Fachhochschule Dortmund
University of Applied Sciences and Arts
Embedded Systems Engineering

Fachhochschule Dortmund
University of Applied Sciences and Arts

Autonomous Transportation-on-Demand: Holistic Autonomous Transportation in Smart Cities

Master Thesis

Author: BRODO, Luca
Matriculation Number: 2780015

Supervisor: Henkler Stefan
Co-Supervisor: Name
Date: 00.2024

Abstract

Declaration

I hereby confirm, that I have written the Master Thesis at hand independently – in case of a group work: my respectively designated part of the work -, that I have not used any sources or materials other than those stated, and that I have highlighted any citations properly.

Contents

6   Part 2 - Graph Transformation Theory   61

7   Summary and Outlook   62

List of Figures

List of Tables

Chapter 1

Motivation

The economic impact of congestion in the United States is substantial, amounting to approximately \$121 billion per year or 1% of the country's GDP ([49]). This figure encompasses not only the staggering 5.5 billion hours lost to traffic congestion but also an additional 2.9 billion gallons of fuel burned. Moreover, these estimates overlook the potential costs associated with negative externalities such as vehicular emissions (including greenhouse gases and particulate matter) ([41]), travel-time uncertainty ([8]), and an elevated risk of accidents ([24]).

Chapter 2

Preliminaries

This chapter provides essential background information and definitions that form the foundation for the subsequent discussions in the document. Firsty, in Section 2.1, a plethora of useful general defitions are provided in order to facilitate the subsequent sections and, as a resul, the rest of the document as well. 2.2 introduces the concept of Model Predictive Control (MPC) as an advanced control strategy for dynamic systems, particularly focusing on the linear case. The linear MPC problem is formulated with a finite prediction horizon, system dynamics, and constraints on state and control variables. Additionally, the stability of MPC systems is briefly discussed, emphasizing the importance of demonstrating recursive feasibility and establishing stability in the sense of Lyapunov.

2.1 General Definitions

Definition 1: Positive Semidefinite A positive semidefinite matrix is a concept in linear algebra and matrix theory. A symmetric matrix A is said to be positive semidefinite if, for any non-zero column vector x, the following inequality holds:

$$x^T A x \geq 0 \tag{2.1}$$

Here, x^T represents the transpose of the vector x, and $x^T A x$ is the quadratic form associated with the matrix A. Another way to express positive semidefiniteness is in terms of eigenvalues. A symmetric matrix A is positive semidefinite if and only if all of its eigenvalues are non-negative.
Mathematically, $A \succeq 0$ indicates positive semidefiniteness.

   Definition 2: Positive Definite A positive definite matrix is a more specific case of a positive semidefinite matrix. A symmetric matrix A is said to be positive definite if, for any non-zero column vector x, the following inequality holds:

$$x^T A x > 0 \tag{2.2}$$

Here, x^T represents the transpose of the vector x, and $x^T A x$ is the quadratic form associated with the matrix A. Alternatively, positive definiteness in terms of eigenvalues is expressed as follows: A symmetric matrix A is positive definite if and only if all of its eigenvalues are strictly positive. In mathematical notation, $A \succ 0$ indicates positive definiteness.

Definition 3: Half-Space A half-space in n-dimensional space is defined by a linear inequality. The general form of a half-space is given by:

$$H = \{x \in \mathbb{R}^n \mid a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq b\} \tag{2.3}$$

Alternatively, in vector form:

$$H = \{x \in \mathbb{R}^n \mid a^T x \leq b\} \tag{2.4}$$

Here, $a_1, a_2, \ldots, a_n$ are real constants, and b is a real constant. The vector $(a_1, a_2, \ldots, a_n)$ is the normal vector to the hyperplane defining the boundary of the half-space. The inequality $a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq b$ represents the side of the hyperplane where the half-space lies. Geometrically, a half-space is one of the two regions divided by a hyperplane. If the inequality is strict ($<$ instead of $\leq$), the half-space does not include points on the hyperplane itself. If the inequality is non-strict ($\leq$), the half-space includes points on the hyperplane.
In two dimensions ($n = 2$), a half-space is a region divided by a straight line. In three dimensions ($n = 3$), it is a region divided by a plane, and so on. The intersection of multiple half-spaces forms a polyhedral region.

Definition 4: Polyhedral Region A polyhedral region is a geometric object in three-dimensional space that is defined as the intersection of a finite number of half-spaces. Formally, a polyhedral region P in three-dimensional space can be expressed as:

$$P = \{\mathbf{x} \in \mathbb{R}^3 \mid \mathbf{a}_i \cdot \mathbf{x} \leq b_i, \quad i = 1, 2, \ldots, n\}$$

where $\mathbf{x}$ is a three-dimensional vector representing a point in space, $\mathbf{a}_i$ are vectors normal to the defining planes, and b_i are constants determining the position of these planes. The inequalities $\mathbf{a}_i \cdot \mathbf{x} \leq b_i$ specify that the point $\mathbf{x}$ lies on or inside the half-space defined by the corresponding plane.

Definition 5: Recursive feasibility The MPC problem is deemed recursively feasible if, for all feasible initial states, assurance of feasibility is maintained at every state along the closed-loop trajectory.

Definition 6: Positively Invariant Set A set Ω is said to be a positively invariant set for a system $x(k+1) = f_\kappa(x(k))$, if

$$x(k) \in \Omega \implies x(k+1) \in \Omega \qquad \forall k \in \mathbb{N}$$

Definition 7: Comparision Functions

For $\mathbb{R}_0^+[0, \infty)$, the following comparison functions exist

$$\mathcal{K} := \left\{ \alpha : \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\middle|\, \alpha \text{ is continuous and strictly increasing with } \alpha(0) = 0 \right\} \tag{2.5}$$

$$\mathcal{K}_\infty := \left\{ \alpha : \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\middle|\, \alpha \in \mathcal{K} \text{ and unbounded} \right\} \tag{2.6}$$

$$\mathcal{KL} := \left\{ \beta : \mathbb{R}_0^+ \times \mathbb{R}_0^+ \to \mathbb{R}_0^+ \,\middle|\, \begin{array}{c} \beta \text{ continous} \\ \beta(\cdot, t) \in \mathcal{K} \quad \forall t \in \mathbb{R}_0^+ \\ \beta(r, \cdot) \text{ is strictly decreasing to } 0 \; \forall r \in \mathbb{R}_0^+ \end{array} \right\} \tag{2.7}$$

Definition 8: Lyapunov Function

Let $Y \subseteq X$ be a forward invariant set and $x^* \in X$. A function $V : Y \to \mathbb{R}^+$ is called a Lyapunov function for $x^+ = g(x)$ if the following two conditions hold for all $x \in Y$:

1. There exist $\alpha_1, \alpha_2 \in \mathcal{K}_\infty$ such that

$$\alpha_1(\|x - x^*\|) \leq V(x) \leq \alpha_2(\|x - x^*\|).$$

2. There exists $\alpha_V \in \mathcal{K}$ such that

$$V(x^+) \leq V(x) - \alpha_V(\|x - x^*\|).$$

Definition 9: Lyapunov Function.

Let $\mathcal{X}$ be a positively invariant set for a system $x(k+1) = f_\kappa(x(k))$ containing a neighbour of the origin in its interior. A function $V : \mathcal{X} \to R_+$ is called a Lyapunov function in $\mathcal{X}$ if for all $x \in \mathcal{X}$:

$$V(x) > 0 \quad \forall x \neq 0$$
$$V(x) = 0$$
$$V(x(k+1)) - V(x(k)) \leq -\alpha(x)$$

2.2 Model Predictive Control

Model Predictive Control (MPC) is an advanced control strategy used for dynamic systems. In the case of a linear time-invariant system, the mathematical description of MPC involves optimizing a cost function over a finite prediction horizon while subject to system dynamics and constraints.
For the purpose of this work, we will only consider the linear case of the MPC. For more details, please refer to [44].
Consider a linear system described by the following state-space equations:

$$x_{k+1} = Ax_k + Bu_k$$
$$y_k = Cx_k \tag{2.8}$$

where:

- x_k is the state vector at time k

- u_k is the control input (or control variable) at time k

- y_k is the output at time k

- A, B, and C are matrices that define the system dynamics.

The objective is to minimize a cost function J over a finite prediction horizon N. The cost function is typically defined as the sum of a quadratic performance index over the prediction horizon.
The choice of cost function is critical and highly depends on the final objective. For example, in case the objective is to track a reference signal, a typical cost function assumes the form described in Equation 2.9.

$$J = \sum_{i=0}^{N-1} \left[(y_{k+i} - r_{k+i})^T Q (y_{k+i} - r_{k+i}) + u_{k+i}^T R u_{k+i} \right] \tag{2.9}$$

where

- r_{k+i} is the reference trajectory at time $k + i$

- $Q \succeq 0$ is the weighting matrix for the state error

- $R \succ 0$ is the weighting matrix for the control input

Alternatively, another typical cost function has the followin form,

$$J = x_N^T P x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i \tag{2.10}$$

where

- x_i is the state vector at time i

- u_i is the control variable at time i

- $Q \succeq 0$ is the weighting matrix for the state error

- $P \succeq 0$ is the weighting matrix indicating the terminal cost

- $R \succ 0$ is the weighting matrix for the control input.

In Equation 2.10, the term $x_N^T P x_N$ is used to mitigate the fact that the MPC is minimizing over a limted time-horizon by ensuring that the system converges to a desired state by the end of the prediction horizon. While the infinite horizon formulation, i.e. $N = \infty$ and $P = 0$, possesses nice properties, such as its robustness and the perfect tradeoff between short- and long-term benefits of actions, it can not be computed in practice. Intuitively, since MPC is dealing with optimization problems under constraints, an infinite horizon will bring to having an infinte amount of variables. This is therefore the motivation behind why the horizon is usually shortened to N steps and the terminal cost is introduced.

The Linear MPC problem, therefore, is formulated as follows.

$$
\begin{aligned}
J^* = \textbf{min}\ & x_N^T P x_N + \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i \\
\textbf{s.t.}\ & x_{k+i+1} = A x_{k+i} + B u_{k+i} \quad \text{for } i = 0, 1, \dots, N-1 \\
& x_{k+i} \in \mathcal{X}, u_{k+i} \in \mathcal{U} \quad \text{for } i = 0, 1, \dots, N-1 \\
& x_{k+N} \in \mathcal{X}_f \\
& x_k = x(k)
\end{aligned}
\tag{2.11}
$$

where J^* is the global minimum of J and $\mathcal{X}, \mathcal{X}_f, \mathcal{U}$ are polyhedral regions. These are used to indicate the constraints over the state and control variables, which have the following form:

$$
\begin{aligned}
x_{\min} \leq x_{k+i} \leq x_{\max} \quad & \text{for } i = 0, 1, \dots, N \\
x_{\min|f} \leq x_{k+i} \leq x_{\max|f} \quad & \text{for } i = 0, 1, \dots, N \\
u_{\min} \leq u_{k+i} \leq u_{\max} \quad & \text{for } i = 0, 1, \dots, N-1
\end{aligned}
$$

Furthermore, the optimization problem is subject to system dynamics:

$$
x_{k+i+1} = A x_{k+i} + B u_{k+i} \quad \text{for } i = 0, 1, \dots, N-1
$$

The solution to this optimization problem provides the optimal control sequence $u_k^*, u_{k+1}^*, \dots, u_{k+N-1}^*$. The first element of this sequence, u_k^*, is then applied to the system, and the optimization problem is solved again at the next time step.

2.2.1 Stability of MPC systems

While there exist multiple ways to prove stability of MPC systems, this section will only focus on the one which is mostly relevant for this work, namely proving stability by leveraging the characteristics of the terminal set. More specifically, by defining $\mathcal{X}_f$ as convex set which includes the origin $(x(N) = 0)$, one can prove stability by insuring certain properties to be satisfied. These properties assume the cost function to have the following form.

$$J(x) = min_{x,u} V_f(X(N)) + \sum_{t=0}^{N-1} I(x(t), u(t)) \qquad (2.12)$$

where $I(x(t), u(t))$ is the stage cost and $V_f(X(N))$ is the terminal cost. Accordingly, these are the properties to satisfy.

1. The stage cost must be strictly positive and zero at the origin.

2. The terminal set is invariant under the local control law $\kappa_f(x)$. Namely

$$x(t+1) = Ax + B\kappa_f(x) \in \mathcal{X}_f \quad \forall x \in \mathcal{X}_f \qquad (2.13)$$

   given that $X_f \subseteq X$ and $\kappa_f(x) \in \mathcal{U}$

3. Establish stability by illustrating that the terminal cost function serves as a Lyapunov function in $\mathcal{X}_f$.

$$V_f(x(t+1)) - V_f(x) \leq -I(x, \kappa_f) \quad \forall x \in \mathcal{X}_f \qquad (2.14)$$

2.3 Graph Transformation Theory

Graph Transformation Theory (GTT),or graph rewriting, is a mathematical framework used to model and analyze the transformation of graphs. Graphs in this context represent structures composed of nodes and edges, and the transformations involve changing the structure of these graphs.

This section focuses on the fundamentals of this discipline and for more information, the reader is invited to consult the following resources: [23], [47], [29], [43] and [45]. The notions for this section have been collected while studying the aforenamed resources.

In order to introduce GTT, some concepts of category theory are necessary.

Definition 10: Category

A category $\mathcal{C}$ consists of the following data:

1. Objects : A group of mathematical objects, denoted as $\mathrm{Obj}(\mathcal{C})$

2. Morphisms (Arrows/Maps): For every pair of objects A and B, there is a set of morphisms denoted as $\mathrm{Hom}_{\mathcal{C}}(A, B)$. If $f \in \mathrm{Hom}_{\mathcal{C}}(A, B)$, we write $f : A \to B$. Morphisms must satisfy two properties:

- Associativity: For every triple of objects A, B, C and morphisms $f : A \rightarrow B$, $g : B \rightarrow C$, and $h : C \rightarrow D$, the composition $(h \circ g) \circ f$ is the same as $h \circ (g \circ f)$.

- Identity : For every object A, there exists an identity morphism $1_A : A \rightarrow A$ such that $f \circ 1_A = f$ and $1_B \circ f = f$ for any morphism $f : A \rightarrow B$.

3. Composition: For each pair of morphisms $f : A \rightarrow B$ and $g : B \rightarrow C$, there exists a composite morphism $g \circ f : A \rightarrow C$ in $\mathrm{Hom}_{\mathcal{C}}(A, C)$. Composition must be associative, i.e., $(h \circ g) \circ f = h \circ (g \circ f)$.

Example 1: Category of graphs A graph can be represented as the category having

1. Objects : tuple $G = (\mathcal{V}, \mathcal{E}, s, t)$ where

   - $\mathcal{V}$ is the set of nodes
   - $\mathcal{E}$ is the set of edges connecting the nodes
   - $s, t : \mathcal{E} \rightarrow \mathcal{V}$ are the source (target) of the edges, referred to as source (target) function

2. Morphisms: $f : G \rightarrow H$ must respect source and target functions, ie:

$$\forall e \in \mathcal{E}.f(s(e)) = s(f(e))$$
$$\forall e \in \mathcal{E}.f(t(e)) = t(f(e))$$

Definition 11: Product in a Category

Let $\mathcal{C}$ be a category. Let X_1 and X_2 be objects of $\mathcal{C}$. A product of X_1 and X_2 is an object X, typically denoted $X_1 \times X_2$, equipped with a pair of morphisms $\pi_1 : X \rightarrow X_1$ and $\pi_2 : X \rightarrow X_2$ satisfying the following universal property:

For every object Y and every pair of morphisms $f_1 : Y \rightarrow X_1$ and $f_2 : Y \rightarrow X_2$, there exists a unique morphism $f : Y \rightarrow X_1 \times X_2$ such that the following diagram commutes:

Definition 12: Coproduct in a Category

Let $\mathcal{C}$ be a category. Consider objects X_1 and X_2 in $\mathcal{C}$. A coproduct of X_1 and X_2 is an object X, often denoted as $X_1 \sqcup X_2$, equipped with a pair of morphisms $i_1 : X_1 \rightarrow X_1 \sqcup X_2$ and $i_2 : X_2 \rightarrow X_1 \sqcup X_2$ satisfying the following universal property:

For every object Y and every pair of morphisms $f_1 : X_1 \rightarrow Y$ and $f_2 : X_2 \rightarrow Y$, there exists a unique morphism $f : X_1 \sqcup X_2 \rightarrow Y$ making the following diagram commute:

14

$$X_1 \xrightarrow{i_1} X_1 \sqcup X_2 \xleftarrow{i_2} X_2$$

$$\downarrow f$$

with f_1 diagonal down-left and f_2 diagonal down-right to Y.

This means that any other morphism $X_1 \sqcup X_2 \to Y$ that respects i_1 and i_2 factors uniquely through the coproduct $X_1 \sqcup X_2$. In other words, the coproduct of X_1 and X_2 is an object $X_1 \sqcup X_2$ along with morphisms i_1 and i_2 such that, for any other object Y and morphisms f_1 and f_2, there exists a unique morphism f making the diagram commute. The coproduct captures the idea of combining objects X_1 and X_2 in a way that respects morphisms to another object Y.

Definition 13: Pushout

Given a category with three objects A, B, and C, and two morphisms (arrows) $f : A \to B$ and $g : A \to C$, the pushout of f and g is an object P along with two morphisms $i_B : B \to P$ and $i_C : C \to P$ such that the following diagram

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ g\downarrow & & \downarrow i_B \\ C & \xrightarrow{i_C} & P \end{array}$$

commutes and such that (P, i_1, i_2) is universal. That is, for any other such triple (Q, j_1, j_2) for which the following diagram commutes, there must exist a unique u : P $\to$ Q also making the diagram commute:

$$\begin{array}{ccc} A & \xrightarrow{f} & B \\ g\downarrow & & \downarrow i_B \\ C & \xrightarrow{i_C} & P \end{array}$$
with j_2 from B to Q, u from P to Q, and j_1 from C to Q.

In simpler terms, the pushout is a construction in category theory that combines two morphisms (arrows) $f : A \to B$ and $g : A \to C$ in a way that respects the existing structure of the category.

Given two objects B and C with morphisms f and g from a common object A, the pushout P is another object, along with two morphisms $i_B : B \to P$ and $i_C : C \to P$. These morphisms create a commutative diagram, meaning that following any path from A to P results in the same composition, regardless of the route taken.

Moreover, the pushout has a universal property, stating that for any other object Q with morphisms j_1 and j_2 forming a similar commutative diagram, there exists a unique morphism $u : P \to Q$ making the entire diagram commute.

In simpler terms, the pushout captures a way of amalgamating information from B and C in a manner that is compatible with the given morphisms, and it

does so in a universal way that any alternative attempt to amalgamate would factor uniquely through the pushout.

Definition 14: The Double Pushout Approach (DPO)
In the DPO, productions have the following form:

$$p : L \xleftarrow{\quad l \quad} K \xrightarrow{\quad r \quad} R$$

where:

- L (Left-hand side (LHS)) is a graph representing the initial pattern or structure to be recognized in the larger graph. This is the part of the graph that the production rule aims to match and replace.

- K (Interface graph) represents the region where the left-hand side L is embedded in the larger graph.

- R (Right-hand side (RHS)) is a graph representing the replacement or transformation that will be applied to the matched portion of the graph. After the application of the production rule, the matched portion of the graph (given by L and K) will be replaced by the structure defined in R.

- $l : K \to L$ is a morphism representing the embedding of the interface graph K into the left-hand side L.

- $r : K \to R$ is a morphism representing the embedding of the interface graph K into the right-hand side R.

The morphisms are usually injective, indicating that each element in the interface graph K has a unique counterpart in L and R, respectively. The overall interpretation of $p : L \leftarrow K \to R$ is that when the left-hand side L and its context K are found in a larger graph, they can be replaced by the right-hand side R according to the specified morphisms l and r.
This production is applied as follows.

$$
\begin{array}{ccccc}
L & \xleftarrow{\quad l \quad} & K & \xrightarrow{\quad r \quad} & R \\
\downarrow{\scriptstyle m} & & \downarrow{\scriptstyle d} & & \downarrow{\scriptstyle m'} \\
G & \xleftarrow{\quad l' \quad} & D & \xrightarrow{\quad r' \quad} & H
\end{array}
$$

So, when this production rule is applied to G, the subgraph D matching the left-hand side L is replaced by the right-hand side R, resulting in the transformed graph H. The morphisms m, d, and m' ensure that the embedding and replacement are consistent with the larger graphs. The commutativity of the diagram ensures that the rewriting process is well-defined and respects the structure of the graphs involved.

1. Matching and Interface: The interface graph K is matched within the original graph G using the matching morphism $d : K \to D$. This identifies where the LHS L is located in G.

2. LHS Embedding: The LHS L is embedded into the original graph G using the embedding morphism $m : L \to G$. This creates a graph that includes the LHS structure.

3. Application of Production Rule: The LHS L is then replaced by the RHS R, respecting the embedding morphisms l and r. This transformation is performed within the identified context given by the interface graph.

4. Resulting Graph: The resulting graph H is obtained after the replacement. The context graph D is embedded into H using $r' : D \to H$, and the RHS R is included in H using $m' : R \to H$.

Example 2: Let's define a graph that represents a simple directed network:

$$G = \{\text{Nodes: } A, B, C, D; \text{ Edges: } (A \to B), (B \to C), (C \to D)\} \qquad (2.15)$$

Let's define a DPO production rule to replace a chain of three nodes with a new node. The production rule is as follows:

$$p : L \leftarrow K \to R \qquad (2.16)$$

Where:

- L (LHS): Three consecutive nodes forming a chain - $A \to B \to C$.

- K (Interface graph): The interface graph is the same chain of nodes - $A \to B \to C$.

- R (RHS): A single node - X.

- $l : K \to L$ is the identity morphism since L and K are the same chain.

- $r : K \to R$ maps the chain of nodes to the single replacement node.

The production rule essentially says that if a chain of three nodes is found in a graph, it can be replaced with a single node. The interface graph K ensures that the replacement occurs in the correct context.

$$
\begin{aligned}
\text{Initial Graph:} &\quad G = \{A \to B \to C \to D\} \\
\text{Production Rule:} &\quad p : \underbrace{A \to B \to C}_{L} \to \underbrace{A \to B \to C}_{K} \to \underbrace{X}_{R} \\
\text{Resulting Graph:} &\quad H = \{X \to D\}
\end{aligned}
$$

After applying the production rule, the chain $A \to B \to C$ has been replaced by a single node X, resulting in the transformed graph H.

Definition 15: Dangling Condition
Let $G = (\mathcal{V}, \mathcal{E})$ be a graph with vertices $\mathcal{V}$ and edges $\mathcal{E}$. The dangling condition

imposes that, for each edge $e_i = (v_s, v_t) \in \mathcal{E}$, if e_i is not deleted, then both its source node v_s and target node v_t should be preserved. This condition is an essential condition in the DPO framework.

Definition 16: Identification Condition

Let $G = (\mathcal{V}_G, \mathcal{E}_G)$ be a graph and $H = (\mathcal{V}_H, \mathcal{E}_H)$ be a host graph. Suppose there exists a morphism $f : \mathcal{V}_G \to \mathcal{V}_H$ and $g : \mathcal{E}_G \to \mathcal{E}_H$ such that two nodes or edges in G, denoted as $v_1, v_2 \in \mathcal{V}_G$ or $e_1, e_2 \in \mathcal{E}_G$, are matched into a single node or edge in H (i.e., $f(v_1) = f(v_2)$ or $g(e_1) = g(e_2)$ via a non-injective morphism), the identification condition imposes that those should be preserved. This condition is an essential condition in the DPO framework.

Chapter 3

Literature Review

This chapter will center on an exploration of various pieces of literature and related works that have been thoroughly reviewed, aligning with the overarching purpose of this study.

To facilitate the reader, this chapter has been divided into multiple subsections, which follow the structure of this work.

3.1   Fleet and Transportation Network

Most of the literature reviewed during the development of this study focused on (Autonomous) Mobility-on-Demand, therefore only considering human mobility.In this case, *Frazzoli et al.* in [61] indentify three main mathematical models used to represent transportation networks: graph-theoretic, queuing theory and continuum models.

Graph-theoretic approach

In graph-theoretic models, the transportation network is modeled as a directed graph $G\langle V, R \rangle$, where V is the set of vertices and $E \subseteq V \times V$ is the set of edges. Usually, each node $v \in V$ represents a location, or a station. In [6], in a more abstract way, nodes are used to represent areas of interest. Each edge $\langle v_1, v_2 \rangle \in E$ rapresents a connection, which could be a road or a collection of those, from v_1 to v_2. Furthermore, each link is usually associated with a metric, such as distance $d : E \rightarrow R_{\geq 0}$. Many studies adopt a dynamic fluid approach, representing AVs and customers not as individual entities but rather as flows moving between nodes. In this framework, models that do not vary with time often simplify to static network flow problems. Alternatively, graph-theoretic models can be integrated with vehicle-centric models, where both AVs and customers are modeled individually.

Queuing-theoretic approach

Queuing theory is concerned with the examination of waiting lines. In the context of AMoD systems, trips are conceptualized as queues between locations, enabling the analysis of various characteristics AVs, closely linked to waiting times. Formally, considering N stations situated at specific geographical locations with m AVs providing mobility services. Customers arrive at each station based on an exogenous stochastic process (usually a Poisson process with rate λ, like for example in [64]) and select destinations with certain probabilities. If AVs are accessible at a station, customers proceed to their respective destinations; otherwise, they exit the system (referred to as passenger loss). Travel times between stations are also stochastic and are typically modeled as exponentially distributed random variables. When formulating policies for AMoD systems, the focus lies in analyzing and regulating the movement of AVs from one queue to another. These models are commonly employed in time-invariant settings. The scenario of infinitely large fleets has garnered particular attention for deriving theoretical insights.

Battery model

This chapter delves into the exploration of various sources and solutions that have been examined in the development of the battery model utilized in this study. As the first work considered, *Montoya et al.* in [36] employed linear piece-wise approximations to approximate the nonlinear charging function, reporting an error of approximately 1%. Additionally, they introduced the concept of different types of charging stations. *Froger et al.* in [15] and *Kancharla et al.* in [28] also employed piece-wise linear functions for their respective charging regimes. Conversely, *Nie et al.* in [39] did not consider the possibilities for vehicle recharging in their model.

In terms of modeling the battery charging profile, diverse approaches have been proposed. *Han et al.* in [22] developed a model based on the internal resistance and voltage of the battery; however, this model was disregarded in favor of others that better captured the battery charging profile based on more relevant factors. *Yu et al.* in [59] proposed a sophisticated model for charging constraints in general. Although their approach differs from the one presented in this work, as it incorporates charging during routing, this aspect was excluded in favor of an alternative approach. *Lee et al.* in [32] considered a simplified battery model that provides states of charge (SoC) as a function of the charging current. The author asserted the general applicability of their approach to every charging profile, provided SoC(t) is a concave and non-decreasing function with SoC(0) = 0, and an inverse function $\text{SoC}^{-1}(a)$ exists for $0 < a < Q$.

3.2 AToD Challanges

AV Dispatching

For the vehicle dispatching problem, multiple solutions have been already proposed. For example, *Vasco et al.* in [1] model it as a linear minimum cost multicommodity flow problem; *Holzapfel et al.* in [25] model it as a MIP problem. Others, on the other hand, make use of other heuristics. For example, *Levin et al.* in [33] and *Mora et al.* in [3] use nearest neighbors. Recently, to improve real-time capabilites, some approaches have been proposed which make use of deep learning, like for e.g. the one proposed by *Yu et al.* in [60].

AV Routing

On a basic level, the routing problem can be reconducted to the Vehicle Routing Problem (VRP) (a thorough analysis and explanation can be found in [53]). Usually, the VRP is solved and analyzed as a static problem, which implies that requests are known in advance. As a result, origins and destination of each trip is also known a priori. As pointed out by *Frazzoli et al.* in [61], in AToD systems requests are dynamic, meaning they are not known in advance. More specifically, the task of managing an AToD system can be viewed as a specific case of the dynamic one-to-one pickup and delivery problem. As pointed out by *Zhang* in [63],to provide a more detailed characterization of these systems, several additional attributes and constraints must be considered:

- Immediate Service Requests: Requests are made for immediate service rather than being scheduled in advance with specified time windows.

- Stochastic Customer Arrivals and Travel Times: The timing of customer requests and the duration of travel are subject to stochastic variability.

- Multi-Occupancy Vehicles: Contrary to [63], vehicles must be considered to have a capacity, rather than being single-occupancy. This is motivated by the fact that recently developed AV are equipped with more than one seat (see for e.g. [17])

Wollenstein-Betech et al. in [57] propose a solution to the routing-rebalancing problem in mixed traffic situations. They propose an interesting two-graph solutions, i.e. one graph for the AToD system and one for the pedestrians which are interconnected. Furthermore, they also consider private and public vehicles. *Pavone et al.* in [42] also propose a solution for the rebalancing problem. Although they mainly consider MoD systems, their approach can be easily applied to AToD systems. Their solution, moreover, focuses only on the rebalancing issue, which is studied using a steady-state fluid model, as pointed out in [57]. Furthermore, this work does not take into consideration congestions in their model.
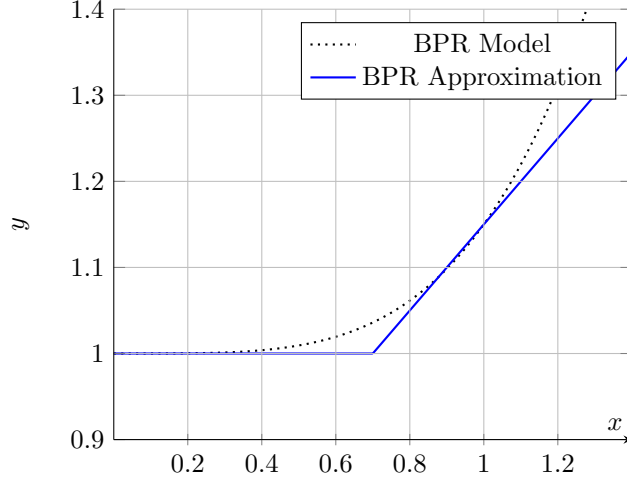
Figure 3.1: BPR Model and its Approximation as a piecewise linear function. In this example, $x_{th} = 0.7$, $x_{max} = 1.4$, $a = 1$, $b = 0.5$

AMoD Rebalancing

In the context of network flow models, rebalancing is usually formulated as a (integer) optimization problems, such as in the works of *Zgraggen et al.* in [62], *Carron et al.* in [9] and of *Smith et al.* in [51]. Crucially, in real-time texting, the rebalancing problem is solved leveraging the model-predictive control framework, such as in the works mentioned above.
Wollenstein-Betech et al. in [58] investigate effective pricing and rebalancing strategies for AMoD systems from a macroscopic planning standpoint. It aims to optimize profits while ensuring system load balance. Using a dynamic fluid model, the study demonstrates equilibrium attainment via pricing policies and develops an optimization framework for determining optimal pricing and rebalancing approaches. They model customers and vehicles located in regions using a queuing approach, i.e. two queues per region representing vehicles and clients. *Salazar et al.* in [48] study the rebalancing problem while considering two important aspects, namely mixed traffic and congestions. The first is treated by modeling the road network and the walking network as two separate entites, which combined make up a supernetwork defining the entire transportation network. The latter are treated according to the Bureau of Public Roads (BPR) model ([40]) , more specifically by approximating their model linearly. The BPR model of congestions can be expressed by Equation 3.1 and is visualized in Fig 3.1 as a black dotted line.

$$f_{BPR}(x) = 1 + 0.15x^4 \qquad (3.1)$$

Since this model would lead to an unconvex, i.e. intractable, problem, the authors propose a piece wise approximation of it, which take the form expressed

22

in Equation 3.2

$$y = \begin{cases} a & \text{if } x \in [0, x_{th}] \\ a + b \cdot (x - x_{th}) & \text{if } x \in [x_{th}, x_{max}] \end{cases} \quad (3.2)$$

where a and b are used to model the height of the horizontal line and the slope of the second line, x_{th} is the threshold of the piecewise approximation and x_{max} defines the approximation window. This approximation can be seen in Fig 3.1 as a blue line.

Equation 3.2 is then used to calculate the traverse time for each edge. While this is a more representative congestion model than the one proposed in this paper in Section 4.1, one might argue that the two serve two different purposes. While the one proposed in [48] serves to express the travel time in terms of the amount of vehicles currently traveling on the node, the one proposed in this work is used to simply limit the amount of vehicles traveling on the link to avoid situations of stop-and-go traffic. Furthermore, it is worth mentionining that this model can be easily integrated into the model by modifying the definition of the travel time function.

In the abovementioned study, the energy consumption of the AVs is also modeled differently compared to this work. Within their network graph G, the energy consumption of each AV with efficiency η traveling through arc (i, j) with distance d_{ij} is expressed as:

$$e_{ij} = (\frac{\rho_a}{2} \cdot A_f \cdot c_d \cdot v_{ij} + c_r \cdot m_v \cdot g) \cdot \frac{d_{ij}}{\eta_{\text{AV}}} \quad \forall (i, j) \in G \quad (3.3)$$

where the aerodynamic drag is determined by the air density ρ_a, the frontal area A_f, the drag coefficient c_d and the moving velocity v_{ij}. The friction of the wheels on the road is determined by the rolling friction coefficient c_r, the mass of the vehicle m_v, and the gravitational acceleration g.

This is a more vehicle-based approach compared to the one adopted in this work, which only considers the battery dischargment, as it takes into consideration various other aspects such as drag and rolling friction. For the purpose of this work, however, it has been chosen to neglect those factors, which could be later introduced into the battery-dischargment model, to favour a more simplistic approach. Furthermore, given that in the aforementioned paper is assumed to have constant velocity $v_{ij} = \dfrac{d_{ij}}{t_{ij}}$, one can trivially adopt a similar solution for the model in this paper by making the same assumption.

Wallar et al. in [56] tackle the rebalancing problem independently from any other AToD challenge, while also considering ride-sharing possibilities. The main idea of the propsed approach is to assign free vehicles to regions, which are computed offline, according to the estimations of traveling request per region. In this case, the requests are estimated using the particle filter. Furthermore, the division in region for the graph is formulated as an integer linear programming problem using a rechebaility matrix R, which indicates whether a station j if

recheable from i within a maximum time t_{max} ($R_{ij} = 1$ else $R_{ij} = 0$). Cleverly, the authors also took into consideration the fact that, although a vehicle is free, it requires some time to reach the assigned rebalancing region. In other words, if a vehicle requires eight minutes to reach a certain region, considering a time horizon of ten minutes, it will be available only for two minutes, i.e. 20% of the time. This is used to limit the oversaturation of vehicles in the respective rebalancing regions.

3.3 MPC for AToDs

3.4 Applications of Graph Transformation Systems

Due to the flexible nature of graph and their intuitive understanding, Graph transformation systems (GTS) are widely use for design exploration. *Voss et al.* in [55] propose a study whre they demonstrate the flexibility of this method in various engineering disciplines. For example, *Zhao et al.* in [65] utilize this approach to explore the design space of a terrain-optimized robot. In the context of mechatronic systems, GTS can be used for the development of components, such as the gear box design proposed in *Fürst et al.* in [16], and of entire systems, as demonstrated by *Gross et al.* in their series of three papers for the development of a satellite ([18], [19], [20]).

In the context of road and traffic management, *Beck et al.* in [5] propose a method to optimize the performance of vehicle routing over a graph netowrk based on GTS. While graph transformations are not used to find a direct solution to the VRP, those are used to transform the graph in such a way that the routing algorithms' performance is optimized.

Raadsen et al. in [43]introduce a category theoretical approach in formalizing transport network transformations by adopting and adapting pattern graph transformation techniques. This study introduces two novel spatial aggregation methods applicable to a class of traffic assignment models, employing a category theoretical approach. The formalization technique, originally developed for quantum physical processes, offers an intuitive graphical representation with a rigorous mathematical foundation. The method shares similarities with regular expressions and functional programming, providing insights for constructing solvers or algorithms. The proposed aggregation methods are compatible with traffic assignment procedures, decomposing the network into a constant free-flowing part and a smaller demand-varying delay part, aiming to reduce computational costs without sacrificing accuracy, as demonstrated in a large-scale case study.

Furthermore, another GTS application in the context of traffic management can be found in *Lara et al.* in [11]. This study introduces a domain-specific visual language, named Traffic, designed for the domain of traffic networks. The syntax of the language is formally defined using meta-modelling. For semantics, the

study employs two approaches: the first involves graph transformation for operational semantics, while the second incorporates timing information and establishes a denotational semantics using Timed Transition Petri Nets (TTPN). The transformation from the Traffic formalism to TTPN is defined through graph transformation. Both semantic approaches are implemented and analyzed using the AToM tool, showcasing their applicability through examples.

OThers

Fehn et al. in [14] modeled a fleet according to energy prices.

Chapter 4

Time-Invariant Model of AToD

This chapter focuses on tackling dispatching, routing and rebalancing problems for an AToD system. First, a time-invariant vehicle-centric model for the transportation network is developed in Section 4.1. The model presented in this document expands upon the one formulated in [6], using graph theory and a vehicle-centric approach to provide flexibility in handling different vehicle capacities and types. Additionally, this model addresses the combination of mobility of people with goods transportation, allowing for a more comprehensive analysis of transportation networks. This model be the basis for the analysis and formulation of the aforementioned problems in Section 4.2.

4.1 Vehicle-centric Model

The model provided in this section will expand the model formulated in [6]. It is worth noting that, although the model presented in [6] is specific for the scenario described in previous sections of the same work, it can be expanded to cover more general use cases.

It follows that the transportation network will be modeled using graph theory. Most of the literature makes use of (Eulerian) fluid-dynamic models ([61]), i.e. customers and AVs are represented as (non-integer) flow between nodes, in this work, this model will be used in addition to a vehicle-centric approach, where customers and AVs will be modeled individually. Contrary to the solutions proposed by most of the literature, this work combines mobility of people with goods transportation, therefore modeling vehicles individually will provided the required flexibility to handle different capacities and the increased complexity given by the plethora and variety of vehicles, as we will see later in this section. The set of vehicles will be refered to as $\mathcal{A}$.

Let $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ being the directed graph representing corresponding transportation network of the city in question, where $\mathcal{V}$ is the set of vertices and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$

is the set of edges. Any vertex, or node, $v \in \mathcal{V}$ represents a location. Following the notation in [6], each node consists of an area of interests. For the context of this paper, the two terms will be used intercheangeably, as they ultimately indicate the same thing. An edge $\langle v_i, v_j \rangle \in \mathcal{E}$ represents a connection, which can consist of a road or a combination of those, linking v_i to v_j. The level of abstraction is decided by the engineer and can be used to regulate the granularity of the model. Locations v_i and v_j can indicate low level key points, such as cross roads or trafficated traffic lights, as well as higher level areas such as residental areas or leisure centers. Similarly, an edge can contain multiple crossroads or joints or simply indicating a path from v_i to v_j, regardless of additional details. A correct granularity is not trivial to decide a priori and a general approach is likewise hard to delineate. Determining the level of abstraction remains, therefore, an engineer's prerogative and highly depends on the system in question. It is also worth noting that the, for conveniency and to better reflect the situation in real world applications, the graph is not a bidirect graph. To model a two-way road, one should simply make use of two edges from locations v_i and v_j.

As also described in [6], each edge is associated with multiple metrics and information.

Firstly, at each edge one must attribute a travel time T. T is a function $T : \mathcal{E} \times \mathcal{A} \to \mathbb{R}_{>0}$, which at each given a vehicle and edge maps a float value $T_{i,j}^a \in \mathbb{R}_{\geq 0}$ indicating the time required for the vehicle of type a to travel the path from v_i to v_j. In this work, the unit of $T_{i,j}^a$ is of particular interest, as it depends on the application.

Secondly, each edge is also associated with a distance $d : \mathcal{E} \to \mathbb{R}_{\geq 0}$, which maps an edge to a float value $d_{i,j}$ indicating the distance from v_i to v_j.

Another factor which is considered in this work, as stated above, is pollution. This metric is associated to each edge and is a function $f : \mathcal{E} \times \mathcal{A} \to \mathbb{R}_{\geq 0}$. To simplify the discussion for this work, f is assumed to produce a certain value refered to as pollution index. In other words, it is not supposed to represent a measurable element, such as CO_2 emissions, but rather a value which can represent multiple quantitative factors and is higher if the combination of path and vehicle type is highly polluting. While it can be argued that the pollution can also be simply a function of d and T, this abstraction does not account for other elements, such as road type, vehicle fuel and road slope. The index f, therefore, is a mathematical abstraction which allows a more flexible model for the system. To use this abstraction, however, each edge must include additional information, including the one mentioned before.

Similarly to what observed by *Zhang et al.* in [46], we can introduce the concept of congestions in the model by constraining the routing by the capacity of each road. In other words, we can associate each edge with a capacity $c : \mathcal{E} \to \mathbb{N}_{>0}$ indicating the limit in terms of car occupancy above which the traffic in that edge slows down eventually reaching a congestion. As pointed out in [46], this semplified model is adequate in this context as well as the aim of this work focusing solely on the control of vehicles to prevent congestion rather than analyzing their behavior in congested network. While it is understood that congestions behave

differently in real world scenarios and other, more sophisticated approaches exist ([34], [54]), for the sake of simplicity, we will assume $T_{ija} = \infty$ if the number of vehicles in that edge from $\langle v_i, v_j \rangle$ to be larger than the capacity c_{ij}.

In addition, each edge $\langle v_i, v_j \rangle$ will also include information regarding traffic limitations, i.e. whether a vehicle type is allowed or not to travel that link. Intuitively, limitations will be represented as a function $s : \mathcal{E} \times \mathcal{A} \to \{0, 1\}$, where 1 implies the vehicle can traver that edge. Following the example in [6], limitations can be because of various factors, like for e.g. weight or height. For this reason, it is convenient to abstract away such details and just indicate wether a link is can be traversed or not. Notably, one could use this representation to transform a bidirect graph G' to be equivalent to G by letting $s(e, a) = 0$ for any one-way road $e = \langle v_i, v_j \rangle$ for all $a \in \mathcal{A}$. For convenience, this function will be incorporated in the definition of the capacity, updating it to become Equation 4.1.

$$c(e = \langle v_i, v_j \rangle) = \begin{cases} 0 & \text{if } s(e) = 0 \\ c(e) & \text{otherwise} \end{cases} \tag{4.1}$$

Following the discussion above, this is equivalent to setting $T_{ija} = \infty$ for all $a \in \mathcal{A}$ according to this model specifications. In this way, we can treat the road as being unaccessible without increasing the number of conditions and decrease the redeability of the model.

As mentioned above, the set of autonomous vehicles is indicated as $\mathcal{A}$ and each vehicle will be modeled as a tuple $\langle \underline{s_a}, \bar{t_a}, S_a, Q_a, I_a^b, R_a^-, R_a^+, \theta_a, B_a(t), \mathcal{R}_a, \mathcal{T}_a \rangle$. $\underline{s_a} \in \mathcal{V}$ and $\bar{t_a} \in \mathcal{V}$ are the starting and terminal node respectively; $Q_a \in \mathbb{R}_{>0}$ will be used to indicate battery capacity and charing rate and discharging rate will be represented as $R_a^+ \in \mathbb{R}_{>0}$ and $R_a^- \in \mathbb{R}_{>0}$ respectively; $\theta_a \in [0, 1]$ is used to model the battery breakpoint and $B_a(t) \in \mathbb{R}_{\geq 0}$ is the state of charge at time t; $\mathcal{R}_a$ is the set of requests assigned to vehicle a and $\mathcal{T}_a$ is the type. For a more detailed understanding of the vehicle type, the reader is encouraged to analyze the discussion in [6]. For the purpose of this work, the vehicle type will be understood as a tuple $\langle P_a, G_a, C_a, F_a \rangle$, where , $G_a \in \mathbb{R}_{\geq 0}$ and $P_a \in \mathbb{R}_{\geq 0}$ are the goods and people capacity and $C_a \in \mathbb{R}_{>0}$ and $F_a \in \mathbb{R}_{>0}$ indicate operational cost and pollution factor respectively. Furthermore, each set of assigned requests $\mathcal{R}_a$ is a subset of the set of all requests in the systems, i.e. $\mathcal{R}_a \subseteq \mathcal{R}$. For the purpose of this model, we will assume each request assignment to be unique, i.e. $\mathcal{R}_a \neq \mathcal{R}_b$ and $\mathcal{R}_a \subseteq \mathcal{R} \setminus \mathcal{R}_b$, for all $(a, b) \in \mathcal{A}$ with $a \neq b$.

The battery will be modeled according to the two operating mode, i.e. charging and discharging. While it is understood that these two operations are highly influenced by multiple factors, it is sensible to make assumptions in order to simplify the model. As it is also widely spread in industry, one can neglect the influence of external factors such as weather condition or intrinsic characteristics of the battery, such as temperature or age. The charging profile will be modelled taking inspiration from the model proposed by *Lee et al.* in [32]. As mentioned above, the vehicles have a charging rate, a battery capacity and a breakpoint, namely R_a^+, Q_a and θ_a respectively. The state of charge at time $t \in \mathbb{R}_{\geq 0}$ of a certain vehicle will be obtained according to the model described in Equation

4.2, which is derived from the CC-CV (Constant current - Constant Voltage) scheme. (A more complete explenation can be found in [35]).
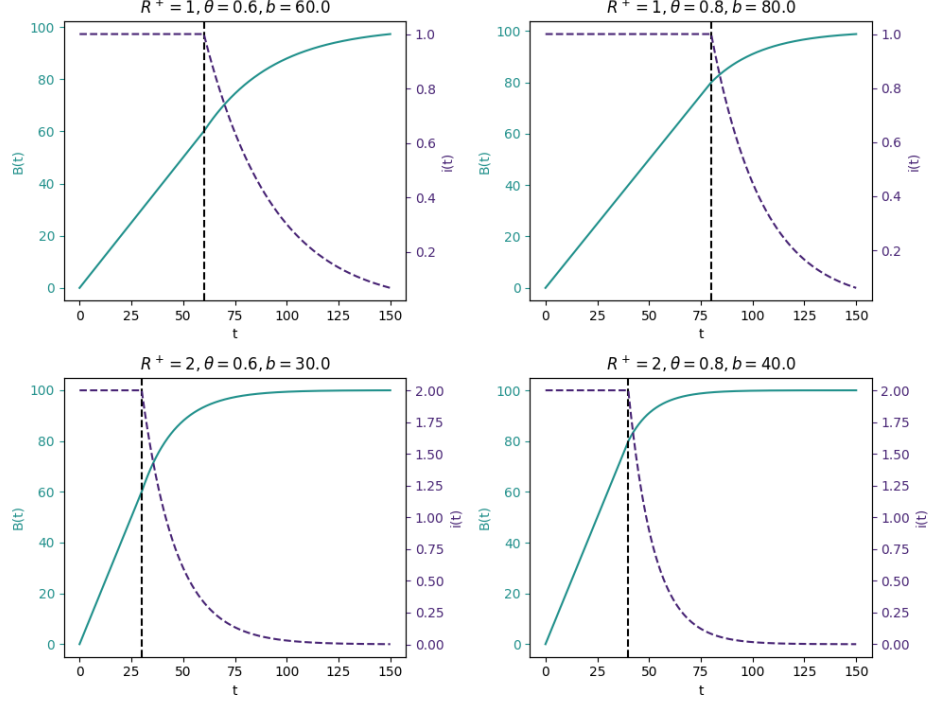

Figure 4.1: Different charging profile obtained according to the model developed in Equation 4.2. B is the State of Charge in Percentage, while $i(t)$ in function of the time unit t. The battery have $Q = 100$

$$B_a(t+1) = \begin{cases} B(t)_a + R_a^+ & \text{if } t \leq b_a \\ \\ Q_a - \dfrac{Q_a i(t)(1 - \theta_a)}{R_a^+} \end{cases} \tag{4.2}$$

with $b_a = \dfrac{\theta_a Q_a}{R_a^+}$, $i(t) = \begin{cases} R_a^+ & \text{if } t \leq b_a \\ R_a^+ e^{-(t-b_a)/\tau} \end{cases}$ and $\tau = \dfrac{C_n}{R_a^+ \theta_a} \in \mathbb{R}_{\geq 0}$ represent-

ing how quickly the current decreases. The value of C_n depends on the charging device considered.

An example of the charging profiles from this model can be seen in Fig 4.1. Since $i(t)$ is assumed to be constant at time $t \leq b_a$, i.e. $i(t) = R_a^+$,the first

condition is derived as follows.

$$B_a(t + \alpha) = B_a(t) + \int_t^{t+\alpha} R_a^+ \, dt$$

$$B_a(t + \alpha) = B_a(t) + R_a^+ \int_t^{t+\alpha} dt$$

$$B_a(t + \alpha) = B_a(t) + R_a^+ \quad t\Big|_t^{t+\alpha}$$

$$B_a(t + \alpha) = B_a(t) + R_a^+ (t + \alpha - t)$$

$$B_a(t + \alpha) = B_a(t) + \alpha R_a^+$$

with $\alpha = 1$.

The discharging, on the other hand, will be modeled as it being directly proportional to the travel time T, as described in Equation 4.3.

$$B_a(t + T_{u,v}^a) = B(t)_a - R_a^- T_{u,v}^a \tag{4.3}$$

The equation describes the relationship between the initial and final battery levels on edge $\langle v_u, v_v \rangle$ during the transit. Following Equation 4.3, one can assign to each edge a rate of battery discharge by defining it as the difference between the state of charge at time t and at time $t + T_{u,v}^a$.

$$e_{u,v}^a = B(t)_a - B_a(t + T_{u,v}^a) = R_a^- T_{u,v}^a \tag{4.4}$$

Equation 4.4 allows to describe the discharging rate as a function of only the edge and the vehicle.

In terms of operational costs, multiple factors should be considered and the analysis must be extended from the one developed in [6], where the operational cost was only in function of the vehicle type. Similarly to the aforementioned work, this model assumes a certain operational cost depending on the vehicle type, however the discussion is also extended in terms of vehicle charging cost. In other words, we can decouple it from the general concept of the operational cost per vehicle and considering in function of the charging or discharging profiles described above. Furthermore, it also makes sense to assign the cost also depending on the distance traveled, i.e. the value d associated to the edges.

Compared to [6], in this work, the model also extends the information associated to the nodes. Nodes will be of two categories namely normal nodes and charging (or depot) stations, whose sets are denoted as $\mathcal{V}_n$, $\mathcal{V}_c$, respectively. We can, therefore, conclude that $\mathcal{V} = \mathcal{V}_n \cup \mathcal{V}_c$. The information associated to the nodes depend from the category each node belongs to. Nodes belonging to $\mathcal{V}_n$ do not possess any specific information, as they are not of high importance for this work. Charging nodes $\mathcal{V}_c$, on the other hand, possess two main characteristics, namely their capacity and their charging ability. Intuitively, the term capacity is used to refer to the number of avilable parking spots, identified as

z_v. Furthermore, three types of charging ability will be considered, i.e. fast, normal and slow charging, which will be based on the model in Equation 4.2. The choice of parameters is explained in Table 4.1.

Type	C_n	τ	θ_a	R_a^+
Slow	20	33.$\bar{3}$	1	0.6
Medium	15	18.17	1	0.8
Fast	10	6.25	2	0.8

Table 4.1: Choice of parameters for the charging stations according to the type. For the charging profile, please refer to Fig 4.1. The first column is the costant related to the charging type, the second is the $\tau = \dfrac{C_n}{R_a^+ \theta_a} \in \mathbb{R}_{\geq 0}$ calculated with the values on the third and fourth column.

It is a sensible choice to limit the domain of the starting and terminal node of each vehicle to the sets of charging and depot nodes, i.e. $\underline{s_a} \in \mathcal{V}_c \cup \mathcal{V}_d$ and $\bar{t}_a \in \mathcal{V}_c \cup \mathcal{V}_d$. In other words, this choice implies that each vehicle's destination will either be the depot at the end of the shift, for example, or a charging node, where its battery can be charged to accomodate future requests.

Requests are modeled as tuples $\langle \underline{s}', \bar{t}', G', P', \lambda, a', b' \rangle$, where $\underline{s}' \in \mathcal{V}_n, \bar{t}' \in \mathcal{V}_n$ represent the pickup and delivery point respectively; $G' \in \mathbb{R}_{\geq 0}$ ($P' \in \mathbb{R}_{\geq 0}$) refers to the amount of goods (people) required to transport, and $\lambda \in \mathbb{R}_{> 0}$ is the rate of requests, in customers per unit time, which, therefore, makes the requests stationary and deterministic. Additionally, requests must be delivered within a time window within $[a', b']$.

4.1.1   Model Evalutation

Some comments are in order. The model in Section 4.1 is time-invariant. According to the definition presented by *Frazzoli et al.* in [61], time invariance in the context of transportation modeling refers to the assumption that the number of requests remain constant over time allowing for the simplification of temporal dynamics and treating specific time intervals as homogeneous units. This modeling concept is employed when the rate of change in transportation demands is deemed slow compared to the average travel time of individual trips, often observed in stable urban environments ([38]). While this model is indeed applied in a relatively dense urban environment, some integrations are required in order to adapt it to time-varying scenarios. Moreover, customer requests are also assumed to be known. This requirement can be fulfilled in practice with requests made in advance or some techniques to estimate requests throughout the day. It is important to note that request estimation might lead to suboptimal performance.

Secondly, as already mentioned in the previous section, the model used to de-

scribe congestions is indeed rather simple and more complex formulation might better capture the phenomena. However, the semplification is considered powerful enough for the purpose of this work and, as pointed out by *Zhang et al.* in [46] as well, more sophisticated models can be used offline using simulation techniques to derive the capacity metric used in this model.

Some comments are in order also regarding vehicles. Firstly, we assume the vehicles to be autonomous and fault proof. It is outside of the scope of this work to deal malfunctioning vehicles or exceptional situations outside of the normale functioning regime of the system.Additionally, we assume the vehicles to be fully electric. This assumption is widely used in literature and it is motivated, among other aspects, also by the recent trends in industry to transition towards electrical mobility. Furthermore, the model provided for the battery charging and discharging profiles is rather simplistics and more sophisticated approaches exist in literature (see Chapter 3 for more details). However, these models should be seen an addition to the corrent approach and, albeit with some potential modifications, it is plausible they can be integrated in the system model. While in [6] vehicles have been designed to be capable of transporting a potentially large number of people, some considerations should be made in this regard. While it is sensitive to consider vehicles capable of transporting up to 50 people in terms of environmental, economical and overall transportation efficiency and while it is also expected that the model designed would work with those vehicles as well, in terms of practical use, it might be more appropriate to make use of those vehicles in different ways. Notably, it should be studied whether it is efficient to tailor the routes of those vehicles according to passenger needs. Firstly, if we do not consider ride-sharing possibilities, the scenario of such large group of people traveling together is very unlikely. Additionally, if we consider ride-sharing, the possible route combinations, which increases drastically as the number of requests and nodes in the graph increase, would likely make routing such vehicle relatively inefficient. Motivated also by the fact that passengers might have common stops, it might be more sensible to treat those vehicles as buses are treated nowadays, i.e. with a pre-determined route among stops which are placed according to the most common stops, such as hospitals or train stations. Similarly, if we consider large vehicles to transport goods, such as large trucks, those are not used for home delivery, but rather used to transport goods among specialized centers. It must be noted that both situations do not invalidate this work, or any previous work. It is clear that the system described in this work in combination to the truck for goods transportation. Regarding people mobility, while buses are indeed an already established and efficient transportation mean, this systems aims at filling up the situations where a bus system is indeed lacking, such as transportation of people with special needs or, in general, more tailored to the specific needs of potential customers.

In this model, we also made certain assumptions on the nodes. Firstly, there is no distinction between depot and charging nodes. The difference between charging and depot stations lays on the fact that charging nodes are not meant to host vehicles for long periods of time, contrary to depots. The difference can be envisioned as if charging stations were gas stations and depots were bus depots.
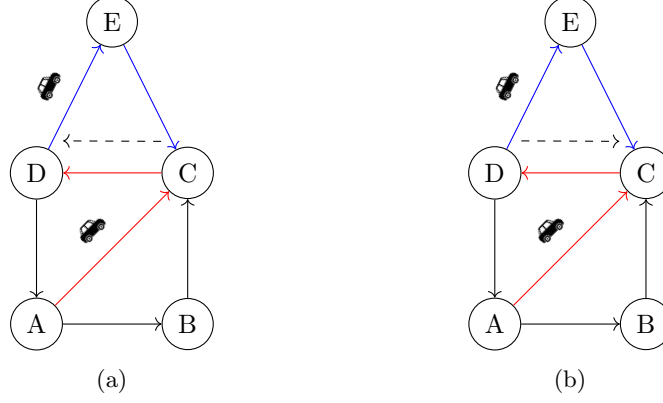
Figure 4.2: Fig 4.2a and Fig 4.2b show a simplified example of a sensible request assignment. In red and blue are the paths the two AVs can traverse, while the dashed blue two different requests (In Fig 4.2a the custoer asks to go from C to D, while in Fig 4.2b the customer asks to go from D to C). In the case of Fig 4.2a, it is more sensible to assign the request to the red AV, while in the case of Fig 4.2b, the blue AV is a better choice.

The model can be trivially extended according to this distinction. Additionally, it is assumed that the charging stations are all of the same types. It might be argued that some stations might have different types of chargers. This characteristics can be reflected by the model simply by 'splitting' them. A station having, for instance, two types of chargers can be represented by two equivalent nodes in the graph having different category. Moreover, while in Table 4.1 we only considered three modes, this can be easily extended.

4.2 Problem Formulation

Before defining the problems for the different challenges faced by the AToD considered in this work, it is essential to lay down the motivations behind certain assumptions made during the conceptual phase.

Firstly, following the previously described model, it is assumed that each vehicle starts and ends at a charging or depot node, $\underline{s_a}$ and $\bar{t_a}$, which do not have to be necessarily the same. Furthermore, if a vehicle leaves from a depot, it is assumed to have a fully charged battery, i.e. $B_a(0) = 100$.

4.2.1 Dispatching

Informally, the dispatching problem can be defined as the task of assigning requests to the most suitable vehicle. There exist already multiple solutions proposed for the problem and we refer to Chapter 3 for a more thorough analysis. Dispatching is critical for the overall system performance and must be done in

a way that can further facilitate the next steps. Clearly, dispatching can not be decoupled and solved as a stand-alone problem. For example, Fig 4.2 shows a simplified situation where a sensible dispatching, which depends on a posterior step, will improve system performance. Nevertheless, during the dispatching problem, some additional elements must be considered as well.

Since the model proposed in Section 4.1 is a vehicle centric model, we can model the dispatching problem with the help of a binary variable x_{ar} defined in Equation 4.2.1.

$$x_{a,r} = \begin{cases} 1 & \text{if } r \text{ is assigned to } a \in \mathcal{A} \\ & \quad\quad\quad\quad\quad\quad\quad\quad \forall r \in \mathcal{R} \\ 0 \end{cases}$$

According to the model, each vehicle has a capacity which must not be exceeded. Such constraint can be expressed as follows for both people and goods.

$$\sum_{r \in \mathcal{R}} P'_r \cdot x_{a,r} \leq P_a \quad \forall a \in \mathcal{A} \tag{4.5}$$

$$\sum_{r \in \mathcal{R}} G'_r \cdot x_{a,r} \leq G_a \quad \forall a \in \mathcal{A} \tag{4.6}$$

Furthermore, if a vehicle is already on the move, the request can be picked up only if the vehicle's charge is enough to satisfy such request as well. In this case, such requirement can be expressed as follows.

$$\sum_{r \in \mathcal{R}} e(\underline{s'_r}, \bar{t'_r}) x_{a,r} \leq B_a \quad \forall a \in \mathcal{A} \tag{4.7}$$

where $e : \mathcal{E} \times \mathcal{E} \to \mathbb{R}_{>0}$ is a function expressing the required energy to go from $\underline{s'_r}$ to $\bar{t'_r}$.

Accordingly, using Equation 4.5, 4.6 and Equation 4.7, one can simply formulate it as an integer programming problem by finding the appropriate cost function to minimize, such as minimizing waiting times. Thanks to this formulation, one can also make sure that each request has been served at most λ_r times , i.e. Equation 4.8.

$$\sum_{a \in \mathcal{A}} x_{a,r} \leq \lambda_r \quad \forall r \in \mathcal{R} \tag{4.8}$$

Furthermore, one must ensure that each request is assigned at most to one vehicle. This is expressed by Equation 4.9.

$$\sum_{r \in \mathcal{R}} x_{a,r} \leq 1 \quad \forall a \in \mathcal{A} \tag{4.9}$$

The above mentioned equations are based upon the work developed by *Hyland et al.* in [26].

This way allows to define the following cost function which expresses the number of served requests.

$$\mathcal{J}_u = \sum_{r \in \mathcal{R}} (1 - \min_{a \in \mathcal{A}}(x_{a,r}, 1)) \tag{4.10}$$

The main strength of this approach is that can be integrated naturally in the formulation for the other steps, like for e.g. the one in Section 4.2.2.

Alternatively, the dispatching problem can be solved independently without being integrated in other steps. For example, in Section 3, some works are mentioned that make use of heuristics such as nearest neighbours. On the one hand,these approaches are known to obtain sub-optimal solutions for the problem; on the other hand, they provide flexibility and might result in less time or space complexity.

4.2.2 Routing

After being assigned to incoming requests, vehicles must be routed in such a way that can reach all customers and therefore satisfy all the requests. In other words, the routing problem consists of determining paths, i.e. a series of edges, each vehicle must travel in the graph to fulfill the requests while respecting all the requirements and minimizing some metrics, i.e. a cost function. Since this problem can be reconduced to the vehicle routing problem, in particular dynamic pickup and delivery problems (reviewd by *Toth et al.* in [52] and *Laporte et al.* in [31]), the formulation used in this work will be based on this family of problems.

Within our model, binary flow variables will be used to identify whether a vehicle should traverse a link. Formally, this can be expressed as

$$V_{u,v}^a = \begin{cases} 1 & \text{if } a \text{ traverses } (u,v) \in \mathcal{E} \\ \\ 0 \end{cases} \qquad \forall a \in \mathcal{A}, \forall u, v \in \mathcal{V}$$

Furthermore, another variable will be needed to deal with time-related requirements. This concept is based on the work in [27]. The decision variable s_u^a indicates the service time of vehicle a at node u. This variable will be relevant only for nodes labeled as being terminal stations for the requests, i.e. $\underline{t'a}$ and will be irrelevant for other nodes.

Accordingly, one can express the abovementioned cost function using this variable. Classical examples of cost functions are for e.g. travel time and travel distance. These are frequently used in literature ([2]), as they are general, in a sense that many other metrics could be reconduced to them. For example, residual charging or operational costs are directly influenced by the two. Furthermore, the formualtion of the cost function derived from these metrics is rather trivial and intuitive, while at the same time producing desirable results in practice. The formulation is described in equation Equation 4.11 and 4.12.

$$\mathcal{J}_T = \sum_{a \in \mathcal{A}} \sum_{(u,v) \in \mathcal{E}} T_{u,v}^a V_{u,v}^a \tag{4.11}$$

$$\mathcal{J}_d = \sum_{a \in \mathcal{A}} \sum_{(u,v) \in \mathcal{E}} d_{u,v} V_{u,v}^a \tag{4.12}$$

In the effort to minimize the environmental impact of the AToD system, the pollution index discussed in previous sections can be utilized to formulate the cost function outlined in Equation 4.13.

$$\mathcal{J}_f = \sum_{a \in \mathcal{A}} \sum_{(u,v) \in \mathcal{E}} f_{u,v}^a V_{u,v}^a \tag{4.13}$$

Up to this point, the metrics under consideration have been geared towards minimization. Put differently, the goal is to reduce travel, encompassing both distance and time, to enhance system performance. Likewise, minimizing environmental impact is crucial in this scenario. However, there are instances where maintaining certain metrics at higher levels is preferable. For instance, closely tied to operational costs and environmental impact, it is advantageous to keep the state of charge at its maximum. Hence, it is imperative to maximize the cost function in Equation 4.14.

$$\mathcal{J}_B = \sum_{a \in \mathcal{A}} B_a \tag{4.14}$$

Finally, while it should also be explored whether the combination of those could improve the overall performance of the system. For this purpose, the cost functions can be combined using weights as follows.

$$\mathcal{J}_{tot} = \lambda_T \mathcal{J}_T + \lambda_d \mathcal{J}_d + \lambda_f \mathcal{J}_f + \lambda_B \mathcal{J}_B \tag{4.15}$$

where $\lambda_i \in \mathbb{R}$ with $i \in \{T, d, f, B\}$ are the weights.

It is not uncommon to also find in literature cost functions which are developed in terms of operational costs, as it provides a general idea to reason about this problem and allows for a systematic evaluation of different dispatching strategies. Moreover, the inclusion of operational costs in the literature emphasizes the real-world impact of dispatching decisions, aligning theoretical models with practical considerations. This connection to tangible costs not only enhances the applicability of proposed solutions but also contributes to the development of more realistic and effective dispatching strategies. Considering that, within our model in Section 4.1, each vehicle is modeled to have an operational cost, one can also derive a cost function similarly to what already done in [6].

Unconstrained Version

Before formulating the routing problem formally, a small consideration must be made. For convenience, the set of nodes recheable from node u by traversing a single edge $\mathcal{N}_u^+$, i.e. the ingoing neighbours of u, and $\mathcal{N}_u^-$ as the set of outgoing neighbours of u.

Additionally, in order to ease the notation, the set containin all the initial stations of each request $r \in R_a$ will be indicated as $\underline{S_a'}$. Likewise, $\bar{T}_a'$ will be the set of terminal stations of each request $r \in R_a$.

On a basic level, i.e. without constraints, one can formulate the *Unconstrained Routing Problem (URP)* as follows. Given a transportation network $\mathcal{G}$, a set of vehicles $\mathcal{A}$ and a set of requests $\mathcal{R}$, defined within the description in section Section 4.1, solve:

min (4.11), (4.12), (4.13) or (4.15)

s.t.

$$\sum_{u \in \mathcal{V}} V_{u,v}^a - \sum_{w \in \mathcal{V}} V_{v,w}^a = 0 \quad \forall a \in \mathcal{A}, v \in \mathcal{V} \setminus \{\underline{s_a}, \bar{t}_a\} \tag{4.16}$$

$$\sum_{u \in \mathcal{N}_{\underline{s_a}}^+} V_{\underline{s_a},u}^a = 1 \quad \forall a \in \mathcal{A} \tag{4.17}$$

$$\sum_{u \in \mathcal{N}_{\bar{t}_a}^-} V_{u,\bar{t}_a}^a = 1 \quad \forall a \in \mathcal{A} \tag{4.18}$$

$$\sum_{u \in \mathcal{N}_{\bar{t}_r'}^-} V_{u,\bar{t}_r'}^a = 1 \quad \forall r \in \bar{R}_a, \forall a \in \mathcal{A} \tag{4.19}$$

$$\sum_{u \in \mathcal{N}_{\underline{s_r'}}^+} V_{\underline{s_a'},u}^a = 1 \quad \forall r \in \bar{R}_a, \forall a \in \mathcal{A} \tag{4.20}$$

$$p_u^a - p_v^a + P_a \cdot V_{u,v} \le P_a - \sum_{r \in \mathcal{R}} P_r', \quad \forall u, v \in \mathcal{V}, v \ne u, v \ne \underline{s_a} \tag{4.21}$$

$$g_u^a - g_v^a + G_a \cdot V_{u,v} \le G_a - \sum_{r \in \mathcal{R}} G_r', \quad \forall u, v \in \mathcal{V}, v \ne u, v \ne \underline{s_a} \tag{4.22}$$

$$g_u^a \le P_a; \quad p_u^a \le P_a, \quad \forall a \in \mathcal{A} \tag{4.23}$$

(4.16) insures that for all edges which do not lead to a source or destination, if a reaches v from a road, an incoming flow will lead to an outgoing one. In other words, they guarantee connections between roads. (4.17) - (4.18) insure that each universal source and destination is reached ones. Similarly, (4.19) - (4.20) achieves the same result, but for each requests. It should be mentioned that to guarantee that those special nodes are reached at most once one can simply change from equalities to inequalites, like for e.g. $\sum_{u \in \mathcal{N}_{\underline{s_a}}^+} V_{\underline{s_a},u}^a \ge 1$.

Finally, (4.21)-(4.23) assure that no sub-tour are presentes. This is inspired from the Miller-Tucker-Zemli formulation and utilizes two continuous decision variables $p^a, g^a \in \mathbb{R}_{\geq 0}$ representing the cumulative load of the vehicle a.

In order to find the minimum number of vehicles required, one can use the method described *Brodo* in [6].

Constrained Version

While it could bring some interesting insights, the URP does not contain the necessary information to provide efficient routing for the scenario considered in this work. The goal is to identify the best possible path that (i) satisfies all the requests and ii is congestion free. The *Congestion-Free Routing Problem (CRR)* is formally defined as follows.

Given a transportation network $\mathcal{G}$, a set of vehicles $\mathcal{A}$ and a set of requests $\mathcal{R}$, defined within the description in section Section 4.1, solve:

$$\min (4.11), (4.12), (4.13) \text{ or } (4.15)$$

$$\text{s.t.}$$

$$(4.16) - (4.20)$$

$$\sum_{a \in \mathcal{A}} V_{u,v}^a \leq c_{u,v} \quad \forall (u,v) \in \mathcal{E} \tag{4.24}$$

$$s_u^a + T_{u,v}^a - M * (1 - V_{u,v}^a) \leq s_v^a \quad \forall (u,v) \in \mathcal{E}, \forall a \in \mathcal{A} \tag{4.25}$$

$$a_v' \leq s_v^a \leq b_v' \quad \forall v \in \mathcal{E}, \forall a \in \mathcal{A} \tag{4.26}$$

$$\sum_{(u,v) \in \mathcal{E}} e_{u,v}^a \cdot V_{u,v}^a \leq B_a(0) \quad \forall a \in \mathcal{A} \tag{4.27}$$

$$M = \max_{(u,v) \in \mathcal{E}} \{b_u + T_{u,v} - a_u\}$$

(4.24) insures the number of vehicles in the link (u,v) do not exceed the capacity of that link. (4.25) establishes the relationship between the service time of each node, implying that the service time of a predecessor must be lower than the successor. (4.26) establish the time window constraints, indicating that it must be within the interval of the request. For nodes which are not associated with a termination node of a request, one will simply set $a_v' = 0$ and $b_a' = \infty$, also insuring that $s_v^a \geq 0$. (4.27) assures that the vehicle charge is enough to cover all path. $B_a(0)$ can be assumed to be 100, i.e. that the batteries are full at the beginning of service.

In this formulation, the sub-tour elimination constraint is not required at it is already imposed by (4.25).

Combination with Dispatching

Multiple solutions have already been proposed in literature which solve the dispatching and routing problem in the same work.

In this work, however, in order to combine dispatching and routing into the same formulation, one must adapt some of the conditions specified in previous sections above. More specifically, each equation related to the requests must be changed to accomodate the fact that requests have not been previously assigned. Accordingly, the *Naive Routing and Dispatching Problem (NRDR)* can be formulated as

$$\min (4.10), (4.11), (4.12), (4.13) \text{ or } (4.15)$$
$$\text{s.t.}$$
$$(4.5), (4.6)$$
$$(4.8), (4.9)$$
$$(4.16)$$
$$(4.24) - (4.27)$$

$$\sum_{u \in \mathcal{N}^+_{\underline{s_a}}} V^a_{\underline{s_a}, u} = 1 \quad r \in \mathcal{R}, \forall a \in \mathcal{A} \tag{4.28}$$

$$\sum_{u \in \mathcal{N}^-_{\overline{t_a}}} V^a_{u, \overline{t_a}} = 1 \quad r \in \mathcal{R}, \forall a \in \mathcal{A} \tag{4.29}$$

$$\sum_{u \in \mathcal{N}^-_{\overline{t'_r}}} V^a_{u, \overline{t'_r}} \geq x_{a,r} \quad \forall r \in \mathcal{R}, \forall a \in \mathcal{A} \tag{4.30}$$

$$\sum_{u \in \mathcal{N}^+_{\underline{s'_r}}} V^a_{\underline{s'_r}, u} \geq x_{a,r} \quad \forall r \in \mathcal{R}, \forall a \in \mathcal{A} \tag{4.31}$$

Constraints (4.28) and (4.29) impose that a vehicle must leave the starting depot and reach its final depot, respectively. (4.30) guarantees that, when a request r is allocated to vehicle a, the vehicle must reach the terminal station of the request at least once. Moreover, equation (4.31) ensures that the vehicle traverses the starting station associated with the assigned request.

These requirements, furtermore, with this formulation, also ensure that a vehicle can only move from one node to the other only if they left the starting depot as well.

As an alternative to (4.28) and (4.29), one can also relate the two constraints to

the request assignment, which would transform the requirements as following.

$$\sum_{u \in \mathcal{N}^+_{\underline{s_a}}} V^a_{\underline{s_a},u} = \max_{r \in \mathcal{R}}\{x_{a,r}\} \quad \forall a \in \mathcal{A} \tag{4.32}$$

$$\sum_{u \in \mathcal{N}^-_{\bar{t_a}}} V^a_{u,\bar{t_a}} = \max_{r \in \mathcal{R}}\{x_{a,r}\} \quad r \in \mathcal{R}, \forall a \in \mathcal{A} \tag{4.33}$$

In other words, a vehicle must leave the starting depot and reach the final depot if a request is assigned to it.

One can argue that the CRR can be reformulate to solve the dispatching in a different way.

Let

$$x_{a,\underline{s_r}} = \begin{cases} 1 & \text{if } \underline{s_r} \text{ is reached by } a \in \mathcal{A} \\ 0 \end{cases} \quad \forall r \in \mathcal{R}$$

and given a transportation network $\mathcal{G}$, a set of vehicles $\mathcal{A}$ and a set of requests $\mathcal{R}$, defined within the description in section Section 4.1, solve:

$$\min (4.11), (4.12), (4.13) or (4.15)$$
$$\text{s.t.}$$
$$(4.16) - (4.18)$$
$$(4.24) - (4.27)$$

$$\sum_{a \in \mathcal{A}} \sum_{u \in \mathcal{N}^+_{\underline{s'_r}}} \alpha_a \cdot V^a_{\underline{s'_r},u} \geq \alpha'_r \quad \forall r \in \mathcal{R}, \forall \alpha \in \{G,P\} \tag{4.34}$$

$$\sum_{a \in \mathcal{A}} \sum_{u \in \mathcal{N}^+_{\underline{s'_r}}} V^a_{\underline{s'_r},u} = \lambda_r \quad \forall r \in \mathcal{R} \tag{4.35}$$

$$\sum_{u \in \mathcal{N}^+_{\underline{s'_r}}} V^a_{\underline{s'_r},u} \leq \sum_{u \in \mathcal{N}^-_{\bar{t'_r}}} V^a_{u,\bar{t'_r}} \tag{4.36}$$

(4.34) insures that request can be served by the vehicle according to its capacity. (4.35) guarantees each request is served λ_r times and (4.36) insures that each vehicle reaches the terminal station of each request.

4.2.3   Rebalancing

In simpler terms, the rebalancing problem revolves around efficiently redistributing autonomous vehicles (AVs) to optimize their responsiveness to new ride requests while minimizing any existing imbalances in the system. The goal is
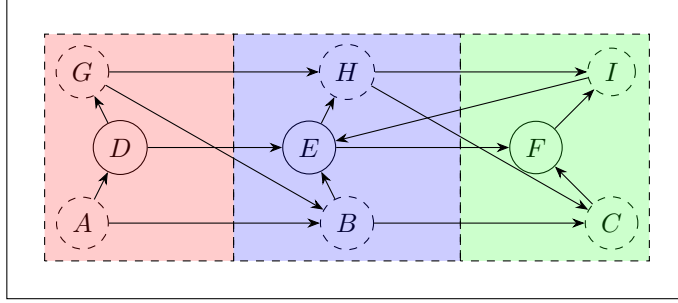
Figure 4.3: Simplfied example for the rebalancing strategy. Nodes D, E, F (continuos circles) to be the depot (or charging stations), while the other nodes (dashed circles) to be normal nodes. The graph has been previously divided in three areas (red, blue and green).

to fine-tune the positioning of AVs, ensuring they are strategically placed to promptly meet user demands and address any inherent irregularities in the distribution of service requests. This challenge is particularly crucial in ride-sharing systems and transportation systems alike, where the dynamic nature of user requests and varying demand across different locations can lead to imbalances in the fleet's distribution. Effectively tackling the rebalancing problem enhances the overall efficiency of the system, providing users with quicker response times and a more evenly distributed service, ultimately contributing to a smoother and more reliable autonomous transportation network.

The formulation of the rebalancing problem is partially inspired from the works of *Zhang et al.* in [46] and *Wallar et al.* in [56].

The model previosuly described allows to reason on the rebalancing model in a novel manner when compared to the literature reviewed in Chapter 3. More specifically one could deal with the rebalancing by leveraging the request arrival rate λ_r and the fact that vehicles are assumed to have a starting and terminal station, $\underline{s_a}$ and $\bar{t_a}$ respectively.

Considering the situation depicted in Fig 4.3 with a set of requests $\mathcal{R}'$ and a sets of vehicles $\mathcal{A}'$. According to this example, the graph is contained within the space Ω, which has been divided in three cells (in red, blue and green). This division has been done arbitrarily in this case, but different strategies can be used ([56], [66]). These regions must be specified, however, in such a way that they contain exactly one node belonging to $\mathcal{V}_c$ and more than $n \in \mathbb{Z}_{\geq 1}$ nodes belonging to $\mathcal{V}_n$. Informally, this means that those regions are build around each charging station or depot and contain a certain amount of normal nodes. Different strategies could be used to determin this amount, like for e.g. all the nodes within a radius r from each terminal node or within a driving distance T_{max}. Nevertheless, as a result of this, one will obtain a set of regions R. Formally, each region around a node $v \in \mathcal{V}_c$ can be defined as a subgraph $\mathcal{G}_v = \langle \mathcal{V}'_v, \mathcal{E}'_v \rangle$,

where

$$\mathcal{V}'_v = \{u \in \mathcal{V}_n : (u,v) \in \mathcal{E} \wedge (v,u) \in \mathcal{E}, f(v,u) = 1\} \cup \{v\} \qquad (4.37)$$
$$\mathcal{E}'_v = \{(u,w) \in \mathcal{E} : u,w \in \mathcal{V}'\} \qquad (4.38)$$

In (4.37), the function $f : \mathcal{V}_c \times \mathcal{V}_n \to \{0,1\}$ is used to establish whether a node u belongs to the region of v ($f(v,u) = 1$) or not ($f(v,u) = 0$), as discussed before. Accordingly, one can obtain the total number of requests of region v ($|\mathcal{R}'_v|$) by considering the requests which have a terminal node in $\mathcal{V}_n$. In other words

$$|\mathcal{R}'_v| = \sum_{r \in \mathcal{R}'_v} \lambda_r \qquad (4.39)$$

where
$$\mathcal{R}'_v = \{r \in \mathcal{R} : \underline{s_r}' \in \mathcal{V}'_v\} \qquad (4.40)$$

In simple words, the set of $\mathcal{R}'_v$ indicates all the requests that start within the region of the node $v \in \mathcal{V}_c$.
Analogally, one can provide an alternative definition for $\mathcal{R}'_v$ in terms of the terminal station u.
$$\mathcal{R}'_u = \{r \in \mathcal{R} : \bar{t}_r' = u\} \qquad (4.41)$$

Requests, therefore, are distributed over those regions and it is not hard to envision scenarios where requests are unequally distributed, i.e. regions with a higher $|\mathcal{R}^v|$ than others. As a result, due to the heterogenous nature of the nodes and the requests, AToD might experience imbalance, as AVs can be scattered through the whole graph if requests have different terminal nodes. Conversely, the opposite might happen, if all the requests have the same terminal node, but start from different areas.
Assuming all the requests in $\mathcal{R}'$ to be served, each vehicle's starting position is now either D, E, or F, i.e. $\underline{s_a} \in \{D, E, F\} \quad \forall a \in \mathcal{A}'$. This is due to the fact that, according to the transportation network model, since the requests have all been served, the vehicles have all reached their previous ending stations $\bar{t}_a$, which became their new starting nodes.
As a result, the solution of the rebalancing problem becomes ensuring that there is a sufficient number of vehicles at the depot (or charging) nodes to fulfill as many requests as possible in the given area, potentially covering all requests.

Unconstrained Rebalancing Problem

To formulate the different versions of the rebalancing problem binary flow variables will be introduced to help with the different formulations. Similarly to Section 4.2.2, those variables will be useed to indicate whether an idle vehicle, i.e. a vehicle not currently transporting customers or goods, should be moving

from a node to another. Formally, this is expresseed as follows.

$$y_{u,v}^a = \begin{cases} 1 & \text{if } a \text{ traverses } (u,v) \in \mathcal{E} \\ \\ 0 \end{cases} \qquad \forall a \in \mathcal{A}, \forall u, v \in \mathcal{V}$$

Within the assumptions made at the beginning of this section, it follows that a vehicle will be considered idle if it is currently at its final station.

For the sake of simplicity and brevity, a generic cost function $\mathcal{J}'$ will be considered for the entirity of this section since the choice of a cost function is similar to the one for Section 4.1.

Finally, within the model established in prior sections, the *Unconstrained Rebalancing Problem (UReP)* can be formulated as following.

Given a transportation network $\mathcal{G}$, a set of idle vehicles $\mathcal{A}'$ and a set of nodes belonging to $\mathcal{V}_c$, defined within the description in section Section 4.1, solve:

$$\min \quad \mathcal{J}'$$
$$\text{s.t.}$$

$$\sum_{u \in \mathcal{V}} y_{u,v}^a - \sum_{w \in \mathcal{V}} y_{v,w}^a = 0 \quad \forall a \in \mathcal{A}', v \in \mathcal{V} \setminus \{\underline{s_a}, \bar{t_a}\} \tag{4.42}$$

$$\sum_{u \in \mathcal{N}_{\underline{s_a}}^+} y_{\underline{s_a},u}^a = 1 \quad \forall a \in \mathcal{A}' \tag{4.43}$$

$$\sum_{\bar{t} \in \mathcal{V}_c} \sum_{u \in \mathcal{N}_{\bar{t}}^-} y_{u,\bar{t}}^a = 1 \quad \forall a \in \mathcal{A}', \tag{4.44}$$

$$\sum_{a \in \mathcal{A}'} \sum_{u \in \mathcal{N}_{\bar{t}}^-} y_{u,\bar{t}}^a \leq z_{\bar{t}} \quad \forall \bar{t} \in \mathcal{V}_c \tag{4.45}$$

$$\sum_{a \in \mathcal{A}'} \sum_{u \in \mathcal{N}_{\bar{t}}^-} G_a \cdot y_{u,\bar{t}}^a \geq \sum_{r \in \mathcal{R}'_{\bar{t}}} G_r' \quad \forall \bar{t} \in \mathcal{V}_c \tag{4.46}$$

$$\sum_{a \in \mathcal{A}'} \sum_{u \in \mathcal{N}_{\bar{t}}^-} P_a \cdot y_{u,\bar{t}}^a \geq \sum_{r \in \mathcal{R}'_{\bar{t}}} P_r' \quad \forall \bar{t} \in \mathcal{V}_c \tag{4.47}$$

(4.42) insures the flow conservation. (4.43) indicates that the vehicles leave the starting node only from one edge. (4.44) guarantees a vehicle must reach only one deposit at the time. (4.45) insures there the capacity of the deposits is respected. Finally, (4.46) and (4.47) are used to ensure the vehicles being rebalanced to a certain station have the capacity to fulfill the requests of the area.

4.2.4 Constrained Congestion-Free Routing and Rebalancing Problem

The final *Constrained Congestion-Free Routing and Rebalancing Problem (CCRRP)* can be formulated as follows.

Given a transportation network $\mathcal{G}$, a set of vehicles $\mathcal{A}$, a set of total requests $\mathcal{R}$ and the sets of requests per region $\mathcal{R}'$ (Equation 4.40), solve:

$$\min (4.11), (4.12) \text{ or } (4.13)$$

$$\text{s.t.}$$

$$(4.16) - (4.17)$$
$$(4.34) - (4.36)$$
$$(4.25) - (4.27)$$

$$\sum_{\bar{t} \in \mathcal{V}_c} \sum_{u \in \mathcal{N}_{\bar{t}}^-} V_{u,\bar{t}}^a = 1 \quad \forall a \in \mathcal{A}, \tag{4.48}$$

$$\sum_{a \in \mathcal{A}} \sum_{u \in \mathcal{N}_{\bar{t}}^-} V_{u,\bar{t}}^a \leq z_{\bar{t}} \quad \forall \bar{t} \in \mathcal{V}_c \tag{4.49}$$

$$\sum_{a \in \mathcal{A}} \sum_{u \in \mathcal{N}_{\bar{t}}^-} G_a \cdot V_{u,\bar{t}}^a \geq \sum_{r \in \mathcal{R}'_{\bar{t}}} G_r' \quad \forall \bar{t} \in \mathcal{V}_c \tag{4.50}$$

$$\sum_{a \in \mathcal{A}} \sum_{u \in \mathcal{N}_{\bar{t}}^-} P_a \cdot V_{u,\bar{t}}^a \geq \sum_{r \in \mathcal{R}'_{\bar{t}}} P_r' \quad \forall \bar{t} \in \mathcal{V}_c \tag{4.51}$$

$$\tag{4.52}$$

Critically, compared to most of the reviewed literature, the strength of this approach lies in the fact that the rebalancing problem resembles more an assignment problem than routing. As a matter of fact, vehicles are routed in such a way that all the requests are guaranteed to be served and at the same time, the inbalance nature of AToD systems is tackled by ensuring that each region has enough vehicle to deal with future requests.

Furthermore, due to its vehicle-centric approach, this formulation offers significant flexibility across several dimensions: (i) it accommodates diverse request types, such as goods or people, (ii) it allows for a wide range of vehicle characteristics and specification, like for e.g. different capacities or dischargin profiles, and (iii) it remains adaptable to various node characteristics, as nodes have different number parking spaces, for example. Simultaneously, it exhibits scalability, with the number of variables increasing linearly in relation to the edges in the graph and the number of vehicles under consideration.

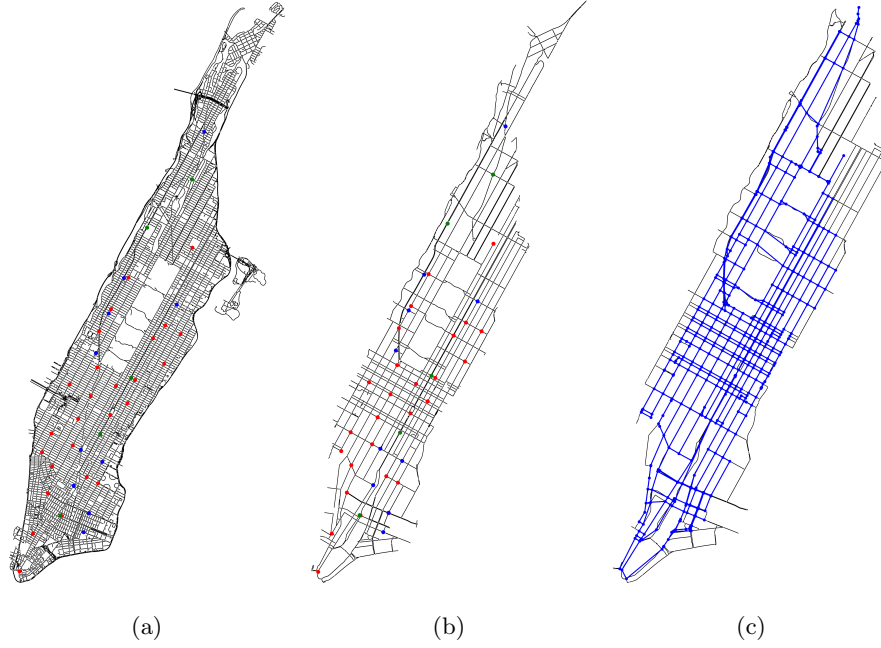(a)   (b)   (c)

Figure 4.4: Manhattan's Road Network Representation. In (a), red points are used to indicate pick-up stations, while blue delineates drop-off points and green is used to pin-point the fictitious depots. These are obtained by the historical data. Furthermore, pick-up points can also be used as drop-off, and viceversa. (b) shows the semplified road network consisitnf of only the most important streets. Finally, (c) shows the final version of the road network after the application of the minimum spanning tree algorithm.

4.3   Use Case

The main purpose of this use case is to demonstrate the applicability of the model in the real world and the feasibility of the solutions presented in Section 4.2. This use case is based on real-world data extracted from taxi rides in manhattan in 2016 ([12]).

The data has been prepared in the following way. Since the data retrieved is based on the whole New York City, in order to decrease the complexity, some filters have been applied to only target the Manhattan area. More specifically, the pick-up and drop-off points which were outside of Manhattan have been filtered out. Subsequentially, these pick-up and drop-off points have been clustered in 40 locations. Finally, these locations have been mapped to the real Manhattan road network and the shortest path between five neighbours nodes has been traced in order to create a simplified road network, which can be seen in Fig 4.4.

As a consequence, the complexity of Manhattan's road network has been sig-

nificantly reduced. Within the context of this project, and specifically in this section, this simplification is deemed acceptable since only proof of concepts are being considered. Furthermore, employing a more intricate representation of the road network would not significantly enhance the objectives at hand. This streamlined approach not only facilitates clearer demonstration of concepts but also expedites the analysis process, enabling a more focused examination of essential elements. Thus, for the current scope of this work, the simplified representation remains suitable and pragmatic. According to the discussion in Section 4.2.3, the region has been divided in multiple areas as well. This can be observed in Fig 4.4a. Although the division in areas it's not explicitly stated, five green-highlited nodes can be observed. These nodes represent depots and have been created to fulfill the same purpose of Section 4.2.3. In order to improve the performance of the simulation, the road network has been simplified by considering only the most important streets in Manhattan, as shown in Fig 4.4b and Fig 4.4c. This approach optimizes resources while still providing valuable insights into the model performance. Fig 4.4c specifically has been obtained by first calculating the shortest path, in terms of distance, of the points highligthed in Fig 4.4a. Furthermore, the Minimum Spanning Tree (MST) algoritm ([37]). The MST is used to calculate the smallest possible connected subtree that spans all the vertices in a given graph. In essence, constructing MST involves iteratively selecting edges with the smallest weight while ensuring that no cycles are formed. This process starts with any arbitrary vertex and gradually expands by adding the lightest edge connected to the existing tree until all vertices are encompassed.

While information regarding customer demands are relatively easy to obtain, other data must be crafted or estimated. In particular, details about the constraints outlined in Section 4.1 can be challenging to gather as they are often specific to certain scenarios or dependent on external factors not captured by the model. For instance, determining the threshold in the congestion model is complex, especially considering it could easily reach thousands in a densely populated city like NYC. Likewise, the time windows of a request is also very specific to the application. For the rest of this work, this information will be derived from the available data and created, if required. For example, the traveling time for each link $T_{u,v}^a$ can be derived with the assumption of a constant speed. Trivially, since the distance is known, the travelling time can be calculated as follows:

$$T_{u,v}^a = \frac{d_{u,v}}{l_{u,v}} \tag{4.53}$$

where $l_{u,v}$ is the speed limit of the link $\langle u, v \rangle$.

- Analyse the following metrics
  - Road usage
  - Vehicle Charge at the end
  - Vehicle distribution at the end of the shift

1

---

[1]The simulation has been carried out using the CBC ([50]) and Gurabi ([21]) solvers used in combination with PuLP ([13]) on a 2018 Intel i7 MacbookPro.

Chapter 5

Model Predictive Control of AToD

This chapter proposes a solution for the dispatcing, routing and rebalancing problems for AToD systems in a different flavour, namely by designing a Model Predictive Control (MPC) to regulate the system. In Section 5.1, a novel model for the system is developed which, contrary to the previous one, is not graph based. Subsequentally, Section 5.2 defines the MPC problem and a thorough discussion on the terminal set and on the terminal cost, which are essential to prove stability. In order to achieve this goal, the feasability of the terminal and of the states set are demonstrated. Finally, in Section 5.3, a simulation is proposed to evluate the overall performance of the system.

5.1   Linear-Discrete Time Model

The model derived in this section is partially inspired from [63].
Consider a transportation network be composed of $|\mathcal{V}|$ stations scattered around a space Ω and $|\mathcal{A}|$ multi-occupancy, goods-carrying vehicles working within the same space Ω. Within this context, customers are assumed to requests rides only from the abovementioned stations. Similarly, goods can be carried out only from one station to another. With this assumption, stations and customers can be considered synonims within this section.
Similarly to the model described in Chapter 4.1, other than carrying goods or people, vehicle are expected to *(i)* serve clients from one station to another and *(ii)* reach locations in Ω in order to avoid system imbalance.
Let $d_{ij} \in \mathcal{R}_{\geq 0}$ be the total lenght of a road $\langle i, j \rangle$. Let $v_{ij}^a(t) \in \{0, 1\}$ indicate whether a transporting vehicle $a \in \mathcal{A}$ is moving from station i to station j $(v_{ij}^a(t) = 1)$ and, likewise, $w_{ij}^a \in \{0, 1\}$ whether an empty vehicle is moving from i to j $(w_{ij}^a(t) = 1)$. Let $V_{ij}(t) = \sum_{a \in \mathcal{A}} v_{ij}^a(t) + w_{ij}^a(t)$ being the total number of vehicles currently circulating on the street $\langle i, j \rangle$, it follows that $V_{ij}(t) \in \mathbb{N}_+$.
When a vehicle is in transit, it is essential to monitor the anticipated duration

until it reaches its destination. In order to do so, let's assume the road $\langle i, j \rangle$ to have a speed limit $l_{i,j}$. Accordingly, since the system is dealing with fully autonomous vehicles, in a typical driving scenario, one can safely assume this to be the cruising speed as well. In other words, one can assume the vehicles to be driving with a speed $l_{i,j}$ over $\langle i, j \rangle$ in normal road conditions. Motivated by safety concerns, the cruising speed cannot be consistently maintained at $l_{i,j}$ due to various factors. These factors may include road conditions, weather conditions, traffic density, or any unforeseen circumstances. Therefore, the actual cruising speed during the journey over road $\langle i, j \rangle$ may vary based on these dynamic elements, ensuring that the autonomous vehicles can adapt to changing conditions and prioritize safety over a fixed cruising speed. One factor which is directly controllable is the traffic density. Therefore, taking inspiration from the BPR model (Fig 3.1), we can approximate the cruising speed according to the amount of vehicles on the road. More specifically, we can modify Equation 3.2 to reflect this condition, and therfore rewriting it as

$$ s_{ij}(V_{ij}) = \begin{cases} l_{ij} & \text{if } V_{ij} \in [0, V_{ij}^{th}] \\ l_{ij} - b \cdot (V_{ij} - V_{ij}^{th}) & \text{if } V_{ij} \in [V_{ij}^{th}, V_{ij}^{max}] \\ 0 & \text{if } V_{ij} \geq V_{ij}^{max} \end{cases} \tag{5.1} $$

$$ \text{with } b = \frac{l_{ij}}{V_{ij}^{th} - V_{ij}^{max}} $$

An example can be seen in Fig 5.1. The last case, i.e. when $V_{ij} \geq V_{ij}^{max}$, is inspired from the congestion model used in Section 4.1 and in this case is modeling stale mate traffic. Intuitively, if too many cars are on the road, these will not be able to circulate with a high speed and eventually stop-and-go traffic will be created.

From the definition, it follows that s_{if} is defined as $s_{ij} : \mathbb{N}_+ \rightarrow \mathbb{R}_{\geq 0}$.

This allows to effectively track the position of the vehicle in terms of time as well. However, the position should be tracked only if the vehicle is currently driving on the street. Therefore, position of the vehicle a over $\langle i, j \rangle$ is propagated using Equation 5.2.

$$ x_{ij}^a(t + \delta) = \begin{cases} x_{ij}^a(t) + s_{ij}(V_{ij}(t)) \cdot \delta & \text{if } v_{ij}^a(t) + w_{ij}^a(t) = 1 \\ 0 & \text{if } v_{ij}^a(t) + w_{ij}^a(t) = 0 \\ 0 & \text{if } i = j \end{cases} \tag{5.2} $$

As mentioned above, Equation 5.2 tracks the position of a moving vehicle, i.e. takes care of the state of moving vehicles. In addition to this, the model also needs to track the number of vehicles currently stationed in i. This is achieved by introducing an additional variable $f_i^a(t)$, which indicates whether a vehicle arrived at a station i. It is defined as following.

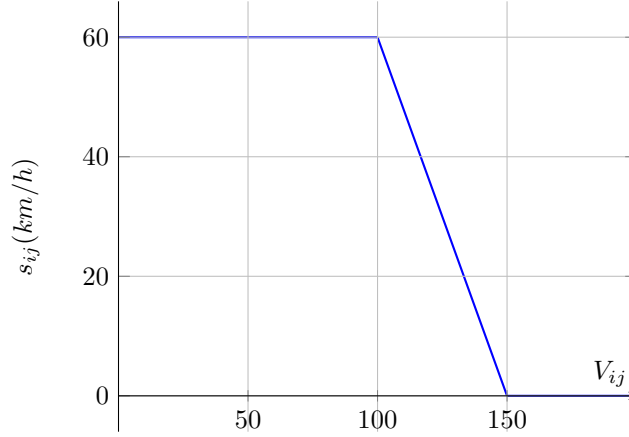$$ f_i^a(t) = \begin{cases} 1 & \text{if } x_{ji}^a(t) = d_{ji} \\ 0 & \text{otherwise} \end{cases} \tag{5.3} $$

Figure 5.1: Cruising speed in function of traffic density. In this example, $l_{ij} = 60km/h, V_{ij}^{th} = 100, V_{ij}^{max} = 150$.

This is propagated with the help of the indicator function.

$$f_i^a(t+1) = f_i^a(t) + 1_{d_{ji}=x_{ji}^a(t)} - \sum_{j\in\mathcal{V}}(v_{ij}^a + w_{ij}^a) \tag{5.4}$$

Equation 5.4 also ensures that a vehicle performs an action on the road $\langle i, j\rangle$ only when stationed at i.

The function 1_x is commonly known as the indicator function, denoting a Boolean variable x that can take values true, false. Specifically, 1_x is defined as follows:

$$1_x = \begin{cases} 1 & \text{if } x \text{ is true} \\ 0 & \text{if } x \text{ is false} \end{cases} \tag{5.5}$$

In other words, 1_x equals 1 when x is true and equals 0 when x is false, making it a convenient way to express the truth value of the variable x in mathematical notation.

Crucially, the vehicles can not perform the aforementioned actions, i.e. waiting, routing and rebalancing, at the same time and this must be insured. Furthermore, the vehicles can not perform the actions on multiple stations or roads. Therefore, the following constraint is necessary.

$$\sum_{i\in\mathcal{V}}(f_i^a(t+1) + \sum_{j\in\mathcal{V}}v_{ij}^a(t) + \sum_{j\in\mathcal{V}}w_{ij}^a(t)) = 1 \tag{5.6}$$

This also implies that, if a vehicle is stationed at a station i, it can not have a position anywhere else.

$$\sum_{i\in\mathcal{V}}(f_i^a(t) + 1_{x_{ji}^a(t)\neq0}) = 1 \tag{5.7}$$

In other words, if I vehicle is travelling through an edge $\langle ji \rangle$, it can only be stationed at i if it finished travelling.

While it is on the one hand interesting to know which station is currently hosting which vehicle, it is more important to know the amount of vehicles currently circulating in the system. This can be achieved by flipping the idea and therefore calculating how many vehicles out of the total number is currently not stationed.

$$F(t) = |\mathcal{A}| - \sum_{i \in \mathcal{N}} \sum_{a \in \mathcal{A}} f_i^a \tag{5.8}$$

Furthermore, one can also calculate the required time $T_{ij}(t)$ based on the speed approximation, as shown in Equation 5.9.

$$T_{ij}^a(t) = \begin{cases} \dfrac{d_{ij} - x_{ij}^a(t)}{s_{ij}(V_{ij}(t))} & \text{if } x_{ij}^a(t) > 0 \\ \\ 0 & \text{otherwise} \end{cases} \tag{5.9}$$

Denoted by $p_{ij}(t)$ and $g_{ij}(t)$ are the transportation and goods delivery requests respectively starting from station i and headed to j. Let $o_{ij}^p(t)$ and $o_{ij}^g(t)$ denoting the number of outstanding requests from i to j for people or goods, respectively, one can describe its propagation as follows:

$$\begin{aligned} o_{ij}^p(t+1) &= o_{ij}^p(t) + p_{ij}(t) - \sum_{a \in \mathcal{A}} P_a \cdot v_{ij}^a(t) \\ o_{ij}^g(t+1) &= o_{ij}^g(t) + g_{ij}(t) - \sum_{a \in \mathcal{A}} G_a \cdot v_{ij}^a(t) \end{aligned} \tag{5.10}$$

It follows then that vehicles can not transport more than what requested, i.e.:

$$\begin{aligned} \sum_{a \in \mathcal{A}} P_a \cdot v_{ij}^a(t) &\le o_{ij}^p(t) + p_{ij}(t) \\ \sum_{a \in \mathcal{A}} G_a \cdot v_{ij}^a(t) &\le o_{ij}^g(t) + g_{ij}(t) \end{aligned} \tag{5.11}$$

At this point, the definition of the control and the state of the system is complete. More specifically, the state of the system is described by the number outstanding requests $o_{ij}(t)$, the position of each moving vehicle $x_{ij}(t)$ and the position of each idle vehicle $f_i^a(t)$. Let the vector x(t) be the column vector column vector created by reshaping and concatenating $o_{ij}(t)$, $x_{ij}(t)$, $T_{ij}^a(t)$ and $f_i^a(t)$, the set of feasible states $\mathcal{X}$ is defined as follows.

$$
\mathcal{X} := \left\{ x = [o_{ij}^p, o_{ij}^g, x_{ij}^a, f_i^a, T_{ij}^a]^T \middle| \begin{array}{c} o_{ij}^p \in (\mathbb{N}_+)^{|\mathcal{V}|}, o_{ii}^p = 0, o_{ij}^g \in (\mathbb{N}_+)^{|\mathcal{V}|}, o_{ii}^g = 0 \\[2mm] f_i^a \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}, (5.7) \\[2mm] x_{ij}^a \in (\mathbb{R}_{\geq 0})^{|\mathcal{A}||\mathcal{V}|}, T_{ij}^a \in (\mathbb{R}_{\geq 0})^{|\mathcal{V}|} \end{array} \right\}
$$
(5.12)

Similarly, considering the control inputs $v_{ij}^a(t)$ and $w_{ij}^a(t)$, one can derive the set of feasible control set $\mathcal{U}(t)$ as follows.

$$
\mathcal{U}(t) := \left\{ u = [v_{ij}^a, w_{ij}^a]^T \middle| \begin{array}{c} v_{ij}^a \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}, v_{ii}^a = 0 \\[2mm] w_{ij}^a \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}, w_{ii}^a = 0 \\[2mm] (5.6),(5.11) \end{array} \right\}
$$
(5.13)

Since (5.6)-(5.11) depend on time, $\mathcal{U}(t)$ is also time-dependent.

Thanks to these formulations, the system can be written as a linear time-dependent system of the form

$$
x(t+1) = Ax(t) + Bu(t)
$$
(5.14)

where $x(t) \in \mathcal{X}$, $u(t) \in \mathcal{U}(t)$ and A and B are the matrix associated to the coefficients of (5.2), (5.4) and (5.10).

5.1.1   Objectives

Clearly, the main objective is to serve all the outstanding requests.

$$
J_1(x(t)) = \sum_{i,j \in \mathcal{V}} o_{ij}(t)
$$
(5.15)

However, the performance of the system can be optimized if other aspects are considered as well.

Since the speed of the vehicles, according to the model, is determined by the number of vehicles on the street, limiting it will improve performance. Vehicles serving requests are responsible for the main objective and their number is directly proportional to the number of outstanding requests. On the other hand, rebalancing vehicles, while help reducing the number of requests, they also contribute to congestions. Therefore, by optimizing the time they spend on the street, the speed of the transporting vehicles is optimized as a result.

$$
J_2(x(t)) = \sum_{i,j \in \mathcal{V}} T_{ij}^a w_{ij}^a(t)
$$
(5.16)

5.1.2   Model Evaluation

Some comments are in order. Requests are treated considering customers and goods as single entities. More specifically, if a group consisting of three customers reach a station, this is treated as three individual requests, therefore $p_{ij}(t) = 3$. While this simplifies the model description and reflects the requirements for goods delivery, it would create an inconvenient scenario for traveling customers. As a matter of fact, it is not hard to envision situations where a group of travelers would like to travel together. However, it is out of the scope of this work to treat this scenario. One might also argue that this simplification effectively treats the model as if vehicles where single-occupancy and it is true in terms of intuition, this addition allows to reduce the number of variables by a constant factor, i.e. the sum of all vehicles' capacity.

The system described in Equation 5.14 implicitly assumes the customer arrival to be known a priori. In order words, this is not treated as noise, but rather as a known quantity.

$$x(t + 1) = Ax(t) + Bu(t) + w(t) \tag{5.17}$$

where $x(t) \in \mathcal{X}$, $u(t) \in \mathcal{U}(t)$, $w(t) = [p_{ij}(t) \quad g_{ij}(t) \quad 0 \quad 0 \quad 0]^T$ and A and B are the matrix associated to the coefficients of (5.2), (5.3) and (5.10).

Equation 5.17 treats customer arrival as noise, or disturbance, which makes reasoning on the system considerably harder. Several techniques ([7], [30]) have been proposed to deal with this type of MPC systems, i.e. Robust MPC. Alternatively, some work propose solutions to estimate this quantity using Deep Learning techniques ([4], [10]). However, the analysis of those techniques and their performance in this scenario is out of the scope of this work. Similarly to Section 4.1, the model must take into account that stations do not have unlimited amoun t of parking spots. As a consequence, the number of vehicles stationed must be limited accordingly and this is achieved as follows.

$$\sum_{a \in \mathcal{A}} f_i^a(t) \leq C_i \tag{5.18}$$

where C_i indicates the number of parking spots available at i.
This is a crucial

5.2   Problem Formulation

The MPC for the AToD problem is formulated as follows.
Given $x(t) \in \mathcal{X}$, determine the controls $u(t), \ldots, u(t + N)$ according to the

following optimization problem.

$$\min_{u(t),\dots,u(t+N)} \quad J_f(x(N)) + \sum_{t=0}^{N-1} I(x(t))$$
$$\textbf{s.t.} \quad x(t+1) = Ax(t) + Bu(t) \tag{5.19}$$
$$x(t) \in \mathcal{X}, \ u(t) \in \mathcal{U}$$
$$x(N) \in \mathcal{X}_f$$

where $J_f(x(t+N))$ is the terminal cost function and $\mathcal{X}_f$ is the terminal set. As the main goal is to prove the stability of (5.19), a proper definition of those will facilitate this objective. The strategy used to prove stability is the one described in Section 2.2.1. For this purpose, the terminal set is required to be defined around an equilibrium point x_*. A good canidate for the equilibrium point in such systems can be found by observing that the system remains at a equilibrium whenever no more requests arrive and the are no more outstanding requests within the system. In other words, when

$$p_{ij}(t) = 0$$
$$g_{ij}(t) = 0 \qquad \forall i, j \in \mathcal{V}$$
$$o_{ij}^p(t) = 0$$
$$o_{ij}^g(t) = 0$$

By Equation 5.10, this implies that, eventually, no more vehicles will transport goods or people anymore. Furthermore, one can also conclude that vehicles will also stop rebalancing themselves. Therefore, o_{ij}, x_{ij}^a and T_{ij}^a all assume value zero. However, there is no way of knowing exactly where vehicles are going to be stationed. Upon inspection, however, one can canclude that vehicles must be stationed somewhere, since they are not driving. In other words, f_i^a could assume any value in $\{0,1\}^{|\mathcal{A}||\mathcal{V}|}$ except $\{0\}^{|\mathcal{A}||\mathcal{V}|}$. Furthermore, at equilibrium, all the vehicles are stationed. That means, the set of possible values for f_i^a is further limited accordingly. To define this set, let's consider a function $n :$ $\mathcal{D}_{f_{ij}^a} \to N$, where $\mathcal{D}'_{f_{ij}^a} = \{0,1\}^{|\mathcal{A}||\mathcal{V}|} \setminus \{0\}^{|\mathcal{A}||\mathcal{V}|}$, which sums the number of 1s in the set. With this addition, we can fully define the set of values of f_i^a at equilibrium.

$$\mathcal{D}_{f_{ij}^a} := \left\{ x \quad | \quad x \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}, n(x) = |\mathcal{A}| \right\} \tag{5.20}$$

While the set is indeed smaller, it is still quite hard to exactly pin-point a specific equilibrium point, as already mentioned above. The difficulty arises because, out of all the elements in (5.20), one can not directly identify the one which describes the system fully without knowing how the system progressed in time. However, one can conclude that the equilibrium points are all elements of the

set described in Equation 5.21.

$$\mathcal{E} := \left\{ e = [o_{ij}^p, o_{ij}^g, x_{ij}^a, f_i^a, T_{ij}^a]^T \middle| \begin{array}{l} o_{ij}^p \in \{0\}^{|\mathcal{V}||\mathcal{A}|}, o_{ij}^g \in \{0\}^{|\mathcal{V}||\mathcal{A}|} \\ \\ f_i^a \in \mathcal{D}_{f_{ij}^a} \\ \\ x_{ij}^a \in \{0\}^{|\mathcal{V}||\mathcal{A}|}, T_{ij}^a \in \{0\}^{|\mathcal{V}||\mathcal{A}|} \end{array} \right\} \tag{5.21}$$

While this could be considered as a candidate for the terminal set $\mathcal{X}_f$, it turns out to be too restrictive. This is due to the fact that it would require all vehicles to be stationed and, therefore, not moving. Some of the conditions, however, can be relaxed by considering some implicit assumptions made during the development of the model. Mainly, vehicles are assumed to be perfect and to never break, therefore, one can consider a request to be satisfied the moment it has been picked up. As a result, while the number of requests is still zero, the conditions for x_{ij}^a, f_i^a and T_{ij}^a can be relaxed.

$$\mathcal{X}_f := \left\{ x_f = [o_{ij}^p, o_{ij}^g, x_{ij}^a, f_i^a, T_{ij}^a]^T \middle| \begin{array}{l} o_{ij}^p \in \{0\}^{|\mathcal{V}||\mathcal{A}|}, o_{ij}^g \in \{0\}^{|\mathcal{V}||\mathcal{A}|} \\ \\ f_i^a \in \{0,1\}^{|\mathcal{V}||\mathcal{A}|}, (5.7) \\ \\ x_{ij}^a \in (\mathbb{R}_{\geq 0})^{|\mathcal{A}||\mathcal{V}|}, T_{ij}^a \in (\mathbb{R}_{\geq 0})^{|\mathcal{V}|} \end{array} \right\} \tag{5.22}$$

By definition, therefore, $\mathcal{E} \subset \mathcal{X}_f \subset \mathcal{X}$.

In addition, thanks to this definition, another desirable conditions apply. Since $x_{ij}^a \in (\mathbb{R}_{\geq 0})^{|\mathcal{A}||\mathcal{V}|}$ and $T_{ij}^a \in (\mathbb{R}_{\geq 0})^{|\mathcal{V}|}$, empty vehicles can also be navigating, which allows rebalancing, too. Moreover, since vehicles are not required to be stationed and the minimum requirement for a request to be satisfied is to be picked up, as long as the capacity of the vehicle stationed in i is enough to cover the request, than the number of outstanding requests is still zero.

$$\begin{aligned} \sum_{a \in \mathcal{A}} P_a \cdot f_i^a(t) &\geq p_{ij}(t) \\ \sum_{a \in \mathcal{A}} G_a \cdot f_i^a(t) &\geq g_{ij}(t) \qquad \forall i, j \in \mathcal{V} \\ o_{ij}^p(t) &= 0 \\ o_{ij}^g(t) &= 0 \end{aligned} \tag{5.23}$$

This further relaxation also allows to better reason about the control law κ_f required to prove stability of the controller. The two main requirements are (i) $\kappa_f \in \mathcal{U}$ and (ii) the set $\mathcal{X}_f$ remains feasible, since they are among the main assumption for the controller stability (Section 2.2.1). Furthermore, at each time, it is required that the capacity of the vehicle leaving a station is equal to

the new demand, as the number of outstanding requests is zero.

$$\sum_{a \in \mathcal{A}} P_a \cdot v_{ij}^a(t) = o_{ij}^p(t)$$
$$\sum_{a \in \mathcal{A}} G_a \cdot v_{ij}^a(t) = o_{ij}^g(t)$$

(5.24)

Vehicles, however, can only leave a station if they are present at that station. On the same note, since there must be enough vehicles at every station to serve the requests, those must be rebalanced in such a way that, once a request arrives, a vehicle is ready to serve it.

The control law that must be designed can be seen as a function mapping $\mathcal{X}_f$ to a subset of $\mathcal{U}$, i.e. $\kappa_f : \mathcal{X}_f \to \mathcal{U}_{\kappa_f} \subseteq \mathcal{U}$. This function must express then need to "conserve" the relation in Equation 5.23, which has the by product of also satisfying Equation 5.11.

Furthermore, as a vehicle leaves a station, another vehicle must take its place. Therefore, the follow relation must also hold.

$$P_a \cdot v_{ij}^a \leq \sum_{j \in \mathcal{V}} \sum_{a' \in \mathcal{A} \setminus \{a\}} P_{a'} \cdot w_{ji}^{a'}$$
$$G_a \cdot v_{ij}^a \leq \sum_{j \in \mathcal{V}} \sum_{a' \in \mathcal{A} \setminus \{a\}} G_{a'} \cdot w_{ji}^{a'}$$

(5.25)

Equation 5.25 is used to ensure that, if a vehicle leaves a station, it must be replaced by one or more vehicles with at least the same capacity.

$$\mathcal{U}_{\kappa_f}(t) := \left\{ u = [v_{ij}^a, w_{ij}^a]^T \middle| \begin{array}{c} v_{ij}^a \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}, v_{ii}^a = 0 \\ w_{ij}^a \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}, w_{ii}^a = 0 \\ (5.6),(5.24),(5.25) \end{array} \right\}$$

(5.26)

It is trivial to demonstrate that $\mathcal{U}_{\kappa_f}(t) \subseteq \mathcal{U}$. By construction Equation 5.6 and Equation 5.11 are satisfied (the latter specifically from Equation 5.24).

A remark is required. Equation 5.24 and Equation 5.25 indeed allow vehicles to be replaced and, therefore, potentially take care of the requests. However, in addition to the assumption made for Equation 5.24, the state of the system will remain in $\mathcal{X}_f$ only as long as the requests are made within a time interval corresponding to the time required to travel from the furthest station to the starting station. In other words, this is a very particular case of extrogenous requests. Therefore, for the rest of this section, the external arriving requests will be considered as zero, i.e. $o_{ij}^p(t) = o_{ij}^g(t) = 0$, therefore treating the system as undisturbed.

This assumption also allows to derive an appropriate cost function for the system. As indicated in Section 2.2.1, the candidate terminal cost must be a Lyapunov function (see Section 2.1 for more details).

Within the assumptions made, one candidate for the terminal cost function can be found by observing the vehicles that are still driving around the system. Intuitively, one can consider the time that the vehicles will spend on the road. At first glance, according to the definition found in Equation 5.9, this would be difficult to prove to be Lyapunov. As a matter of fact, the speed of the vehicles depends on the amount of vehicles on the road, which would mean the function is not guaranteed to strictly decrease. In this situation, however, since we are dealing with an undisturbed system near its equilibrium, new vehicles will not be put it motion. In other words, at time t, if there are $\sum_{i,j \in \mathcal{V}} V_{ij}(t)$ vehicles in the whole system, this number will only decrease as the system progresses, since eventually vehicles will become stationed. As a consequence, the speed of the vehicles will also improve and, therefore, the time will decrese.

Therefore, the final const function $J(x) : \mathcal{X}_f \to R_+$ can be constructed in the following way.

$$J_f(x(N)) := \sum_{i,j \in \mathcal{V}} \sum_{a \in \mathcal{A}} T_{ij}^a \tag{5.27}$$

Proposition 1: *Within the definition of $\mathcal{X}_f$, (5.27) is a Lyapunov Function in $\mathcal{X}_f$*

Proof. Three conditions must be met.

1. The function must be strictly positive, except at zero, i.e.

$$\sum_{i,j \in \mathcal{V}} \sum_{a \in \mathcal{A}} T_{ij}^a > 0$$

This is indeed true by definition of T_{ij}^a. More specifically, when the system is not at zero, then there are vehicles moving ($\sum_{i \in \mathcal{V}} \sum_{j \in \mathcal{V}} \sum_{a \in \mathcal{A}} w_{ij}^a > 0$). Then, at least one vehicle is moving. As a result, $x_{ij}^a > 0$ and consequently $T_{ij}^a > 0$.

2. Secondly, the function must be assume the value of zero at equilibrium. ln other words, given any point $x_{\mathcal{E}} \in \mathcal{E}$

$$J_f(x_{\mathcal{E}}) = 0$$

At equilibrium, there are no vehicles moving. Clearly, the terminal cost function is zero.

3. J_f must decrease $\forall x \in \mathcal{X}_f$.

$$J(x(k+1)) - J(x(k)) \leq 0$$

Let's consider a vehicle a. If the vehicle is not rebalancing, it can be considered at equilibrium, hence $T_{ij}^a = 0$. The sum of the timings for

57

these vehicles do not contribute to J_f. More specifically, their sum is equal to 0.

If the vehicle is moving, since it can not move backwards, as time increases, by propagation of x_{ij}^a, its position always increases untile $x_{ij}^a = d_{ij}$. As x_{ij}^a increases, T_{ij}^a decreases, since the speed of the vehicles can not decrease, as discussed above. If $x_{ij}^a = d_{ij}$, then $T_{ij}^a = 0$, falling in the scenario above.

5.2.1 Proof of Stability

For the rest of this section, the notation will be eased as follows.

$$x = x(t)$$
$$u = u(t)$$
$$x^+ = x(t+1)$$

In order to prove stability of (5.19), it is imperative to first demonstrate the feasibility of the state sets $\mathcal{X}$ adn $\mathcal{X}_f$.

Proposition 2: *(Feasibility of $\mathcal{X}$) Let $x \in \mathcal{X}$ and $u \in \mathcal{U}(t)$, then $x_+ \in \mathcal{X}$*

Proof. Let $x^+ = [o_{ij}^+, x_{ij}^{a+}, f_i^{a+}, T_{ij}^{a+}]^T$ and $u = [v_{ij}^a, w_{ij}^a]^T$. Since $u \in \mathcal{U}$, then Equation 5.11 is satisfied, therefore $o_{ij}^+ \in (\mathbb{N}^+)^{|\mathcal{V}|}$. By assumption, $o_{ii}^+ = 0$ is also satisfied.

The condition $f_i^{a+} \in \{0,1\}^{|\mathcal{A}||\mathcal{V}|}$ can be proven with the help of Equation 5.4. At first vehicles are either present at a station i or not. If vehicles are not stationed, then no action can be taken due to Equation 5.6 (if $u \in \mathcal{U}$, then it is satisfied). Otherwise, if vehicles are indeed stationed, then $f_i^a = 1$. If a vehicle moves, i.e. if $w_{ij}^a(t) = 1$ or $v_{ij}^a(t) = 1$, because of the first condition of Equation 5.2,then as of Equation 5.4, then, at the next step $f_i^{a+} = 0$. In this scenario, at the next time step, the first condition applies. Furthermore, Equation 5.7 is satisfied, since the vehicle is traveling towards the station. If, on the other hand, the vehicle is approaching the station, i.e. $f_i^a = 0$ and $w_{ji}^a(t) = 1$, then $f_i^{a+} = 1$. Equation 5.7 is still satisfied because of Equation 5.6, as the latter considers all the transportation network and, therefore $x_{ij}^{a+} = 0$ as a result of w_{ij}^{a+} or (v_{ij}^{a+}) being set to 0.

To prove $x_{ij}^{a+} \in (\mathbb{R}_{\geq 0})^{|\mathcal{A}||\mathcal{V}|}$, one can use a similar argument by observing the defintion of the propagation of x_{ij}^a in Equation 5.2. In case $v_{ij}^a(t) + w_{ij}^a(t) = 0$ or $i = j$, the condition is satisfied, since $x_{ij}^{a+} = 0$. On the other hand, in case of $v_{ij}^a(t) + w_{ij}^a(t) = 1$, the condition is satisfied by definition of s_{ij} and $V_{ij}(t)$.

Finally, $T_{ij}^{a+} \in \mathbb{R}_{\geq 0}$ can be proven as a result of the discussion made previously.

It is necessary to prove the following.

$$\frac{d_{ij} - x_{ij}^a(t)}{s_{ij}(V_{ij}(t))} \geq 0$$

$$d_{ij} - x_{ij}^a(t) \geq 0$$

$$d_{ij} \geq x_{ij}^a(t)$$

Since the variable x_{ij}^a tracks the position of the vehicle on the road, this can not be bigger than the road itself. Furthermore, due to Equation 5.6, this is also salvaguarded, as the vehicles becomes stationed if it reaches the end of the road.

Proposition 3: *(Feasibility of $\mathcal{X}_f$) Let $x \in \mathcal{X}_f$ and $\kappa_f(x) \in \mathcal{U}_{\kappa_f}(t)$, then $x_+ \in \mathcal{X}_f$*

Proof. Let $x^+ = [o_{ij}^+, x_{ij}^{a+}, f_i^{a+}, T_{ij}^{a+}]^T$ and $\kappa_f(x) = [v_{ij}^a, w_{ij}^a]^T$. By assumption, $o_{ii}^+ = 0$ is satisfied. Furthermore, since the system is treated as undisturbed, $o_{ij}^+ = 0$. Because $\kappa_f(x) \in \mathcal{U}_{\kappa_f}(t)$, then Equation 5.24 is satisfied, implieng there is no vehicle leaving the station to serve a request.
If $w_{ij}^a = 0$, then nothing happens within the system, therefore all the vehicles remain at the station, then $f_i^{a+} \in \{0,1\}^{|\mathcal{V}||\mathcal{A}|} \; \forall i \in \mathcal{V}$. Consequently $x_{ij}^{a+} = 0$, $T_{ij}^{a+} = 0$, i.e. both belonging to $\mathbb{R}_{\geq 0}$.
If vehicles are in movement, i.e. $w_{ij}^a = 1$, a similar argument to the one proposed for the feasibility of $\mathcal{X}$ can be made. Because of Equation 5.4, $f_i^{a+} = 0$ and $x_{ij}^{a+} \in (\mathbb{R}_{\geq 0})^{|\mathcal{A}||\mathcal{V}|}$, by definition of s_{ij} and $V_{ij}(t)$. Similarly, the same reason applies for $T_{ij}^{a+} \in \mathbb{R}_{\geq 0}$. Eventually, as the vehicle approached the station, i.e. $d_{ji} = x_{ji}^a(t)$, then $f_i^{a+} = 1$ due to Equation 5.4. Since Equation 5.6 is respected ($\kappa_f(x) \in \mathcal{U}_{\kappa_f}(t)$), then (5.7) is also respected, as $w_{ij}^{a+} = 0 \implies x_{ij}^{a+} = 0$.
Condition (5.25) is always respected, assuming there is no new requests.

Model stability

Proposition 4: *(Stability of 5.19)*
Given $\mathcal{X}_f$ and J_f defined in (5.22) and (5.27), respectively, and let $\kappa_f : \mathcal{X} \to \mathcal{U}_{\kappa_f}$, then the controller defined in (5.19) is stable in the sense of Lyapunov.

Proof. The proof is divided in two parts. In the first part, the recursive feasibility of the controller is proved. Secondly, the stability is proven by showing than the optimal cost function J^* is a Lyapunov function.

1. As stated by Proposition 2, $\mathcal{X}$ is feasible. Let $x \in \mathcal{X}$ and $[u_0^*, u_1^*, \ldots, u_{N-1}^*]$ be an optimal control sequence calculated at x. At x^+ the control sequence $[u_0^*, u_1^*, \ldots, \kappa_f(x_N^*)]$ is feasible. This is because $x_N \in \mathcal{X}_f$ and, therefore,

$x_+ = Ax^* + B\kappa_f(x_N^*) \in \mathcal{X}_f$, since $\mathcal{X}_f$ is feasible, as proven in Proposition 3.

This proves recursive feasibility.

2. Given the optimal cost function

$$J^*(k) = J_f(x_N^*) + \sum_{i=0}^{N-1} I(x_i^*, u_i^*)$$

At $x(k+1) = x_1^*$, the following needs to be shown

$$J^*(k+1) \leq \widetilde{J}(k)$$

where $\widetilde{J}(k)$ is the candidate function and calculated at $\widetilde{U} = \{u_1^*, u_2^*, \ldots, \kappa_f(x_N^*)\}$. Therefore

$$J^*(k+1) \leq \sum_{i=1}^{N-1} I(x_i^*, u_i^*) + I(x_N^*, \kappa_f(x_N^*)) + J_f(Ax_N^* + B\kappa_f(x_N^*))$$

$$J^*(k+1) \leq \sum_{i=1}^{N-1} I(x_i^*, u_i^*) + I(x_0^*, u_0^*) - I(x_0^*, u_0^*) + J(x_N^*, \kappa_f(x_N^*)) + J_f(Ax_N^* + B\kappa_f(x_N^*))$$

$$J^*(k+1) \leq \sum_{i=0}^{N-1} I(x_i^*, u_i^*) - I(x_0^*, u_0^*) + I(x_N^*, \kappa_f(x_N^*)) + J_f(Ax_N^* + B\kappa_f(x_N^*))$$

$$\text{Since } J^*(k) = J_f(x_N^*) + \sum_{i=0}^{N-1} I(x_i^*, u_i^*)$$

$$J^*(k+1) \leq J^*(k) - I(x(k), u_0^*) + \underbrace{J_f(Ax_N^* + B\kappa_f(x_N^*)) + I(x_N^*, \kappa_f(x_N^*)) - J_f(x_N^*)}_{\leq 0 \text{ because } J_f \text{ is a Lyapunov function}}$$

$$\implies J^*(k+1) - J^*(k) \leq -I(x(k), u_0^*) \quad I(x, u) > 0 \text{ for } x, u \neq 0$$

As a result, the optimal cost is a Lyapunov function. Hence, the system is asymptotically stable.

5.3 Use Case/Simulation / Evaluation

Chapter 6

Part 2 - Graph Transformation Theory

- Use graph theory to automatically add nodes

- use graph grammar to explore the design space

What we can do with GTs is analyzing the behaviour of the cars to, on the other hand, control the infastructure. That means, traffic lights depending on vehicles. PRoving there will be co congestions. More specifically, you can also start from the nodes and build a graph based on the requests. So you are building the infrastracture from the requests.

In this regards, if for example you have 4 nodes like that
A-B
—X—
C-D

I need to put a traffic light in the middle, which is controlled by the cars and the amount. So basically it gets green or red depending on the vehicles on the roads. Furthermore, you can also move based on the pedestrians. Also, where to plae charging stations and how to operate them.

Chapter 7

Summary and Outlook

In this work, we only considered one solution to solve the dispatching problem. While the solution found is guaranteed to be optimal, other algorithms could be used as well. For example, one possible approach is to develop an ad-hoc algorithm for it. For instance, requests could be assigned greedily to vehicles according to some heuristics, such as nearest neighbours, or according to the vehicle status, i.e. state of charge, capacity or traveling path. Requests should be assigned in such a way that are still in line with the previous path (see Fig 4.2). One could, for example, consider as input the previously calculated path for it.

In Chapter 5, ride-sharing is not considered. It should be studied its effect on the performance and one direction to persue is to refine the model to account for it. For example, one should also keep track of the vechile capacity as the system progresses, in terms of goods and travelers.

You can use GTS also for other purposes. For example, one interesitng idea is to use it to basically automate the design and reconfiguration of vehicles based on the requests. Say you can estimate the requests and what is required from them. For example, if a place is a hospital, you need a vehicle that is able to (i) stop all traffic because it has the priority, (ii) carry people tjat are dyomg and (iii) host the doctors on board as well. In this way, you can basically explore the "design space" of vehicles configurations. At the same time, you can do that with the infrastructure creation / modification. Say you have a roads and possibilities, you want to update traffic lights with the apposite transmission equipment to stop traffic, i.e. V2X.

Bibliography

[1] The dynamic vehicle allocation problem with application in trucking companies in brazil. *Computers and Operations Research*, 76:118–133, 2016.

[2] Ahmed Abdulaal, Mehmet H. Cintuglu, Shihab Asfour, and Osama A. Mohammed. Solving the multivariant ev routing problem incorporating v2g and g2v options. *IEEE Transactions on Transportation Electrification*, 3(1):238–248, 2017.

[3] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467, 2017.

[4] Bahman Askari, Tai Le Quy, and Eirini Ntoutsi. Taxi demand prediction using an lstm-based deep sequence model and points of interest. In *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 1719–1724, 2020.

[5] J. Beck, Patrick Prosser, and E. Selensky. Graph transformations for the vehicle routing and job shop scheduling problems. volume 2502, pages 60–74, 10 2002.

[6] Luca Brodo. Autonomous mobility-on-demand: A novel framework to study amod in smart cities. Paper not yet submitted for publication, 2023.

[7] Peter J. Campo and Manfred Morari. Robust model predictive control. *1987 American Control Conference*, pages 1021–1026, 1987.

[8] Carlos Carrion and David Levinson. Value of travel time reliability: a review of current evidence. *Transportation Research Part A: Policy and Practice*, 46:720–741, 2012.

[9] Andrea Carron, Francesco Seccamonte, Claudio Ruch, Emilio Frazzoli, and Melanie N. Zeilinger. Scalable model predictive control for autonomous mobility-on-demand systems. *IEEE Transactions on Control Systems Technology*, 29(2):635–644, 2021.

[10] Kai Fung Chu, Albert Y.S. Lam, and Victor O.K. Li. Travel demand prediction using deep multi-scale convolutional lstm network. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1402–1407, 2018.

[11] Juan de Lara, Hans Vangheluwe, and Pieter J. Mosterman. Modelling and analysis of traffic networks based on graph transformation. 2005.

[12] B. Donovan and D. B. Work. New york city taxi trip data (2010-2013). http://dx.doi.org/10.13012/J8PN93H8, 2014.

[13] Iain Dunning, Stuart Mitchell, and Michael O'Sullivan. Pulp: A linear programming toolkit for python. *Department of Engineering Science, The University of Auckland*, September 2011.

[14] Fabian Fehn, Florian Noack, and Fritz Busch. Modeling of mobility on-demand fleet operations based on dynamic electricity pricing. In *2019 6th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, pages 1–6, 2019.

[15] Aurélien Froger, Ola Jabali, Jorge E. Mendoza, and Gilbert Laporte. The electric vehicle routing problem with capacitated charging stations. *Transportation Science*, 56(2):460–482, 2022.

[16] M. Fürst, J. Götz, M. Otto, et al. Automation of gearbox design. *Forsch Ingenieurwes*, 86:409–420, 2022.

[17] German Aerospace Center (DLR). Nemo Bili - Mobility from the First to the Last Mile, Year of access. Accessed on: 04.01.2024.

[18] J Gross and S Rudolph. Geometry and simulation modeling in design languages. *Aerospace Science and Technology*, 54:183–191, 2016.

[19] J Gross and S Rudolph. Modeling graph-based satellite design languages. *Aerospace Science and Technology*, 49:63–72, 2016.

[20] J Gross and S Rudolph. Rule-based spacecraft design space exploration and sensitivity analysis. *Aerospace Science and Technology*, 59:162–171, 2016.

[21] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2023.

[22] Kyoungseok Han, Tam W. Nguyen, and Kanghyun Nam. Battery energy management of autonomous electric vehicles using computationally inexpensive model predictive control. *Electronics*, 9(8), 2020.

[23] Reiko Heckel. Graph transformation in a nutshell. *Electronic Notes in Theoretical Computer Science*, 148(1):187–198, 2006. Proceedings of the School of SegraVis Research Training Network on Foundations of Visual Modelling Techniques (FoVMT 2004).

[24] Deborah A. Hennessy and Donald L. Wiesenthal. Traffic congestion, driver stress, and driver aggression. *Aggressive Behavior*, 25:409–423, 1999.

[25] Andreas Holzapfel, Heinrich Kuhn, and Michael G. Sternbeck. Product allocation to different types of distribution center in retail logistics networks. *European Journal of Operational Research*, 264(3):948–966, 2018.

[26] M. Hyland and H. S. Mahmassani. Dynamic autonomous vehicle fleet operations: optimization-based strategies to assign avs to immediate traveler demand requests. *Transportation Research Part C: Emerging Technologies*, 92:278–297, 2018.

[27] Brian Kallehauge, Jesper Larsen, Oli Madsen, and Marius Solomon. *Vehicle Routing Problem with Time Windows*, pages 67–98. 03 2006.

[28] S. R. Kancharla and G. Ramadurai. Electric vehicle routing problem with non-linear charging and load-dependent discharging. *Expert Systems With Applications*, 160:113714, 2020.

[29] Hans-Joerg Kreowski, Renate Klempien-Hinrichs, and Sabine Kuske. Some essentials of graph transformation. *Studies in Computational Intelligence*, 25, 01 2006.

[30] W. Langson, I. Chryssochoos, S.V. Raković, and D.Q. Mayne. Robust model predictive control using tubes. *Automatica*, 40(1):125–133, 2004.

[31] G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43:408–416, 2009.

[32] Chungmok Lee. An exact algorithm for the electric-vehicle routing problem with nonlinear charging time. *Journal of the Operational Research Society*, 72:1–24, 03 2020.

[33] Michael W. Levin, Kara M. Kockelman, Stephen D. Boyles, and Tianxin Li. A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application. *Computers, Environment and Urban Systems*, 64:373–383, 2017.

[34] C Robin Lindsey and Erik T Verhoef. Congestion modelling. Technical report, Tinbergen Institute Discussion Paper, 1999.

[35] Haining Liu, Ijaz Haider Naqvi, Fajia Li, Chengliang Liu, Neda Shafiei, Yulong Li, and Michael Pecht. An analytical model for the cc-cv charge of li-ion batteries with application to degradation analysis. *Journal of Energy Storage*, 29:101342, 2020.

[36] Alejandro Montoya, Christelle Guéret, Jorge E. Mendoza, and Juan G. Villegas. The electric vehicle routing problem with nonlinear charging function. *Transportation Research Part B: Methodological*, 103:87–110, 2017. Green Urban Transportation.

[37] NetworkX Development Team. networkx.algorithms.tree.mst.minimum_spanning_edges. NetworkX 2.6.2 documentation, 2021. Retrieved 2024-02-11.

[38] Heinz Neuburger. The economics of heavily congested roads. *Transportation Research*, 5(4):283–293, 1971.

[39] Zi-Hao Nie, Qiang Yang, En Zhang, Dong Liu, and Jun Zhang. Ant colony optimization for electric vehicle routing problem with capacity and charging time constraints. In *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 480–485, 2022.

[40] Bureau of Public Roads. Traffic assignment manual. Technical report, U.S. Dept. of Commerce, Urban Planning Division, 1964.

[41] P. Pant and R. M. Harrison. Estimation of the contribution of road traffic emissions to particulate matter concentrations from field measurements: a review. *Atmospheric Environment*, 77:78–97, 2013.

[42] Marco Pavone, Stephen L Smith, Emilio Frazzoli, and Daniela Rus. Robotic load balancing for mobility-on-demand systems. *The International Journal of Robotics Research*, 31(7):839–854, 2012.

[43] Mark P.H. Raadsen, Michiel C.J. Bliemer, and Michael G.H. Bell. Introducing pattern graph rewriting in novel spatial aggregation procedures for a class of traffic assignment models. 2016-12-01.

[44] James B. Rawlings, David Q. Mayne, and Moritz M. Diehl. *Model Predictive Control: Theory, Computation, and Design.* Nob Hill Pub., 2020.

[45] Gabriele Taentzer Reiko Heckel. Graph transformation for software engineers. 01 2020.

[46] Federico Rossi, Rick g, and Marco Pavone. Autonomous vehicle routing in congested road networks. *CoRR*, abs/1603.00939, 2016.

[47] Grzegorz Rozenberg. Handbook of graph grammars and computing by graph transformation. 01 1997.

[48] Mauro Salazar, Matthew Tsao, Izabel Aguiar, Maximilian Schiffer, and Marco Pavone. A congestion-aware routing scheme for autonomous mobility-on-demand systems. pages 3040–3046, 06 2019.

[49] David Schrank, Bill Eisele, and Tim Lomax. Texas transportation institute 2012 urban mobility report. Technical report, Texas Transportation Institute, Texas A&M University, College Station, TX, 2012.

[50] Dirk Schumacher, Jan Ooms, Bakhtiyar Yapparov, and John O. Hanson. *rcbc: COIN CBC MILP Solver Bindings*, 2022. R package version 0.1.0.9001.

[51] Stephen L. Smith, Marco Pavone, Mac Schwager, Emilio Frazzoli, and Daniela Rus. Rebalancing the rebalancers: optimally routing vehicles and drivers in mobility-on-demand systems. In *2013 American Control Conference*, pages 2362–2367, 2013.

[52] P. Toth and D. Vigo. *Vehicle Routing: Problems, Methods, and Applications*. Society for Industrial and Applied Mathematics, Philadelphia, 2014.

[53] Paolo Toth and Daniele Vigo. *The Vehicle Routing Problem*. Society for Industrial and Applied Mathematics, 2002.

[54] Erik T Verhoef. Time, speeds, flows and densities in static models of road traffic congestion and congestion pricing. *Regional Science and Urban Economics*, 29(3):341–369, 1999.

[55] C Voss, F Petzold, and S Rudolph. Graph transformation in engineering design: an overview of the last decade. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 37(e5):1–17, 2023.

[56] Alex Wallar, Menno Van Der Zee, Javier Alonso-Mora, and Daniela Rus. Vehicle rebalancing for mobility-on-demand systems with ride-sharing. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4539–4546, 2018.

[57] Salomón Wollenstein-Betech, Arian Houshmand, Mauro Salazar, Marco Pavone, Christos G. Cassandras, and Ioannis Ch. Paschalidis. Congestion-aware routing and rebalancing of autonomous mobility-on-demand systems in mixed traffic. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7, 2020.

[58] Salomón Wollenstein-Betech, Ioannis Ch. Paschalidis, and Christos G. Cassandras. Joint pricing and rebalancing of autonomous mobility-on-demand systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 2573–2578, 2020.

[59] James J. Q. Yu and Albert Y. S. Lam. Autonomous vehicle logistic system: Joint routing and charging strategy. *IEEE Transactions on Intelligent Transportation Systems*, 19(7):2175–2187, 2018.

[60] James J. Q. Yu, Wen Yu, and Jiatao Gu. Online vehicle routing with neural combinatorial optimization and deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3806–3817, 2019.

[61] Gioele Zardini, Nicolas Lanzetti, Marco Pavone, and Emilio Frazzoli. Analysis and control of autonomous mobility-on-demand systems. *Annual Review of Control, Robotics, and Autonomous Systems*, 5(1):633–658, 2022.

[62] Jannik Zgraggen, Matthew Tsao, Mauro Salazar, Maximilian Schiffer, and Marco Pavone. A model predictive control scheme for intermodal autonomous mobility-on-demand. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1953–1960, 2019.

[63] Rick Zhang. *MODELS AND LARGE-SCALE COORDINATION ALGO-RITHMS FOR AUTONOMOUS MOBILITY-ON-DEMAND*. Ph.d. thesis, Stanford University, Department of Aeronautics and Astronautics, 2016. Submitted to the Department of Aeronautics and Astronautics.

[64] Rick Zhang and Marco Pavone. A queueing-network approach to the analysis and control of mobility-on-demand systems. *Proceedings of the American Control Conference*, 2015, 09 2014.

[65] Allan Zhao, Jie Xu, Mina Konaković Luković, Josephine Hughes, Andrew Speilberg, Daniela Rus, and Wojciech Matusik. Robogrammar: Graph grammar for terrain-optimized robot design. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020.

[66] Pengbo Zhu, Isik Sirmatel, Giancarlo Trecate, and Nikolas Geroliminis. Idle-vehicle rebalancing coverage control for ride-sourcing systems. pages 1970–1975, 07 2022.