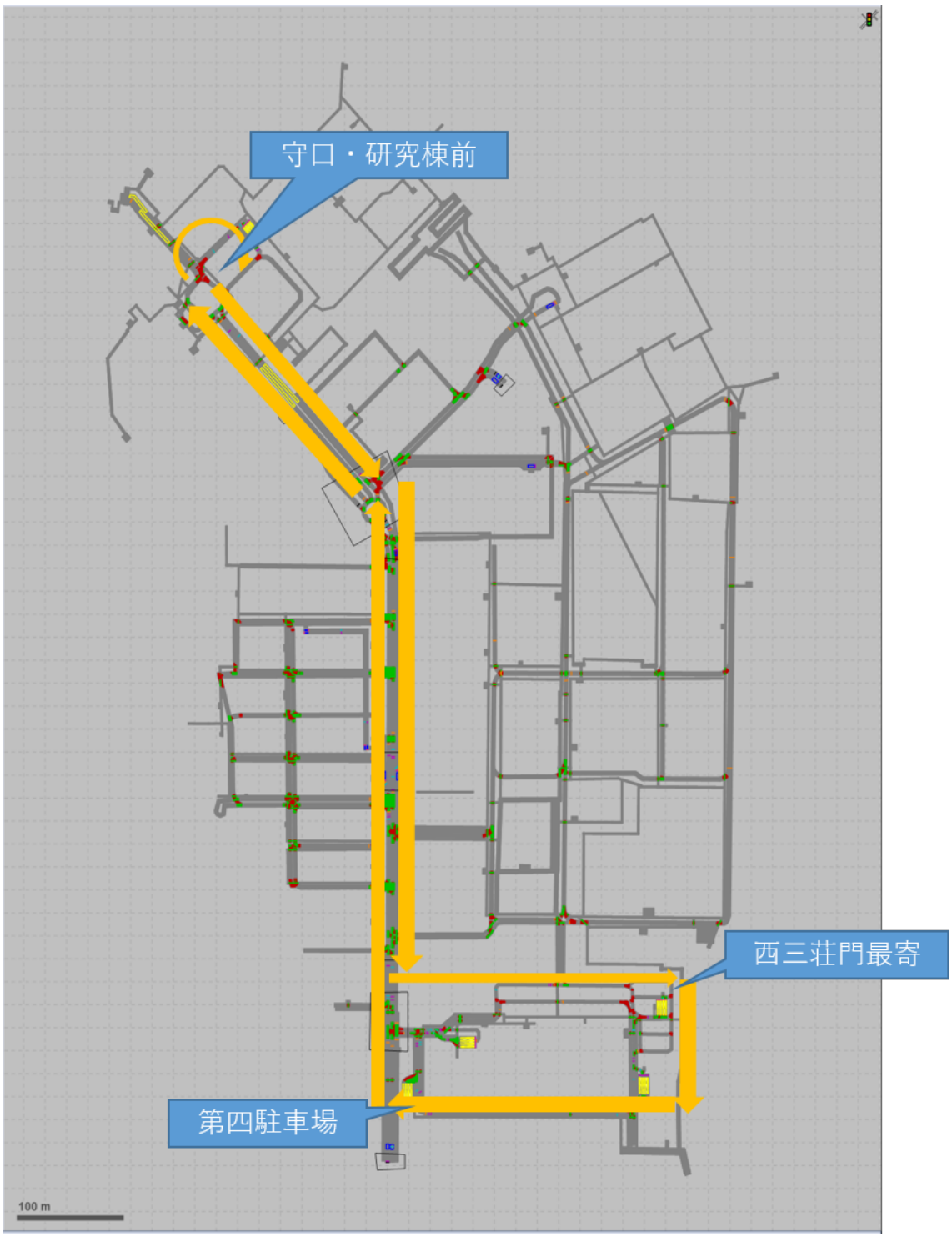

図 5.5: 中間ステーションを配置しない運行経路

Fig. 5.5: Operation route without intermediate stations

図 5.6: 中間ステーションを配置する運行経路

Fig. 5.6: Operation route with the intermediate station

表 5.4: 実行時間比較

Tab. 5.4: Execution time comparison

1 ノードシングルプロセス（全探索）	9287 時間
30 ノードで 120 並列（全探索）	81.12 時間
1 ノードで焼きなまし法 (バランス型)	172.1 時間
1 ノードで焼きなまし法 (安全性重視)	127.7 時間
1 ノードで焼きなまし法 (利便性重視)	127.7 時間
1 ノードで焼きなまし法 (経済性重視)	121.9 時間

表 5.5: 最適解の探索結果

Tab. 5.5: Optimal solution search result

条件	バランス型	安全性重視	利便性重視	経済性重視
(α , β , γ)	(-1,1,-1)	(-3,1,-1)	(-1,3,-1)	(-1,1,-3)
希望短縮時間 (s)	120	60	0	120
最高速度 (km/h)	10	6	10	10
乗車定員（人）	3	2	4	4
投入台数	5(2-3)	5(3-2)	5(3-2)	5(2-3)
中間ステーション	あり	あり	あり	あり
M の値	6.16	-0.66	165.5	-114.2

参考のため最適解の探索過程を示す。焼きなまし法による最適解探索と安全性、利便性、経済性及び評価指標Mの値の変遷の様子を図 5.7〜図 5.10 に示す。図 5.7 から順にバランス型、安全性重視、利便性重視、経済性重視の際の探索の様子である。図中の (a) は評価指標である安全性（Risk）、利便性（Value）、経済性（Cost）を 3 軸にとり、式 5.1 で計算される評価指標 M の値で色付けしている。(b)、 (c)、 (d) 、(e) は、それぞれ探索の試行に対する Risk、Value、 Cost、 評価指標 M の変化である。

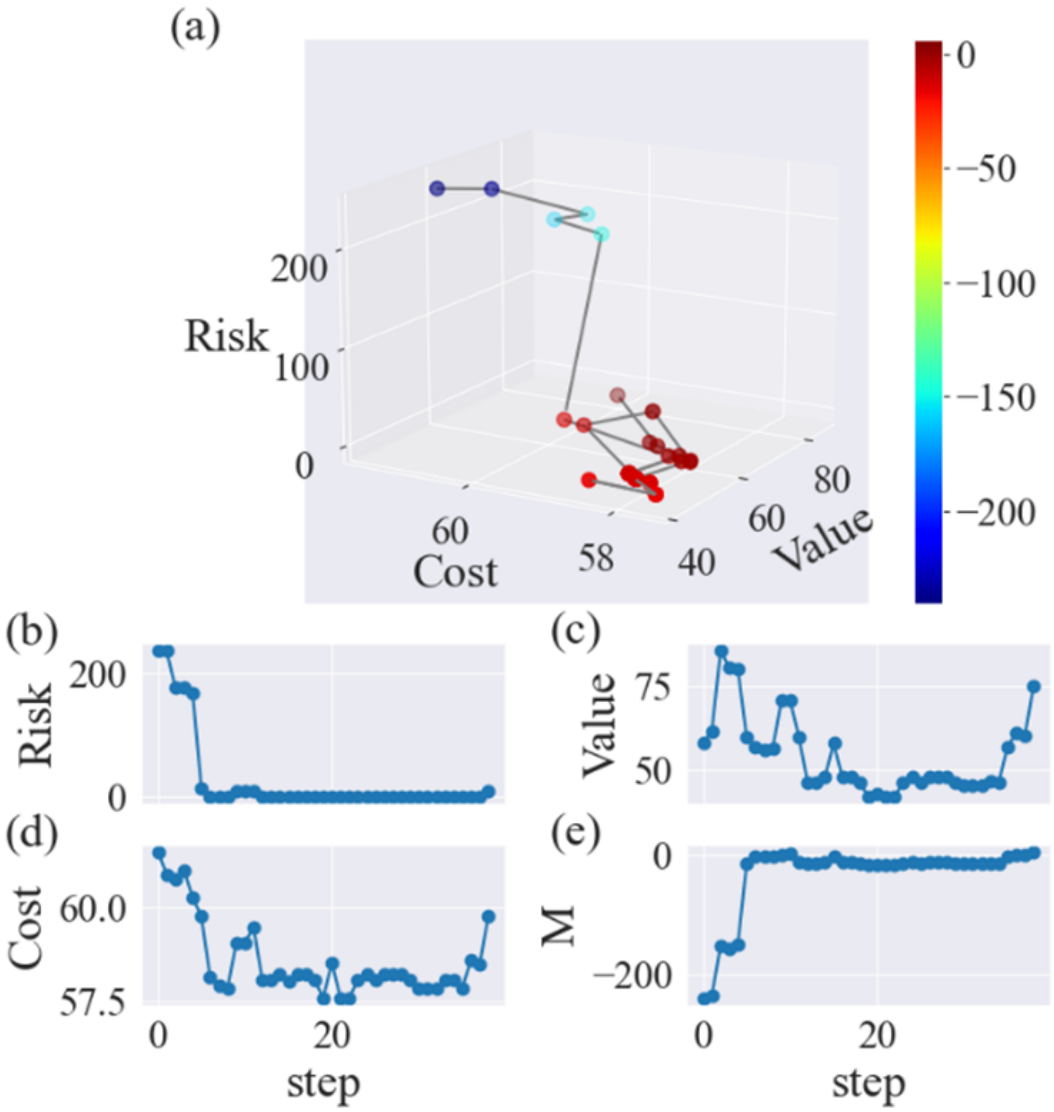

図 5.7: バランス型条件における最適解の探索過程

Fig. 5.7: Search process for optimal solution under balanced conditions

5.6.4　モビリティサービスの導入に向けた最適解の吟味

前節で算出した最適解から実運用すべきモビリティサービスを特定する。バランス型の最適解は、最大 4 名の乗車定員ではなく、3 名の車両を投入する解である。単純な費用便益計算では、最

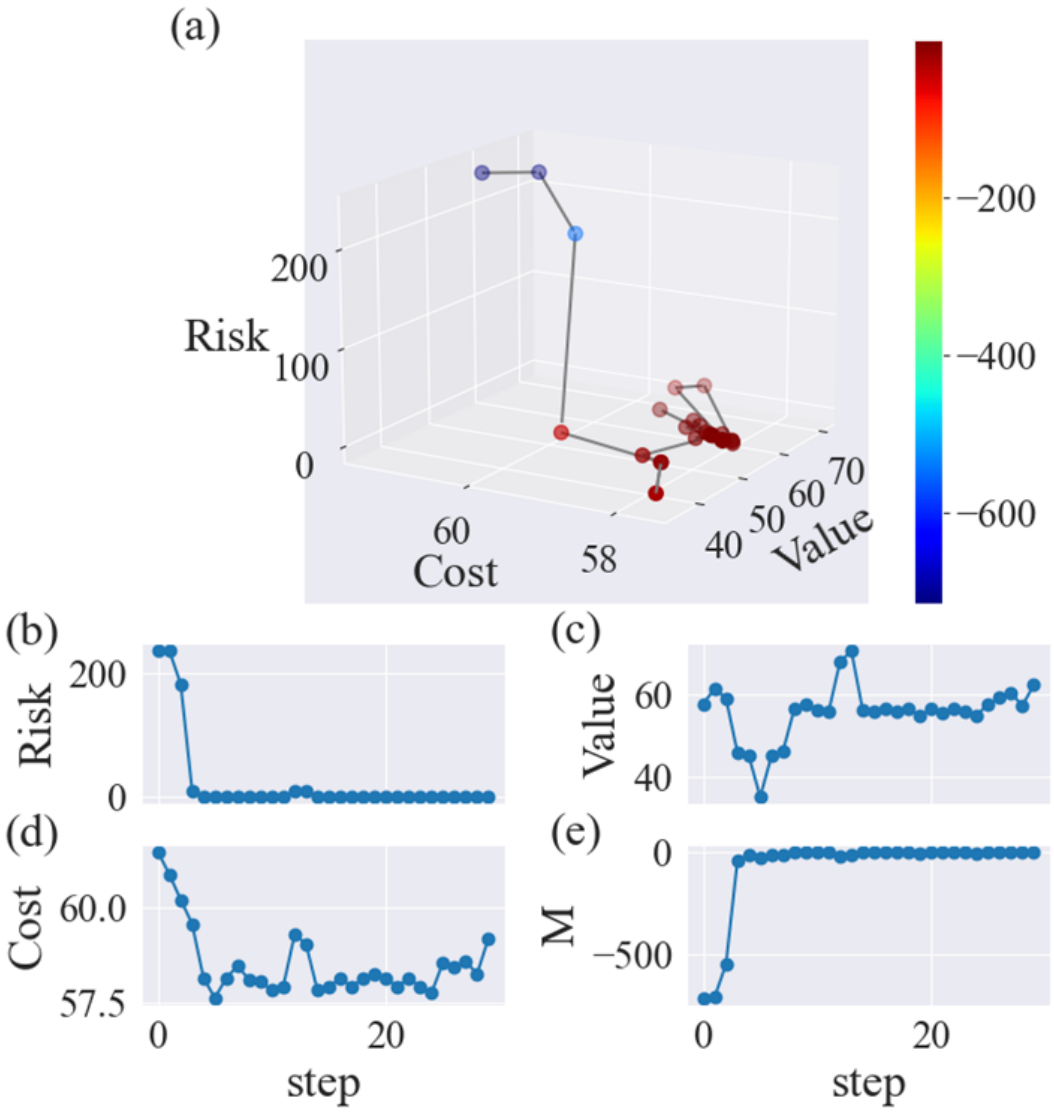

図 5.8: 安全性重視条件における最適解の探索過程

Fig. 5.8: Search process for optimal solution under safety-oriented conditions

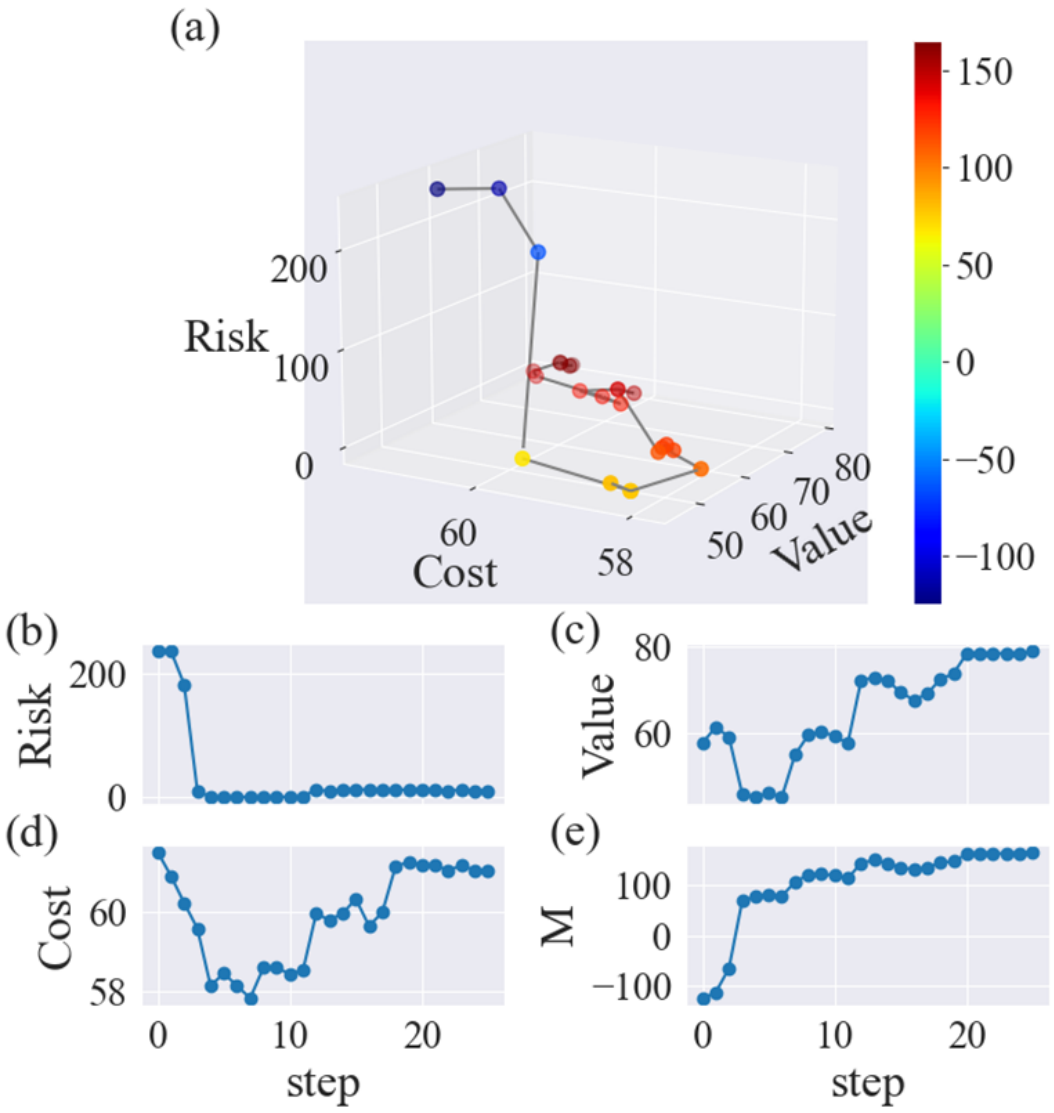

図 5.9: 利便性重視条件における最適解の探索過程

Fig. 5.9: Search process for optimal solution under value-oriented conditions

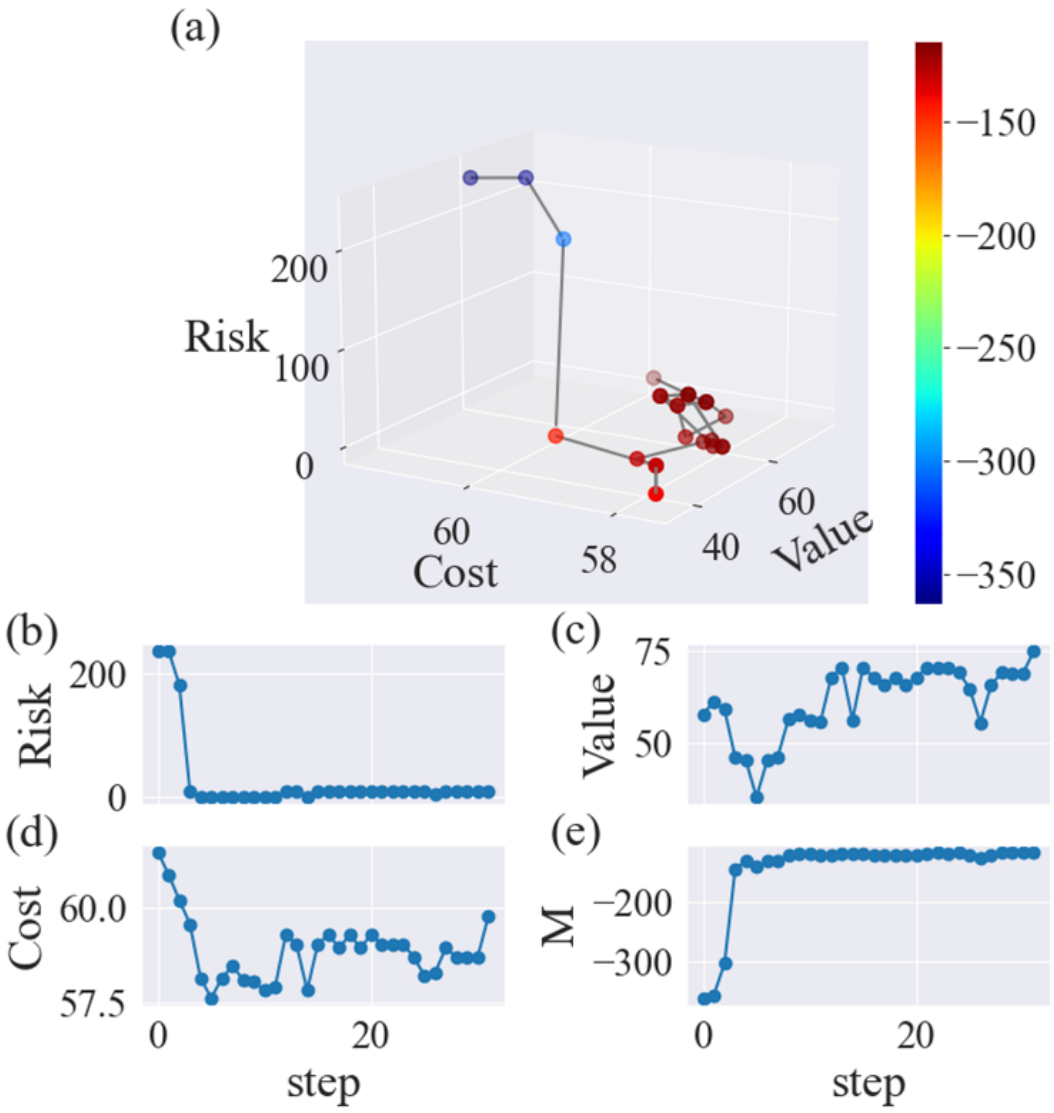

図 5.10: 経済性重視条件における最適解の探索過程

Fig. 5.10: Search process for optimal solution under cost-oriented conditions

大定員での運行の便益が高いと判断しがちであるが、乗車定員を増やすと車両が大型化し総重量が増え、安全性の指標が悪化するにもかかわらず空席で走らせることが多くなり利便性の指標が良化しないため、3 名の乗車定員の車両の運行が最適であると判定している。安全性重視の最適解は、最高速度が低く小型であるため、より安全である。利便性重視の最適解は、バランス型と比較して希望短縮時間を小さくする解となっている。すなわち、乗車判定時に徒歩で移動するのと同等もしくは少しでも短縮できる人が乗車する解が最適となる。これは現実世界では、予約システムで積極的に空車を提示して乗車を促し、利用者に乗車のメリットを訴求することで実現を目指すことになる。これに対し経済性重視の最適解では、希望短縮時間は、バランス型と同等だが乗車定員は 4 名の比較的大きな車両を投入すべきという結論となっている。表 5.5 で示した M の値が正の値となるのは利便性重視の運行である。このとき意思決定者は、利便性重視の運行を行ってよいか、安全性の指標を確認し、リスク発生場所を特定し追加対策することで、より好ましい運行が探索できないか検討する。

図 5.11 は、ヒヤリハットの発生場所を可視化したものである。図 5.11 の左（最大速度 10km/h 運行時）のヒヤリハット発生地点は、交通量と道路状況を勘案したリスク箇所として現場での感覚と一致している。

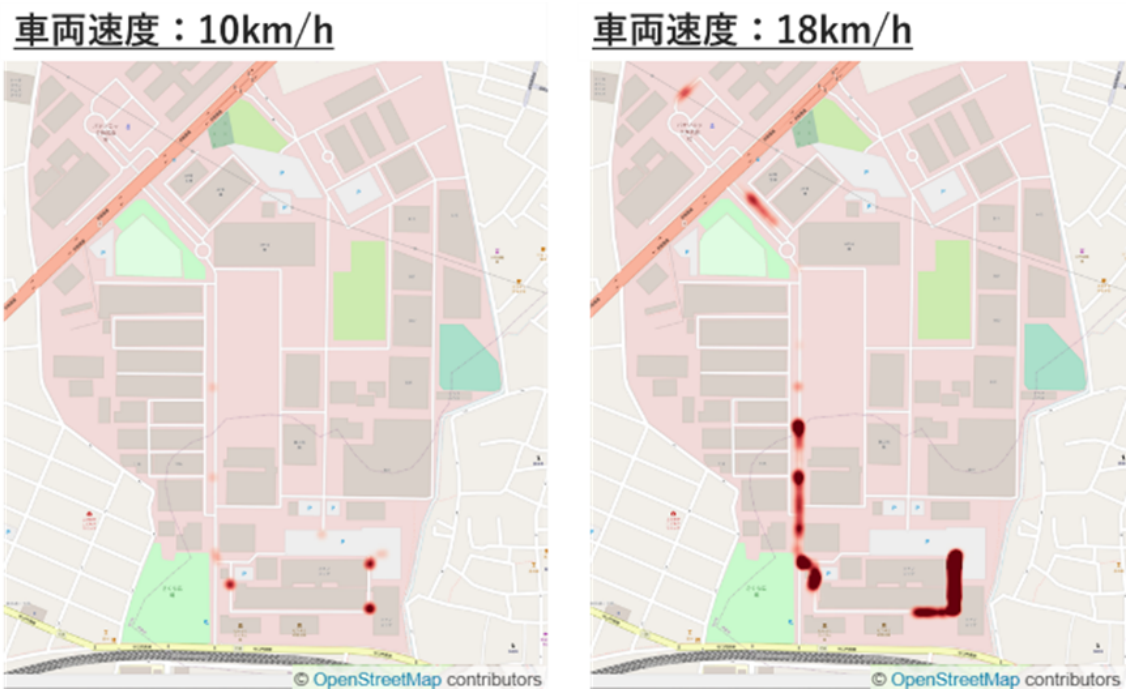

図 5.11: ヒヤリハットの発生場所

Fig. 5.11: Location candidate of the traffic accident

意思決定者は、それぞれの解の安全性、経済性、利便性の指標の値やヒヤリハット事象の発生確率が高い場所の情報を参考にし、当該エリアに導入するモビリティサービスや利用促進策を決定する。また、ヒヤリハット事象の発生頻度や発生しやすい場所は、シミュレーション結果で確認できる。例えば、図 5.11 の右図で示されるヒヤリハット発生場所の最高速度を制限した運行計画を再立案し、再びシミュレーションにより仮説の評価を行って納得できる運行計画を決定する。

5.7　本章のまとめ

本章では、人の活動データの分析に基づき、安全性、利便性、経済性を同時に満たすモビリティサービスを開発・改善する手法として、企業内の建物の出入り口のデータを活用し、人の移動を再現し、自動運転車両による運行サービスを評価する手法を示した。安全性の指標としては、人と車両の近接回数から算出した事故の損害額、経済性の指標としては、移動時間の短縮効果、経済性の指標としては車両の運行に関するコストを用いた。実際に現実に発生する人流データを再現し、これら 3 つの指標を同時に評価する手法は、著者の知る限りこれまで報告されていない。人の移動の再現は高コストであるため 30 ノードを用いた並列処理での実行時間を評価し、シングルプロセスで 9287 時間要する評価を 81.12 時間で実行できることを確認した。焼きなまし法を用いて、バランス型と安全性、利便性、経済性を重視した条件で最適解を探索し、1 ノードを用いた処理で、174.1 時間から 121.9 時間で探索できることを示した。すなわち、わずか数並列で 1 週末程度の実用的な時間で探索できることを確認した。

シミュレーションで抽出された最適解は、安全性の指標を考慮しているため、単純な投資対便益分析では選定されない解となっている。シミュレーション結果で明らかになったヒヤリハット発生場所は、現実の運用での観測とかなり一致している。

本研究で示した手法は、費用便益の分析に加え、計算機の能力を用いて人や車両の動きを再現することで安全性視点を強化し、車両の空車走行による非効率性等、経済性の評価を含めたサービスの全体像を把握・評価する手法である。

実世界で発生する人流をサイバー空間で再現し、複雑に絡み合うサービス因子の関係性を実装したシミュレーションで安全性を担保しつつ、利便性や経済性の均衡点を解析する手法は、自動運転車両が様々なエリアに実装されていく中で重要になると考える。大型車両を導入する、走行路を変更する、最高速度を上げる、新たなステーションや安全性の担保のため、専用レーンやガードレールを設置する等、実世界に実装するには、危険であったり、高コストであったりするような施策の評価は、サイバー空間での再現が容易である。

特定のエリアに対し、自動運転を含むモビリティサービスの導入を決定する意思決定者にとって、ヒヤリハット事象が発生しうる場所や発生確率は大きな関心事となる。本章で示したサービス設計・評価手法は、費用対便益の評価に、ヒヤリハット事象の発生度合や損害額といった安全性の指標を加えて同時評価を行い、インタラクティブにサービス案を提示することで、エビデンスベースで意思決定を支援するものである。この手法は、今後、新たな自動運転機能を備えた様々なモビリティを特定のエリアに導入する際に有効である。

今後は、自走配送ロボットを用いた公道でのラストマイル配送サービスの設計と評価に本提案手法を適用し、投入するサービスやエリアに合った適切な安全性、利便性、経済性の指標や選定車両・台数、適用エリアの拡大等、サービスの段階的投入を検討できる手法への拡張やマルチモーダルな交通の乗り換えへの適用を通じて、ひな形化、一般化を推進したい。

第6章　モノを運ぶサービス設計・評価への適用

本章では、3 章で説明した手法をモノを運ぶモビリティサービスの設計・評価に適用した事例を説明する。本章の内容は文献 [74] の発表内容に信号時相論理による評価指標の記述を追加したものである。

6.1　導入

自動配送ロボット（以後 ADR:(Autonomous Delivery Robot) と略記）の公道走行は、改正道路交通法（2022 年 4 月 19 日に成立）が施行される 2023 年度以後、遠隔操作型小型車として届け出ることで実施可能となる。このため、ADR を用いた配送サービス（以後 ADS:(Autonomous Delivery Service) と略記）の社会実装が進み、既存の配送サービスにおける人手不足の解消効果などが期待されている。日本では 2023 年 4 月に改正道路交通法が施行され届け出制による公道走行が認められており、神奈川県の藤沢 SST と東京都丸の内でこの制度を利用した実証が進められている [18]。図 6.1 は、実験の様子を示す写真である。

ADS の設計においては、サービス提供領域の広さ、投入する ADR の種別、投入する機体の台数、カバーする配送需要の量等、様々な要因が関連するため、サービスの全体像を把握し、最適解を選定するには多くの時間を要する。

モビリティサービスを特定エリアに導入する際の一連の工程は図 3.1 に示した通りである。まず、新たなモビリティサービスを特定の地域（エリア）に導入する場合、要求される事項を指標化する。この時、詳細な性能指標を直感的に共通理解可能な信号時相論理 (STL) で表現する。次に、当該エリアの需要、交通量の調査を行い、機体の種別や投入台数といった運行候補を複数パターン選定する。その後、配送元及び配送先ステーション配置、走行路、運行数、運行形態といった導入するモビリティサービスの仮説を立案する。その後、これらのサービスを安全に実現するための安全設計を行う。この過程で不安全な運行を排除する。例えば、機能安全設計が不十分な

図 6.1: 藤沢 SST での配送サービスの例

Fig. 6.1: Autonomous delivery service in FujisawaSST

大型車両を歩車分離のない住宅地で降雪時に高速運行させるといったサービス仮説はこの工程で棄却する。この後、安全性、利便性、経済性の観点から妥当か計算機シミュレーションを用いた定量評価を行い、導入するモビリティサービスの仕様について意思決定を行う。

6.1.1 信号時相論理（再訪含む）

2.5.3 で説明した通り、信号時相論理（Siganal Temporal Logic: STL）[45] は、動的なシステムの状態軌道等の「信号」に対して、時間制約を含む複雑な仕様を柔軟かつコンパクトに記述することができる形式手法の一つである。

STL 式は、時間に伴い変化する信号の様々な性質を記述することができる。例えば、STL 式 $\mathcal{G}_{[0,\infty]}(x - 300 > 0 \rightarrow \mathcal{F}_{[0,3]}(10 - y < 0))$ は、「x が 300 より大きくなったとき、いつでも 3 秒以内に y が 10 より小さくなる」ということを意味する。構文や意味論の詳細は、2.5.3 節を参照いただきたい。

STL の二値意味論（Boolean semantics）では、信号 σ の STL 式 ϕ の充足は、他の形式論理同様、$\sigma \vDash \phi$ で表される。あるいは、STL の堅牢意味論（robust semantics）では、効率的な計算アルゴリズムで計算可能とするために実数値を用いて、$[\sigma, \phi] \in \mathbb{R} \cup \{-\infty, \infty\}$ で示される。これにより、性質の充足度や違反度を定量的かつ詳細に測定することができる (文献 [34] 参照)。つまり、STL 式 ϕ は、その堅牢な意味論によって目的関数 $[\sigma, \phi]$ として考えることができる。本章では、時間的空間的に変化する指標として、路上駐車時間の制約記述に STL で定義されている表現を採用する。これにより複数の意思決定者の間で指標に関する理解が深まる事が期待できる。これは、図 3.1 のプロセスにおいて、意思決定者を支援できる。意思決定者は、安全性、経済性、利便性等の様々な指標でモビリティサービスを評価したいと考えている。これらの評価指標を STL 式で表現すると、(1) それほど手間がかからず、(2) 評価指標の意味がより直観的に伝わりやすくなり、(3) 多くの特性を表現できるため有益である。時間的に空間的に変化する指標は、C 言語等でも表現できるが、この場合、意思決定者は長い C 言語の記述を読む必要があり効率的でない。すなわち、シミュレータに実装された評価指標の意味を意思決定者 (通常は C 言語を読まない人) に説明し共有することは困難となる。

6.1.2 本章の構成

本章では、この手法の仮説の評価の工程を拡張し、モビリティサービス導入の意思決定者が、シミュレータを用いてインタラクティブに最適解を選定する手法を提案する。特定地域での ADS の最適解の探索に適用し、複数の被験者による評価で提案手法の有効性を確認する。

次の節で ADS の設計・評価手法への要求を述べる。6.3 節でインタラクティブな設計・評価手法を提案し、6.4 節で特定地域での配送サービスを題材に本手法を適用した評価について述べ、6.5 節でまとめる。

6.2 自動配送ロボットによる配送サービス設計・評価への要求

ADR を用いた配送サービスの設計・評価に対する要求事項を述べる。

6.2.1　サービスの安全性

人や自転車など様々な交通参加者との混在状況の住宅地などで自由軌道を走行し、自律移動を行う ADR の運行の安全性は担保されなくてはならない。すなわち、他の交通参加者との事故を回避し、歩行者を優先し、さらには歩行者を怖がらせない安心な挙動を行うといった安全性の担保が必要となる。

6.2.2　サービスの事業性（経済性、利便性）

ADR の日本の公道での最高速度は時速 6km と定められている。単位面積当たりの配送需要が少ない地域では少ない台数、需要が多い地域では多数の機体を投入する必要があると想定できる。

人手による配送と異なり、ADS では、休日や夜間など人手の確保が難しい時期や時間帯でも配送サービスを実現でき、より多くの潜在需要を取り込める可能性がある。また、ADR の荷室と荷主の荷物の関係は管理されるため、配送時間を数分単位で指定できる等、エンドユーザにとっての利便性が向上する可能性がある。利便性が高まった配送に対して、適切な配送価格を設定することで配送サービスの収益性が高まる可能性がある。

6.2.3　安全性、経済性、利便性の観点での最適解の探索

ADS の設計は、安全性、経済性、利便性のバランスで決定する必要がある。ADR の運行場所と時間帯によって、人等との混在度合いが変わってくることが予想される。人等との混在状況により最高速度を減速させると安全性は向上するが、配送件数が減り経済性が悪化し、オンデマンド配送の正着率が下がり利便性は悪化する可能性がある。

幅の狭い歩道の通行を回避する走行路を選択すれば、安全性は向上するが経済性が悪化する。相対速度が大きい自動車が往来する幅の広い道路の横断時などセンサー能力だけでは不十分な場所で遠隔オペレータによる安全確認を行えば安全性は担保出来る。このような横断を多数発生させる配送を行えば、多くの配送需要を取り込める可能性はありエンドユーザの利便性が向上するが、遠隔オペレータの負担が増え、その分運行効率が下がり、1 名の遠隔オペレータで多数の機体の監視操作を行うことができなくなり、結果として経済性が悪化する。

ADR に高性能なセンサーを搭載し、認知・判断・操作のすべての工程において冗長化を行えば、機体の挙動の信頼度及び安全性が向上する。しかし、配送サービスを担う小型低速の ADR に高速移動する自動車と同じ冗長度を組み込めば、自動車の自動運転車両と同等のコストが必要となり経済性が悪化する。

すなわち、ADS の設計では、安全性、経済性、利便性を左右するモビリティサービスの決定因子の多数の組み合わせの中から最適解を探索する必要がある。

6.2.4　時間的空間的制約

ADR は車両ではなく歩行者扱いであるため駐車禁止の概念は適用されないが「道路での滞留時間」と「店頭待ち時間」は住民のサービス受容性に影響する。特定地域に多くの ADR を投入しすぎると路上での待ち時間が増え交通の妨げとなったり、別の事故を誘発する可能性もある。このため路上での停止時間は関心事の一つである。

6.2.5　サービス全体像の把握と選択理由の説明可能性

前述のように、ADS のサービス決定因子は複数存在し、その膨大な組み合わせがある。このため、サービスの可能性の全体像が把握しにくい。しかし、サービスの導入を行う意思決定者は、サービスの可能性の全体像を把握した上で、自らが選択するサービスの最適解の選択を説明可能とし、導入するモビリティサービスの仕様についてステークフォルダと合意形成していく必要がある。

6.3　インタラクティブな設計・評価手法の提案

前節で説明した要求を満たすため、サービスの導入を行う意思決定者がミクロ交通系シミュレータを用いたシステムで最適解を選択する手法を提案する。

6.3.1　意思決定者とのインタラクションを含む処理フロー

モビリティサービスの決定因子の組み合わせは膨大となる。シミュレーションパラメータの組み合わせのすべてを全探索すると、例えば、6.4 節で説明するような特定地域における ADR による数時間程度の荷物の配送サービスの組み合わせをすべて実行すると 100 年以上のシミュレーション時間が必要となり、現実的な時間で最適解を探索できない。このため、提案システムでは、最適解の候補となるシミュレーションパラメータから、いくつかのシミュレーションパラメータを変動させ、それぞれのシミュレーションを CPU のコアの数だけ並列実行する。このシミュレーションの実行単位をシミュレーションバッチと呼ぶ（以後 SB と略記）。システムが SB の実行結果のパレートフロントを安全性、経済性、利便性の評価軸上に可視化し、モビリティサービスの意思決定者が可視化されたパレートフロントの中から最も好ましい解を選択する。

システムは、選択された好ましい解に基づきシミュレーションパラメータセットの中からいくつかのパラメータについて改良し、次の SB を生成する（図 6.2）。図中のバッチは SB を指す。

以上の操作を繰り返すことで、意思決定者の戦略を反映しながら納得性の高い解を効率的に探索する。

6.3.2　シミュレーション環境

意思決定者との対話環境は、JupyterLab [4] を利用している。シミュレーションバッチの実行部は、Python [12] 及び SimPy [13] を用いて構築した。後述するパレートフロントの提示には、Plotly [10] を用いている。

6.3.3　パレートフロントの提示

システムは、SB の終了毎に、安全性、経済性、利便性の評価軸でパレート最適解を抽出する。すなわち、特定のシミュレーションの結果を他のシミュレーション結果と比較し、安全性、経済性、利便性の評価指標のうち、どれかは良化している場合、パレート最適解として出力画面上に表示する。いずれも優れていない場合、パレート最適解から除外し表示しない。図 6.3 に出力画面の例を示す。図中の Risk、Cost、Value は、それぞれ安全性、経済性、利便性の評価軸を表しており、SB の世代は色で区別されている。

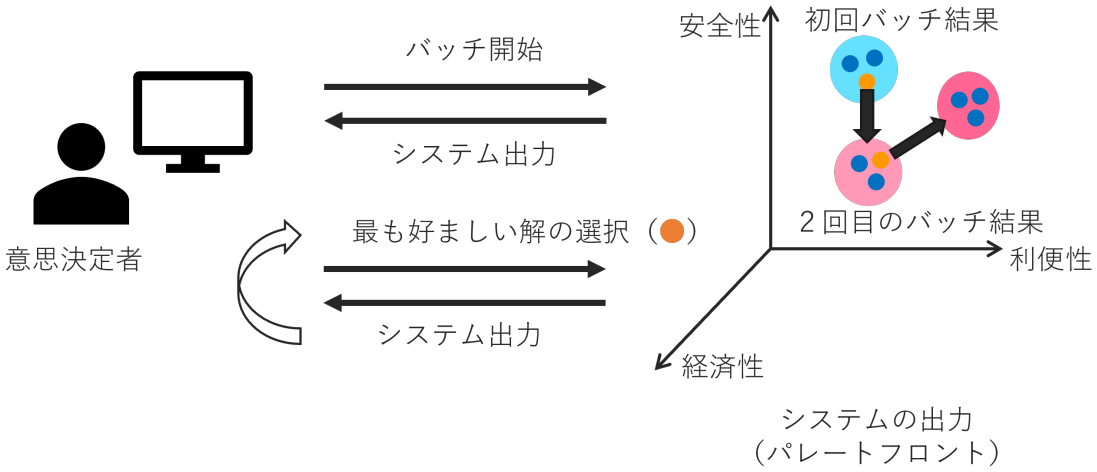

図 6.2: インタラクティブな設計・評価手法

Fig. 6.2: The Interactive Design and Evaluation Method

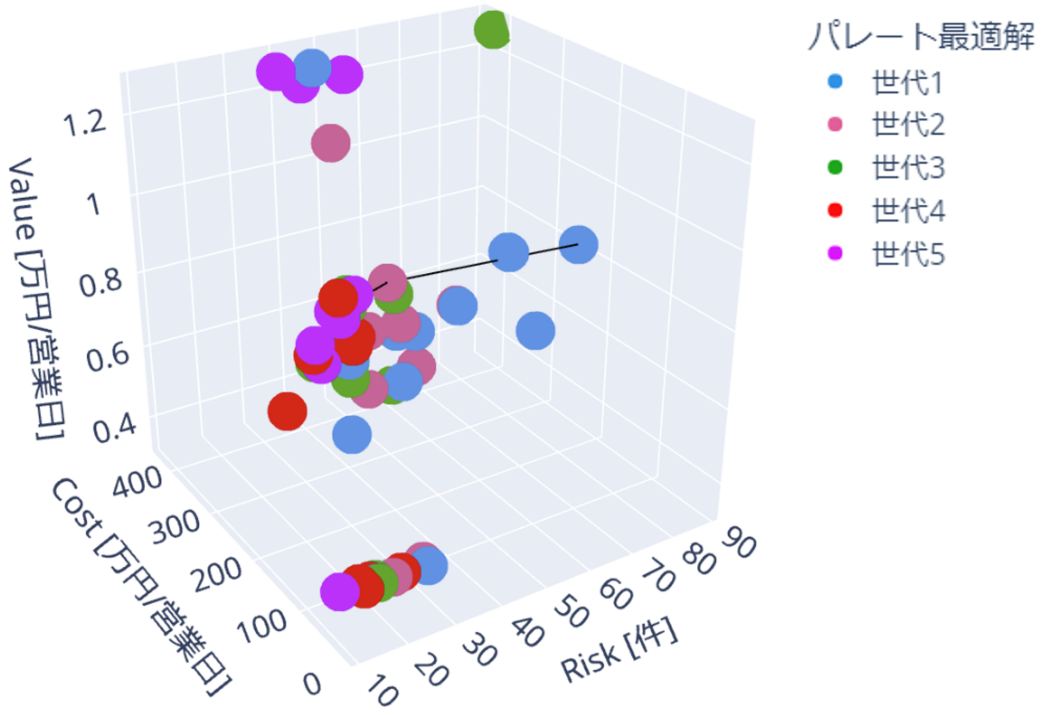

図 6.3: システムの出力画面（パレートフロント）

Fig. 6.3: System Output (Showing the Pareto front)

6.3.4　シミュレーションパラメータの表示

意思決定者は、出力画面に表示されたパレートフロント上のマウスのポインタを操作することで、それぞれのシミュレーションのパラメータの値と安全性、利便性、経済性の評価値を確認できる。３次元の出力画面はマウスをドラッグすることで視野角を変更できる。また、スクロール操作で拡大縮小表示できる。図 6.4 は、図 6.3 と同じ SB の角度を変えて拡大表示し、マウスオーバーによりシミュレーションパラメータを表示させた様子を表したものである。

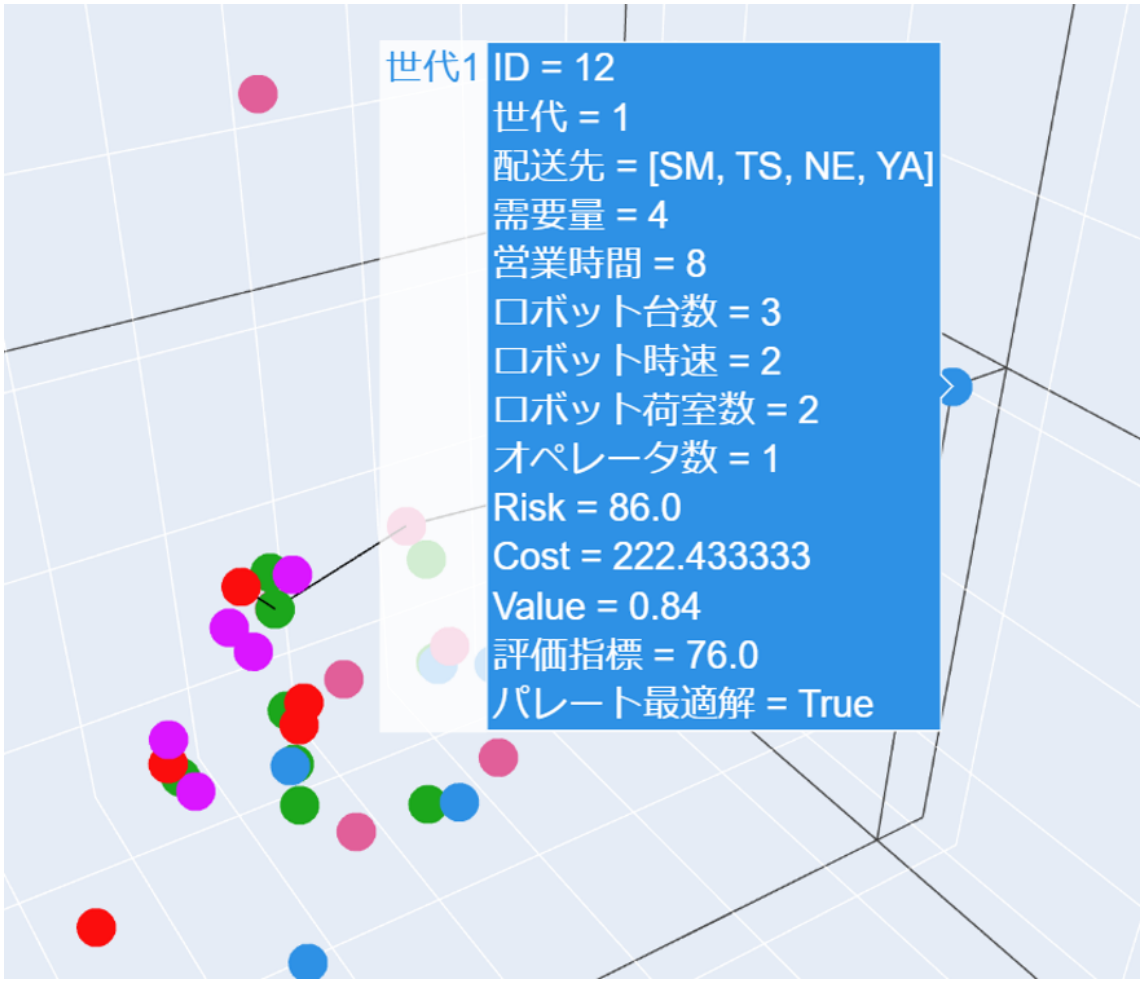

図 6.4: マウスオーバー時の表示画面

Fig. 6.4: The screen shot showing simulation parameter on mouse

6.3.5　最も好ましい解の提示と次回バッチのシミュレーションパラメータの算出

意思決定者は、出力画面に表示されるパレートフロントのうち、最も好ましいと考えられる解を一つ選択し、マウスオーバーし、そのシミュレーションを示すＩＤを確認する。意思決定者が、シミュレーションを示すＩＤをシステムに入力すると、システムは、A: 選択したシミュレーションと B: 評価指標が最良のシミュレーションのパラメータの差分から、次の SB で実行するシミュレーションパラメータを生成する。次の SB のパラメータ生成の例を図 6.5 に示す。x_1, x_2 はシミュレーションパラメータを示す。B から A の方向に拡張した範囲を算出し、その範囲内で次世代の

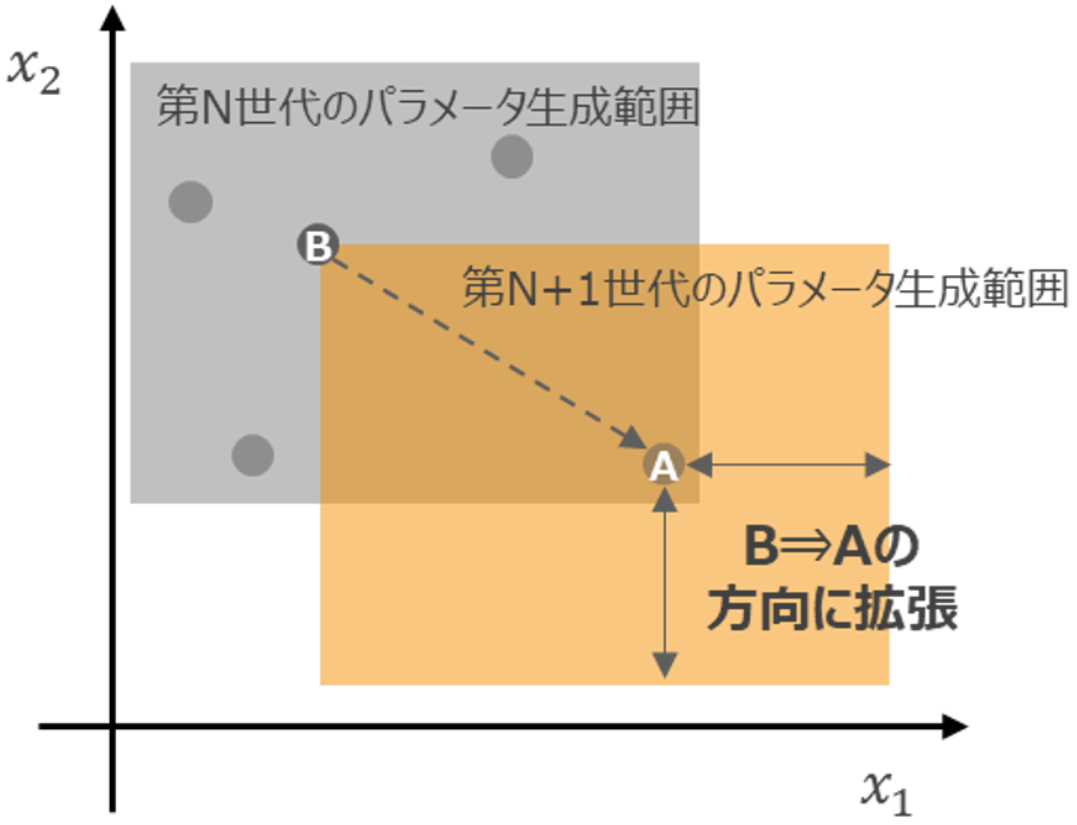

図 6.5: 次回バッチのシミュレーションパラメータ算出方法

Fig. 6.5: Method for calculate simulation parameters for the next simulation batch

SB のサイズのシミュレーションパラメータの組を、乱数を用いて生成する。本章では、SB のサイズは 16 を採用している。なお、評価指標には、安全性、利便性、経済性の各出力についてシミュレーションごとに比較して順位付けし、その合計値を利用している。

6.3.6　評価の終了と選択した解のトラジェクトリの表示

意思決定者が評価の終了を入力すると、システムは各 SB において、意思決定者が選択した好ましい解のトラジェクトリを安全性、利便性、経済性の軸上に表示する。図 6.6 は、その画面を示した図である。この画面により、意思決定者は自分の選択の履歴を確認できるとともに、探索空間の全体像の把握につながる。さらに、評価不足な項目を気づくこともでき、その場合はパラメータや探索戦略を変更し再評価することができる。

6.3.7　モビリティサービスの導入決定もしくは再評価

意思決定者は、実行可能で十分な最適解が得られたと判断すれば選択したモビリティサービスの社会実装を進める。前節までに説明した評価の中で、さらに詳細なシミュレーションや、シミュレーションモデルの見直しの必要性を感じた場合、図 3.1 の「特定エリアの運用候補を複数選出」に戻って再評価を行う。

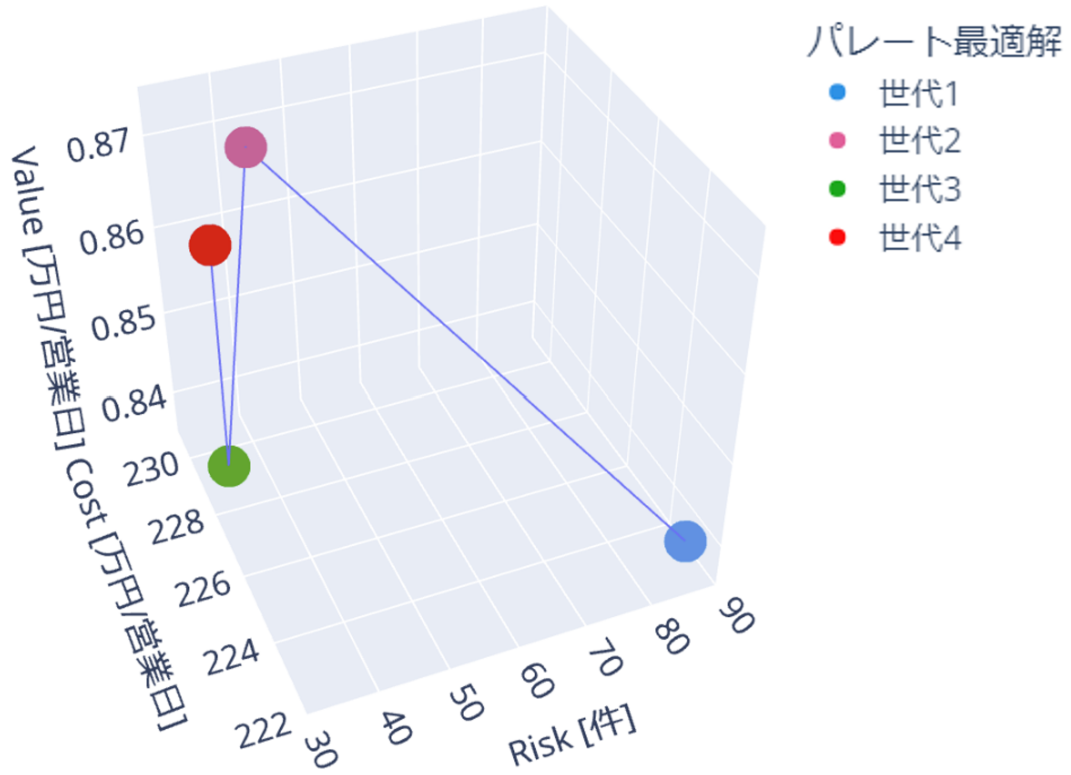

図 6.6: 意思決定者が選択した好ましい解のトラジェクトリ

Fig. 6.6: The trajectory of selected preferable solution on each generation by the decision maker

6.4　特定地区での配送サービス設計を題材とした提案手法の評価

藤沢地域での ADS の設計を題材として、6.3 節で説明した手法で最適解を探索する。複数の被験者による評価により本手法の有効性を確認する。

6.4.1　藤沢地域の ADS

藤沢 SST [2] は、2014 年に街開きを行ったスマートシティで、JR 東海道線藤沢駅からバスで 15 分程度の場所に 19 ヘクタール 東京ドーム 5 つ分の敷地に郊外型住宅中心の複合開発が行われたエリアで約 1900 名が居住している地域である。 藤沢 SST では、2020 年 12 月より自動配送ロボットを用いた実証実験が行われている [17]。

異なる需要量を発生させる複数の店舗の配送サービスの存在を仮定し、その配送需要に応じて複数の ADR が配送を行う状況を模倣し、安全性、利便性、経済性の指標で評価し、インタラクティブに最適解を探索する。

6.4.2　シミュレーションパラメータ

藤沢 SST の住民に配送すると仮定する店舗候補と配送先を図 6.7 に示す。図中の青丸は配送元となる店舗であり、赤色の丸が配送先である。住民が一定確率で当該店舗に対して毎週配送を依頼する、Weekly Active User（以後 WAU と略記）となると仮定する。WAU の発生確率と配送先の世帯数から 1 日当たりの配送発生件数に換算したものがシミュレーションパラメータとなる。この配送需要を ADR が稼働営業時間内で配送する。この営業時間、配送サービスに投入する ADR の台数、ADR の速度、荷室の数、投入する遠隔オペレータ数もシミュレーションパラメータである。

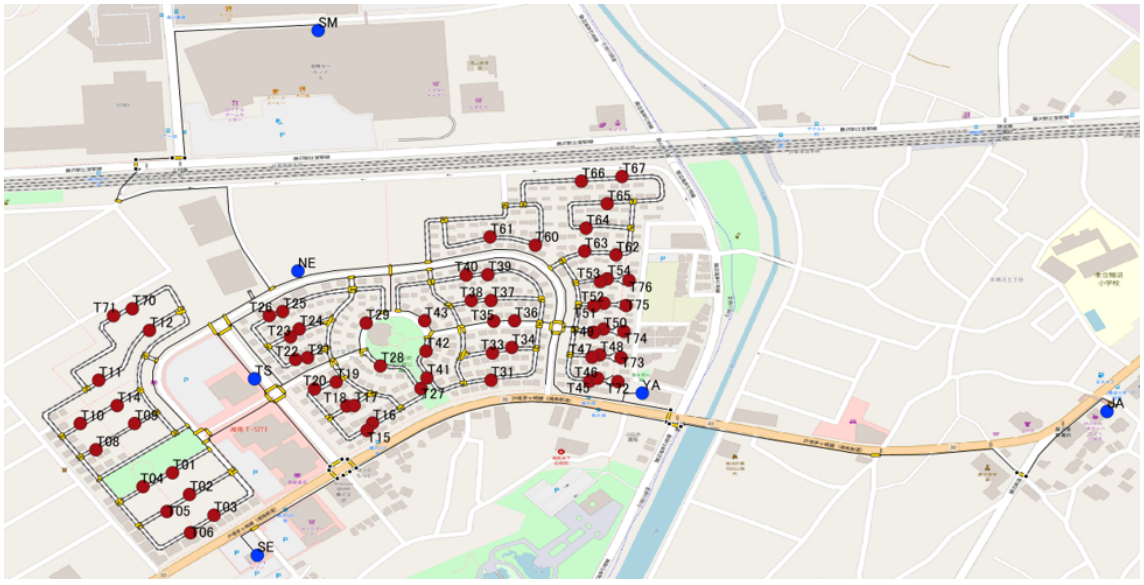

図 6.7: 藤沢地域の配送元（青丸）配送先（赤丸）走行路

Fig. 6.7: The delivery source(blue) and destinations(red) in Fujisawa SST

遠隔オペレータは、ADR が走行中に経験するイベントを処理するために操作を行う。イベントの発生と遠隔オペレータの操作時間は、我々が同地域での実証の経験をベースにポアソン分布で

生成する。イベントの具体例と操作時間については、駐車車両の回避等、交差点の横断時の操作であり、駐車車両の回避等については、実証で観測された遠隔オペレーションの発生頻度をポアソン分布で再現し、操作時間は実証経験した平均時間（定数）とする。同地域に複数台の ADR が稼働している場合、1 台の ADR の操作に対し 1 人の遠隔オペレータが対応し、全遠隔オペレータが ADR を操作中の場合は残りの ADR は停止する。信号機のない交差点を通過する際は、遠隔オペレータが操作して交差点を横断し、その操作時間は ADR に設定された速度で交差点を移動する時間とする。信号機のある交差点を通過する際の操作時間は、信号機のサイクルの半分の値（定数：54 秒）とする。また、ADR への荷物を積み下ろしにかかる時間は、注文が入った商品をピッキングして積載する配送元側、ADR が配送先に到着して顧客が商品を取り出す配送先側とも、一定時間（定数：300 秒）とする。シミュレーションパラメータの一覧を表 6.1 に示す。パラメータの組み合わせの総数は 36,288,000 通りとなる。

表 6.1: シミュレーションパラメータ一覧

Tab. 6.1: List of Simulation Parameters

分類	項目	パラメータ	数
需要	配送元 (集荷先)	(SM, SE, JA, TS, NE, YA) の 6 拠点の組み合わせ	63
	需要量 (WAU)	{1, 2, 3, 4, 5, 6, 7, 8, 9, 10} %	
		→ {1, 3, 4, 6, 7, 9, 10, 11, 13, 14} 件/日	10
サービス	営業時間	{1, 2, ⋯,24} 時間	24
	ロボット台数	{1, 2, 3, ⋯, 9, 10}	10
	ロボット時速	{1, 2, 3, 4, 5, 6} km/h	6
	ロボット荷室数	{1, 2, 3, 4}	4
	積荷積卸時間	定数 300 秒	1
	オペレータ数	{1, 2, ⋯, 9, 10} 人	10
	遠隔介入頻度	平均発生間隔 449 秒のポアソン分布	1
環境	青信号点灯周期	定数 54 秒	1

6.4.3 シミュレーション結果の評価指標

シミュレーション結果は、以下に示す、安全性、経済性、利便性の指標で評価する。これら 3 指標のパレート最適解を 3 次元グラフで可視化する。

安全性の指標

安全性の指標 R は、当該地域で遠隔オペレータの介入の頻度をポアソン分布で近似して発生させたリスクイベントの合計数として、次の式で示す指標を採用する。

$$R = -\{\sum_{i=1}^{n} W(i)\}$$

W(i) は、機体 i が営業時間内に経験するリスクイベントの回数である。n は、ロボット台数である。

経済性の指標

経済性の指標 C は、ADR の減価償却費用を合計したものと、遠隔オペレータを雇う費用と、店舗の初期費用の減価償却費と運営費用とを合計した値で、次の式で表すものを採用する。

$$C = -\{\sum_{i=1}^{n} C_{adr}(i) + \sum_{j=1}^{m} C_{ope}(j) + \sum_{k=1}^{p} C_{env}(k)\}$$

$C_{adr}(i)$ は、機体 i の減価償却費（日割り）であり、$C_{ope}(j)$ は、時給で雇用される遠隔オペレータ j を営業時間分だけ雇う費用であり、m は、オペレータ数である。$C_{env}(k)$ は、店舗の初期費用の減価償却費（日割り）と営業時間中の運営費用（日割り）の合計値である。k は発送元の拠点を示す。例えば、配送元 SM と TS の 2 拠点の配送元からの需要を ADR が配送する ADS のシミュレーションの場合、p は 2 となる。

運用者視点での利便性の指標

運用者視点での利便性の指標 V_{ope} は、営業時間中に完了した配送数に各店舗が設定する配送料を掛けた合計金額として次の式で表すものを採用する。

$$V_{ope} = \sum_{i=1}^{n} \sum_{j=1}^{m} V_{adr}(i,j)$$

$V_{adr}(i,j)$ は、機体 i が営業時間中に店舗 j への配送を完了した数である。

6.4.4 インタラクティブな最適解の探索手法の評価

複数の被験者がインタラクティブに最適解を探索し、本システムの有効性を評価する。被験者は、ADS を導入する意思決定者であり、当該地域への ADS を 1 週間程度の時間で決定する立場であることを前提として、6.3 で説明したインタラクティブな評価手法で 10 回を上限として複数バッチ回、パレート最適解を探索し、納得できる最適解が得られた時点で探索を終了する。

6.4.5 評価指標

評価指標は選択した ADS を導入する意思決定者が最適解を得る過程として適切かという観点で、以下の 5 つの質問を抽出した（表 6.2）。

最適解に到達する時間が実用的か（実行時間）

複数のバッチ回数分の最適解の探索は、実用的な時間で探索できたかどうか、を実行時間に関する指標とする。なお、実用的な時間とは、被験者が意思決定に許容される時間として概ね 1 週間の時間が与えられるとして、このシミュレータを用いた作業に使える時間が妥当かどうかで判断を仰いだ。

表 6.2: 評価アンケートの質問

Tab. 6.2: Questionnaire for the Evaluation

No	項目	質問
1	実行時間	1 週間の時間を与えられ最適なサービス仕様を導出することになったとして、本手法で結果を算出する時間は実用的だったでしょうか。
2	UI	各世代において最も良いと思う解は選択しやすかったでしょうか。
3	納得性	本手法を利用して決定した最適解は納得できるものでしたか。
4	全体把握性	本手法により十分網羅的に候補を評価できたと考えますか。
5	説明性	本手法を利用して決定した最適解についてその決定理由を説明できますか。

最適解の選択しやすさ（UI 評価）

各 SB の終了時に表示される出力画面で、最適解が選択しやすかったかどうか、を UI に関する指標とする。

最適解の納得性、全体把握性、説明性

本手法で得られる最適解は、局所最適解である可能性があるが、選んだ最適解は納得できるか、また、サービス可能な解空間の全体像が把握できたか、選択理由を他のステークフォルダに説明できるか、という評価指標をそれぞれ、納得性、全体把握性、説明性の評価指標する。

6.4.6 評価

11 人の被験者に本手法を用いた最適解の探索を実施してもらい、前節で説明した評価指標について 5 段階評価のアンケートで確認した。最適解の納得性、全体把握性、説明性に関して答えやすくするため、被験者には、事前に評価指標を説明し本システムを用いて最適解を探索する前に直感的な最適解を決定してもらった。その後、本システムで最適解を探索してもらいアンケートに答えてもらった。各評価指標に対する 5 段階評価の内訳を図 6.8 に示す。5 段階評価で最も低い評価を 1、最も高い評価を 5 として、平均値を計算した。評価の高い順に、1. 実行時間 (平均 4.6)、5. 説明性 (平均 3.9)、2. UI(平均 3.7) 、3. 納得性 (平均 3.6)、4. 全体把握性 (平均 3.5) となった。本手法は、比較的実用性と説明性の点は評価が高いものの、一方で、選びやすさという点で UI 部分、被験者の選択を繰り返すことによる最適解に近づいているという納得感の部分、さらに、探索空間の全体把握性の部分について優先的な改善が必要である。

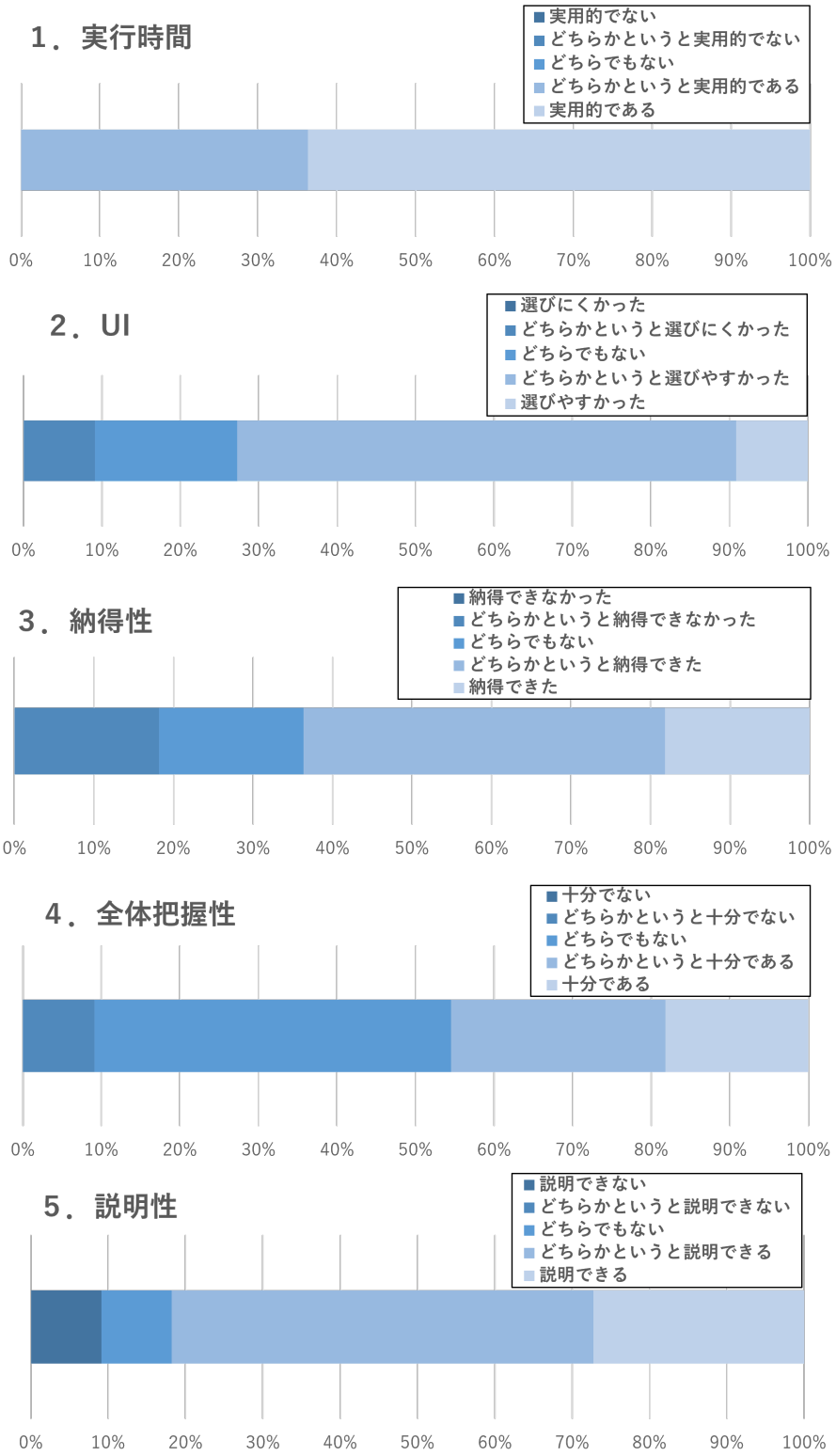

図 6.8: 主観評価の集計結果

Fig. 6.8: Aggregate results of the subjective evaluation

6.4.7　考察

前節での評価結果を考察を述べる。

実行時間

本指標の評価平均値は 4.6 であり、平均的には「実用的である」という結果を得た。11 人の被験者に対し表 3 に示す仕様のマシンを使って最適解探索を行った。探索における 1 回の SB の実行に要する時間の平均値は 5.0 分、SB の実行回数の平均値は 4.6 回であった。各被験者が最適解に達するまでの時間の平均値は 80.0 分であり、そのうち、6.3 で示したユーザによる解の選択に要した思考時間の平均値は 57 分となる（ただし、本実験とは関係ない作業の時間も含まれる）。6.4.5 節に示した意思決定者に与えられる 1 週間の許容時間と比較し、実用的と判断できる。

表 6.3: 使用機材の仕様

Tab. 6.3: Specifications of equipment used (1 node)

CPU	Intel(R) Xeon(R) Gold 6234 @ 3.30GHz × 16
Memory	64GB × 8
Storage	1TB SSD

最適解の選択しやすさ（ＵＩ評価）

本指標の評価平均値は 3.7 であり、平均的には「どちらかというと選びやすい」という結果を得た。改善点を被験者に口頭で尋ねたところ、パレートフロント上の特定 2 つの比較表示をする画面があるとよいといった、最適解の選択を支援する情報提供を要望する意見とリスク発生個所の表示する機能等シミュレーション詳細内容を表示する機能を希望する意見が聞かれた。

最適解の納得性、全体把握性、説明性

納得性の指標の評価平均値は 3.6 であり、平均的には「どちらかというと納得できた」という結果、全体把握性の指標の評価平均値は 3.5 であり、平均的には「どちらかというと十分である」という結果、説明性の指標の評価平均値は 3.9 であり、平均的には「どちらかというと説明できる」という結果を得た。比較的簡素な情報提示にも関わらず高い値が得られたと感じている。これは、意思決定者自身が持つ過去の経験を引き出し可視化できたことにより好意的な印象も加わった結果ではないかと考えている。

評価のまとめ

前小節までに見てきたように、本章で提案したインタラクティブな評価手法は概ね好評であると言える。しかし、納得性、全体把握性、説明性に関して、最も悪い評点を付けた被験者がいた。彼らにヒヤリングしたところ、自らが評価したい指標が表れていないことが原因であった。具体的には、利便性の指標として運用者視点での利便性の指標を用いているが、利用者視点の利便性の指標が表現されていないことが原因であった。

6.4.8 追加評価 1

前節の評価を受け、利用者視点での利便性の指標を追加した 4 指標用いて評価実験を行った。

利用者視点での利便性の評価指標

利用者視点での利便性の指標 V_{usr} は、正着率を表すもので営業時間中に到着予定時刻から一定時間以内に到着した配送数の割合として次の式で表すものを採用する。なお、到着予定時刻は、配送距離を平均的な停止時間を考慮した表定速度で除して求める。

$$V_{usr} = \sum_{i=1}^{n} \sum_{j=1}^{m} \frac{V_{jit}(i,j,2)}{(V_{adr}(i,j)}$$

$V_{jit}(i,j,k)$ は、到着予定時刻から k 分以内に到着した数を示す関数である。本章では、ダイヤ通りの運行する鉄道に類似しているため、いくつかの鉄道会社で延着と判断する際に用いられている 2 分を採用した (k=2)。

4 指標のパレートフロントの提示

6.3 で説明した 3 指標のパレートフロントの提示を 4 指標に拡張した。具体的には、安全性、経済性、運用者視点での利便性、利用者視点での利便性の合計 4 指標を表示する。本追加検証では、利用者視点での利便性の指標をプロットする円の大きさで表現した（図 6.9）。

追加評価 1 の質問

4 指標表示のパレートフロントを用いて 6.4.4 節と同様の評価を行った。被験者に表 4 示す評価指標について 5 段階評価のアンケートで確認した。

表 6.4: 4 追加評価アンケートの質問

Tab. 6.4: Additional Questionnaire for the Evaluation

No	項目	質問
追 1	利用者視点の利便性指標の有効性	軸を 4 つに増やしたグラフについて、解を選択するうえで、利用者の利用しやすさ（正着率）も考慮できるようになりましたか。
追 2	4 軸の UI	軸を 4 つに増やしたグラフについて、以前と比べて、軸が増えた割に解は選びやすくなりましたか。
追 3	最大軸数	本追加評価に参加し、最大いくつの評価軸まで適正な考慮のうえ、解を選択できると考えますか。

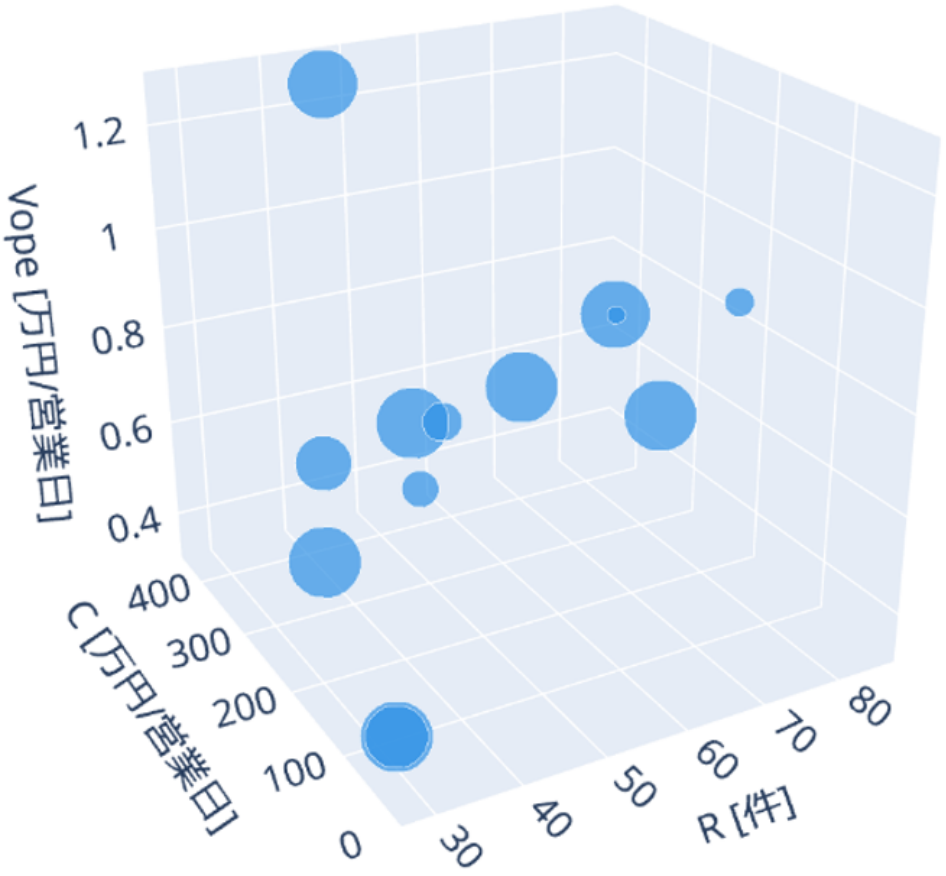

図 6.9: 4 指標のパレートフロント

Fig. 6.9: The pareto front with 4 indices

追加評価 1 の結果

9 人の被験者に 4 指標表示のパレートフロントを用いて最適解の探索を実施してもらった。前節で説明した評価指標について 5 段階評価のアンケートで確認した。各追加質問対する 5 段階評価の内訳を図 6.10 に示す。同様に、5 段階評価で最も低い評価を 1、最も高い評価を 5 として、平均値を計算した。その結果、1. 利用者視点の利便性指標の有効性 (平均 4.3)、2. 4 軸の UI(平均 3.8)、3. 最大軸数 (平均 3.4) となった。正着率の導入によって、利用者視点の利便性指標がある程度有効であると確認できた。また、4 軸目を追加した UI についても、平均的すると「どちらかというと選びやすい」という結果であり、概ね好評であった。ただ、最大軸数については、3 と 4 が拮抗し 1 人分 3 が多く、今回の追加実験で高々4 つの軸までの同時評価が限界であり、これまで通り 3 つの軸であれば今回の被験者全員が適正に評価できるという結論になった。特に、点の大きさで示した 4 つ目の指標について、これまでの XYZ 軸と見え方が異なるため、4 つの軸を公平に捉えることが難しいという意見があった。

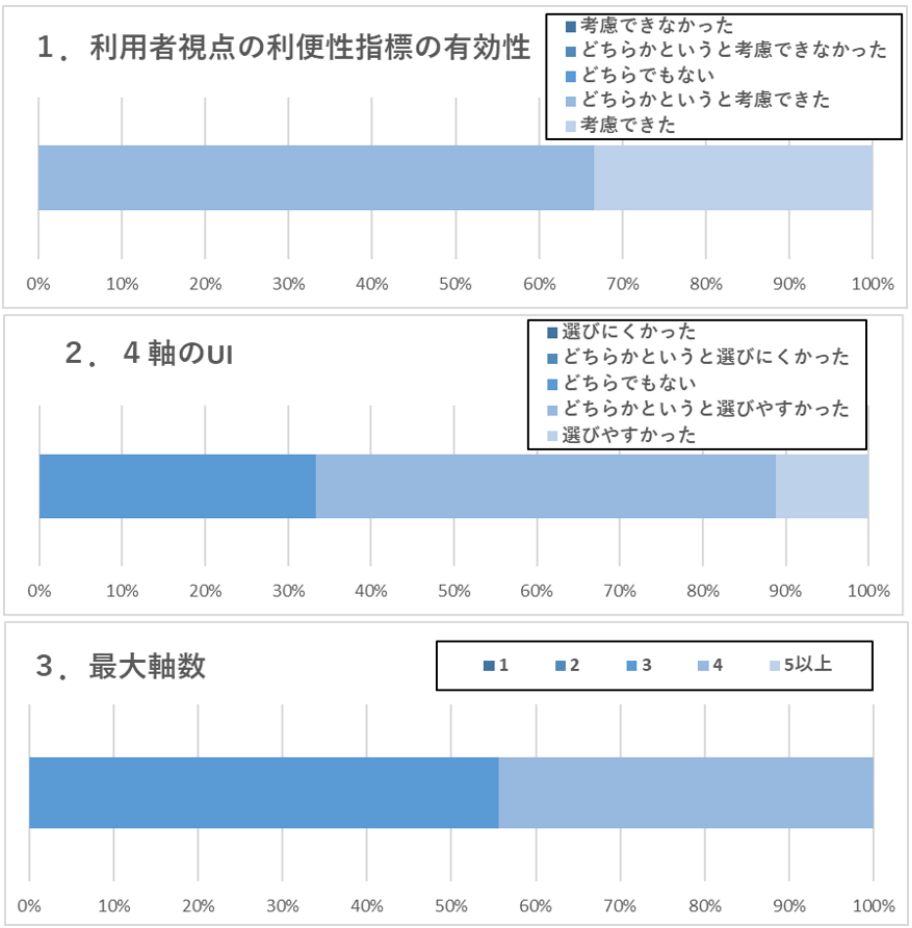

図 6.10: 追加の主観評価の集計結果

Fig. 6.10: Additional Aggregate results of the subjective evaluation

追加評価 1 のまとめ

多目的最適化の評価で用いられる 2 軸ずつを複数の 2 次元グラフで表示する散布図は、予備実験の結果「選びづらい」との評価となった (次節参照)。このため本章の追加評価 1 では、4 指標のパレートフロントの提示を用いた評価を行った。追加評価 1 の結果、利用者視点での利便性を追加した 4 軸での評価により、利用者視点での利便性も意識した評価でも本手法で最適解の選択手法の評価は、概ね好評となった。評価する軸の数については、3 及び 4 が適切だと分かったため、今後さらなる評価軸の追加を進める場合は、多段階での評価など一度に 5 つ以上の評価にならないような工夫を進める必要がある。

6.4.9 追加評価 2

追加評価 1 によって、安全性、経済性、利便性の他にもう一つの評価指標を追加した 4 軸での評価も現実的に実施可能であることが分かった。そこで、住民のサービス受容性に影響が大きい「道路での滞留時間」と「店頭待ち時間」を加味した評価を行った。これらの指標は時間的空間的制約であるため、6.1.1 節で説明した信号時相論理を用いて表記する。

指標の正確な記述

6.4.3 で述べた、安全性、経済性、利便性に加えて時間的空間的制約の評価指標を定量的な式で定義し、複数の意思決定者間で共通の理解を促す。この工程で信号時相論理を使用して、評価指標を記述し正確な理解に繋げる。

6.4.10 追加評価 2 の実験

提案システムを用いて、藤沢地域における ADR による配送サービスの設計・評価を実施した。数人の被験者に提案手法をアンケートにより評価してもらった。

評価指標

配送需要の異なる複数の店舗が存在することを想定し、複数の ADR が配送需要に応じて配送を行う状況を模倣し、安全性、経済性、利便性の指標と新たな制約条件の充足で評価する。評価基準を複数の意思決定者間で正確に理解できるようにするために、利便性の指標の一部であり新たな制約である、時間的空間的変動を含む評価指標である「道路での滞留時間」と「店頭待ち時間」の記述には、6.1.1 節で説明した信号時相論理の記述を使う。

安全性指標（リスク指標） 本指標は、6.4.3 節で用いたものと同じである。

経済性の指標 本指標は、6.4.3 節で用いたものと同じである。

利便性の指標 本指標は、6.4.3 節で用いたものと同じである。

店舗での滞留時間 店舗の手前の道路での滞留時間 T_s は、次の STL 式で表現する。

$$T_s = \sum_i Area_{[0,\infty]}(\sum_j Rcw_{i,j} > 1)$$

$Rcw_{i,j}$ は、店舗 i で ADRj が荷物をハンドリング（積み下ろし）している ADR の台数である。ここで重要なことは「Area」様相による STL の拡張を使用していることである。これは、標準の「グローバル」様相と「将来」様相とよく似ているが (2.5.3 参照)、堅牢性の評価関数である Area は、水平方向の「時間」的なマージンの余裕（つまり停止時間）に敏感であり、停車の許容量を超え違法駐車となりうることを意味する、垂直方向の「値」マージン ($\sum_j Rcw_{i,j}$ が 1 を超えるロボットの数) を積算する値を記録する。（つまり、駐車可能なスーペースは 1 台分とする。）すなわち、面積 (Area) の様相により、より粒度の細かい目的関数の表現が可能となり、最適化の判断に有益となる。詳細については、文献 [58] を参照いただきたい。

図 6.11 は ADRa と ADRb が店舗 i に到着することで、店舗の手前の道路での滞留時間 T_s が積算される様子を図示したものである。

路上待ち時間 路上待ち時間 Wt は、次の STL 式で表される。

$$Wt = \sum_i Area_{[0,\infty]}(G_{[-T,0]}\phi_i)$$

ϕ_i は、ADRi が配達中に道路上に停止しているかどうかを示す論理式である。T は 60 秒である。様相表現である $G_{[-T,0]}\phi_i$ は、60 秒を超えて路上に停止していた状態を指す。その時間は、Area 様相により、全てのロボットに関して積算される。このように Area 様相を使うと異常時に発生する様々な量を表現することができる。

図 6.12 は ADRa と ADRb が路上で滞留する度に、道路での滞留時間 W_t が積算される様子を図示したものである。

シミュレーションパラメータ

シミュレーションパラメータは表 6.1 と同じものを用いる。

比較方法

6.3 節で述べたインタラクティブな対話型最適解選択手法（以下、方法 X と略記）と Optuna [9] を用いたパレート最適解を自動選択手法（以下、方法 Y と略記）を比較する。なお、Optuna ではベイジアン最適化の 1 つとして確率密度関数を推測しながらパレート解を求める探索手法である MOTPE (Multi Objective Tree-structed Parzen Estimato) アルゴリズム [55] を利用している。

複数の被験者によるアンケートで両方式を比較する。モビリティ サービスを導入する意思決定者は、限られた時間内で複数の意思決定者が満足するソリューションを選択する必要がある。本実験では、方法 X と方法 Y の特徴を明らかにするために、表 6.5 に示す、戦略反映、説明可能性、網羅性に関する質問を用意した。

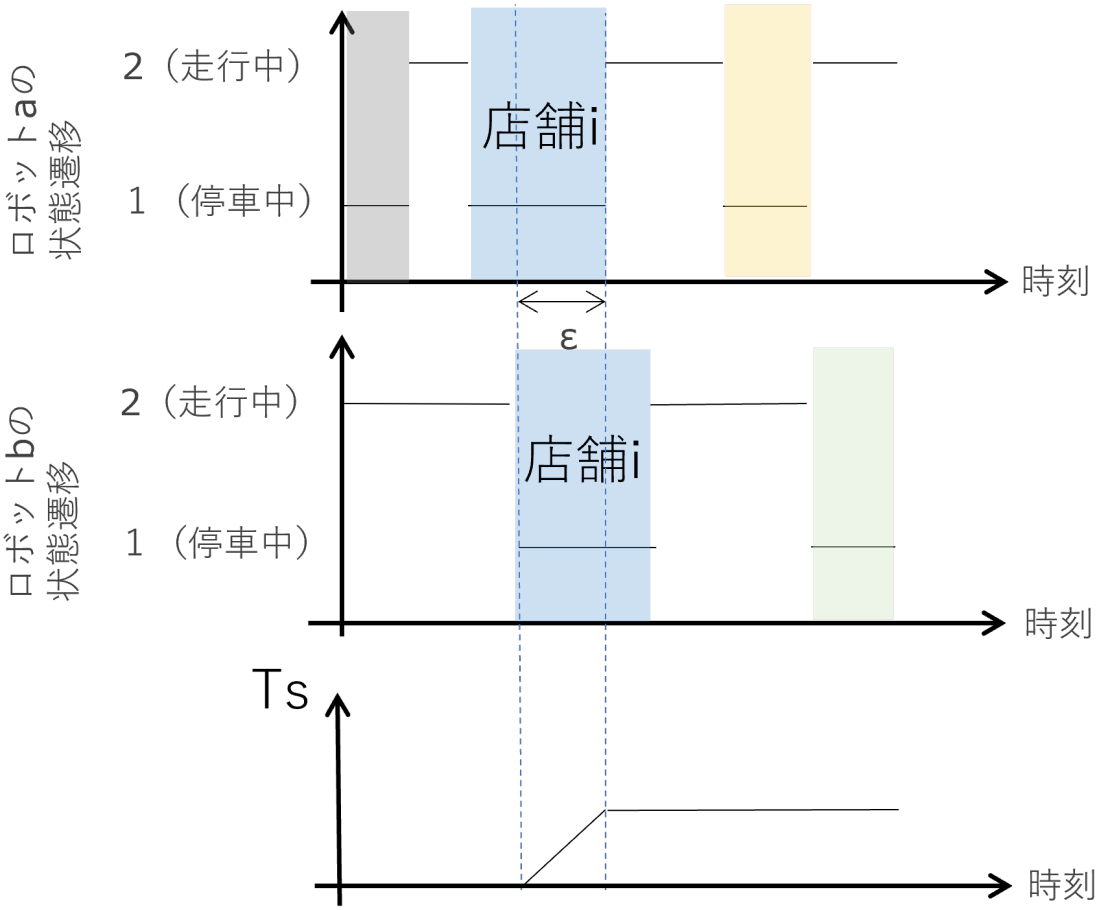

図 6.11: 店頭待ち時間

Fig. 6.11: Total time of parking in front of stores

表 6.5: 評価アンケートの質問

Tab. 6.5: Questionnaire for the Evaluation

項目	質問
戦略反映	自分の戦略をより反映した結果が得られるのは、どちらの方法だと思いますか
説明可能性	最適解の選択理由を説明しやすい方法はどちらですか
網羅性	どちらの方法が解決策をより網羅的に探せたと感じましたか

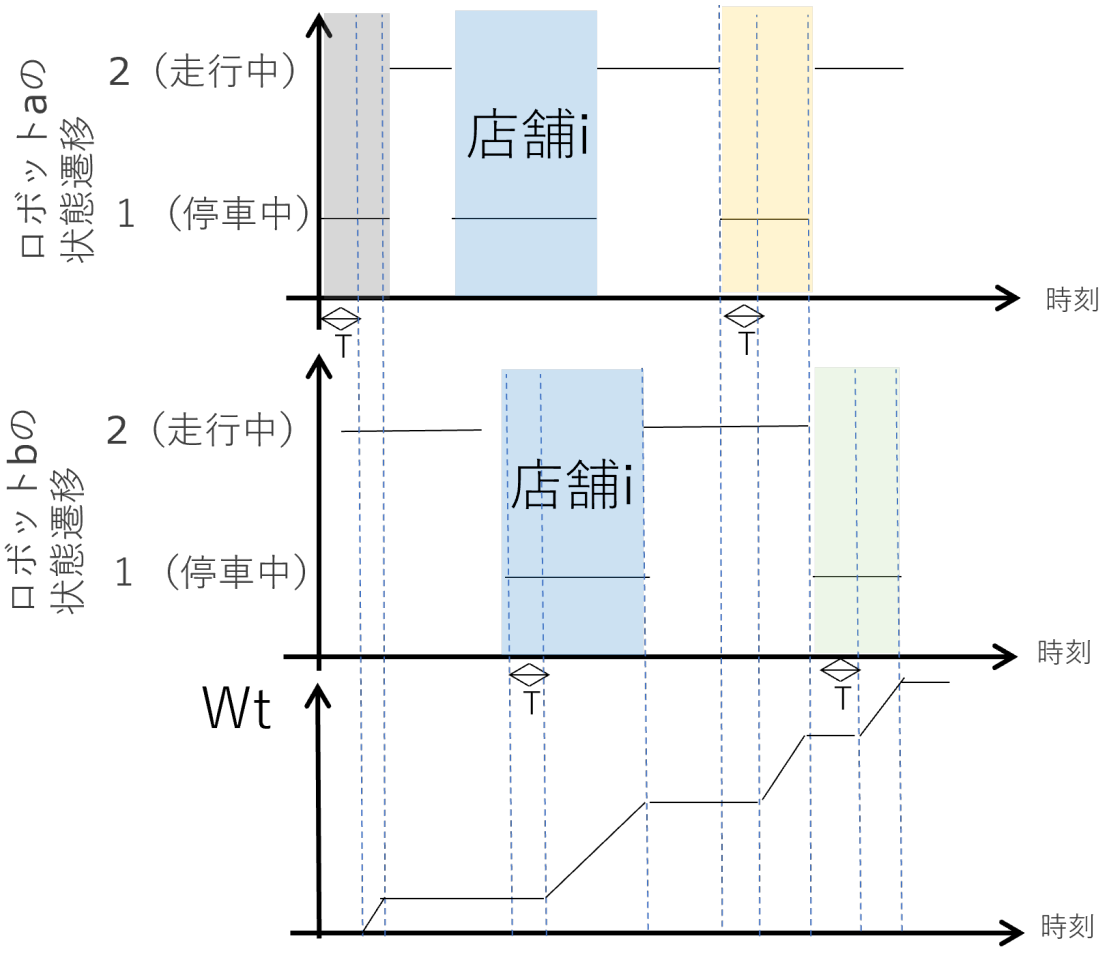

図 6.12: 路上での滞留時間

Fig. 6.12: Total time of parking on the road

6.4.11　追加評価 2 の実験結果

被験者 10 名に 方法 X と 方法 Y による最適解探索実験を行ってもらい、前小節で述べた 5 段階の主観アンケート調査を行った。各被験者には評価指標について事前に説明し、評価指標についての共通理解が得られた状態で実験を開始した。信号時相論理の記述を使用したため、評価基準の理解に関して曖昧性を廃した状態で実験を開始できた。被験者には最適解選択の効果を実感しやすいように、実験開始前に直感的に最適解だと思う解を選択してもらった。

図 6.13 に被験者実験のアンケート結果を示す。方式 X が良い場合 1, 方式 Y が良い場合 5 として、各質問に数値で答えてもらった。戦略反映項目については、平均値が 2.2 であり、方法 X の方が意思決定者の戦略をより反映していることがわかる。説明可能性の平均スコアは 2.9 で拮抗している。評価値としは 3 未満であり方式 X が僅かに優れており、投じた人数は同数であった。一方、網羅性に関しては、方法 Y が平均スコア 4.1 で優れていた。

意思決定者にとって、選択したシナリオに自らの戦略が反映されており、高度に説明可能で説得力があることがサービス導入の決断に非常に重要である。説明可能性は、選好性と網羅性という 2 つのカテゴリーに分類できると考えられてる。 提案手法は意思決定者の戦略を反映したが、網羅性の点で改善の余地があると言える。

最後に、決定を下す前に、方法 X と方法 Y の間で評価されたシナリオの数と 10 人の被験者の平均とを比較する。 その結果、方法 X では平均 76.8 シナリオ、方法 Y では平均 2,048 シナリオが必要であったことが分かった。したがって、方法 X は、以前の方法よりも計算と決定を行う効率が約 25 倍高いことになる。逆に言えば、方法 X においてシミュレーションバッチサイズを大きくすれば網羅性に関しても改善できる余地があると推測できる。

6.4.12　追加評価 2 のまとめ

ロボット配送サービスを設計するには、膨大な選択肢がある。膨大な組み合わせの中から意思決定者に受け入れられるためには、意思決定者の戦略を反映しながらサービス仕様を決定することが重要である。また、自由軌道を走行する自動運転技術を含むモビリティサービスの実現には、費用対効果だけでなく安全性も評価する必要がある。

3 章では、安全性、経済性、利便性の評価指標を同時に評価する方法を提案している。また、6 章では、対話型多目的最適化手法を提案し、さらには第四の評価指標も同時に評価する手法を提案した。追加評価 2 で、形式手法を用いて、意思決定者の間で時間的空間的な変化を含む評価指標を正確に把握しながら評価する方法を示した。本手法を特定の町におけるロボット配達サービスの設計に適用し、被験者実験において「戦略反映」の観点から本手法の妥当性を確認した。モビリティサービスには、移動中の停留時間や店舗での待ち時間など、時間的空間的に変化する指標が数多く存在する。シミュレータを用いた評価に STL 式を導入することにより、提案方式で設計・評価できる項目が増え、より実用的になったことが確認できた。

6.4.13　複数の意思決定者による最適解の分析

本章で行った実験において、複数の評価者（意思決定者）が最終的に選択した最適解を表 6.6 に示す。この最適解の明細は、評価 6.4.8 に参加した 9 名のものである。

それぞれの評価者が選んだ最適解の安全性、経済性、利便性の指標値を可視化したものを図 6.14 に示す。評価者 A は、低コスト、低利益のスモールスタートの解を最適解として選択し、評価者

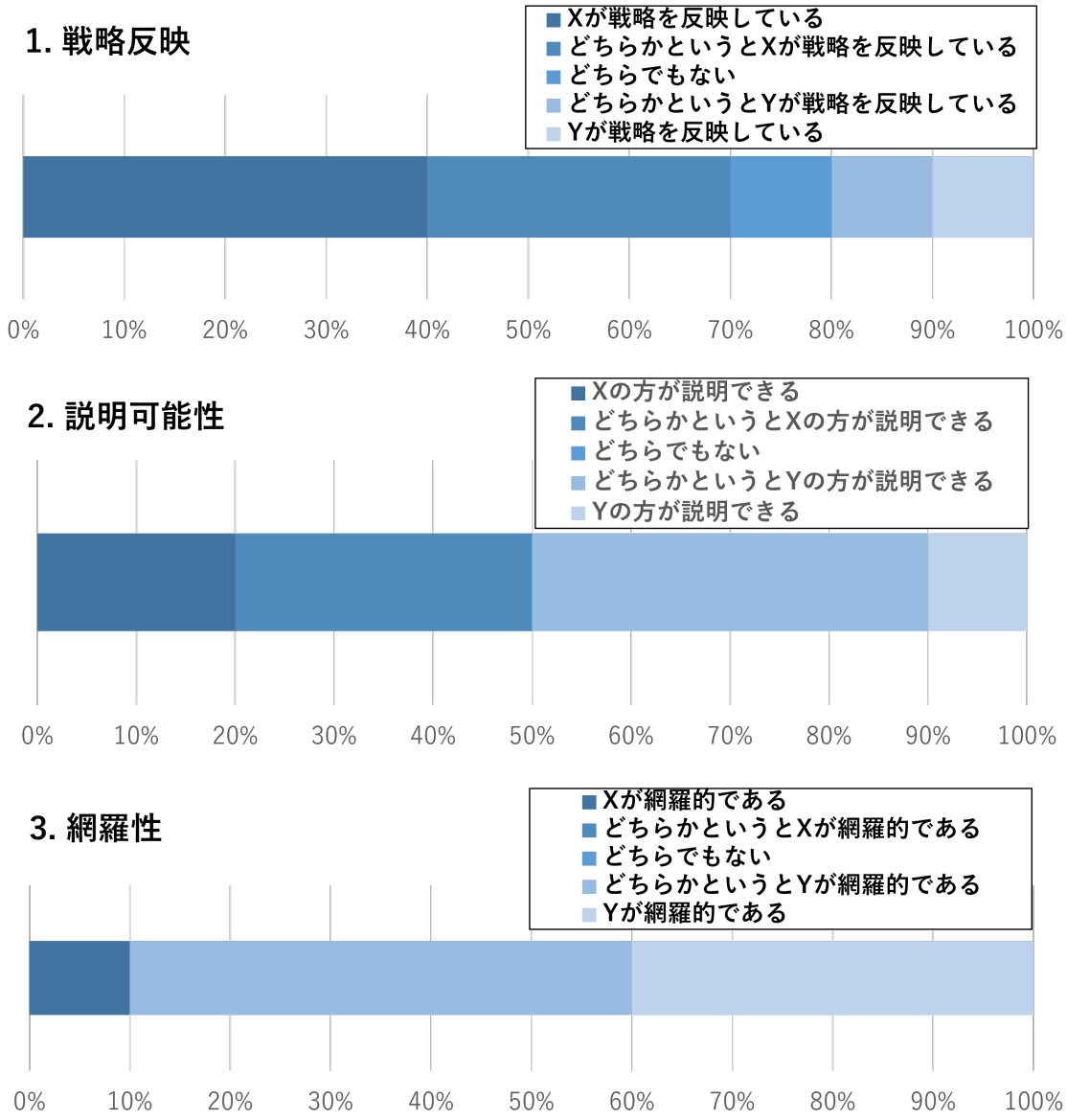

図 6.13: 被験者実験の結果

Fig. 6.13: The result of subjective evaluation

表 6.6: 複数の意思決定者の最適解

Tab. 6.6: The Best Scenario Choosen by each evaluator

評価者	A	B	C	D	E	F	G	H	I
配送元数	3	4	4	4	4	5	3	3	4
WAU	10	7	7	7	6	4	5	8	6
営業時間	8	7	9	8	9	5	7	8	8
ロボット台数	1	3	4	4	4	4	5	3	4
ロボット時速	6	6	6	6	6	6	6	6	5
ロボット荷室数	3	4	4	4	3	4	3	3	3
オペレータ数	2	1	1	2	1	2	3	1	2
安全性	13	31	30	30	31	31	27	27	33
経済性	28.2	222	227	228	227	415	223	212	228
利便性	0.42	0.82	0.82	0.82	0.87	1.26	0.75	0.66	0.87

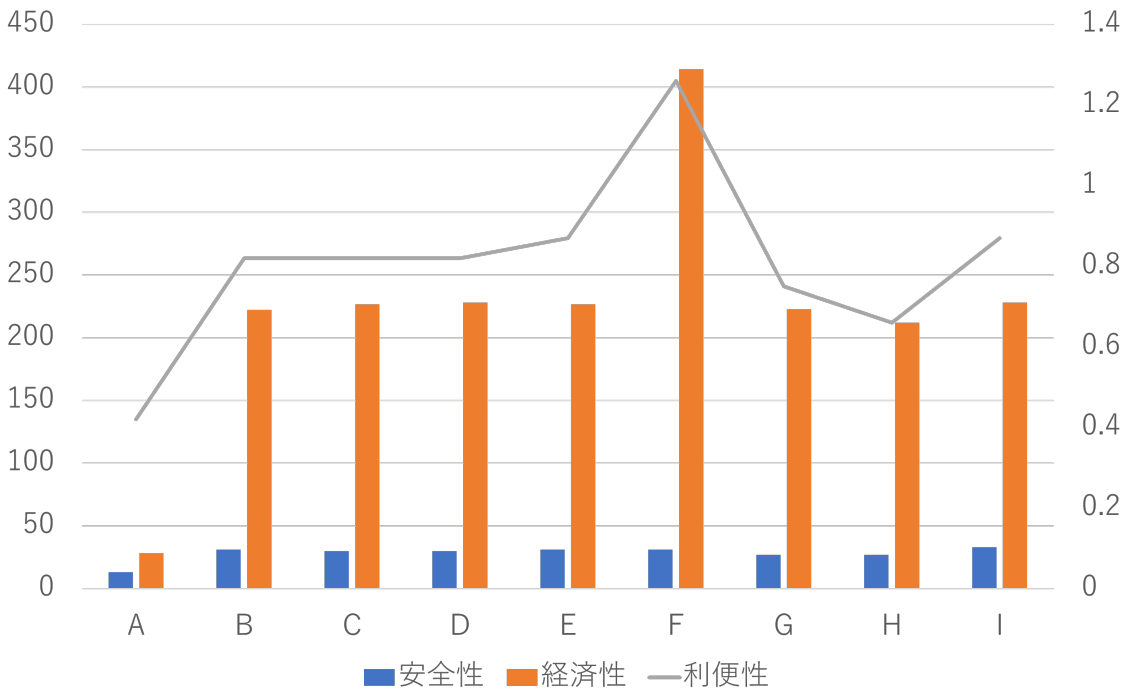

図 6.14: 評価者別最適解の指標値

Fig. 6.14: The indeces of he best solutions

F は、高コスト、高利益の解を選択しており、その他の評価者は類似している解を最適解に選んでいることが分かる。評価者 A が選んだ解の経済性の指標が他の評価者が選んだ解と比べて良化する大きな要因は配送元数にある。これは、配送元によって店舗開拓に要する費用が異なり、店舗の初期費用 $C_{env}(k)$ に影響を与えるためである。評価者 F が選んだ解の経済性の指標が他の評価者が選んだ解と比べて悪化する原因も同様である。

評価者が選んだ最適解は、直感的に考える解とは違う解を選んでいる。つまり、直感的には、少ない台数の RDA を 24 時間稼働させるのが最も効率が良いのではないかと考えてしまうのではないだろうか。ところが、多くの評価者が選んだ最適解は、3〜4 台の RDA を 8 時間前後稼働させる解が最適と判断している。これは、各配送元の店舗で配送需要を発生させる一因は、WAU により上限が決まることがある。1 日あたりの配送需要に上限があるため、24 時間稼働させても最適とはならない。需要を多く発生させるため配送元となる店舗を多く開拓すると配送元が互いに離れているため複数の荷室を有する RDA を配置しても効率的な配送ができない。このような要因があるため安全性、経済性、利便性の評価指標を総合的に判断し、直感とは異なる解が選択されている。

一般に複数の意思決定者でサービスの導入を決定する場合、複数の評価者が参加する決定会議が行われ、多数決による議決が行われることが想定される。本実験で得られた最適解は、経営的視点から低リスクの解を選んだ評価者 A と高リスクの解を選んだ評価者 F の意見を決定会議の場で聞き、極端に良い意見がでなければ、多くの評価者が選択している解を最適解として採択すると考えられる。

本手法では、評価者の戦略が反映されている解を選択しているため、このような決定会議でも評価者が選択理由を説明しやすい。本実験では、それぞれの評価者が選択した最適解は、9 名中 7 名が類似したものとなった。そのため上に述べたような決定会議でも円滑に採択する解が選出されると考えられる。具体的には、4 つの配送元から荷室 3 つないし 4 つ有するロボットを 4 台投入し 8 時間程度営業するサービスを展開する解を中心に決定会議で議論が行われると想定できる。すなわち、本実験の結果、それぞれの評価者が性質の異なる局所最適解をバラバラに選択したわけではなく、多くの評価者が比較的類似した解を選択しているため、決定会議を開催しても結論が得られなくなるリスクは低いといえる。

6.5 本章のまとめ

自動配送ロボットによる配送サービスの設計には多様な選択肢がある。新たなモビリティによるモビリティサービスの導入には、費用便益の評価に加えて、人ごみを通過する量や交通量が多い道路を横断する頻度など安全性に関わる評価も加え、安全性、経済性、利便性の評価指標の同時評価が必要である。モビリティサービスの決定要因の組み合わせは膨大となるため、本章では、シミュレータをインタラクティブに用いる手法を提案評価した。運用者視点での利便性と利用者視点の利便性に拡張した提案手法を特定地域でのモビリティサービスの評価に適用した結果、十分実用的な時間で最適解を探索することができたため被験者の評価は概ね好評であった。

5 章では、安全性・経済性・利便性に関する評価指標の線形和を目的関数とし、それぞれの指標に対する重みを導入していた。一方で、本章では、意思決定者の選択を反映することで探索が進んでいくため、重みを仮定する必要のない汎用性の高い手法となっている。さらに、本手法では、運用者視点・利用者視点の 2 種類の利便性の指標を導入した 4 指標で最適解を選定できることも確認できた。すなわち、広範囲な評価指標で新たなモビリティサービスを評価できる手法となった。

形式手法を用いて、意思決定者の間で時間的空間的な変化を含む評価指標を正確に把握しながら評価する方法を提案し、提案手法を特定の町におけるロボット配達サービスの設計に適用し、被験者実験において「戦略反映」の観点から本手法の妥当性を確認した。

モビリティサービスには、移動中の停留時間や店舗での待ち時間など、時間的空間的に変化する指標が数多く存在する。シミュレータを用いた評価に STL 式を導入することにより、提案方式で設計・評価できる項目が増え、より実用的になったことが確認できた。

複数の評価者が選択した最適解を分析した結果、類似した解を選ぶ傾向も見られた。すなわち、本手法は、複数の意思決定者による意思決定が行われる場合でも、バラバラな解を選ぶことがなく、また各評価者が自らの最適解の選択理由を説明できるため、円滑に決定会議としての解を採択できると考えられる。

今後は、より現実的な数値やモデルを用いてリアリティを増した評価や他地域のモビリティサービスの評価への適用を進めたい。

（参考）散布図による表現

多目的最適化のパレートフロント提示には、図 6.15、図 6.16 で示すような 2 軸ずつ複数の 2 次元グラフで表示する散布図が用いられる。追加実験では、図 6.15、図 6.16 についても提示したが、予備実験での被験者 3 人いずれも判断材料として利用しなかったという結果になった。

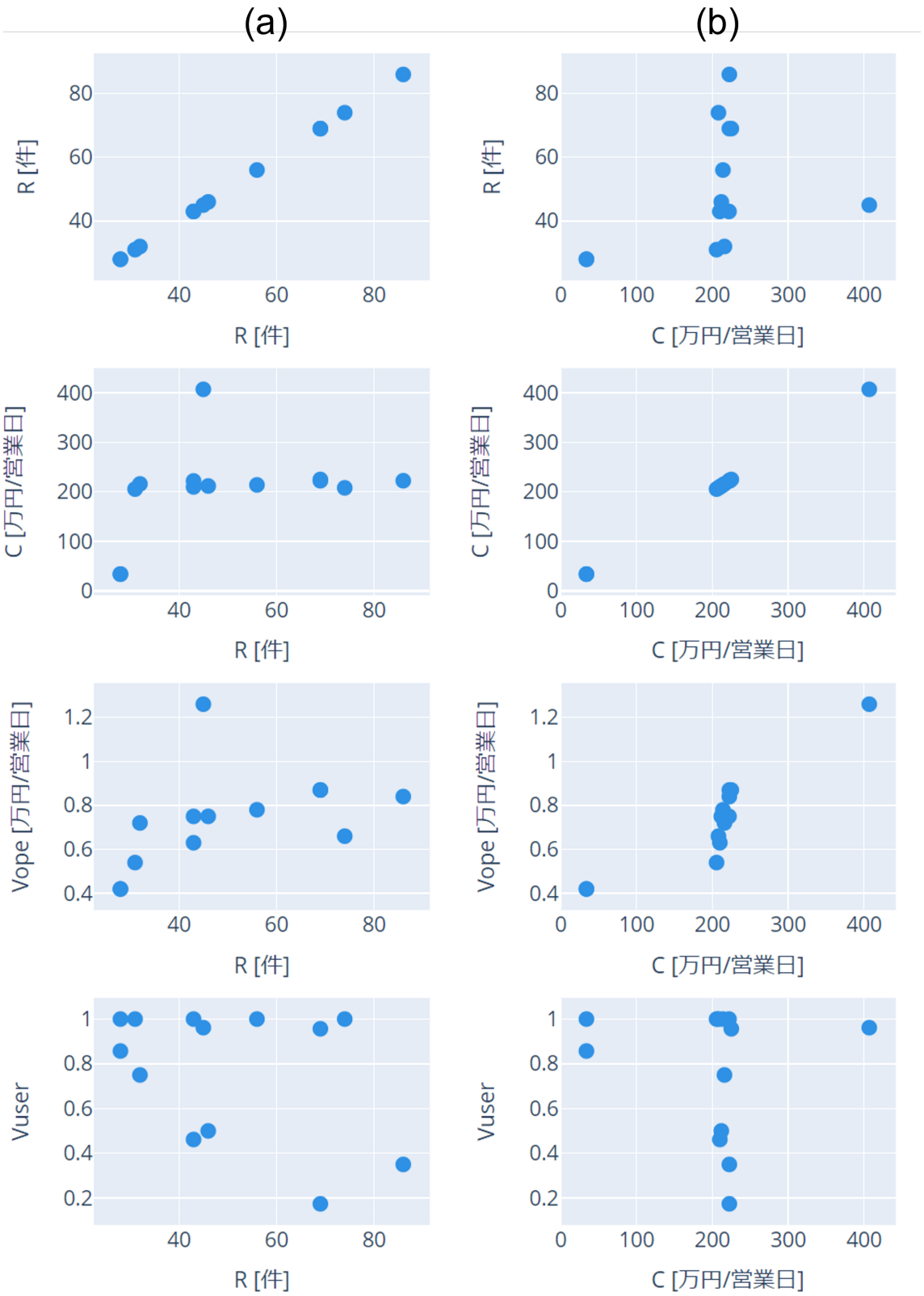

図 6.15: 散布図によるパレートフロントの表現 (a) 安全性 R とその他の指標 (b) 経済性 C とその他の指標

Fig. 6.15: Representation of the Pareto front with a scatterplot (a)safety vs. the others (b)economic vs. the others

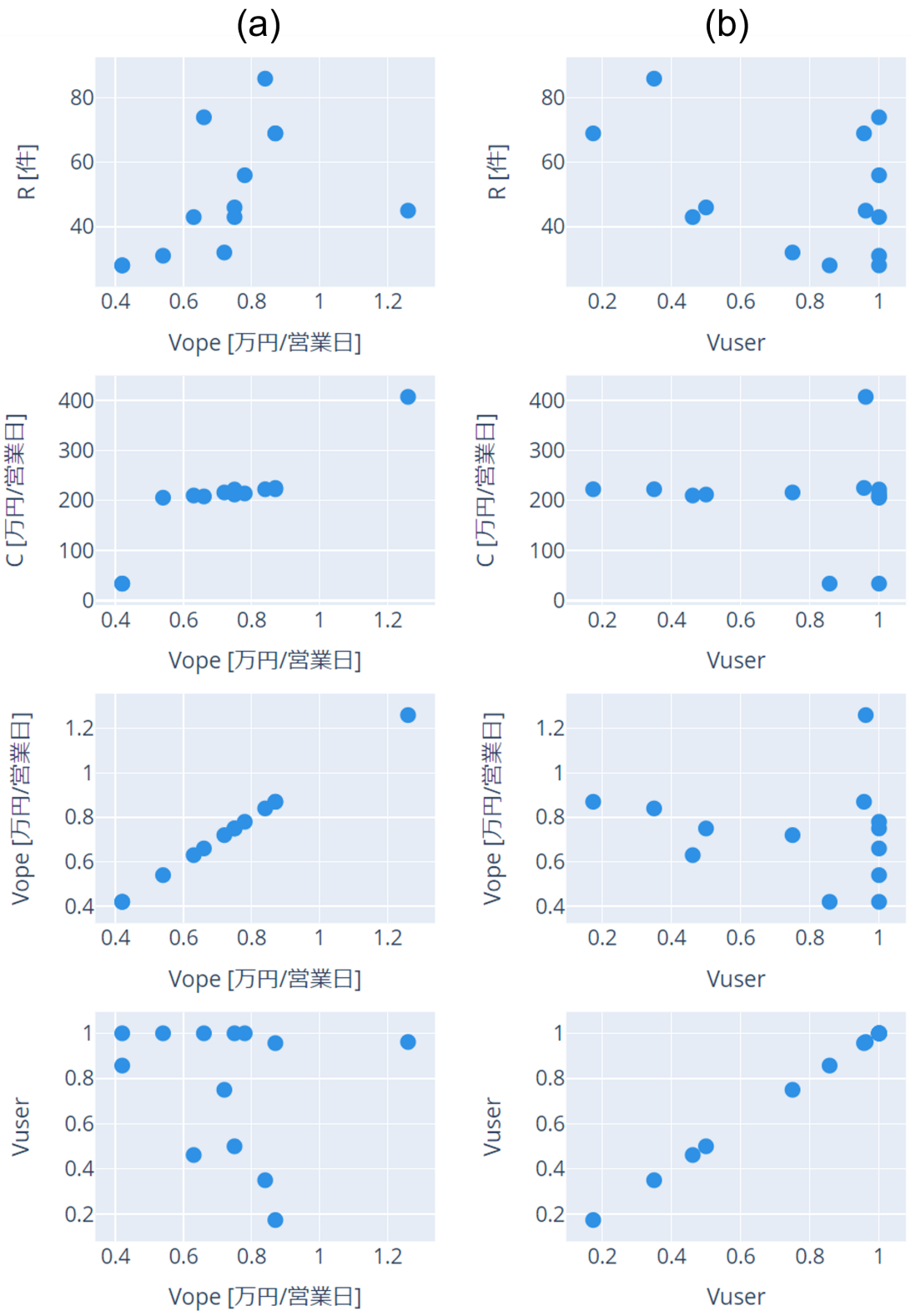

図 6.16: 散布図によるパレートフロントの表現 (a) 運用者視点の利便性 Vope とその他の指標 (b) 利用者視点 Vusr の利便性とその他の指標

Fig. 6.16: Representation of the Pareto front with a scatterplot (a)efficiency for user vs. the others (b)efficiency for operator vs. the others

第7章　結論/総括

交通システムが単なる移動手段から、社会的、経済的、環境的な影響が大きい都市システムの一部となり人々の生活やビジネス、社会的関係、環境などに大きな影響を与えるため、より総合的なアプローチが求められている。

これまで古くから、大規模なインフラ投資を伴う交通計画では、交通インフラ整備がマクロ経済への効果や各地域の人口や経済力への影響を推計するため様々なモデルが提案されてきた。過去、様々な指標が乱立する状況で交通インフラ整備が行われてきた反省から、国土交通省では、鉄道や道路計画の費用便益分析のガイドラインを定めてきた。つまり、大規模な投資の意思決定を行うために関連するステークフォルダの間で統一的な視点で評価する方式を定めてきており、これが長く用いられてきた。

7.1　結論

本研究では、費用便益分析による交通計画の限界を計算機科学、計算機能力の進展で超えることを目標に、新たなモビリティサービスの設計・評価の手法を提案してきた。提案手法は、新たな要求が発生し新たなモビリティが新たな環境で運用されることを念頭に、複数の意思決定者が多数サービスの選択肢から最適解を導出できるモビリティサービス設計支援・評価手法であり、形式仕様の活用で評価指標を正確に把握しつつ、事象駆動シミュレータを対話的に用いる手法である。図 3.1 に示したフローに従い、モビリティサービスの評価指標の定義に形式記述を積極的に活用する。既存の費用便益分析で用いられてきた経済性、利便性の指標のみならず、人々が生活する道路で自由軌道を走行する自律移動車両がより安全な走行経路を設計・評価するための安全性の指標を加えた 3 指標の同時評価を可能とする。モビリティサービスを提供する地域（エリア）での需要や当該エリアでの交通参加者の交通量を調査し、当該エリアでの運行パターンの候補を抽出する。これらをシミュレータを用いて再現し、評価指標の値の変化で最適な解を抽出する。

計算機科学の研究の成果である形式仕様記述とモデル検査の力を用いてライドシェアサービスの活性特性 (liveness property) の検査を試みた。モビリティサービスは、例えば出産や事故時の配車、薬剤の配送、緊急車両の手配等、その活性の担保が望まれる並列システムの一つとして位置づけられる。予約したのに配車されない事態が発生する可能性があるシステムは社会に受け入れがたい。線形時相論理を用いてクリプキ構造となるようにライドシェアシステムの形式仕様を記述した。汎用的なデータ構造を記述可能な仕様記述言語である Maude を用いて、地図や人や車の配置をパラメータとして記述する例を示した。並列システムでは、公平性の仮定を置かないと活性の特性が満たされなくなる反例が発生することがある。本研究で記述したライドシェアシステムのモデル検査においても強い公平性の仮定を置く必要がある事例を指摘した。公平性の仮定をおいたモデル検査では計算量が爆発的に増加するため、現在の計算機でも計算不能となるケースがあることが知られている。これはクリプキ構造をオートマトンに変換する際に現在の計算機が保有しているメモリでは収容できなくなることが一因である。この問題対して、緒方教授が提案された分割統治アプローチを適用することで、約 7.7 時間必要となるモデル検査を高速化 し約

6 秒で検査できること、現在の計算機ではメモリ不足となり計算不能なモデル検査を実現可能な計算時間（約 3 分 43 秒）で実施可能であることを示した。

シミュレータを用いて計算機能力を活用して、新たなモビリティサービスの設計・評価を行う方法の実践として、人を運ぶ事例とモノを運ぶ事例に提案手法を適用した。人を運ぶ事例としては、大阪地区での自動運転ライドシェアシステムのサービスを題材に評価を行った。大阪地区では、社員がビルの出入りの際に社員カードで時刻を刻印するデータがあるため、1 日 30 万トリップ程度の人の動きをほぼ正確に再現できる。既存の費用便益分析で用いられてきた経済性、利便性の指標にみならず、自動運転車両と人とがある一定の距離内に近接する事象の頻度と車両の重量及び速度から事故発生時の障害度を算出する安全性の指標を定義して、これら 3 指標を加味した最適解を HPC である StarBED を活用した全探索と焼きなまし法を用いた最適解探索で探索した。この結果、最大積載人数の 4 名/台の車両を運行することで最適解が得られるという直感とは反して、3 人乗りの車両を運行したほうが最適であるという結果を得ている。また、シミュレーションでは、人と車が近接する場所を特定できるため、当該事象の発生場所を地図上に可視化したところ、実際の自動運転車両と人との近接事象が発生した場所と合致することが確認できている。すなわち、危険事象が発生する場所を回避する経路を再設計したり、当該場所を通過時の最高速度を制限した新たな運航計画を作成することでより最適な解が得られることが分かった。計算機の力を使って網羅的に検査することで得られた解であると言える。

シミュレータを用いて計算機能力を活用して、新たなモビリティサービスの設計・評価を行う方法のもう一つの実践として、藤沢 SST での自動配送ロボットによる配送サービスを題材に評価を行った。藤沢 SST では、約 1900 名の住民が様々な消費財等を発注して生活を行っている。複数の店舗が配送元となることを想定して、配送サービスに投入するロボットの台数や営業時間といった因子をシミュレーションパラメータとして最適解の探索を行った。パラメータの組み合わせの数は、10 億を超えるため、適切な単位でシミュレーションを複数回実施し、パレート解を安全性、経済性、利便性の評価軸の上に表示して、意思決定者が最適解を選択することで次のシミュレーションバッチを行う対話的な方法で最適解を探索する方法を提案した。アンケートによる主観評価により提案手法の実行時間、UI、納得性、全体把握性（網羅性）、説明性に関する評価を行い、実行時間 (平均 4.6)、 説明性 (平均 3.9)、 UI(平均 3.7) 、 納得性 (平均 3.6)、 全体把握性 (平均 3.5) の評価を得ている。比較的実用時間と説明性の点は評価が高いが、UI や被験者の選択を繰り返すことによる最適解に近づいているという納得性、探索空間の全体把握性に関しては、改善の余地があるという結果が得られた。また、運用者視点の利便性（配達数）のみならず利用者視点での利便性の評価指標（到着予定時刻から一定時間以内に到着した配送数の割合 : 正着数）を評価軸に加えたいとの意見を得たため、これを評価指標に加えた 4 評価指標を同時に評価する方法を提案した。具体的には、4 つ目の指標は、3 次元空間にプロットする点の大きさで可視化することで合計 4 つの指標を同時に参照しながらパレート解を比較できる方法を提案した。この手法による解探索について、追加した指標を参考に評価できたか、4 軸の評価方法での選びやすさ（UI)、最大考慮可能な指標の数について主観評価で確認した。この結果、1. 利用者視点の利便性指標の有効性 (平均 4.3)、2. 4 軸の UI(平均 3.8)、3. 最大軸数 (平均 3.4) となった。正着率の導入によって、利用者視点の利便性指標がある程度有効であると確認できた。また、4 軸目を追加した UI についても、平均的すると「どちらかというと選びやすい」という結果であり、概ね好評であった。ただ、最大軸数については、3 と 4 が拮抗し 1 人分 3 が多く、今回の追加実験で高々4 つの軸までの同時評価が限界であり、これまで通り 3 つの軸であれば今回の被験者全員が適正に評価できるという結論を得た。

さらに、計算機科学の研究の成果である形式仕様記述である信号時相論理（STL）を用いて評

価指標を記述し、複数の意思決定者の間で共通認識を得た上で、シミュレータを用いて計算機能力を活用して、新たなモビリティサービスの設計・評価を行う方法の実践として、藤沢 SST での自動配送ロボットによる配送サービスを題材にさらなる評価を行った。安全性、経済性、利便性に加え、新たな制約として、「道路での滞留時間」と「店頭待ち時間」を考慮した最適解の探索をシミュレータを用いた対話的な手法で探索した。例えば、路上での滞留時間 Wt は、時間的空間的な値の積算値を表現する Area 関数を用いて STL の様相記述で、$Wt = \sum_i Area_{[0,\infty]}(G_{[-T,0]})\phi_i$ と表現できる。(ここで ϕ_i は、ADRi が配達中に道路上に停止しているかどうかを示す論理式、T は 60 秒であり、様相表現である $G_{[-T,0]}\phi_i$ は、60 秒を超えて路上に停止していた時間を指す。) 被験者にはこれらの評価指標についての共通理解が得られた状態で、既存方式である Optuna を用いた最適解の探索と提案方式の探索について、戦略反映、説明可能性、網羅性に関する質問で評価してもらった。結果、提案方式の方が評価者の戦略を反映しているが、網羅性に関しては、既存方式の方が高い評価を得た。提案方式のシミュレーション実行回数と既存方式の実行回数を比較すると既存方式の方が 25 倍程度多くシミュレーションを回していることから、網羅性の向上には、シミュレーションバッチの回数（現在 16）を増した方が望ましいことが分かった。モビリティサービスには、移動中の停留時間や店舗での待ち時間など、時間的量的に変化する指標が数多く存在する。シミュレータを用いた評価に STL 式を導入することにより、提案方式で設計・評価できる項目が増え、より実用的な設計・評価方法となった。

7.2 提案手法の利点と論拠

5 章及び 6 章で見てきたように、本研究の提案手法では、費用便益分析では評価できなかった安全性の指標も同時に評価できるため新モビリティのサービス設計・評価に適用できる。図 7.1 は、図 3.1 に示したフローに本研究で主な示した利点と論拠を述べている章・節を示したものである。ここで本研究による提案手法の利点と論拠を表 7.1 まとめる。

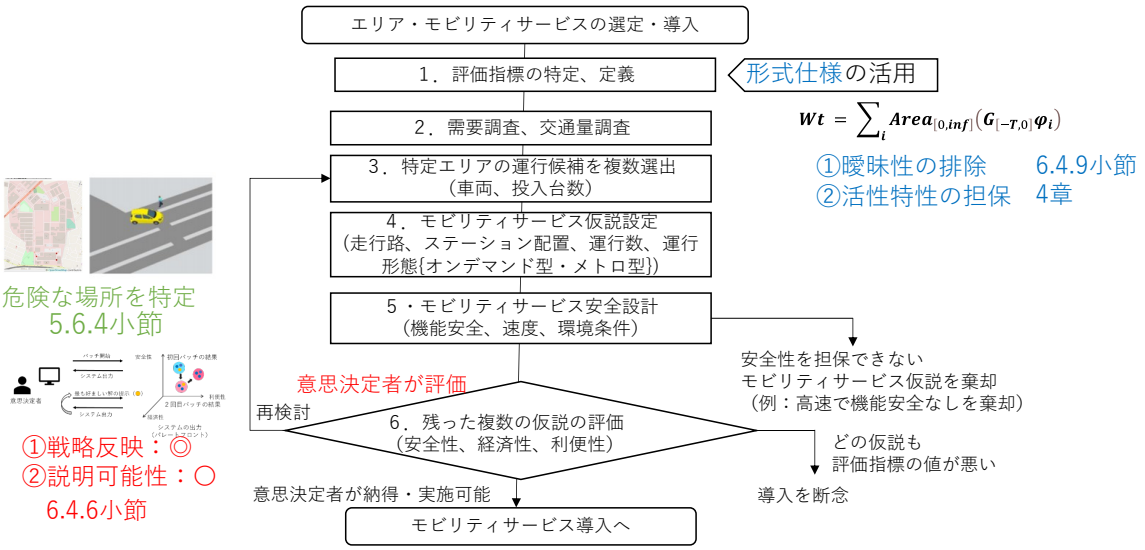

図 7.1: 主な利点及び論拠と提案フローの関係

Fig. 7.1: Relationship between main advantages and rationale and proposal flow

これらの利点から、仮説を立案し検証することが望まれているモビリティサービス、特に新たなモビリティの投入が必要となるモビリティサービス設計・評価に提案手法は好適である。

表 7.1: 提案手法の利点と論拠
Tab. 7.1: Advantages and rationale for the proposed method

利点	論拠
データドリブンな分析に基づきシミュレーションで再現	
人の移動を再現させることでヒヤリハット事象の発生場所が特定できるなど安全性の評価の精度が高まる	5.6.4 で、ヒヤリハット発生場所の抽出例を例示
静的な統計分析に基づく設計、数理計画法（OR）と比較して、モビリティサービスに投入するモビリティの変更時等、安全性の指標が変化する場合でも柔軟に評価し、最適解が探索できる	5.6 節で、乗車定員が異なる車両を含めた最適解探索を実施
モビリティサービスの仮説が変更になれば、モデルを再考して（高速に）再評価できる	3.2 節で、再評価を含めたフローを提案。6.4.8 及び 6.4.9 で、評価軸を追加した評価を例示
計算機を使った可能性の探索	
実証により確認するには高コスト、高リスクと考えられる解についても定量的に評価できる	5.6 節で、高リスクな速度や乗車定員が多い車両の運用を含めた評価を実施
分析の抜け漏れ防止、人知を超えた変化点・変化量の把握が可能となり効率的な解の発見につながる	**全探索** 5.6 節で、HPC を用いた全探索を実施 **評価者の興味の範囲の探索** 6.3 節で、対話的にシミュレータを用いた手法を提案。6.4 節で、対話的な探索を実施。
解の空間の全体像が把握可能となり、選択した最適解の説明性が向上する	5.6.2 で、現実的な時間で最適解を探索した結果を提示
対話的な解探索であるため、意思決定者の戦略を反映した最適解が選択可能となり、また説明性が向上する	6.4.6 で、納得性、全体把握性、説明性の評価結果を提示。6.4.11 で、意思決定者の戦略を反映した解を選びやすいという評価結果を提示。
形式仕様記述を用いる手法	
モデル検査で、活性特性（liveness）を含め、設計の意図通りに動くことを担保できる	4 章で、活性特性を含めたモデル検査を実施
意思決定者の間で時間的空間的な変化を含む評価指標を正確に把握しながら評価することができる	6.4.9 で、道路での滞留時間、店舗待ち時間の定義に形式記述を活用し評価者の理解を統一

7.3 今後の展望

モビリティサービスは、マクロ経済への効果や各地域の人口や経済力にも影響を与えるインフラとしての側面がある。本研究の例で引用した利便性の指標では、このようなインフラとしての効果は導入していない。しかし、本研究で再現した人流や物流は、経済行為の結果であり、その行為で交換される価値は経済活動を表したものとなる。本研究で取り上げた実践例では、日単位の活動に関してシミュレーションすることでモビリティの運行の最適解を探索してきた。他方、計算機の能力が進展し、また本研究で例示したような HPC を用いた並列化がもっと容易に手軽に行えるようになれば、週単位、月単位、季節単位といった需要や人々の行動の変容もシミュレーションできることが期待できる。また、形式仕様記述やその機械処理に関する計算機科学の進展や量子計算科学の進展は、計算可能な世界を飛躍的に広げる可能性もある。そのようなことが現実になれば、特定エリアのモビリティサービスの設計・評価に留まらず大規模な交通インフラ投資の意思決定や政策決定にも本研究で提案したような手法が適用可能になることが期待できる。今後は、様々なモビリティサービスの導入・設計に本研究で提案した手法を適用し方式をブラッシュアップするのみならず、上に述べた社会システムの設計・改善に適用できる手法へと発展させていきたい。

謝 辞

篠田陽一教授には、高信頼マルチキャスト、適応流量制御、無線網の模倣等、本研究を開始する前から多面的に研究開発の指導をいただいた。本研究については、社会人コースにて土日や祝日にも関わらず、長期にわたり広い視野と高い視点からご指導いただいた。緒方和博教授には、副テーマとして線形時相論理の世界に入門させていただいた。単なる情報空間の探索では達成できない活性特性 (liveness property) のモデル検査がオートマトンへの変換により達成可能となっている事実は、入門者としては衝撃の事実であり、偉大な計算機科学の進歩の存在、その先に広がる計算可能な世界の広がりを教えていただいた上で、さらには国際学会で発表できるレベルまで研究成果を引き上げていただいた。ロボットデリバリー協会の佐藤知正代表理事におかれては、自動配送ロボットの社会実装に向け安全基準の策定と前述の無線網の模倣技術を応用した検査方法を含む認証の仕組みの構築にあたりご指導いただき、政府が目指す日程に沿って道路交通法の施行を実現に可能に導いていただいた。また、本研究のドクター論文の発表に対しても丁寧にご指導をいただいた。名古屋大学の河口信夫教授には、多地点配送に関する研究や IRTF SAM-WG での標準化活動等、本研究の論文執筆への指導以外にも、多くのご指導をいただいた。宇多仁准教授には、1999 年の修士の在学時代から、FreeBSD の導入や WIDE プロジェクトの案内など先輩として研究開発の初期段階への導入でお世話になった。また、本研究の論文執筆あたっても指導を引き受けていただいた。パナソニックホールディングス株式会社の東島勝義氏は、Risk、Cost、Value の 3 指標でサービスの評価を捉えるべきであるとの問題意識を教えていただいた。九津見洋デジタル AI 技術センター長には副業制度等が確立する前の時代から私の社会人ドクターコースで本テーマを扱うことに同意いただいた。東京都公立大学法人高等専門学校の知念賢一教授には、クラウドへの高レートコンテンツアップロード機構の研究や、本研究の StarBED での実験や論文化に関して、パナソニックホールディングス株式会社河本弘和氏、株式会社パナソニックシステムネットワークス開発研究所澤井薫氏と共に研究の深化に協力いただいた。信号時相論理の本研究への導入にあたっては、国立情報学研究所の蓮尾一郎教授に有益な示唆をいただいた。NICT の安田真悟氏には、実験において、OS の複製方法等に関し助言をいただいた。宮地利幸北陸 StarBED 技術センター長はじめ同センター各位には、StarBED での検証実験にサポートをいただいた。パナソニックホールディングス株式会社モビリティ事業戦略室及びデジタル AI 技術センターの関係メンバーには、大阪地区の企業内エリアにおける実証のデータ提供や運行実証において、協力をいただいた。株式会社 PTV グループジャパンの三浦基嗣氏には Vissim の用法指導やライセンス供給でご協力いただいた。対話的な最適解の選択の実験においては、パナソニックホールディングス株式会社の同センター社員に協力をいただいた。また、本学の教務各位には、再入学を推奨いただき本研究のドクター論文化につながった。最後に妻晶子は土日を使った研究活動をいつも陰から支えてくれた。また、娘奈美は誤字チェックに力を貸してくれた。各位に感謝申し上げたい。

参考文献

[1] Ais: Abbreviated injury scale. https://www.aaam.org/abbreviated-injury-scale-ais/. (Accessed on 2023/08/12).

[2] Fujisawasst home page. https://fujisawasst.com/JP/. (Accessed on 2023/08/12).

[3] Htcondor. https://research.cs.wisc.edu/htcondor/. (Accessed on 2023/08/12).

[4] Jupyterlab. https://jupyter.org/. (Accessed on 2023/12/16).

[5] Mapbox. https://www.mapbox.com/. (Accessed on 2023/08/12).

[6] Maxar. http://www.digitalglobe.com/. (Accessed on 2023/08/12).

[7] Nacto, blueprint for autonomous urbanism. https://nacto.org/publication/bau2/. (Accessed on 2023/07/02).

[8] Open street map. https://www.openstreetmap.org/. (Accessed on 2023/08/12).

[9] Optuna homepage. https://optuna.readthedocs.io/en/stable/index.html. (Accessed on 2023/08/13).

[10] Pploty. https://plotly.com/python/. (Accessed on 2023/12/16).

[11] Ptv vissim. https://www.ptvgroup.com/en/solutions/products/ptv-vissim/. (Accessed on 2023/08/12).

[12] Python. https://www.python.org/. (Accessed on 2023/12/16).

[13] Simpy. https://simpy.readthedocs.io/en/latest/. (Accessed on 2023/12/16).

[14] 公益社団法人日本交通政策研究会講演「転換期の都市交通計画」. https://www.nikkoken.or.jp/pdf/symposium/JRCTP20190925.pdf. (Accessed on 2023/07/02).

[15] 国土交通省「2040 年、道路の景色が変わる ～人々の幸せにつながる道路～」. https://www.mlit.go.jp/road/vision/01.html. (Accessed on 2023/07/02).

[16] 自動配送ロボットの社会実装に向けて. https://www.meti.go.jp/press/2022/03/20230327001/20230327001-2.pdf. (Accessed on 2023/08/26).

[17] 藤沢地区での自動配送ロボット実証. https://news.panasonic.com/jp/press/data/2020/12/jn201207-2/jn201207-2.html. (Accessed on 2023/08/12).

[18] 日本初、届出制に基づく自動配送ロボットの運用を開始. https://news.panasonic.com/jp/press/jn230801-1. (Accessed on 2023/08/26).

[19] Raja Ben Abdessalem, Annibale Panichella, Shiva Nejati, Lionel C Briand, and Thomas Stifter. Testing autonomous cars for feature interaction failures using many-objective search. In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, pages 143–154, 2018.

[20] Nicole Adler, Eric Pels, and Chris Nash. High-speed rail and air transport competition: Game engineering as tool for cost-benefit analysis. Transportation Research Part B: Methodological, 44(7):812–833, 2010.

[21] Shaukat Ali, Paolo Arcaini, Dipesh Pradhan, Safdar Aqeel Safdar, and Tao Yue. Quality indicators in search-based software engineering: An empirical evaluation. ACM Transactions on Software Engineering and Methodology (TOSEM), 29(2):1–29, 2020.

[22] Allysson Allex Araújo, Matheus Paixao, Italo Yeltsin, Altino Dantas, and Jerffeson Souza. An architecture based on interactive optimization and machine learning applied to the next release problem. Automated Software Engineering, 24:623–671, 2017.

[23] Paolo Arcaini, Ezequiel Castellano, Fuyuki Ishikawa, Hirokazu Kawamoto, Kaoru Sawai, and Eiichi Muramoto. Incremental search-based allocation of autonomous robots for goods delivery. In 2023 IEEE Congress on Evolutionary Computation (CEC), pages 1–10, 2023.

[24] Kyungmin Bae and José Meseguer. Model checking linear temporal logic of rewriting formulas under localized fairness. Science of Computer Programming, 99:193–234, 2015.

[25] Fabio Bellifemine, Agostino Poggi, and Giovanni Rimassa. Jade–a fipa-compliant agent framework. In Proceedings of PAAM, volume 99, page 33. London, 1999.

[26] Alessandro Calò, Paolo Arcaini, Shaukat Ali, Florian Hauer, and Fuyuki Ishikawa. Generating avoidable collision scenarios for testing autonomous driving systems. In 2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST), pages 375–386. IEEE, 2020.

[27] Sagar Chaki, Edmund M. Clarke, Joël Ouaknine, Natasha Sharygina, and Nishant Sinha. State/event-based software model checking. In Eerke A. Boiten, John Derrick, and Graeme Smith, editors, Integrated Formal Methods, pages 128–147, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[28] Seongjin Choi and Hwasoo Yeo. Framework for simulation-based lane change control for autonomous vehicles. In 2017 IEEE intelligent vehicles symposium (IV), pages 699–704. IEEE, 2017.

[29] Manuel Clavel, Francisco Durán, Steven Eker, Patrick Lincoln, Narciso Martí-Oliet, José Meseguer, and Carolyn Talcott. All about maude-a high-performance logical framework: how to specify, program, and verify systems in rewriting logic, volume 4350. Springer, 2007.

[30] Krajzewicz Daniel, H Georg, and W Peter. Sumo (simulation of urban mobility) an open-source traffic simulation. In Proc. 4th Middle East Symposium on Simulation and Modelling, pages 183–187, 2002.

[31] Pedro M d'Orey, Ricardo Fernandes, and Michel Ferreira. Empirical evaluation of a dynamic and distributed taxi-sharing system. In 2012 15th International IEEE Conference on Intelligent Transportation Systems, pages 140–146. IEEE, 2012.

[32] Bayo Dosunmu. Delivering london 2012: transport demand forecasting. In Proceedings of the Institution of Civil Engineers-Transport, volume 165, pages 257–266. Thomas Telford Ltd, 2012.

[33] Jonas Eliasson. A cost–benefit analysis of the stockholm congestion charging system. Transportation Research Part A: Policy and Practice, 43(4):468–480, 2009.

[34] Georgios E Fainekos and George J Pappas. Robustness of temporal logic specifications for continuous-time signals. Theoretical Computer Science, 410(42):4262–4291, 2009.

[35] Lucas ER Fernandes, Vinicius Custodio, Gleifer V Alves, and Michael Fisher. A rational agent controlling an autonomous vehicle: Implementation and formal verification. arXiv preprint arXiv:1709.02557, 2017.

[36] Nathan Fulton, Stefan Mitsch, Jan-David Quesel, Marcus Völp, and André Platzer. Keymaera x: An axiomatic tactical theorem prover for hybrid systems. In Automated Deduction-CADE-25: 25th International Conference on Automated Deduction, Berlin, Germany, August 1-7, 2015, Proceedings 25, pages 527–538. Springer, 2015.

[37] Alessio Gambi, Marc Mueller, and Gordon Fraser. Automatically testing self-driving cars with search-based procedural content generation. In Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis, pages 318–328, 2019.

[38] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. Physical review E, 51(5):4282, 1995.

[39] LF Henderson. The statistics of crowd fluids. nature, 229(5284):381–383, 1971.

[40] Linda E. Karjalainen and Sirkku Juhola. Urban transportation sustainability assessments: a systematic review of literature. Transport Reviews, 41(5):659–684, 2021.

[41] Scott Kirkpatrick, C Daniel Gelatt Jr, and Mario P Vecchi. Optimization by simulated annealing. science, 220(4598):671–680, 1983.

[42] Terje Kristensen and Nnamdi Johnson Ezeora. Simulation of intelligent traffic control for autonomous vehicles. In 2017 IEEE International Conference on Information and Automation (ICIA), pages 459–465. IEEE, 2017.

[43] Antti Lajunen. Energy consumption and cost-benefit analysis of hybrid and electric city buses. Transportation Research Part C: Emerging Technologies, 38:1–15, 2014.

[44] Yixing Luo, Xiao-Yi Zhang, Paolo Arcaini, Zhi Jin, Haiyan Zhao, Fuyuki Ishikawa, Rongxin Wu, and Tao Xie. Targeting requirements violations of autonomous driving systems by dynamic evolutionary search. In 2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE), pages 279–291. IEEE, 2021.

[45] Oded Maler and Dejan Nickovic. Monitoring temporal properties of continuous signals. In International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems, pages 152–166. Springer, 2004.

[46] José Meseguer. Localized fairness: A rewriting semantics. In International Conference on Rewriting Techniques and Applications, pages 250–263. Springer, 2005.

[47] Stefan Mitsch, Khalil Ghorbal, David Vogelbacher, and André Platzer. Formal verification of obstacle avoidance and navigation of ground robots. The International Journal of Robotics Research, 36(12):1312–1340, 2017.

[48] Toshiyuki Miyachi, Ken-ichi Chinen, and Yoichi Shinoda. Starbed and springos: Large-scale general purpose network testbed and supporting software. In Proceedings of the 1st international conference on Performance evaluation methodolgies and tools, pages 30–es, 2006.

[49] Eiichi Muramoto, Kazuhiro Ogata, and Yoichi Shinoda. Formal specification and model checking of a ride-sharing system in maude. In International Workshop on Structured Object-Oriented Formal Language and Method, pages 187–204. Springer, 2019.

[50] Florian Neukart, Gabriele Compostella, Christian Seidel, David Von Dollen, Sheir Yarkoni, and Bob Parney. Traffic flow optimization using a quantum annealer. Frontiers in ICT, 4:29, 2017.

[51] Doug Newcomb. You won't need a driver's license by 2040. https://www.wired.com/2012/09/ieee-autonomous-2040/, 2012.

[52] Kazuhiro Ogata. Model checking liveness properties under fairness & anti-fairness assumptions. In 2013 20th Asia-Pacific Software Engineering Conference (APSEC), volume 1, pages 565–570. IEEE, 2013.

[53] Kazuhiro Ogata. A divide & conquer approach to liveness model checking under fairness & anti-fairness assumptions. Frontiers of Computer Science, 13:51–72, 2019.

[54] Kazuhiro Ogata and Kokichi Futatsugi. Comparison of maude and sal by conducting case studies model checking a distributed algorithm. IEICE transactions on fundamentals of electronics, communications and computer sciences, 90(8):1690–1703, 2007.

[55] Yoshihiko Ozaki, Yuki Tanigaki, Shuhei Watanabe, and Masaki Onishi. Multiobjective tree-structured parzen estimator for computationally expensive optimization problems. In Proceedings of the 2020 genetic and evolutionary computation conference, pages 533–541, 2020.

[56] Vasumathi Raman, Alexandre Donzé, Mehdi Maasoumy, Richard M Murray, Alberto Sangiovanni-Vincentelli, and Sanjit A Seshia. Model predictive control with signal temporal logic specifications. In 53rd IEEE Conference on Decision and Control, pages 81–87. IEEE, 2014.

[57] Aurora Ramirez, Jose Raul Romero, and Christopher L Simons. A systematic review of interaction in search-based software engineering. IEEE Transactions on Software Engineering, 45(8):760–781, 2018.

[58] Sota Sato, Atsuyoshi Saimen, Masaki Waga, Kenji Takao, and Ichiro Hasuo. Hybrid system falsification for multiple-constraint parameter synthesis: A gas turbine case study. In Marieke Huisman, Corina Păsăreanu, and Naijun Zhan, editors, Formal Methods, pages 313–329, Cham, 2021. Springer International Publishing.

[59] Wai Yuen Szeto and Yu Jiang. Transit route and frequency design: Bi-level modeling and hybrid artificial bee colony algorithm approach. Transportation Research Part B: Methodological, 67:235–263, 2014.

[60] Mauricio Byrd Victorica, Paolo Arcaini, Fuyuki Ishikawa, Hirokazu Kawamoto, Kaoru Sawai, and Eiichi Muramoto. Stability-aware exploration of design space of autonomous robots for goods delivery. In 2023 27th International Conference on Engineering of Complex Computer Systems (ICECCS), pages 177–186, 2023.

[61] Chi Zhang, Yuehu Liu, Danchen Zhao, and Yuanqi Su. Roadview: A traffic scene simulator for autonomous vehicle simulation testing. In 17th International IEEE Conference on Intelligent Transportation Systems (ITSC), pages 1160–1165. IEEE, 2014.

[62] 岡野舜, 高山宇宙, 三浦清洋, and 森本章倫. レベル 4 の自動運転車導入における乗降環境を考慮した街路空間に関する研究. 交通工学論文集, 6(2):A_105–A_112, 2020.

[63] 亀山章. 高速道路インターチェンジ周辺の土地利用の変遷. 信州大学農学部紀要, 25(2):85–100, 1988.

[64] 潮 俊光 金島 昂輝. 時相論理制約を用いた online pickup and delivery 問題. In 電子情報通信学会 技術研究報告 (信学技報, システム数理と応用研究会 MSS2021-11) vol. 121, no. 92, pages 54–59, 2021.

[65] 桑原昌広, 吉岡顕, 松本浩和, and 早田敏也. 公共交通連携向けワンウェイ型カーシェアリングのステーション候補探索手法提案・検証. 交通工学論文集, 6(2):B_11–B_18, 2020.

[66] 警察庁. 令和 4 年改正道路交通法（遠隔操作型小型車の交通方法等）の概要. https://www.npa.go.jp/bureau/traffic/selfdriving/roadtesting/kaiseidourokoutuuhou.pdf. (Accessed on 2023/07/02).

[67] 幸田亮一. ドイツにおける第二次産業革命と「経営科学」 : G・シュレジンガー再考. 産業経営研究 = Studies of Economics and Business, 41:57–73, 03 2022.

[68] 国土交通省. 交通分野における sdgs 実現に向けた取組. https://www.mlit.go.jp/report/press/content/001428153.pdf. (Accessed on 2023/07/02).

[69] 国土交通省道路局. 多様なニーズに応える道路空間のあり方に関する検討会について. https://www.mlit.go.jp/road/ir/ir-council/diverse_needs/pdf01/04.pdf. (Accessed on 2023/08/12).

[70] 国土交通省道路局. 費用便益分析マニュアル. https://www.mlit.go.jp/road/ir/ir-hyouka/ben-eki_2.pdf. (Accessed on 2023/07/02).

[71] 藤井 聡 根津 佳樹. 交通インフラ投資によるマクロ経済への影響分析のためのシミュレーションモデル masrac の構築. 科学・技術研究, 5(2):185–195, 2016.

[72] 山本真之, 梶大介, 金森亮, and 落合純一. 都市部における自動運転ライドシェアのシミュレーション分析. Denso technical review/デンソーテクニカルレビュー 編, 24:36–41, 2019.

[73] 村本衛一, 河本弘和, 東島勝義, 古川量也, 澤井薫, 知念賢一, 篠田陽一, and 三浦基嗣. 移動手段が選択可能な生活圏における自動運転電動車両の運行のサービス設計・評価手法の提案. 情報処理学会論文誌コンシューマ・デバイス & システム (CDS), 12(2):1–11, 2022.

[74] 村本衛一, 河本弘和, 東島勝義, 澤井薫, and 篠田陽一. 自動配送ロボットによる配送サービスのインタラクティブな設計・評価手法の提案. 情報処理学会論文誌コンシューマ・デバイス & システム (CDS), 13(2):20–30, 2023.

[75] 太田悟史, 安田真悟, 湯村翼, and 高野祐輝. 次世代サイバー演習環境に向けて. マルチメディア, 分散協調とモバイルシンポジウム 2016 論文集, 2016:1776–1782, 2016.

[76] 大口敬, 赤羽弘和, et al. 道路交通技術必携 2018. 交通工学研究会, Cambridge, UK, 2018.

[77] 波床 正敏 中川 大, 西村 嘉浩. 鉄道整備が市町村人口の変遷に及ぼしてきた影響に関する実証的研究. 土木計画学研究・論文集, 11:57–64, 1993.

[78] 東島勝義. コミュニティを支える次世代モビリティサービス. 都市計画, 7(6 353 号):74–75, 2021.

[79] 東島勝義 本田義雅. 自動運転サービス開発と当社構内でのサービス実証 (オートモーティブ特集). パナソニック技報= Panasonic technical journal, 67(1):84–86, 2021.

Appendix

Maudeによる安全性検証、活性特性検証の計算時間の計測

形式仕様による研究を今後引継いでくれる方々を想定して、本研究の 4 章で開発したモデル検査のソースコードを示す。具体的には 4.6.3 の計算時間の計測で用いた Maude のソースコードを以下に示す。利用した Maude のバージョンは 2.0 である。

```
***
*** reserve cars 11th-version
***              with fairness assumption on car
***              on map 11 locaion (m11) m6=short dividing version
***     This source code based on Maude verion 2.0
***
*** the functional modules
***
fmod NODE is sort Node .
 ops na nb nc nd ne nf ng nh ni nj nk emptynode : -> Node [ctor] .
endfm

fmod SOUP {D :: TRIV} is
  sort Soup{D} .
  subsort D$Elt < Soup{D} .
  op empty : -> Soup{D} [ctor] .
  op _ _ : Soup{D} Soup{D} -> Soup{D} [ctor assoc comm id: empty] .
  op _\in_ : D$Elt Soup{D} -> Bool .
  var S : Soup{D} .
  vars E E1 E2 : D$Elt .
  eq E E = E . *** idempotent
  eq E \in (E S) = true .
  eq E \in S = false [owise] .
endfm

fmod LIST {D :: TRIV} is
```

```
  sorts List{D} NnList{D} .
  subsorts NnList{D} < List{D} .
  op empty : -> List{D} [ctor] .
  op _ | _ : D$Elt List{D} -> NnList{D} [ctor] .
  op _ @ _ : List{D} List{D} -> List{D} .
  op top : NnList{D} -> D$Elt .
  op tail : List{D} -> List{D} .
  vars L L2 : List{D} .
  var E : D$Elt .
  eq empty @ L = L .
  eq (E | L) @ L2 = E | ( L @ L2 ) .
  eq top ( E | L ) = E .
  eq tail ( empty ) = empty .
  eq tail ( E | L ) = L .
endfm

view Node  from TRIV to NODE is
 sort Elt to Node .
endv

***
*** Definision of shortest path  on the MAP
***

fmod ROUTEENTRY is sort Routeentry .
 pr NODE . pr NAT . pr LIST{Node} .
 op routeentry : List{Node} Nat -> Routeentry [ctor] .
 op dist : Routeentry -> Nat .
 op route : Routeentry -> NnList{Node} .
 op _ isshorter _ : Routeentry Routeentry -> Bool .
 vars RE RE1 RE2 :  Routeentry . var D : Nat .
 var NL : List{Node} .
 eq dist( routeentry(NL,D) ) = D .
 ceq RE1 isshorter RE2 = true if dist(RE1) < dist(RE2) .
 eq RE1 isshorter RE2 = false [owise] .
 eq route( routeentry(NL,D) ) = NL .
```

```
endfm

view Routeentry  from TRIV to ROUTEENTRY is
 sort Elt to Routeentry .
endv

fmod DBENTRY is sort Dbentry .
 pr NODE .
 pr LIST{Routeentry} .
 op dbentry : Node Node List{Routeentry}  -> Dbentry [ctor] .
 op src : Dbentry -> Node .
 op dst : Dbentry -> Node .
 op rtes : Dbentry -> List{Routeentry} .
 vars N1 N2 : Node . var RES : List{Routeentry} .
 eq src(dbentry(N1,N2,RES)) = N1 .
 eq dst(dbentry(N1,N2,RES)) = N2 .
 eq rtes(dbentry(N1,N2,RES)) = RES .
endfm

view Dbentry  from TRIV to DBENTRY is
 sort Elt to Dbentry .
endv

***
*** Define Map as soup of route entry
***

fmod DBENTRY-M6 is
 pr DBENTRY . pr SOUP{Dbentry} .
 op routedb-m6 : -> Soup{Dbentry} .
 eq routedb-m6  = (
d6nana d6nanb d6nanc d6nand d6nane d6nanf d6nbna d6nbnb d6nbnc d6nbnd d6nbne d6nbnf d6ncna
d6ncnb d6ncnc d6ncnd d6ncne d6ncnf d6ndna d6ndnb d6ndnc d6ndnd d6ndne d6ndnf d6nena d6nenb
d6nenc d6nend d6nene d6nenf d6nfna d6nfnb d6nfnc d6nfnd d6nfne d6nfnf  ) .
 op d6nana : -> Dbentry .
 eq d6nana = dbentry( na, na, routeentry( na | empty, 0) |  empty ) .
```

```
op d6nanb : -> Dbentry .
eq d6nanb =  dbentry( na, nb, routeentry( na | nb | empty , 1 ) |  empty ) .
op d6nanc : -> Dbentry .
eq d6nanc =  dbentry( na, nc, routeentry( na | nb | nc | empty , 2 ) |  empty ) .
op d6nand : -> Dbentry .
eq d6nand =  dbentry( na, nd, routeentry( na | nb | nd | empty , 2 ) |  empty ) .
op d6nane : -> Dbentry .
eq d6nane =  dbentry( na, ne, routeentry( na | nb | nd | ne | empty , 3 ) |  empty ) .
op d6nanf : -> Dbentry .
eq d6nanf =  dbentry( na, nf, routeentry( na | nb | nc | nf | empty , 3 ) |  empty ) .
op d6nbna : -> Dbentry .
eq d6nbna =  dbentry( nb, na, routeentry( nb | nc | na | empty , 2 ) |  empty ) .
op d6nbnb : -> Dbentry .
eq d6nbnb = dbentry( nb, nb, routeentry( nb | empty, 0) |  empty ) .
op d6nbnc : -> Dbentry .
eq d6nbnc =  dbentry( nb, nc, routeentry( nb | nc | empty , 1 ) |  empty ) .
op d6nbnd : -> Dbentry .
eq d6nbnd =  dbentry( nb, nd, routeentry( nb | nd | empty , 1 ) |  empty ) .
op d6nbne : -> Dbentry .
eq d6nbne =  dbentry( nb, ne, routeentry( nb | nd | ne | empty , 2 ) |  empty ) .
op d6nbnf : -> Dbentry .
eq d6nbnf =  dbentry( nb, nf, routeentry( nb | nc | nf | empty , 2 ) |  empty ) .
op d6ncna : -> Dbentry .
eq d6ncna =  dbentry( nc, na, routeentry( nc | na | empty , 1 ) |  empty ) .
op d6ncnb : -> Dbentry .
eq d6ncnb =  dbentry( nc, nb, routeentry( nc | na | nb | empty , 2 ) |  empty ) .
op d6ncnc : -> Dbentry .
eq d6ncnc = dbentry( nc, nc, routeentry( nc | empty, 0) |  empty ) .
op d6ncnd : -> Dbentry .
eq d6ncnd =  dbentry( nc, nd, routeentry( nc | na | nb | nd | empty , 3 ) |  empty ) .
op d6ncne : -> Dbentry .
eq d6ncne =  dbentry( nc, ne, routeentry( nc | na | nb | nd | ne | empty , 4 ) |  empty).
op d6ncnf : -> Dbentry .
eq d6ncnf =  dbentry( nc, nf, routeentry( nc | nf | empty , 1 ) |  empty ) .
op d6ndna : -> Dbentry .
eq d6ndna =  dbentry( nd, na, routeentry( nd | ne | nf | nb | nc | na | empty , 5 ) |
```

```
 empty ) .
 op d6ndnb : -> Dbentry .
 eq d6ndnb =  dbentry( nd, nb, routeentry( nd | ne | nf | nb | empty , 3 ) |  empty ) .
 op d6ndnc : -> Dbentry .
 eq d6ndnc =  dbentry( nd, nc, routeentry( nd | ne | nf | nb | nc | empty , 4 ) |  empty).
 op d6ndnd : -> Dbentry .
 eq d6ndnd = dbentry( nd, nd, routeentry( nd | empty, 0) |  empty ) .
 op d6ndne : -> Dbentry .
 eq d6ndne =  dbentry( nd, ne, routeentry( nd | ne | empty , 1 ) |  empty ) .
 op d6ndnf : -> Dbentry .
 eq d6ndnf =  dbentry( nd, nf, routeentry( nd | ne | nf | empty , 2 ) |  empty ) .
 op d6nena : -> Dbentry .
 eq d6nena =  dbentry( ne, na, routeentry( ne | nf | nb | nc | na | empty , 4 ) |  empty).
 op d6nenb : -> Dbentry .
 eq d6nenb =  dbentry( ne, nb, routeentry( ne | nf | nb | empty , 2 ) |  empty ) .
 op d6nenc : -> Dbentry .
 eq d6nenc =  dbentry( ne, nc, routeentry( ne | nf | nb | nc | empty , 3 ) |  empty ) .
 op d6nend : -> Dbentry .
 eq d6nend =  dbentry( ne, nd, routeentry( ne | nf | nb | nd | empty , 3 ) |  empty ) .
 op d6nene : -> Dbentry .
 eq d6nene = dbentry( ne, ne, routeentry( ne | empty, 0) |  empty ) .
 op d6nenf : -> Dbentry .
 eq d6nenf =  dbentry( ne, nf, routeentry( ne | nf | empty , 1 ) |  empty ) .
 op d6nfna : -> Dbentry .
 eq d6nfna =  dbentry( nf, na, routeentry( nf | nb | nc | na | empty , 3 ) |  empty ) .
 op d6nfnb : -> Dbentry .
 eq d6nfnb =  dbentry( nf, nb, routeentry( nf | nb | empty , 1 ) |  empty ) .
 op d6nfnc : -> Dbentry .
 eq d6nfnc =  dbentry( nf, nc, routeentry( nf | nb | nc | empty , 2 ) |  empty ) .
 op d6nfnd : -> Dbentry .
 eq d6nfnd =  dbentry( nf, nd, routeentry( nf | nb | nd | empty , 2 ) |  empty ) .
 op d6nfne : -> Dbentry .
 eq d6nfne =  dbentry( nf, ne, routeentry( nf | nb | nd | ne | empty , 3 ) |  empty ) .
 op d6nfnf : -> Dbentry .
 eq d6nfnf = dbentry( nf, nf, routeentry( nf | empty, 0) |  empty ) .
endfm
```

```
fmod DBENTRY-M6P is
 pr DBENTRY . pr SOUP{Dbentry} .
 op routedb-m6p : -> Soup{Dbentry} .
 eq routedb-m6p  = (
dbnana dbnanb dbnanc dbnand dbnane dbnanf dbnbna dbnbnb dbnbnc dbnbnd dbnbne dbnbnf dbncna
dbncnb dbncnc dbncnd dbncne dbncnf dbndna dbndnb dbndnc dbndnd dbndne dbndnf dbnena dbnenb
dbnenc dbnend dbnene dbnenf dbnfna dbnfnb dbnfnc dbnfnd dbnfne dbnfnf  ) .
 op dbnana : -> Dbentry .
 eq dbnana = dbentry( na, na, routeentry( na | empty, 0) |  empty ) .
 op dbnanb : -> Dbentry .
 eq dbnanb =  dbentry( na, nb, routeentry( na | nb | empty , 1 ) |  empty ) .
 op dbnanc : -> Dbentry .
 eq dbnanc =  dbentry( na, nc, routeentry( na | nb | nc | empty , 2 ) |  empty ) .
 op dbnand : -> Dbentry .
 eq dbnand =  dbentry( na, nd, routeentry( na | nb | nd | empty , 2 ) |  empty ) .
 op dbnane : -> Dbentry .
 eq dbnane =  dbentry( na, ne, routeentry( na | nb | nc | ne | empty , 3 ) |  empty ) .
 op dbnanf : -> Dbentry .
 eq dbnanf =  dbentry( na, nf, routeentry( na | nb | nc | nf | empty , 3 ) |  empty ) .
 op dbnbna : -> Dbentry .
 eq dbnbna =  dbentry( nb, na, routeentry( nb | nc | na | empty , 2 ) |  empty ) .
 op dbnbnb : -> Dbentry .
 eq dbnbnb = dbentry( nb, nb, routeentry( nb | empty, 0) |  empty ) .
 op dbnbnc : -> Dbentry .
 eq dbnbnc =  dbentry( nb, nc, routeentry( nb | nc | empty , 1 ) |  empty ) .
 op dbnbnd : -> Dbentry .
 eq dbnbnd =  dbentry( nb, nd, routeentry( nb | nd | empty , 1 ) |  empty ) .
 op dbnbne : -> Dbentry .
 eq dbnbne =  dbentry( nb, ne, routeentry( nb | nc | ne | empty , 2 ) |  empty ) .
 op dbnbnf : -> Dbentry .
 eq dbnbnf =  dbentry( nb, nf, routeentry( nb | nc | nf | empty , 2 ) |  empty ) .
 op dbncna : -> Dbentry .
 eq dbncna =  dbentry( nc, na, routeentry( nc | na | empty , 1 ) |  empty ) .
 op dbncnb : -> Dbentry .
 eq dbncnb =  dbentry( nc, nb, routeentry( nc | na | nb | empty , 2 ) |  empty ) .
```

```
op dbncnc : -> Dbentry .
eq dbncnc = dbentry( nc, nc, routeentry( nc | empty, 0) |  empty ) .
op dbncnd : -> Dbentry .
eq dbncnd =  dbentry( nc, nd, routeentry( nc | nf | nd | empty , 2 ) |  empty ) .
op dbncne : -> Dbentry .
eq dbncne =  dbentry( nc, ne, routeentry( nc | ne | empty , 1 ) |  empty ) .
op dbncnf : -> Dbentry .
eq dbncnf =  dbentry( nc, nf, routeentry( nc | nf | empty , 1 ) |  empty ) .
op dbndna : -> Dbentry .
eq dbndna =  dbentry( nd, na, routeentry( nd | ne | nf | nb | nc | na | empty , 5 )
|  empty ) .
op dbndnb : -> Dbentry .
eq dbndnb =  dbentry( nd, nb, routeentry( nd | ne | nf | nb | empty , 3 ) |  empty ) .
op dbndnc : -> Dbentry .
eq dbndnc =  dbentry( nd, nc, routeentry( nd | ne | nf | nb | nc | empty , 4 ) |  empty).
op dbndnd : -> Dbentry .
eq dbndnd = dbentry( nd, nd, routeentry( nd | empty, 0) |  empty ) .
op dbndne : -> Dbentry .
eq dbndne =  dbentry( nd, ne, routeentry( nd | ne | empty , 1 ) |  empty ) .
op dbndnf : -> Dbentry .
eq dbndnf =  dbentry( nd, nf, routeentry( nd | ne | nf | empty , 2 ) |  empty ) .
op dbnena : -> Dbentry .
eq dbnena =  dbentry( ne, na, routeentry( ne | nf | nb | nc | na | empty , 4 ) |  empty).
op dbnenb : -> Dbentry .
eq dbnenb =  dbentry( ne, nb, routeentry( ne | nf | nb | empty , 2 ) |  empty ) .
op dbnenc : -> Dbentry .
eq dbnenc =  dbentry( ne, nc, routeentry( ne | nf | nb | nc | empty , 3 ) |  empty ) .
op dbnend : -> Dbentry .
eq dbnend =  dbentry( ne, nd, routeentry( ne | nf | nd | empty , 2 ) |  empty ) .
op dbnene : -> Dbentry .
eq dbnene = dbentry( ne, ne, routeentry( ne | empty, 0) |  empty ) .
op dbnenf : -> Dbentry .
eq dbnenf =  dbentry( ne, nf, routeentry( ne | nf | empty , 1 ) |  empty ) .
op dbnfna : -> Dbentry .
eq dbnfna =  dbentry( nf, na, routeentry( nf | nb | nc | na | empty , 3 ) |  empty ) .
op dbnfnb : -> Dbentry .
```

```
 eq dbnfnb =  dbentry( nf, nb, routeentry( nf | nb | empty , 1 ) |  empty ) .
 op dbnfnc : -> Dbentry .
 eq dbnfnc =  dbentry( nf, nc, routeentry( nf | nb | nc | empty , 2 ) |  empty ) .
 op dbnfnd : -> Dbentry .
 eq dbnfnd =  dbentry( nf, nd, routeentry( nf | nd | empty , 1 ) |  empty ) .
 op dbnfne : -> Dbentry .
 eq dbnfne =  dbentry( nf, ne, routeentry( nf | nd | ne | empty , 2 ) |  empty ) .
 op dbnfnf : -> Dbentry .
 eq dbnfnf = dbentry( nf, nf, routeentry( nf | empty, 0) |  empty ) .
endfm

fmod DBENTRY-M9 is
 pr DBENTRY . pr SOUP{Dbentry} .
 op routedb-m9 : -> Soup{Dbentry} .
 eq routedb-m9  = (
d9nana d9nanb d9nanc d9nand d9nane d9nanf d9nang d9nanh d9nani d9nbna d9nbnb d9nbnc d9nbnd
d9nbne d9nbnf d9nbng d9nbnh d9nbni d9ncna d9ncnb d9ncnc d9ncnd d9ncne d9ncnf d9ncng d9ncnh
d9ncni d9ndna d9ndnb d9ndnc d9ndnd d9ndne d9ndnf d9ndng d9ndnh d9ndni d9nena d9nenb d9nenc
d9nend d9nene d9nenf d9neng d9nenh d9neni d9nfna d9nfnb d9nfnc d9nfnd d9nfne d9nfnf d9nfng
d9nfnh d9nfni d9ngna d9ngnb d9ngnc d9ngnd d9ngne d9ngnf d9ngng d9ngnh d9ngni d9nhna d9nhnb
d9nhnc d9nhnd d9nhne d9nhnf d9nhng d9nhnh d9nhni d9nina d9ninb d9ninc d9nind d9nine d9ninf
d9ning d9ninh d9nini  ) .
 op d9nana : -> Dbentry .
 eq d9nana = dbentry( na, na, routeentry( na | empty, 0) |  empty ) .
 op d9nanb : -> Dbentry .
 eq d9nanb =  dbentry( na, nb, routeentry( na | nb | empty , 1 ) |  empty ) .
 op d9nanc : -> Dbentry .
 eq d9nanc =  dbentry( na, nc, routeentry( na | nb | nc | empty , 2 ) |  empty ) .
 op d9nand : -> Dbentry .
 eq d9nand =  dbentry( na, nd, routeentry( na | nb | nd | empty , 2 ) |  empty ) .
 op d9nane : -> Dbentry .
 eq d9nane =  dbentry( na, ne, routeentry( na | nb | nc | ne | empty , 3 ) |  empty ) .
 op d9nanf : -> Dbentry .
 eq d9nanf =  dbentry( na, nf, routeentry( na | nb | nc | nf | empty , 3 ) |  empty ) .
 op d9nang : -> Dbentry .
 eq d9nang =  dbentry( na, ng, routeentry( na | nb | ng | empty , 2 ) |  empty ) .
```

```
op d9nanh : -> Dbentry .
eq d9nanh =  dbentry( na, nh, routeentry( na | nb | nd | nh | empty , 3 ) |  empty ) .
op d9nani : -> Dbentry .
eq d9nani =  dbentry( na, ni, routeentry( na | nb | nc | ne | ni | empty , 4 ) |  empty).
op d9nbna : -> Dbentry .
eq d9nbna =  dbentry( nb, na, routeentry( nb | nc | na | empty , 2 ) |  empty ) .
op d9nbnb : -> Dbentry .
eq d9nbnb = dbentry( nb, nb, routeentry( nb | empty, 0) |  empty ) .
op d9nbnc : -> Dbentry .
eq d9nbnc =  dbentry( nb, nc, routeentry( nb | nc | empty , 1 ) |  empty ) .
op d9nbnd : -> Dbentry .
eq d9nbnd =  dbentry( nb, nd, routeentry( nb | nd | empty , 1 ) |  empty ) .
op d9nbne : -> Dbentry .
eq d9nbne =  dbentry( nb, ne, routeentry( nb | nc | ne | empty , 2 ) |  empty ) .
op d9nbnf : -> Dbentry .
eq d9nbnf =  dbentry( nb, nf, routeentry( nb | nc | nf | empty , 2 ) |  empty ) .
op d9nbng : -> Dbentry .
eq d9nbng =  dbentry( nb, ng, routeentry( nb | ng | empty , 1 ) |  empty ) .
op d9nbnh : -> Dbentry .
eq d9nbnh =  dbentry( nb, nh, routeentry( nb | nd | nh | empty , 2 ) |  empty ) .
op d9nbni : -> Dbentry .
eq d9nbni =  dbentry( nb, ni, routeentry( nb | nc | ne | ni | empty , 3 ) |  empty ) .
op d9ncna : -> Dbentry .
eq d9ncna =  dbentry( nc, na, routeentry( nc | na | empty , 1 ) |  empty ) .
op d9ncnb : -> Dbentry .
eq d9ncnb =  dbentry( nc, nb, routeentry( nc | na | nb | empty , 2 ) |  empty ) .
op d9ncnc : -> Dbentry .
eq d9ncnc = dbentry( nc, nc, routeentry( nc | empty, 0) |  empty ) .
op d9ncnd : -> Dbentry .
eq d9ncnd =  dbentry( nc, nd, routeentry( nc | nf | nd | empty , 2 ) |  empty ) .
op d9ncne : -> Dbentry .
eq d9ncne =  dbentry( nc, ne, routeentry( nc | ne | empty , 1 ) |  empty ) .
op d9ncnf : -> Dbentry .
eq d9ncnf =  dbentry( nc, nf, routeentry( nc | nf | empty , 1 ) |  empty ) .
op d9ncng : -> Dbentry .
eq d9ncng =  dbentry( nc, ng, routeentry( nc | na | nb | ng | empty , 3 ) |  empty ) .
```

```
op d9ncnh : -> Dbentry .
eq d9ncnh =  dbentry( nc, nh, routeentry( nc | nf | nd | nh | empty , 3 ) |  empty ) .
op d9ncni : -> Dbentry .
eq d9ncni =  dbentry( nc, ni, routeentry( nc | ne | ni | empty , 2 ) |  empty ) .
op d9ndna : -> Dbentry .
eq d9ndna =  dbentry( nd, na, routeentry( nd | ne | ni | nc | na | empty , 4 ) |  empty).
op d9ndnb : -> Dbentry .
eq d9ndnb =  dbentry( nd, nb, routeentry( nd | ne | nf | nb | empty , 3 ) |  empty ) .
op d9ndnc : -> Dbentry .
eq d9ndnc =  dbentry( nd, nc, routeentry( nd | ne | ni | nc | empty , 3 ) |  empty ) .
op d9ndnd : -> Dbentry .
eq d9ndnd = dbentry( nd, nd, routeentry( nd | empty, 0) |  empty ) .
op d9ndne : -> Dbentry .
eq d9ndne =  dbentry( nd, ne, routeentry( nd | ne | empty , 1 ) |  empty ) .
op d9ndnf : -> Dbentry .
eq d9ndnf =  dbentry( nd, nf, routeentry( nd | ne | nf | empty , 2 ) |  empty ) .
op d9ndng : -> Dbentry .
eq d9ndng =  dbentry( nd, ng, routeentry( nd | ne | nf | nb | ng | empty , 4 ) |  empty).
op d9ndnh : -> Dbentry .
eq d9ndnh =  dbentry( nd, nh, routeentry( nd | nh | empty , 1 ) |  empty ) .
op d9ndni : -> Dbentry .
eq d9ndni =  dbentry( nd, ni, routeentry( nd | ne | ni | empty , 2 ) |  empty ) .
op d9nena : -> Dbentry .
eq d9nena =  dbentry( ne, na, routeentry( ne | ni | nc | na | empty , 3 ) |  empty ) .
op d9nenb : -> Dbentry .
eq d9nenb =  dbentry( ne, nb, routeentry( ne | nf | nb | empty , 2 ) |  empty ) .
op d9nenc : -> Dbentry .
eq d9nenc =  dbentry( ne, nc, routeentry( ne | ni | nc | empty , 2 ) |  empty ) .
op d9nend : -> Dbentry .
eq d9nend =  dbentry( ne, nd, routeentry( ne | nf | nd | empty , 2 ) |  empty ) .
op d9nene : -> Dbentry .
eq d9nene = dbentry( ne, ne, routeentry( ne | empty, 0) |  empty ) .
op d9nenf : -> Dbentry .
eq d9nenf =  dbentry( ne, nf, routeentry( ne | nf | empty , 1 ) |  empty ) .
op d9neng : -> Dbentry .
eq d9neng =  dbentry( ne, ng, routeentry( ne | nf | nb | ng | empty , 3 ) |  empty ) .
```

```
op d9nenh : -> Dbentry .
eq d9nenh =  dbentry( ne, nh, routeentry( ne | nf | nd | nh | empty , 3 ) |  empty ) .
op d9neni : -> Dbentry .
eq d9neni =  dbentry( ne, ni, routeentry( ne | ni | empty , 1 ) |  empty ) .
op d9nfna : -> Dbentry .
eq d9nfna =  dbentry( nf, na, routeentry( nf | nb | nc | na | empty , 3 ) |  empty ) .
op d9nfnb : -> Dbentry .
eq d9nfnb =  dbentry( nf, nb, routeentry( nf | nb | empty , 1 ) |  empty ) .
op d9nfnc : -> Dbentry .
eq d9nfnc =  dbentry( nf, nc, routeentry( nf | nb | nc | empty , 2 ) |  empty ) .
op d9nfnd : -> Dbentry .
eq d9nfnd =  dbentry( nf, nd, routeentry( nf | nd | empty , 1 ) |  empty ) .
op d9nfne : -> Dbentry .
eq d9nfne =  dbentry( nf, ne, routeentry( nf | nd | ne | empty , 2 ) |  empty ) .
op d9nfnf : -> Dbentry .
eq d9nfnf = dbentry( nf, nf, routeentry( nf | empty, 0) |  empty ) .
op d9nfng : -> Dbentry .
eq d9nfng =  dbentry( nf, ng, routeentry( nf | nb | ng | empty , 2 ) |  empty ) .
op d9nfnh : -> Dbentry .
eq d9nfnh =  dbentry( nf, nh, routeentry( nf | nd | nh | empty , 2 ) |  empty ) .
op d9nfni : -> Dbentry .
eq d9nfni =  dbentry( nf, ni, routeentry( nf | nd | ne | ni | empty , 3 ) |  empty ) .
op d9ngna : -> Dbentry .
eq d9ngna =  dbentry( ng, na, routeentry( ng | nd | ne | ni | nc | na | empty , 5 ) |
  empty ) .
op d9ngnb : -> Dbentry .
eq d9ngnb =  dbentry( ng, nb, routeentry( ng | nd | ne | nf | nb | empty , 4 ) |  empty).
op d9ngnc : -> Dbentry .
eq d9ngnc =  dbentry( ng, nc, routeentry( ng | nd | ne | ni | nc | empty , 4 ) |  empty).
op d9ngnd : -> Dbentry .
eq d9ngnd =  dbentry( ng, nd, routeentry( ng | nd | empty , 1 ) |  empty ) .
op d9ngne : -> Dbentry .
eq d9ngne =  dbentry( ng, ne, routeentry( ng | nd | ne | empty , 2 ) |  empty ) .
op d9ngnf : -> Dbentry .
eq d9ngnf =  dbentry( ng, nf, routeentry( ng | nd | ne | nf | empty , 3 ) |  empty ) .
op d9ngng : -> Dbentry .
```

```
eq d9ngng = dbentry( ng, ng, routeentry( ng | empty, 0) |  empty ) .
op d9ngnh : -> Dbentry .
eq d9ngnh =  dbentry( ng, nh, routeentry( ng | nd | nh | empty , 2 ) |  empty ) .
op d9ngni : -> Dbentry .
eq d9ngni =  dbentry( ng, ni, routeentry( ng | nd | ne | ni | empty , 3 ) |  empty ) .
op d9nhna : -> Dbentry .
eq d9nhna =  dbentry( nh, na, routeentry( nh | ne | ni | nc | na | empty , 4 ) |  empty).
op d9nhnb : -> Dbentry .
eq d9nhnb =  dbentry( nh, nb, routeentry( nh | ne | nf | nb | empty , 3 ) |  empty ) .
op d9nhnc : -> Dbentry .
eq d9nhnc =  dbentry( nh, nc, routeentry( nh | ne | ni | nc | empty , 3 ) |  empty ) .
op d9nhnd : -> Dbentry .
eq d9nhnd =  dbentry( nh, nd, routeentry( nh | ne | nf | nd | empty , 3 ) |  empty ) .
op d9nhne : -> Dbentry .
eq d9nhne =  dbentry( nh, ne, routeentry( nh | ne | empty , 1 ) |  empty ) .
op d9nhnf : -> Dbentry .
eq d9nhnf =  dbentry( nh, nf, routeentry( nh | ne | nf | empty , 2 ) |  empty ) .
op d9nhng : -> Dbentry .
eq d9nhng =  dbentry( nh, ng, routeentry( nh | ne | nf | nb | ng | empty , 4 ) |  empty).
op d9nhnh : -> Dbentry .
eq d9nhnh = dbentry( nh, nh, routeentry( nh | empty, 0) |  empty ) .
op d9nhni : -> Dbentry .
eq d9nhni =  dbentry( nh, ni, routeentry( nh | ne | ni | empty , 2 ) |  empty ) .
op d9nina : -> Dbentry .
eq d9nina =  dbentry( ni, na, routeentry( ni | nc | na | empty , 2 ) |  empty ) .
op d9ninb : -> Dbentry .
eq d9ninb =  dbentry( ni, nb, routeentry( ni | nc | na | nb | empty , 3 ) |  empty ) .
op d9ninc : -> Dbentry .
eq d9ninc =  dbentry( ni, nc, routeentry( ni | nc | empty , 1 ) |  empty ) .
op d9nind : -> Dbentry .
eq d9nind =  dbentry( ni, nd, routeentry( ni | nc | nf | nd | empty , 3 ) |  empty ) .
op d9nine : -> Dbentry .
eq d9nine =  dbentry( ni, ne, routeentry( ni | nc | ne | empty , 2 ) |  empty ) .
op d9ninf : -> Dbentry .
eq d9ninf =  dbentry( ni, nf, routeentry( ni | nc | nf | empty , 2 ) |  empty ) .
op d9ning : -> Dbentry .
```

```
 eq d9ning =  dbentry( ni, ng, routeentry( ni | nc | na | nb | ng | empty , 4 ) |  empty).
 op d9ninh : -> Dbentry .
 eq d9ninh =  dbentry( ni, nh, routeentry( ni | nc | nf | nd | nh | empty , 4 ) |  empty).
 op d9nini : -> Dbentry .
 eq d9nini = dbentry( ni, ni, routeentry( ni | empty, 0) |  empty ) .
endfm

fmod DBENTRY-M11 is
 pr DBENTRY . pr SOUP{Dbentry} .
 op routedb-m11 : -> Soup{Dbentry} .
 eq routedb-m11  = (
d11nana d11nanb d11nanc d11nand d11nane d11nanf d11nang d11nanh d11nani d11nanj d11nank
d11nbna d11nbnb d11nbnc d11nbnd d11nbne d11nbnf d11nbng d11nbnh d11nbni d11nbnj d11nbnk
d11ncna d11ncnb d11ncnc d11ncnd d11ncne d11ncnf d11ncng d11ncnh d11ncni d11ncnj d11ncnk
d11ndna d11ndnb d11ndnc d11ndnd d11ndne d11ndnf d11ndng d11ndnh d11ndni d11ndnj d11ndnk
d11nena d11nenb d11nenc d11nend d11nene d11nenf d11neng d11nenh d11neni d11nenj d11nenk
d11nfna d11nfnb d11nfnc d11nfnd d11nfne d11nfnf d11nfng d11nfnh d11nfni d11nfnj d11nfnk
d11ngna d11ngnb d11ngnc d11ngnd d11ngne d11ngnf d11ngng d11ngnh d11ngni d11ngnj d11ngnk
d11nhna d11nhnb d11nhnc d11nhnd d11nhne d11nhnf d11nhng d11nhnh d11nhni d11nhnj d11nhnk
d11nina d11ninb d11ninc d11nind d11nine d11ninf d11ning d11ninh d11nini d11ninj d11nink
d11njna d11njnb d11njnc d11njnd d11njne d11njnf d11njng d11njnh d11njni d11njnj d11njnk
d11nkna d11nknb d11nknc d11nknd d11nkne d11nknf d11nkng d11nknh d11nkni d11nknj d11nknk
  ) .
 op d11nana : -> Dbentry .
 eq d11nana = dbentry( na, na, routeentry( na | empty, 0) |  empty ) .
 op d11nanb : -> Dbentry .
 eq d11nanb =  dbentry( na, nb, routeentry( na | nb | empty , 1 ) |  empty ) .
 op d11nanc : -> Dbentry .
 eq d11nanc =  dbentry( na, nc, routeentry( na | nb | nc | empty , 2 ) |  empty ) .
 op d11nand : -> Dbentry .
 eq d11nand =  dbentry( na, nd, routeentry( na | nb | nd | empty , 2 ) |  empty ) .
 op d11nane : -> Dbentry .
 eq d11nane =  dbentry( na, ne, routeentry( na | nb | nc | ne | empty , 3 ) |  empty ) .
 op d11nanf : -> Dbentry .
 eq d11nanf =  dbentry( na, nf, routeentry( na | nb | nc | nf | empty , 3 ) |  empty ) .
 op d11nang : -> Dbentry .
```

```
eq d11nang =  dbentry( na, ng, routeentry( na | nb | ng | empty , 2 ) |  empty ) .
op d11nanh : -> Dbentry .
eq d11nanh =  dbentry( na, nh, routeentry( na | nb | nd | nh | empty , 3 ) |  empty ) .
op d11nani : -> Dbentry .
eq d11nani =  dbentry( na, ni, routeentry( na | nb | nc | ne | ni | empty , 4 ) |  empty).
op d11nanj : -> Dbentry .
eq d11nanj =  dbentry( na, nj, routeentry( na | nb | nd | nh | nj | empty , 4 ) |  empty).
op d11nank : -> Dbentry .
eq d11nank =  dbentry( na, nk, routeentry( na | nb | nd | nh | nj | nk | empty , 5 ) |
  empty ) .
op d11nbna : -> Dbentry .
eq d11nbna =  dbentry( nb, na, routeentry( nb | nc | na | empty , 2 ) |  empty ) .
op d11nbnb : -> Dbentry .
eq d11nbnb = dbentry( nb, nb, routeentry( nb | empty, 0) |  empty ) .
op d11nbnc : -> Dbentry .
eq d11nbnc =  dbentry( nb, nc, routeentry( nb | nc | empty , 1 ) |  empty ) .
op d11nbnd : -> Dbentry .
eq d11nbnd =  dbentry( nb, nd, routeentry( nb | nd | empty , 1 ) |  empty ) .
op d11nbne : -> Dbentry .
eq d11nbne =  dbentry( nb, ne, routeentry( nb | nc | ne | empty , 2 ) |  empty ) .
op d11nbnf : -> Dbentry .
eq d11nbnf =  dbentry( nb, nf, routeentry( nb | nc | nf | empty , 2 ) |  empty ) .
op d11nbng : -> Dbentry .
eq d11nbng =  dbentry( nb, ng, routeentry( nb | ng | empty , 1 ) |  empty ) .
op d11nbnh : -> Dbentry .
eq d11nbnh =  dbentry( nb, nh, routeentry( nb | nd | nh | empty , 2 ) |  empty ) .
op d11nbni : -> Dbentry .
eq d11nbni =  dbentry( nb, ni, routeentry( nb | nc | ne | ni | empty , 3 ) |  empty ) .
op d11nbnj : -> Dbentry .
eq d11nbnj =  dbentry( nb, nj, routeentry( nb | nd | nh | nj | empty , 3 ) |  empty ) .
op d11nbnk : -> Dbentry .
eq d11nbnk =  dbentry( nb, nk, routeentry( nb | nd | nh | nj | nk | empty , 4 ) |empty).
op d11ncna : -> Dbentry .
eq d11ncna =  dbentry( nc, na, routeentry( nc | na | empty , 1 ) |  empty ) .
op d11ncnb : -> Dbentry .
eq d11ncnb =  dbentry( nc, nb, routeentry( nc | na | nb | empty , 2 ) |  empty ) .
```

```
op d11ncnc : -> Dbentry .
eq d11ncnc = dbentry( nc, nc, routeentry( nc | empty, 0) |  empty ) .
op d11ncnd : -> Dbentry .
eq d11ncnd =  dbentry( nc, nd, routeentry( nc | nf | nd | empty , 2 ) |  empty ) .
op d11ncne : -> Dbentry .
eq d11ncne =  dbentry( nc, ne, routeentry( nc | ne | empty , 1 ) |  empty ) .
op d11ncnf : -> Dbentry .
eq d11ncnf =  dbentry( nc, nf, routeentry( nc | nf | empty , 1 ) |  empty ) .
op d11ncng : -> Dbentry .
eq d11ncng =  dbentry( nc, ng, routeentry( nc | na | nb | ng | empty , 3 ) |  empty ) .
op d11ncnh : -> Dbentry .
eq d11ncnh =  dbentry( nc, nh, routeentry( nc | nf | nd | nh | empty , 3 ) |  empty ) .
op d11ncni : -> Dbentry .
eq d11ncni =  dbentry( nc, ni, routeentry( nc | ne | ni | empty , 2 ) |  empty ) .
op d11ncnj : -> Dbentry .
eq d11ncnj =  dbentry( nc, nj, routeentry( nc | nf | nd | nh | nj | empty , 4 ) | empty).
op d11ncnk : -> Dbentry .
eq d11ncnk =  dbentry( nc, nk, routeentry( nc | nf | nd | nh | nj | nk | empty , 5 ) |
 empty ) .
op d11ndna : -> Dbentry .
eq d11ndna =  dbentry( nd, na, routeentry( nd | ne | ni | nc | na | empty , 4 ) | empty).
op d11ndnb : -> Dbentry .
eq d11ndnb =  dbentry( nd, nb, routeentry( nd | ne | nf | nb | empty , 3 ) |  empty ) .
op d11ndnc : -> Dbentry .
eq d11ndnc =  dbentry( nd, nc, routeentry( nd | ne | ni | nc | empty , 3 ) |  empty ) .
op d11ndnd : -> Dbentry .
eq d11ndnd = dbentry( nd, nd, routeentry( nd | empty, 0) |  empty ) .
op d11ndne : -> Dbentry .
eq d11ndne =  dbentry( nd, ne, routeentry( nd | ne | empty , 1 ) |  empty ) .
op d11ndnf : -> Dbentry .
eq d11ndnf =  dbentry( nd, nf, routeentry( nd | ne | nf | empty , 2 ) |  empty ) .
op d11ndng : -> Dbentry .
eq d11ndng =  dbentry( nd, ng, routeentry( nd | ne | nf | nb | ng | empty , 4 ) | empty).
op d11ndnh : -> Dbentry .
eq d11ndnh =  dbentry( nd, nh, routeentry( nd | nh | empty , 1 ) |  empty ) .
op d11ndni : -> Dbentry .
```

```
eq d11ndni =  dbentry( nd, ni, routeentry( nd | ne | ni | empty , 2 ) |  empty ) .
op d11ndnj : -> Dbentry .
eq d11ndnj =  dbentry( nd, nj, routeentry( nd | nh | nj | empty , 2 ) |  empty ) .
op d11ndnk : -> Dbentry .
eq d11ndnk =  dbentry( nd, nk, routeentry( nd | nh | nj | nk | empty , 3 ) |  empty ) .
op d11nena : -> Dbentry .
eq d11nena =  dbentry( ne, na, routeentry( ne | ni | nc | na | empty , 3 ) |  empty ) .
op d11nenb : -> Dbentry .
eq d11nenb =  dbentry( ne, nb, routeentry( ne | nf | nb | empty , 2 ) |  empty ) .
op d11nenc : -> Dbentry .
eq d11nenc =  dbentry( ne, nc, routeentry( ne | ni | nc | empty , 2 ) |  empty ) .
op d11nend : -> Dbentry .
eq d11nend =  dbentry( ne, nd, routeentry( ne | nf | nd | empty , 2 ) |  empty ) .
op d11nene : -> Dbentry .
eq d11nene = dbentry( ne, ne, routeentry( ne | empty, 0) |  empty ) .
op d11nenf : -> Dbentry .
eq d11nenf =  dbentry( ne, nf, routeentry( ne | nf | empty , 1 ) |  empty ) .
op d11neng : -> Dbentry .
eq d11neng =  dbentry( ne, ng, routeentry( ne | nf | nb | ng | empty , 3 ) |  empty ) .
op d11nenh : -> Dbentry .
eq d11nenh =  dbentry( ne, nh, routeentry( ne | nf | nd | nh | empty , 3 ) |  empty ) .
op d11neni : -> Dbentry .
eq d11neni =  dbentry( ne, ni, routeentry( ne | ni | empty , 1 ) |  empty ) .
op d11nenj : -> Dbentry .
eq d11nenj =  dbentry( ne, nj, routeentry( ne | nf | nd | nh | nj | empty , 4 ) |empty).
op d11nenk : -> Dbentry .
eq d11nenk =  dbentry( ne, nk, routeentry( ne | nf | nd | nh | nj | nk | empty , 5 ) |
 empty ) .
op d11nfna : -> Dbentry .
eq d11nfna =  dbentry( nf, na, routeentry( nf | nb | nc | na | empty , 3 ) |  empty ) .
op d11nfnb : -> Dbentry .
eq d11nfnb =  dbentry( nf, nb, routeentry( nf | nb | empty , 1 ) |  empty ) .
op d11nfnc : -> Dbentry .
eq d11nfnc =  dbentry( nf, nc, routeentry( nf | nb | nc | empty , 2 ) |  empty ) .
op d11nfnd : -> Dbentry .
eq d11nfnd =  dbentry( nf, nd, routeentry( nf | nd | empty , 1 ) |  empty ) .
```

```
op d11nfne : -> Dbentry .
eq d11nfne =  dbentry( nf, ne, routeentry( nf | nd | ne | empty , 2 ) |  empty ) .
op d11nfnf : -> Dbentry .
eq d11nfnf = dbentry( nf, nf, routeentry( nf | empty, 0) |  empty ) .
op d11nfng : -> Dbentry .
eq d11nfng =  dbentry( nf, ng, routeentry( nf | nb | ng | empty , 2 ) |  empty ) .
op d11nfnh : -> Dbentry .
eq d11nfnh =  dbentry( nf, nh, routeentry( nf | nd | nh | empty , 2 ) |  empty ) .
op d11nfni : -> Dbentry .
eq d11nfni =  dbentry( nf, ni, routeentry( nf | nd | ne | ni | empty , 3 ) |  empty ) .
op d11nfnj : -> Dbentry .
eq d11nfnj =  dbentry( nf, nj, routeentry( nf | nd | nh | nj | empty , 3 ) |  empty ) .
op d11nfnk : -> Dbentry .
eq d11nfnk =  dbentry( nf, nk, routeentry( nf | nd | nh | nj | nk | empty , 4 ) | empty).
op d11ngna : -> Dbentry .
eq d11ngna =  dbentry( ng, na, routeentry( ng | nd | ne | ni | nc | na | empty , 5 ) |
 empty ) .
op d11ngnb : -> Dbentry .
eq d11ngnb =  dbentry( ng, nb, routeentry( ng | nd | ne | nf | nb | empty , 4 ) | empty).
op d11ngnc : -> Dbentry .
eq d11ngnc =  dbentry( ng, nc, routeentry( ng | nd | ne | ni | nc | empty , 4 ) | empty).
op d11ngnd : -> Dbentry .
eq d11ngnd =  dbentry( ng, nd, routeentry( ng | nd | empty , 1 ) |  empty ) .
op d11ngne : -> Dbentry .
eq d11ngne =  dbentry( ng, ne, routeentry( ng | nd | ne | empty , 2 ) |  empty ) .
op d11ngnf : -> Dbentry .
eq d11ngnf =  dbentry( ng, nf, routeentry( ng | nd | ne | nf | empty , 3 ) |  empty ) .
op d11ngng : -> Dbentry .
eq d11ngng = dbentry( ng, ng, routeentry( ng | empty, 0) |  empty ) .
op d11ngnh : -> Dbentry .
eq d11ngnh =  dbentry( ng, nh, routeentry( ng | nd | nh | empty , 2 ) |  empty ) .
op d11ngni : -> Dbentry .
eq d11ngni =  dbentry( ng, ni, routeentry( ng | nd | ne | ni | empty , 3 ) |  empty ) .
op d11ngnj : -> Dbentry .
eq d11ngnj =  dbentry( ng, nj, routeentry( ng | nd | nh | nj | empty , 3 ) |  empty ) .
op d11ngnk : -> Dbentry .
```

```
eq d11ngnk =  dbentry( ng, nk, routeentry( ng | nd | nh | nj | nk | empty , 4 ) | empty).
op d11nhna : -> Dbentry .
eq d11nhna =  dbentry( nh, na, routeentry( nh | ne | ni | nc | na | empty , 4 ) | empty).
op d11nhnb : -> Dbentry .
eq d11nhnb =  dbentry( nh, nb, routeentry( nh | ne | nf | nb | empty , 3 ) |  empty ) .
op d11nhnc : -> Dbentry .
eq d11nhnc =  dbentry( nh, nc, routeentry( nh | ne | ni | nc | empty , 3 ) |  empty ) .
op d11nhnd : -> Dbentry .
eq d11nhnd =  dbentry( nh, nd, routeentry( nh | ne | nf | nd | empty , 3 ) |  empty ) .
op d11nhne : -> Dbentry .
eq d11nhne =  dbentry( nh, ne, routeentry( nh | ne | empty , 1 ) |  empty ) .
op d11nhnf : -> Dbentry .
eq d11nhnf =  dbentry( nh, nf, routeentry( nh | ne | nf | empty , 2 ) |  empty ) .
op d11nhng : -> Dbentry .
eq d11nhng =  dbentry( nh, ng, routeentry( nh | ne | nf | nb | ng | empty , 4 ) | empty).
op d11nhnh : -> Dbentry .
eq d11nhnh = dbentry( nh, nh, routeentry( nh | empty, 0) |  empty ) .
op d11nhni : -> Dbentry .
eq d11nhni =  dbentry( nh, ni, routeentry( nh | ne | ni | empty , 2 ) |  empty ) .
op d11nhnj : -> Dbentry .
eq d11nhnj =  dbentry( nh, nj, routeentry( nh | nj | empty , 1 ) |  empty ) .
op d11nhnk : -> Dbentry .
eq d11nhnk =  dbentry( nh, nk, routeentry( nh | nj | nk | empty , 2 ) |  empty ) .
op d11nina : -> Dbentry .
eq d11nina =  dbentry( ni, na, routeentry( ni | nc | na | empty , 2 ) |  empty ) .
op d11ninb : -> Dbentry .
eq d11ninb =  dbentry( ni, nb, routeentry( ni | nc | na | nb | empty , 3 ) |  empty ) .
op d11ninc : -> Dbentry .
eq d11ninc =  dbentry( ni, nc, routeentry( ni | nc | empty , 1 ) |  empty ) .
op d11nind : -> Dbentry .
eq d11nind =  dbentry( ni, nd, routeentry( ni | nc | nf | nd | empty , 3 ) |  empty ) .
op d11nine : -> Dbentry .
eq d11nine =  dbentry( ni, ne, routeentry( ni | nc | ne | empty , 2 ) |  empty ) .
op d11ninf : -> Dbentry .
eq d11ninf =  dbentry( ni, nf, routeentry( ni | nc | nf | empty , 2 ) |  empty ) .
op d11ning : -> Dbentry .
```

```
eq d11ning =  dbentry( ni, ng, routeentry( ni | nc | na | nb | ng | empty , 4 ) | empty).
op d11ninh : -> Dbentry .
eq d11ninh =  dbentry( ni, nh, routeentry( ni | nc | nf | nd | nh | empty , 4 ) | empty).
op d11nini : -> Dbentry .
eq d11nini = dbentry( ni, ni, routeentry( ni | empty, 0) |  empty ) .
op d11ninj : -> Dbentry .
eq d11ninj =  dbentry( ni, nj, routeentry( ni | nc | nf | nd | nh | nj | empty , 5 ) |
 empty ) .
op d11nink : -> Dbentry .
eq d11nink =  dbentry( ni, nk, routeentry( ni | nc | nf | nd | nh | nj | nk | empty,6 )
 |  empty ) .
op d11njna : -> Dbentry .
eq d11njna =  dbentry( nj, na, routeentry( nj | nh | ne | ni | nc | na | empty , 5 ) |
 empty ) .
op d11njnb : -> Dbentry .
eq d11njnb =  dbentry( nj, nb, routeentry( nj | nh | ne | nf | nb | empty , 4 ) | empty).
op d11njnc : -> Dbentry .
eq d11njnc =  dbentry( nj, nc, routeentry( nj | nh | ne | ni | nc | empty , 4 ) | empty).
op d11njnd : -> Dbentry .
eq d11njnd =  dbentry( nj, nd, routeentry( nj | nh | ne | nf | nd | empty , 4 ) | empty).
op d11njne : -> Dbentry .
eq d11njne =  dbentry( nj, ne, routeentry( nj | nh | ne | empty , 2 ) |  empty ) .
op d11njnf : -> Dbentry .
eq d11njnf =  dbentry( nj, nf, routeentry( nj | nh | ne | nf | empty , 3 ) |  empty ) .
op d11njng : -> Dbentry .
eq d11njng =  dbentry( nj, ng, routeentry( nj | nh | ne | nf | nb | ng | empty , 5 ) |
 empty ) .
op d11njnh : -> Dbentry .
eq d11njnh =  dbentry( nj, nh, routeentry( nj | nh | empty , 1 ) |  empty ) .
op d11njni : -> Dbentry .
eq d11njni =  dbentry( nj, ni, routeentry( nj | nh | ne | ni | empty , 3 ) |  empty ) .
op d11njnj : -> Dbentry .
eq d11njnj = dbentry( nj, nj, routeentry( nj | empty, 0) |  empty ) .
op d11njnk : -> Dbentry .
eq d11njnk =  dbentry( nj, nk, routeentry( nj | nk | empty , 1 ) |  empty ) .
op d11nkna : -> Dbentry .
```

```
 eq d11nkna =  dbentry( nk, na, routeentry( nk | nj | nh | ne | ni | nc | na | empty , 6 )
  |  empty ) .
 op d11nknb : -> Dbentry .
 eq d11nknb =  dbentry( nk, nb, routeentry( nk | nj | nh | ne | nf | nb | empty , 5 ) |
  empty ) .
 op d11nknc : -> Dbentry .
 eq d11nknc =  dbentry( nk, nc, routeentry( nk | nj | nh | ne | ni | nc | empty , 5 ) |
  empty ) .
 op d11nknd : -> Dbentry .
 eq d11nknd =  dbentry( nk, nd, routeentry( nk | nj | nh | ne | nf | nd | empty , 5 ) |
  empty ) .
 op d11nkne : -> Dbentry .
 eq d11nkne =  dbentry( nk, ne, routeentry( nk | nj | nh | ne | empty , 3 ) |  empty ) .
 op d11nknf : -> Dbentry .
 eq d11nknf =  dbentry( nk, nf, routeentry( nk | nj | nh | ne | nf | empty , 4 ) | empty).
 op d11nkng : -> Dbentry .
 eq d11nkng =  dbentry( nk, ng, routeentry( nk | nj | nh | ne | nf | nb | ng | empty , 6 )
 |  empty ) .
 op d11nknh : -> Dbentry .
 eq d11nknh =  dbentry( nk, nh, routeentry( nk | nj | nh | empty , 2 ) |  empty ) .
 op d11nkni : -> Dbentry .
 eq d11nkni =  dbentry( nk, ni, routeentry( nk | nj | nh | ne | ni | empty , 4 ) | empty).
 op d11nknj : -> Dbentry .
 eq d11nknj =  dbentry( nk, nj, routeentry( nk | nj | empty , 1 ) |  empty ) .
 op d11nknk : -> Dbentry .
 eq d11nknk = dbentry( nk, nk, routeentry( nk | empty, 0) |  empty ) .
endfm

***
*** Definision of the status and constat
***

fmod PSTAT is sort Pstat .
 ops pidle prequest pwait pride : -> Pstat [ctor] .
endfm
```

```
fmod PID is sort Pid . *** persion id
 pr NAT .
 op pidnone : -> Pid  [ctor] .
 op pid : Nat -> Pid  [ctor] .
endfm

fmod CID is sort Cid . *** car id
 pr NAT .
 op cidnone : -> Cid  [ctor] .
 op cid : Nat -> Cid  [ctor] .
endfm

***
*** Definision of functions
***

fmod DSTLIST is sort Dstlist .
 pr NODE .
 op empty : -> Dstlist [ctor] .
 op _ | _ : Node Dstlist -> Dstlist [ctor] .
 op nextdist : Dstlist Node -> Node .
 op top : Dstlist -> Node .
 vars N N1 : Node . vars DL DL1 DL2 : Dstlist .
 eq nextdist( empty , N ) = emptynode .
 eq nextdist( N | DL, N ) =  top(DL) .
 eq nextdist( N | DL , N1 ) = nextdist( DL , N1) [owise] .
 eq top( empty ) = emptynode .
 eq top( N | DL ) = N .
endfm

fmod PERSON is sort Person .
 pr NODE . pr PSTAT . pr CID . pr PID . pr DSTLIST .
 op person : Pid Pstat Node Node Cid Dstlist -> Person [ctor] .
 ***             state-of-person
 ***             src-node dst-node-of-request Cid-reserved-or-riding
 ***             list-of-destination-to-go-around
```

```
endfm

view Person  from TRIV to PERSON is
 sort Elt to Person .
endv

fmod CSTAT is sort Cstat .
 ops cidle creserved cservice : -> Cstat [ctor] .
endfm

fmod CAR is sort Car .
 pr NODE . pr CSTAT . pr CID . pr PID .
 op ecar : -> Car [ctor] . *** empty-car
 op car : Cid Cstat Node Node Pid  -> Car [ctor] .
 ***             current dst pid-for-reserved-or-service
 op cnode : Car -> Node .
 op _ isvacant : Car  -> Bool .
 vars N N1 N2 : Node . var C : Car . var CID : Cid . var PID : Pid .
 var CST : Cstat .
 eq cnode (car(CID,CST,N1,N2,PID)) = N1 .
 eq car(CID, cidle , N , N1 , PID ) isvacant = true .
 eq C isvacant = false [owise] .
endfm

view Car  from TRIV to CAR is
 sort Elt to Car .
endv

fmod TRAN is
 pr CID . pr PID .
 sort Tran .
 op notran : -> Tran [ctor] .
 op startreq : -> Tran [ctor] .
*** op book : Pid -> Tran [ctor] .
 op book : Cid -> Tran [ctor] .
 op rsvmove : Cid  -> Tran [ctor] .
```

```
 op ridesvc : Cid  -> Tran [ctor] .
 op svcmove : Cid  -> Tran [ctor] .
 op getoff  : Cid  -> Tran [ctor] .
 op getoff2 : Cid  -> Tran [ctor] .
 op startreq : Pid  -> Tran [ctor] .
 op rsvmove : -> Tran [ctor] .
 op ridesvc : Pid  -> Tran [ctor] .
 op svcmove : Pid  -> Tran [ctor] .
 op getoff  : Pid  -> Tran [ctor] .
 op getoff2 : Pid  -> Tran [ctor] .
endfm

***
*** Define the Ride-share system as Module
***

mod CRSV is
 pr CAR . pr SOUP{Car} .
 pr PERSON . pr SOUP{Person} .
 pr NODE . pr SOUP{Node} .
 pr DBENTRY . pr SOUP{Dbentry} .
 pr ROUTEENTRY . pr LIST{Routeentry} . pr DSTLIST .
 pr TRAN .
 sorts Crsv Sys .
 subsort Crsv < Sys .
 *** observable components
 op _ _ : Sys Sys -> Sys [ assoc comm] .
 op cars:_ : Soup{Car} -> Crsv .
 op ps:_ : Soup{Person} -> Crsv .
 op db:_ : Soup{Dbentry} -> Crsv .
 op tran:_ : Tran -> Crsv .
 *** functions
 op vcars : Soup{Car} -> Soup{Car} .
 op nearestvacant _ of _ on _ : Node Soup{Car} Soup{Dbentry} -> Soup{Car} .
 op dbmatch : Node Node Soup{Dbentry} -> Soup{Dbentry} .
 op getrtes : Soup{Dbentry} -> List{Routeentry} .
```

```
op ndbmatch : Soup{Node} Node Soup{Dbentry} -> Soup{Dbentry} .
op getrtes : Soup{Dbentry} -> List{Routeentry} .
op shortestroute : List{Routeentry} -> Routeentry .
op sortbydistance : List{Routeentry} -> List{Routeentry} .
op partition : Routeentry List{Routeentry} List{Routeentry} List{Routeentry} ->
List{Routeentry}  .
op if _ then {_} else {_} : Bool List{Routeentry} List{Routeentry} -> List{Routeentry} .
op nexthop : Node Node Soup{Dbentry} -> Node .
op cnodes : Soup{Car} -> Soup{Node} .

op carsonnode : Node Soup{Car} -> Soup{Car} .
op distance : Node Node Soup{Dbentry} -> Nat . *** for debug
vars C C1 : Car . var CS : Soup{Car} . var CID : Cid . var CST : Cstat .
vars P P1 : Person . var PS : Soup{Person} . var PID : Pid .
vars N N1 N2 N3 : Node . vars NS NS1 NS2 : List{Node} .  var NSO : Soup{Node} .
var RDB : Soup{Dbentry} . var DBE : Dbentry .
vars RE RE1 RE2 : Routeentry . vars RES RES1 RES2 : List{Routeentry} .
var DL : Dstlist . var T : Tran .
eq cnodes(empty) = empty .
eq cnodes((C CS)) = ( cnode(C)  cnodes(CS) ) .
eq vcars(empty) = empty .
ceq vcars((C CS)) = (C  vcars(CS)) if (C isvacant) .
eq vcars((C CS)) =  vcars(CS) [owise] .
eq shortestroute( RES ) = top( sortbydistance (RES ) ) .
eq sortbydistance(empty) = empty .
eq sortbydistance(RE | empty) = RE | empty .
eq sortbydistance(RE1 | RE2 | RES) = partition(RE1, RE2 | RES, empty, empty) .
eq partition(RE1, empty, RES1, RES2) = sortbydistance(RES1) @
                                       ( RE1 | sortbydistance( RES2)) .
eq partition(RE1, RE2 | RES, RES1, RES2) = if RE2 isshorter RE1 then
            { partition (RE1, RES, RE2 | RES1, RES2 ) } else
            { partition (RE1, RES, RES1, RE2 | RES2 ) } .
eq if true then {RES1} else {RES2} = RES1 .
eq if false then {RES1} else {RES2} = RES2 .
eq dbmatch(N1, N2, empty ) = empty .
ceq dbmatch(N1, N2, (DBE RDB)) = (DBE dbmatch(N1, N2, RDB))
```

```
      if (src(DBE) == N1 ) and (dst(DBE) == N2) .
 eq dbmatch(N1, N2, (DBE RDB)) = dbmatch(N1, N2, RDB) [owise] .
 eq ndbmatch((N empty), N2, RDB) = dbmatch(N, N2, RDB) .
 eq ndbmatch((N NSO), N2, RDB) = (dbmatch(N, N2, RDB) ndbmatch(NSO, N2, RDB)) .
 eq getrtes (dbentry(N1,N2,RES) empty) = RES .
 eq getrtes (dbentry(N1,N2,RES) RDB) = RES @ getrtes(RDB) .
 eq nearestvacant N of CS on RDB =  carsonnode(top(route(
      shortestroute(getrtes(ndbmatch(cnodes(vcars(CS)), N, RDB ) )))),CS) .
 eq nexthop(N, N1, RDB) =  top(tail(route(
      shortestroute(getrtes(dbmatch( N, N1, RDB)))))) .
 eq carsonnode(N,empty) = empty .
 eq carsonnode(N,(car(CID, CST, N, N1, PID) CS)) = (car(CID, CST, N, N1, PID)
                 carsonnode(N, CS)) .
 eq distance(N,N1,RDB) =  dist(shortestroute(getrtes(dbmatch(N,N1,RDB)))) .

***
***  Main rewriting rules
***


 crl [startreq] :
                 (ps: (person(PID, pidle, N1, N2, CID, DL) PS) )
                 (tran: T)
         =>
         (ps: (person(PID, prequest, N1, N2, CID, DL) PS) )
                 (tran: startreq)
                  if not(N1 == N2) .

 crl [book] :
             (ps: (person(PID, prequest, N, N1, cidnone, DL) PS))
             (cars: (car(CID, cidle, N2, N3, pidnone) CS)) (db: RDB )
             (tran: T)
          =>
     (ps: (person(PID, pwait, N, N1, CID, DL) PS))
             (cars: (car(CID, creserved, N2, N, PID) CS)) (db: RDB )
             (tran: book(CID))
```

```
            if car(CID, cidle, N2, N3, pidnone) \in
  (nearestvacant N of  (car(CID, cidle, N2, N3, pidnone) CS) on RDB) .

crl [rsvmove] :
              (cars: (car(CID, creserved, N, N1, PID ) CS) ) (db: RDB )
              (tran: T)
         =>
      (cars: (car(CID, creserved, nexthop(N, N1, RDB ), N1, PID) CS))
              (db: RDB ) (tran: rsvmove(CID) ) if not ( N == N1 ) .

rl [ridesvc] :
        (cars: (car(CID, creserved, N, N, PID) CS))
                (ps: (person(PID, pwait, N, N1, CID, DL) PS))
                (tran: T)
         =>
        (cars: (car(CID, cservice, N, N1, PID) CS))
                (ps: (person(PID, pride, N, N1, CID, DL) PS))
        (tran: ridesvc(CID)) .

crl [svcmove] :
                (cars: (car(CID, cservice, N, N1, PID) CS))
                (ps: (person(PID, pride, N, N1, CID, DL) PS)) (db: RDB)
                (tran: T)
         =>
                (cars: (car(CID, cservice, nexthop(N, N1, RDB ), N1, PID) CS))
                (ps: (person(PID, pride, nexthop(N, N1, RDB ), N1, CID, DL) PS))
                (db: RDB ) (tran: svcmove(CID)) if not ( N == N1 ) .

crl [getoff] :
              (cars: (car(CID, cservice, N, N, PID) CS))
              (ps: (person(PID, pride, N, N, CID, DL) PS))
              (tran: T)
         =>
              (cars: (car(CID, cidle, N, N, pidnone) CS))
              (ps: (person(PID, pidle, N, nextdist( DL, N), cidnone, DL) PS))
              (tran: getoff(CID))
```

```
               if not (nextdist( DL, N) == emptynode ) .

 crl [getoff2] :
               (cars: (car(CID, cservice, N, N, PID) CS))
               (ps: (person(PID, pride, N, N, CID, DL) PS))
               (tran: T)
          =>
       (cars: (car(CID, cidle, N, N, pidnone) CS))
               (ps: (person(PID, pidle, N, top(DL), cidnone, DL) PS))
               (tran: getoff2(CID))
               if (nextdist( DL, N) == emptynode ) .

endm

***
*** Initail status
***

mod CAR-INIT is
 pr CRSV . pr DBENTRY-M6 . pr DBENTRY-M6P .
 pr DBENTRY-M9 . pr DBENTRY-M11 .
 ops init inic2p2 inic2p3 inic2p4 inic2p5 : -> Sys .
 ops     im6c2p2 im6c2p3 im6c2p4 im6c2p5 : -> Sys .
 ops             im6c3p2 im6c4p2 im6c5p2 : -> Sys .
 ops     im6pc2p2 im6pc2p3 im6pc2p4 im6pc2p5 : -> Sys .
 ops              im6pc3p2 im6pc4p2 im6pc5p2 : -> Sys .
 ops     im9c2p2 im9c2p3 im9c2p4 im9c2p5 : -> Sys .
 ops             im9c3p2 im9c4p2 im9c5p2 : -> Sys .
 ops     im11c2p2 im11c2p3 im11c2p4 im11c2p5 : -> Sys .
 ops             im11c3p2 im11c4p2 im11c5p2 : -> Sys .
 ops car1 car2 car3 car4 car5 : -> Car .
 ops per1 per2 per3 per4 per5 : -> Person .
 ops ic2 ic3 ic4 ic5 : -> Soup{Car} .
 ops ip2 ip3 ip4 ip5 : -> Soup{Person} .
 var S : Sys .
```

```
eq car1 = car(cid(1), cidle, nd, nd, pidnone) .
eq car2 = car(cid(2), cidle, nc, nc, pidnone) .
eq car3 = car(cid(3), cidle, nb, nb, pidnone) .
eq car4 = car(cid(4), cidle, ne, ne, pidnone) .
eq car5 = car(cid(5), cidle, nf, nf, pidnone) .

eq per1 = person(pid(1), pidle, nf, na, cidnone,
      (na | nc | nf | empty)) .
eq per2 = person(pid(2), pidle, nb, nc, cidnone,
      (nc | nd | nb | empty)) .
eq per3 = person(pid(3), pidle, nd, ne, cidnone,
      (ne | nb | nd | empty)) .
eq per4 = person(pid(4), pidle, nc, nd, cidnone,
      (nd | ne | nc | empty)) .
eq per5 = person(pid(5), pidle, nd, nb, cidnone,
      (nb | ne | nd | empty)) .

eq ic2 = ( car1 car2 ) .
eq ic3 = ( car1 car2 car3 ) .
eq ic4 = ( car1 car2 car3 car4 )  .
eq ic5 = ( car1 car2 car3 car4 car5 ) .

eq ip2 = ( per1 per2 ) .
eq ip3 = ( per1 per2 per3 ) .
eq ip4 = ( per1 per2 per3 per4 ) .
eq ip5 = ( per1 per2 per3 per4 per5 ) .

eq init = (db: routedb-m6) (cars: ic2) (ps: ip2) (tran: notran) .
eq im6c2p2 = (db: routedb-m6) (cars: ic2) (ps: ip2) (tran: notran) .
eq im6c2p3 = (db: routedb-m6) (cars: ic2) (ps: ip3) (tran: notran) .
eq im6c2p4 = (db: routedb-m6) (cars: ic2) (ps: ip4) (tran: notran) .
eq im6c2p5 = (db: routedb-m6) (cars: ic2) (ps: ip5) (tran: notran) .

eq im6c3p2 = (db: routedb-m6) (cars: ic3) (ps: ip2) (tran: notran) .
eq im6c4p2 = (db: routedb-m6) (cars: ic4) (ps: ip2) (tran: notran) .
eq im6c5p2 = (db: routedb-m6) (cars: ic5) (ps: ip2) (tran: notran) .
```

```
 eq im6pc2p2 = (db: routedb-m6p) (cars: ic2) (ps: ip2) (tran: notran) .
 eq im6pc2p3 = (db: routedb-m6p) (cars: ic2) (ps: ip3) (tran: notran) .
 eq im6pc2p4 = (db: routedb-m6p) (cars: ic2) (ps: ip4) (tran: notran) .
 eq im6pc2p5 = (db: routedb-m6p) (cars: ic2) (ps: ip5) (tran: notran) .

 eq im6pc3p2 = (db: routedb-m6p) (cars: ic3) (ps: ip2) (tran: notran) .
 eq im6pc4p2 = (db: routedb-m6p) (cars: ic4) (ps: ip2) (tran: notran) .
 eq im6pc5p2 = (db: routedb-m6p) (cars: ic5) (ps: ip2) (tran: notran) .

 eq im9c2p2 = (db: routedb-m9) (cars: ic2) (ps: ip2) (tran: notran) .
 eq im9c2p3 = (db: routedb-m9) (cars: ic2) (ps: ip3) (tran: notran) .
 eq im9c2p4 = (db: routedb-m9) (cars: ic2) (ps: ip4) (tran: notran) .
 eq im9c2p5 = (db: routedb-m9) (cars: ic2) (ps: ip5) (tran: notran) .

 eq im9c3p2 = (db: routedb-m9) (cars: ic3) (ps: ip2) (tran: notran) .
 eq im9c4p2 = (db: routedb-m9) (cars: ic4) (ps: ip2) (tran: notran) .
 eq im9c5p2 = (db: routedb-m9) (cars: ic5) (ps: ip2) (tran: notran) .

 eq im11c2p2 = (db: routedb-m11) (cars: ic2) (ps: ip2) (tran: notran) .
 eq im11c2p3 = (db: routedb-m11) (cars: ic2) (ps: ip3) (tran: notran) .
 eq im11c2p4 = (db: routedb-m11) (cars: ic2) (ps: ip4) (tran: notran) .
 eq im11c2p5 = (db: routedb-m11) (cars: ic2) (ps: ip5) (tran: notran) .

 eq im11c3p2 = (db: routedb-m11) (cars: ic3) (ps: ip2) (tran: notran) .
 eq im11c4p2 = (db: routedb-m11) (cars: ic4) (ps: ip2) (tran: notran) .
 eq im11c5p2 = (db: routedb-m11) (cars: ic5) (ps: ip2) (tran: notran) .

endm

***
*** include model-chcker.maude
***
in model-checker .

***
```

```
*** define propositions such liveness properties as
***    Request Person will eventually Ride (RPweR)
***    Vacant Car will eventually Serve (VCweS)
***

mod CAR-PREDS is
 pr CAR-INIT . inc SATISFACTION .
 subsort State < Sys .
 ops inservice hasdst reserved : Cid -> Prop .
 ops isidle hasperson doublebook isvacant : Cid -> Prop .
 ops reserving riding havedst reachdst : Pid -> Prop .
 op enabled : Tran -> Prop .
 op applied : Tran -> Prop .
 var C : Car . var CID : Cid . var CST : Cstat .
 vars N N1 N2 N3 : Node . var PID : Pid .  var CS : Soup{Car} . var S : Sys .
 var PSTAT : Pstat . var PS : Soup{Person} .
 var CID2 : Cid . var PSTAT2 : Pstat . var PID2 : Pid .
 vars DL DL2 : Dstlist . var T : Tran .
 eq (ps: (person(PID, pidle, N1, N2, CID, DL) PS)) S
         |= enabled(startreq(PID)) = true .
 eq (ps: (person(PID, pidle, N1, N2, CID, DL) PS)) S
         |= enabled(startreq(PID)) = true .
 eq (ps: (person(PID, prequest, N, N1, cidnone, DL) PS))
              (cars: (car(CID, cidle, N2, N3, pidnone) CS)) S
         |= enabled(book(CID)) = true .
***      |= enabled(book(PID)) = true .
 eq (cars: (car(CID, creserved, N, N1, PID ) CS) ) S
         |= enabled(rsvmove(CID)) = true .
 eq (cars: (car(CID, creserved, N, N1, PID ) CS) ) S
         |= enabled(rsvmove) = true .
 eq (cars: (car(CID, creserved, N, N, PID) CS))
           (ps: (person(PID, pwait, N, N1, CID, DL) PS)) S
         |= enabled(ridesvc(CID)) = true .
 eq (cars: (car(CID, creserved, N, N, PID) CS))
           (ps: (person(PID, pwait, N, N1, CID, DL) PS)) S
         |= enabled(ridesvc(PID)) = true .
```

```
eq (cars: (car(CID, cservice, N, N1, PID) CS))
          (ps: (person(PID, pride, N, N1, CID, DL) PS)) S
        |= enabled(svcmove(CID)) = true .
eq (cars: (car(CID, cservice, N, N1, PID) CS))
          (ps: (person(PID, pride, N, N1, CID, DL) PS)) S
        |= enabled(svcmove(PID)) = true .
eq (cars: (car(CID, cservice, N, N, PID) CS))
          (ps: (person(PID, pride, N, N, CID, DL) PS)) S
        |= enabled(getoff(CID)) = true .
eq (cars: (car(CID, cservice, N, N, PID) CS))
          (ps: (person(PID, pride, N, N, CID, DL) PS)) S
        |= enabled(getoff(PID)) = true .
eq (cars: (car(CID, cservice, N, N, PID) CS))
          (ps: (person(PID, pride, N, N, CID, DL) PS)) S
        |= enabled(getoff2(CID)) = true .
eq (cars: (car(CID, cservice, N, N, PID) CS))
          (ps: (person(PID, pride, N, N, CID, DL) PS)) S
        |= enabled(getoff2(PID)) = true .

eq (tran: T) S |= applied(T) = true .

eq (cars: (car(CID, cservice, N, N1, PID) CS)) S |= inservice(CID) = true .
ceq (cars: (car(CID, CST, N, N1, PID) CS)) S |= hasdst(CID) = true
                  if not (N == N1) .
eq (cars: (car(CID, CST, N, N1, PID) CS)) S |= hasdst(CID) = false [owise] .
eq (cars: (car(CID, cidle, N, N1, PID) CS)) S |= isidle(CID) = true .
eq (cars: (car(CID, CST, N, N1, PID) CS)) S |= isidle(CID) = false [owise] .
ceq (cars: (car(CID, CST, N, N1, PID) CS)) S |= hasperson(CID) = true
                  if not (PID == pidnone) .
eq (cars: (car(CID, CST, N, N1, PID) CS)) S |= hasperson(CID) = false
           [owise] .
eq (cars: (car(CID, cidle , N , N1 , PID) CS)) S |= isvacant(CID) = true .
eq (cars: (car(CID, CST, N, N1, PID) CS)) S |= isvacant(CID) = false [owise] .
eq (cars: (car(CID, creserved , N , N1 , PID) CS)) S |= reserved(CID) = true .
eq (cars: (car(CID, CST, N, N1, PID) CS)) S |= reserved(CID) = false [owise] .
```

```
 eq (ps: (person(PID, prequest, N, N1, CID, DL) PS)) S |= reserving(PID) = true .
 eq (ps: (person(PID, PSTAT, N, N1, CID, DL) PS)) S |= reserving(PID) = false
                     [owise] .
 eq (ps: (person(PID, pride, N, N1, CID, DL) PS)) S |= riding(PID) = true .
 eq (ps: (person(PID, PSTAT, N, N1, CID, DL) PS)) S |= riding(PID) = false
                     [owise] .
 ceq (ps: (person(PID, PSTAT, N, N1, CID, DL) PS)) S |= havedst(PID) = true
                     if not (N == N1) .
 eq (ps: (person(PID, PSTAT, N, N1, CID, DL) PS)) S |= havedst(PID) = false
                     [owise] .
 ceq (ps: (person(PID, PSTAT, N, N1, CID, DL) PS)) S |= reachdst(PID) = false
                     if not (N == N1) .
 eq (ps: (person(PID, PSTAT, N, N1, CID, DL) PS)) S |= reachdst(PID) = true
                     [owise] .

 ceq  (ps: (person(PID, PSTAT, N, N1, CID, DL)
            person(PID2, PSTAT2, N2, N3, CID2, DL2) PS)) S
           |= doublebook(CID) = true  if (CID == CID2) .
 eq  (ps: (person(PID, PSTAT, N, N1, CID, DL)
            person(PID2, PSTAT2, N2, N3, CID2, DL2) PS)) S
           |= doublebook(CID) = false  [owise] .

endm

***
*** Module for model cheking (including fairness assumptions)
***

mod CAR-CHECK is
 pr CAR-PREDS . inc MODEL-CHECKER . inc LTL-SIMPLIFIER .
 ops svccarrun idlecarnoperson nodublebooking : Cid -> Formula .
 op vacantcarcanserve : Cid -> Formula .
 ops requestmancanride manmove : Pid -> Formula .
 op  sf : Tran -> Formula .
 op  wf : Tran -> Formula .
 op fair : Cid -> Formula .
```

```
 op fair : Pid -> Formula .
 ops fair1 fair1s fair2 fair3 : Cid -> Formula .
 ops wfair1 wfair2 wfair3 : Cid -> Formula .
 var CID : Cid . var PID : Pid . var T : Tran .
 eq sf(T) = ([] <> enabled(T)) -> ([] <> applied(T)) .
 eq wf(T) = (<> [] enabled(T)) -> ([] <> applied(T)) .

*** eq fair(CID) = wf(startreq) /\ wf(book(CID)) /\ wf(rsvmove(CID))
***                /\ wf(ridesvc(CID)) /\ wf(getoff(CID)) /\ wf(getoff2(CID)) .
 eq fair(CID) = wf(startreq) /\ sf(book(CID)) /\ wf(rsvmove(CID))
               /\ wf(svcmove(CID)) /\ wf(ridesvc(CID))
               /\ wf(getoff(CID)) /\ wf(getoff2(CID)) .
*** eq fair(PID) = wf(startreq(PID)) /\ wf(book(PID)) /\ wf(rsvmove)
***               /\ wf(ridesvc(PID)) /\ wf(getoff(PID)) /\ wf(getoff2(PID)) .
*** eq fair1(CID) = wf(startreq) /\ wf(book(CID)) .
*** eq fair1s(CID) = wf(startreq) /\ sf(book(CID)) .
 eq wfair1(CID) = isvacant(CID) |-> reserved(CID) .
 eq fair2(CID) = wf(rsvmove(CID)) /\ wf(ridesvc(CID)) /\
                 wf(svcmove(CID)) /\ wf(getoff(CID)) /\ wf(getoff2(CID)) .

 eq svccarrun(CID) = ([] (inservice(CID) /\ hasdst(CID))) . *** wrong
 eq idlecarnoperson(CID)  = ([] (isidle(CID) -> ~(hasperson(CID)))) .
 *** Request Person will eventually Ride(RPweR)
 eq requestmancanride(PID) = (reserving(PID) |-> riding(PID)) .
 eq nodublebooking(CID)  = ([] ~(doublebook(CID)) ) .
 *** Vacant Car will eventually Serve (VCweS)
 eq vacantcarcanserve(CID) = (isvacant(CID) |-> inservice(CID)) .
 eq manmove(PID) = (havedst(PID) |-> reachdst(PID)) .
endm

***
***  Experiment source code for model checking
***
***
*** search [1] in CAR-INIT : init =>* (cars: ( car(cid(1),creserved,nc,na,pidnone)
car(cid(2), cidle, nc, nc, pidnone) ) ) S .
```

```
*** search [1] in CAR-INIT : init =>* (ps: (person(pid(1), pidle, nf, nf,cidnone)
person(pid(2), pidle, na, na, cidnone)) ) S .

*** search [1] in CAR-CHECK : init =>* personreacha(pid(1),na) .

*** false
*** red in CAR-CHECK : modelCheck(init, svccarrun(cid(1))) .

*** true
*** red in CAR-CHECK : modelCheck(init, idlecarnoperson(cid(1))) .

*** false (without fair condition)
*** red in CAR-CHECK : modelCheck(init, (requestmancanride(pid(1)) /\
requestmancanride(pid(2)))) .

*** true
*** red in CAR-CHECK : modelCheck(ini1t, nodublebooking(cid(1))) .

*** might be look to be  true but false, because cid1,2 serve pid2 forever
*** rewrites: 1800369 in 2392180ms cpu (2441749ms real) (752 rewrites/second)
*** result [ModelCheckResult]: counterexample
*** red in CAR-CHECK : modelCheck(init, ( (fair(cid(1)) /\ fair(cid(2))) ->
 (requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .

*** might be look to be ture but falses
*** red in CAR-CHECK : modelCheck(init, ( (fair(cid(1)) /\ fair(cid(2))) ->
(vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2))))) .
*** should be true .
*** red in CAR-CHECK : modelCheck(init, ( (fair(pid(1)) /\ fair(pid(2))) ->
 (requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .

*** should be true
*** red in CAR-CHECK : modelCheck(init, ( (fair(pid(1)) /\ fair(pid(2)) /\
fair(cid(1)) /\ fair(cid(2))) ->
(requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .
```

```
***Maude> red in CAR-CHECK : modelCheck(init, ( (fair(pid(1)) /\ fair(pid(2))) ->
(requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .
***reduce in CAR-CHECK : modelCheck(init, fair(pid(1)) /\ fair(pid(2)) ->
***    requestmancanride(pid(1)) /\ requestmancanride(pid(2))) .
***rewrites: 18859417 in 2803008ms cpu (8169018ms real) (6728 rewrites/second)
***result Bool: true

*****************************************
***
*** for measurement of computation time
***
*** red in CAR-CHECK : modelCheck(init, manmove(pid(1))) .
***
*** unlimit -s unlimited is necesary
***(
red in CAR-CHECK : modelCheck(im6c2p2, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p3, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p4, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p5, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p2, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p3, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p4, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p5, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p2, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p3, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p4, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p5, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6c2p2, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p3, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p4, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im6c2p5, idlecarnoperson(cid(2))) .

red in CAR-CHECK : modelCheck(im6pc2p2, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p3, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p4, nodublebooking(cid(1))) .
```

```
red in CAR-CHECK : modelCheck(im6pc2p5, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p2, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p3, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p4, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p5, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p2, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p3, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p4, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p5, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im6pc2p2, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p3, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p4, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im6pc2p5, idlecarnoperson(cid(2))) .

red in CAR-CHECK : modelCheck(im9c2p2, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p3, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p4, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p5, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p2, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p3, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p4, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p5, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p2, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p3, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p4, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p5, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im9c2p2, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p3, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p4, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im9c2p5, idlecarnoperson(cid(2))) .

red in CAR-CHECK : modelCheck(im11c2p2, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p3, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p4, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p5, nodublebooking(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p2, nodublebooking(cid(2))) .
```

```
red in CAR-CHECK : modelCheck(im11c2p3, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im11c2p4, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im11c2p5, nodublebooking(cid(2))) .
red in CAR-CHECK : modelCheck(im11c2p2, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p3, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p4, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p5, idlecarnoperson(cid(1))) .
red in CAR-CHECK : modelCheck(im11c2p2, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im11c2p3, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im11c2p4, idlecarnoperson(cid(2))) .
red in CAR-CHECK : modelCheck(im11c2p5, idlecarnoperson(cid(2))) .

)

***
*** for measurement of computation time of liveness properties with fariness assumtions
***

***(
red in CAR-CHECK : modelCheck(im6c2p2, ( (fair(pid(1)) /\ fair(pid(2))) ->
(requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .
red in CAR-CHECK : modelCheck(im6pc2p2, ( (fair(pid(1)) /\ fair(pid(2))) ->
(requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .
red in CAR-CHECK : modelCheck(im6c2p3, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3)) )
-> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
requestmancanride(pid(3)) ) ) ) .
red in CAR-CHECK : modelCheck(im6pc2p3, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3)))
-> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
 requestmancanride(pid(3)) ) ) ) .
red in CAR-CHECK : modelCheck(im6c2p4, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3))
/\ fair(pid(4))) -> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
 requestmancanride(pid(3)) /\ requestmancanride(pid(4)) ) ) ) .
red in CAR-CHECK : modelCheck(im6c2p4, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3))
/\ fair(pid(4))) -> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
requestmancanride(pid(3)) /\ requestmancanride(pid(4)) ) ) ) .
```

```
red in CAR-CHECK : modelCheck(im9c2p2, ( (fair(pid(1)) /\ fair(pid(2))) ->
 (requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .
red in CAR-CHECK : modelCheck(im9c2p3, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3)) )
-> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
requestmancanride(pid(3)) ) ) ) .
red in CAR-CHECK : modelCheck(im9c2p4, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3))
/\ fair(pid(4))) -> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
 requestmancanride(pid(3)) /\ requestmancanride(pid(4)) ) ) ) .

red in CAR-CHECK : modelCheck(im11c2p2, ( (fair(pid(1)) /\ fair(pid(2))) ->
 (requestmancanride(pid(1)) /\ requestmancanride(pid(2))))) .
red in CAR-CHECK : modelCheck(im11c2p3, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3)))
-> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
requestmancanride(pid(3)) ) ) ) .
red in CAR-CHECK : modelCheck(im11c2p4, ( ( fair(pid(1)) /\ fair(pid(2)) /\ fair(pid(3))
 /\ fair(pid(4))) -> ( requestmancanride(pid(1)) /\ requestmancanride(pid(2)) /\
 requestmancanride(pid(3)) /\ requestmancanride(pid(4)) ) ) ) .

)
***(
red in CAR-CHECK : modelCheck(im6c2p2, ( (fair(cid(1)) /\ fair(cid(2))) ->
 (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2))))) .
red in CAR-CHECK : modelCheck(im6pc2p2, ( (fair(cid(1)) /\ fair(cid(2))) ->
 (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2))))) .
red in CAR-CHECK : modelCheck(im9c2p2, ( (fair(cid(1)) /\ fair(cid(2))) ->
 (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2))))) .
red in CAR-CHECK : modelCheck(im11c2p2, ( (fair(cid(1)) /\ fair(cid(2))) ->
 (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2))))) .

red in CAR-CHECK : modelCheck(im6c3p2, ( (fair(cid(1)) /\ fair(cid(2)) /\ fair(cid(3)))
-> (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2)) /\
 vacantcarcanserve(cid(3)) ))) .

red in CAR-CHECK : modelCheck(im6c4p2, ( (fair(cid(1)) /\ fair(cid(2)) /\ fair(cid(3))
/\ fair(cid(4)) ) -> (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2)) /\
vacantcarcanserve(cid(3)) /\ vacantcarcanserve(cid(4)) ))) .
```

```
red in CAR-CHECK : modelCheck(im6c5p2, ( (fair(cid(1)) /\ fair(cid(2)) /\ fair(cid(3))
/\ fair(cid(4)) /\ fair(cid(5)) ) -> (vacantcarcanserve(cid(1)) /\
vacantcarcanserve(cid(2)) /\ vacantcarcanserve(cid(3)) /\ vacantcarcanserve(cid(4)) /\
 vacantcarcanserve(cid(5)) ))) .
)

***(
*** false
red in CAR-CHECK : modelCheck(init, (fair1(cid(1)) -> wfair1(cid(1)))) .
red in CAR-CHECK : modelCheck(init, (fair1(cid(2)) -> wfair1(cid(2)))) .
*** true
red in CAR-CHECK : modelCheck(init, (fair1s(cid(1)) -> wfair1(cid(1)))) .
red in CAR-CHECK : modelCheck(init, (fair1s(cid(2)) -> wfair1(cid(2)))) .
red in CAR-CHECK : modelCheck(init, (fair2(cid(1)) -> wfair2(cid(1)))) .
red in CAR-CHECK : modelCheck(init, (fair2(cid(2)) -> wfair2(cid(2)))) .
*** false
red in CAR-CHECK : modelCheck(init, ((fair1(cid(1)) /\ fair1(cid(2))) -> (wfair1(cid(1))
 /\ wfair1(cid(2))))) .
*** true
red in CAR-CHECK : modelCheck(init, ((fair1s(cid(1)) /\ fair1s(cid(2))) ->(wfair1(cid(1))
 /\ wfair1(cid(2))))) .
*** true
red in CAR-CHECK : modelCheck(init, ((fair2(cid(1)) /\ fair2(cid(2))) -> (wfair2(cid(1))
/\ wfair2(cid(2))))) .
*** true
red in CAR-CHECK : modelCheck(init, ((wfair1(cid(1)) /\ wfair1(cid(2)) /\ fair2(cid(1))
/\ fair2(cid(2)) ) -> ( vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2)) ) )) .
*** true
red in CAR-CHECK : modelCheck(init, ((fair1s(cid(1)) /\ fair1s(cid(2)) /\ fair2(cid(1))
/\ fair2(cid(2)) ) -> ( vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2)) ) )) .

)
***(
red in CAR-CHECK : modelCheck(im6pc2p2, ( (wf(startreq) /\ sf(book(cid(1))) /\
sf(book(cid(2))) ) -> (wfair1(cid(1)) /\ wfair1(cid(2))) )) .
```

```
red in CAR-CHECK : modelCheck(im6pc2p2, (
  (wf(rsvmove(cid(1))) /\ wf(rsvmove(cid(2))) /\
   wf(ridesvc(cid(1))) /\ wf(ridesvc(cid(2))) /\
   wfair1(cid(1)) /\ wfair1(cid(2))) ->
 (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2)))) .

red in CAR-CHECK : modelCheck(im6pc3p2, ((wf(startreq) /\
  sf(book(cid(1))) /\ sf(book(cid(2))) /\ sf(book(cid(3))))
  ) -> (wfair1(cid(1)) /\ wfair1(cid(2)) /\ wfair1(cid(3))) ) .

red in CAR-CHECK : modelCheck(im6pc3p2, (
 (wf(rsvmove(cid(1))) /\ wf(rsvmove(cid(2))) /\ wf(rsvmove(cid(3))) /\
  wf(ridesvc(cid(1))) /\ wf(ridesvc(cid(2))) /\ wf(ridesvc(cid(3))) /\
  wfair1(cid(1)) /\ wfair1(cid(2)) /\ wfair1(cid(3)) ) ->
 (vacantcarcanserve(cid(1)) /\ vacantcarcanserve(cid(2)) /\
  vacantcarcanserve(cid(3)) ) )) .
quit .
)
```