UNIVERSITÀ DEGLI STUDI DI MILANO

DOCTORAL SCHOOL
Mathematical Sciences

DEPARTMENT
Mathematics

DOCTORAL COURSE
Mathematics and Statistics for Computational Sciences

Exploiting available urban transportation resources with taxi sharing and rapid transportation networks: a case study for Milan

Alessandro GIOVANNINI

Supervisor: Prof. Giovanni RIGHINI
Co-Supervisor: Dott. Luca TOSI

Ph.D. school coordinator: Prof. Giovanni NALDI

A.A. 2014-2015

Contents

Introduction

In this dissertation we present research conducted in collaboration with AMAT[1] for more efficiently exploiting the available resources in terms of public transportation means and taxis. The aim is to increase, qualitatively and quantitatively, public mobility services in order to reduce the private motorized mobility and related externalities in the urban context.

In order to increase the efficiency of the taxi service and Local Public Transportation (LPT) network we:

- evaluate whether an urban massive transportation on-demand service, named Taxi Sharing, could be an opportunity for the city of Milan according to the level of service for citizens and revenues of taxi drivers;
- explore the possibility of planning, in presence of Taxi Sharing, a rapid LPT optimized for users without movement impairments according to users traveling and walking time.

Taxi Sharing allows users to move directly from departure point to final destination like with an individual taxi service, but sharing the vehicle with other users [1]. In Taxi Sharing systems users send a ride request to the operating station, which processes it with a dedicated software and informs them about ride conditions. The operating station, according to incoming user requests, continuously forwards to taxi drivers routing instructions, in order to serve users optimally.

Taxi Sharing rides take a few minutes more than individual taxi rides, but sharing the taxi results in lower fees. The Taxi Sharing service could satisfy the needs of:

- people desiring a sustainable transportation service that is faster and more comfortable than traditional public transportation and, at the same time, cheaper than the individual taxi service;
- night commuters, who need a means of transport when the public transportation service is reduced or absent, as an alternative to car use or individual taxis;

[1] Urban Mobility Agency of Milan.

- the elderly population, who could benefit from this service thanks to its convenience (door-to-door commute, seat assured, help in weights carrying) and to its lower price. By increasing elders' mobility, social isolation could be reduced [2];

- people suffering from motor impairment, who face problems using the traditional public transportation system. By assuring a simple, fast and low-cost moving method, their social integration would be supported, both in working and recreational perspectives.

The optimization problem related to Taxi Sharing service is referred to in literature as Dial-a-Ride Problem (DARP). The DARP includes two main subproblems called *allocation* and *routing*; *Allocation* refers to the partition of requests between the available vehicles, while *routing* concerns routing of vehicles to serve the allocated requests. DARP is an extension of the Traveling Salesman Problem (TSP) and is a NP-hard problem [3] from the point of view of computational complexity theory.

Algorithms to solve the Dial-a-Ride Problem (DARP) have been developed in the last decades [4, 5] in order to optimize door-to-door transportation services. Most of the developed algorithms are heuristics like tabu search [6, 7, 8] and genetic algorithms [9, 10].

On demand services have mainly been developed for low demand regions or time periods, for specific categories of users [11, 12, 13] or as a feeder service [14, 15]. These systems are usually based on few vehicles, typically mini-buses; they allow for rather wide time windows on departure and arrival and sometimes they are forced to use a limited set of predefined origins and destinations. The potential of on demand services as a spread urban means of transport has not been extensively studied in literature, with only some papers assessing this possibility using heuristics [20, 21].

We developed a new technique to optimize a high quality spread Taxi Sharing service in an urban context starting from state-of-the-art DARP optimization algorithms. In the Taxi Sharing system, time windows on pick-up and delivery times are narrow, the service is provided by many small vehicles (the existing fleet of taxis) and each point in the road network can be an origin or a destination.

These features allow an enumeration of all possible subsets of incoming users' requests for each vehicle and to compute in real time an optimal set of routes by solving a large set partitioning problem with state-of-the-art integer linear programming solvers. Owing to this fast global optimization capability, the system allows for a high quality service without any need of booking the ride in advance. In appendix B we classify our problem in the framework of Dynamic Vehicle Routing Problems (DVRPs).

Taxi Sharing is linked to the mobility system of the city in which it is developed. On the one hand, the conditions of the road network influence the performances of Taxi Sharing service; on the other the implementation of Taxi

Sharing allows new possibilities in planning rapid Local Public Transportation (LPT). Rapid LPT is achieved through optimal stop spacing [22], that:

- is influenced by the presence of Taxi Sharing, that offer an alternative to people with movement impairments;
- can decrease total passenger journey time, even though the walking time is longer;
- can increase the commercial speed[2] which affects transit time, operating costs and offer level.

The dissertation is divided in two parts:

- in part I we describe in detail the Taxi Sharing service, the new optimization method and the developed simulator (chapter 1, 2, 3);
- in part II we present the feasibility study for the city of Milan, according to three development scenarios for Taxi Sharing service, its integration with a rapid LPT and an informal related survey (chapter 4, 5, 6);

In chapter 1 we explain the characteristics of Taxi Sharing and we analyze the users and taxi drivers experiences according to main basic assumptions related to the service.

In chapter 2 we present the optimization method with respect to users' cases and taxi drivers' cases presented in chapter 1. We detail the three processes with a focus on the *optimizer* that relies upon a circular repetition of routes generation and set partitioning.

In chapter 3 we present an agent based Taxi Sharing simulator that enables to forecast how the service works from a quantitative perspective. We detail which inputs are needed to run the simulation, and which statistics are obtained on the quality, effectiveness, and efficiency of the service.

In chapter 4 we present three development scenarios according to demand level. For each analyzed scenario:

- we present the used data and parameters;
- we discuss the performance of the service in terms of number of requests serviced per hour, the average travel time and waiting time, the number of taxis simultaneously on duty, the ride fare and the taxi revenue;
- we show the details concerning the time and size of the optimization methods.

[2]The commercial speed is defined as the total distance traveled divided by total time taken (including scheduled holding).

In chapter 5 we present new possibilities in planning rapid Local Public Transportation (LPT) related to the realization of Taxi Sharing. We explore this possibility with a case study on tram line 9 concerning the effects in terms of total variation in user traveling time and increase of commercial speed achievable with the optimal stops spacing.

In chapter 6 we present the results of a survey related to rapid LPT and Taxi Sharing. We describe the sample who answered the questionnaire, and then we illustrate the interesting results that we gathered on more than 700 answers, about optimal stop spacing and use of Taxi Sharing.

Part I

Taxi Sharing

Chapter 1

Service specifications

Taxi Sharing is a dynamic on demand service provided by a taxi fleet in which the interaction between users and taxi drivers is handled by an operating station that processes the data regarding users and taxi drivers to optimize the service as shown in Figure 4.1.

Figure 1.1: Taxi Sharing actors.

The operative station relies on three main processes, detailed in chapter 2, to optimize the service:

- the *answerer* responds to users asking for ride conditions;
- the *optimizer* manages the fleet of *active taxis*, i.e. those taxis already on duty;
- the *finder* tries to *activate* new taxis when needed, looking within the *available* taxis.

In this chapter we detail the user and taxi driver cases and the interaction among them, as well as with the three processes of the operative station. In section 1.1 the user cases are detailed: *asking for ride conditions*, *sending the request*, *canceling the request* and *interrupting the ride*. In section 1.2 the taxi driver cases are detailed: *becoming available* mode, *available* mode, *becoming active* mode, *navigation* mode, *boarding user* mode, *disembarking user* mode, *interrupting ride* mode, *waiting* mode and *end of ride* mode.

1.1 The user experience

The user communicates with the operative station using one of the most popular technologies (APP, WEB, telephone with query's voice recognition software, telephone).

Figure 1.2 shows the flowchart of the request from when the request reaches the service until it is serviced.

In the next paragraphs we will detail the user cases and the requests states, in relation with the taxi driver cases and the optimization processes.

Case u1: asking for ride conditions

The user asks for a ride when he needs it, without booking it in advance. A user that wants to ask for a Taxi Sharing ride sends the request in which he must specify departure address, destination address, number of customers. At this point, the request is in the state of *new* request.

The *answerer* processes the query and informs the customer about the operational details of the ride: maximum arrival time to destination and trip cost. Now the request is in the state *to be confirmed*.

The maximum arrival time takes into account that some detours might be done to pick up or deliver other clients. A detour that would cause to reach the destination after the maximum arrival time will not be carried out. In section 2.2 it is detailed how the maximum arrival time is computed considering the time of the day, and the departure and arrival addresses. In section 2.3 it is detailed how the system evaluates the possibility to service together different requests by the same taxi according to the maximum arrival times communicated to users.

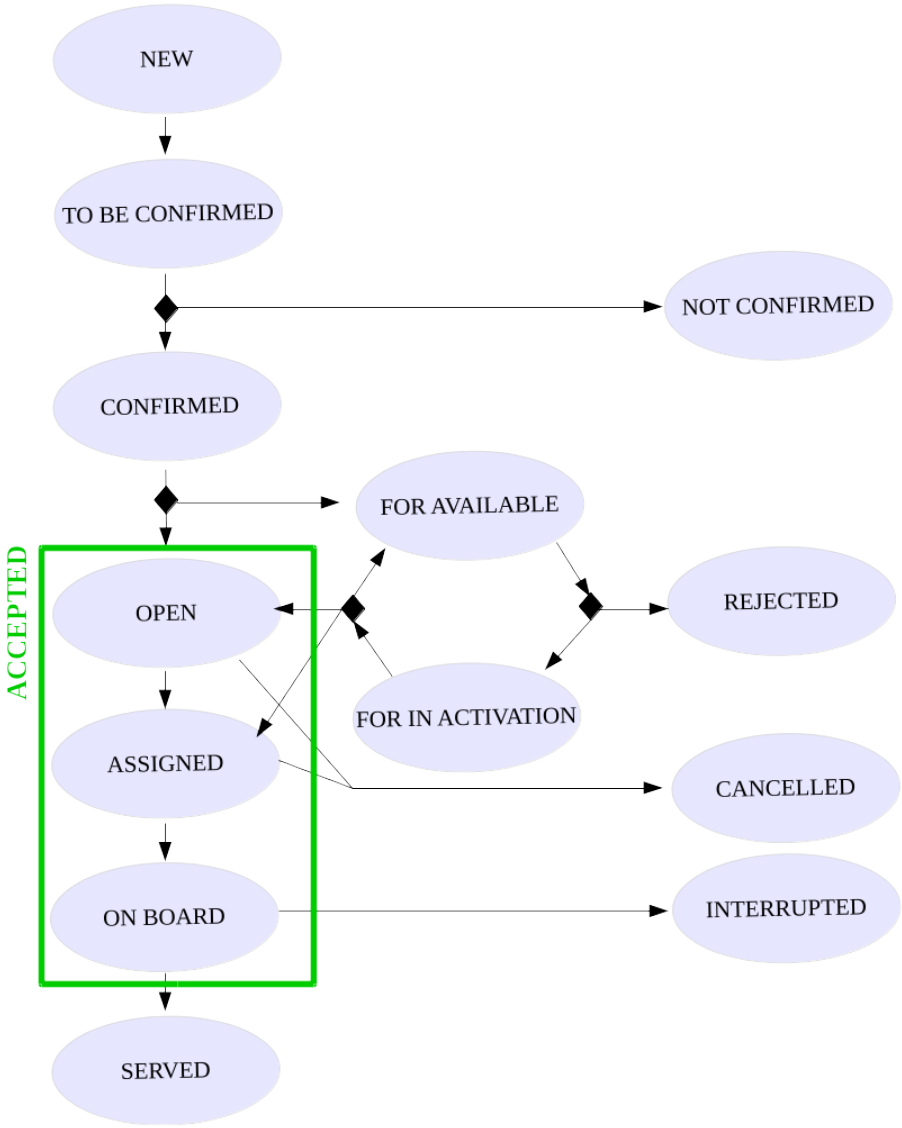


Figure 1.2: The flowchart of the request states.

The cost of the ride depends only on departure and destination addresses, and number of users, but doesn't depend either on the actual time and length of the ride or on the actual ride sharing. In section 2.2 the reasons for this key point are explained and it is detailed how the cost is computed.

Maximum arrival time to destination and trip cost depend only on the request and do not consider the actual taxi availability, which is checked only when the user sends the request.[1]

Case u2: sending the request

Once the user receives the ride details (maximum arrival time to destination and ride cost), he/she is asked if he/she is interested in the ride. If the user is not interested in the ride, the request state becomes *not confirmed*, otherwise, if the user is interested in the ride, the request state becomes *confirmed*.

At this time the *optimizer* tries to find an *active* taxi to satisfy the request, and if the taxi is found the request is *accepted*, otherwise the request state is set *for available* and the *finder* has to search for an *available taxi* that can satisfy the request. If the *finder* does not find a suitable *available taxi* the request is *rejected*, otherwise the request state is updated to *for in activation*. If the selected taxi accepts the ride (*case t2*), the request becomes *accepted*, otherwise it becomes again *for available*. This cycle is repeated until the request becomes *accepted* or *rejected*. The user is informed only when his/her request becomes *accepted* or *rejected*.[2]

As shown in Figure 1.2 there are three states of *accepted* requests:

- the request state is *open* if the request has been accepted by the service, but the choice of the taxi that will service it is still pending;
- the state is *assigned* if the user is still waiting for the taxi, but the taxi that will satisfy the request is established;
- the state is *on board* if the user is on board.

For any reason the user can decide to quit the service after his/her request has been accepted and before he/she has reached his/her destination.

[1] It is also possible to allow the user to skip the Case $U1$ and to send the request without asking for ride conditions. In this case, the user would be informed on ride conditions only if his/her request is accepted.

[2] To avoid rejecting requests, it could be possible to increase maximum arrival time to destination. It could be done off line using different parameters or online to guarantee the acceptability of the request. We have however decided to consider the possibility to reject a request if the quality of service would be to low. Furthermore, since the system is dynamic, as the taxis fleet providing the service the user can resend the request after some time and it would likely be accepted.

Case u3: canceling the request

Before getting on board, the user can cancel his/her own request by communicating to the service the request details and his/her intention to cancel it. If the service recognizes the request, it communicates to the user that the request has been deleted, and the state becomes *canceled*.

Case u4: interrupting the ride

When the user is on board, he/she can decide to end his/her ride before reaching the destination. In this case the user must communicate his/her intention to the driver who will let him leave the car: the state of the request becomes *interrupted*. The interaction between the user and the taxi driver in this case is not mediated by the operating station but it is direct. The taxi driver informs the operative station as detailed in section 1.2.

1.2 The taxi driver experience

The taxi driver that provides the Taxi Sharing service is an individual taxi driver who enrolled in the service and can provide both individual service and Taxi Sharing service. The taxi driver communicates with the operative station using a smart phone or a tablet with an APP that relies on the gps function of the device.

Figure 1.3 shows the flowchart of states related to taxis. In the next paragraphs we will detail the taxi drivers' cases and states, in relation with the users' cases and the optimization processes.

A taxi driver who wants to offer the Taxi Sharing service must enroll in the service. The service knows the users' capacity of each *enrolled* taxi.

Case t1: becoming available mode

When an *enrolled* taxi driver is waiting for a new ride, he/she can decide whether to become *available* for the Taxi Sharing service. The taxi driver who decides to become *available*, has to communicate until when he/she is available to service taxi sharing requests; we refer to this datum as maximum working time. In section 2.3 we will detail that maximum working time is used by the operative station in order to assign requests to the selected taxi, only if the request can be satisfied by the taxi driver before his/her maximum working time. When the taxi becomes *available* the service queues it in the parking area linked to the position detected by the gps.

Case t2: available mode

When a taxi is *available*, it periodically forwards its position to the operative station; the taxi driver can move in the city, but if he/she changes parking area

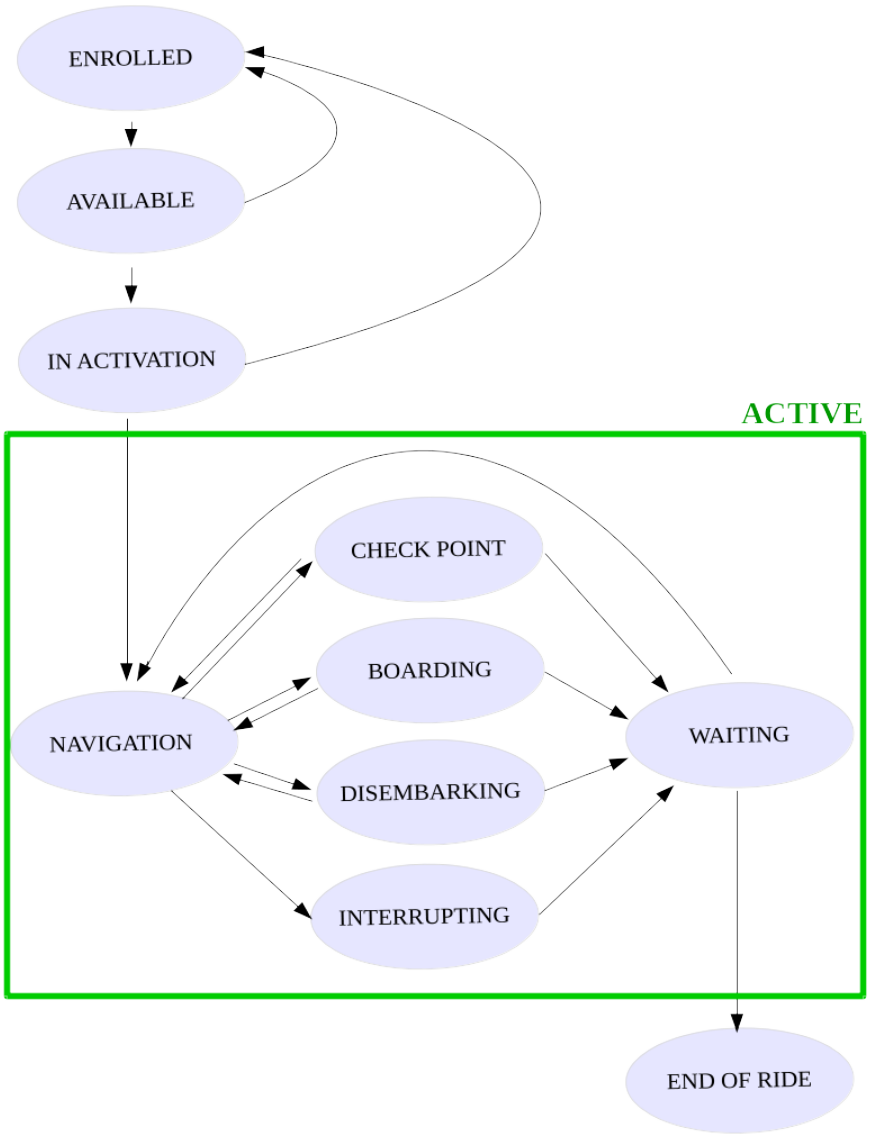


Figure 1.3: The taxi mode flowchart.

it is queued in the new parking area at the last position. An *available* taxi can always decide to exit from the *available* state.

When the service needs a new taxi, the *finder* selects an *available* taxi in order to make it an *active* taxi; the request for a new ride is forwarded to the taxi driver and the taxi state is updated to *in activation*.

Case t3: becoming active mode

A taxi driver, who receives a call for starting a new ride, has to decide if he/she wants to accept the ride or not. If the taxi driver does not accept the ride, the state is reset to *enrolled* and he/she needs to become *available* again in case he/she wishes to be re-queued in the parking. If the driver accepts a Taxi Sharing ride, then his/her taxi becomes an *active* taxi and starts moving around the city, servicing user requests until the end of his/her duty. We underline that the taxi has to accept a Taxi Sharing ride only at the beginning of his/her ride and subsequently users requests are assigned as default until the end of his/her duty. If the taxi driver needs to interrupt his/her ride, he/she has to reset the maximum working time.

When the taxi accepts the ride, the system forwards the driver an ordered list of *fixed* stops to be visited, and the details of the requests *assigned* to the taxi. A stop is a position on the road network. The stop is defined as *departure* if it corresponds to the origin of a request, it is defined as *destination* if it corresponds to the destination of a request and it is defined as *check-point* if it corresponds to a position on the road unlinked with any request but in that direction the taxi is supposed to go[3]. If the stop is of type *departure* or *destination*, the identifier of the request associated to this stop is linked to it. When a stop is communicated to a taxi driver, it is because it has been *fixed* by the *optimizer*, and hence it can not be changed. When a stop of type *departure* is fixed, the state of the related request is updated from *open* to *assigned* and hence the taxi that will service it is established. Every *active* taxi knows the *fixed* stops of its program, and has to reach the first of the list. In section 2.3 we will detail how the *optimizer*, during the active period, forwards the taxi driver new *fixed* stops to be visited and the details of the new requests *assigned* to the taxi.

The *active* taxi can be in six different states, as shown in Figure 1.3, related to the working mode detailed in the next paragraphs.

Case t4: navigation mode

When a taxi driver accepts a new ride he/she enters *navigation* mode. In *navigation* mode the taxi driver has to reach a point that can be the departure/destination address of a request or a *check point*. The taxi driver can decide whether to follow the navigator gps or to take another route to reach

[3]Check-points are introduced to improve service flexibility as detailed in section 2.3.1.

the point. In *navigation* mode the gps device periodically sends the position detected by the gps to the operative station. This information is used by the operative station to forecast when the taxi will reach the stop to which is directed.

When the taxi reaches the stop it has to inform the service. If the stop is a *check point*, the taxi remains in *navigation* mode towards next stop (if the stop is the last stop of the list then the taxi enters *waiting* mode). If the stop is the departure address of a request, the taxi enters *boarding user* mode. If the stop is the destination address of a request, the taxi enters *disembarking* user mode. If during the ride an user on board asks to leave the car before reaching his/her destination, the taxi enters *interrupting ride* mode.

Case t5: boarding user mode

When the taxi driver reaches a stop that is the departure address of an user, the taxi enters *boarding* user mode. The taxi driver sees all details of the request (departure/destination addresses, number of users, maximum arrival time and fare) and asks the user to confirm these data in person. If all data are confirmed, the taxi enters *navigation* mode towards next stop (if there are any); otherwise he/she enters *waiting* mode. If the user does not show up or is not interested in the booked ride according to destination addresses and number of users, the taxi driver communicates this *exception* to the operative station and enters *waiting* mode for new instructions[4].

Case t6: disembarking user mode

When the taxi driver reaches the stop which is the destination address of an on board user, the taxi enters *disembarking* user mode. When the taxi driver is ready to leave again, he/she informs the operative station and enters *navigation* mode towards next stop (if there is); otherwise he enters *waiting* mode.

Case t7: interrupting ride mode

If an on board user asks to leave the car before reaching his/her destination, the taxi enters *interrupting* ride mode. The taxi driver lets the user disembark the vehicle and communicates this *exception* to the operative station, signaling the identifying number of the user who left the taxi. The taxi enters *waiting*

[4]Users' delay are not tolerated since it would affect level of service for other users as well as the income of the taxi drivers, which is related to the number of users served each hour. We consider that, since the service is without reservation, the user is supposed to ask for a ride when is ready to leave, and hence the ride fee is charged to users also if he/she does not show up, or if he/she is not interested in the booked ride due to mistakes in the insertion of destination address or in the number of users. We consider that with these rules the user will behave in a correct way as a shared service requires: a wrong destination imply a suboptimal solution in terms of taxi selection, and a delay imply a lost of time for other users and taxi drivers.

mode, awaiting new instructions, since the previous schedule also included the destination of the user who left the taxi.

Case t8: waiting mode

The taxi driver enters *waiting* mode if he/she visited its last *fixed* stop to be visited, or if an *exception* arose: the user did not show up at the departure address, the user did not confirm the request details, the user interrupted his/her ride. The taxi driver in this mode must wait for new instructions. The *optimizer* handles the *exceptions* and updates the schedule of the *active* taxi as described in section 2.3, and if the *optimizer* establishes a new stop to be reached, the taxi enters *navigation* mode; otherwise, the taxi enters *end of ride* mode as detailed in the next paragraph.

Case t9: end of ride mode

The taxi enters *end of ride* mode if during the *waiting* mode period, the *optimizer* did not fix any new stops for the taxi. This happens if there are no users on board and no requests can be serviced by the taxi, depending on its position and maximum riding time. The details are presented in section 2.3. When the taxi enters this mode the service gives him/her the details of the ride: starting time, ending time, number of users serviced, total revenue. The taxi ride is finished and the taxi state is updated to *enrolled*. To start a new ride it is necessary to enter *becoming available* mode.

Chapter 2

The optimization method

The optimization of the Taxi Sharing service relies on three processes: the *answerer*, the *optimizer* and the *finder*. First of all, in section 2.1 we will describe the graph that represents the road network; it is the base of the Taxi Sharing service, since the departure and destination of requests are on the road network, and the taxis are moving on the road network. In section 2.2 we will detail how the *answerer* process gives the information about ride cost and maximum arrival time to destination to users asking for the ride conditions. In section 2.3 we will describe how the optimizer, the core process, optimizes the whole service according to the assignment of user requests and routing of taxi drivers. In section 2.4 we will detail how the *finder* decides which *available* taxi to activate if the *active* ones are non sufficient for servicing the *confirmed* users requests. The time schedule of the three processes, shown in Figure 2.1, is the following:

- the *answerer* runs every time a *new request* arrives to the service and processes one request at a time;
- the *optimizer* runs continuously during all service, and starts a new cycle just after the end of the previous cycle;
- the *finder* starts a new cycle after the end of the previous cycle only if there are some requests *for available* taxi, otherwise it starts a new cycle when one or more *for available* requests arrive from the *optimizer* or from an *in activation* taxi driver who refused a new ride.

2.1 The graph

We used a directed weighted graph $G(N, A)$ with N the set of nodes and $A \subset N \times N$ the set of arcs. Every arc $a \in A$ is hence an ordered pair of nodes $a = (n_f; n_t)$ and we say that the arc a goes from node $f(a) = n_f$ to node $t(a) = n_t$, where $f : A \mapsto N$ and $t : A \mapsto N$ relates an arc respectively to

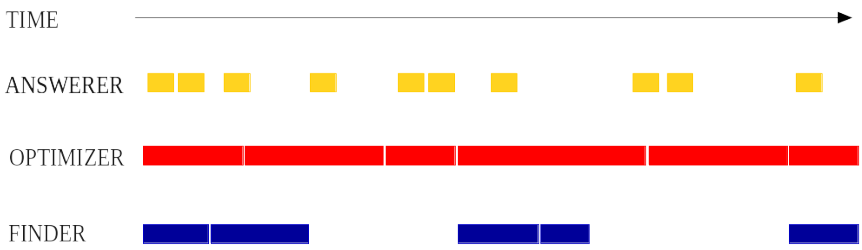


Figure 2.1: The time schedule of the three processes.

its starting and its ending node. We use a weight function $w : A \mapsto R_+$, in which the weight of each arc is given by its average travel time. A ordered couple of arcs $(a_i; a_j)$ is considered of consecutive arcs if $t(a_i) = f(a_j)$ and turn restrictions are considered as a pair of forbidden consecutive arcs (a_i, a_j). Common dynamic programming algorithms to compute shortest paths, like the Dijkstra algorithm, do not explicitly consider the turn restrictions or the fact that a node could be visited more than once by a shortest path in presence of turn restrictions. For these reasons, in order to model turn restrictions, we construct a line graph as shown in [23].

The directed weighted line graph $G(N_D, A_D)$ with a weight function $w_d(a) \mapsto R_+$ of a primal directed weighted graph $G(N, A)$, is constructed as follow:

1. The bijective function $d : A \mapsto N_D$ connects every arc in the primal graph to a node in the line graph.

2. For every couple of not forbidden consecutive arcs in the primal graph (a_i, a_j), there is an arc ϵ_t in the line graph from $d(a_i)$ to $d(a_j)$. The set of arcs of the line graph is the union of these arcs: $A_D = \bigcup_t \epsilon_t$

3. The cost function w_d of the arcs in the line graph is equal to the cost of the first arc in the primal graph: $w_d(\epsilon_t) = w(d^{-1}(f(\epsilon_t)))$.

The line graph allows the use of the Dijkstra algorithm and its faster variations [24, 25] to compute the shortest path between each couple of nodes of the line graph. We refer to $c_D(n_i, n_j)$ as the cost of the optimal path from node $n_i \in N_D$ to $n_j \in N_D$. The shortest path between nodes in the line graph represents the shortest path in the primal graph from the arc connected to the starting node of the line graph to the arc connected to the ending node in the line graph. The cost of the shortest path in the primal graph, according to how we had defined the weight function in the line graph, represents the time needed to move from the beginning of the starting arc to the beginning of the ending arc.

We identify a point on the road network with the arc to which the point belongs, a, and the distance of the point from the beginning of the arc, f. In this chapter and in the next ones we refer to this information as **position**: $p = (a, f)$. In two way roads, the position $p = (a, f)$ is correlated also to the street sides, since its arc a is oriented. In one way roads, the position $p = (a, f)$ is unlinked with the street side, it is not relevant in optimization process since both street side are accessible from arc a.

To compute the traveling time from a position $o = (a_o, f_o)$ to a position $d = (a_d, f_d)$ we use the formula:

$$T(o, d) = c_D(d(a_o), d(a_d)) - \frac{f_o}{v_o} + \frac{f_d}{v_d} \tag{2.1.1}$$

where v_o and v_d are respectively the average speed on arc a_o and a_d. We refer to this function in the subsequent chapters as traveling time, and it represents the expected traveling time at average speed of the fastest path from position o to position d.

If we otherwise define the weight function $w : A \mapsto R_+$ on the primal graph $G(N, A)$ to represent the length of each arc, we can compute the distance from a position $o = (a_o, f_o)$ to a position $d = (a_d, f_d)$ with the following formula:

$$D(o, d) = c_D(d(a_o), d(a_d)) - f_o + f_d \tag{2.1.2}$$

The length of the shortest path between origin and destination of an user's request is used to compute the cost of a ride as described in section 2.2.

The optimization method described requires us to calculate a large number of traveling times between two positions. To this aim two different methods are possible:

1. to calculate the optimal cost $c_D(n_i, n_j)$ in real time when needed between a pair of nodes of the line graph and use the function T to calculate the traveling time between positions. Using this approach it would be possible use a caching system whenever the shortest path for the same arc pair is necessary.

2. to calculate in advance all optimal costs $c_D(n_i, n_j)$ within each pair of nodes of the line graph and use the function T to compute the traveling time between positions. This approach require to recompute the matrix periodically to take into account new data on road network speeds.

The main advantage of the second method is the negligible time to compute T once all optimal costs are precomputed, the drawback is the time needed to compute all optimal costs and the space needed to store them. The choice between the two methods is linked to the graph size and to the frequency of data updating referring to the average speed for each arc. The updating frequency affects the recomputing or the updating optimal frequency of all optimal costs. In this dissertation we do not examine in depth this aspect, but we underline that this task is fully parallelizable with both approaches. With the first approach

since the computation of distances needed in processes related to taxi i are unlinked with computation of distances needed in processes related to taxi j, if $i \neq j$. With the second approach since the computation of all distances from node i is unlinked with the computation of all distances from node j, if $i \neq j$. For these reasons the computation of distances is not critical in the development of the service, but have to be taken into account in order to scale properly the needed hardware according to the choosen method and the dimension of the city. For the city of Milan on which we concentrate our research, the second approach is viable in terms of time and space for managing the service even with a personal computer[1].

In the simulation of the service for the city of Milan presented in the next chapters we used the second method, since we judge it to be the more suitable according to the used data in terms of speed on the road network during the day.

2.2 The answerer module

The *answerer* processes the queries of users asking for ride conditions as they reach the service, and computes maximum arrival time to destination and the ride fare.

The maximum arrival time to destination, mt, is a function of the direct time, $T(o, d)$, needed to travel from origin o to destination d, and it is computed according to:

$$mt = pt + \alpha + (1 + \beta) * T(o, d) \tag{2.2.1}$$

where pt is the present time, and α and β are parameters that can depend on the time. This computation does not take into account the actual ability of the service to accept the ride. To set these parameters it is necessary to evaluate a trade-off between the level of service guaranteed to users and the ratio of effective acceptance of users requests from the service. Maximum arrival time to destination, mt, is not a service guarantee, since road condition can be unpredictable. This value is used to decide which request can be served by the same taxi, and hence it guarantees that a detour that would imply reaching the destination after mt traveling on the road network at expected speed, would not be done.

The cost of the ride, c, is function of the direct length $D(o, d)$ of the optimal path between origin o and destination d, and it is computed according to:

$$c = n_u * (\gamma + \delta * D(o, d)) \tag{2.2.2}$$

where n_u represents the number of users, γ and δ are parameters that can depend on the time. We propose to link the cost only to the direct length of the

[1]Since from each node it is necessary to visit every node (in order to compute the distance to every node) no speed up techniques to reduce the space of visited nodes is needed, hence we used Dijkstra algorithm. Data on needed time to compute all the distances and space to store them are reported in chapter 4.

ride and the number of users, to allow the user to know the price in advance. In this way the price is not linked to the route that the taxi will make to service other users and to the actual sharing of the ride. In this way the variability is not suffered by the users but by the taxi driver; however, since the taxi driver is servicing many users, the variability at the end of the working day is low.

Finally the state of the *new request* becomes *to be confirmed*. At this point the user is informed of the conditions and, if interested, he/she can forward the request to the service.

2.3 The optimizer module

The *optimizer* is the core process of the optimization of Taxi Sharing, since this process optimizes the whole service according to the assignment of user request and routing of taxi drivers. The *optimizer* executes consecutively the following six steps:

1. The step *covering optimally requests* updates the programs of the *active* taxis according to *new confirmed* requests and the updated positions of taxi drivers.
2. The step *closing taxi* evaluates if an *active* taxi does not have any stop to visit and, in this case, the ride of the taxi will end.
3. The step *dealing exceptions* updates the schedules of the *active* taxis according to the exceptions thrown by the taxi.
4. The step *deleting canceled requests* updates the schedules of the *active* taxi that services a request that was previously canceled.
5. The step *considering visited stops* updates the schedules according to the stops visited by the *active* taxis.
6. The step *fixing new stops* evaluates if it is necessary to establish some stops within their schedule according to taxis positions and their schedule.

The first step is the core step and considers all *active* taxis and all *confirmed* and *accepted* requests together and it is detailed in section 2.3.1. From the second to the last step they involve one taxi at a time, these steps are detailed in section 2.3.2.

2.3.1 Service optimization

This section is divided in four parts:

A) In the first part we define the mathematical problem starting from the data regarding *active* taxis and *confirmed* and *accepted* requests;

B) In the second part we show how to compute all the possibilities to serve the users requests;

C) In the third part we detail the integer linear programming (ILP) model needed to cover optimally the user requests;

D) In the fourth part we describe how to obtain from the solution of the mathematical problem the updated schedules for *active* taxis.

A) From data to mathematical problem

Every *active* taxi has a schedule resulting from the previous cycle of the *optimizer* or from the *finder* if it is a new *active* taxi. Its schedule has a sequence of stops *fixed* or *not fixed*. In the process of updating taxi schedules we can decide only after the last *fixed* stop. If there is not any *fixed* stop the taxi is in *waiting mode* (in the last visited stop). In any case we have a position from which we can decide which road the taxi has to follow: we refer to this position as pd, position of departure. We compute the expected time of departure td from the position of departure, pd, with the following two formulas:

$$tr = \begin{cases} tp + T(p, ps_1) + t(s_1) + \sum_{i=1}^{n-1}[T(ps_i, ps_{i+1}) + t(s_{i+1})] & \text{if } n > 0 \\ tp & \text{if } n = 0 \end{cases} \tag{2.3.1}$$

where:

- n is the number of *fixed* stops to be still visited
- $T : P \times P \mapsto R_+$ is the function that gives the time needed to move from the first position to the second position
- $t : S \mapsto R_+$ is the function that gives the expected time to visit every stop according to its type: *departure*, *destination* or *check point*.
- tp indicates the time when the latest taxi position update p has been received
- p is the the position of the taxi according to the latest update received by the *active* taxi
- ps_i is the position of stop number i in the list of fixed stops to be still visited
- s_i is the stop number i in the list of *fixed* stops still to be visited
- tr indicates the time when the taxi will be ready to leave from the last fixed stop still to be visited

The second formula takes into account the time that the taxi has to wait for the updated schedules:

$$td = \max(tr, tu) \tag{2.3.2}$$

where tu is the expected time to give the taxi the updated schedules, that takes into account the computational time needed by the processes described in this chapter.

For every *active* taxi i we define pd_i and td_i as the related position pd and time td from where we have to decide the updated route for taxi i. The updated route for each *active* taxi i is constructed considering pd_i, td_i and all the *confirmed* and *accepted* requests. There are four sets of requests according to the state of requests:

1. The set C is the set of *confirmed* requests. These requests are still not *accepted* by the service. The *optimizer* has to evaluate if the *active* taxis can accept these requests or if it is necessary to activate an *available* taxi.

2. The set O is the set of *open* requests. These requests have been *accepted* by the service. The departure and the destination of an *open* request are in the schedule of an *active* taxi, i.e. the taxi that covers it, but the related stops are not *fixed*. For this reason, the taxi that will service these requests can still change.

3. The set A is the set of *assigned* requests. For these requests the taxi that will service them has been decided, since the stops related to their departure are *fixed*.

4. The set B is the set of *on board* requests. The users related to these requests are on board the taxis that are servicing them.

For each *active* taxi, i, we define the subsets $O_i \subseteq O$, $A_i \subseteq A$ and $B_i \subseteq B$ of requests respectively *covered* by, *assigned* to, *on board* of taxi i. We say that an *open* request r is *covered* by the *active* taxi i if a *departure* and a *destination* stops (*not fixed*) of taxi i are related to request r.

If we look forward to position pd_i, the taxi i will have visited all *fixed* stops of its schedule. We define the set $D_i \subseteq A_i \cup B_i$ as the set of request in $A_i \cup B_i$ with destination following pd_i. The routing of stops related to requests in D_i is fixed until pd_i and after it can change since the stops after pd_i are not *fixed*. The requests in the set $A_i \cup B_i \setminus D_i$ are serviced by taxi i before position pd_i and do not need to be considered in updating the taxi schedules.

The program of taxi i after position pd_i contains the *destination* stops of requests $\in D_i$ and *departure* and *destination* stops of requests $\in O_i$ and some *check point* stops. If we remove from this program the stops of type *check point*, we obtain the previous optimal permutation of position, po_i, related to destination of requests $\in D_i$ and position related to departure and destination of requests $\in O_i$. This permutation can be needed, as detailed in part *B)* point *6.*, in case no permutation of taxi i can serve the requests $\in D_i \cup O_i$ respecting all the constraints (this can be due to the lateness of taxi i with respect to his schedule). This permutation can be needed to guarantee the solvability of the problem, and hence to ensuring to service all the accepted requests.

Let H and J be sets of requests, we define $P(H, J)$ as the set of all the permutations of the position related to destination of requests $\in H$ and to departure and to destination of request in J. With this notation we can say that $po_i \in P(D_i, O_i)$.

For every *active* taxi i and every set of requests K, we define $F_i(D_i, K) \subset P(D_i, K)$ as the subset of *feasible* permutations for the taxi i. A permutation $c \in P(D_i, K)$ is feasible for the taxi i if taxi i can start from position pd_i at time td_i and visit all the positions in the sequence induced by c before the maximum visiting time associated to each position and the capacity constraint of the vehicle is satisfied. The system is fully dynamic, and hence all requests are supposed to be sent to the system when the user is ready to leave[2]. For this reason there are not lower bound on the visiting time for any stop. For *destination* stops there is an upper bound caused by the maximum arrival time of the user related to the stop. A permutation $c \in P(D_i, K)$ is feasible for taxi i, i.e. $c \in F_i(D_i, K)$, if $\forall k \in \{1, ..., n\}$, with $n = |D_i| + 2|K|$ the number of positions in the permutation c, the time constraint and the capacity constraint are satisfied.

1. Time constraint on user requests:

$$td_i + T(pd_i, c(1)) + t(c(1)) + \sum_{h=1}^{k-1} [T(c(h), c(h+1)) + t(c(h+1))] \leq u(k) \tag{2.3.3}$$

   where:

   - $c(k)$ is the k-position in the permutation c
   - $u(k)$ is the upper bound on the k-position in the permutation c considering $u(k) = +\infty$ if $c(k)$ is a departure.

2. Time constraint on taxi working time:

$$td_i + T(pd_i, c(1)) + t(c(1)) + \sum_{h=1}^{n-1} [T(c(h), c(h+1)) + t(c(h+1))] \leq wt_i \tag{2.3.4}$$

   where:

   - $c(k)$ is the k-position in the permutation c

[2]This choice is due to the fact that since the service is not relying on a fixed fleet of vehicles but on a dynamic fleet of taxis is not possible to ensure the availability of a vehicle considering the potential high demand for a taxi sharing service. This can be considered as a future improving of the service, that need to ask the taxi drivers their availability in advance, or to limit the number of accepted booked requests to a given number, in order to have a high probability of having enough taxi on duty to serve all the accepted booked requests.

- wt_i is the maximum working time of taxi i.

3. Capacity constraint:

$$us_i + \sum_{h=1}^{k} x(c(h))us(c(h)) \leq U_i \tag{2.3.5}$$

where:

- us_i is the number of users related to requests in D_i
- $x(s) = 1$ if the stop s is of type *departure* and $x(s) = -1$ if the stop s is of type *destination*
- $us(s)$ represents the number of user related to stop s
- U_i represents the capacity of taxi i

We want to select for each *active* taxi i a set of requests $K_i \in O \cup C$ and a permutation $s_i \in P(D_i, K_i)$ that satisfy:

1. $s_i \in F_i(D_i, K_i)$ or it is the previous permutation po_i.
2. $K_i \cap K_j = \emptyset$ if $i \neq j$
3. $O \subseteq \bigcup_{i=1}^{n} K_i$

The aim is to minimize an objective function that takes into account the quality of selected permutations $\{s_i\}$ and the number of requests $\in O \cup C \setminus \bigcup_{i=1}^{n} K_i$.

In the first point it is necessary to consider the previous permutation, even if it is not feasible, in order to guarantee the solvability of the problem. The second point prevents from servicing the same request by more than one taxi. The third point asks that all *open* requests are *covered* by *active* taxis.

The requests in $\in O \cup C \setminus \bigcup_{i=1}^{n} K_i$ are requests that will not be accepted by the *active* taxis. The *finder* process will try to assign them to an *available* taxi.

B) Permutation and group generation

In order to generate for every *active* taxi i all feasible permutations $s \in F_i(D_i, K)$ for every set $K \in O \cup C$ we use the following procedure:

1. defining the set of set G_i as an empty set.
2. defining the set of permutations S_i as an empty set.
3. generating all feasible permutations $F_i(D_i, \emptyset)$ allowing to service the *on board* and *assigned* requests of taxi i and adding the feasible permutations to S_i.

4. if there is at least one feasible permutation we add $\emptyset$ to G_i.

5. iterating on all requests $r \in O \cup C$ and:

   - for every permutation $s \in F_i(D_i, K)$ t.c. $K \in G_i$, we verify the possibility to insert the departure and the destination of r in all the possible ways in s.
   - we add all obtained feasible permutations, $F_i(D_i, K \cup r)$, to S_i
   - if $|F_i(D_i, K \cup r)| \geq 1$ we add the set $K_i \cup r$ to G_i.

6. if $O_i \notin G_i$, we add O_i to G_i and po_i to S_i.

G_i is the set of set $K \subset O \cup C$ *t.c.* $|F_i(D_i, K)| \geq 1$, we refer to G_i as the set of groups of requests that taxi i can cover.

Theoretically this procedure can lead to an exponential number of requests groups and associated feasible permutations. However, since all requests are imminent (due to the fact that there is no reservation), the probability that two different requests in $C \cup O$ can be serviced by the same taxi i is low, considering also the *on board* and *assigned* requests in D_i and, hence, the number of groups does not explode, as shown in chapter 4. Furthermore, from an application perspective, since these computations are independent from taxi i to taxi j the process is fully parallelizable.

C) Set partitioning

We have to assign to every permutation s a cost that takes into account the time needed by the taxi to service the requests and the quality of the service for the users. To compute the cost, we use the following formula:

$$c(s) = T(s) - \eta * M(s) \tag{2.3.6}$$

where:

1. $T(s)$ represents the time needed by the taxi to service permutation s;
2. $M(s)$ is the sum, on all the destinations included in s, of the difference between the maximum arrival time to destination (according to user requests) and the expected arrival time of the taxi;
3. η is a positive parameter.

For every group K of requests, *covered* by a taxi i, we assign the cost according to the following formula:

$$c(K) = \min_{s \in F_i(D_i, K)} c(s) \tag{2.3.7}$$

and we define the permutation $\overline{s}$ that realizes the minimum as the *active* permutation associated to group K.

We define the cost of not accepting a *confirmed* request $r \in C$ with the following formula:

$$q(r) = \theta * length(r) \tag{2.3.8}$$

where θ is a parameter and $length(r)$ is the time needed to travel from the departure to destination of request r.

We introduce the ILP model that has to be solved in order to choose the optimal permutation for each *active* taxi.

Sets:

- O: *open* requests
- C: *confirmed* requests
- $R = O \cup C$
- T: *active* taxis
- G_i: groups $\forall i \in T$

Parameters:

- $a_{rgi} = \begin{cases} 1 & \text{if group } g \in G_i, \text{ of taxi } i \in T, \text{ cover } r \in R \\ 0 & \text{otherwise} \end{cases}$
- $c_{gi} =$ group cost $g \in G_i$, of taxi $i \in T$
- $q_r =$ cost of do not cover $r \in C$

Variables:

- $x_{gi} = \begin{cases} 1 & \text{if group } g \in G_i, \text{ of taxi } i \in T \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$
- $y_r = \begin{cases} 1 & \text{if requested } r \in C \text{ is refused} \\ 0 & \text{if requested } r \in C \text{ is accepted} \end{cases}$

Objective function:

$$\min \sum_{i \in T} \sum_{g \in G_i} c_{gi} x_{gi} + \sum_{r \in C} q_r y_r \tag{2.3.9}$$

Constraints:

$$\sum_{g \in G_i} x_{gi} = 1, \ \forall i \in T \tag{2.3.10}$$

$$\sum_{i \in T} \sum_{g \in G_i} a_{rgi} x_{gi} = 1, \ \forall r \in O \tag{2.3.11}$$

$$\sum_{i \in T} \sum_{g \in G_i} a_{rgi} x_{gi} + y_r = 1, \ \forall r \in C \tag{2.3.12}$$

The first constraint ensures that exactly one group is selected for each taxi. The second constraint ensures that every *open* request is covered by an *active* taxi. The third constraint ensures that every *confirmed* request is either covered by an *active* taxi or it is refused.

The set partitioning is a $NPhard$ problem [26] and, hence, the time to solve the problem to optimality can be extremely large if the number of variables is high. However, from an application perspective, it is possible to start the optimization from an initial feasible solution selecting $K_i = O_i \ \forall i \in T$ and to give to the *optimizer* a time limit for the optimization. If the time limit is reached before having found the optimal solution, the incumbent solution is accepted. In the analyzed scenarios, presented in chapter 4, even though the number of variables reaches 17268, it has not been necessary to set a time limit, since the optimal solution was always found in few seconds due to the fact that the ratio of $a_{rgi} = 1$ is low, since the problem is highly constrained as detailed in chapter 4.

The requests $r \in C$ are not accepted by *active* taxis if in the solution $y_r = 1$, and in this case their state is updated to *for available*. The requests $r \in C$ are accepted by *active* taxis if in the solution $y_r = 0$, and in this case their state is updated to *open*. For every *open* request we update the associated taxi according to the group covering it.

For each *active* taxi i we have selected a group K_i for which $x_{gi} = 1$, and to this group is associated an active permutation $\overline{s_i}$ that realizes the minimum cost: we call it as the chosen permutation of taxi i. In the next paragraph we describe how to obtain from the chosen permutations the updated schedules for each *active* taxi.

D) Construction of new schedules

In order to generate new groups and permutations, it is necessary to know where the taxi will be in the short future, since the present position of a taxi would be out of date during the time needed by the processes presented in this chapter, and hence the new schedule may not be feasible according to the new position. To know where the taxi would be in the future, a first option could be to consider the first departure/destination position in its schedule, and to construct from there new permutation. However the first departure/destination position can also be not closed in time and space to the present position, and this would imply to fix the schedule of the taxi for a long time, loosing the possibility to serve other users on the way. For these reasons, in order to improve the flexibility of the service, if two consecutive positions in the chosen permutation of some taxi are too far away, one or more *check points* are inserted in the middle.

We want two consecutive stops in the schedule to have a distance of maximum T_{max} time within them. We compute the number of check points to be inserted within two consecutive positions of the chosen permutation according to the formula:

$$N_{cp} = \lfloor \frac{T(p_i, p_{i+1})}{T_{max}} \rfloor \tag{2.3.13}$$

and the time distance between two stops is computed with:

$$t_{cp} = \frac{T(p_i, p_{i+1})}{N_{cp}} \tag{2.3.14}$$

Once we have the number of check points N_{cp} and the time distance between two consecutive stops t_{cp}, we generate the stops in the middle with position on the shortest path respecting the computed time distance between stops.

This process is done for every pair of consecutive positions in the chosen permutation $\overline{s_i}$ of every *active* taxi in order to obtain updated schedules for every *active* taxi.

2.3.2 Taxi management

In this section we present from the second to the last step of the *optimizer*. Differently from the first step, they involve one taxi at a time.

2) Closing taxis This process evaluates whether, after the schedules updating, there are some *active* taxis without any stop to visit. In this case, it means that in the optimal solution, computed in the last optimization cycle, this taxi has no role in servicing *open* and *confirmed* requests. This happens because there are not suitable requests for the taxi, according to its position, or because the taxi is close to its maximum working time and the taxi driver can not satisfy any request before the end of his/her working shift. Regardless of the reason, this process ends the ride of every *active* taxi without any stop in its schedule, and the taxi enters the *end of the ride* mode.

3) Dealing exceptions If some *active* taxi has thrown some exceptions of type *interrupted* or *no-show*, as described in details in section 1, this process manages them. The aim of this process is to update their schedules taking into account that one or more stops are not necessary to be visited any more. The needed steps are the following:

1. we consider the actual position of the taxi and the actual time and we generate the set P_f of all the feasible permutations of the positions related to requests *on board*, *assigned* or *covered* by the taxi.
2. we choose the permutation in P_f with the minimum cost computed as described in section 2.3.1.
3. once we have the chosen permutation, we add in the middle some check points, if necessary, as described in section 2.3.1, and update their schedules; hence those taxis exit the *waiting* mode and enter the *navigation* mode if their schedule has at least one stop or, otherwise, they enter the *end of ride* mode.

4) Deleting canceled requests If a request had been *canceled* by users we need to update the schedule of the *active* taxi covering it. We can change the schedule of the taxi only after the last *fixed* stop. For *fixed* stops that are referred to the *canceled* request, we change their type into *check point*. To update the schedule we execute the following steps:

1. we consider the positions of departure Pd and the time of departure td as described in section 2.3.1;
2. we generate the feasible permutation $F_i(D_i, O_i)$ removing the *canceled* request from the *assigned* or from the *covered* requests;
3. we choose the permutation $s \in F_i(D_i, O_i)$ with the minimum cost computed, as described in section 2.3.1;
4. once we have the chosen permutation, we add in the middle some check points if necessary, as described in section 2.3.1, and we update the schedule and the request state to *canceled*.

5) Considering visited stops This process updates the state of requests related to stops that were visited by some *active* taxis. We can have the following cases:

1. if the visited stop is of *departure* type, we update the state of the respective request to *on board*;
2. if the visited stop is of *destination* type, we update the state of the respective request to *serviced*;
3. if the visited stop is of *check point* type, we do nothing.

Once a stop has been considered, it is removed from the schedule of the taxi.

6) Fixing new stops This process establishes the stops within the schedules of *active* taxis in order to prevent the taxi from visiting the last *fixed* stop, since, in this case, the taxi would enter the *waiting* mode and so losing time. At once, it is important to avoid to fix too many stops, otherwise the flexibility of the service, in updating the optimal strategy to service the requests, would be reduced. To fix new stops we use the following procedure:

1. we compute the maximum time related to the next optimization cycle in which the *optimizer* will fix new stops as: $T_{next} = pt + mc$, where pt is the present time and mc is the maximum cycle time[3].
2. we compute for every stop s in the schedule the earliest arrival time of the taxi to stop s: E_S considering the taxi moving at speed limit from the latest updated position to the stops in the order specified by the schedule.

[3] In the simulator described in section 3, every cycle has the same length and hence $mc = dt = 30$ seconds in every cycle.

3. we fix the stops s for which: $E_s \leq T_{next}$.

4. if a *fixed* stop is of *departure* type we update the status of the related request from *open* to *assigned*.

In this way we are reasonably sure that the taxi will not visit the last *fixed* stop before the time in which the process of the next optimization cycle will fix other stops. The *fixed* stops are forwarded to the taxi drivers with the details of the possible new requests *assigned* to the taxi.

2.4 The finder module

In this section we describe the third process involved in finding *available* taxis for the requests that have not been accepted by *active* taxis.

The output of this process is:

- to select *available* taxis to send the request of starting a new ride;
- to produce the schedules that allow selected taxis to service the related requests.

The structure of this section is similar to section 2.3.1, since the aim of both is to construct schedules. The main differences with the *optimizer* result from the fact that the *available* taxis have not *on board* and *assigned* requests and that schedules have to be constructed only for the selected *available* taxis. This section is divided in four steps:

A) In the first part we define the mathematical problem starting from the data about *available* taxis and *for available* requests;

B) In the second part we show how to compute all the possibilities to service the requests *for available* taxis;

C) In the third part we detail the ILP model needed to cover optimally the requests *for available* taxis;

D) In the fourth part we describe how to obtain, from the solution of the mathematical problem, the schedules for the selected *available* taxis.

A) From data to mathematical problem

For every *available* taxi the position, the parking area in which the taxi is queued and the time when the taxi had been queued are known.

Taxis can start their ride at time t, which represents the expected time to give taxis their schedules. For each taxi i, we define p_i and t as the information about position and time from where we have to construct their routes.

Let K be a set of requests, we define $P(K)$ as the set of all the permutations of the position related to the departure and destination of requests in K. We

say that a permutation $c \in P(K)$ is *feasible* for taxi i, and we write $c \in F_i(K)$, if taxi i can start from position p_i at time t and visits all stops in the sequence c before the maximum visiting time associated to each stop and the capacity constraint of the vehicles is respected.

We define the set V as the set of requests with state *for available*, indicating the requests that the *finder* has to assign to *available* taxis. The problem consists in selecting which *available* taxi is to activate and in selecting for every *activated* taxi i a set of requests $K_i \in V$ and a permutation $s_i \in F_i(K_i)$ that satisfy:

1. $K_i \cap K_j = \emptyset$ if $i \neq j$

in order to minimize an objective function that takes into account the quality of selected permutations $\{s_i\}$ and the number of requests in $V \setminus \bigcup_{i=1}^{n} K_i$.

The requests in $V \setminus \bigcup_{i=1}^{n} K_i$ are requests that can not be accepted by *available* taxis and, hence, are *rejected* by the service.

B) Permutation and group generation

In order to generate for every *available* taxi i all feasible permutations $s \in F_i(K)$ for every set $K \in V$, we use the following procedure:

1. define the set of set G_i as an empty set;
2. define the set of permutations S_i as an empty set;
3. we add $\emptyset$ to G_i and the trivial permutation with zero positions to S_i.
4. iterate on all the requests $r \in V$ and:
   - for every permutation $s \in F_i(K)$ t.c. $K \in G_i$, we verify the possibility to insert the departure and the destination of r in all the possible ways in s.
   - we add all the obtained feasible permutations, $F_i(K \cup r)$ to S_i
   - if $|F_i(K \cup r)| \geq 1$ we add the set $K \cup r$ to G_i.
5. we remove $\emptyset$ from G_i and the trivial permutation with zero positions from S_i.

G_i is the set of set $K \subset V t.c. |F_i(K)| \geq 1$; we refer to G_i as the set of groups of requests that taxi i can cover.

C) Set partitioning

We have to assign to every permutation s a cost that takes into account the time needed by the taxi to serve the requests and the quality of the service for the users. We compute the cost with the same formula defined in 2.3.1.

$$c(s) = T(s) - \eta * M(s) \tag{2.4.1}$$

For every group K of requests covered by a taxi i, we assign the cost according to the following formula:

$$c(K) = \min_{s \in F_i(K)} c(s) \tag{2.4.2}$$

and we define the permutation $\overline{s}$ that realizes the minimum as active permutation associated to group K.

We define the cost of not accepting a *confirmed* request $r \in V$ with the following formula:

$$q(r) = \delta * length(r) \tag{2.4.3}$$

where δ is a parameter and $length(r)$ is the time needed to travel from the departure to destination of request r. We underline that the parameter δ is a different parameter from θ which is used by the *optimizer* to compute the cost of not accepting a request with the *active* taxis; each of them have to be set independently.

We introduce the ILP model that it is necessary to solve in order to choose which taxi has to be activated and the optimal permutation for each activated taxi.

Sets:

- V: requests *for available* taxis
- T: *available* taxis
- G_i: groups $\forall i \in T$

Parameters:

- $a_{rgi} = \begin{cases} 1 & \text{if group } g \in G_i, \text{ of taxi } i \in T, \text{ cover } r \in V \\ 0 & \text{otherwise} \end{cases}$
- $c_{gi} =$ group cost $g \in G_i$, of taxi $i \in T$
- q_r: the cost of not accepting request $r \in V$
- p_i: identify the parking area of taxi $i \in T$
- n_i: identify the position of taxi $i \in T$ in the parking area p_i

Variables:

- $x_{gi} = \begin{cases} 1 & \text{if group } g \in G_i, \text{ of taxi } i \in T \text{ is selected} \\ 0 & \text{otherwise} \end{cases}$
- $y_r = \begin{cases} 1 & \text{if request } r \in V \text{ is } \textit{refused} \\ 0 & \text{if request } r \in V \text{ is } \textit{accepted} \end{cases}$

- $z_i = \begin{cases} 1 & \text{if taxi } i \in T \text{ is } \textit{activated} \\ 0 & \text{otherwise} \end{cases}$

Objective function:

$$\min \sum_{i \in T} \sum_{g \in G_i} c_{gi} x_{gi} + \sum_{r \in V} q_r y_r \tag{2.4.4}$$

Constraints:

$$\sum_{g \in G_i} x_{gi} = z_i, \ \forall i \in T \tag{2.4.5}$$

$$\sum_{i \in T} \sum_{g \in G_i} a_{rgi} x_{gi} + y_r = 1, \ \forall r \in V \tag{2.4.6}$$

$$z_i \leq z_j, \ \forall i, j \in T \text{ t.c. } p_i = p_j \text{ and } n_j < n_i \tag{2.4.7}$$

The first constraint ensures that at most a group can be selected for every taxi, and that the taxi is activated if and only if one of its groups is selected. The second constraint ensures that every request is either covered by a taxi or refused. The third constraint ensures that the queued time is respected at each parking area.

Taxis $i \in T$ are chosen to be activated if in the solution $z_i = 1$, and in this case their state is updated to *in activation*. The requests $r \in V$ are not accepted by the service and hence are *rejected* if in the solution $y_r = 1$. The requests $r \in V$ are *covered* by some selected *available* taxi if in the solution $y_r = 0$, and in this case their state is updated to *for in activation*.

D) Construction of new programs

From the optimal permutation of the selected groups of *in activation* taxi the schedules for taxi drivers are constructed according to the procedure described in section 2.3.1, with the difference that the first stop of each schedule is *fixed* as default.

At this point the request of starting a ride is forwarded to the chosen *available* taxis. If a *available* taxi accepts the ride its state becomes *active*, the *covered* requests are accepted, and their state is set to *open* (or to *assigned* if the stop related to the departure is *fixed*). If an *available* taxi does not accept the ride, its state is updated to *enrolled* and the request state is reset to *for available*.

Chapter 3

Agent-based simulation

In this chapter we describe the Taxi Sharing simulator which allows to forecast how the Taxi Sharing service will work taking into account citizens, taxi drivers and the municipality. For the client what it is important is how much time he/she will have to wait for the taxi, how much time the detours to serve other clients on the same vehicle will take, and how much the fare is. For the taxi driver what is important is how many clients the taxi will serve each hour and how much he/she will earn from the Taxi Sharing activity. For the municipality, it is important to know which proportion of the transportation demand can be serviced by the service, how much it can help in reducing the use of private cars in urban context, and whether the service needs to be subsidized. To answer these questions, we developed an agent based Taxi Sharing simulator; we refer to [27, 28, 29] as examples of agent based simulation methods in the transportation field.

The developed simulator enables the authority of cities to forecast how Taxi Sharing could work from a quantitative perspective, with respect to users, taxi drivers and to the municipality. The simulator needs all the data which are necessary to run the service mentioned in the previous chapter: road network, taxi features and parameters; furthermore, the simulator needs some data to simulate the behavior of users and taxi drivers agents. In section 3.1 we present the main features of the simulator. In sections 3.2 and 3.3 we detail respectively how the interaction of users and taxi drivers with the service is simulated. In section 3.4 we present how the simulator can be used to have a trial of the service and, finally, in section 3.5 we detail which statistics the simulator provides on the quality, effectiveness, and efficiency of the service.

3.1 Characteristics

The simulator is designed to run simulations of the service for 24 hours, in order to have a complete overview of the performance of the service in the different periods of the day. To enable the simulator to run the simulations of 24 hours of

service in less then 24 hours, the simulator runs in cycle the following processes:

1. *answerer* process
2. *optimizer* process
3. *finder* process

One cycle simulates $dt = 30$ *seconds* of service and, hence, it is enough to run $n = \frac{24*3600}{30} = 2880$ cycles to simulate 24 hours of service.

All data about the road network and parameters are supposed to be constant within each hour of the day and potentially different among different hours, as it is presented in chapter 4.

3.2 Simulation of customers

According to the user cases presented in section 1.1, the user behavior is simplified:

- we simulate *Case U1: asking for the ride conditions*;
- we suppose the user always sends the request according to *Case U2: sending the request*;
- we do not simulate *Case U3: deleting the request*;
- we do not simulate *Case U4: interrupting the ride.*

In *Case U2* we suppose that user always sends the request. This is equivalent to simulate only the *confirmed* requests, i.e. those that have to be serviced by taxi sharing. We do not simulate the process of confirming/not confirming a ride according to maximum arrival time to destination and price, since we have not developed a elasticity model of user's confirmation ratio with respect to price and time.

Case U3 is not simulated since it is supposed to be rare cases, because the service is without reservation.

Case U4 is not simulated, but we underline that from a service performance perspective it will increase the efficiency of the service, and the quality for other users, since the ride is already paid and the taxi is free from the task to visit the destination of the user when he/she interrupts the ride.

The expected number of requests $E_h[n_r]$ for every hour of the day has to be given as an input to generate the number of requests arriving to the service every cycle, $dt = 30$ seconds, according to a Poisson distribution with $\lambda = \frac{E_h[n_r]*dt}{3600}$ that represents the expected number of requests that arrives every dt seconds.

For every request it is necessary to generate the departure point and the destination point. To this aim, it is necessary to upload in the simulator the following data[1]:

[1] In chapter 4 we present the details of the data used for the feasibility study for the city of Milan.

- a set of zones that cover the road network;
- for every hour of the day t, a origin/destination matrix OD_t, where $OD_t[i, j]$ rapresents the probability that the request has the departure in zone i and the destination in zone j.

According to the hour of the day, t, the departure and destination zones of the request are generated with matrix OD_t. Once the zones are selected, in order to generate the departure and destination points, they are sampled with uniform distribution within all the point of the arcs assigned[2] to the selected zones.

Finally we generate the number of users for every request according to a discrete distribution with probability $p_i \in [0; 1]\ \forall i \in \{1, ..., n_{max}\}$, where p_i represents the probability that the request is for i users and n_{max} represents the maximum number of users allowed by the service.

3.3 Simulation of taxis

According to the taxi cases presented in section 1.2, also the taxi behavior is simplified:

- we simulate *Case T1: becoming available mode* as it is;
- we simulate *Case T2: becoming active mode* supposing that taxis always accept the ride[3];
- we simulate *Case T3: navigation mode* supposing that on board users never ask to leave the car before arriving to destination and, hence, we do not consider *Case T6: ride interrupting mode*;
- we simulate *Case T4: boarding user mode* supposing that users are always present and confirm the details of the ride[4];
- we simulate *Case T5: disembarking user mode*, *Case T7: waiting mode* and *Case T8: end of the ride mode* as they are described in section 1.2.

It is possible to enhance the simulator to consider additional cases, if it is necessary to test the impact on the performance of the service following possible exceptions, but we didn't explore these possibilities for the reasons above presented.

[2] Each arc is assigned to the zone containing the longest part of the arc.

[3] If a taxi driver do not accept a ride, this mean that he forgot to remove his availability, and hence he is not really *available*. We simulate only taxi that are really *available* and hence ready to accept a ride.

[4] We suggest that the fee of the ride is paid by the user also in case of no show or wrong information that lead the user to be uninterested to the booked ride. With this assumption the efficiency of the service increase if some user does not show up, or is not interested in the ride, since the revenue is the same, but the taxi has not to visit the destination of the user request.

The number of taxis $ntaxi_h$ for every hour of the day has to be given as an input, in order to generate new *available* taxis when the sum of the number of *active* taxis and *available* taxis is less than $ntaxi_h$. When a new *available* taxi is generated it is chosen randomly among the taxis that are *enrolled* but are not already *available* neither *active*. The taxi position is selected randomly on the arcs of the network and the maximum time for ending the taxi ride is set to be t_{max} after the actual time.

To simulate taxi movement on the road network during *navigation* mode, we suppose that the taxi moves on the road network at the average speed of the arc on which the taxi is. Traffic lights and other cars on the network are not explicitly considered since both aspects are included in the average speed. In this way the actual position of the *active* taxi is updated following the shortest path in the direction of the first stop that has to be visited.

To simulate the *boarding* and *disembarking* user mode a fixed time bt is supposed to be needed for servicing the users. After bt seconds that the taxi has reached a stop associated to a departure or a destination, this stop is marked as visited.

3.4 Interactive simulation of requests

The simulation can run in real time in order to monitor on the screen the ongoing simulation, visualizing the waiting people and the moving taxis, the positions of which are updated every second, in Figure 3.1 it is reported a screenshoot.

It is possible to zoom in and out to focus the attention on a specific area or to have an overview of the service. The users requests, the taxis' availability and the speed of taxis on the network are simulated according to uploaded data.

To see how the service reacts to a specific request, it is possible to insert a request selecting on the monitor the desired departure and destination point. The simulator answers with the maximum arrival time to destination, and with the fare of the ride. If the request is accepted by the service, it is possible to monitor on the screen when the taxi arrives to the departure point to pick up the user, which route it follows to pick up or deliver other users, and when it reaches the destination. This real time functionality of the simulator allows to have a trial of the service.

3.5 Results

While a simulation is executed, relevant data about users and taxi drivers are recorded. For each user ride during the simulation the number of users, the positions of departure and destination, when the taxi ride is asked, when the taxi arrive, when the destination is reached, the detour time and the fare of the ride is recorded. An estimation of travel time from origin to destination with an individual taxi, compared to the travel time with Taxi Sharing allows to compute how much time has been lost due to detours done for servicing other

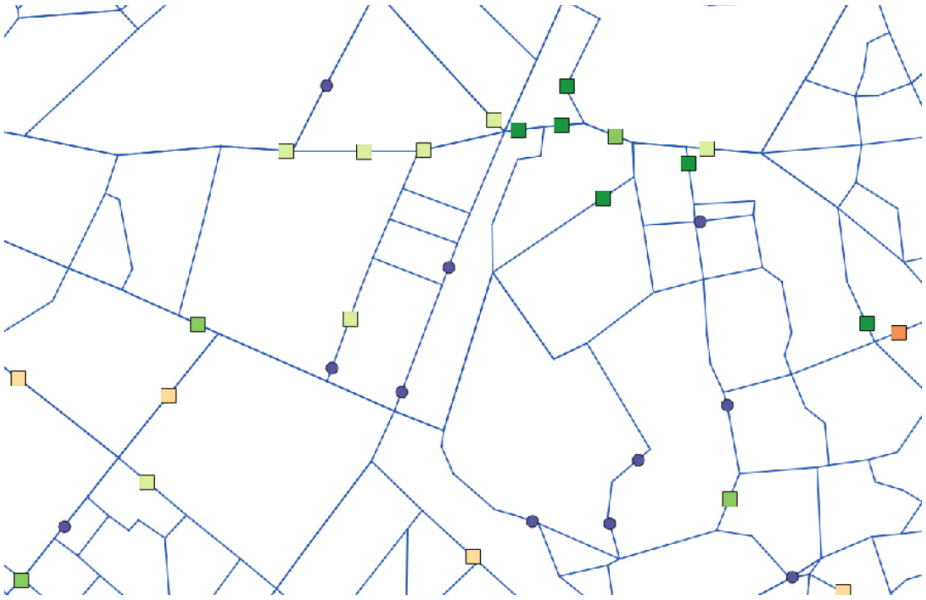


Figure 3.1: Taxi Sharing vehicles are presented as square, with a color depending on the number of users on board; waiting users are blue circles.

clients on board. For each taxi ride, the simulator records from which parking and when it starts, when it finishes, how many clients it services and the revenue for the driver.

These data for all users rides and all taxis rides allow to compute statistics with respect to users and taxi drivers. With respect to users, the simulator reports the number of serviced users, the average waiting time and the average detour time. This information is provided in an aggregate form and splits spatially and temporally, since the quality of the service can depend on the area of the city or on the hour of the day. With respect to taxi drivers, simulator reports: the number of taxi rides, the average number of serviced users per hour, the relevant average revenue per hour and the average length of the taxi ride. This information is provided in an aggregate form and splits spatially and temporally, since the efficiency of the service can depend on the area of the city or on the hour of the day.

To compare the efficiency of Taxi Sharing in different scenarios and with that of individual taxi, the simulator computes an efficiency index that we define with the following formula:

$$e = \frac{Rs}{Rt} \tag{3.5.1}$$

where:

- $Rs = \sum_{r \in S} D(o_r, d_r)$ represents the sum of the length of all the serviced requests;
- $Rt = \sum_{t \in T} d(t)$ represents the total sum of traveled distance of all taxis;

The efficiency is a index related to the amount of served request per unit of traveled distance. Traveled distance reflects use of resources in terms of human work and energy used, as well as pollution produced.

In chapter 4 we report details about the data utilized and the results obtained from three different simulations executed for the Municipality of Milan using the simulator presented in this chapter.

Part II

A feasibility study for Milan

Chapter 4

Development scenarios for Taxi Sharing

The feasibility study for the city of Milan has been done in collaboration with the mobility agency of the Municipality of Milan, AMAT, in order to evaluate how Taxi Sharing could work. The interest of the Municipality for this service is to increase the sustainable mobility service in order to reduce the dependency on private cars and to enhance the offer of public transportation.

The city of Milan has a surface of 182 Km^2, the population is 1.263.000, with a commuting population of about 850.000. In the age structure of the population of the city, the pyramid reaches its peak for the age range 40-44 years, both males and females. These data must be considered in a general frame of aging of the Italian population. Along with Germany, Italy is worldwide second only to Japan in the increase rate of population aged 65 years or more [36].

Taxi Sharing service can be suitable to the city of Milan for the following reasons: the aging society, the high price of individual taxis, the aim of reducing the use of car within the city and, at the same time, to enhance the offer of mobility services. The current taxi fleet, consisting of 4855 taxis, it is not fully exploited, since taxi drivers often lose time at the parking areas waiting for a client and, on the other hand, during special events the demand is high and the fleet is not able to service all requests. These aspects are inefficient both for taxi drivers and users and have an answer in Taxi Sharing. The efficiency of the individual taxi, as defined in section 3.5, has been computed, from data about individual taxi ride in Milan to be 42% and, hence, quite low.

The Municipality of Milan is facing the challenge of reducing by 50% the use of private cars in the city of Milan, according to the referendum about urban mobility approved, with 79% of consent, on June 13th, 2011. To enhance the mobility services it is crucial to use in the most efficient way all available resources.

In the executed simulations Taxi Sharing can be used to go from every point to every point of the city of Milan 24 hours a day. In section 4.1 we present a

simulation with a level of demand for the Taxi Sharing around 0.5% of the total mobility, which represents the use of car sharing system in the city of Milan[1]. In section 4.2 we present a simulation with a level of demand for the Taxi Sharing around 2% of the total mobility, which represents the use of individual taxi in the city of Milan[2]. In section 4.3 we present a simulation with a level of demand for the Taxi Sharing around 10% of the total mobility; this scenario represents the potentiality of the service.

In the next paragraphs we present the data used in the simulations and the parameters set to the same values in all scenarios according to the optimization algorithm and the simulator.

Data

The road network graph is the base on which the simulation is executed, since the departure and destination points of requests are on the road network, and the taxis are waiting and moving on the road network. We use a graph[3] of the city of Milan with all the information detailed in section 2.1. Based on size of the graph, $|N| = 14021$ and $|E| = 26873$ we decide to compute in advance all traveling times between all pairs of arcs[4].

To simulate Taxi Sharing requests we use an origin destination matrix, related to a division of the municipality of Milan in 373 zones. For every hour t, the matrix of the total demand of mobility, TOD_t, made by the municipality of Milan in 2005 and successively updated, specifies on statistical base how many trips are made from each zone to any other zone. To obtain the origin/destination probability matrix for taxi sharing requests, as detailed in section 3.2, we use the following formula[5]:

$$OD_t[i,j] = \frac{TOD_t[i,j]}{\sum_i \sum_j TOD_t[i,j]} \tag{4.0.1}$$

In Figure 4.1 we present the level of total mobility inside the city of Milan according to the hour of the day.

The use of Taxi Sharing, in respect of the total demand, has been estimated to change during the day, being subject to different levels of traditional public transportation offer, as shown in details in the next sections. The spatial distributions of requests of the Taxi Sharing in different areas of the city, has been supposed to be as the total mobility[6].

The number of taxi working in each different hour of the day has been computed according to the working shift of each taxi driver of the city of Milan.

[1] AMAT elaboration on Car Sharing data

[2] AMAT elaboration on Taxi data

[3] AMAT road network graph elaboration

[4] The space needed to store all the distance is equal to 2.9 GB, and the time needed to compute all the distances using 8 process in parallel mode is 33 minutes.

[5] With this approach we suppose the demand for Taxi Sharing to be a ratio of the total mobility demand but having the same origin/destination distribution.

[6] According to the matrix OD_t

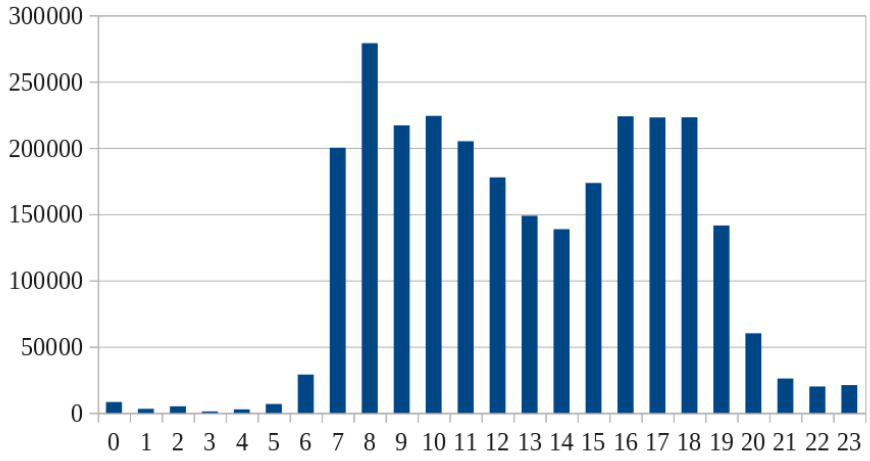


Figure 4.1: Number of movements within the city of Milan according to the hour of the day.

In Figure 4.2 we present these data. The number of taxi driver offering Taxi Sharing service in the different hour of the day had been computed as a ratio of the number of taxi drivers as shown in the next sections.

The average speed of the taxi on the road network has been computed from gps data of taxi drivers in an aggregated form. We use the computed average speed of every hour on all the road network. It would be a problem to run the service using this type of data, since the prediction of travel time of each arc it is important to the optimization of the service. For simulation purpose, since the aim is to test average performance of the service, we judge that the drawback of this assumption is negligible. The Figure 4.3 presents the average speed (Km/h) in each hour of the day.

We suppose all the taxis to have $capacity = 3$, considering one user on the front seat and two user on the rear seats, to allow each user to have his/her own space, and board and alight the vehicle from his/her own car door without the need to bother other users.

Parameters

In this section we report the parameters that we have set to the same value in all the simulations according to the optimization algorithm and to the simulator.

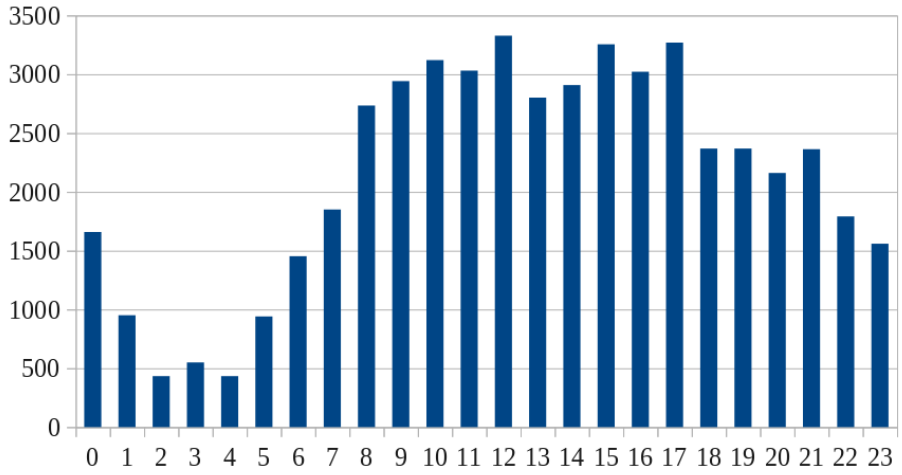


Figure 4.2: Number of working taxis in the city of Milan according to the hour of the day.

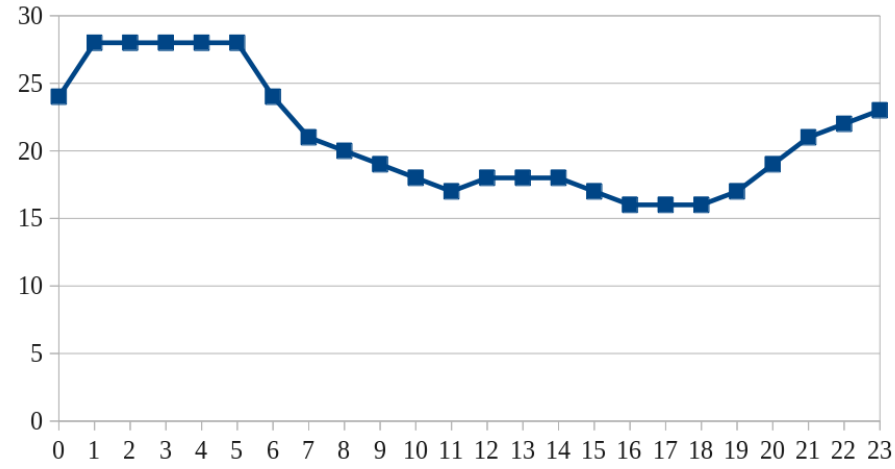


Figure 4.3: Average speed Km/h in the city of Milan according to the hour of the day.

According to the simulator we set:

- the parameters used to generate the number of users of each request[7]:

$$p_i = \begin{cases} 0.8 & \text{if } i = 1 \\ 0.2 & \text{if } i = 2 \end{cases} \tag{4.0.2}$$

- the parameters used to set the working shift of taxi drivers[8]: $t_{max} = 3\ hours$
- the parameters used to simulate the boarding and disembarking times[9]: $b_t = 30\ seconds$

According to the algorithm we set:

- the function that gives the needed time at each stop, coherently with the respective function used in the simulator:

$$t(s) = \begin{cases} 30\ seconds & \text{if } s \text{ is a departure or a destination} \\ 0 & \text{if } s \text{ is a checkpoint} \end{cases} \tag{4.0.3}$$

- the parameter used to compute the cost of each permutation: $\eta = 0.5$
- the parameter used to compute the cost of not accepting a request with the *active* taxis: $\theta = 2$
- the parameter used to compute the cost of refusing a request: $\delta = 4$
- the parameter used to compute the number of checkpoints to be inserted between two stops: $T_{max} = 120\ seconds$

We set these parameters, to fix value in every hour of the day, in all the presented, simulation since we found experimentally that they are averagely effective. A further research step is the dynamic adaption of parameters according to service conditions. This can improve sligthly the efficiency of the service, since we found little change in service performances, according to these parameters calibration. What influence strongly the efficiency of the service is the speed on the road network, the demand level, and parameters α and β.

[7] This value as been set coherently with the assumption that most of the ride are individual trip, and considering that three people can not book taxi sharing since the capacity has been set to three and hence would not be possible to share the ride with other users. We consider that an option could be to allow groups of three people to use taxi sharing with the rule that them would use rear seat, letting the front seat free for another user. In this way three people would occupy the place that normally is occupied by maximum two users.

[8] This value come from conversations engaged with some taxi drivers about the maximum time of a taxi sharing ride.

[9] This time has been estimated considering that as a urban mobility system most of the ride are without luggage, and the assumption that the payment method is cashless.

The effects on the service of these value are assessed in the three scenarios presented in the next sections and in appendix A where we present a focus on the calibration parameter β.

In each one of the next sections we detail the values of parameters that depend on the simulation and we present the obtained results.

4.1 Low demand scenario

In this scenario we test the performance of Taxi Sharing supposing the lowest level of demand of the three presented scenarios. In the next sections we present the additional parameter used for this scenario, the results obtained concerning the performances of the service and the details on the computational effort needed to optimize the service.

4.1.1 Parameters

The use of Taxi Sharing, in respect to the total demand, has been estimated to change during the day according to different levels of traditional public transportation offer, as shown in Figure 4.4.

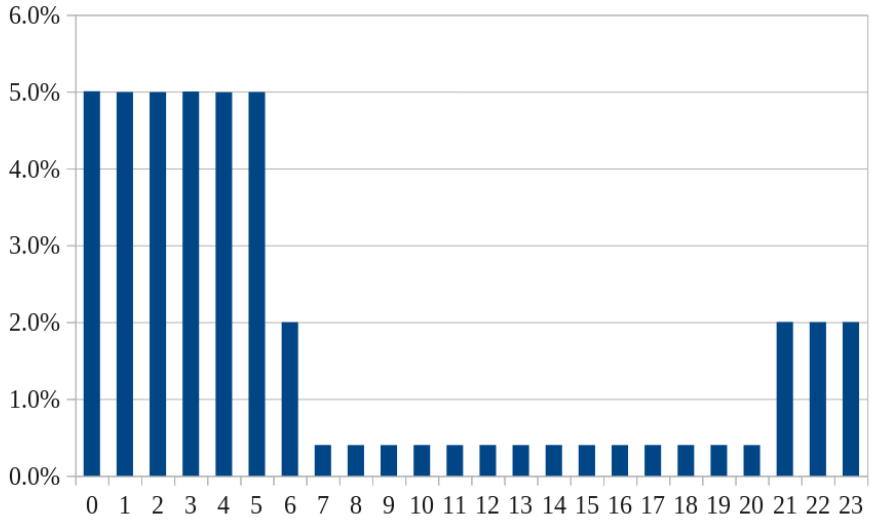


Figure 4.4: Use of Taxi Sharing in respect of the total mobility according to the hour of the day.

The total demand of mobility inside the city of Milan, presented in Figure 4.1, has been multiplied by the proportion of Taxi Sharing use to obtain the number of Taxi Sharing users per hour according to the hour of the day

as shown in Figure 4.5. The daily number of expected users is 13811 which corresponds to 0.5% of the daily mobility inside the city of Milan.

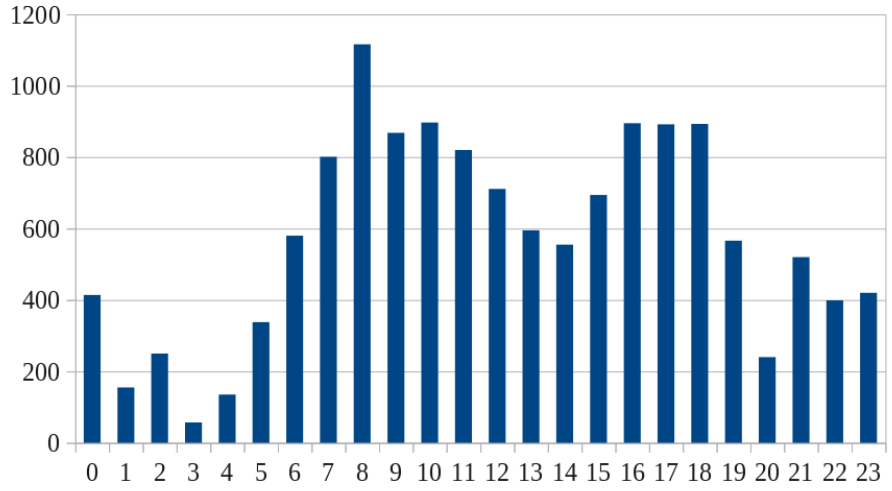


Figure 4.5: Taxi Sharing demand according to the hour of the day.

We suppose that each hour 10% of taxi drivers on duty offers Taxi Sharing service. With this assumption and the data about the number of taxi drivers listed in Figure 4.2, we compute the number of taxi drivers offering Taxi Sharing service according to each hour of the day, as presented in Figure 4.6.

The parameters α and β, used to compute the maximum arrival time to destination, as described in section 2.2, are set to have the values represented respectively in Figure 4.7 and in Figure 4.8. These parameters are quite high, hence in the worst cases the time of the ride can be high in respect to the direct trip. Nevertheless they have to be high, since the level of demand is quite low, in order to likely combine different users requests together. We present in appendix A a focus on the calibration of β on service quality and on computational complexity.

We set the parameters used to compute the cost of each ride, as presented in section 2.2, as follows: $\gamma = 3$ and $\delta = 0.8$. The cost of a ride for one user, according to the length of the ride in Km, is presented in Figure 4.9.

4.1.2 Service performances

Using the data presented in the previous section, we run a 24 hours simulation of Taxi Sharing within Milan. In the next paragraphs we present the obtained results of the simulation of the service according to users and to taxi drivers.

The number of serviced users is 13724 and the number of refused users is 127. This value represents 1% of the total demand; hence it is really low also

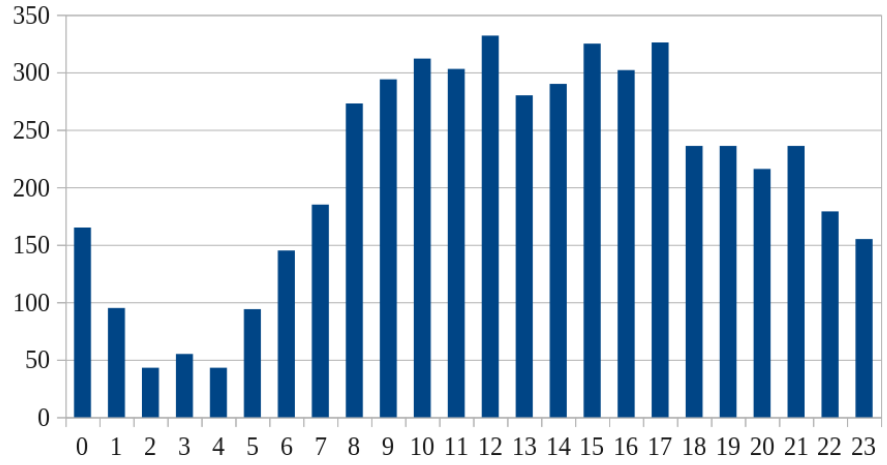


Figure 4.6: Number of taxi drivers offering Taxi Sharing service according to the hour of the day.

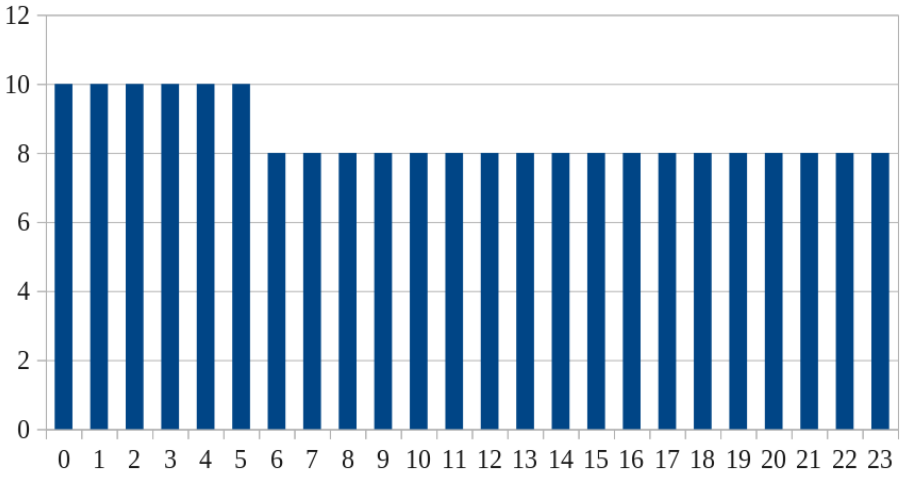


Figure 4.7: The value of α in minutes according to the hour of the day.

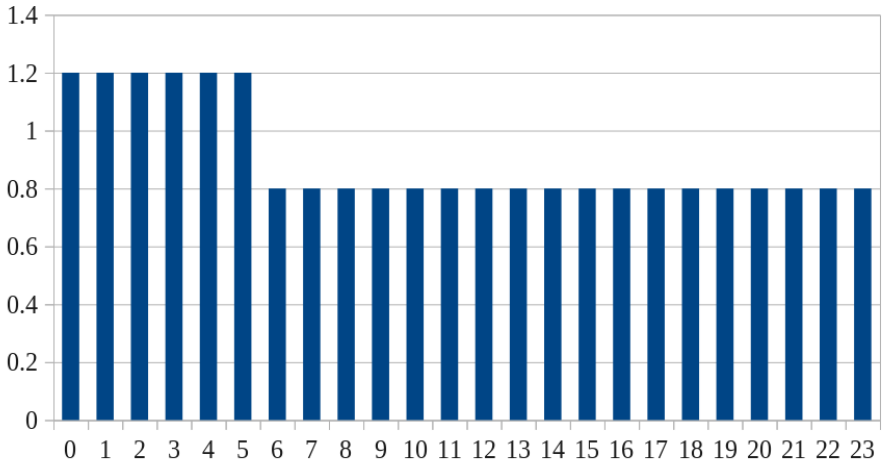


Figure 4.8: The value of β according to the hour of the day.

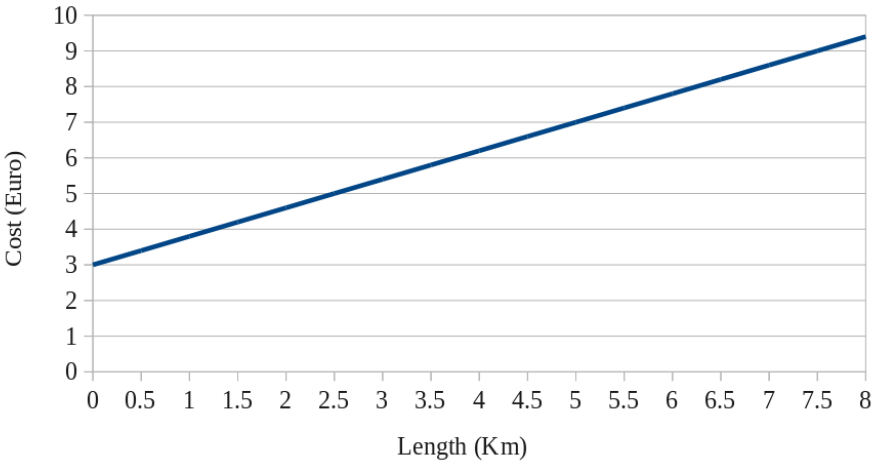


Figure 4.9: The cost of a ride for one user according to the distance between his/her departure and his/her destination.

considering that, since it is a dynamic service, the refused users can forward the request some time later and likely they are accepted.

The average waiting time is 6 minutes and 46 seconds, while the average detour time is 4 minutes and 43 seconds. These results show that the quality of Taxi Sharing is high enough to be considered a good service even though the worst cases can be a bit long, as mentioned in section 4.1.1. In Figure 4.10 and in Figure 4.11 we present respectively how waiting and detour times change according to the hour of the day.

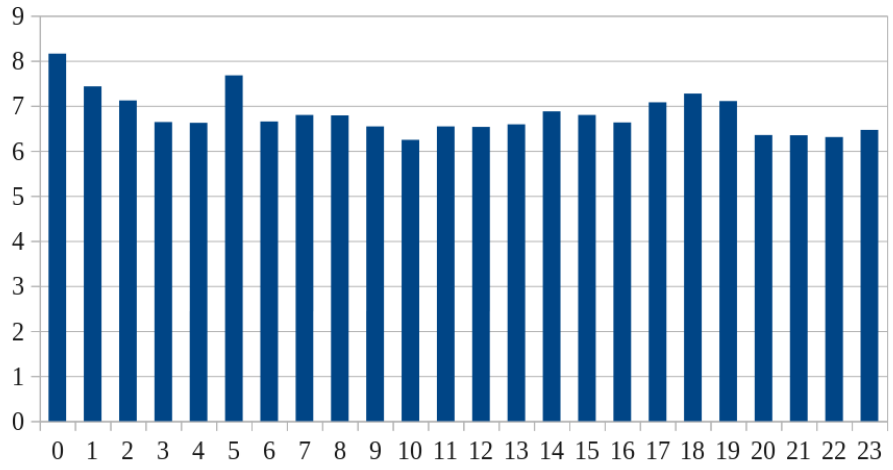


Figure 4.10: The average waiting time in minutes according to the hour of the day.

The waiting time keeps quite constant during the day, while according to the detour time we underline that it is slightly higher in the rush hours due to the higher possibility to share the ride with other users.

As per taxi drivers, the number of taxi rides is 1569 and the average length of a taxi ride is 1 hours and 33 minutes. These data show that, when a taxi starts a Taxi Sharing ride, even if hypothetically it can satisfy just a request, this does not happen since other requests are continuously assigned to the taxi and the ride is prolonged forward for more than one hour in average. The average number of serviced users per hour is 5.64[10], and the average revenue per hour with the taxi fare presented above is 34 *Euros/hour*. The efficiency in servicing requests, how defined in section 3.5, is 94%, more than twice the value of individual taxi.

In Figure 4.12 we present the average number of *active* taxis used to serve

[10]In a normal taxi service, according to conversations with some taxi drivers in Milan, a taxi serves on average two ride, of one or more users, each hour. There are no official report on this datum, since a taxi driver do not have to declare when he/she serves a request.

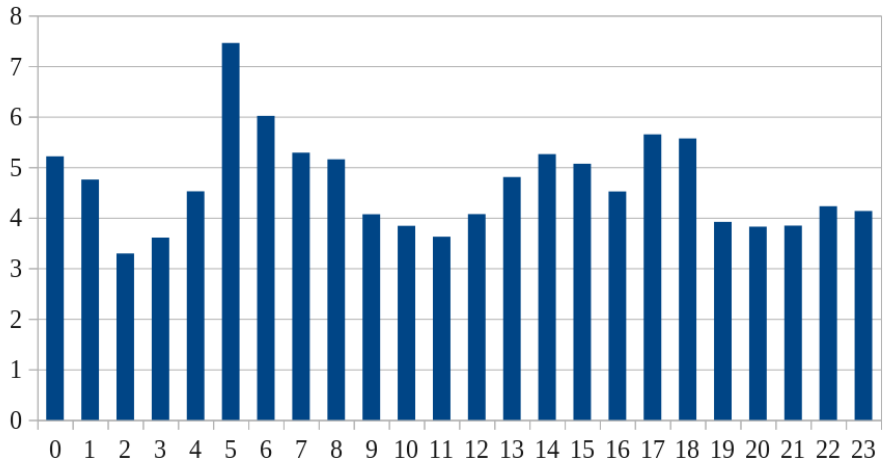


Figure 4.11: The average detour time in minutes according to the hour of the day.

the requests, according to the hour of the day.

4.1.3 Computational details

In this section we focus on the dimensions and the time needed to solve the set partitioning embedded in the *optimizer* process described in section 2.3, since it is NP hard. The time needed to generate the groups and permutation is not critical because it can be done in parallel mode concerning different taxis, furthermore the needed time is negligible as presented in appendix A.

To solve the ILP model we used CPLEX ILOG in parallel mode with 8 threads.

We define:

- $h = \#\{R\} = \#\{O \cup C\}$ as the number of the union of *open* and *confirmed* requests;
- $m = \sum_{i \in T} \#\{G_i\}$ as the total number of groups;
- $n = \sum_{i \in T} \sum_{K_i \in G_i} \#\{Perm(D_i, K_i)\}$ as the total number of permutations associated to m groups;
- $u = \#\{a_{rgi} = 1 | r \in R, i \in T, g \in G_i\}$ as the number of not null coefficient in the set partitioning matrix;
- z as the time in seconds needed to solve the set partitioning.

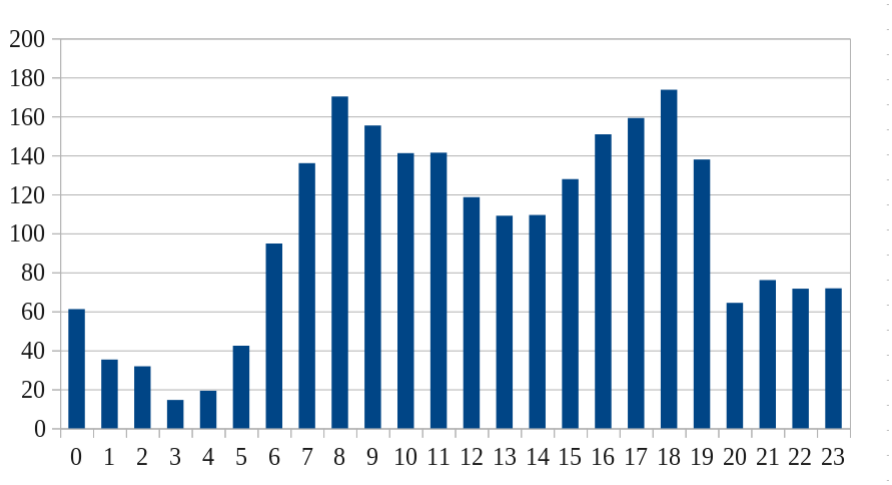


Figure 4.12: Active taxis according to the hour of the day.

How described in Section 3.1, the simulator runs a cycle of processes every $dt = 30$ seconds; in Table 4.1 we report the minimum, maximum and average value, on all cycles of 24 hour simulation, of h, m, n, u and z. About data reported in Table 4.1, we can notice that:

- the size of the problem does not explode according to permutations and groups and it is always solvable in few seconds;
- the matrix of the set partitioning is extremely sparse, having in average less elements different from zero than columns;
- in average $u < m$ since there are also groups that do not cover any *open* or *confirmed* request;
- the average number of requests and groups it is sufficiently high, this is the reason of the quality of service.

In Figure 4.13 we report the average number of *open* and *confirmed* requests (h) according to the hour of the day; we underline that during night hours the number of h is low, nevertheless in these hours the parameters α and β used to compute the time windows are higher as shown in Figure 4.7 and in Figure 4.8.

In Figure 4.14 we report average number of groups (m), permutations (n) and not null coefficients (u) according to the hour of the day; we notice that highest number of permutations is at $hour = 0$ when the level of demand is not the highest, but it is still quite high and the parameters α and β used to compute the time windows are already higher as shown in Figure 4.7 and in

Data	minimum	maximum	average
h	1	85	39.5
m	7	927	297.8
n	8	1684	366.4
u	1	1573	240.1
z	0.02	2.47	0.65

Table 4.1: Minimum, maximum and average of h, m, n, u and z

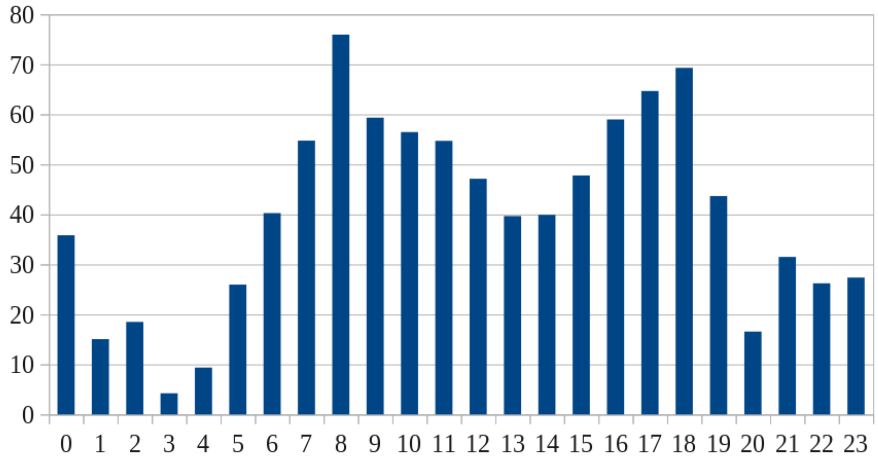


Figure 4.13: Average number of *open* and *confirmed* requests (h) according to the hour of the day.

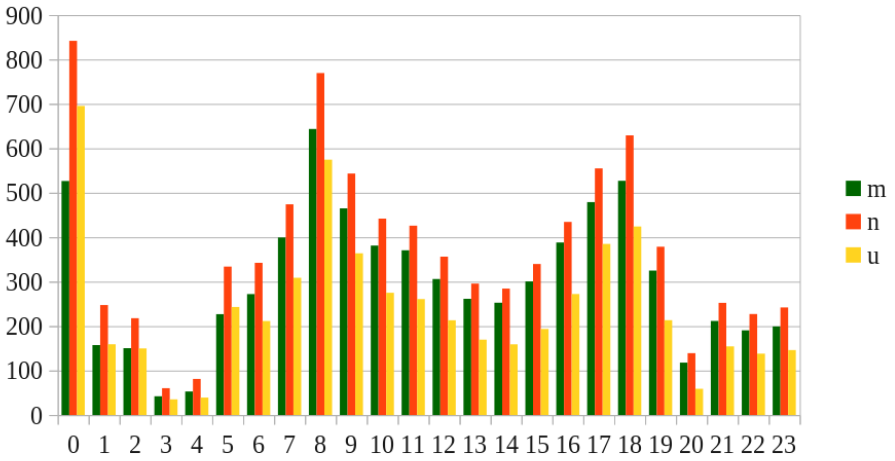


Figure 4.14: Average number of groups (m), permutations (n) and not null coefficients (u) according to the hour of the day.

Figure 4.8. This shows how the number of groups and permutations is strictly dependent on the parameters used to compute the maximum arrival time.

In Figure 4.15 we report the number of groups (m) according to the number of *open* and *confirmed* requests (h); these are not average values, but every dot represents the values of a cycle. We notice how the relation is linear if we omit the cloud of dots in the region $[25; 45] \times [400; 1000]$ that coincide with the values of cycles in $hour = 0$.

In Figure 4.16 we report the time needed to solve the set partitioning (z) according to the number of groups (m); these are not average values, but every dot represents the values of a cycle. We notice that in most optimization cycles, the set partitioning needs less than 1.5 seconds to be solved.

4.2 Medium demand scenario

In this scenario we test the performance of Taxi Sharing supposing a level of demand higher than the previous and similar to the demand of individual taxi. In this condition it is easier to combine different requests together and hence the quality and efficiency of the service could be higher and the fare of the service could be lower. In the next sections we present the additional parameters used for this scenario, the results obtained concerning the performances of the service, and the details on the computational effort needed to optimize the service.

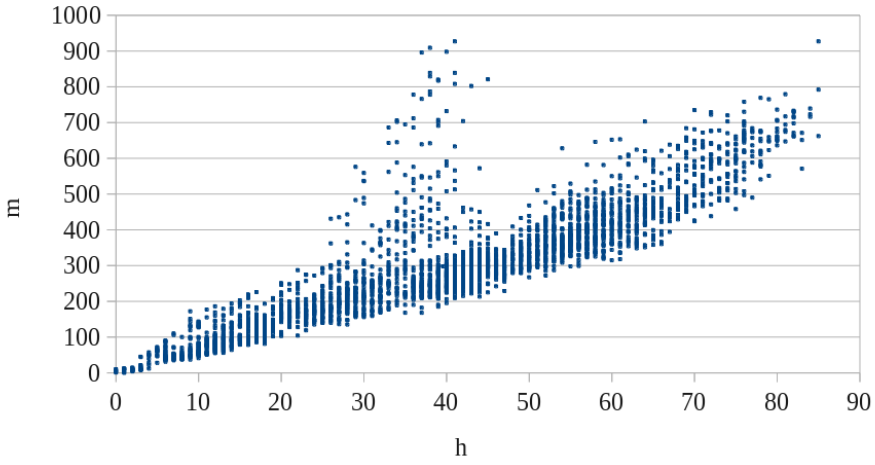


Figure 4.15: Number of groups (m) according to the number of *open* and *confirmed* requests (h).

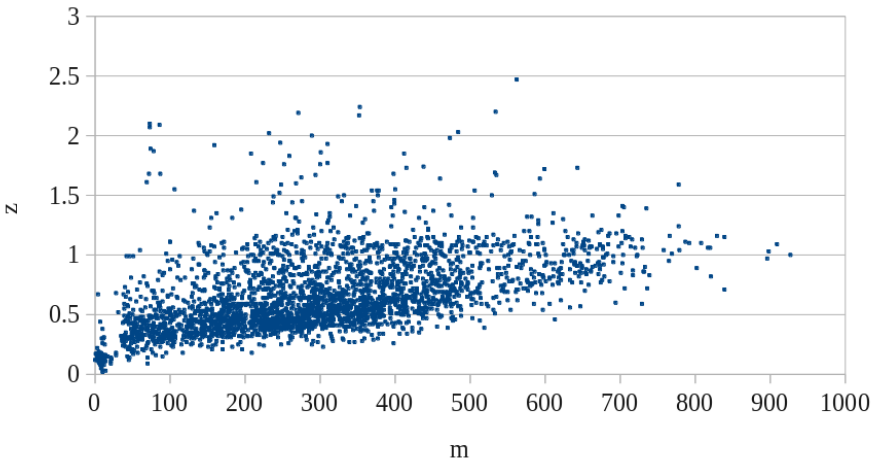


Figure 4.16: Time needed to solve the set partitioning (z) according to the number of groups (m).

4.2.1 Parameters

The use of Taxi Sharing, in respect to the total demand, has been estimated to change during the day according to different levels of traditional public transportation offer, as shown in Figure 4.17.

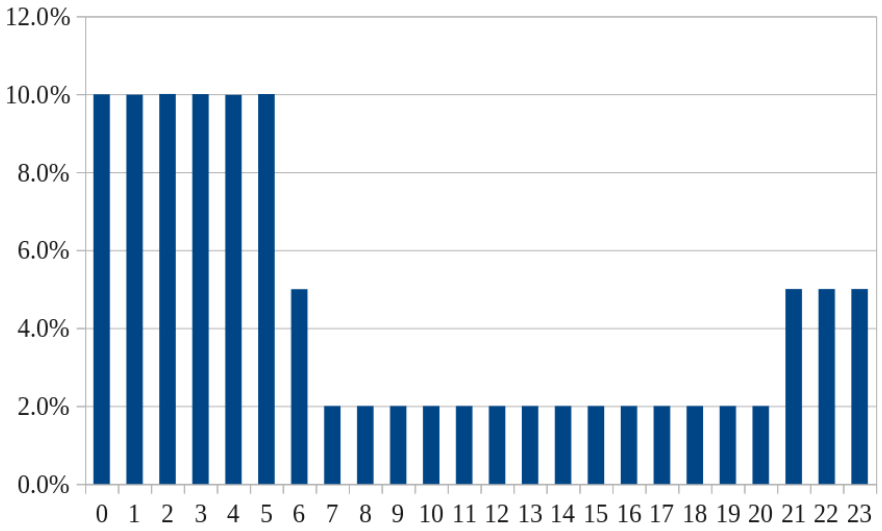


Figure 4.17: Use of Taxi Sharing in respect of the total mobility according to the hour of the day

The total demand of mobility inside the city of Milan, presented in Figure 4.1, has been multiplied by the proportion of Taxi Sharing use to obtain the number of Taxi Sharing users per hour according to the hour of the day as shown in Figure 4.18. The daily number of expected users is 60216 which corresponds to 2.18% of the daily mobility inside the city of Milan.

We suppose that each hour 40% of taxi drivers on duty offers Taxi Sharing service. With this assumption and the data about the number of taxi drivers presented in Figure 4.2, we compute the number of taxi drivers offering Taxi Sharing service according to each hour of the day, as presented in Figure 4.19.

The parameters α and β used to compute the maximum arrival time to destination, as described in section 2.2, are set to have the values represented respectively in Figure 4.20 and in Figure 4.21. Thanks to the higher demand these parameters are lower with respect of the previous scenario, and, hence, the travel time in the worst case is lower especially in day hours. It is possible to use lower value for α and β since the higher demand increase the probability to have requests that can be served by the same taxi with short detour. We present in appendix A a focus on the calibration of β on service quality and on computational complexity.

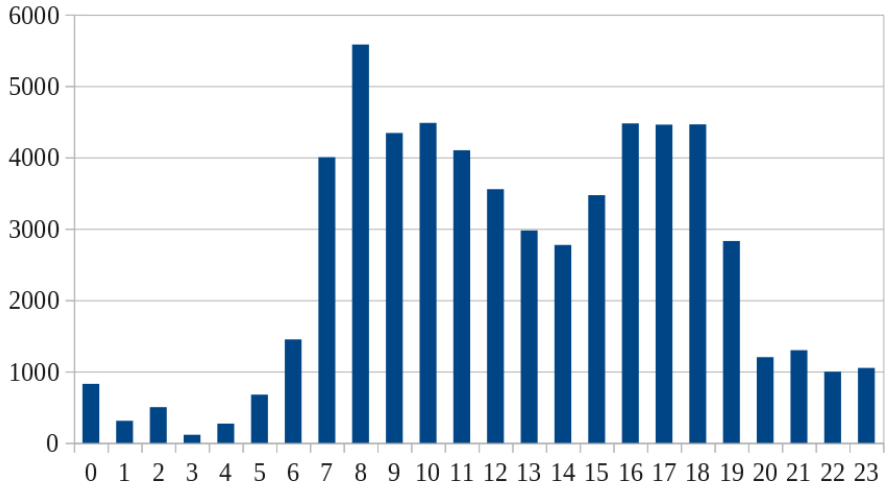


Figure 4.18: Taxi Sharing demand according to the hour of the day.

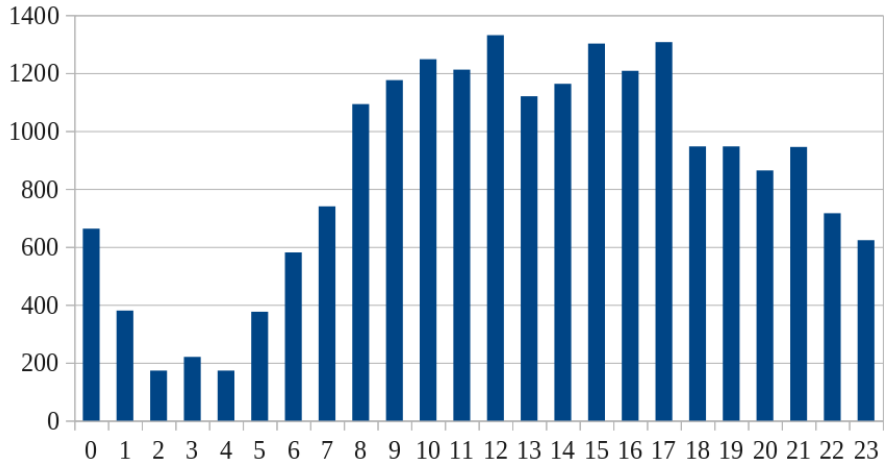


Figure 4.19: Number of taxi drivers offering Taxi Sharing service according to the hour of the day.

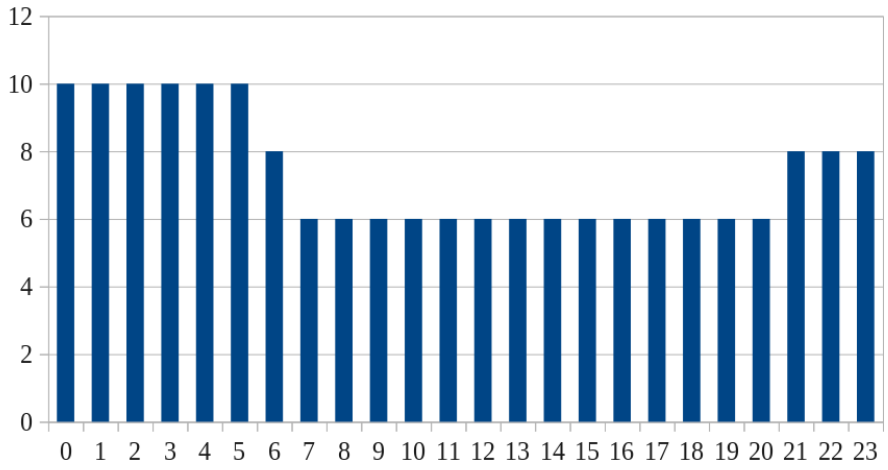


Figure 4.20: The value of α in minutes according to the hour of the day.

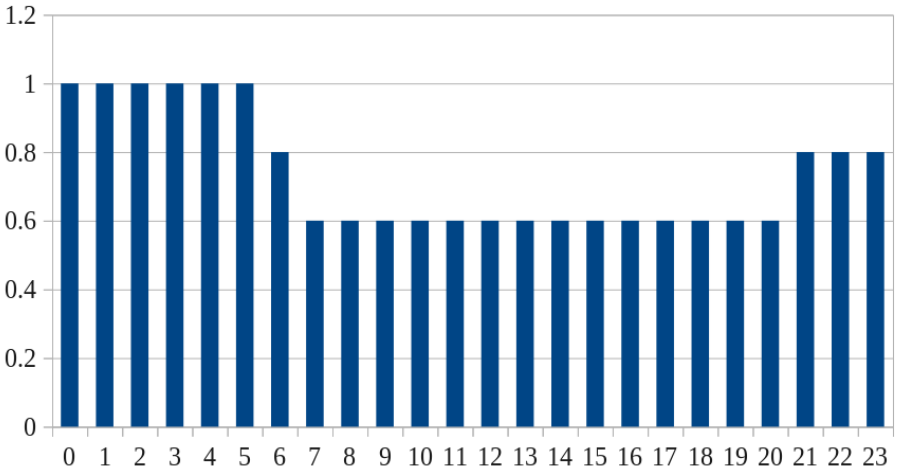


Figure 4.21: The value of β according to the hour of the day.

We set the parameters used to compute the cost of each ride as presented in section 2.2 as follows: $\gamma = 3$ and $\delta = 0.6$. These values let users save 0.20 $Euros/Km$ with respect of the scenario with lower demand. The cost of a ride for one user, according to the length of the ride in Km, is presented in Figure 4.22.

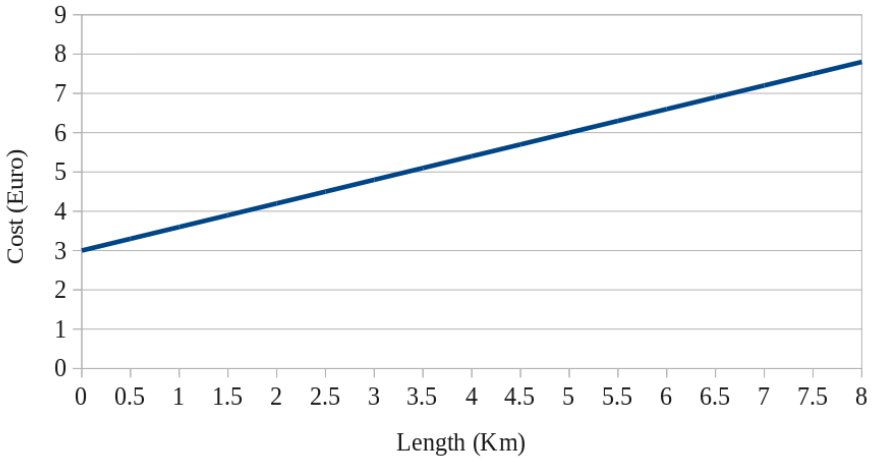


Figure 4.22: The cost of a ride for one user according to the distance between his/her departure and his/her destination.

4.2.2 Service performances

Using the data presented in the previous section, we run a 24 hours simulation of Taxi Sharing within Milan. In the next paragraphs we present the obtained results of the simulation of the service according to users and to taxi drivers.

The number of serviced users is 59508 and the number of refused users is 733. This value as in the previous scenario represents 1% of the total demand; hence it is really low also considering that, since it is a dynamic service, the refused users can forward the request some time later and likely are accepted.

The average waiting time is 5 minutes and 33 seconds, while the average detour time is 4 minutes and 5 seconds. These results, in respect of the results presented in the previous scenario, show how in average the user time is around 2 minutes lower considering both waiting and detour time. In Figure 4.23 and in Figure 4.24 we present respectively how waiting and detour times change according the hour of the day.

The waiting time is quite constant during the day, while according to the detour time we underline that it is slightly higher in the rush hours due to the higher possibility to share the ride with other users.

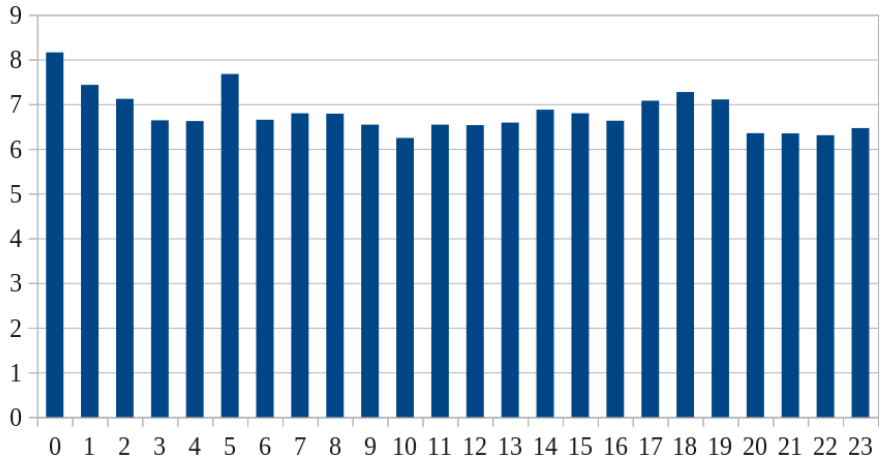


Figure 4.23: The average waiting time in minutes according to the hour of the day.

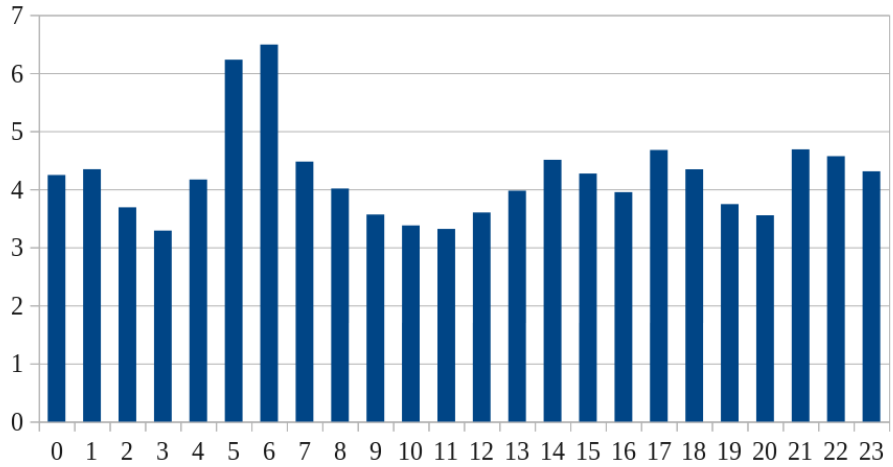


Figure 4.24: The average detour time in minutes according to the hour of the day.

As per taxi drivers, the number of taxi rides is 5127 and the average length of a taxi ride is 1 hours and 48 minutes. These data show that in this scenario the taxi rides are in average 15 minutes longer than in the previous scenario according to the higher level of demand that decrease the possibility to end a ride before the maximum working time. The average number of serviced users per hour is 6.42, this is 0.8 higher than in the previous scenario, and for this the average revenue per hour with the taxi fare presented above is 34 *Euros/hour* as in the previous scenario, even if the fares are lower. The efficiency in satisfying the requests, as defined in section 3.5, is 1.08%, this is a value 157% higher than individual taxi efficiency. In Figure 4.25 we present the average number of *active* taxis used to serve the requests, according to the hour of the day.

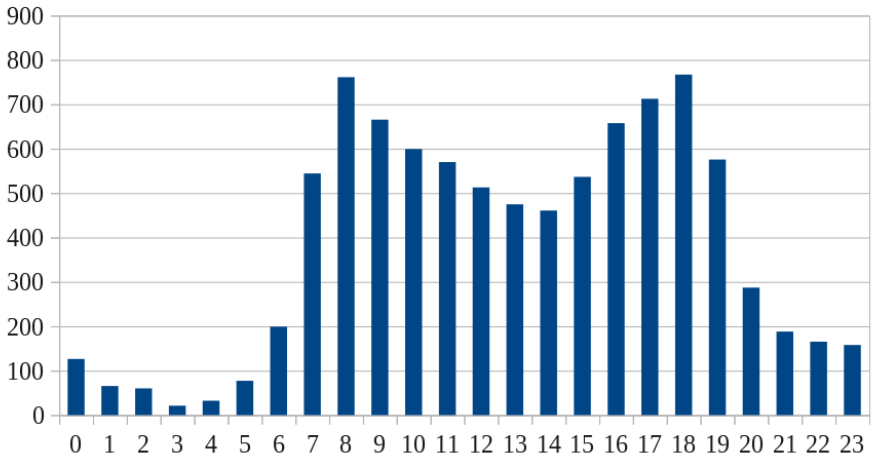


Figure 4.25: Active taxis according to the hour of the day.

4.2.3 Computational details

In this section our focus is on the dimensions and the time needed to solve the set partitioning embedded in the *optimizer* process described in section 2.3. We refer to h, m, n, u and z how they are defined in section 4.1.3.

In Table 4.3 we report the minimum, maximum and average value, on all cycles of 24 hour simulation, of h, m, n, u and z. As per data reported in Table 4.3 we can notice that:

- also with a higher demand the size of the problem does not explode, according to permutations and groups and it is always solvable in few seconds; this is related to stricter parameters α and β. We underline how with a higher demand it is possible to decrease the values of α and β to

guarantee a better service and this is also functional to the solvability of the problem as discussed in appendix A;

- the matrix of the set partitioning is extremely sparse also in this scenario; this is mainly dependent on the fact that every request is imminent.

Data	minimum	maximum	average
h	1	308	130.5
m	11	3755	1263.6
n	14	5084	1488.7
u	3	4106	1082.1
z	0.06	2.66	1.0

Table 4.2: Minimum, maximum and average of h, m, n, u and z

In Figure 4.26 we report the average number of *open* and *confirmed* requests (h) according to the hour of the day; we underline that during night hours the number of h is low; nevertheless in these hours the parameters α and β, used to compute the time windows, are higher as presented in Figure 4.20 and in Figure 4.21.

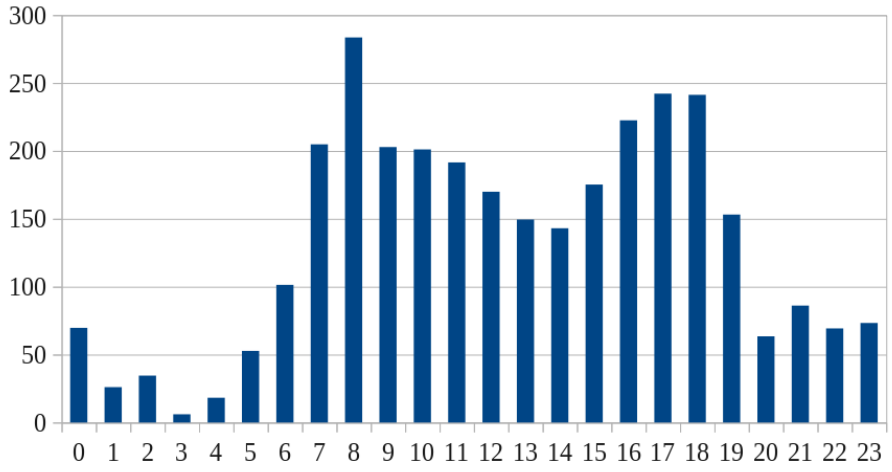


Figure 4.26: Average number of *open* and *confirmed* requests (h) according to the hour of the day.

In Figure 4.27 we report average number of groups (m), permutations (n) and not null coefficients (u) according to the hour of the day; we notice that at $0 \leq hour \leq 5$ we have $u > m$ when the level of demand is not the highest,

but the parameters α and β, used to compute the time windows, are higher, as presented in Figure 4.20 and in Figure 4.21. This shows how the number of groups and not null coefficients is strictly dependent on the parameters used to compute the maximum arrival time.

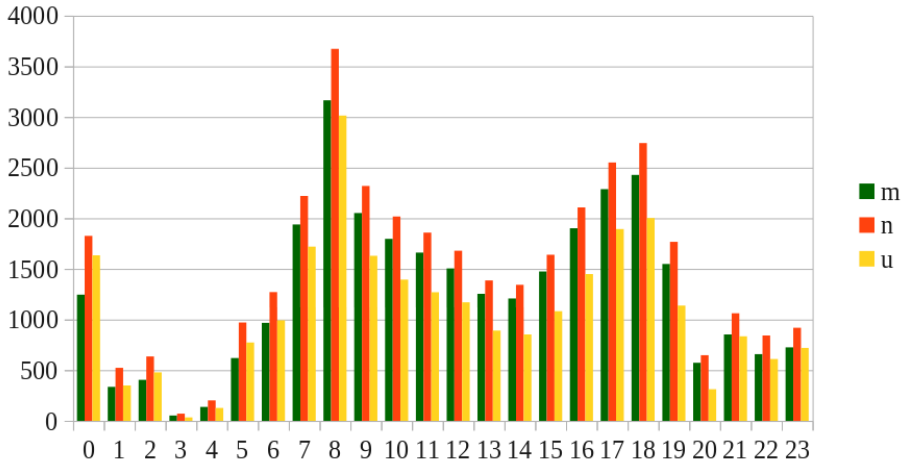


Figure 4.27: Average number of groups (m), permutations (n) and not null coefficients (u) according to the hour of the day.

In Figure 4.28 we report the number of groups (m) according to the number of *open* and *confirmed* requests (h); these are not average values, but every dot represents the values of a cycle. We notice how the relation is linear if we omit the cloud of dots in the region $[50; 80] \times [1000; 2000]$ that coincide with the values of cycles in $hour = 0$.

In Figure 4.29 we report the time needed to solve the set partitioning (z) according to the number of groups (m); these are not average values, but every dot represents the values of a cycle. We notice that with $m \geq 1000$ the relation is strictly linear. This can be due to the fact that the number of groups is above 1000 mainly in the day hours, as shown in figure 4.27, and in these hours the parameters α and β are stricter, as shown in figure 4.20 and 4.21. For this reasons even tough the number of groups is higher due to the higher number of requests and active taxis the set partitioning is highly constrained.

4.3 Car-free scenario

In this scenario we test the performance of Taxi Sharing supposing a revolution in the mobility of the city of Milan to see which are the potentiality of the taxi fleet. We test the performances of Taxi Sharing in a mobility system with a

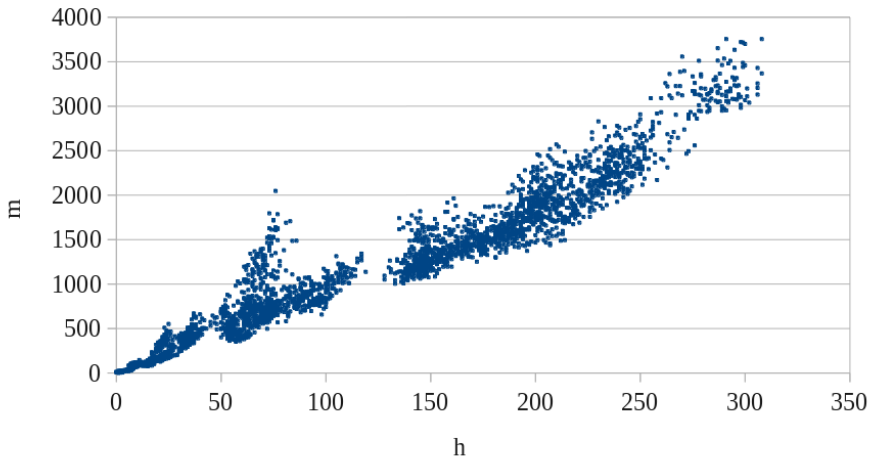


Figure 4.28: Number of groups (m) according to the number of *open* and *confirmed* requests (h).

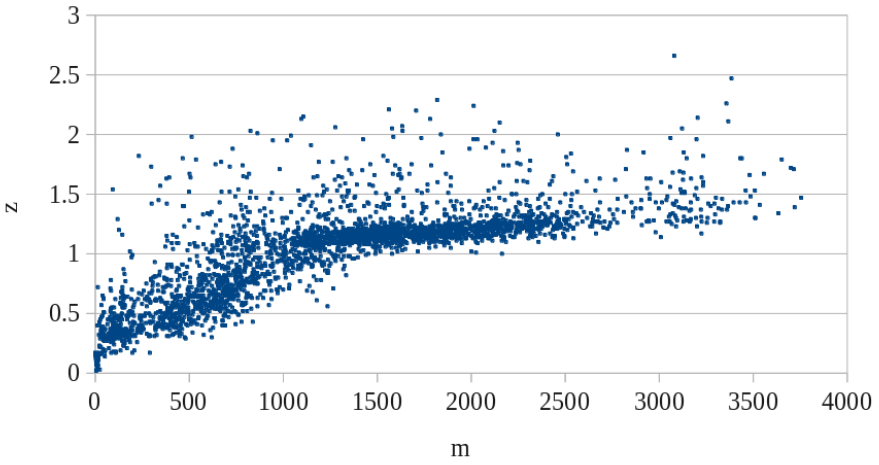


Figure 4.29: Time needed to solve the set partitioning (z) according to the number of groups (m).

low level of use of private motorized vehicles, and a high use other means of transport within the city.

In the next sections we present the additional parameter used for this scenario, the results obtained concerning the performances of the service and the details on the computational effort needed to optimize the service.

4.3.1 Parameters

The use of Taxi Sharing, in respect to the total demand, has been estimated to change during the day according to different levels of traditional public transportation offer, as shown in Figure 4.30.

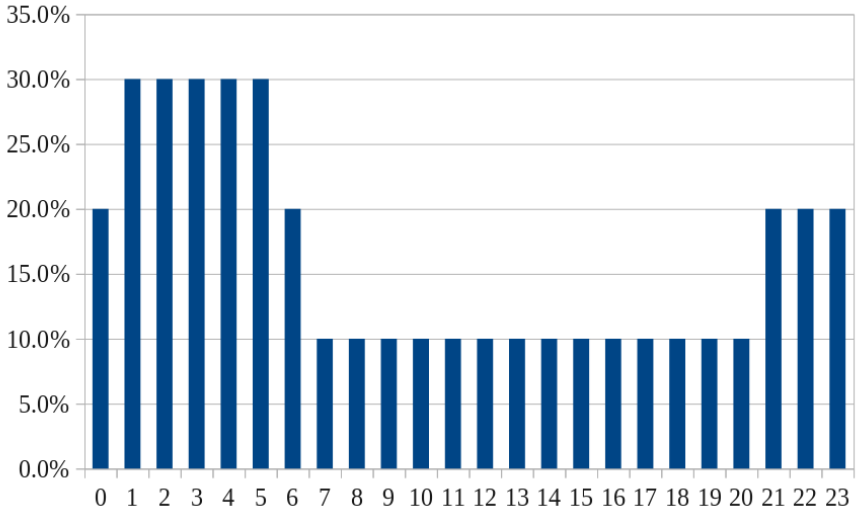


Figure 4.30: Use of Taxi Sharing in respect of the total mobility according to the hour of the day.

The total demand of mobility inside the city of Milan, presented in Figure 4.1, has been multiplied by the proportion of Taxi Sharing use to obtain the number of Taxi Sharing users per hour according to the hour of the day as shown in Figure 4.31. The daily number of expected users is 290061 which corresponds to 10.51% of the daily mobility inside the city of Milan.

We suppose that all the taxi drivers offer Taxi Sharing service. Since the number of taxi drivers presented in Figure 4.2, have not the same pattern of the demand presented in Figure 4.31, we suppose the number of taxi driver to be always 3328 that is the maximum number of taxi driver in working shift at the same time. In the section related to the performance of the service we will present the number of *active* taxis needed to satisfy the demand.

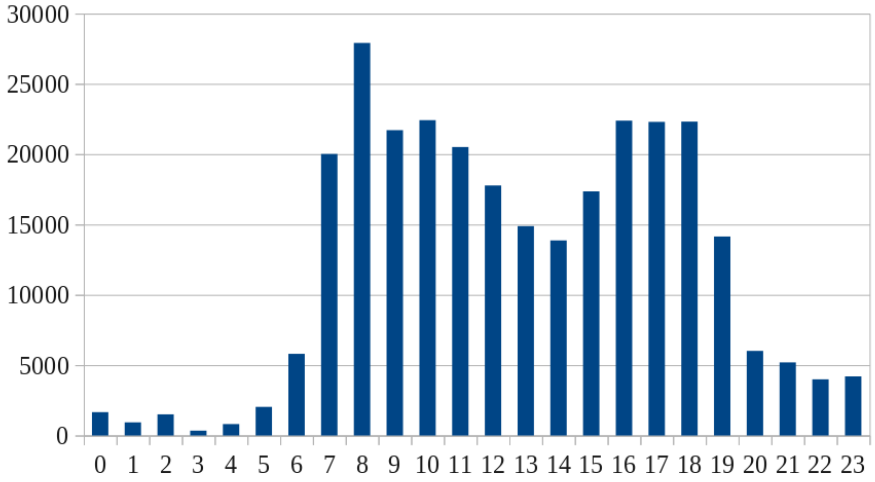


Figure 4.31: Taxi Sharing demand according to the hour of the day.

We suppose the average speed on the road network to be higher than the actual speed, as presented in Figure 4.32, as a consequence of the low level of private motorized mobility.

To compute the maximum arrival time to destination, as described in section 2.2, we set the parameter $\alpha = 5$ in all the hours of the day and the parameter β to have the value represented in Figure 4.33. Thanks to the higher demand these parameters can be extremely low with respect of the previous scenario and in absolute and hence the travel time of the worst case is also good. For example during the day ($\alpha = 5\ min$, $\beta = 0.3$, $vel = 24\frac{Km}{h}$), considering a waiting time of 4 minutes, on a ride's request long 4 Km the detour time to serve other users could be maximum 4 minutes. We present in appendix A a focus on the calibration of β on service quality and on computational complexity.

We set the parameters used to compute the cost of each ride as presented in section 2.2 as follows: $\gamma = 2$ and $\delta = 0.5$. These low values are possible since the higher demand and the higher speed increase the efficiency of the service. The cost of a ride for one user, according to the length of the ride in Km, is presented in Figure 4.34.

4.3.2 Service performances

Using the data presented in the previous section, we run a 24 hours simulation of Taxi Sharing within Milan. In the next paragraphs we present the obtained results of the simulation of the service according to users and to taxi drivers.

The number of serviced users is 285873 and the number of refused users is

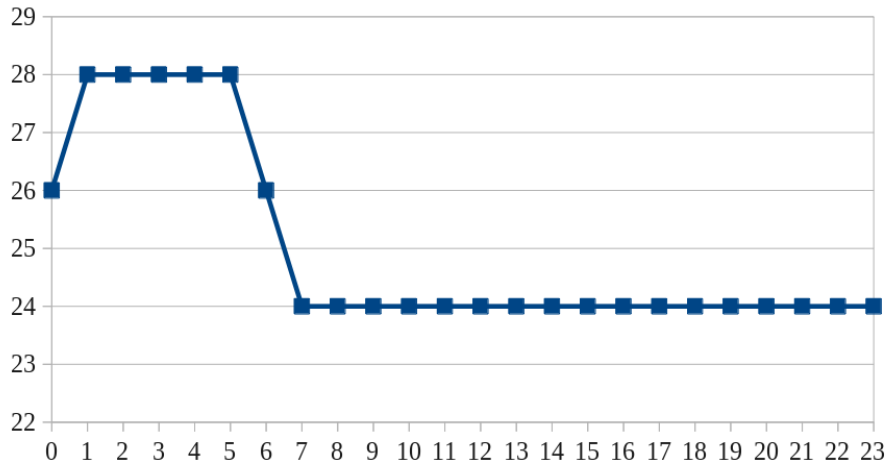


Figure 4.32: Average speed on the road network according the hour of the day, supposing a low level of private motorized vehicles.

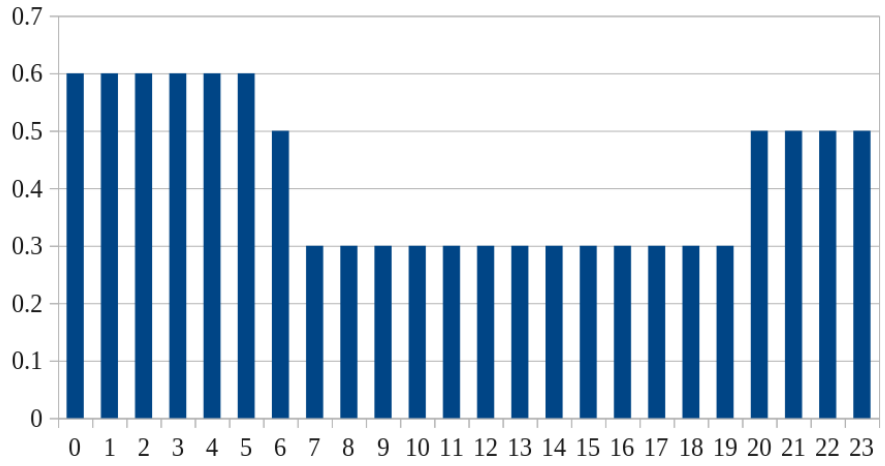


Figure 4.33: The value of β according to the hour of the day.

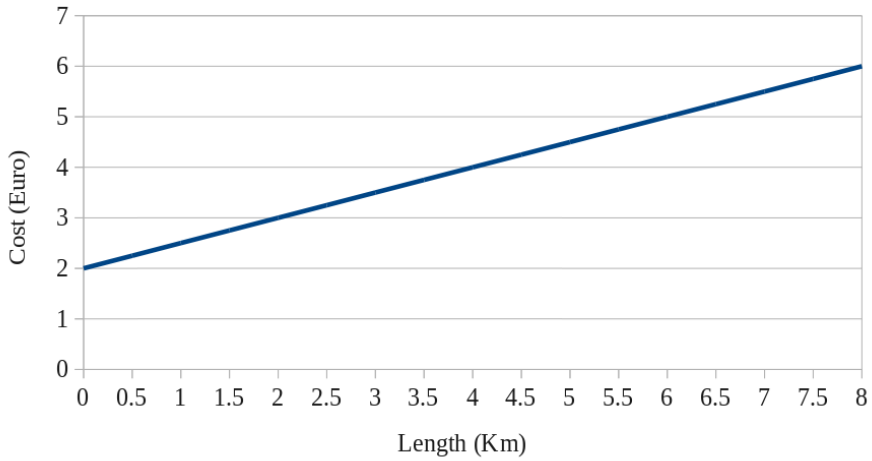


Figure 4.34: The cost of a ride for one user according to the distance between his/her departure and his/her destination.

4012. This value as in the previous scenario represents 1% of the total demand, hence it is really low also considering that, since it is a dynamic service, the refused users can forward the request some time later and likely are accepted.

The average waiting time is 3 minutes and 48 seconds, while the average detour time is 2 minutes and 9 seconds. These results are impressive, and they show how Taxi Sharing in a car-free city can offer lower waiting time and driving time than those of an individual taxi today. In fact the time lost due to sharing the ride (2 : 09) is less that what it is saved thanks to the higher speed on the road network.

In Figure 4.35 and in Figure 4.36 we present respectively how waiting and detour times change according the hour of the day. We underline how the waiting time and the detour time are low also during night hours.

As per taxi drivers, the number of taxi rides is 19641 and the average length of a taxi ride is 1 hours and 38 minutes. The total working time of taxi drivers is 32160 hours; this means that the actual taxi fleet of 4855 taxis could serve the 10.5% of the mobility in the city of Milan if every taxi driver works on average 6 hours and 40 minutes per day.

The average number of serviced users per hour is 8.87, this is 2.5 higher than the previous scenario, and for this the average revenue per hour with the taxi fare presented above is again 34 $Euros/hour$, even if the fares are consistently lower. This result is due both to the higher speed and to the higher demand. The efficiency in satisfying the requests, as defined in section 3.5, is 1.15%, this is a value 173% higher than individual taxi. This datum reflects the increase

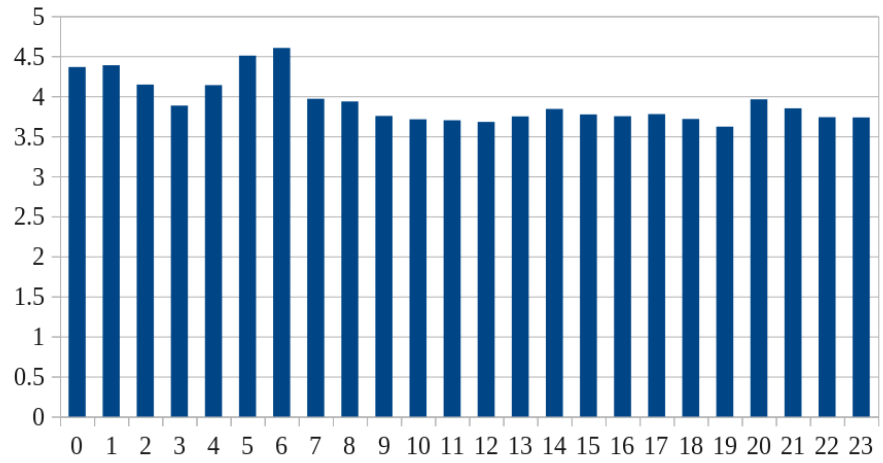


Figure 4.35: The average waiting time in minutes according to the hour of the day.

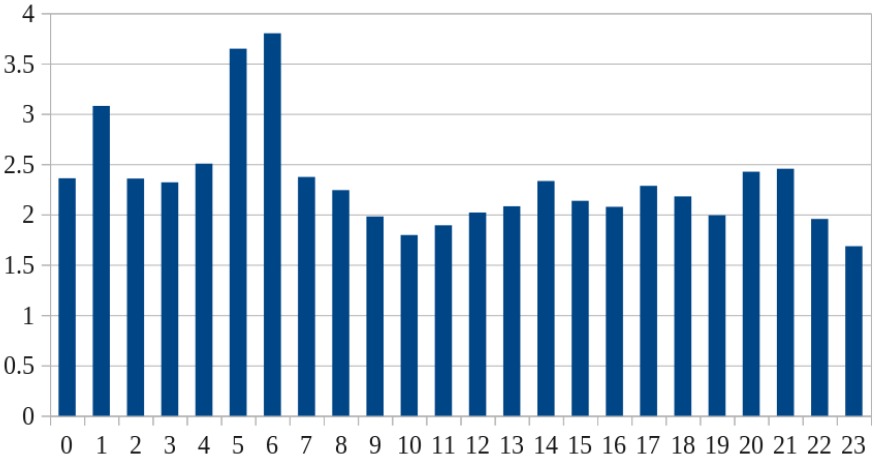


Figure 4.36: The average detour time in minutes according to the hour of the day.

of served user request for each traveled Km, and hence it does not consider the increase in efficiency due to the higher speed. Computing the index regarding increase of served user request for each *worked hour* we obtain for individual taxi in the actual scenario $0.42 * 20Km/h = 8.4$ *served km/hour*, while for the Taxi Sharing in the car free city scenario we obtain $1.15 * 24Km/h = 27.6$ *served km/hour*[11]. This shows that the efficiency that taxis fleet can obtain providing Taxi Sharing service in a car free city context it is more than three time the actual efficiency.

This shows that Taxi Sharing could represent an important service in and for an urban mobility with a low rate of motorized individual mobility. Indeed Taxi Sharing works at the best in a city without congestion, and can represents a tool to reach this aim since it can improve and differentiate the offer of mobility service providing a pool of choices to citizens.

In Figure 4.37 we present the average number of *active* taxis needed to provide Taxi Sharing compared with the number of available taxis in the city of Milan according to the hour of the day.

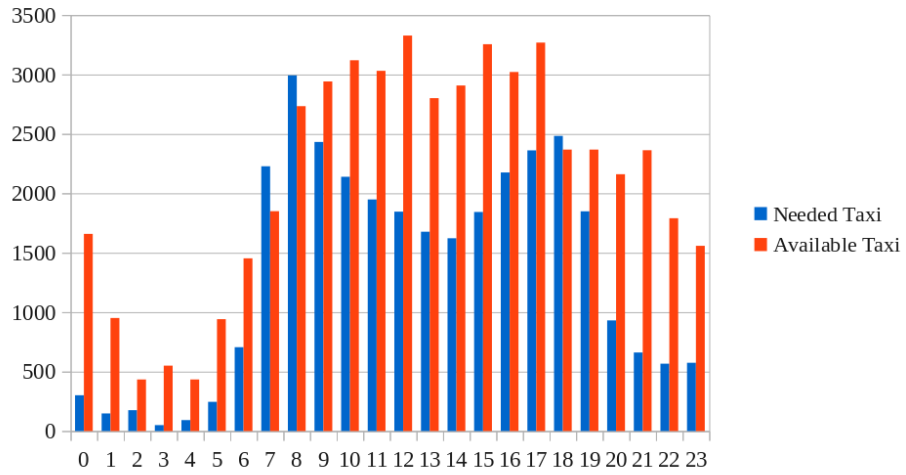


Figure 4.37: Active taxis according to the hour of the day compared with working taxi

4.3.3 Computational details

In this section our focus is on the dimensions and the time needed to solve the set partitioning embedded in the *optimizer* process described in section 2.3. We

[11] This is consistent with the number of served users/hour that is 8.87 considering that each request involve in average 1.2 users, and that the requests length is in average around 4 Km.

refer to h, m, n, u and z how them are defined in section 4.1.3.

In Table 4.3 we report the minimum, maximum and average value, on all cycles of 24 hour simulation, of h, m, n, u and z. As per data reported in Table 4.3 we can notice that:

- also with a higher demand the size of the problem does not explode according to permutations and groups and it is always solvable in few seconds, this is related to stricter parameters α and β. We underline how with a higher demand it is possible to decrease the values of α and β to guarantee a better service and this is also functional to the solvability of the problem;
- the matrix of the set partitioning is extremely sparse also in this scenario, it is mainly dependent on the fact that every request is imminent.

Data	minimum	maximum	average
h	6	908	351.8
m	40	15609	4764.9
n	49	17268	5246.5
u	8	15664	4075.1
z	0.11	6.01	1.7

Table 4.3: Minimum, maximum and average of h, m, n, u and z

In Figure 4.38 we report the average number of *open* and *confirmed* requests (h) according to the hour of the day; we underline that during night hours the number of h is low, nevertheless in these hour the parameters β used to compute the time windows are higher as presented in Figure 4.33.

In Figure 4.39 we report average number of groups (m), permutations (n) and not null coefficients (u) according to the hour of the day; we notice that $u < m$ also at $0 \leq hour \leq 5$ differently from what reported in the previous scenario. This is due to the fact, that in this scenario, also in the night hours the parameters α and β are lower and hence less groups can accommodate more that one *open* request at the same time.

In Figure 4.40 we report the number of groups (m) according to the number of *open* and *confirmed* requests (h); these are not average values, but every dot represents the values of a cycle. We evidence that also in the peak hour the number of groups and permutations does not explode.

In Figure 4.41 we report the time needed to solve the set partitioning (z) according to the number of groups (m); these are not average values, but every dot represents the values of a cycle. We notice that with $1500 \leq m \leq 8000$ the relation seems to be linear.

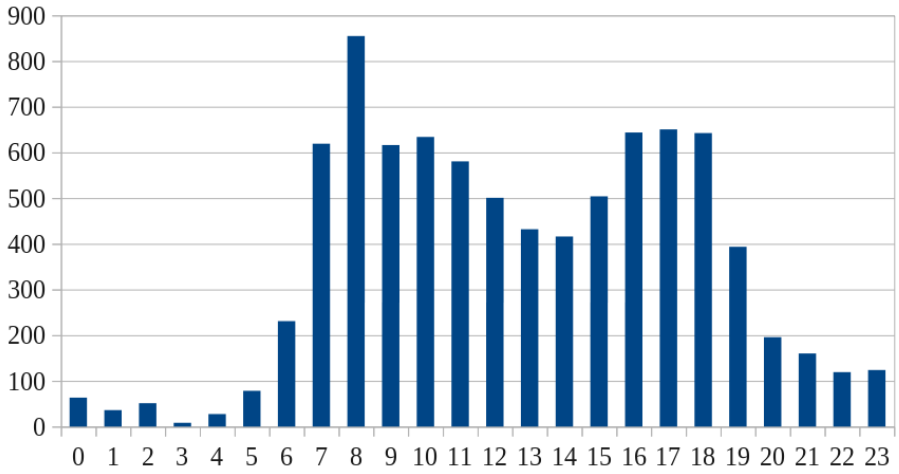


Figure 4.38: Average number of *open* and *confirmed* requests (h) according to the hour of the day.

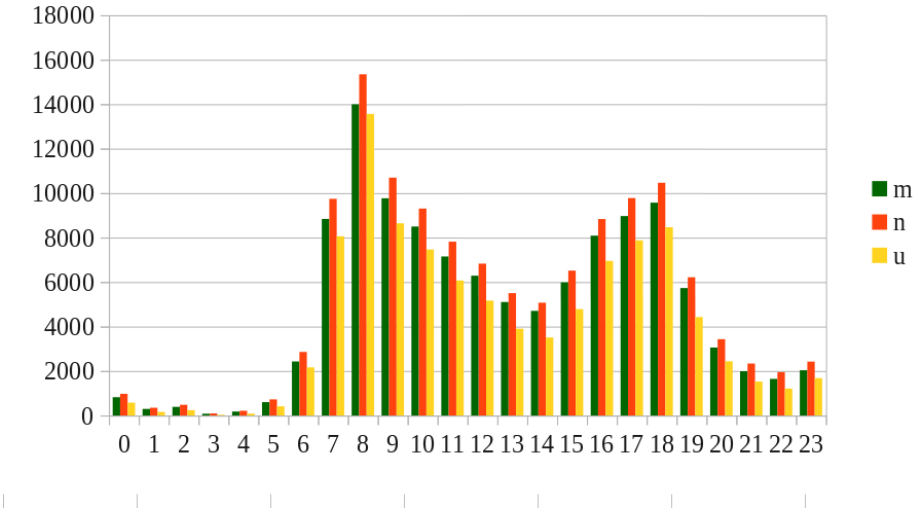


Figure 4.39: Average number of groups (m), permutations (n) and not null coefficients (u) according to the hour of the day.

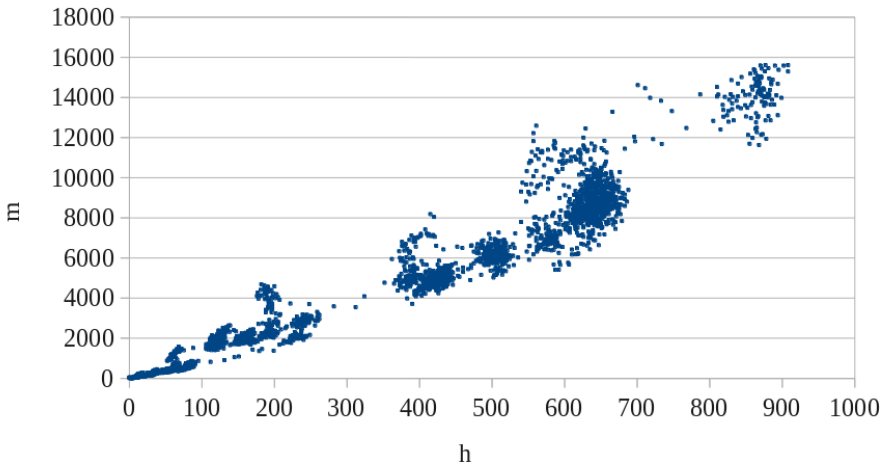


Figure 4.40: Number of groups (m) according to the number of *open* and *confirmed* requests (h).

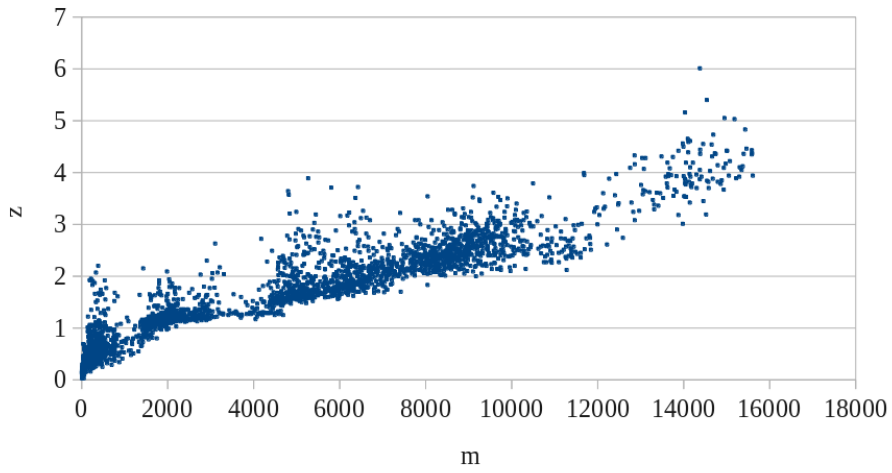


Figure 4.41: Time needed to solve the set partitioning (z) according to the number of groups (m).

In all the three analyzed scenarios the computational efforts needed to optimize the service are negligible, and hence this aspect is not critical for the development of Taxi Sharing. Concerning the quality of the service the results show that the Taxi Sharing's level of service is acceptable in the scenario with low demand and that it increases in the scenarios with higher demand, because of the higher possibility to share the ride with negligible detours. In the car free city scenario Taxi Sharing can serve up to 10% of the total mobility demand with the current taxis fleet, guaranteeing a shorter traveling time than individual taxis in the actual scenario.

Taxi Sharing induces a win-win-win situation since the advantages for users, taxi drivers and municipality derive from the higher efficiency of the Taxi Sharing service, which can rise up to more than three times the efficiency of individual taxis. According to the presented results, for the developing of Taxi Sharing service, we judge the mobility system in which it is planned concerning both the demand for Taxi Sharing and the congestion level of the road network to be determinant.

In turn a more efficient LPT network can reduce the number of owned and used cars and as a consequences it can reduce the congestion and increase the demand for mobility service, both aspects important for Taxi Sharing service. To summarize, Taxi Sharing is a service that:

- works in the most efficient way in a car-free scenario;
- can help to reach this result, not only because of the provided service itself, but also thanks to its integration with LPT.

In the next chapter we assess new possibilities in planning rapid Local Public Transportation related to the realization of Taxi Sharing service.

Chapter 5

Rapid LPT planning in presence of a taxi sharing service

In this chapter we present new possibilities in planning rapid Local Public Transportation (LPT) related to the realization of Taxi Sharing. The integration of LPT with on demand services can lead to a bimodal service [30]. The literature on Network Design Problem is wide [31, 32, 33, 34] and range from routes to stops planning. We limit in this dissertation to the specific problem of optimal stop spacing for a single line, and we access it with a case study on a tram line.

In planning a local public transportation service, a crucial aspect is to establish optimal stop spacing, since it affects passengers walking time and the operating speed of a route, which in turn affects both transit time and operating costs. Densely spaced public transport stations obviously decrease the walking distance, but also increase in-vehicle time and supply costs [22]. On the other hand, eliminating service stops, entails a longer walking distance, but speeds up the system and reduces the operating costs. It is, therefore, necessary to evaluate an optimal trade off between conflicting advantages, in which the optimal solution is influenced by both walking speed and the cost of walking time with respect to in-vehicle time.

The actual distance between stops in the city of Milan is quite short, since LPT needs to serve every type of user, and hence also people with movement impairments. Taxi Sharing as a door-to-door transportation means represents a comfortable solution to people with movements impairments;[1] therefore, in presence of Taxi Sharing the stop spacing can be reoptimized considering as users of LPT only people without movements impairments and hence establishing a

[1] Considering the highest price of Taxi Sharing in respects of LTP, it is possible to provide economical benefits to people with movement impairments in case of Taxi Sharing service use. The resources could derive from the higher efficinecy of LPT.

longer distance between stops[2].

In this chapter we present an analysis of the effects in terms of total variation in user traveling time and increase of commercial speed achievable with the suppression of some stops. The case study is tram line 9, which spans from Stazione Centrale to Stazione di Porta Genova. The analysis has been carried out with the support of AMAT, in relation to the project "Linee T" that aims to enslave traffic lights.

This chapter is composed of three parts: in section 5.1 we detail the used data, in section 5.2 we analyze the effects of the suppression of each stop on the total traveling time for users and in section 5.3 we present the increase of commercial speed in relation to the suppressed stops.

5.1 Data

To perform the analysis presented in this section we used data collected by AMAT, in a monitoring activity of LPT service. The data were collected in a survey campaign that took place between 12/03/2014 and 19/11/2014, and monitored 51 races for tram line 9. The collected data concern:

- the number of people boarded and alighted at every stop;
- the number of people on board departing from each stop.

With additional on board monitoring which took place between 28/10/2014 and 07/11/2014, the geographic coordinates of the tram were being detected every second with a gps device. With the monitoring of 12 races, we determined the following average data:

- average waiting time for each red: $t_r = 16$ seconds
- average time required to make each stop: $t_s = 18$ seconds
- average time lost due to the deceleration and acceleration: $t_{da} = 14$ seconds
- average commercial speed excluding the phase of deceleration, stop and acceleration due to traffic lights and stops: $v_f = 7m/s$

The time that the tram requires to perform a stop depends on the number of users boarded and alighted. The time necessary to perform a stop was estimated with a linear regression on the data collected in the monitoring campaign as:

$$T = 11 + 0.36 * N \ seconds \tag{5.1.1}$$

where N is the number of users boarded and alighted. This function indicates that the average base time required for each station is: $t_b = 11$ seconds with an additional time for every user that boards and alights of 0.36 seconds.

[2]The survay presented in chapter 6 shows a high availability to walk longer in exchange of a faster transportation network.

Assuming that the total number of users that use the tram does not increase or decrease with the suppression of some stops, the contribution of the second term of the sum remains unchanged[3]. The time that would be saved for each suppressed stop is therefore attributable to the average base time needed for each station $t_b = 11$ plus the time lost in the phase of deceleration and acceleration $t_{da} = 14$, so it totals 25 seconds.

5.2 Users traveling time

We define the total users journey time as the sum on all users of the journey time given by the walking time needed to reach the closest station, the in-vehicle time needed to reach the destination stop and the on foot time needed to reach the destination.

We evaluate the utility $u(j)$, of a stop j, as the effect on the total users journey time of the suppression of stop j, by comparing:

1. the time $l(j)$ that would be lost by those who currently use stop j, because of the greater distance required to reach the earlier or later stop;
2. the time $s(j)$ that would be saved by those on board due to stop j suppression.

For each stop j on line 9 we determine the utility of stop j with the following formula:

$$u(j) = l(j) - s(j) \tag{5.2.1}$$

and we note that utility of stop j is positive, $u(j) \geq 0$, if $l(j) \geq s(j)$ and, hence, as a consequence of the suppression of stop j, the time lost by users that were using stop j is higher of the time saved by users that are on board during stop j. Conversely the utility of stop j is negative, $u(j) \leq 0$, if $l(j) \leq s(j)$ and hence as a consequence of the suppression of stop j, the time lost by users that were served by stop j is smaller than the time saved by users that are on board during stop j.

We considered as catchment area of a line the set of all buildings located less than 500 meters on the road graph from a bus stop of that line. Assuming the suppression of stop j, we compute the new distance from the nearest stop for every building in the catchment area of stop j. Assuming that the demand generated and attracted to stop j is distributed uniformly between the building in the catchment area of stop j, we compute the average increase of the walking distance if stop j is suppressed.

To compute the average increase in walking time from the average increase of the walking distance we used a walking speed of 80 meters/minute. The

[3] Even though we access a case study on tram line 9, it is has to be consider in a optimization of the entire LPT network. The users' choice of which line to use is correlated to the LPT network, for this reasons we do not consider the variation in the number of users, since it is not accessible considering only a line.

value obtained for each stop j reflects the distance from the nearest adjacent stops and the topography of the road network in the catchment area of the stop. The value of $l(j)$ is then determined by multiplying the average time to walk to the former or following stop, for users using stop j, by the average number of boarded and alighted users at stop j.

To determine the saved time $s(j)$ of on board users (if stop j is suppressed) we multiply the time lost by the tram to make the stop by the average number of users on board the tram during stop j (that would have an advantage in traveling time from the suppression of stop j).

In Figure 5.1 we report the utility $u(j)$ of each stop j of line 9 from Central Station to Porta Genova expressed in minutes. We define the set of stops $C = \{j|u(j) \leq 0\}$ as the set of candidate stops to be suppressed due to having negative overall utility. We have $|C| = 13$ that represents 48% of the stops in B.

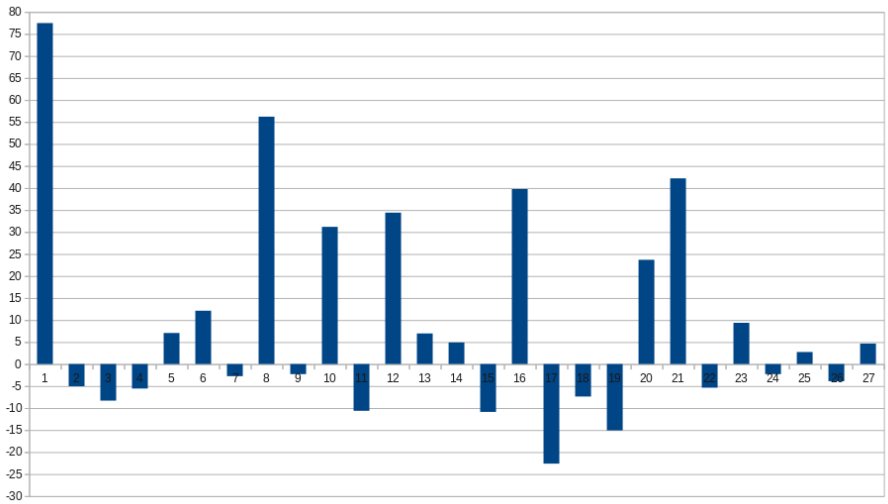


Figure 5.1: The utility $u(j)$ of each stop j of line 9 from Central Station to Porta Genova expressed in minutes.

We introduce an ILP model used to maximize the total users utility, defined as the sum on stops j of $u(j)$.

Sets:

- B: the the ordered set of all the stops

Parameters:

- u_j the utility of stop $j \in B$

Variables:

- $x_j = \begin{cases} 1 & \text{if stop } j \in B \text{ is mantained} \\ 0 & \text{if stop } j \in B \text{ is suppressed} \end{cases}$

Objective function:

$$\max \sum_{j \in B} u_j x_j$$

Constraint:

- $x_j + x_{j+1} \geq 1 \; \forall j \in B$

The constraint excludes the possibility that two subsequent stops are both suppressed, since the calculation of the increased walking distance, $l(j)$, is based on the assumption that there are contiguous stops. In this case the instance is trivial. In the optimal solution $\sum_{j \in B}(1 - x_j) = 11$, which represent the number of stops to be suppressed is 40% of the stops. According to the distances between consecutive stops, in the actual scenario the average distance is 292 meters and the maximum distance is 456 meters, while in optimal stops suppression scenario the average distance is 506 meters and maximum distance is 775 meters.

In Figure 5.2 we present the map of the stops in relation to optimal solution. The color represents the utility, green for stops j with $u(j) > 0$ and red for stops j with $u(j) < 0$. Suppressed stops are marked with X, stops with a green annulus are stops with negative utility that are not suppressed due to the constraint on subsequent stops.

In the optimal solution the overall utility is $\sum_{j \in B} x_j u_j = 337$ minutes while without any suppression the overall utility is $\sum_{j \in B} u_j = 250$ minutes and hence with the optimal stops suppression the users save in total 87 minutes considering both walking and in vehicle time. In the next section we assess the impact of optimal stop suppression on commercial speed.

5.3 Commercial speed

In the previous section we have considered the effects of the suppression of some stop on the user traveling time. In this section we estimate the effects caused on the commercial speed of the suppression of some stops with and without the enslavement of traffic lights. We consider the average time expected for each intersection with traffic light, the time required to perform the stops and the time lost due to the deceleration and acceleration up to commercial speed.

We estimate the commercial speed as a function of the number of red traffic lights n_r and the number of stops n_s with the following function:

$$v(n_r, n_s) = \frac{\Delta S}{\frac{\Delta S}{v_f} + (t_r + t_{da})n_r + (t_s + t_{da})n_s} \tag{5.3.1}$$

where:

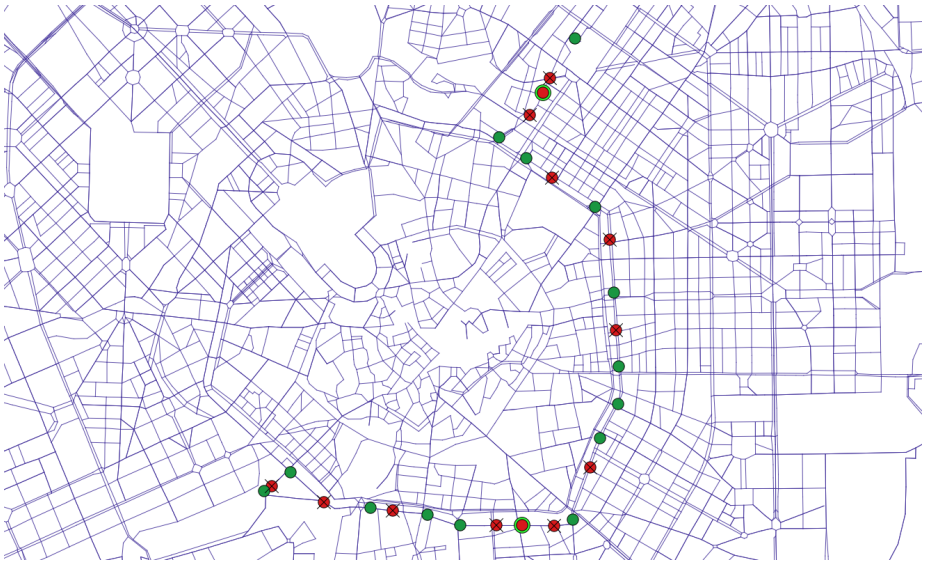


Figure 5.2: Optimal stops suppression on line 9.

- ΔS is the length of the line;
- v_f, t_r, t_s, t_{da} are parameters defined in section 5.1

According to the optimal solution found to minimize the total users traveling time for tram line 9, presented in section 5.2, we analyze the commercial speed computed with the previous formula fixing $n_s = 27$ to represent the actual scenario and $n_s = 16$ to present the optimal scenario based on to the suppression of 11 stops. In Figure 5.3 we present the graph of $a(x) = v((1-x) * 26, 27)$ that represents the commercial speed as function of the percentage of green traffic lights with all stops, and $b(x) = v((1-x) * 26, 16)$ that represents the commercial speed as function of the percentage of green traffic lights with the optimal number of stops.

We report in Table 5.1 the obtained values of absolute and in percentage increase of commercial speed with the suppression of the optimal number of stops in the scenario without and with the enslavement of traffic lights. We note that the increase of commercial speed is higher both in absolute and in percentage in presence of traffic lights enslavement. This occurs because, with the priority at traffic lights, the incidence of the time used for the stops in percentage is more relevant on the overall journey time and therefore on the commercial speed.

In addition to traffic lights priority and optimal stop spacing can be implemented other side measures as:

- physical protection of paths;

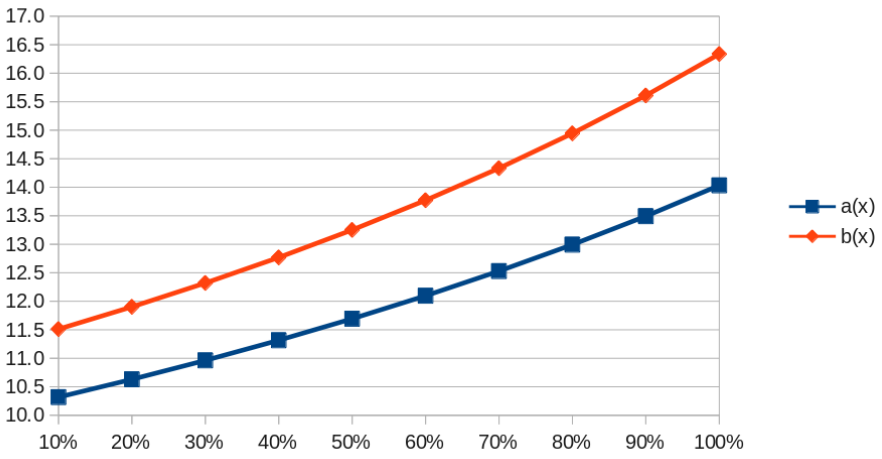


Figure 5.3: The commercial speed as function of the percentage of green traffic lights in the actual scenario, $a(x)$, and in the optimal scenario with stops suppression, $b(x)$

Scenario	x	$a(x)$	$b(x)$	$b(x) - a(x)$	$\frac{b(x)-a(x)}{a(x)}$
Without priority	30%	11.0	12.3	1.3	12%
With priority	90%	13.5	15.6	2.1	16%

Table 5.1: Absolute and in percentage increment of commercial speed with the suppression of the optimal number of stops in the scenario without and with enslavement of traffic lights.

- mitigation of interferences;
- reorganization of the intersections.

If we suppose that the overall results of these actions allow the increase of the average cruise speed (excluding the phase of deceleration, stop and acceleration due to traffic lights and stops) by 10%, the cruise speed would increase from $v_f = 7m/s$ to $v_f = 7.7m/s$.

With this increase of commercial speed of vehicles, the effect of the stops would become more relevant with respect to the total journey time and therefore the effects on the commercial speed of the optimal stops suppression should increase.

We report in Table 5.2 the obtained values of absolute and percentage increment of commercial speed with the suppression of the optimal number of stops in the scenario without and with enslavement of traffic lights supposing $v_f = 7.7m/s$.

Scenario	x	$a(x)$	$b(x)$	$b(x) - a(x)$	$\frac{b(x)-a(x)}{a(x)}$
Without priority	30%	11.4	12.9	1.5	13%
With priority	90%	14.2	16.5	2.4	17%

Table 5.2: Absolute and in percentage increment of commercial speed with the suppression of the optimal number of stops in the scenario without and with enslavement of traffic lights supposing the implementation of side measures of protection.

We compare in Table 5.3 commercial speed without (v_0) and with (v_1) the implementation of side measures (with $v_f = 7m/s$ and $v_f = 7.7m/s$ respectively) in the following cases:

- in the actual scenario;
- in the scenario with the suppression of the optimal number of stops;
- in the scenario with the enslavement of traffic lights;
- in the scenario with the suppression of the optimal number of stops and the enslavement of traffic lights.

We note that the increase of commercial speed obtained with the implementation of measures of protection is more relevant in absolute and in percentage if the measure of optimal stop suppression and priority traffic lights are also implemented. We note that the commercial speed with protection measures, optimal stops suppression and priority traffic lights could increase from 11 Km/h to 16.5 Km/h and hence by 50%.

Optimal stops suppression	Priority traffic lights	v_0	v_1	$v_1 - v_0$	$\frac{v_1 - v_0}{v_0}$
no	no	11.0	11.4	0.4	3.6%
yes	no	12.3	13.9	0.6	4.9%
no	yes	13.5	14.2	0.7	5.2%
yes	yes	15.6	16.5	0.9	5.8%

Table 5.3: Absolute and in percentage increment of commercial speed with measures of protection in different scenarios based on optimal stops suppression and priority traffic lights.

Chapter 6

Questionnaire on rapid LPT and Taxi Sharing

In this section we present the results of a survey related to rapid LPT and Taxi Sharing in which we gathered interesting results from more than 700 answerers. A short questionnaire was spread mainly through social networks, collecting more than 500 answers in 4 days showing a great interest of citizen in mobility choices. Composed by 7 closed questions and an open space for opinions, the questionnaire was developed to be answered quickly. The first five questions aimed to identify the user (age, sex, residence, reasons for traveling in Milan, used means of transport). The last two questions were the core questions: the first to measure the inclination to walk more to reach the tram/bus stop in exchange for a shorter traveling time; the second to assess the inclination to use Taxi Sharing, given a fare of 3,50 Euro.

In section 6.1 we describe the sample that answered the questionnaire, in section 6.2 we illustrate the answers to the question about stops reduction, and in section 6.3 we illustrate the answers to the question about the use of Taxi Sharing.

6.1 Sample

We want to state beforehand that the questionnaire does not have a statistical value, due to the way the survey has been conducted. For this reason, we describe the answers without doing any statistical inference. As a further step, improving the reliability of the survey sample would allow for more accurate evaluations.

The first question is about age; Figure 6.1 represents the age ranges of the people answering the survey.

The second question is about gender; 60% are females and 40% are males.

The third question is about the relation with the city of Milan; 77% of the sample lives in Milan, 14% are commuters and 9% visits the town occasionally.

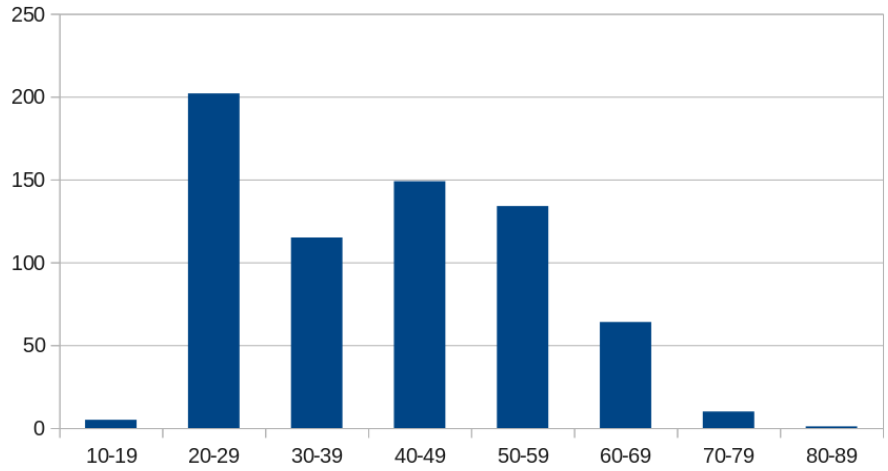


Figure 6.1: The age ranges of the sample

In the fourth question we investigate the main reasons of movement; it is possible to give more than one answer: work, study, shopping and leisure; in Figure 6.2 we present the answers.

In the fifth question we investigate the most used means of transportation; as in the previous case, it is possible to give more than one answer, choosing among: LPT, car, taxi, motorbike, bike and foot; in Figure 6.3 we present the answers.

6.2 Rapid LPT

In this section we present and discuss the results obtained from question 6, which regards the users availability to walk more in return for a faster surface service. In section 5.3 we have shown a case study on tram line 9, were we obtained that the commercial speed could increase by 50% with enslavement of traffic lights, protection measures and optimal stop suppression. Optimal stop suppression can increase commercial speed in the analyzed case up to 17% and hence can proportionally increase the capacity of the line. For this reason optimal stop suppression can support a higher demand of LPT and a correlated lower use of private motorized vehicle and hence, indirectly, facilitate the realization of priority traffic lights and protection measures. The increase by 50% in the commercial speed would not be a direct consequence of the sole optimal stop suppression, but adopting this measure would make this result more achievable.

If commercial speed increases by 50%, a saving of 33% in the traveling time can be obtained, according to the following formula:

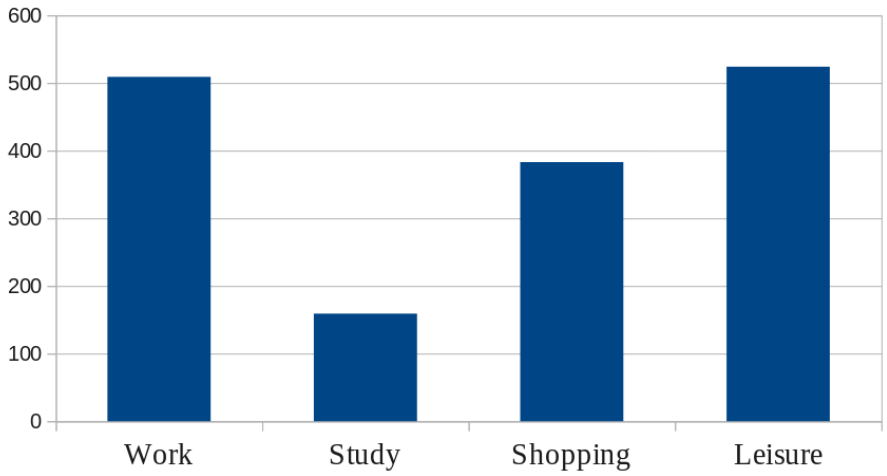


Figure 6.2: The reasons of movement for the sample

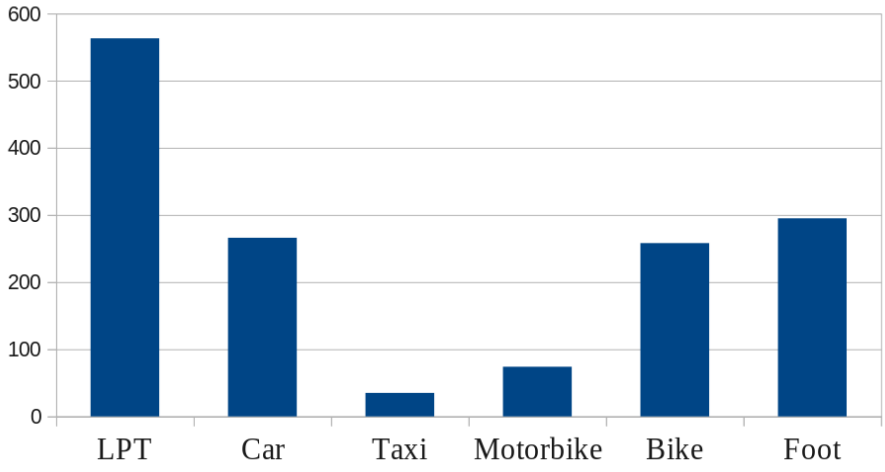


Figure 6.3: The means of transportation used by the sample

$$T_1 = \frac{S}{v_1} \tag{6.2.1}$$

$$v_2 = 1.5\ v_1 \tag{6.2.2}$$

$$T_2 = \frac{S}{v_2} = \frac{S}{1.5\ v_1} = \frac{1}{1.5} T_1 = 0.66\ T_1 \tag{6.2.3}$$

where S is the length of the line, T_1 and v_1 are time and speed in the actual scenario, and T_2 and v_2 are time and speed in the optimized scenario.

For these reasons, we have asked the users the question: "How much longer would you walk to reach the stop, if the Public Transport higher speed will allow you to save 10 minutes on a bus/tram ride which currently takes 30 minutes?".

In Figure 6.4 we show the answers.

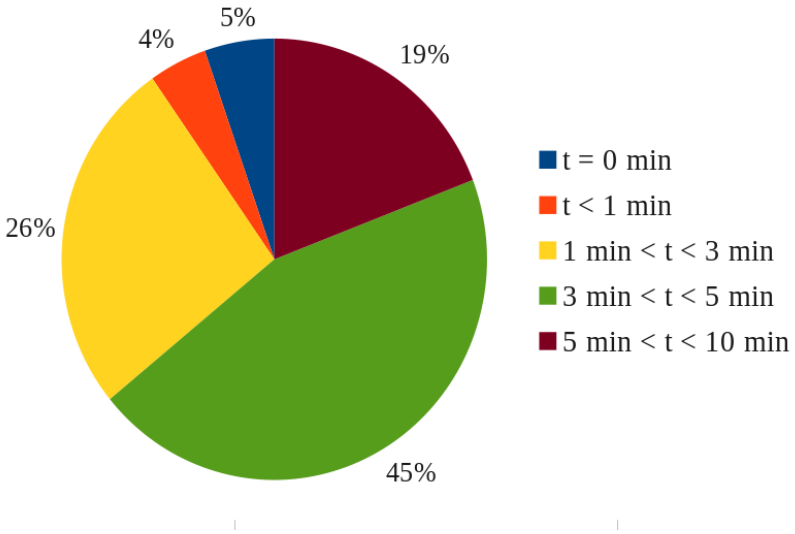


Figure 6.4: Answers to question: "How much longer would you walk to reach the stop, if the Public Transport higher speed will allow you to save 10 minutes on a bus/tram ride which currently takes 30 minutes?"

The data presented in Figure 6.4 show a high users availability to walk some more minutes in exchange for a faster service; we point out that:

- 90% are willing to walk $1-3$ minutes or more.
- 64% are willing to walk $3-5$ minutes or more.

In the case study on tram line 9, presented in section 5.2, in the optimal solution according to stops suppression, the average time necessary to reach the next stop, for users that were using a suppressed stop to board or alight the

tram, is 86 seconds in average. Therefore, in average, even if the departure and destination stops are both suppressed, the longer walking distance in total is less than 3 minutes.

In Figure 6.5 we present the answers of residents and commuters, and we notice that the commuters are more willing to walk more in exchange for a faster trip. This can be due to the fact that their trips are longer and that they are used to longer distances between stops.

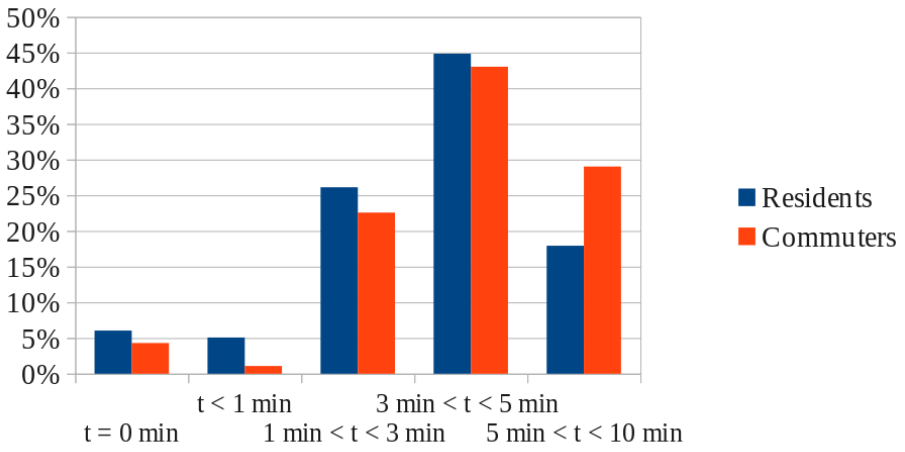


Figure 6.5: Answers to question: "How long would you walk more to reach the stop, if the Public Transport higher speed will allow you to save 10 minutes on a bus/tram ride which currently takes 30 minutes?". according to residence.

6.3 Taxi Sharing

In this section we present and discuss the answers to the question about the possible frequency of utilizing Taxi Sharing service. In section 4 we have shown three different scenarios of Taxi Sharing service according to the demand level and the conditions on the road network in relation to the average speed. The question in the survey refers to the car-free city scenario presented in section 4.3.

We asked the users the question: "If the Taxi Sharing fare for a ride is 3,50 Euros, how often would you use it?". In Figure 6.6 we show the answers.

The data presented in Figure 6.6 show a high potential interest towards Taxi Sharing; we underline that:

- 12% of the sample declared that they would use Taxi Sharing 3-5 times a week or more.

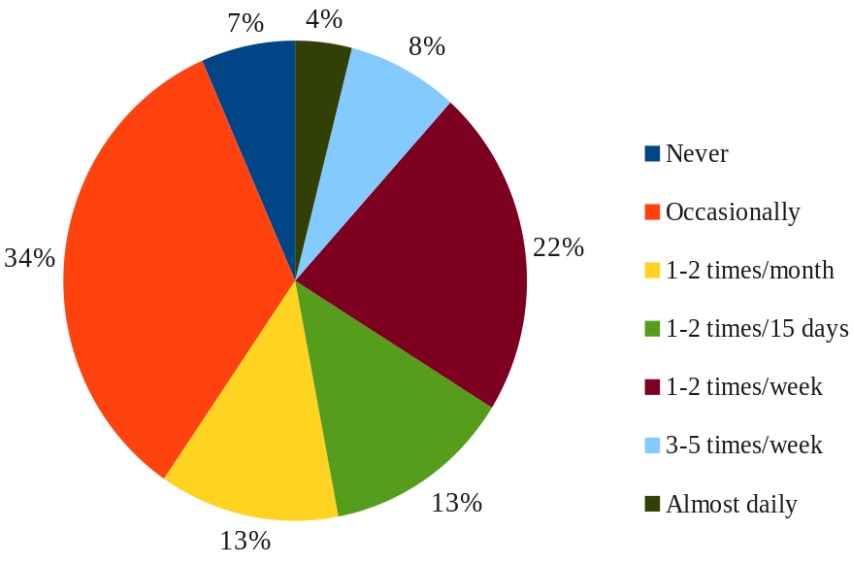


Figure 6.6: Answers to question: "If the Taxi Sharing fare for a ride is 3,50 Euros, how often would you use it?"

- 34% of the sample declared that they would use Taxi Sharing 1-2 times a week or more.

The answers presented in Figure 6.6 could be used to assess the daily demand for Taxi Sharing service as presented in Table 6.1. In the first column we report the options, in the second column we report the percentage of the sample that selected the corresponding option, in the third column we multiplied this percentage by the overall number of potential frequent users, given by the sum of residents (1250000) and commuters (850000) and hence in total 2100000. In the fourth column we converted the options of frequency of use, which were expressed in words, in numbers of daily expected ride per person, in column fifth we multiplied these values by the respective number of users of each group, obtaining the daily expected requests for each group. With the described procedure we estimated the daily number of rides to be in total 288543, that represents 10.5% of the mobility in the city of Milan.

In Figure 6.7 we present the answers of car users and others, and we note that car users declare they would use Taxi Sharing more frequently. This can be due to them being more used to a higher budget, considering congestion charge and parking fee. We underline that 15% of car users declare they would use Taxi Sharing $3-5$ times a week or almost daily.

Options	Answers percentage	Number of persons	Expected daily rides per person	Expected daily rides
Never	7%	137956	0	0
Occasionally	34%	711241	0.01	7112
1-2 times/month	13%	263650	0.05	13182
1-2 times/15 days	13%	275912	0.1	27591
1-2 times/week	22%	465985	0.2	93197
3-5 times/week	8%	162482	0.5	81240
Almost daily	4%	82774	0.8	66218

Table 6.1: Computation of daily Taxi Sharing rides, based on the results of the survey.

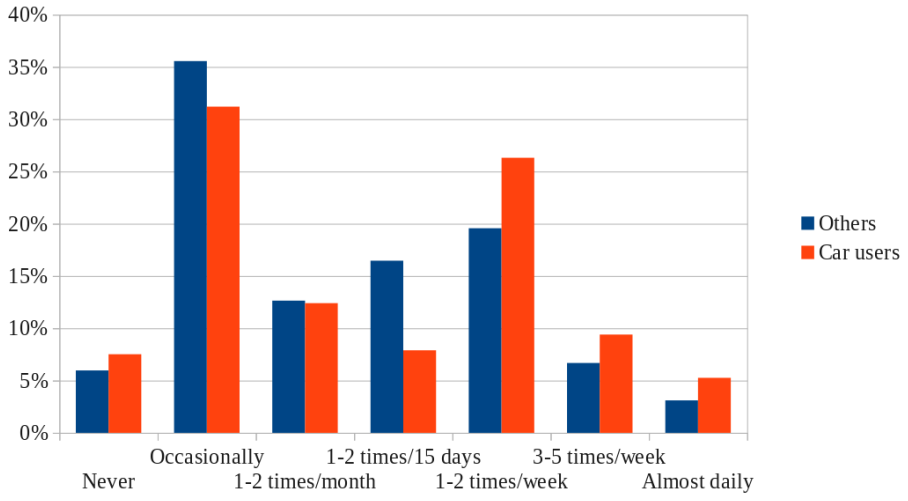


Figure 6.7: Answers to question: "If the Taxi Sharing fare for a ride is 3,50 Euros, how often would you use it?" according to car using.

Conclusion

In this dissertation we assess Taxi Sharing as an urban massive mobility system as well as a correlated rapid transportation network for exploiting the available urban transportation resources, by way of a case study of Milan. The research conducted in collaboration with AMAT has shown a huge possibility for qualitatively and quantitatively increasing public mobility services by using the available resources in the most efficient way, considering both transportation means and human drivers with respect to taxis and LPT.

A new technique for optimizing a high quality spread Taxi Sharing service in an urban context, provided by the existing taxis fleet, is presented. The service is designed to have a high quality, ensured by narrow time windows on pick-up and delivery time. These features allowed us to cyclically enumerate all possible subsets of incoming users requests for each vehicle, and to compute an optimal set of routes in real time by solving, in a few seconds, a large set partitioning problem, with state-of-the-art integer linear programming solvers. Owing to this fast global optimization capability, the system guarantees a high quality service without any need of booking the ride in advance.

The algorithm has been embedded in an agent based simulator of the Taxi Sharing service. The developed simulator enables the authorities of cities to forecast how the service could work with respect to users and taxi drivers. With regard to users, the simulator reports the number of serviced users, the average waiting time and the average detour time. With regard to taxi drivers, the simulator reports the number of taxi rides, the average number of serviced users per hour, the relevant average revenue per hour and the average length of the taxi ride.

The simulator was used to conduct a feasibility study for the city of Milan regarding three development scenarios, according to demand level. The results show that Taxi Sharing level of service is acceptable in the scenario with low demand and it increases markedly in the scenarios with higher demand according to the higher possibility to share the ride with negligible detours.

In the car free city scenario Taxi Sharing can serve up to 10% of the total mobility demand with the current taxis fleet, guaranteeing a shorter traveling time with respect to individual taxi in the actual scenario. Taxi Sharing induces a win-win-win situation since the advantages for users, taxi drivers and municipality derive from the higher efficiency of Taxi Sharing service, that can rise up to more than three time the efficiency of individual taxi. We consider

these results the main scientific contribution of this dissertation, since they show that the development of Taxi Sharing could have great social, economical and environmental impacts[1].

Afterward we analyzed new possibilities for planning rapid Local Public Transportation (LPT) resulting from the realization of Taxi Sharing. We assessed this possibility with a case study on tram line 9 concerning the effects in terms of total variation in the traveling time of users and increase in commercial speed achievable with the optimal suppression of stops. The results show that in the optimal solution 40% of the stops have to be suppressed in order to minimize the traveling time of users, and that the optimal suppression of stop, with protection measures and priority traffic lights, can lead to an increase in commercial speed by 50%.

Finally, we included the results of an informal survey conducted through social networks to assess opinions of citizens on the studied measures. The high and spontaneous participation in the survey shows people's high interest in mobility choices. The answers of the sample, that we recall to be without statistical consistence, show that 90% of the sample would be willing to walk up to 1-3 minutes or longer to save traveling time, and show a high potential demand for Taxi Sharing, in a quantity of about 300.000 daily rides.

In conclusion, we state that Taxi Sharing might have an important role in urban mobility, since its implementation allows for achieving a more sustainable and efficient mobility:

1. With the existing taxis fleet, Taxi Sharing can satisfy 10% of the total mobility, providing a high level of service, also to people with movements impairments;

2. A rapid Local Public Transportation (LPT) can be planned if Taxi Sharing take care of people with movements impairments, and, with an increase in the commercial speed up to 50%, the LPT level of service can improve;

3. In a low congested scenario, that can be achieved thanks to a more efficient public mobility system with regards to Taxi Sharing and rapid LPT, the role of soft mobility would increase, entailing positive impacts.

With regards to Taxi Sharing, a further research step is the dynamic adaption of parameters according to service conditions. On the other hand, in this dissertation we assessed the possibility to plan a rapid LPT with respect to a line, considering the rest of the network as a datum. Presuming the presence of Taxi Sharing, a further research step in this direction would be to investigate the problem of an optimal network design.

[1] The results obtained for the city of Milan are not linked with any peculiarity of Milan and, hence, similar results are likely to be obtained in others cities.

Appendices

Appendix A

Parameter β calibration

In this appendix we assess the effects of parameter β[1] on the quality and efficiency of the service, and on the computational complexity. For every one of these areas of interest we do a focus on the three analyzed scenarios: low demand, medium demand and car-free. For every scenario we focus our analysis on $hour = 14$, to avoid the effects due to different levels of demand, and different origin/destination matrix in different hours of day. For every scenario we present the results of 10 different simulations according to the value of β ranging from β_{min} to β_{max}. In order to limit the aleatory effects, the results presented in the next sections refer to average values of a 5 hours long simulation, supposing all the conditions and parameters equal those of $hour = 14$, as presented in the chapter 4, with the exception of parameter β that ranges in $[\beta_{min}, \beta_{max}]$. The value of β used at $hour = 14$ in the scenario low demand, presented in section 4.1, is $\beta = 0.8$. For this scenario we considered 10 values of $\beta \in [0.2, 2]$. The value of β used at $hour = 14$ in the scenario medium demand, presented in section 4.2, is $\beta = 0.6$. For this scenario we considered 10 values of $\beta \in [0.1, 1]$. The value of β used at $hour = 14$ in the scenario car free, presented in section 4.3, is $\beta = 0.3$. For this scenario we considered 10 values of $\beta \in [0.05, 0.5]$.

Quality of the service

We access the effect of β on the quality of the service through three main indicators: the number of refused requests, the average waiting time and the average detour time. In Figure A.1, Figure A.2, Figure A.3 we present the percentages of refused requests in the three scenarios. We underline that the percentages of refused requests are decreasing more than linearly according to β.

In Figure A.4, Figure A.5, Figure A.6 we present the average waiting and detour time in the three scenarios. We notice that both waiting and detour time

[1] The average effects of parameter α are the same of parameter β, the only difference is that a higher α mainly penalizes short requests while a higher β mainly penalizes long requests.

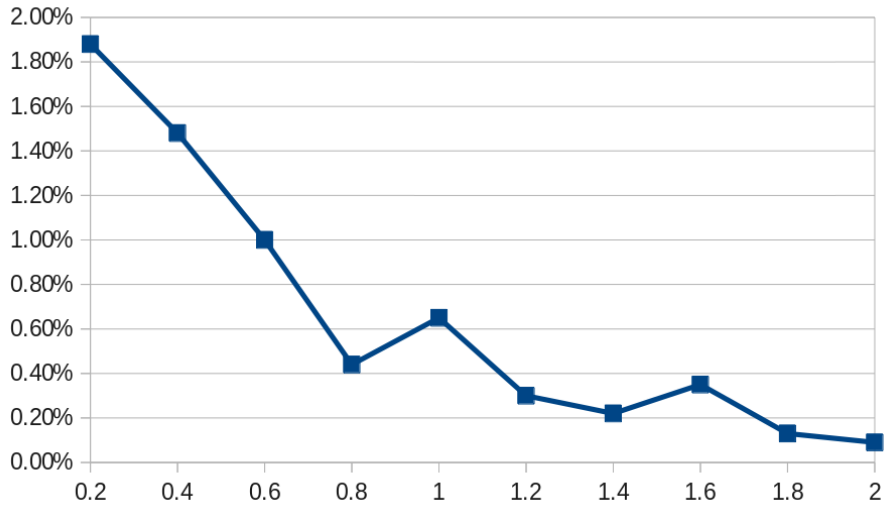


Figure A.1: Percentage of refused requests according to the value β, in low demand scenario.

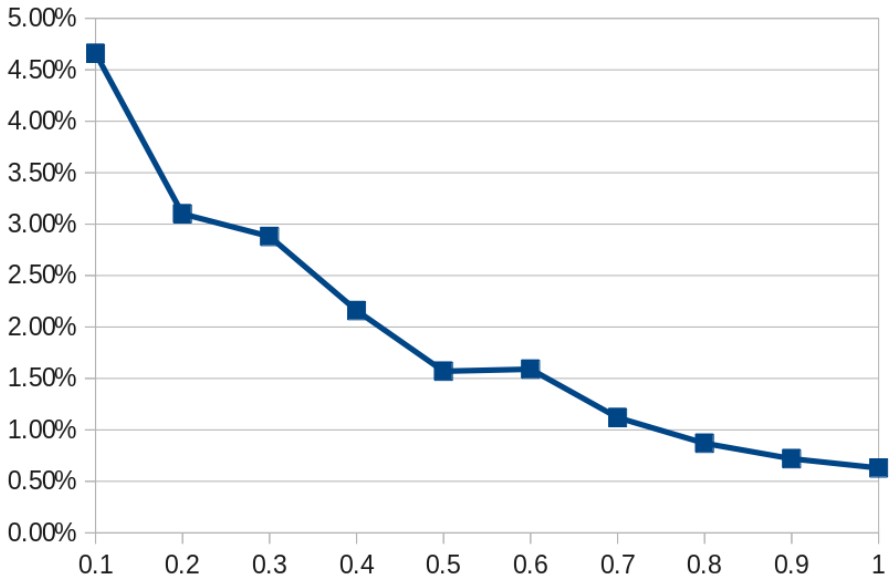


Figure A.2: Percentage of refused requests according to the value β, in medium demand scenario.

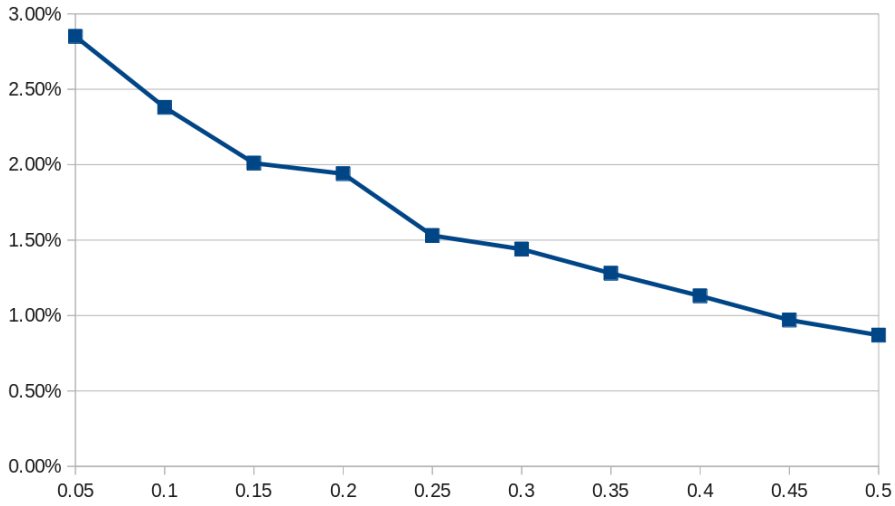


Figure A.3: Percentage of refused requests according to the value β, in car free scenario.

are increasing linearly according to β, as a consequence of the higher maximum arrival time to destination. In the next section we access the effects of β on the efficiency of the service.

Efficiency of the service

We access the effect of β on the efficiency of the service through the average number of served users per hour by each taxi. In Figure A.7, Figure A.8, Figure A.9 we present the average number of served users per hour in the three scenarios. We notice that the number of served users per hour increases steeply in the beginning and tends to stabilize for higher values of β. We underline the difference with the service quality according to waiting and detour times that instead increase linearly. This brings to the conclusion that for high values of β, the further increase of β entails a reduction in the quality of the service with little gain in the efficiency of the service. In the next section we access the effects of β on the computational complexity.

Computational complexity

We access the effect of β on the computational complexity through five main indicators: the number of open and confirmed requests, the number of groups and permutations, the time needed to generate all the groups and permutations

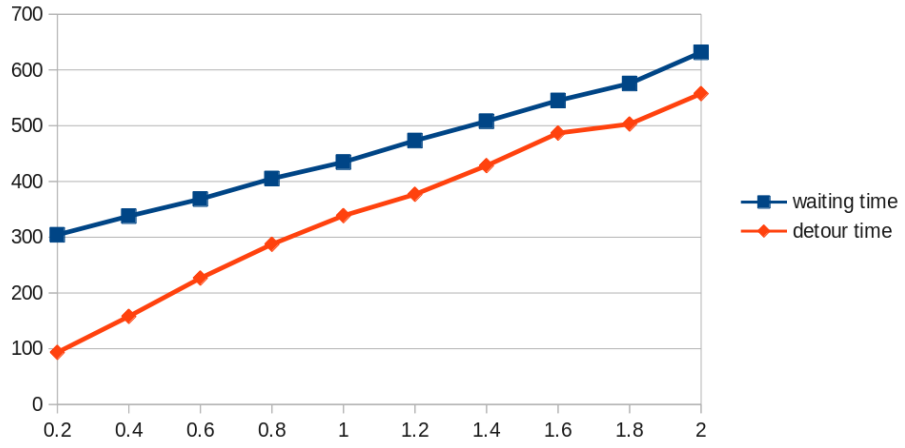


Figure A.4: Average waiting and detour time in seconds according to the value β, in low demand scenario.

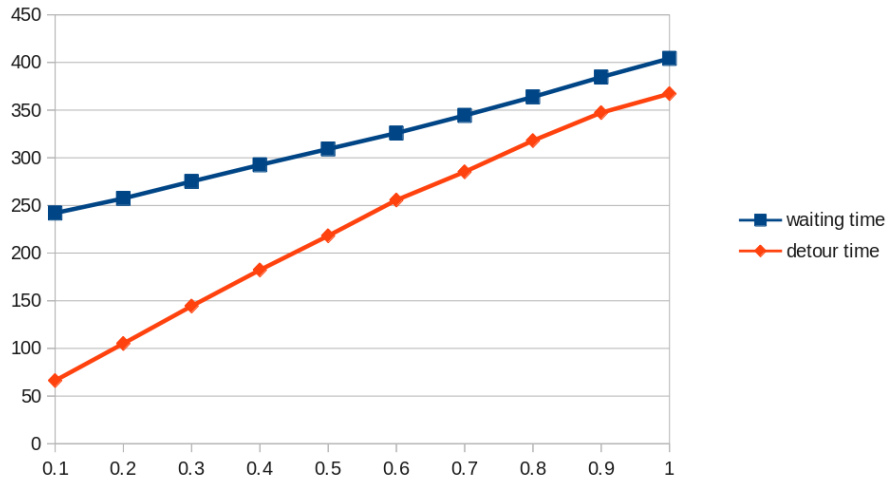


Figure A.5: Average waiting and detour time in seconds according to the value β, in medium demand scenario.

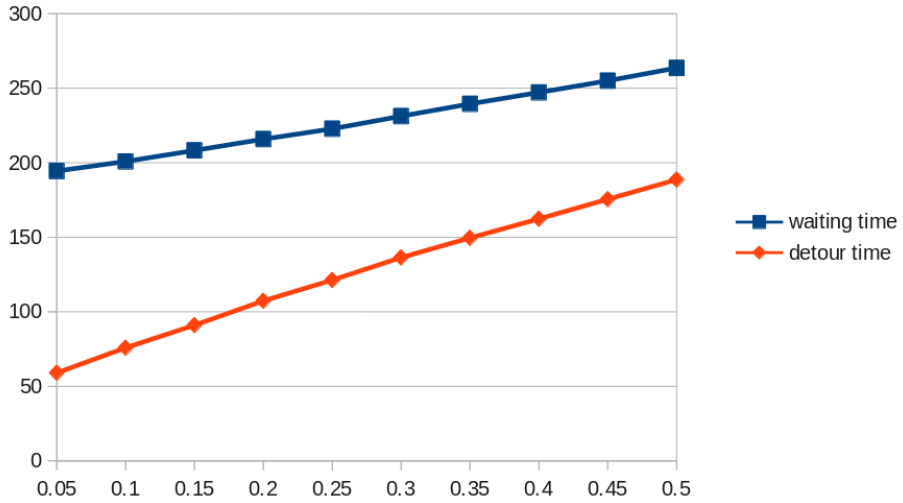


Figure A.6: Average waiting and detour time in seconds according to the value β, in car-free scenario.

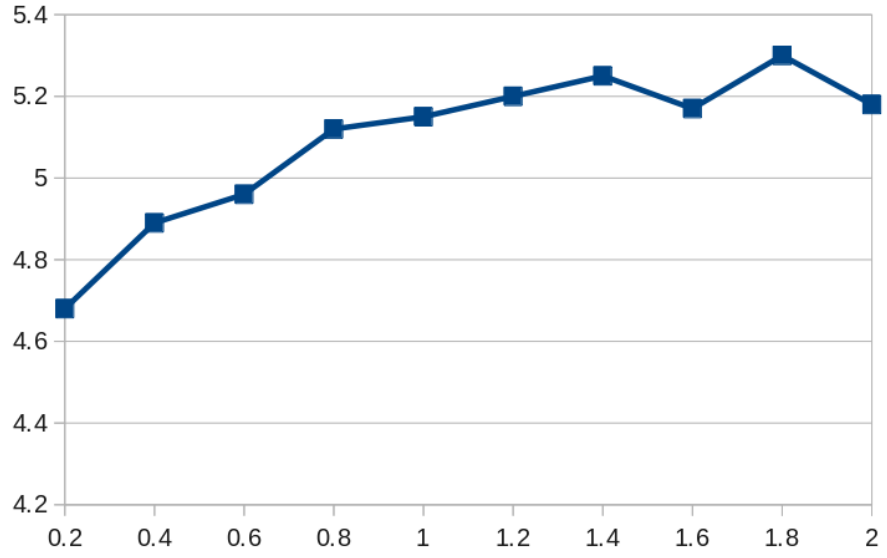


Figure A.7: Average number of served users per hour according to the value β, in low demand scenario.

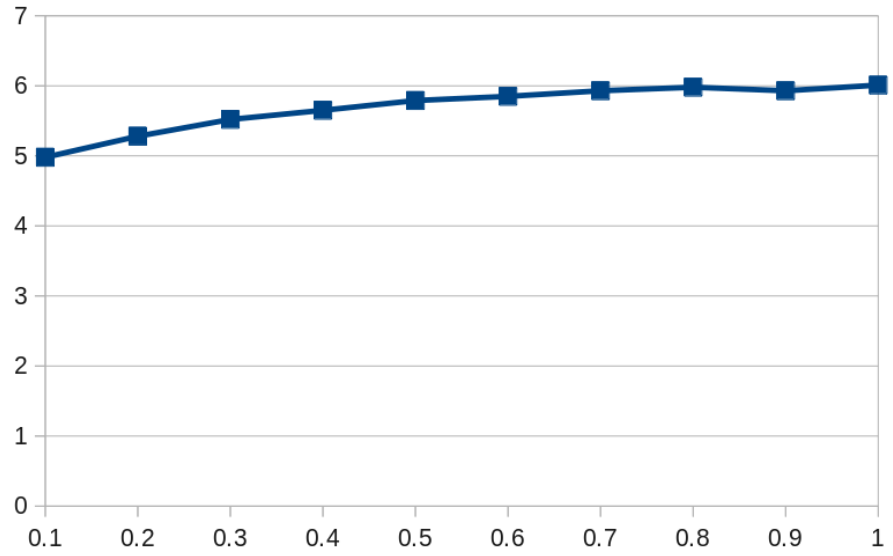


Figure A.8: Average number of served users per hour according to the value β, in medium demand scenario.

and the time needed to solve the set partitioning problem. In Figure A.10, Figure A.11, Figure A.12 we present the number of open and confirmed requests. We notice that the number of open and confirmed requests is increasing linearly according to β, as a consequence[2] of the higher waiting time that also increases linearly as presented in the previous section.

In Figure A.13, Figure A.14, Figure A.15 we present the number of groups and permutations in the three scenarios. We notice that the number of groups and permutations is increasing more then linearly according to β as a consequence of the higher number of open and confirmed requests and the higher marginal times. We notice that the number of groups and permutation would explode for higher values of β, and this would bring computational issues.

In Figure A.16, Figure A.17, Figure A.18 we present the time in milliseconds needed to generate groups and permutations in the three scenarios. We notice that the time is increasing more then linearly according to β. However even in the scenario where the time is the highest, the time is below 0.5 seconds.

In Figure A.19, Figure A.20, Figure A.21 we present the time in milliseconds needed to solve the set partitioning problem in the three scenarios. We notice that the time is increasing more then linearly according to β. However even in

[2]A request is fixed by the optimizer when its departure is going to be visited by the taxi soon. For this reason, longer waiting time imply the requests to remain open longer resulting in a higher number of open requests.

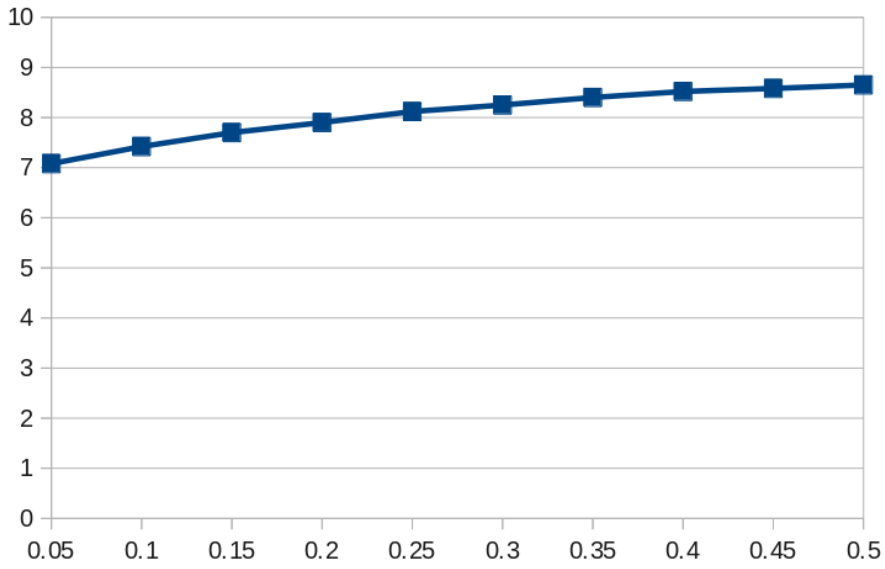


Figure A.9: Average number of served users per hour according to the value β, in car free scenario.

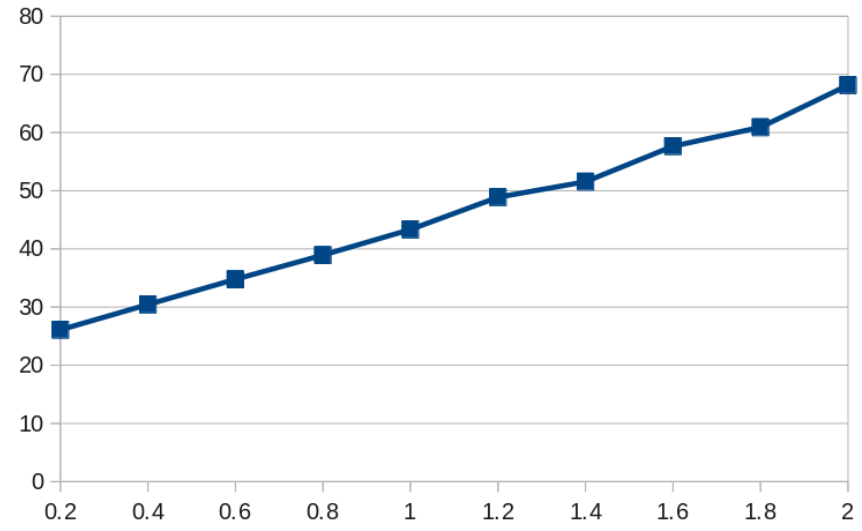


Figure A.10: Average number of open and confirmed requests according to the value of β, in low demand scenario.

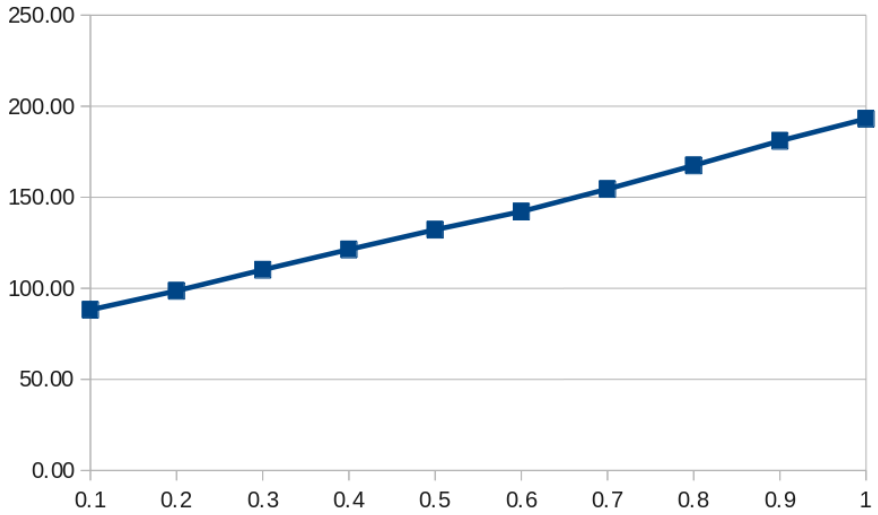


Figure A.11: Average number of open and confirmed requests according to the value of β, in medium demand scenario.

the scenario where the time is the highest, the time is below 2.5 seconds.

To conclude, we underline that for values of β higher than those used in the presented simulation, the number of groups and permutations would explode. This would lead to a huge time needed to generate groups and permutations and to solve the set partitioning problem and, hence, the proposed method would not be suitable.

Though for value higher than those used in the presented simulation the efficiency of the service would increase marginally at the cost of a linear decreasing of the service quality and, hence, they can be excluded from the range of suitable values of β, with respects to the quality and efficiency of the service in all the analyzed scenarios.

The fact that the values of β that lead to optimal service are the same that are in the range of solvability of the problem with the proposed method, is related to the fact that the service works properly with an adequate number of groups. A huge number of groups, that would be unmanageable, would imply little gain in the efficiency at the cost of a consistent decrease in the service quality.

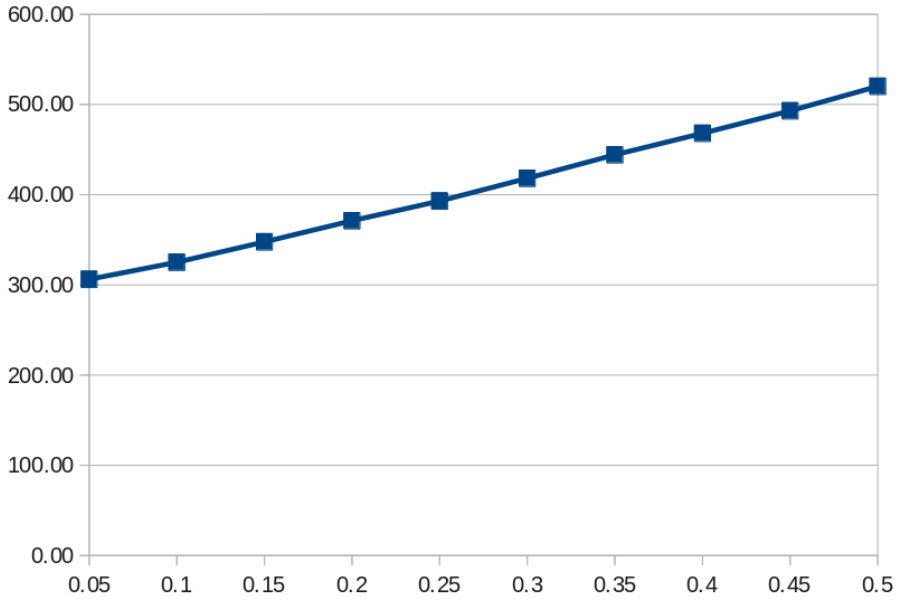


Figure A.12: Average number of open and confirmed requests according to the value of β, in car free scenario.

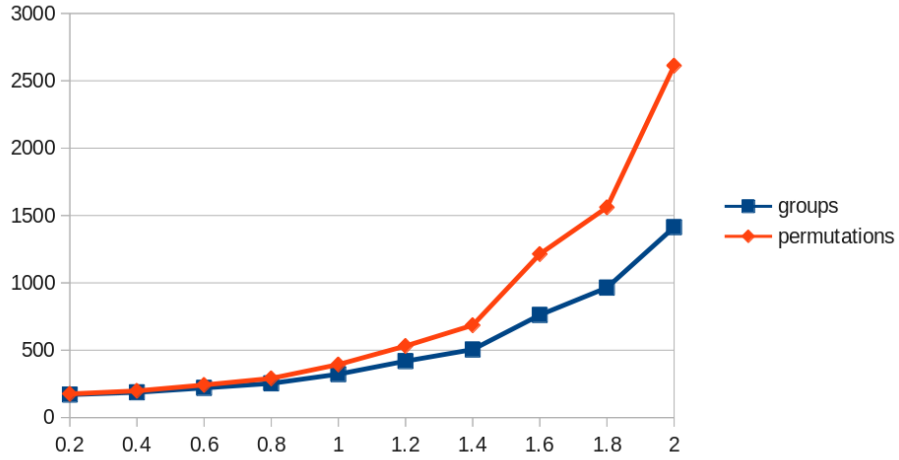


Figure A.13: Average number of groups and permutation according to the value of β, in low demand scenario.

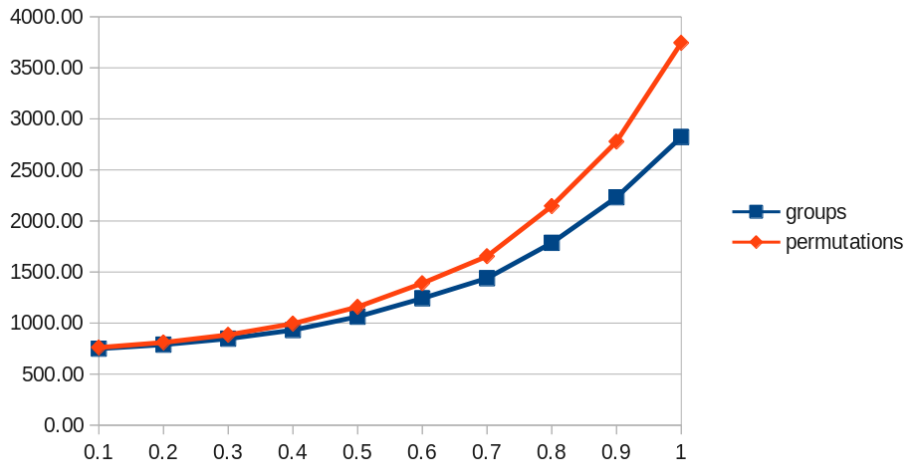


Figure A.14: Average number of groups and permutation according to the value of β, in medium demand scenario.

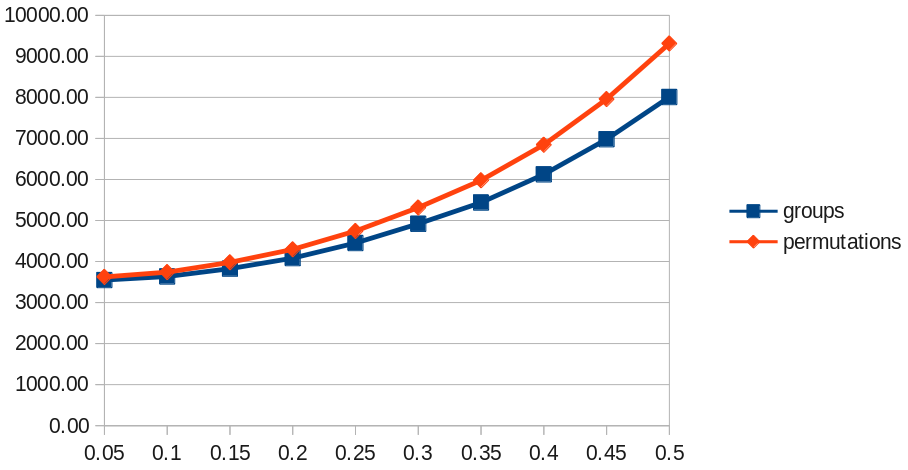


Figure A.15: Average number of groups and permutation according to the value of β, in car free scenario.

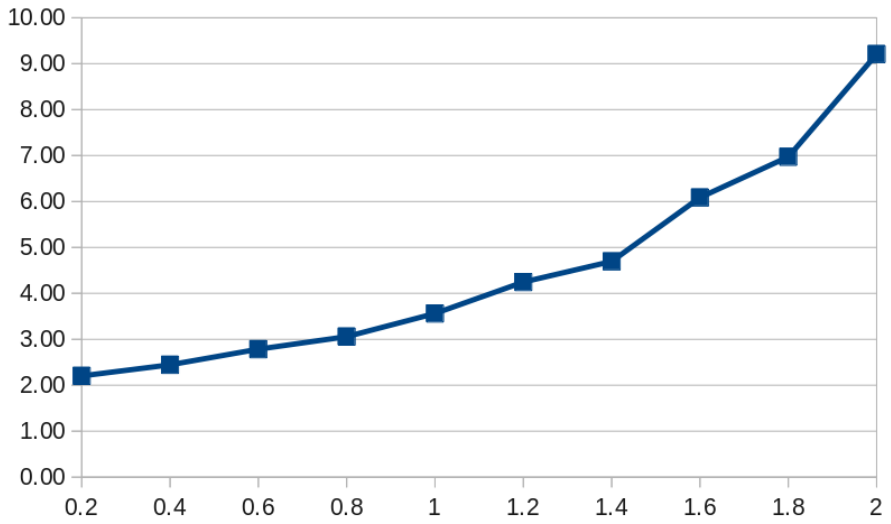


Figure A.16: Average time in milliseconds needed to generate groups and permutations according to the value of β, in low demand scenario.

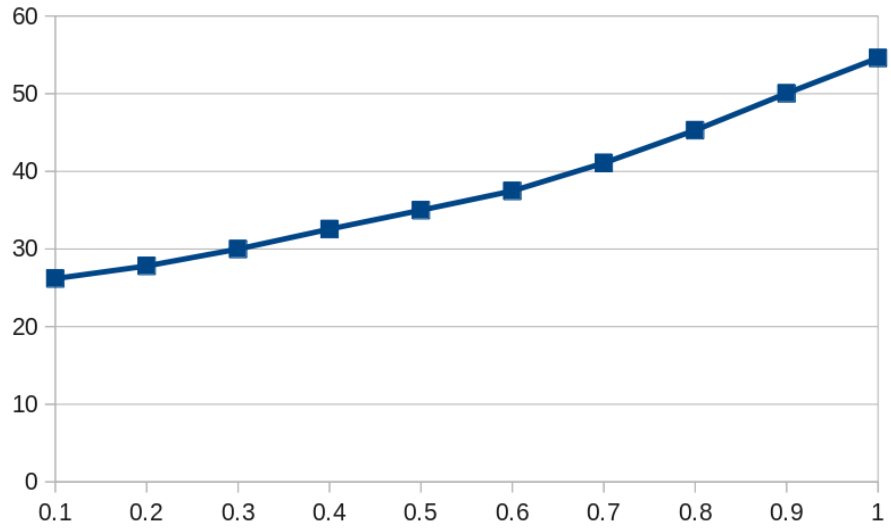


Figure A.17: Average time in milliseconds needed to generate groups and permutations according to the value of β, in medium demand scenario.

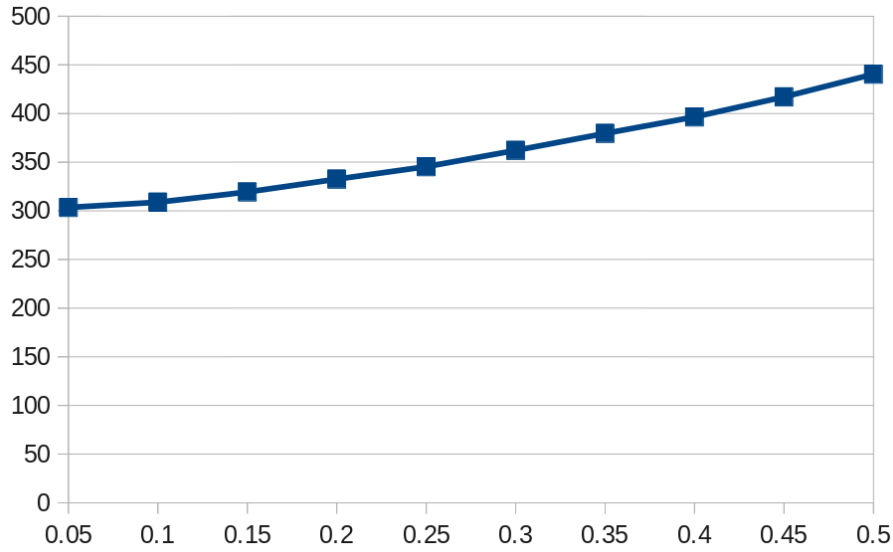


Figure A.18: Average time in milliseconds needed to generate groups and permutations according to the value of β, in car free scenario.

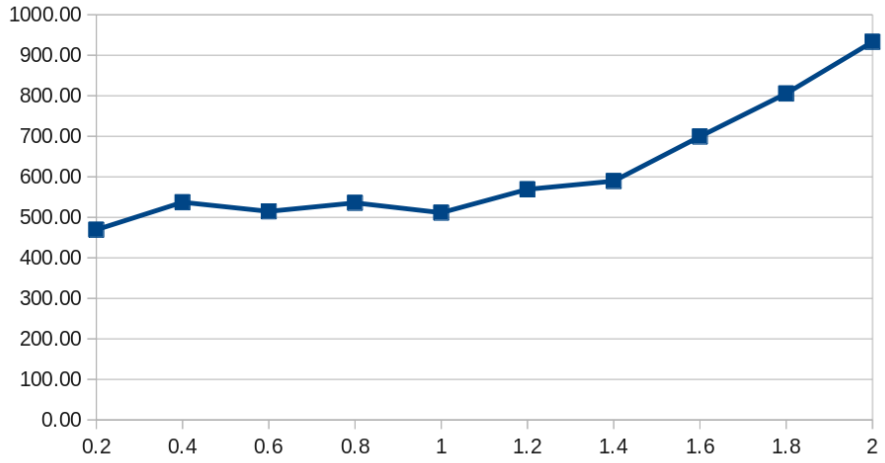


Figure A.19: Average time in milliseconds needed to solve the set partitioning problem according to the value of β, in low demand scenario.

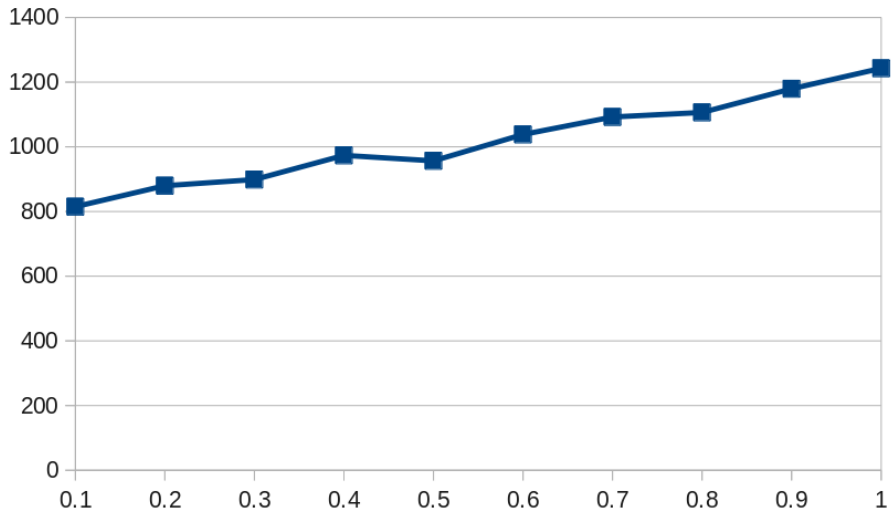


Figure A.20: Average time in milliseconds needed to solve the set partitioning problem according to the value of β, in medium demand scenario.

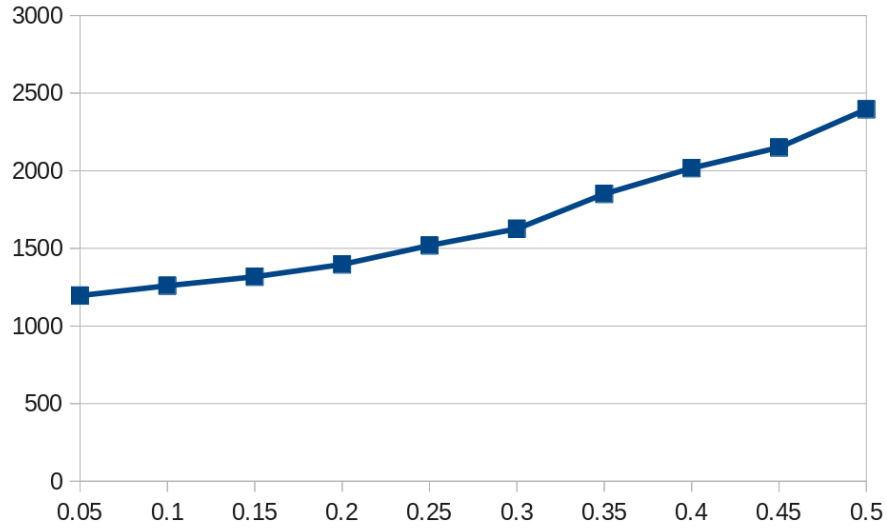


Figure A.21: Average time in milliseconds needed to solve the set partitioning problem according to the value of β, in car free scenario.

Appendix B

Dynamic Vehicle Routing Problems framework

In this appendix we collocate the developed Taxi Sharing optimization method in the framework of Dynamic Vehicle Routing Problems (DVRPs). With regards to recent published survey papers on this topic in the last decades the research had been abundant: the surveys [38, 39, 40] report respectively 154, 161 and 166 articles.

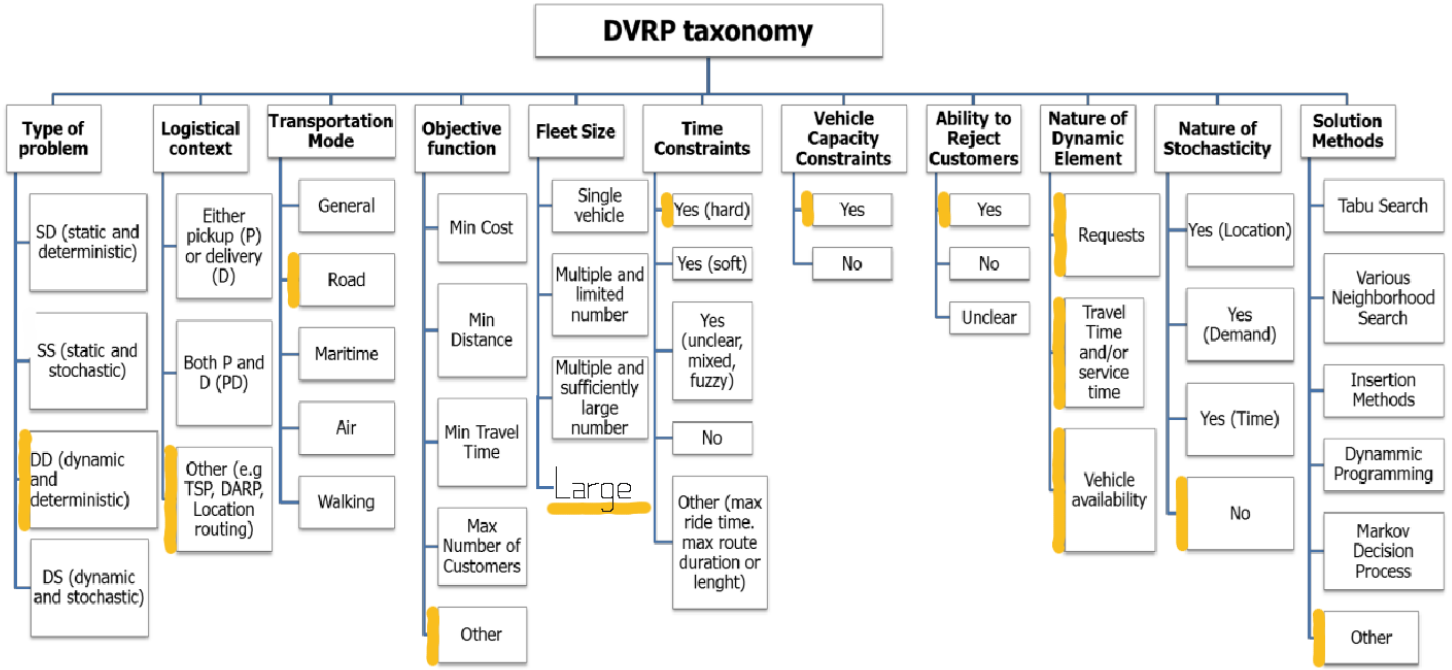
In the next figure, on page 113, we classify our problem according to the taxonomy presented in the most recent survey paper on the topic [40]. The taxonomy is based on the following 11 criteria: (1) type of problem, (2) logistical context, (3) transportation mode, (4) objective function, (5) fleet size, (6) time constraints, (7) vehicle capacity constraints, (8) the ability to reject customers, (9) the nature of the dynamic element, (10) the nature of the stochasticity (if any), and (11) the solution method.

Regarding the nature of dynamism the main source in our problem are requests. The degree of dynamism δ can be defined, as in [41], as the ratio between the number of dynamic requests n_d and the total number of requests n_{tot} as follows:

$$\delta = \frac{n_d}{n_{tot}} \tag{B.0.1}$$

in our problem we have a degree of dynamism $\delta = 100\%$, since all the requests are dynamic and, furthermore, all requests are imminent.

Also taxi availability and road condition are dynamic. The frequency for updating taxi availability is every few seconds, i.e. every reoptimization cycle. The frequency for updating the matrix of traveling time, since the computation is fully parallelizable, is only limited by the available hardware and the availability of updated information on the road conditions. According with the data presented in chapter 4, with 64 dedicated parallel processes, the distances matrix related to the city of Milan, can be updated with a frequency higher


DVRP taxonomy
Type of problem
SD (static and deterministic)
SS (static and stochastic)
DD (dynamic and deterministic)
DS (dynamic and stochastic)
Logistical context
Either pickup (P) or delivery (D)
Both P and D (PD)
Other (e.g TSP, DARP, Location routing)
Transportation Mode
General
Road
Maritime
Air
Walking
Objective function
Min Cost
Min Distance
Min Travel Time
Max Number of Customers
Other
Fleet Size
Single vehicle
Multiple and limited number
Multiple and sufficiently large number
Large
Time Constraints
Yes (hard)
Yes (soft)
Yes (unclear, mixed, fuzzy)
No
Other (max ride time. max route duration or lenght)
Vehicle Capacity Constraints
Yes
No
Ability to Reject Customers
Yes
No
Unclear
Nature of Dynamic Element
Requests
Travel Time and/or service time
Vehicle availability
Nature of Stochasticity
Yes (Location)
Yes (Demand)
Yes (Time)
No
Solution Methods
Tabu Search
Various Neighborhood Search
Insertion Methods
Dynammic Programming
Markov Decision Process
Other

than once every 5 minute[1].

We do not take into account stochasticity related to no show in our problem, since all the request are forwarded only when the taxi is desired, and hence the probability that they are canceled is low. As a service implementation detail, we consider that the fee is paid when the ride is call and, hence, this implies an even lower probability of canceled requests. From a service performance perspective, the fact that the higher the number of already paid rides canceled the higher the quality of the service, means that the presented results represent a lower bound on the effective quality of the service. Concerning travel time we do not consider stochasticity, but we consider to update periodically the road traveling time, and the entire service is reoptimized every few seconds with the last updated distances matrix.

Regarding the fleet size, that is related to the number of requests to be serviced, the survey [40] reports that "most of the papers we have reviewed belong to the multiple and limited number of vehicles category". Concerning the few articles reported in the survey, related to the category of fleet size "Multiple and sufficiently large number", the research on DARP presented in [42] reports a number of 762 requests in 10 hours of service for an hospital service. The number of requests, related to other articles in this category, are of the same order of magnitude.

In the car free scenario, presented in section 4.3, the number of handled requests is 289.885, and hence three order of magnitude higher than that presented in [42]. For this reasons we had considered stating that our problem stand out, according to fleet size, the three categories introduced in the survey [40] and, hence, we add the category "Large". This is due to the fact that, while most of the articles related to DARP focus the attention on special category of users, we assess an urban massive transportation service.

The "Solution Method" is "Other" with respects to those listed in the taxonomy presented in [40], since it is possible to enumerate all possibles groups and permutation. This is an indirect consequence of the high number of requests and vehicles that allow no-booking polices and strict time windows on arrival time. This approach had been designed to optimize a massive on demand service with the described characteristics, and it has showed itself to work efficiently in all the analyzed scenarios.

[1]We underline that according to distances computation also other approaches can be followed, and the more efficient depends on the frequency in travel times data updating.

Bibliography

[1] Teal, R. (1981). *Shared-ride taxi services as community public transit: final report.* The Administration.

[2] Brake, J., Nelson, J. D., & Wright, S. (2004). *Demand responsive transport: towards the emergence of a new market segment.* Journal of Transport Geography, 12(4), 323-337.

[3] Jia, Y., Wang, C., and Wang, L. (2009). *A rolling horizon procedure for dynamic pickup and delivery problem with time windows.* International Conference on Automation and Logistics, 1:2087-2091.

[4] Jaw, J.-J., Odoni, A. R., Psaraftis, H. N., and Wilson, N. H. M. (1986). *A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows.* Transportation Research Part B: Methodological, 20(3):243-257.

[5] Madsen, O. B. G., Ravn, H. F., and Rygaard, J. M. (1995). *A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives.* Annals of Operations Research, 60:193-208.

[6] A. Colorni and G. Righini, *Modeling and optimizing dynamic dial-a-ride problems.* International Transactions in Operational Research, 8: 155-166, 2001.

[7] Cordeau, J.-F. and Laporte, G. (2002b). *A tabu search heuristic for the static multi-vehicle dial-a-ride problem.* Transportation Research, 37:579-594.

[8] Berbeglia, G., Cordeau, J.-F., and Laporte, G. (2009). *A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem.* HEC Montréal.

[9] W.R. Jih, C.Y. Kao and F.Y.J. Hsu. *Using Family Competition Genetic Algorithm in Pickup and Delivery Problem with Time Window Constraints.* In Proceedings of the 2002 IEEE International Symposium on Intelligent Control, Vancouver, Canada, 496 - 501, 2002.

[10] F.B. Pereira, J. Tavares, P. Machado and E. Costa. GVR: *a New Genetic Representation for the Vehicle Routing Problem.* Artificial Intelligence and

Cognitive Science. 13th Irish Conference, AICS 2002. Proceedings (Lecture Notes in Artificial Intelligence Vol. 2464), 95 - 102, 2002.

[11] Cordeau, J. F., & Laporte, G. (2007). *The dial-a-ride problem: models and algorithms.* Annals of Operations Research, 153(1), 29-46.

[12] Grootenboers, F., De Weerdt, M., & Zargayouna, M. (2010). *Impact of competition on quality of service in demand responsive transit.* In Multiagent System Technologies (pp. 113-124). Springer Berlin Heidelberg.

[13] Palmer, K., Dessouky, M., & Abdelmaguid, T. (2004). *Impacts of management practices and advanced technologies on demand responsive transit systems.* Transportation Research Part A: Policy and Practice, 38(7), 495-509.

[14] Lee, K. T., Lin, D. J., & Wu, P. J. (2005). *Planning and design of a taxipooling dispatching system.* Transportation Research Record: Journal of the Transportation Research Board, (1903), 86-95.

[15] Mulley, C., & Nelson, J. D. (2009). *Flexible transport services: A new market opportunity for public transport.* Research in Transportation Economics, 25(1), 39-45.

[16] Jean-Franois Cordeau, Gilbert Laporte, (2002a). *The Dial-a-Ride Problem (DARP): Variants, modeling issues and algorithms.* 4OR: A Quarterly Journal of Operations Research, 1:89-101.

[17] Diana, M. and Dessouky, M. M. (2004). *A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows.* Transportation Research Part B: Methodological, 38(6):539-557.

[18] Ahuja, Ravindra K., 1956- & Dial, Robert B. & Orlin, James B., 1953-, 1998. *Fast infeasibility detection algorithms for dial-a-ride transit problems*, Massachusetts Institute of Technology (MIT), Sloan School of Management, WP 4005-98, 1998.

[19] Fisher ML and Jaikumar R, *A generalised assignment heuristic for vehicle routing*, Networks 11, 109-124 (1981).

[20] Martinez, L. M., Correia, G. H., & Viegas, J. M. (2015). *An agent-based simulation model to assess the impacts of introducing a shared-taxi system: an application to Lisbon (Portugal).* Journal of Advanced Transportation, 49(3), 475-495.

[21] Xiang, Z., Chu, C., & Chen, H. (2006). *A fast heuristic for solving a large-scale static dial-a-ride problem under complex constraints.* European Journal of Operational Research, 174(2), 1117-1139.

[22] Li, H., & Bertini, R. (2009). *Assessing a model for optimal bus stop spacing with high-resolution archived stop-level data.* Transportation Research Record: Journal of the Transportation Research Board, (2111), 24-32.

[23] Caldwell, T., *On finding minimum routes in a network with turn penalties*, Communications of the ACM 4(2), 107-108 (1961).

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

[25] Ahuja, R. K., Mehlhorn, K., Orlin, J., & Tarjan, R. E. (1990). *Faster algorithms for the shortest path problem.* Journal of the ACM (JACM), 37(2), 213-223.

[26] Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: a guide to NP-completeness.*

[27] Davidsson, P., Henesey, L., Ramstedt, L., Tarnquist, J., & Wernstedt, F. (2005). *An analysis of agent-based approaches to transport logistics.* Transportation Research part C: emerging technologies,Â 13(4), 255-271.

[28] Roorda, M. J., Cavalcante, R., McCabe, S., & Kwan, H. (2010). *A conceptual framework for agent-based modelling of logistics services.* Transportation Research Part E: Logistics and Transportation Review, 46(1), 18-31.

[29] Weyns, D., Omicini, A., & Odell, J. (2007). *Environment as a first class abstraction in multiagent systems.* Autonomous agents and multi agent systems,14(1), 5-30.

[30] Liaw C.F., White C.C., Bander J. (1996) *A decision support system for the bimodal dial-a-ride problem* IEEE Transactions on Systems, Man and Cybernetics Part A: Systems and Humans 26(5), 552-565.

[31] Baaj, M. H., & Mahmassani, H. S. (1995). *Hybrid route generation heuristic algorithm for the design of transit networks.* Transportation Research Part C: Emerging Technologies, 3(1), 31-50.

[32] Ceder, A., & Wilson, N. H. (1986). *Bus network design.* Transportation Research Part B: Methodological, 20(4), 331-344.

[33] Bielli, M., Caramia, M., & Carotenuto, P. (2002). *Genetic algorithms in bus network optimization.* Transportation Research Part C: Emerging Technologies, 10(1), 19-34.

[34] Mandl, C. E. (1980). *Evaluation and optimization of urban public transportation networks.* European Journal of Operational Research, 5(6), 396-404.

[35] Darbéra, R. (2010). *Taxicab regulation and urban residents' use and perception of taxi services: a survey in eight cities.* In 12th World Conference on Transport Research (p. 36).

[36] United Nations Department of Economic and Social Affairs, Population Division. 2011. *World Population Prospects. The 2010 Revision.* New York: United Nations.

[37] Commissione delle Comunita' Europee, *Proposta per un programma di ricerca sulle citta' senza auto*, Rapporto finale della ricerca *Citta' senza auto*, 1992.

[38] Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). *A review of dynamic vehicle routing problems.* European Journal of Operational Research, 225(1), 1-11.

[39] Bektas, T., Repoussis, P. P., & Tarantilis, C. D. (2014). *Dynamic Vehicle Routing Problems. Vehicle Routing: Problems, Methods, and Applications*, 18, 299.

[40] Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2015). *Dynamic vehicle routing problems: Three decades and counting.* Networks.

[41] Lund, K., Madsen, O. B., & Rygaard, J. M. (1996). *Vehicle routing with varying degree of dynamism.*

[42] Schilde, M., Doerner, K. F., & Hartl, R. F. (2014). *Integrating stochastic time-dependent travel speed in solution methods for the dynamic dial-a-ride problem.* European journal of operational research, 238(1), 18-30.