Master Project

Experimental Comparison of Autonomous Vehicles Routing Optimization Algorithms

Author:	Prisca Aeby [1]	prisca.aeby@epfl.ch
Supervisors:	Bastien Rojanawisut [2]	bastien.rojanawisut@bestmile.com
	Boi Faltings [3]	boi.faltings@epfl.ch

July 30, 2017

[1] Computer Science, École Polytechnique Fédérale de Lausanne, Switzerland

[2] Scala Backend Software Engineer, BestMile, Switzerland

[3] Artifical Intelligence Laboratory , School of Computer and Communication Sciences, École Polytechnique Fédérale de Lausanne, Switzerland, `liawww.epfl.ch`

Abstract

Your abstract.

Contents

1 Introduction

General problem: two types of services, on-demand and fixed-line => explain why we focus on fixed line (all constraints of the platform) Fixed line because now roads are not made for more complex systems Goal: run simulations as close as possible to real world situations => how much it can improve the system performances to be able to forecast demand bring concrete solution to transport models which are implemented now

2 Related Literature

The scope of vehicle scheduling problems related to our specific case study can range from fixed-line bus services to on-demand systems. In fact, electric shuttles' routes and stops are predefined but timetables can be flexible to respond to the known real-time demand. The set of constraints differs from traditional bus transportation, for example workforce regulations can be eliminated as the shuttles are not human driven but a battery management component needs to be introduced into the planning process. As we will see in this chapter, current research often concentrates on one specific aspect of the scheduling strategy, simplifying some constraints, assessing the system's performance using various types of metrics and considering different inputs. In what follows, we describe in section 2.1 some techniques used in the traditional public transport industry to guarantee a good quality of service. In section 2.2 we describe the concept of Demand-Adaptive Transit Systems. Finally, we detail in section 2.3 an approach proposed by the Urban Transport Systems Laboratory at EPFL (LUTS) for scheduling autonomous vehicles activities.

2.1 Public Transport

Within the standard transit organization, policies and standards affect a lot the development of transport strategies and how people interact with the systems. The planning process of a fixed line bus transportation service is mainly composed of three different tasks: route planning and deciding service frequencies as well as the service timing. Route planning consists of defining a sequence of stops composing each route and how those routes are interconnected. Deciding service frequencies implies specifying the number of vehicles per unit time which need to pass a given route (often expressed in vehicles per hour). A common measure used to express the ideal distance between vehicles is the headway, the inverse of the frequency, in other word the fixed interval at which vehicles are coming at a station. The service frequencies set by the transport organization can be chosen based on different policies (see Pine et al. (1998)):

1. **Fixed headway**: the agency establishes a fixed interval ant time at which vehicles come to stations. It is convenient for customers as they have access to an exact schedule, but it is hard to keep the time between vehicles constant as it is vulnerable to external disturbances.

2. **Demand-based headway**: in existing transport services, agencies can adapt the timetables based on the observed demand at the stations (number of passenger boardings/deboardings) in order to reach the desired passenger load in vehicles.

3. **Performance-based headway**: the goal is to find the headway for which performance standards are optimized. Those costs are usually measured during a service period, for example a day. It may include the service productivity (e.g. the revenue per passenger per hour), the cost effectiveness of the service or the overall effectiveness (e.g. net subsidy per passenger).

All those scheduling strategies suffer from the well-known bunching effect: two or more buses arrive at the same time at a stop with the first one being overcrowded and the other one empty. This phenomenon occurs because of external disruptions in the service (e.g. stochastic passenger arrivals at stops, traffic jams, etc) (see Camps and Romeu (2016)) slowing down one of the bus which pick up passengers who would have taken the next bus.

Several corrective measures based on different strategies have been developed to overcome this problem. There are mainly two different holding approaches to mitigate bus bunching as described in van Oort et al. (2010):

- **Headway-Based Holding:** vehicles arriving with a shorter headway at a stop (or holding point) wait to restore the headway distribution. If they arrive with a longer headway, it is possible to speed up the buses by skipping stations. The analytical study Cortés et al. (2010) proved the efficiency of using headway holding strategy and bus skipping (considering the extra waiting time of passenger whose station has been skipped) with the two-dimensional objective function composed of the regularization of bus headway on the one hand and the level of service on the other hand with respect to a circular route scenario.
- **Schedule-Based Holding:** in the case of fixed timetables, schedule-based holding involves holding a vehicle at a stop if it is ahead of its schedule and dispatch it immediately otherwise.

Zhao et al. (2016) proposed a method using boarding limits at stations to control buses which does not involve bus accelerating or waiting at stations and thus does not influence the customers' travel time and does not disturb the traffic. They do not consider a fixed schedule and a priori target headway. However, they proved that their self-adjusting control stabilizes the headway spontaneously in a short time.

Adra et al. (2004) study the importance of the vehicle load effects on the emission of vehicles through the study of the load factor and the empty running rate. The passenger load factor measures the efficiency of a transportation system, representing the capacity utilization of the seats. It is often expressed as the ratio between the passenger-kilometers traveled to seat-kilometers available.

As stated by He (2015), the majority of earlier studies conducted on maintaining a balanced headway uses arrival time of the current bus at the current stop and arrival times of the preceding buses but does not take advantage of real time information like the vehicles' geolocation. Later methods proposed new approaches assuming availability of locations and even real-time arrivals of passengers to each bus stop. For example, Daganzo and Pilachowski (2011) proposed a solution taking into account the distance between the current bus and the preceding and following buses to adjust the speed of the current bus. It includes holding buses at stations, accelerating or decelerating.

The studies carried out within the traditional fixed-line bus services handle situations where the demand is consistently strong over the territory and where the fleet is composed of high-capacity vehicles. When the demand is weaker, it is complicated to operate an economical and frequent transit system as the resources are shared by few people and are very costly (e.g. driver salaries, fuel expenses, etc.). The autonomous fleets are composed of vehicles with lower capacity, but it is easier to dynamically adapt their scheduling strategy at low cost and have access to real-time information.

2.2 Demand-Adaptive Transit Systems

Demand-Adaptive System (DAS), or Demand-Responsive Transit (DRT), is a personalized type of transportation displaying features from both fixed-line services and on-demand systems. In such systems, passengers are picked up and dropped off on-demand at predetermined stops, and unrequested stops are skipped. There are still compulsory stops, which are served within a predefined time window, but the routes are adapted based on the the requested stops. A method to determine the time windows at the compulsory stops has been proposed by Crainic et al. (2010). Li et al. (2007) depicted the advantages to substitute fixed-line bus services to DRT services in two cities in California with low demand density.

Gabriel et al. (2008) address the issues of evaluating DAS services in comparison to traditional transit services and fully on-demand systems. The transit systems' evaluation denotes the part of the planning process dedicated to the study of the behaviour and the performance of the line under various conditions with respect to demands, policies and costs. They mention the main steps of transit lines' evaluation in order to tune operating parameters impacting the system performance under different scenarios for cost-benefit analyzes: 1) the scenario, parameters and policies are specified, as well as the demand to serve 2) the line is designed 3) the operation of the line is simulated (during a specified time-horizon) 4) results are collected and performance measures are computed. The importance of demand generation in step 1) differs from one system to the other: whilst in fixed lines transits static methods are often used to simulate an average demand, dynamic methods are used to simulate the time-dependent stop-to-stop requests in on-demand systems. The performance measurements in step 4) vary as well from one system to the other: in fixed-lines the service quality is mainly represented by the buses' punctuality and the constant interarrival times whereas for on-demand systems it is expressed by the specific users' waiting time and journey time. Gabriel et al. (2008) propose a framework for evaluating DAS lines including the scenario input, the optimization modules for the design and operation of the line and the simulation module which yield the statistical information about the performance.

2.3 Vehicles Assignment to Routes

Bongiovanni et al. (2016) present an optimization approach for the autonomous vehicles scheduling within a transit system composed of circular routes. They separate the planning process in two main tasks: line planning and vehicle scheduling. Line planning corresponds to defining the routes and the required headway on each route over the planning horizon. Vehicle scheduling refers to computing a general assignment plan for each vehicle for each time period over the planning horizon of one day. It means that each vehicle is assigned to a specific route or to a recharging station for each time period. The underlying assumption is that the vehicles will not transfer

between routes and charging stations too often which makes the time discretization of the planning horizon possible. They focus on the vehicle scheduling task which takes as input the predefined routes, frequencies, fleet size and battery model. They present the scheduling problem as a mixed-integer linear programming (MILP) with battery constraints and the goal of the objective function being to maximize the total battery charge levels at the end of the planning horizon.

Chakroborty et al. (2001) tackle in their study the optimal fleet size distribution, so the fleet size of each route, and the scheduling strategy to adopt which minimizes the waiting time of passengers at their point of origin and the transfer times from one route to an other. They formulate the problem as a non-linear mixed integer programming problem (NLMIP) with resource limitations and service related constraints such as the fleet size, a minimum required headway, a minimum fleet size on each route and a maximum transfer time for passengers. They consider in the objective function only the level of service offered to the passengers, taking into account the initial waiting time of passengers and the total transfer time of passengers but no vehicles' operating cost. They mention the lack of work on this topic, possibly because of its extreme complexity, but consider the most promising attempts at solving the optimal scheduling problem being the ones using simple binary Genetic Algorithms as the optimization tool.

3 Problem Formulation

In this section we formulate the mathematical model describing the autonomous vehicles circular line system. All the variables are listed in Table 1. In addition to traditional bus transit system descriptions, we introduce the concept of battery stations and we include dynamic bookings from one station to an other station on the loop. As we have access to information during the service horizon time H, we denote a discrete timestamp t which takes values between $[0, H]$ at fixed sampling $interval$.

3.1 Circular Route

The circular line in this transportation system is a route which starts at a certain station, traverses other stations once and terminates at the starting station. The route is represented by set of stations and a set of directed edges E linking each station $s \in S$ to its following station. The set of charging stations C includes stations located on the stops within the loop: $C \subset S$. If the charging station is not used by any vehicle of the fleet we have $A^s = 1$. A stop s can be mandatory (vehicles always stop when they reach the stop) or optional (the stop can be skipped). If the stop is mandatory the boolean variable M^s is set to 1, and 0 otherwise. The maximum speed of an edge, representing the traffic at time t, is expressed as $speed_t^e$. The total distance of the circular line is given by len.

3.2 Vehicles

Each vehicle composing the fleet $v \in V$ moving on the circular closed line is characterized by a capacity c^v, a maximum charge q^v, a recharge rate α^v, a battery consumption rate β^v and a maximum speed s^v. The battery level of vehicle v at the timestamp t is $I_t^v \in [0, q^v]$. The battery change between the preceding timestamp and the current timestamp t is given by Δ_t^v. The edge it is currently traversing is e_t^v and

Notation	Signification
$S = \{1, ..., n\}$	Set of stations on the route
$E = \{n + i, ..., 2n\}$	Set of directed edges linking the stations
$V = \{1, ..., v\}$	Set of vehicles composing the fleet
$V^{\#} \subset V$	Set of active vehicles (not charging)
$C \subset S$	Set of charging locations
$B = \{1, ...b\}$	Set of bookings made during the scheduling horizon
$B^* \subset B$	Set of bookings which have been satisfied at the end of H
H	Length of the scheduling horizon
$interval \in \mathbb{R}$	Interval at which values are sampled
$T = \{t = interval * k, k \in \mathbb{N} \mid 0 < t < H\}$	Set of timestamps at regular $interval$
$T^* = \{t^* = 1\text{h} * k, k \in \mathbb{N} \mid 0 < t < H\}$	Set of hourly timestamps
c^v	Maximum load capacity of vehicle $v \in V$
q^v	Maximum charge of vehicle $v \in V$
α^v	Battery recharge rate of vehicle $v \in V$
β^v	Battery consumption rate of vehicle $v \in V$
s^v	Maximum speed of vehicle $v \in V$
$speed_t^e$	Speed of edge e at time t
$I_t^v \in [0, q^v]$	Battery level of vehicle v at time t
Δ_t^v	Battery change between t and its preceding timestamp
e_t^v	Edge $e \in E$ which vehicle v is traversing at time t
$next_t^v$	Distance to the next vehicle in front of v on the loop at time t
$o_t^v \in [0, c^v]$	Number of passengers in vehicle v at time t
$wait_t^v$	Seconds vehicle v needs to stop at time t
$b_{i,j} \in B$	Request of trip from station i to station j, $i, j \in S$
t^b	Time at which the request $b \in B$ has been made
n^b	Number of passengers for booking $b \in B$
p^b	Pick-up timestamp of booking b
d^b	Drop-off timestamp of booking b
w^b	Waiting time of booking b
ρ^b	Journey time of booking b
dem_{t^*}	Demand density, number of bookings made between t^* and $t^* + 1\text{h}$
len	Total length of the circular route
$A^s, s \in C$	1 if s is an available charging station, 0 otherwise
M^s	1 if s is the stop is mandatory, 0 otherwise
$1_{V^{\#}}(v, t)$	1 if the vehicle v is active and not charging at time t
$1_{B^*}(b)$	1 if request b has been satisfied

Table 1: Problem sets and parameters

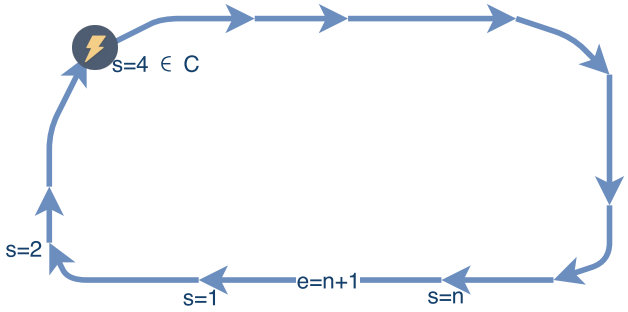


Figure 1: Example of a circular route

the distance to the next vehicle in front of it on the loop is given by $next_t^v$. The occupancy of the vehicle is expressed as $o_t^v \in [0, c^v]$. The vehicle is travelling either at its maximum speed or at the limitation of the current edge. Its current speed is therefore $\max\{s^v, speed^{e_t^v}\}$. When a vehicle is charging at a station $1_{V\#}(v,t) = 0$ and it can not carry any passengers $o_t^v = 0$. A vehicle entering a station at time t might by forced to wait $wait_t^v$ seconds.

3.3 Bookings and Demand

The set B represents the users' bookings made during the service horizon. A booking $b_{i,j} \in B$ requests a trip from station i to j at time t^b for a group of n^b persons. If the booking has been satisfied, the boolean variable D_b is set to 1 and we know that it has been executed by vehicle $v^b \in V$. The pickup time of booking b at station i is p^b and drop-off time at station j is d^b. The waiting time w^b equals $p^b - t^b$ and the journey time ρ^b is $d^b - p^b$. We denote B^* the set of bookings which have been satisfied, so the bookings for which $1_{B^*(b)} = 1$. Moreover, we define the demand density dem_{t^*} the number of bookings which have been made between t^* and $t^* + 1\text{h}$, so we have $dem_{t^*} = |B^\#|$, with $\forall b \in B^\# : t^* \leq t^{*b} < (t^* + 1\text{h})$. $t*$ is a discrete hourly timestamp over the scheduling horizon: $t^* \in T^*$ with $T^* = \{t^* = 1\text{h} * k, k \in \mathbb{N} | 0 < t^* < H\}$.

3.4 Output Metrics

At the end of the service horizon H, several metrics can be derived in order to evaluate the performance of the overall system dispatching the vehicles to serve the requests.

Battery

The total battery consumed by a vehicle v is the sum of the battery changes over the scheduling horizon when the vehicle is not charging:

$$battery^v = \sum_{t \in T} \delta_t^v * (1_{V\#}(v,t))$$

The total battery cost for the entire system is therefore the sum of the battery consumption of each vehicle composing the fleet:

$$batteryCost = \sum_{v \in V} battery^v$$

Occupancy

The average occupancy of a vehicle v over the scheduling horizon T is given by:

$$avgOccupancy^v = \frac{\sum_{t \in T} o_t^v}{|T|}$$

The average occupancy does not take into account the time vehicles spend at charging stations and therefore can not carry any passenger. As explained in section 2.1, a common unit used in transportation measurement is the load factor, which expresses the efficiency of a vehicle based on the kilometers traveled by passengers over the total kilometers that could have been driven considering its maximum capacity. In our case, as we are mainly interested in how the battery of the vehicles has efficiently been used, we define the passenger-battery measure: it is the battery that has been consumed by the vehicle during the passenger journey time. The passenger-battery for one vehicle is therefore the sum of the passenger-battery of the bookings it has satisfied. The seat-battery available represents the maximum possible passenger-battery if the vehicle would have been always full: it is the battery consumption of the vehicle times its maximum capacity. The vehicles' load factor is its passenger-battery over its seat-battery:

$$loadFactor^v = \frac{\sum_{t \in T} \Delta_t^v * (1_{V\#}(v,t)) * o_t^v}{battery^v * c^v}$$

We can compute the average load factor of the fleet as:

$$avgLoadFactor = \frac{\sum_{v \in V} loadFactor^v}{|V|}$$

Waiting Time

The average waiting time among the satisfied bookings is given by:

$$avgWaitingTime = \frac{\sum_{b \in B^*} w^b}{|B^*|}$$

We measure the variance of the waiting time as well which expresses the stability of the waiting time over all the bookings. As we want the waiting time to be as stable as possible we want to minimize $stabilityWaitingTime = \widehat{Var}(w)$.

Journey Time

The average journey time among the satisfied bookings is given by:

$$avgJourneyTime = \frac{\sum_{b \in B^*} \rho^b}{|B^*|}$$

Completed Bookings Ratio

At the end of the scheduling horizon, some bookings may not have been satisfied. The ratio of bookings which have been satisfied is given by:

$$completedBookingsRatio = \frac{|B^*|}{|B|}$$

4 Methodology

In order to evaluate the behaviour and performance of the system under various conditions with respect to the line configurations, the booking scenarios and the vehicles scheduling alternatives, we undertake analyses under laboratory conditions by simulating the operation of the system on the platform dispatching the vehicles. One of our main goal in the analysis we conduct on the line is to get measures as close as possible to real world conditions. In order to achieve this, a dynamic system is required which handles the time-dependent events of the system. In section 4.1 we describe the behaviour of the different interconnected services simulating the vehicles' behaviour reacting to external conditions and how the overall system is evaluated. In section 4.2 we present the variations of scheduling strategies used to dispatch the vehicles on the line. The parameters characterizing the simulation framework and the ones introduced in the scheduling strategy are summarized in table 3.

4.1 Simulation Framework

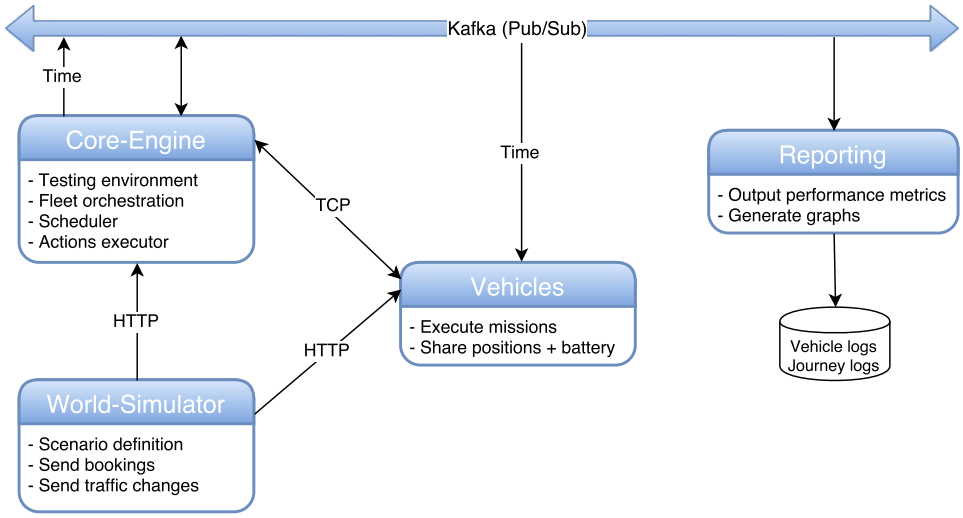


Figure 2: Interconnected modules composing the simulation framework

Figure 2 illustrates the different web services and how they are linked to each others. Services can communicate by exchanging messages via Kafka which is an asynchronous publish-and-subscribe message broker. In what follows we describe the behaviour of each module and the simulation parameters which can be tuned.

4.1.1 World-Simulator

The world-simulator is a web service which is used to start the simulation, send the booking requests and send traffic updates information through the HTTP connection established with the core-engine application. At the beginning of the simulation, it sets the time control to a given ratio which is sent to the core-engine application. A day can be simulated in m minutes which corresponds to a ratio r of $m/24h$. The simulated time $simTime$ is therefore r times the real time. The world-simulator is instantiated with a scenario describing the external events which affect the overall system. It mimics the

bookings which would have been made by users with the booking application and sent through a REST API. Every simulated second at time $simTime$ it checks if there are bookings with $t^b = simTime$ and speed updates to send to the core-engine application with $speed_t^e, t = simTime$.

4.1.2 Simulated Vehicles

Vehicles are connected to the core-engine through an established TCP protocol. They can send status messages and receive missions, like going to a specific charging station or waiting at a stop at for $wait^v_{simTime}$ seconds. A vehicle executes every mission it receives from the core-engine. The vehicles are simulated concurrently following an actor model. Each vehicle is a different actor part of the actor system. When the vehicles web service receives an HTTP request to start the simulation with a given fleet size from the world-simulator, an actor is created for each vehicle, which announces its position to the core-engine through a TCP message and subscribes to the time topic. The vehicles start at equal distance of each others on the loop with the battery completely recharged and their position and battery level is then updated every $1000/rate$ milliseconds.

4.1.3 Core-Engine

The core-engine application is a distributed system that combines and processes real-time information for coordinating and optimizing the fleet of autonomous vehicles. Several testing environment components are instantiated by the core-engine application: the map, the vehicles' configurations and fleet schedule through the day. The map is a GeoJSON file which is translated into the application to a directed graph with an edge between every stop and charging stations. The vehicles' configurations include all the parameters enumerated in section 3.2. The core-engine handles the bookings it receives from the world-simulator, updates the edges' speed of the graph and sends the actions to be executed by the vehicles. When it receives the starting simulation time and ratio r from the world-simulator, it publishes the updated time to the corresponding Kafka topic. The scheduler component manages the number of vehicles which need to be on service at different times during the day and sends corresponding missions to the vehicles. The positions and battery states received from the vehicles through the TCP connection are then published to the corresponding Kafka topics.

4.1.4 Time Synchronization

The different time-dependent services use a *timesource* controller which is a time simulator allowing the services to use a time behaving in a transformed way. Each *timesource* has in addition to the ratio r a $shiftDuration$ which shifts the simulated time by a constant value and a $startTimestamp$ which is a synchronization marker used to create multiple *timesources* which are consistent with each others. $Timesources$ created at different time with the exact same ratio r, $startTimestamp$ and $shiftDuration$ are perfectly synchronized as long as the system clocks of the running services are synced as well. In order to get the simulated time on the different services we can simply compute it with the following formula:

$$simTime = startTimestamp + (currentSystemTime - startTimestamp) * r + shiftDuration$$

4.1.5 Reported Metrics

The reporting web-service subscribes to the vehicle and journey topics and writes the stream events to its database. An event for vehicle v at time t includes its speed, position, battery level and battery state x_t^v. There is an event only for a booking which has been satisfied $b \in B^*$ and it includes all the parameters listed in section 3.3. At the end of the simulation, events are fetched from the database at fixed sampling $sampleInterval$ and then those logs are analyzed in order to compare different scenarios in term of their overall performance. All the metrics listed in section 3.4. In addition, we represent visually the simulation's performances with different graphs. In order to evaluate the different behaviours among the vehicles, we plot the hourly average occupancy of each vehicle during the scheduling horizon H. It is useful to see if all the vehicles are carrying passengers through the scheduling period or only few of them. We compare as well the number of vehicles which are at charging stations to the hourly average waiting time. It gives a good indicator to determine if the time at which the vehicle has been sent to charge was appropriate or if it impacts a lot the service in terms of quality of service provided to the passengers.

4.2 Scheduling

The approach proposed by Bongiovanni et al. (2016) described in section 2.3 is formulated as an optimization problem which considers the optimal headway for each route as an input but does not specify how to define it. Many real world aspects are not considered in their model. For example, it does not include transfers costs to charging stations or the time it takes to stop at stations and board passengers. It does not evaluate any demand scenarios neither. Our approach is to focus on scenarios as close as possible to real world conditions and find heuristics to define the number of vehicles needed on the fixed line in order to have a good balance between the battery consumption and the quality of service offered to the passengers. In what follows we describe how the core-engine handles the missions it sends to the vehicles based on the dynamic events it receives.

4.2.1 Vehicles' Activities Schedule

When a vehicle v arrives at a station j at $simTime$, it checks if it is carrying passengers which have made a booking $b_{i,j}$. If it is the case it needs to stop and its load is updated. If any booking $b_{j,k}$ has not been picked up yet $b \notin B^*$ and that the condition $o^v_{simTime} + n^{b_{j,k}} \leq c^v$ holds then it picks up the passenger(s) from the booking request. It waits $i^v_{simTime}$ seconds based on the headway computation which is explained in section 4.2.2. If the vehicle does not need to stop to drop off or pick up passengers and that the station s is not mandatory $M^s = 0$ then it can simply skip the station.

The battery level $I^v_{simTime}$ of vehicle v is controlled every time a vehicle arrives at a station. If it is under a threshold $minBatt$, it can not pick up any passenger any more, need to drop off every passenger at their requested drop-off station and finally goes the the closest charging station. When a vehicle is recharging and there is a booking b not picked up yet, $1_{B^*} = 0$, then the vehicle is dispatched on the loop again if its battery level is above a threshold denoted $enoughBatt$.

4.2.2 Headway

Even though the vehicles start at equal distance at the beginning of the scheduling horizon, if no strategy is adopted to maintain an equal distance between them the system will suffer from the well-known bus bunching effect described in section 2.1. In fact, the buses picking up passengers (which need to wait longer at stations) are slowing down and their preceding buses will catch them up, especially if some stations are not mandatory and they can skip them. As we have at our disposal updated vehicles' states, we can adopt a decentralized strategy in order to dynamically balance the distances between the active vehicles operating on the loop. The idea is to compute the ideal distance between the vehicles and to stop longer the ones too close to their following vehicle on the line, or on the other hand stop less if they are too far. As explained in section 4.2.1, each time a vehicle stops at a station we compute the number of seconds it has to wait until it can leave if does not skip the station. Let $default$ be the default waiting time in seconds at a station. The time a vehicle v waits at a station at time $simTime$ is $wait^v_{simTime} = default * ratio$, with $ratio$ computed as follows:

- The ideal distance between each vehicle is given by $ideal = len/|V^*|$, V^* being the set of active vehicles on the line with $V^* \in V$ and $\forall v \in V^* : x^v_{simTime} = 0$.
- We compute the distance to the next vehicle on the line $next^v_{timTime}$. Let $ratio$ be the ratio between the ideal distance and the distance to the following car $next^v_{simTime}/ideal$.

It holds then that if $ratio < 1$ the next vehicle is too far so the vehicle needs to wait less than $ideal$ and otherwise the next vehicle is too close and it waits more as $ratio > 1$. Moreover, the time a vehicle waits at a station is bounded from below by $waitMin$ and from above by $waitMax$ in order to avoid irrational values.

4.2.3 Dynamic Fleet Size

A fixed fleet of vehicles always operating on the line might not be an optimal solution to offer a good customer level of service whilst optimizing the fleet costs. In fact, passenger demand differ through the day and it might be substantially more cost-effective and efficient to adapt the number of active vehicles during the scheduling horizon H. For example, if the demand density is low at certain times of the day then it does not make sense to have many vehicles active on the loop as some of them will not be used at all. On the other hand, if the demand is high and passengers are unable to board the first coming shuttle it would increase drastically the waiting time. Several questions arise concerning the exact strategy to adopt in order to define the minimal number of active vehicles needed based on the demand and the frequency at which fleet should be rescheduled.

Our strategy implies to decide how many vehicles should be active at each hour of the scheduling horizon based on the demand density. In what follows we will first explain how the fleet is rescheduled when the number of vehicles needed on the line changes. We will then present the different approaches taken in our study to find a balance between the cost of the running vehicles and the quality of service offered to the passengers.

Rescheduling the Vehicles When the fleet size has to be changed decisions must be taken in order to adapt the number of vehicles on the line in an efficient way. It implies activating or deactivating the appropriate vehicles and scheduling their next activities. The main idea resides in optimizing the time vehicles are not being used by recharging them. The vehicles will have to go to a charging station and it eliminates the problem of knowing where to pause them as they can not be waiting indefinitely at stops. There are two situations to handle: the fleet size has to be increased or decreased by δ vehicles. Scheduling events are created at each hour of the scheduling horizon in the Core-Engine application which handles rescheduling the vehicles.

- **Increase the fleet size:** we need to decide which vehicles to reincorporate into the fleet from the ones being at the charging stations $V^{\#}$. The goal is to insert the ones which have the highest battery level and eventually they will not have to be sent to charge whilst they will be operating on the loop as it affects the system performance. The δ vehicles with the highest energy level are activated to go back to the route. If the fleet size is not large enough $|V^{\#}| < \delta$, all the vehicles in $V^{\#}$ will be activated. If the battery level of a vehicle v is not sufficient to leave the charging station $I^{v}_{simTime} < enoughBatt$ then it stays on charge until its battery level reaches $enoughBatt$.
- **Decrease the fleet size:** following the same logic as the choice of vehicles to activate, the vehicles with the lowest battery level within the active vehciles $V^{\#}$ will be sent to the charging stations. They will first drop off any passengers they carry as described in section 4.2.1. However, even if their battery level reaches $enoughBatt$ they are not sent back to the route if they have not been reactivated in the meantime.

Optimal Fleet Size The goal is to find an optimal number of vehicles which need to be active on the line based on the demand density. There are therefore two main questions that arise: how can we define that a fleet size is *optimal* and what is exactly the *demand*. In what follows we will justify the strategical choices and heuristics used in this experimental study.

As explained in section 2, most of the studies we can relate to our specific problem of autonomous vehicles scheduling on fixed line formulate an optimization problem with several inequality, equality and combinatorial constraints. There is one specific objective function representing the cost to minimize which is often composed of the fleet operating costs and/or measures osf the level of service offered to the passengers. Different techniques can be used to find optimal or near-optimal solutions satisfying the constraints. There are some exact algorithms which find the optimal solution(s) but there are often not applicable for that kind of complex NP-hard problems as many problem instances are intractable. Heuristic methods must be used instead which find near-optimal solutions. Some software solvers can also be used and they often provide good solution for linear programming solution. However, when the scheduling problem is transposed into an optimization problem there are many real-world aspects that are omitted and many simplifications and assumptions are made. This is precisely the reason why we explore different scheduling strategies by simulating the real comportment of vehicles in an environment as close as possible to reality. The simulation framework for evaluating the system described in 4.1 is therefore close to the evaluation method

used by Gabriel et al. (2008) for DAS described in section 2.2, with the values we want to minimize close to the ones often used in the objective function of the optimization formulations. We now present how we formulate the demand at different times of the scheduling horizon and how we choose that a fleet size is sufficient to serve the demand.

- **Passengers Demand** In transit system for which the goal is to provide a given frequency of service, in other words to keep a perfect headway between vehicles, the demand is generalized as the number of passengers per hour at each stop. This demand can be time-dependent and vary through the scheduling horizon. The ideal number of vehicles can be directly derived using standard transportation mathematical models. In standard vehicle routing optimization problems, the demand is often expressed stochastically as a probability to have a given number of passengers going from one station to an other station. An other approach, used for example by Chakroborty et al. (2001) when choosing the number of vehicles for each route, is to define the demand as the number of passengers arriving between the successive arrivals of two buses at a specific station. In our case, we are mostly interested in the general demand over the loop as the vehicles go from one station to the following one and we do not modify their route. We define therefore the demand density as the number of bookings made over one hour from any starting stations on the loop. From the definition described in section 3.3 we have therefore that the demand density at the hour t^* is dem_{t^*}. It is reasonable to assume that the system is able to predict the number of demands which will be made at each hour of the scheduling horizon. In fact, whilst traditional bus transportation services make use of theoretical models, electric shuttles systems can take advantage of gathering real-life data and build prediction models through a machine learning pipeline. Models' accuracy can be continuously improved by training it on new data coming from the usage of the electric shuttles transport service. Useful real-life data used to build demand prediction models include the bookings made by passengers but also weather conditions, traffic and people behaviour through the area (e.g. residential, work or shopping areas). All the metrics described in section 3.4 can also be gathered and used to improve the system performance.
- **Optimality** We need to define a strategy to find an optimal number of vehicles based on the demand density of each hour. We adopt two distinct strategies inspired from the different formulations of the objective function in optimization problems. The first approach is to include both the fleet operating cost and the level of service offered to the passengers in the computation of the total cost to minimize. The second one is to include the minimum level of service which need to be achieved in the constraints set and therefore to minimize the fleet operating costs. For both strategies, in order to gather data about the overall performance and compute the different costs, we run several simulations with varying the fleet size as ell as the demand. The demand is simulated between dem^- and dem^+ and the fleet size between V^- and V^+. Those simulations are run over a scheduling horizon of one hour $H = 1\text{h}$ and all the metrics described in section 3.4 are collected. Vehicles have an autonomy larger than one hour which eliminates any noise coming from the battery scheduling of the vehicles.

    1. In the first strategy we consider defining a cost function including both

the cost of the vehicles and service provided to the passengers. For a fixed demand, the total energy consumption will grow proportionally to the fleet size whilst the waiting time decrease. This is logical as the more vehicles are active on the line the higher the service frequency will be. As we want to find a balance between the battery consumption and the quality of service offered to the passengers we sum the energy cost and the waiting time cost. In order to compare those two metrics which are not on the same scale, we normalize them between 0 and 1 by feature scaling $(x - min)/(max - min)$. We have to take into account the fact that some bookings made over the simulated hour are not completed by the end of the scheduling horizon H. We need then to maximize the percentage of completed bookings over the hour as the ones not satisfied will be reported to the next hour. We can therefore divide the energy and waiting costs by the percentage of bookings completed. The cost function for the demand $dem = |B|$ with a fleet size $|V|$ is then:

$$cost_{dem,|V|} = \frac{batteryCost + avgWaitingTime}{e^{completedBookingsRatio}}$$

$batteryCost$ is normalized between 0 and 1 with min being the $batteryCost$ when we run the simulation for the same demand with V^- vehicles and max with V^+ vehicles. The same scaling is done for the $avgWaitingTime$. Taking the exponential of the $completedBookingsRatio \in [0, 1]$ seems a reasonable measure as it will take values between $[1, e = 2.72]$ and that it is a factor influencing a lot the performance of the simulation over one hour. The optimal number of vehicles for demand dem is then the fleet size $|V^\circ|$ which minimizes the cost function: $min_{V^\circ \in [V^-, V^+]}(cost_{dem,|V^\circ|})$.

2. The second strategy is to define a level of service to be provided to the passengers and to find the minimum number of vehicles needed to satisfy the service constraint. The idea is to find a function that gives the $avgWaitingTime$ from the demand and the fleet size $avgWait_{dem,|V|}$ and then to find the minimum fleet size $|V^\circ|$ required so that the $avgWait_{dem,|V^\circ|}$ is under an acceptable threshold denoted $accWaitingTime$. As we have the data for the $avgWaitingTime$ based for each dem and $|V|$ combinations it is possible to run for example linear or non-linear regressions to get the equation for the function $avgWait_{dem,|V|}$. An example of this procedure applied to a real-life scenario is explained in details in section 5.

- **Dynamic Fleet Size Scheduling** For both strategies described above, we define the optimal fleet size $optFleet(dem) = |V^\circ|$. The dynamic fleet size scheduling strategy for a scheduling horizon H consists then in computing at each hour t^* of the scheduling horizon H what is the optimal fleet size $optFleet(dem_{t^*})$ and to create appropriate increase or decrease events.

5 Numerical Experiments

The purpose of this section is to get insights as close as possible to real-life conditions by running the simulations on a project which is undergoing in Mountain View,

	Signification
r	Simulation ratio: simulated time = r times real time
$simTime$	Current simulated time on the different applications
$rate$	Rate at which vehicles' state is updated
$sampleInterval$	Interval at which events are fetched from the reporting database
$minBatt$	A vehicle v needs to finish its mission(s) and go to charge if $I_t^v < minBatt$
$enoughBatt$	A vehicle v, x_t^v, can leave the charging station if $I_t^v < minBatt$
$default$	Default waiting time at station in seconds
$waitMax$	Maximum time a vehicle can wait at a station
$waitMin$	Minimum time a vehicle can wait at a station
V^-, V^+	Simulations between fleet size V^- and V^+ for finding optimal fleet size
dem^-, dem^+	Simulations between demand dem^- and dem^+ for finding optimal fleet size
$opt(dem) = \|V^\circ\|$	Optimal fleet size $\|V^\circ\|$ for demand density dem

Table 2: Simulation framework and scheduling parameters

California. The fixed line has been designed with predefined stops and the demand is generated by analyzing available transportation data. The vehicles' properties are derived from the furnished model from autonomous shuttle brand Navya. In what follows we describe in section 5.1 the different parameters used in the simulation framework in order to get stable results. In section **??**

Same number of vehicles that they would have, like in Sion they have only two vehicles. show what would be the gain of the network design

4.1.

tel the criticallevelbattery: 0.3 10 charging stations so they can always go describe battery model

5.1 Simulation Settings

The analysis of the different performance metrics under different scenarios makes sense only if the output metrics of the simulations are stable. We need therefore to find a trade-off between the speed at which simulations are run and the stability of the results. In fact, the framework runs on the company's software which is able to interact with real vehicles, which works perfectly in real-time. However some computations may influence the results when the time is accelerated on the different applications, especially because of the access to the various databases, the HTTP requests, the Kafka Topics and the complexity of the concurrent actor system simulating the vehicles. The multi-threaded Scala applications run on the same machine, a MacBook Pro 2.3 GHz Intel Core i7, with 6GB ram allocated to the Core-Engine application, 3GB to the Vehicles, 2GB to the World-Simulation and 2GB to the Reporting.

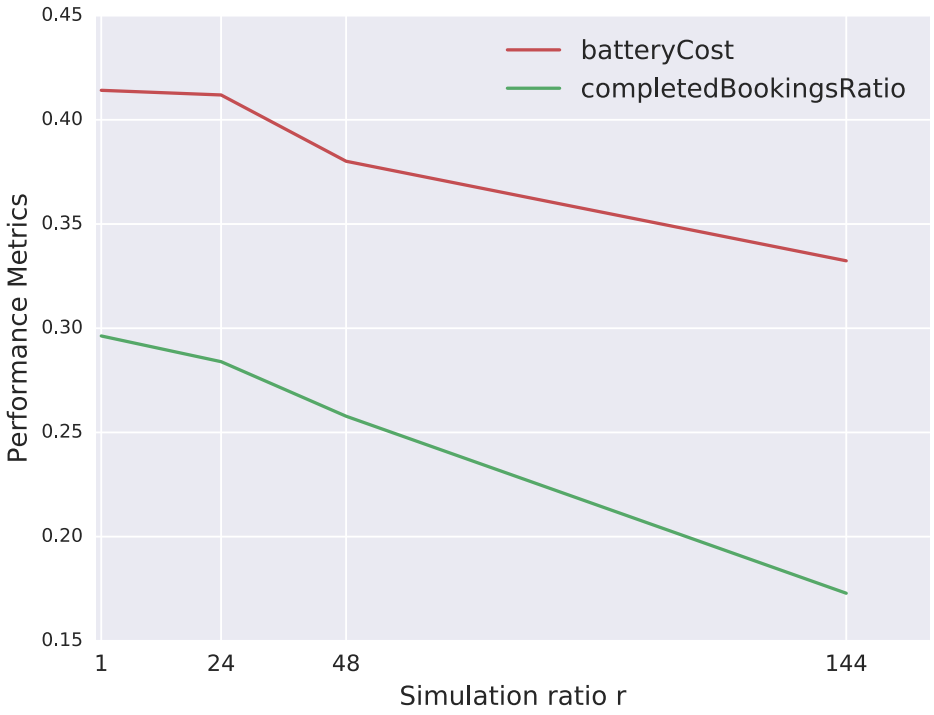
In order to assess the simulation framework stability, we run exactly the same scenario with different simulation parameters and choose the most appropriate ones with respect to the performance metrics. We want to find what simulation parameters output metrics as close as the ones simulated without accelerating the time, $r = 1$.

Simulations are run over a scheduling horizon H of one hour with a demand density

$dem_0 = 27$ which corresponds to sending a booking every 2'13". The fleet is composed of five homogeneous vehicles which operate on the same fixed line. When varying the simulation framework parameters, each simulation is run three times and the average performance metrics are reported.

Simulation Speed In figure 3 the *batteryCost* and *completedBookingsRatio* are reported for a sample of growing ratios r. For example $r = 24$ means simulating a day in one hour and $r = 144$ is equivalent to ten minutes. As you can observe, with $r = 144$ the metrics are far from the real metrics. In fact, the system can not handle all computations and the vehicles' positions are not updated consistently. There is a percentage change with the real value (at r=1) of the *batteryCost* of -19% and for the *completedBookingsRatio* of -41%. However, when the time is accelerated 24 times performance metrics remain really close to the real values with a percentage change of -0.53% for the *batteryCost* and -4% for the *completedBookingsRatio*. We choose therefore that running all simulations with a ratio $r = 24$ is a reasonable trade-off between the stability of the metrics and the computation time.

Figure 3: Performance metrics at growing simulation ratio r



Vehicles' Actor Speed As explained in section 4.1.2 the actors simulating the vehicles update their state (position and battery level) every $1000/rate$ milliseconds. If *rate* is too high the actor system end up with too many unhandled messages. On the other hand, if *rate* is too small the vehicles skip stops. A good compromise is achieved with a *rate* of 3.

Simulation Parameter	Value
r	24
$rate$	3
$interval$	10 seconds
$sampleInterval$	1 second

Table 3: Used simulation framework's parameters

Time Consistency Between Applications We need to assess if the Core-Engine application receives the bookings at the same time the World-Simulator application sends the booking through a REST API. In fact, there might be some delay as the World-Simulator checks every second if it needs to send bookings, so if there is a real-time delay of one second it can increase proportionally to the simulation ratio r. We compare the time at which the booking is supposed to be sent $simTime^b$ from the World-Simulator to the reported time in the logs from the Reporting application which populates its database at $sampleInterval$ of one second. We obtain that on average the difference is of 5 seconds and it never exceeds 55 seconds. It is totally acceptable as we run simulations over one day. Moreover, the vehicles' logs we get from the Reporting application are at regular $interval$ of 10 seconds.

5.2 Scenario Settings

In order to compare different scheduling strategies and different scenarios, we run all scenarios with the same type of vehicles on the same fixed line. In what follows all the parameters used for the simulation are detailed.

5.2.1 Vehicles

Each vehicle v composing the fleet V has a capacity $c^v = 10$. The battery level varies from 0 to $q^v = 1$. The battery discharges, proportionally to its speed s given in miles per hour, at each actor step of simulated duration $\delta = r * 0.33$ seconds by:

$$((0.0261905 * s^2 + 1.70238 * s + 95.3571) * s) * \Delta * \beta^v$$

and similarly it recharges by $\Delta * \alpha^v$. The homogeneous fleet has ratios α and β set to 1 unless stated otherwise. As explained in section 4.2.1, vehicles finish their missions and go to charge if their battery level is under $minBatt = 0.3$. Vehicles go at constant speed of 12.4 miles per hour as in cities with the current shuttle technologies and road settings it is not possible for them to go faster.

5.2.2 Graph

The fixed line is situated in Mountain View, California, with 32 stops situated at strategical points on the loop based on the types of area (accommodations, shopping centers, industry, etc). In current running autonomous transportation infrastructure, like for example in Sion for the project SmartShuttle, two vehicles are running on a fixed line and there is one depot where they can recharge. Currently it is reasonable to assume that all charging stations are centered at one place on the loop. We set therefore ten charging stations at the same place. The length TODO of the loop is of ... km. TODO put image of graph

5.2.3 Simulated Demand

The goal of the simulations is to send realistic bookings in this specific loop over the scheduling horizon of one day in order to get meaningful metrics. As there is no available information of the demand distribution on this specific loop, other data can be analyzed and realistic bookings can be extrapolated from it. The simulated demand is estimated analyzing traffic information data of the area of study provided by HERE Maps. This data set provides real-time traffic flow tags for the main road segments in the USA. There is for example the free-flow (maximum speed allowed) over the average speed at each minute of the day for each segment and the estimated number of vehicles. 20% of the total number of vehicles is considered as public transport. The data over one week from Monday to Friday is selected for analysis. Generalized additive model is used to predict the number of vehicles for every minute at each segment of the fixed loop.

I have tried regression to obtain the trend of n during one day and predict it for one day of weekday and weekend. Because of the complexity of the model, regression models are not able to capture all the sources of variation here, so to cqpture the trend, I have used generalized additive model (GAM) and ggplot to optain the smooth line. Using the GAM, I predicted number of vehicles for every min at each segment and according to change in the number I considered a pick up or drop of for the segment. Here is the ggplots with smooth lines

5.3 Headway Experiments

5.4 Generating Data

5.5

6 Conclusion

6.1 Summary

6.2 Future Work

References

Adra, N., Michaux, J. L., and Andre, M. (2004). Analysis of the load factor and the empty running rate for road transport. Artemis - assessment and reliability of transport emission models and inventory systems. Rapport de recherche.

Bongiovanni, C., Kaspi, M., and Geroliminis, N. (2016). Scheduling autonomous vehicles activities. Technical report, Urban Transport Systems Laboratory EPFL.

Camps, J. M. and Romeu, M. E. (2016). Headway adherance. detection and reduction of the bus bunching effect. *AET papers repository*.

Chakroborty, P., Deb, K., and Sharma, R. (2001). Optimal fleet size distribution and scheduling of transit systems using genetic algorithms. *Transportation Planning and Technology*, 24(3):209–225.

Cortés, C. E., Sáez, D., Milla, F., Núñez, A., and Riquelme, M. (2010). Hybrid predictive control for real-time optimization of public transport systems' operations based on evolutionary multi-objective optimization. *Transportation Research Part C: Emerging Technologies*, 18(5):757–769.

Crainic, T. G., Errico, F., Malucelli, F., and Nonato, M. (2010). Designing the master schedule for demand-adaptive transit systems. *Annals of Operations Research*, 194(1):151–166.

Daganzo, C. F. and Pilachowski, J. (2011). Reducing bunching with bus-to-bus cooperation. *Transportation Research Part B: Methodological*, 45(1):267–277.

Gabriel, T., Fausto, C., Malucelli, E. F., and Nonato, M. (2008). A proposal for the evaluation of demand-adaptive transit systems. *Annals of Operations Research*.

He, S.-X. (2015). An anti-bunching strategy to improve bus schedule and headway reliability by making use of the available accurate information. *Computers & Industrial Engineering*, 85:17 – 32.

Li, Y., Wang, J., and Chen, J. (2007). *Design of a Demand-responsive Transit System*. California PATH working paper. California PATH Program, Institute of Transportation Studies, University of California at Berkeley.

Pine, R., Niemeyer, J., and Chisholm, R. (1998). *Transit Scheduling: Basic and Advanced Manuals*. Report (Transit Cooperative Research Program). National Academy Press.

van Oort, N., Wilson, N., and van Nes, R. (2010). Reliability improvement in short headway transit services. *Transportation Research Record: Journal of the Transportation Research Board*, 2143:67–76.

Zhao, S., Lu, C., Liang, S., and Liu, H. (2016). A self-adjusting method to resist bus bunching based on boarding limits. *Mathematical Problems in Engineering*, 2016:1–7.