Université Libre de Bruxelles

Faculty of Science

Computer Science Dept.

MEMO-F-508 : Master Thesis

Traffic Assignment Problem : Shortest Path Optimization in Directed Graph with Multi-agent Population and Central (partial) Authority

Authors

Vajda Quentin-Emmanuel

May 13, 2015

Academic year 2014 - 2015

Contents

1 Introduction

The subject of this master thesis is the traffic assignment problem (*TAP*) in both static and dynamic traffic networks.
This first part covers the basics surrounding the problem as well as a summary of state-of-the-art techniques to solve it. The following section is an overall review of all the concepts and definitions associated with traffic assignment. After that, a second section will cover the most effective and well-known analytical methods. The third and last section of this first part will introduce the simulation-based approach and more specifically the agent-based simulations.

The second part will delve deeper in agent-based simulations capabilities to solve and optimize the *TAP*. More precisely, a study of the effect of external information, such as route guidance systems, on the global travel time of traffic. In particular the use of a central authority dispensing trip guidance to a certain portion of the drivers' population. This makes the authority a service provider to the directed agents.
The design of such a service provider will be defined in greater details in this second section, with extra attention brought to its two main components:

- The learner - responsible for the correct prediction of the current traffic network state based on the data mining of all previous travel times observed
- The (request) handler - responsible for the exploitation of the predictions in order to optimize global travel time by dispensing the shortest path to any users asking for guidance

Different variations of both these main component have been designed and implemented - the specifics of which will be found in the second part of this master thesis, along with the details of the simulation process.

A third section will be consecrated to the benchmarking of the different implemented methods as well as an analysis of the consequences of changes in the proportion of drivers having access to the service provider.

2 The Traffic Assignment Problem & General Concepts

This section covers the general concepts associated with traffic assignment. These concepts range from what is a traffic network and how to represent it, to the most interesting properties of such networks.
Taken into account here is both the computational optimization aspect and the real life equivalent of these concepts.

2.1 The Network

The traffic networks considered here are close representation of real life networks. As defined by [35], a traffic network can be represented by a directed graph where each link represent a street and each node is a road intersection. It is possible to associate a link to each lane, however, the modeling of a vehicle changing lane would add much more complexity. Some other transit networks use bidirectional links in their representation.

Commonly, the traffic is represented by origin-destination (O-D) rates, usually in the form of matrix. Each pair of nodes in the network can be a different O-D pair. The demands represent the number of drivers coming from the origin node and going to the destination node. This is also sometimes referred to as the trip departure matrix.

The assignment of traffic in the network can be done on two different levels. At the macroscopic level, the flows of drivers having in common the same O-D pair are assigned to entire routes through the network. On a much smaller scale, traffic can be assigned at a microscopic level where each individual driver is given its own route to take.
Clearly the later is closer to reality, however there is a balancing issue due to its much higher complexity and computational requirements. The macroscopic approach is more common in analytical models where computational costs are inherently dependent on the number of assignments ; whereas the individual assignment tends to be preferred for simulation based approaches.

The users of a network are considered reasonable in their route choice. Their decision is based on the information they possess and is deterministic : no user will deliberately take a worse route if he knows that their exists a better

one. However, the idea of "worse" and "better" route varies widely due to personal preferences of each motorist and to the kind of information perceived. In general it is accepted that the quantity minimized in the "best" route is the travel time a user perceives over it.

Table 1: Basic Traffic Network Notations

$\mathcal{N}$	Nodes set of network
$\mathcal{A}$	Arcs/links set of network
$\mathcal{R}$	Set of origin nodes $\mathcal{R} \subset \mathcal{N}$
$\mathcal{L}$	Set of destination nodes $\mathcal{L} \subset \mathcal{N}$
$\mathcal{K}_{rs}$	Set of routes from origin $r \in \mathcal{R}$ to destination $s \in \mathcal{L}$
q_{rs}	Demand for *O-D* pair $r - s$
$\delta_{a,k}^{rs}$	Indicator variable equal to
	1 if a is on path k of *O-D* pair $r - s$, and 0 otherwise

Table 1 gives the mathematical notations for the very general concepts described above.
In the most general and closest to reality case, all nodes can be origin and destination, hence :

$$\mathcal{N} = \mathcal{R} = \mathcal{L}$$

2.2 Link Flow

Also known as the *traffic flow* on a link, the *link flow* is the amount of vehicles passing through a given point over a certain time period. Usually the unit of choice is veh/hr, or vehicles per hour.
The *link flow* measurement can be done at different positions of a link ; at either end of the link or at its middle position would be common examples. These multiple readings can have different values, as explained by [22]. For instance the reading at the start of a link would differ from the reading over the same period at the end of said link by the number of vehicles present on the link at the start and at the end of the time period.
It can however be proven that when vehicles flows are constant, link flows are quite logically not affected by this phenomenon.

2.3 Link Load

The second measurement type considered is the link load. It is the measure of the amount of vehicles present on a link at a given time.
This measure can be derived from link flow as the number of vehicles on a link is equal to the number of vehicles having entered it but having not left it.

2.4 Route Flow Assignment

Based on the *O-D* pairs and the demand for each of them, the route flow assignment proceeds by linking part of each demand, part of the flow emanating from a node and having a common destination, to a certain route, as explained by [31]. Obviously links can and will be shared by multiple flows ; whether these flows have the same *O-D* pair does not matter here.
If multiple flows share the same *O-D* pair, it is assumed that the route they take is different by at least one link as otherwise they should be regrouped. It is also assumed that no route flow is null.

A complete flow assignment gives the flow on each link by simply summing the flows of each route passing through a specific link.

2.5 Individual Assignment

Individual assignment differs from flow route assignment in the sense that each flow only consists of one vehicle here. The constraint of route of two flows having the same *O-D* pair being different is here relaxed.
Link flow can be just as easily computed by considering each vehicle as its own flow.

The particularity of this assignment is that a single vehicle takes a route only once (or once periodically) and so the flow generated by it is not continuous but rather time-dependent.

A key aspect of individual assignment is its capability to simulate much more complex motorists behavior, as explained by [29]. Indeed, when drivers are regrouped together and considered as a unique flow, it becomes impossible

to take into account personal choice of each user based on its previous experience in the network. This feedback is extremely important in modeling complex users behavior, as will be detailed more in the section concerning simulation based methods.

2.6 Static Transportation network

A static transportation system has constant demands on each origin-destination pair. This also means that the flows in such a system are not time dependent as they are always equal and that travel times also are constant. Table 2 gives the precise notations specific to static networks.

Table 2: Static Traffic Network Notations

x_a	Flow on arc a, x $= (\dots, x_a, \dots)$
t_a	Travel time on arc a, t $= (\dots, t_a, \dots)$
v_{rs}	Average travel time on routes with *O-D* pair $r - s$
f_k^{rs}	Traffic flow on route k of *O-D* pair $r-s$, $f^{rs} = (\dots, f_k^{rs}, \dots)$, f $= (\dots, f^{rs}, \dots)$
c_k^{rs}	Travel time on route k of *O-D* pair $r - s$

In such a network, the measurement of choice for the traffic is the link flow. The knowledge of link flow allows us to determine the travel time, and as such the cost, of having a given flow going through a particular link.
The phenomenon explained in Section 2.2 where measurements are different depending where they are taken on a link does not take place in static networks. This is due to the constant nature of the flows, meaning that the number of vehicles present on a link at any time is equal, and namely the number of vehicles at the start of the time period where the measure is performed is similar at the end of said period.

2.6.1 Static Traffic Assignment

The static traffic can be, and usually is, represented as a matrix of flows departing from one node and traveling to another, the *O-D* flows. The

assignment problem in such case has as a solution a set of routes associated to each (parts) of these flows.

2.7 Dynamic Transportation Network

The difference between dynamic and static traffic networks is that the former has time dependent origin-destination demands. This generalization over the static model is much more realistic. Indeed the traffic flows in real life are time dependent : for instance the number of vehicles circulating at a peak hour is vastly superior to any other time of the day.
It leads to time dependent travel times and vehicles flows on each route and arc. The notation remains the same except for a time relationship.

The increased realism of this type of network brings some new constraints, such as the choice of measurement of the traffic. The link flow that was used in the static case is here not sufficient : as shown by [22] the dynamic aspect of the network forces us to use the link load instead. Figure 1 shows the relationship between link flow and travel time in link belonging to a dynamic network. A highly congested road will have a very low link flow and a very long travel time which is indistinguishable based on link flow from a mostly free link where the flow is very low due to the lack of vehicles going through it while the travel time is short.

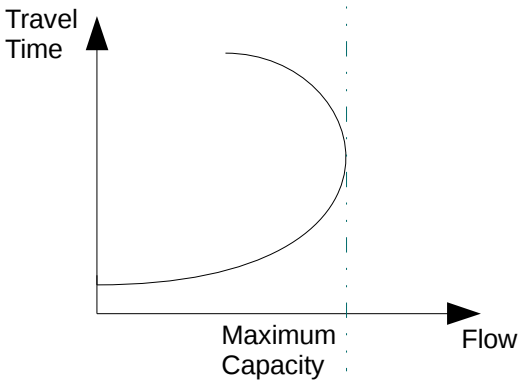
2.7.1 Dynamic Traffic Assignment

The dynamic aspect of the traffic imply that the simple representation of demands as a single flow per O-D pair is not sufficient. Introduced by [21], a model of the dynamic traffic is to have for each O-D pair a set of flows each corresponding to a certain time interval. In such a network, the traffic assignment is that of the time-dependent flows.
The static assignment is then a special case of the more general dynamic assignment, in which there is only one time period observed.

2.8 Goals & Evaluation

Detailed by [1], the traffic assignment problem can simply try to optimize many factors. This optimization can happen on the level of the user or on a more general level of the whole system or system manager.

Figure 1: Relationship between travel time and link flow.



Unless specified otherwise, the optimization aspect considered in the rest of this work will be the traveling time, personal for each user and global on the system-wide level.

2.8.1 For the User

This kind of optimization is the most natural. In real life drivers do have a tendency to take greedy choice with regards to their route choice.

Minimizing Travel Time The most common optimization aspect, both in real life and for the *TAP*, is the minimization of travel time for each user. Travel time encompasses both the length of the route taken and the congestion on each street passed.

Minimizing Travel Distance Closely linked to the travel time, the travel distance differs in the aspect that the shortest path in length often takes arterial streets which tends to be the place of most traffic jams, sometimes making the shortest path the one taking the longest time while a longer route can be a detour around a traffic jam and end up being much shorter

in terms of travel time.

Maximizing Route *Simplicity* A much less common aspect in *TAP* while remaining very close to reality, is the route simplicity. It is usually taken into account by multi-goal optimization as well as by agent-based simulation where the agent behavior is extremely detailed.
A route can be simple in many ways :

- The prior knowledge of route can be key for certain users. Not switching their daily routine can, for some, be more important than taking a shorter trip.
- Another classic aspect of route simplicity is the incline of drivers to take roads with fewer bifurcations and turns. This comes from the fact that both bifurcations and turns usually involve coming to a halt and some prefer keeping a constant speed over a greater but more variable average speed.
- The road type and number of lanes can also be a factor. Clearly a "bigger" road, like an avenue, is simpler to navigate than a more intricate one lane residential area street.

Maximizing Average Speed Linked to road simplicity, the ability to drive at a constant speed is sometimes highly regarded by users. To achieve a high average speed, the route taken usually has to be very "simple".

2.8.2 For the System (or System Manager)

These factors of optimization are much more complicated to achieve in reality, as the drivers tend to prioritize their personal cost rather than to cooperate to achieve better results for the entire system.
It is nonetheless very interesting when considering that optimizing for the whole system would mean a better overall use of the network as well as a better user satisfaction on average.

The main approach dealing with system-wide optimization is the simulation based one, and more specifically the study of influencing drivers route choice. This is explained in greater detailed in another section hereunder.

Minimizing Total Travel Time Very similar to the individual travel time optimization, it concerns the total sum of travel time across all vehicles and keeping it as low as possible. It usually involves having some users taking a longer road to allow others to travel faster.

Maximizing Average Link Load Having an average usage of the entire network constant over all links is interesting as it would mean that all available resources are used as efficiently as possible.

2.9 User Equilibrium

A user equilibrium in a traffic network is a state where none of the drivers would want to change their route. This happens when for none of the users a change in their own personal route planning would decrease its cost or in other words, no improvement over their cost can be achieved by their individual action. In this scenario, no user would change their path and the system would attain and remain in a state of equilibrium : the user equilibrium (*UE*)

There can be multiple *UE* on a single network, this comes from the fact that *UE* could be seen as local minimum, where the only possibility to leave such minimum would be through cooperation of multiple users.

Figure 2 shows a simplistic (and unrealistic) static network. In this network there only exists one flow of 6 vehicles with *O-D* pair A-D that has to be split between the three possible routes. A *UE* traffic assignment for this network is given in Table 3. It is clear that this is indeed a *UE* as if any driver were to change his route to any of his two other possibilities, he would increase his personnal travel time.

2.9.1 Static & Dynamic

Both static and dynamic traffic networks share the same definition of *UE*. However, they differ in terms of complexity to find such equilibrium. This comes from the fact that static equilibrium is found through static traffic assignment and dynamic assignment through dynamic assignment and that the mathematical models behind both are largely different, as detailed by [21].

2.9.2 Stochastic

Detailed by [35] and [31], the stochastic user equilibrium happens when the motorists are considered as having incomplete or imperfect informations about their environment and precisely about the travel times on each links. This comes from the fact that in practice drivers do not know the actual precise time it would take them to go through a street. Uncertain link costs lead to inaccurate shortest-path consideration for each individual
This aspect is often modeled using perceived travel times when users have to take a decision in their path choice. These perceived link costs are themselves represented by random variables. The stochasticity of the user equilibrium in such a scenario comes from these random variables.
This kind of *UE* is a generalization of the "normal" *UE* explained previously which is simply a case were all the random variables are absolutely accurate and have a null variance.

2.10 Global Optimum

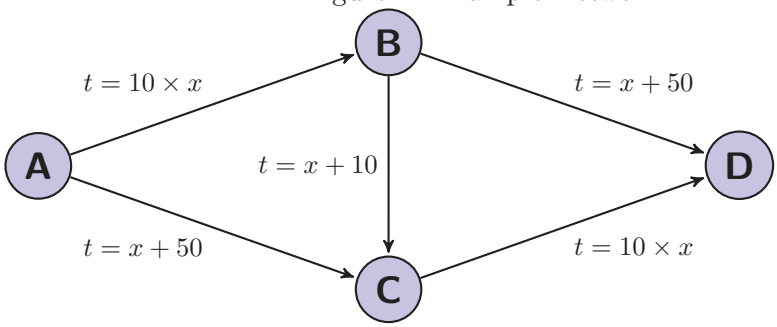
The global optimum of a system is the assignment of traffic on said system such that the total cost of all traffic movement is minimized. The cost function can change the optimum.
Similarly to the *UE*, the usual cost function is the travel time, and in this case, the total sum of travel times for all drivers.

The optimum of a network can differ fundamentally from a *UE* as often the global optimum requires some drivers to take routes giving them a longer travel time to allow others to reach a shorter travel time. This notion of "sacrifice" is in opposition with the *UE* of the drivers who have to take this sub-optimal[1] path.
The main concern of this master thesis being the use of external route guiding systems to influence the system towards such global optimum, this subject will be detailed in greater details in following sections.

The difference between *UE* and global optimum can be seen on the example network shown in Figure 2 already used in Section 2.9. A global optimum assignment for this static network is given in Table 3. We can clearly see

[1]The sub-optimality is with regards to their own travel time , not to the total travel time of the whole population of drivers.

Figure 2: Example Network



there the difference in global travel times.
This example also shows us the instability of the global optimum assignment. Indeed, the drivers taking route A-B-D could consider changing their route to A-B-C-D, as the perceived cost of both paths is the same.

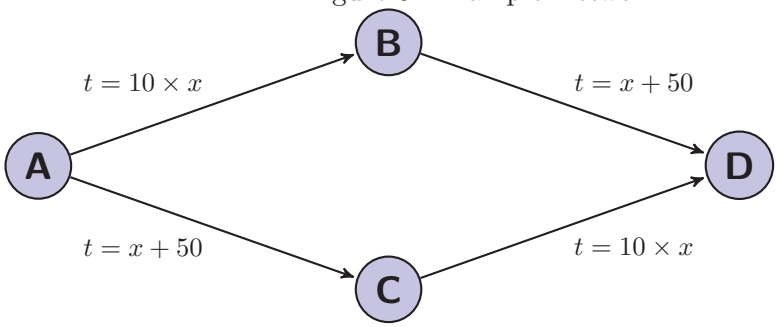
Table 3: Assignment & Travel Times on Example Network

	Traffic Assignment			Link Costs					Total Cost
	A-B-D	A-B-C-D	A-C-D	AB	AC	BC	BD	CD	
UE	2	2	2	40	52	12	52	40	$4 \times (52 + 40) + 2 \times (80 + 12) = 552$
Global Optimum	3	0	3	30	53	10	53	30	$6 \times (53 + 30) = 498$

2.10.1 Braess Paradox

The Braess paradox is defined as the phenomenon in which a link is added to a traffic network and this addition worsens the global travel time. The addition of a link can indeed shift the *UE* to a worse position due to the non-cooperation of the different users, as explained in [4].
The fact that a seemingly positive action, like adding a road to a traffic network, has a negative effect on the total travel time of users going through this network makes this a paradox.
Figure 3 represent a static, simplistic traffic network. In it exists a single flow of 6 drivers going from node A to node D. The symmetry of this network would suggest the *UE* (and global optimum) of traffic assignment for it is to split evenly the traffic between the two possible routes. This would yield two routes with the same cost of $53 + 30$, and as such the individual cost of all drivers would be 83 and the total travel time of the system 498.
The addition of a very performing link with cost $t = 10 + x$ between node B

Figure 3: Example Network



and C would perfectly showcase the Braess paradox. The resulting network was already used as an example previously, as it is Figure 2. We have already computed the *UE* assignment for this network, and its total cost of 552. The "improvement" of the network would then indeed shift the *UE* in a worse position in terms of global travel times.

2.11 The FiFo Constraint & Properties

The First in First out (*FiFo*) constraint in traffic network is the natural idea that a vehicle entering a link will not leave it sooner than another vehicle having entered the same link earlier.
Realistically of course, individual vehicles do not travel at the same speed and can pass one another. However these are only exceptions whereas on average vehicles entering a link will travel through it at the same average speed and leave it in the same order they entered. In general, a traffic assignment should then satisfy the constraint to represent "average" real life behavior. The generalization admitted here has an underlying benefit in the form of a relaxation of the constraint : "small" violations are allowed. Of course, the exact "smallness" is dependent on the instance of the problem.

Static systems, as shown by [10], do not have the concern of any added requirement based on the *FiFo* constraint. In static networks, travel time is constant for all vehicles of a link flow, which implicitly means that the *FiFo* requirement is met
However, this is not the case when it comes to dynamic networks.

2.11.1 Non-convexity of DTA

As explained by [11] the violation of the *FiFo* constraint comes from unrealistic travel times. These are only present in dynamic network representation where flows of vehicles are time dependent. In such dynamic networks, the traffic assignment problem is called dynamic traffic assignment (DTA)

Older works on the subject at hand have showed that taking into account the *FiFo* requirement adds non-convex constraints on the traffic assignment. These implicitly add a very high complexity to the optimization of the system. This non-convexity is independent of the optimum researched, local or global.
Fortunately more recent studies, including [11] have shown that it is possible to approach DTA without breaking the *FiFo* requirement while retaining convex constraints, resulting in much simpler optimization.

3 Analytical Methods

Many models exist both for *TAP* and *DTAP* and they all have in common a key aspect : the way to solve them. The solving mechanism is indeed most of the time following Algorithm 1, detailed by [35] for static network, or Algorithm 2 shown by [32].

For both algorithms, the initial solution can be a simple one; for instance [29] suggests a simple heuristic where the best routes for each *O-D* pair is computed while considering the network empty of any other traffic. Another possible initialization would be an all or nothing flow assignment also based on an empty network.
The new route discovery is also identical for both the static and dynamic networks.
The demands reassignment on the newly computed routes is identical for both. It differs however between flow assignment and individual assignment by the fact that, the former assigns parts of each flow on the new routes whereas the later assigns each traveler individually and independently on their preferred path.
The main difference between the two algorithms is, quite obviously, the dynamic aspect and its time-dependent demands. This leads to the fact that the demands loaded on the network are done so while taking into account the departure time. The congestion and delays computation is also different due to the traffic's dynamic nature as it needs to be done for each time period.

The termination happens when the flows make a *UE* (or *DUE*) which happens when the demands and routes do not change across consecutive iterations. This is guaranteed to happen when the new routes are generated based on a best-reply scheme ; that is the new routes only ever improve or are equal in quality than the previous one for each motorist. This naturally leads to a stagnation in a *(D)UE*.

The difference between models then comes from the set of constraints put in place such that the generation of new routes stays in the feasible set of solutions for the minimization problem ; as indeed the problem solved is a minimization of travel times in order to find a *UE*.

Some very general constraints can be found in most of the models :

Flow conservation Which states that the sum of all flows entering an arc should be equal to the flow exiting it. This should also take into consideration the flow generated by origin nodes and consumed by destination nodes.

Flow propagation Represents the idea that the sum of all flows exiting a link should be equal to the sum of flows entering other links connected to said node added to the sum of flows exiting the system there.

Nonnegativity Simply implies that all link flows are null or strictly positive.

Algorithm 1: Static assignment iterative algorithm

```
Initialization of a solution;
while not in UE do
    Load demands on network;
    Compute congestion and delays;
    Compute new routes based on delays;
    Distribute demands on new routes;
end
```

Algorithm 2: Dynamic assignment iterative algorithm

```
Initialization of a solution;
while not in UE do
    Load demands on network based on departure time;
    Compute congestion and delays;
    Compute new routes based on delays;
    Distribute demands on new routes;
end
```

3.1 Frank-Wolfe Algorithm

The Frank-Wolfe algorithm applied to *TAP* is the most basic case of the template described above. It iterates over the following four steps :

1. Compute through linear programming the best perturbation that can be applied to the currently assigned flows while validating a set of

constraints[2]. This results in a perturbed version of the flows as well as with the perturbation itself.

2. Test if the perturbed flows have worsened the travel times, in this case stop with the previous flows as a solution.
3. Find the best weights for the perturbation and apply it.
4. Test for convergence of the new flows with regards to the previous ones.

As explained by [31], this algorithm converges to the set of *UE* when applied to a static network. However, the most basic version of the algorithm explained here has been found to have a very slow convergence rate. It is the reason why many improvements and extensions have been designed for it. These usually involve changes to the calculations of the perturbation step size ; or use results of previous iterations to improve the current choice of perturbation.

3.2 Method of Successive Averages

The method of successive averages (*MSA*) uses Algorithm 1 already described previously. In this method, new route flows are generated through a weighted random direction perturbation over the previous flows. The weight of which is dependent on the average of all previous flow perturbations, as explained by [35].

3.3 Model Based on *FiFo* Violations

Presented by [23], there exists a model based on *FiFo* violations capable of solving the *TAP*. However these are not *FiFo* constraints at the level of a link in the network but on the whole flow of vehicles associated with each *O-D* pairs.
The principle of this *O-D* based *FiFo* is that no vehicle leaving a node should arrive sooner to its destination than another vehicle having left the same node earlier or concurrently and going to the same target node.

O-D FiFo violations, in static networks, can be quantified based on the average travel time per *O-D* pair. This comes from the fact that the constant

[2]Usually the flow propagation, conservation and nonnegativity constraints.

flow over time means all flows can be seen as starting at the same time; and as such all different routes linked to a *O-D* pair should have the same travel time. The formula for the violation J_k^{rs} on route k of O-D pair $r-s$ is the following :

$$J_k^{rs} = q_{rs} f_k^{rs} (c_k^{rs} - v_{rs})$$

The total violation can be seen as a sum of all violations dependent on the flows assignment. The change of flows between iterations should be done with respect to this equation such that all J_k^{rs} tend to zero.
When the system has no *O-D FiFo* violations, it is said to be in *O-D FiFo* equilibrium or partial user equilibrium *(PUE)*. The partiallity comes from the fact that clearly all *UE* do respect the *O-D FiFo* constraints, while the opposite is not true. Indeed, *O-D FiFo* constraints only take into account used path which in turn allow better paths to remain unused.
A *PUE* is a *UE* if none of the unused routes are shorter than the used one, for each *O-D* pairs.

It can be proven that *PUE* that are not *UE* are unstable due to the possible existence of unused shorter routes. This leads to the fact that a slight change of flows towards any of these better routes would make all flows shift towards a new equilibrium.
It can also be proven that on the other hand, *UE* are stable.

The same principle can be followed in dynamic networks where the *PUE* concept can be extended to its dynamic equivalent. The major difference being the obvious time dependency.

3.4 *DTAP*

We previously discussed the complexity increase that dynamic networks imply. Based on this, it is obvious that methods and models can be specifically made to suit this particular type of networks.
Of course, static models being equivalent to dynamic models where only one time period is observed, the methods exposed here can be applied to static networks as well.

3.4.1 Dynamic User-optimal route choice

Expressed by [14], the dynamic user-optimal condition represents the idea that, in *DUE*, any route whose associated flow is non-null imply the fact

that the actual travel cost is minimal over this route.
Very similar to the *O-D FiFo* constraints model, there exists a model using the variational inequality approach to find *DUE* in discrete time dynamic networks.

The predictive user equilibrium, defined by [36], takes its root in the model described above. It aims for used flows that are all faster than any unused flows would be, using predicted travel times during a given time period. Its goal is to try to converge to a *UE*, as usual.
The proposed way to solve this model is through the use of *MSA* as well as an algorithm very similar to that of Dijkstra to compute at each time period the shortest path from any origin node r to any associated destination s.

3.4.2 Multinomial Logit Approach

A model brought forth by [5] bases itself around the probability for users to take a certain route at a certain time. It uses probabilities derived from the satisfaction perceived by drivers, itself implied by the travel cost over the path.
These probabilities are computed over the current network flows and are used to generate a new flow assignment tending towards optimality.

This approach is applicable to dynamical network and has been proven to result in user optimal route choice.

3.4.3 Precise Link Flow Model

In the previously described models and methods for the dynamic traffic assignment, the main method of measurement was travel times based on the exiting flows of vehicles on each links. This is however not a realistic approach, as discussed by [12]. Indeed, a fully accurate representation of trip time and flows per link should take into account vehicle density, speed as well as their flow rate.
However, adding such precision directly would induce a set of non-linear or non-convex constraints which in turn would make for intractable calculations and computations.

For this reason, [12] introduced a model where travel times and their relationship with traffic flows are not used for the generation of constraints, but

for the creation of time-space links. These new links are then the ones used for flow assignment.
This can be seen as a split of each original network links in smaller time dependent links. The only unusual constraints added by this model are to ensure the consistency of flows traveling over the same physical link at different times. These force a later flow to slow down if a denser, and slower, flow is present ahead of it on the link. The concept of "ahead on the link" being available due to the space-time links.
This approach can implicitly generate traffic assignment that respect the *FiFo* constraint. It does however require the use of additional and more common constraints, such as the flow conservation, propagation and non-negativity, in order to create valid, or feasible, traffic flows.

4 Simulation Based Methods

TAP can be approached by way of simulated traffic. Both the individual behavior of each drivers or the more general behavior of flows can be the target of a simulation.

Simulation methods are used to solve *TAP* similarly to analytical methods ; that is the same algorithms described previously (1 and 2) are used. The difference being in the evaluation of network delays which is here done through simulation instead of being calculated.
The new paths generation based on delays can also, in the case of agent-based simulation, be generated through the preferences of each agent. These personalized route tend to be closer to how real drivers would adapt their paths and lead to faster convergence in *UE*.

4.1 Off-line & On-line Simulation

The first of two distinct categories of simulation uses for *TAP* are off-line simulations. These are the more classical approach where simulations are directly used to compute traffic condition such as delays and travel times. They are usually performed on fixed testing sets of *O-D* demands and yield more theoretical results. The off-line aspect allows testing of different settings, parameters and possibly techniques without external requirements such as real time computation times. It also allows for a more controlled environment where exceptional scenarios which may occur in reality can be avoided.

On-line simulations differ greatly from the former as they encompass the use of simulation to be able to predict future condition based on the current live system state. Usually used by traffic system management or control for both creation of accurate forecasts and users guidance.
The nature of this approach adds a time sensitivity constraint on computations : indeed the simulations should be finished earlier than the situation simulated happens otherwise the prediction is worthless. As explained by [18], this involves the fact that such simulations usually do not try to approximate (dynamic) user equilibrium in the previously discussed iterative manner but rather tend to approximate optimal traffic assignment on a query by query manner. This compromise lead to the fact that on-line

simulation tends to yield lesser quality results than off-line. It should however be noted that on-line simulations in practice are usually more robust to exceptional situations than their off-line counterpart ; solely based on the fact that such exceptional scenarios are not taken into account during the conception of off-line simulations.
It is clear that any on-line simulations could be used off-line. In practice it is often done to test different strategies and parameterizations or to certify the correct behavior of the system.

4.2 Agent Based Simulation

The recent increase in computation capabilities has made it possible to simulate a previously unimaginable amount of independent agents concurrently, as explained by [34]. This is extremely useful for *TAP* and specifically for the individual assignment aspect of the problem.
An agent based simulation of a traffic network where each motorist is a separated agent leads to more accurate representation of drivers behavior due to the more detailed decision scheme that can be used. [25] tells us that it also allows for a better understanding and analysis of users interactions as well as their heterogeneity. Indeed, it is clear that two different drivers will not have the same reaction or choice when faced with the same situation in real life.

[1] also explores the idea of cooperation between drivers where information service providers (*ISP*) are themselves simulated agents with their own goals. These goals are on the system-wide level and usually involve helping the motorists in their choice of route such that the overall travel cost is reduced. This allows for indirect agent cooperation through communications between motorists and *ISP* as will be detailed hereunder.

4.2.1 Complex Behavior Representation

The main attribute of agent-based simulation for *TAP* being the ability to detail individual driver's trips and to allow a feedback from previous experiences, it is natural that a large aspect of such simulations are dedicated to these microscopic aspects of the system. Namely, as explained by [29], the behavior, trip planning, preferences and interactions of motorists.

Within Day Re-planning Generally individual users have the choice of a route whenever they depart and are considered done when they arrive at their destination. However, agent-based simulations allows us to "follow" each driver throughout their daily travels ; carrying feedback of previous trip in their later trip planning. This concept is usually only extended to the length of a day, and called within-day re-planning.

Explained by [29], the first representation of this is the choice of each driver to take certain trip at certain time based on their acquired knowledge of the system ; for instance to decide to grocery shop when they believe no congestion will be present on their route.
The second, possibly even more crucial aspect, is the fact that being able to follow each driver individually allows for closer to reality traffic demands : take the grossly exaggerated case of a user having two trips to make, driving to work and driving back home.
A classic dynamical traffic demand would have a trip registered in the morning from the users residence to its working place, and another at the end of the day in the opposite direction, without any correlation between the two, as the concept of agent history is not present. If the travel time going to its job took longer than a workday, the user would be represented in the system twice at the same time, as the classic demand would not know of this inconsistency. However, the microscopic aspect of agent based simulation would allow to dynamically change the return trip departure so that it takes place after the first trip ended.
Obviously, taking care of individual agent history requires more computational power. Often the advantage of a more realistic traffic network and the possibility of user knowledge feedback outweighs the added resources requirement.

Agents Indirect Cooperation As introduced briefly previously, the best way to simulate motorists cooperation is through higher level agents such as information service providers.
In a simulation setup introduced by [1], users agents communicate with *ISP* agent and query them for a route to their destination. The *ISP* then computes a set of possible routes based on both the current state of the network, the preferences of the *ISP* and its knowledge of the users preferences.
The state of the network comprise both perceived travel times registered by

the *ISP* and predictions[3].
The *ISP* agents preferences should correspond to the global goal of the system, usually to optimize the overall travel times but any of the previously identified, in Section 2.8.2, goals would work.
The users' goals can be any combination of those identified in Section 2.8.1.
When received, the set of possible routes are analyzed by the user agent and if one is satisfactory, a message is sent to the *ISP* to acknowledge the route choice. Otherwise, the *ISP* is notified and produces a new set of routes after updating the known user preferences and giving them more importance than to its own. This is done in order to always find an acceptable route in this second set, which is once again sent to the user for approval. The satisfiability of the received paths is based on each driver's actual preferences.
The whole process should be done in reasonable time as to not delay the user. Each *ISP* should also be capable to process requests from many drivers at the same time.

This negotiation allows shifting large amount of vehicles towards more favorable paths with regards to the global system and as such form indirect cooperation between users through the use of an external aid with whom the cooperation is direct. At the same time, users have an input with regards to their route, and their preferences are not ignored.

An interesting aspect of this manipulation of motorists towards a global optimum was brought forth by [4] : the proportion of drivers receiving the advices of an *ISP* can have a massive impact on the overall improvement.

Coordination Through Games There exists in reality forms of coordination, as explained by [24], where motorists make choice while taking into account what they expect the choice of others to be. This aspect relates to game theory, and tends to be far from cooperation as users only adapt to improve their personal travel costs.

The corresponding model make each agent choose their route, simulates the network based on the agents choice and reward agents for the quality of their choice ; these steps can then be repeated. The quality is dependent on the resulting network conditions. Coordination appears due to the fact that

[3]Predictions could for instance be based on previous day(s) travel times as well as currently experienced network congestion.

each motorist decision system is entirely based on the reward system whom is itself based on the global travel costs. Indeed, the choices performed in any subsequent iteration will depend on the reward received for the user's current choice.
Theoretical results show that this model results in high user satisfaction, the quantity derived from the network condition experienced by the driver during his trip and his preferences.
More details can also be added to this model, such as a two-step decision process where agents take an initial decision before being confronted with a traffic forecast[4] which may influence them to revise their route choice.

4.2.2 External Information & Influence over Agent

[17] explains us the possibility of intelligent agents that are capable of taking decision based on a set of personal preferences and on the basis of their perception of the environment. In the case of *TAP*, the environment can be perceived through many external channels of information. The decision taken are based around the choice of route, and this at two distinct levels, detailed hereunder : trip planning and *en-route*. The personal preferences considered here only concern the willingness of an agent to change its daily route pattern when confronted with external information that would suggest a better path is available.

Types of external Informations Multiple different external informations types can influence a drivers choice, and as such could be modeled in an agent-based simulation, as explained by [1]. A (non-exhaustive) list of those that will be considered here is the following :

- The most basic external information is, quite obviously, the drivers sight. A motorist can see up ahead if the traffic is congested or not and can based on this simple information decide to take a detour or not.
- Forecast of traffic congestion usually happens on dedicated radio segments. They usually only alert drivers to the presence of out of the ordinary traffic jams and sometimes offer alternative routes.

[4]The forecast is in this case based on the first decision which is "communicated" to the forecast *ISP*.

- Information signs indicating the congestion of the road ahead, usually only found on highways or arterial streets, will inform drivers of the traffic delays expected.
- Route Guidance Systems usually tell users what the shortest path, in terms of length, is to their exact destination. In certain case, these will also take into account traffic congestion. The fact that the information dispensed here concerns the entire trip, and not just a single link, make route guidance systems much more useful to users.
- Maps can of course be used to determine the shortest path. On-line dynamic maps showing traffic jams also exist, if very rare. The later can be compared to normal route guidance system while the former is in our case irrelevant as it would not influence daily driving patterns.

Detailed by [17], the different types of information can be classified in four more general categories :

- Qualitative information is the most basic information that can be perceived ; it would simply inform a driver if a path is (more than usually) congested. The direct perception of the driver (his sight) and in some cases radio traffic forecast fall in this category.
- Quantitative information adds to qualitative information the actual travel time perceived by other drivers at this exact moment. Both radio forecast and indicative signs give quantitative information.
- Predictive information are an addition over quantitative information where travel times in the near future are predicted and communicated to the user. Such predictions can happen for radio forecast as well as informational signs.
- Prescriptive information consists of qualitative information to which is added a description of an alternate route. Route guidance systems give prescriptive information when taking the congestion of the system into account while computing the shortest path for the user. However they do not directly display the qualitative information : it is implicit in the sense that when an alternate route is suggested it is due to high congestion on the usual route. The models of agent cooperation through communication with *ISP* described in Section 4.2.1 falls into this category.

Effect on Trip Planning Trip planning is the a-priori choice of route the user will take. It can be influenced by most external information, though

the most influential are maps and route guidance systems when the trip is exceptional. On the other hand, usual trips tend to be planned based on traffic forecasts alone.

Effect *en Route* The influence of external information on drivers during their travel is the most interesting one to represent, as it could indirectly represent agents cooperation.
This effect adds the possibility for a user to change its route dynamically based on the external information it receives while driving ; usually to avoid traffic jams. The change of plans in simulation can be key to a dynamic system where interactions between agents are represented through the use of *ISP* agents.

[18] tells us that the *en route* information received can be received in three different ways :

- At discrete point in time, such as radio forecast.
- At discrete point in space, such as traffic information signs.
- Continuously available, such as the traffic condition seen by the driver.

Random Drivers Drivers refusing to accept advice they are given is an easily achievable concept with the use of simulation. Indeed, as explained by [4], the fact that some drivers will expose a random reaction to external information, and notably to prescriptive alternative routes, is easily simulable. These random reactions usually involve not following the advice which can simply be simulated by giving a certain probability that a route given to an agent will be ignored and that said agent will resume its normal driving pattern.
The probability can, and in general is, part of each agent individual preferences. In some cases, it can also be dynamically evolving according to the quality of previously experienced trip. In other words, if the advised paths followed prior resulted in terrible driving conditions, it is less likely that a driver will continue to listen to its route guidance system.

5 System Modelling & Design

The general design used, inspired by the ones exposed in [1], [9], [27] and [24], can be seen as 4 separate entities: the traffic network itself, the simulation, the request handler (or service provider) and finally the learner. The general overview of their interaction and functionalities can be seen on Figure 4.

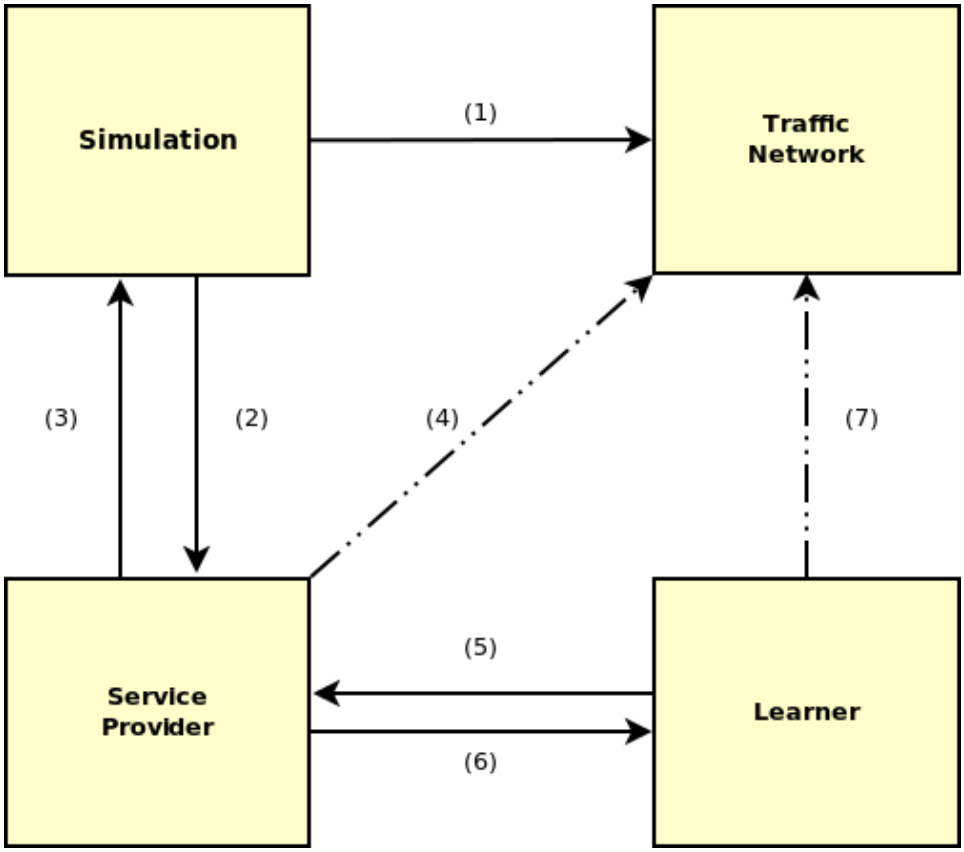


Figure 4: General design overview

1. The simulation models the movement of drivers moving on the traffic network.
2. Agents of the simulation (drivers), if directed, update the central authority (the provider) on their experienced travel times.
3. Service provider computes optimized routes for the directed agents of

the simulation.

4. The request handler uses the general structure of the traffic network in its routing mechanism.
5. Learner provides predicted travel times to the service provider.
6. Service provider transmits the experienced travel times it received from the simulation to the learner, such that the learner can update its predictive model.
7. Learner creates and maintains a predictive model of the traffic flows and travel times on the roads of the network.

5.1 Traffic Network Model

The traffic network itself needs to be represented. The primary static information about it should indeed be the first and most important features to be taken into account. Among these we should find the following:

- The number of intersections (or nodes) composing the network, each of these should be given a unique identification number i such that $i \in 1...n$, where n is the total number of such nodes.
- The positions of each road and intersections. This is, quite clearly, needed to later be able to simulate the movement and find routes on said network during the simulation. The information stored here is the list of each road exiting a given node (intersection) as well as the starting and ending point of each road ; as such we can denote each road R_{ij} going from node i to node j, two neighbouring nodes.
- The length of each road will be needed to compute the travel times. It shall be denoted L_{ij}
- The number of traffic lanes of each road. Even though we will not take into account the possibility to overtake another driver, the number of lanes is essential to model accurate travel times. Its notation is NL_{ij}, the number of lanes on road R_{ij}.
- The speed limit of each road which should be used to determine the minimum, or free flow, travel time. In the following, this value shall be identified as SL_{ij}, the speed limit of the road connecting intersection i and j.

Based on the above, other static characteristics of a network's arcs can be evaluated:

- Capacity of each road, based on the number of lanes and the length. It is noted C_{ij}, the capacity of R_{ij} and is calculated according to the following ([7]):

$$C_{ij} = C_{perc}\frac{L_{ij}}{3.5} \times NL_{ij} \tag{1}$$

  Where the value 3.5 comes from the average length of a car, and the constant C_{perc} is set to 0.65 to account for more realistic separation between each drivers.

- The minimum travel time[5] which gives the absolute minimum time needed to traverse a given road in the best condition - that is that said driver has the road to himself - while of course respecting the legal speed limitations. As such this value, noted $\widetilde{TT}_{ij}$, depends on both L_{ij} and SL_{ij} and is computed according to the following formula:

$$\widetilde{TT}_{ij} = TT_{perc}\frac{L_{ij}}{SL_{ij}} \tag{2}$$

  Where the constant TT_{perc} is set to 1.6 to account for the impossibility to drive at a constant speed coming from the need for acceleration, the time cost of intersections, traffic lights and crosswalks. As such, depending on the network studied, this value could be changed. The value chosen here is high as the networks that will be analysed in the later sections are part of high density cities.

- The preprocessed shortest path between each OD pair is computed following the classical *Dijkstra* ([26]) algorithm while using the $\widetilde{TT}$ as the metric giving the cost of each arc of our network. The information generated here is double: both the first node to be taken if following the shortest path, noted $\widetilde{NXT}_{ij}$, and the total cost of said path, $\widetilde{COST}_{ij}$, linking origin i to destination j, two nodes of the network.

All the above static attributes of the road network should then be used to be able to determine the dynamic ones:

- The current drivers count on a road, $DC_{ij}(t)$. This is directly linked to the current capacity usage.

[5] Sometimes referenced as the free flow travel time.

- The current absolute travel time, $TT_{ij}(t)$, is the theoretical value which follows the *BPR* (Bureau of Public Roads) formula ([37]):

$$TT_{ij}(DC) = \widetilde{TT}_{ij}(1 + \alpha \left(\frac{DC}{C_{ij}}\right)^{\beta}) \tag{3}$$

  α and β constants are set to respectively 0.15 and 4, as described in [19]. The parameter DC is the drivers count, in this case generally $DC_{ij}(t)$
  This value should not be directly used for simulation purposes to determine the actual travel time experienced as it is highly unrealistic to expect all motorists to drive the exact same *theoretical* way. Which leads us to…

- The actual experienced travel time, derived from the above which adds to $TT_{ij}(t)$ a random noise following a normal distribution of variance equal to five percent of the absolute travel time. This value has as an inferior bound $\widetilde{TT}_{ij}$ and a superior bound of $1000 \times \widetilde{TT}_{ij}$ to avoid unrealistic behavior where the actual travel time could tend to infinity on a highly used road segment due to the exponential nature of the *BPR* formula.

The last two attributes, the dynamic ones, should be inaccessible to the service provider part of the design due to the fact that in reality such value cannot be known with exactitude. However this is not the case for the static values.
In other words, only the simulation should use the dynamic informations generated by the traffic network throughout its simulating process while all other entities should only use the static attributes.

5.2 The simulation

The simulation itself, charged of handling the dynamic data described above and the movement of the drivers along the network, follows an event based design ([28]). In the used set-up, each motorist is represented by an agent of the simulation.
The events based aspect means that a queue of events is kept. For each of these events a time stamp is required to keep track of the advance of time. The events themselves are created and added to the queue whenever an agent enters or leaves a road.

The other major part of the simulation is that upon initialization, the agents are each attributed routes schedules representing the activities of the simulated person behind the agent ([6]).
These schedules are the traffic flow equivalent of this implementation. They are composed of a list of destinations and departure times pair, each corresponding to a trip in the network. As soon as such a route becomes available, that its departure time has been reached by the simulation time, its agents will start driving to the destination but only if the previous destination had already been reached. This allows for a realistic behaviour at this level.
These routes are generated upon initialization for the entire duration of the simulation period, counted in days. The creation of the schedule can be of any type described hereunder:

- Simplest routes generation: for each one of the day simulated, each agent receives a singular destination randomly chosen amongst all possible nodes as well as a randomly chosen departure time ranging across all 24 hours of the day. In this scenario, the initial location of each agent is also randomly picked amongst all nodes.
  Obviously the schedules created with this method are absolutely unrealistic and as such have not been used for any results' generation.

- Smarter routes generation: a more realistic way of generating trips for the agent population is to firstly assign to each agent a home and work node before picking a random work departure time that the agent should respect daily. Travels to work from home and the other way around are then assigned where the former has a random departure picked from a normal distribution centred on the work departure previously chosen, and the later is scheduled for another random time picked from another normal, this time centred 8 hours after the work departure.
  Added to this most basic set of rules, a probability for an intermediary stop on the way to and from work is added. These additional stops are scheduled such that the agent should get to their randomly picked[6] and stay there for an amount of time close to 45 minutes before taking off to his original destination.
  In this case, it is also considered possible for the agent to go out in the evening. For every simulation day, each agent is given a 20% probability to have another trip to a random destination, departing around 20h and coming back home around 2 hours later.

[6] In this case, the randomly chosen node is always different from the home and work ones of that agent.

The final different type of scheduling made here is when an agent is considered not to be working. A number of random stops are then generated for him throughout the day without ever going to work ; each of these stops is approximately 45 minutes long.

- Smarter weekly routes generation: adding the concept of week to the previous method is straightforward. Monday through Friday agents are supposed to follow the above while on weekends they only use *non-working* routes generation.
- Realistic routes generation: To allow this method to exist, more information needs to be known about the network. Precisely it is needed to know for each node if it is residential, commercial or industrial ; or in other words what nodes should be favoured as home nodes, as intermediary stops or work nodes respectively. The rest of the route generation will then simply follow the above with the exception that whenever a node has to be randomly picked, it has a much higher chance to be picked amongst those of the corresponding type.

Once the agents have their routes schedules, the simulation itself can begin. Initially the events queue is filled with the first departure time of each agent. These are *enter network* type events.
After this, until the queue is empty, time advances to the next event and asks the agent which it concerns to handle it. This will be detailed in the next section ; however it is necessary to state that agents will add new events to the queue afterwards. These can be of the following types:

- Enter road - whenever the agent is on the move and goes from one road to another.
- Enter network - whenever the agent is not moving and enters the first road of his route. Happens for the very first departure as well as whenever the agent stays put after reaching his destination.
- Leave network - whenever the agent has reached the destination of his current trip.

5.2.1 Agents

The basic core behaviour of the agents of the simulation is quite simple ; as explained above they need to handle events by creating the correct following event while also keeping any other component of the model updated on their movement ([13]).

Agents need then to update the roads to inform them of their presence ; that is update the drivers count on the road the agent is entering as well as the road he is leaving, if any.
The only other feature of the agents is comprised within the choice of the next road to take, which is dependent on the agent's type.

Free agents are the most basic and mindless possible when it comes down to their choice of route: they will always follow the shortest path preprocessed by *Dijkstra* algorithm as described in Section 5.1.

Directed agents represent the drivers that will follow the advice given by the service provider and will keep it updated on their position and by extension their experienced travel time on each road they encounter.
They thus simply choose the road given to them by the central authority as can be seen on Figure 5.

Informed agents are extending the concept of free agents in the sense that they are not directed by a central authority but they are smarter in their decision making. Indeed these are able to check the current state of the neighbouring roads before deciding to engage on one of them. This is done to represent the fact that motorists can in reality observe the state of the traffic amount of the roads they are about to enter.
The algorithm used for this informed decision is the reduced $A*$ one with a depth of one that is described in Section 5.3.2 using that extra observed information as the prediction and the normal free flow times in the role of a heuristic.

5.3 The Service Provider

5.3.1 Learner

The learning of travel times and, by extension, capacity usage of the different roads of a network is an important part of the service provider. Indeed, it will allow predictions to be made, themselves used to direct agents by the request handler; these predicted travel times will be referenced as PT_{ij} where i and j are neighbouring nodes.
Different types of learning can be distinguished, each different in terms of prediction quality and method, computation cost, learning mechanism and

stored information.
Furthermore, as it will be detailed below, some request handlers will need to determine whether a road has been *fully learned* meaning that the learner can safely state that its prediction should be accurate independently of the understandable instability experienced by the learning mechanisms until sufficient information has been given to them. These can be seen as reinforcement learning algorithm as detailed in [3].

Simplest learner Undoubtedly the first design that comes to mind capable of learning travel costs in a traffic network is to keep a single average for each road. This average evolves as each new travel time is recorded. These records are tallied up and their count is also kept, such that whenever a prediction is made the only required computation is a division, keeping the computational cost low.
The size of the stored data is similarly small: linear in the number of roads.
The criteria determining if a road is fully learned is in this scenario very direct and simple: if the number of recorded travel for a given road is over a threshold of 100 the road is considered fully learned.

Interval learner Logical evolution and improvement over the previous method, multiple averages are done here for many time intervals over a daily cycle.
The computations are thus equivalent with the added need to calculate which interval a new entry should be added to when it comes to the learning part. The major change is obviously in terms of needed storage newly depending on the number of intervals used.
Predictions are also more complicated. The predicted cost is indeed based not only on the interval corresponding to the time of request ; but also on its neighbours. In general a weighted sum[7] of the values of the exact interval and the neighbouring one is done. For extra precision, interval not considered fully learned are not taken into account during this sum ; instead they are replaced by the basic heuristic $\widetilde{TT}_{ij}$.
The number of intervals will determine the precision and quality of predictions while having a cost in storage, computation and learning speed: the smaller the interval the longer it will take for enough data to be recorded during it to achieve an effective learning.

[7]The weights are expected to be centred the current interval and decreasing the further away from it we get.

The *fully learned* value of a road at a given time of prediction is to be computed similarly to the prediction itself but instead of summing the averages stored for each neighbouring intervals, the weighted sum is done over the simpler learned factor of whether an interval has had more than 100 recorded events added to it. We can thus conclude that this learner will achieve *fully learned* status much slower than the simpler one described above.

***LV* learner** The *LV*, or last visited, learner only memorizes a fixed amount of the last travel times experienced by directed agents over each road. As such the stored data stays relatively small, in $O(n)$ the number of roads in the network.
This time dependent information gives place to an interesting way of making predictions ; weighted based on the time elapsed since the travel time has been recorded. When multiple records are stored different weights are given to them depending on the order they came in as well as the time elapsed.

Such a method is designed to allow precise predictions, especially during rush hour where many times should be recorded and the list of last visited times should be up to date. When little traffic is to be registered on the roads, the quality of predictions should suffer as the time between each recorded time will increase. However the predictions shouldn't be used either, as the traffic flow is very limited at this time by definition - as such the effective loss of quality with regards to actual usage of the predictions should be small.
The computational cost here is minimal ; only a small weighted sum has to be done on the stored values. Said stored value should be, in order to be optimal, kept in a fixed size *FiFo* container, due to the nature of the learning algorithm.

This learning method is different from the previous when it comes to verifying whether sufficient information are known to guarantee an accurate prediction. Indeed the method used here is simply to check how many of the stored entries are more recent than a 30 minutes threshold. The learned factor is then dependent of the proportion of these recent entries among the total being greater than 66%. This however does imply that, contrary to the other learners, the stability of the predictions will oscillate with this design as a road can go back and forth between being fully learned and not.

Error backwards propagation learner Inspired by the classical learning dynamics algorithm $Q-$learning [3], this learner keeps for each road a prediction value as well as a *fully learned* flag. The query for the prediction is thus the simplest of all: simply return the value stored if it is considered sufficiently learned, otherwise average said value with the absolute minimum travel time of the road.
The updating mechanism is where the entire complexity of this design resides, while remaining relatively simple overall giving us good performance. The general idea here is to compute the difference between the prediction that would be done for the road where new information is learned and this new information: the experienced travel time. This difference is actually the error of prediction. The following simple algorithm is then applied to update $pred_{ij}$ the prediction for R_{ij} :

Algorithm 3: Update prediction with backwards error propagation

parameters: R_{ij}, ett: experienced travel time;
error $= pred_{ij} - ett$;
$pred_{ij} = pred_{ij} - (error \times \ LF)$;
if $\frac{|error|}{ett}$ *under a given threshold* **then**
  mark R_{ij} as fully learned;
end

Where the threshold is a constant set to 0.05 ; meaning that if the prediction error was more than 5% of the actual travel time, the road is not considered learned ; and LF is the learning factor, a parameter of this kind of learner whose default value is 0.9.

5.3.2 Request Handler

The request handler[8] is the two way link between agents it directs and the learner entity. It will transfer the information concerning the state of traffic experienced by the agents to the learning mechanism and use the prediction made by the learners to create optimized paths for its directed agents.
To increase the quality of predictions, multiple learners can be used by a single request handler with the objective of having complimentary learning mechanisms in-between the different learners. The handler, whenever it needs a prediction, queries all its learners and compute a weighted sum of all the separate predictions.

[8]Sometimes referred to as service provider.

The basic flow of operation happening whenever an agent queries its handler asking what next road he should engage on as well as for the operations surrounding the update of the learner model (transiting through the handler) can be seen on Figure 5. On this diagram, the simplest case of one learner for one handler is represented, but the scheme could be easily extended for the multiplicity of learners.

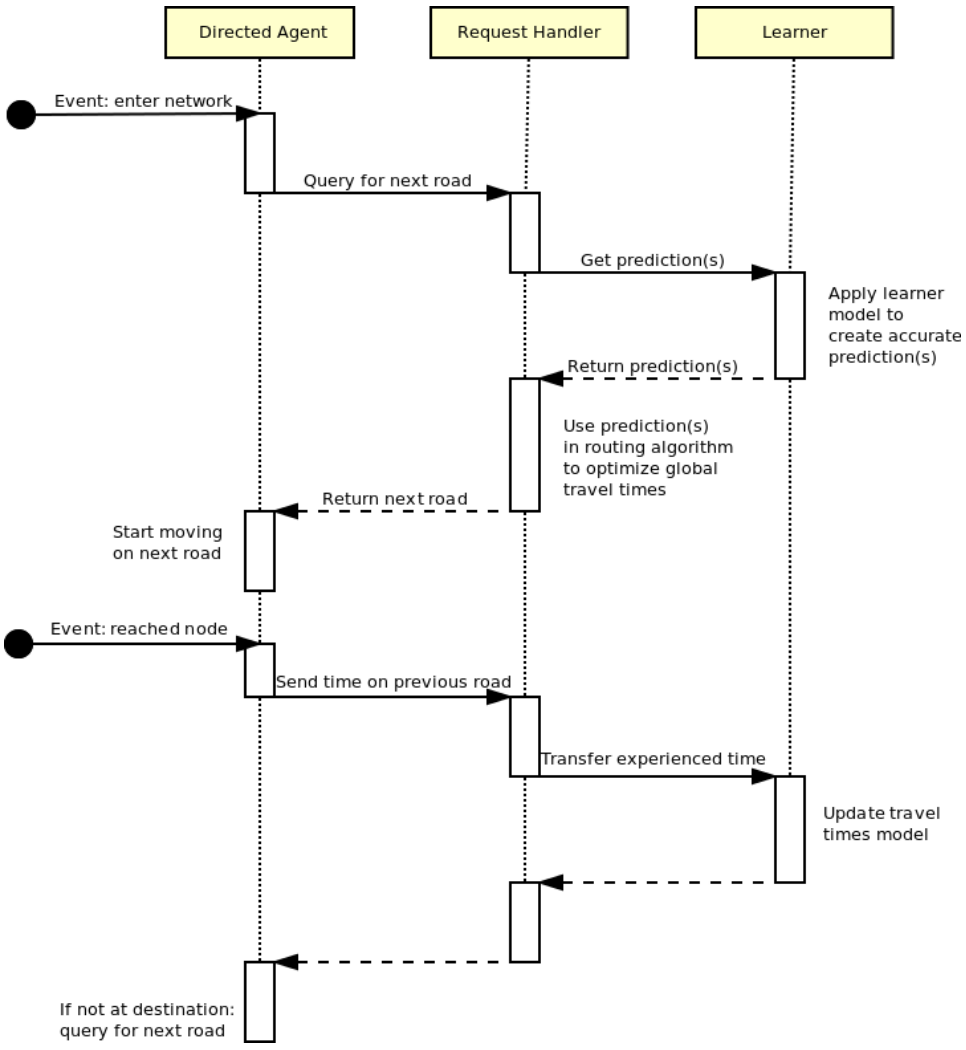


Figure 5: Sequence of a directed agent's request in the service provider

Marginal costs Another aspect of the handler is that it can use two types of predictions: general and marginal costs. The general one being that which one would expect in a greedy optimization scenario ; only the travel time of the agent making the query is accounted for in the prediction. On the other hand, the marginal costs, concept detailed by [20], tries to take into account the repercussions of assigning the road to the agent by computing not only the travel time that precise agent would experience but by adding to that the expected cost added to the other agents present on said road.
To achieve that, it is necessary to inverse the formula giving the travel time based on the drivers count, in order to obtain the formula for the drivers count based on the travel time, the predicted drivers count (PDC_{ij}):

$$PDC_{ij}(PT_{ij}) = C_{ij} \left(\frac{\frac{PT_{ij}}{\widetilde{TT}_{ij}} - 1}{\alpha} \right)^{-\beta} \tag{4}$$

It then becomes possible, with this predicted drivers count, to compute the marginal cost of one more agent on that road using again the original travel time formula:

$$MCost(PT_{ij}) = \left[PT_{ij} - TT_{ij}(PDC_{ij}(PT_{ij})) \right] \left(PDC_{ij}(PT_{ij}) - 1 \right) \tag{5}$$

This new metric is used to lessen the greed of a particular agent and try to push the traffic network further from the *UE* and closer to a global optimum by giving more importance on the (negative) influence of an agent on the others.

The details on how the handler optimizes the choice of next road for each of the different used algorithms can be seen below.

***A** handler** The first algorithm implemented is the famous $A*$ algorithm ([16, 39]) reputed for its capabilities as a shortest path heuristic method. Its pseudo-code can be found in Algorithm 4.
In our case, it is primarily used due to its very good solution quality to computation time required ratio. Secondarily, the fact that this algorithm reconstructs the entire path from the current agent position to its final destination at each query made makes it resemble a flexible trip planning method, in the sense that it is highly likely for the agent to be given the entire path computed at the first query performed from its origin, thus being trip planning instead of en route guidance ; unless the predictions are

massively updated throughout the trip.
In other word, the entire trip is (re)planned based on the current (predicted) traffic conditions after the driver reaches any intersections, giving the procedure more dynamism to its results with regards to the constantly changing state of the traffic flows than a simple trip planning could achieve.

Algorithm 4: *A** algorithm for request handler

```
parameters: origin, destination;
initial empty map entries;
initial empty queue openset where items are ordered by f_score;
add current node to entries and openset;
while openset not empty do
    current = openset.top();
    openset.pop();
    if current == destination then
        openset.flush();
    end
    mark entries[current] as visited;
    for neighbor in current.getNeighbors() do
        if neighbor not in entries or entries[neighbor] is not marked
        then
            g_score = entries[current].g_score +
            prediction(R_{current,neighbor});
            if neighbor not in entries or entries[neighbor].g_score >
            g_score then
                initial entry for neighbor in entries;
                entries[neighbor].g_score = g_score;
                entries[neighbor].f_score = g_score +
                \widetilde{TT}_{neighbor,destination};
                entries[neighbor].previous = current;
                add entry to openset;
            end
        end
    end
end
while enries[current].previous != origin do
    current = entries[current].previous;
end
return road to current;
```

Reduced *A with variable depth** The *A** algorithm can be reduced to add a more *en route* aspect to the road selection by stopping the search after a fixed depth is reached. The heuristic depth first search performed

by normal $A*$ is still performed but any node that is at *depth* number of links from the origin is added to a list of end nodes ; the algorithm then chooses the best end point when all possibilities have been done following the same preference rule as in normal $A*$. These modifications are shown in the pseudo-code of Algorithm 5.
The quality of directions given is obviously worst when compared to normal $A*$, but the computational requirements are also smaller.

Exploring handler A basic modification of any of the previously detailed request handlers is to add an increased exploration oriented behaviour. This is done by simply adding a probability that at any time the handler is queried by a directed agent, it has a fixed probability[9] t instead of following its core algorithm based on predictions from the learner, it will randomly return one of the adjacent road as the result while favourings the non *fully explored* ones[10].
The role of this change is to push the handler behaviour away from exploitation of the learned information, towards more exploration. This is inspired by $\epsilon - greedy$([33, 38]) exploration algorithm.

[9] In general, the value used for this probability is 10%

[10] Basic constrains are also applied to make sure it is impossible for the agent to cycle ; in general only roads that will bring the agent closer to its destination are thus considered as valid.

Algorithm 5: Reduced $A*$ algorithm for request handler

```
parameters: depth, origin, destination;
initial empty map entries;
initial empty queue openset where items are ordered by f_score;
initial empty queue endset where items are ordered by f_score;
add origin to entries and openset;
entries[origin].depth = 0;
while openset not empty do
    current = openset.top();
    openset.pop();
    if current == destination or current.depth == depth then
        add entries[current] to endset;
    end
    mark entries[current] as visited;
    for neighbor in current.getNeighbors() do
        if neighbor not in entries or entries[neighbor] is not marked
        then
            g_score = entries[current].g_score +
            prediction(R_{current,neighbor});
            if neighbor not in entries or entries[neighbor].g_score >
            g_score then
                initial entry for neighbor in entries;
                entries[neighbor].g_score = g_score;
                entries[neighbor].f_score = g_score +
                TT~_{neighbor,destination};
                entries[neighbor].depth = entries[current].depth + 1;
                add entry to openset;
            end
        end
    end
end
current = endset.top();
while enries[current].previous != origin do
    current = entries[current].previous;
end
return road to current;
```

5.4 Implementation Specifications

The model discussed above has been implemented in *C++*. An overview of the class hierarchy used can be found below, after which the details of the different file format used are given.

5.4.1 Class hierarchy

The implementation if divided into five packages:

Global Containing the main file as well as a class regrouping general utilities (regarding random number generation), a class *weights* used whenever a weighted sum is performed (to store the weights) and a header containing all globally defined constants.

TrafficNetwork Containing the class representing the actual traffic network, which are self-explanatory: *Node*, *Road* and *TrafficNetwork*. The last one keeping track of the two other as well as being responsible for the preprocessing of the absolute shortest paths and the reading/writing of the input/output network files. In this package we can also find a class named *TwinRoad*, a child of *Road* corresponding to the concept of two roads going either direction and sharing a single lane.

The final class here is the *RoadMonitor*, an implementation of the observer pattern which will record information about changes in drivers count over a given road.

Simulation Containing all element linked to the general simulation, the classes:

NetworkSim Handling the initialization of the agents as well as the ordering of the events (of the simulation).

Event An event of the simulation concerning one agent and having a time at which it will happen.

Agent The general concept of agent defining the normal behaviour of all agent type with regards to the handling of events and the creation of those.

FreeAgent*, *InformedAgent* and *DirectedAgent The child of class *Agent* defining the exact choice of route through the network.

ServiceProvider Containing the hierarchy of request handlers, with *RequestHandler* at the top. The child classes *AStarHandler* and *ReducedAStarHandler* are self-explanatory. The last child, *ExploringHandler* adds the exploration behaviour to another handler by being a wrapper around another included handler.

Learner Containing the hierarchy of learners, with *NetworkLearner* at the top. Its child classes *SimplifiedLearner*, *MediumLearner*[11], *EBackpropLearner* and *LVLearner* are all simply the implementation of the corresponding learning mechanism explained previously in this chapter.

A class called *SimpleRoadInfo* is also present here. It is used to store the changing average travel times by both the simplified and intervals learners.

5.4.2 File format

The files representing the traffic networks, used as input of the simulation, are *JSON* object files. The exact format of those file is as following:

metric A flag giving the metric used for both distances and speed. A value of 1 indicates that distances are expressed in kilometres and speeds in kilometres per hour.

nodes A list of node objects ; each composed of:

- **x & y** The coordinates of the node (only used in the visualization).
- **id** The integer, unique, identification of the node.
- **type** The flag giving the exact type of the node. Its value can be: 0 if undefined, 1 if residential, 2 if work place and 3 if commercial.

roads A list of road segment objects ; each composed of:

- **startId & endId** The unique identification of respectively the starting and ending node of the road segment.
- **name** The name the road, in the form of a string of characters (only used in the visualization).
- **length** The length of the road segment, expressed in the metric defined above.

[11] It is the intervals based learner.

speedLimit The speed limit on the road segment, expressed in the metric defined above.

nbBands The number of lanes of the road. A value of 0 means that a single lane is shared between two road segments going in opposite directions.

The output of the simulation is composed of two files[12].

The first one gives raw daily statistics for each agents. It is a plain text file where the first line gives the random number generator seed used by the simulation, the number of simulated days followed by the number of agents of each of the three types, in this order: free, informed and directed. All subsequent lines follow the same format: id of the agent, day of simulation, experienced travel time (in seconds) and travelled distance (in meters).

The second file is a complete log of all simulated activities on the network in the form of a *JSON* file, extending the input one. Besides all the fields already present in the input, the following are added:

time_index The index of the timestamps in the data field of each road's added information.

driversCount_index The index of the drivers' counts in the data field of each road's added information.

timePrecision The precision used for the timestamps. As all times respect the metric of the input, they are expressed in hours. However in order to reduce the size of this output file, they all are multiplied by the precision factor. The value are then rounded to the nearest integer ; which given the default precision corresponds to about a tenth of a second.

roadsInfos A list of extra information, one entry per road each composed of:

startId & endId The unique identification of respectively the starting and ending node of the road segment.

capacity The computed capacity of the road segment.

data A list of every time an agent as left or entered this road segment. Each entry is composed of a timestamp and the updated count

[12]When a output prefix for these two files is specified, otherwise they are not generated.

of drivers on the road[13].

To simplify access to the input and output *JSON* files, a library called *jansson*[14] was used. It is the only external dependency of this implementation.

A traffic network creation, edition and visualization tool was also created. It takes the form of a *Javascript* applet and uses *Processing.js*[15] library as base.

The source code for all the above is available, as a *Git* repository, at https://github.com/qvajda/TAP-Sim

[13]The exact index of these two values in each of the sub-list is defined by the two fields detailed just above.

[14]See http://www.digip.org/jansson

[15]See http://processingjs.org/

6 Results Analysis

All the results were generated by the implementation of traffic network simulation described previously. The only used routes generation method is however the most realistic one, using different node types, in order to best exploit the real networks used as input.
To allow the system to reach stability, the simulations are all of 7 (simulated) days.
To minimize the effect of extreme scenarios caused by the random aspect of the simulation, all results are generated for twelve consecutive *sample* runs before being averaged to give the real results. For instance, the average travel time of all agents is computed separately for each of the samples before being summed and divided to obtain the desired statistic.

6.1 Input network

The following results are based on a real life traffic network exported from *Open Street Map*[16] and transformed to the file format of the implementation described above. The *OSM* database not containing the information concerning the node type (residential, commercial or workplace) that information was later added by hand while trying to respect reality and the general fact that the outskirts of a city are more residential while the city center is where offices are located while commercial zones are situated in between the two.

The network used in the statistical results generation is one of Brussels' district: Ixelles. This graph contains 279 nodes and 657 individual links ; the later being representative of road segments between two intersections are thus more numerous than the streets present in the same area.

6.2 A first complete example

Before comparing algorithms, learning techniques ...we will start with a simple example on the Ixelles network (Figure 6). The simulation parameters are the following:

- Simplified learner with fully learned threshold value of 100.

[16]The file exported from *OSM* is filtered and transformed to the correct input file format by a Python script, also available on the Github repository.

Figure 6: The Ixelles network in *OSM* and its graph form

- $A*$ request handler without exploration behaviour, using non-marginal travel costs.
- Agent population of 20.000 out of which 90% are free agents (18.000) and 10% are directed agents (2.000)

This first simulation and its results shall be referred to as Simulation 1.

On Figure 7 we can see the resulting travel times experienced by both free and directed agents on each day of the simulated week. On the graph, the coloured boxes correspond to the 25% to 75% quartile while the black bar is the median and the red dots are the respective averages. The whiskers represent the furthest outlier from the box that is still within 1.5 times the interquartile range ; the actual outlying points, situated past the whiskers

are not draw for clarity and readability of the graphs.

Combined with Table 4[17] giving the raw numerical values of both means and standard deviations, a first observation can be made about the overall stability of the results over the week. The experienced travel times of each agent over the week does seem to stay roughly the same as the different averages are all within 300 seconds from one another ; without any distinct continuous increase or decrease with time. Figure 8 tells us that the distances registered by the motorists are even more stable as they only ever differ by a dozen meters from each other.
This difference of stability is related to the amount of congestion happening, itself dependent on two important factors:

First of all the random aspect of routes generation including the departure times of each agent and their potential intermediary destination (see Section 5.2). These will affect the amount of congestion in that whenever many agents depart at almost the same time and/or need to use a given street at the same time. This will, in the case of free agents, be highly dependable on the initial route generation. Directed agents will also contribute to this effect, but to a lesser extent as the service provider should advise them to avoid the already congested zones.

The second factor comes from the directed agents and their heuristic choice of itinerary. Indeed the $A*$ request handler will answer queries based on the learned information and the predictions of the learner. When a road is known to be highly congested, the service provider will try to direct its agents towards less optimal[18] paths which should help alleviate the bulk of the traffic jam, while giving the directed agents an even longer travel time than usual. The inherent uncertainty of the learning mechanism will also have an impact here. The predictions being by nature not one hundred percent accurate some agents will be directed into the congested roads while other will take longer paths to avoid non-existent congestion. Which then leads to the higher variance in the directed agents time and distance costs but also influences the travel times of free agents in the case of drivers directed towards the traffic jams.

[17] This table and all others present in this section show travel times rounded to the nearest second and distance rounded to the nearest meter.

[18] Less optimal in terms of absolute travel times.

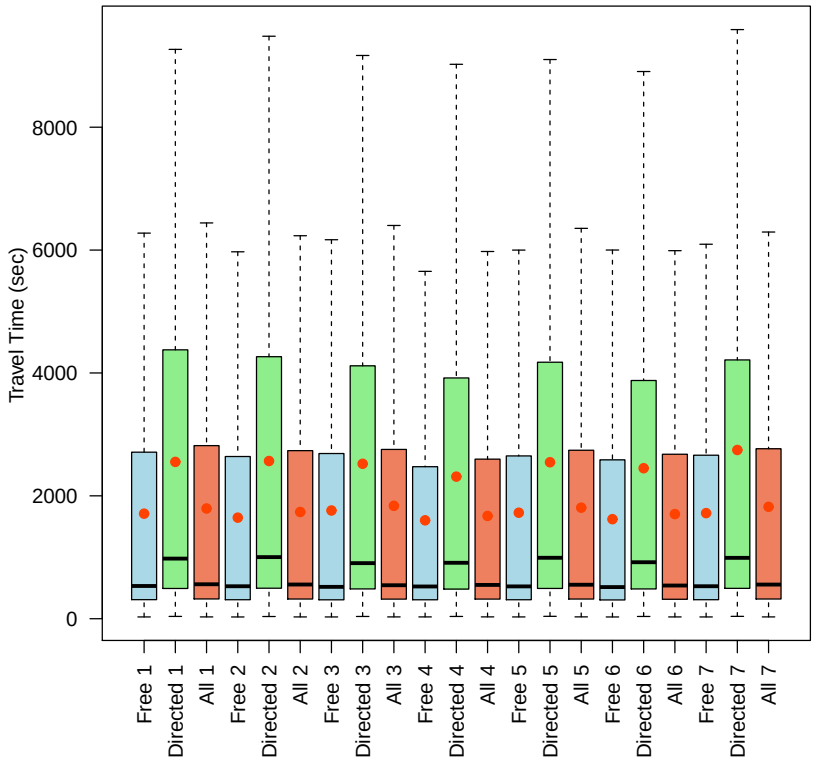


Figure 7: Travel times per agents type and per day of Simulation 1

After this first observation on the stability of the system across the span of an entire week, we can do a similar analysis of the results of a single day. The first evident feature is that free agents drive around 2.2km less and spend 700 to 800 seconds less in their car per day. Relatively this means directed agents drive 60% longer distances while only taking 45% more time. This can be seen as both positive and negative. On the one hand directed agents seem able to avoid traffic jams by taking alternative paths while on the other hand these alternative paths are significantly more longer.

The high deviation of travel times for all type of agents on the scope of a single day can be partially attributed to the congestion of the system as

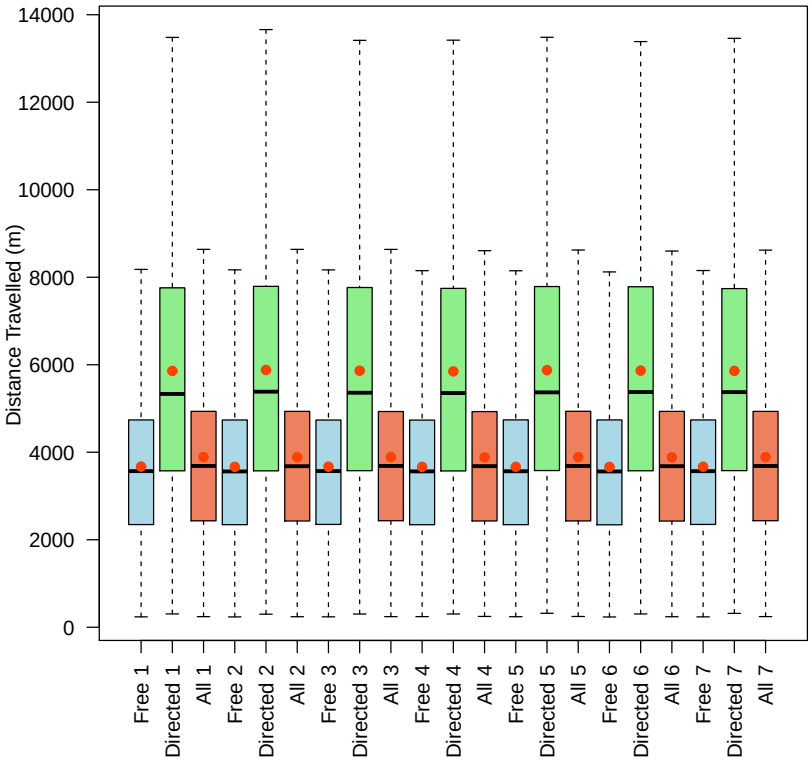


Figure 8: Distances per agents type and per day of Simulation 1

discussed above, but also to due to the method of generation of routes. Indeed, the nodes for both home and work place being picked at random will cause great amount of variance in travel distances and times. However, the distances follow a more normal distribution, with an average and median much closer to one another which would again be a proof of the congestion level of the network.

To further observe the congestion, it is necessary to leave the agent point of view to look at the roads themselves. Figures 9, 10 and 11 show the count of roads whose capacity use exceeds a given percentage over time, with the first one showing that information for the entire week and the two others being closer looks at, respectively, the morning and afternoon rush hour of

	Day 1			Day 2			Day 3		
	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All
Avg. TT	1710	2552	1794	1645	2567	1737	1761	2522	1837
TT Dev.	2141	2843	2235	2075	2803	2177	2552	3164	2629
Avg. Dist.	3672	5858	3891	3667	5881	3888	3670	5864	3890
Dist. Dev.	1861	3231	2142	1857	3243	2144	1856	3223	2138
	Day 4			Day 5			Day 6		
	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All
Avg. TT	1602	2313	1673	1725	2547	1807	1620	2449	1703
TT Dev.	2033	2559	2102	2353	2808	2415	2094	2790	2188
Avg. Dist.	3664	5849	3883	3667	5876	3888	3664	5866	3884
Dist. Dev.	1852	3201	2131	1852	3214	2135	1854	3219	2137
	Day 7			All					
	Free	Dir.	All	Free	Dir.	All			
Avg. TT	1719	2746	1821	1683	2528	1768			
TT Dev.	2266	3178	2393	2216	2877	2306			
Avg. Dist.	3670	5860	3889	3668	5865	3888			
Dist. Dev.	1857	3200	2135	1856	3219	2137			

Table 4: Mean and deviation of travel times and distances in Simulation 1

the first day of simulation. We can see there that while a good portion of all roads are lightly encumbered[19] only less than half of those use more than 25% of their capacity and only a dozen roads have more than 75% capacity used concurrently. However, the few roads that can be said to he heavily congested, those using more than their total capacity, do so for an elongated period. From this we can say that traffic jams do occur during the busiest times of each day and that they take a non negligible amount of time to dissipate. This is the cause for the above mentioned higher variance of experienced travel times than distances, as agent stuck on these roads remain caught in traffic for long periods of time.

The lower peak of the afternoon rush hour compared to the morning and the following second peak of traffic around 8pm every day is due to the initial routes' generation which gives agent a higher percentage of having a commercial stop on their way back home than on their way to work and to the possibility for agents to go out in the evenings. The former being

[19]More than 10% capacity used.

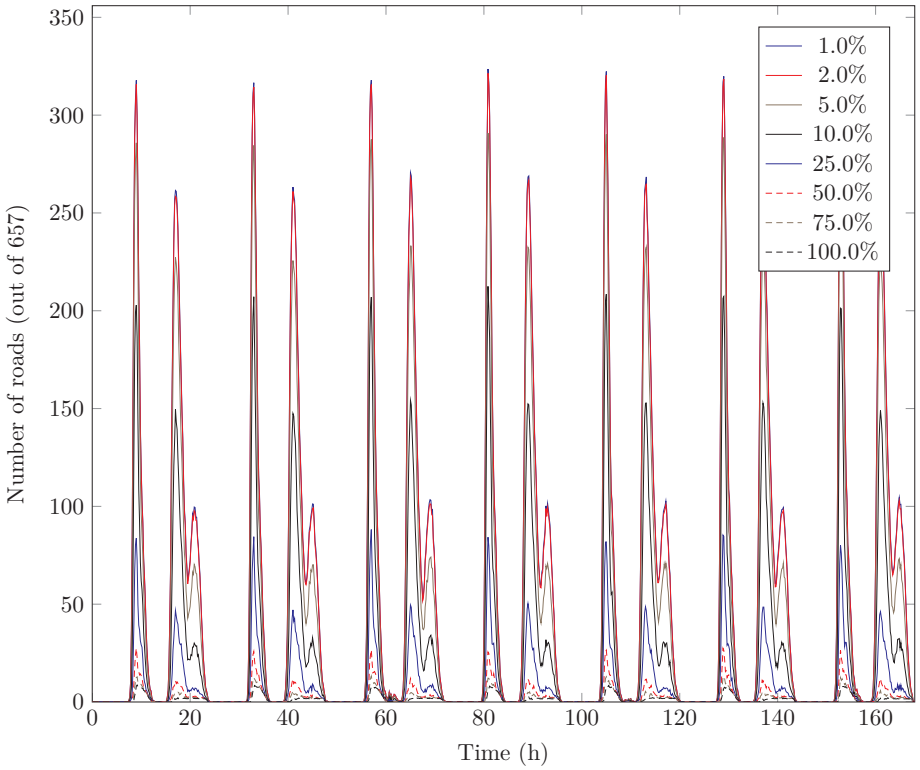


Figure 9: Road capacity over time of Simulation 1

responsible for the lower peak while the combination of both forms the evening traffic peak when motorists going out encounter those coming home after their commercial stop.

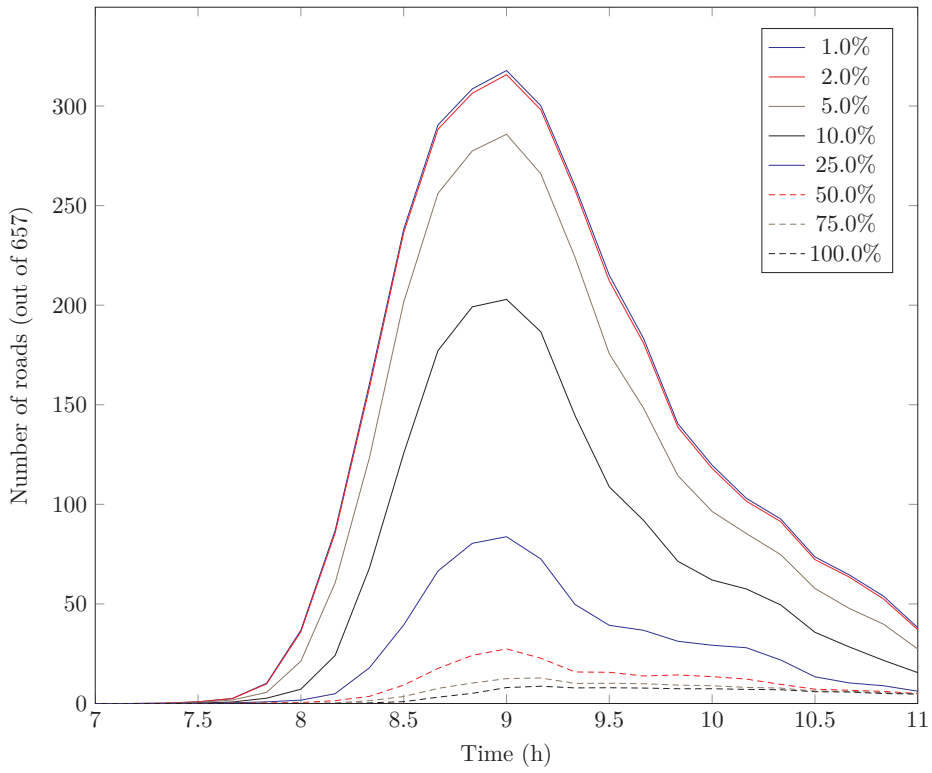


Figure 10: Road capacity over time of Simulation 1 on day 1 during morning rush hour

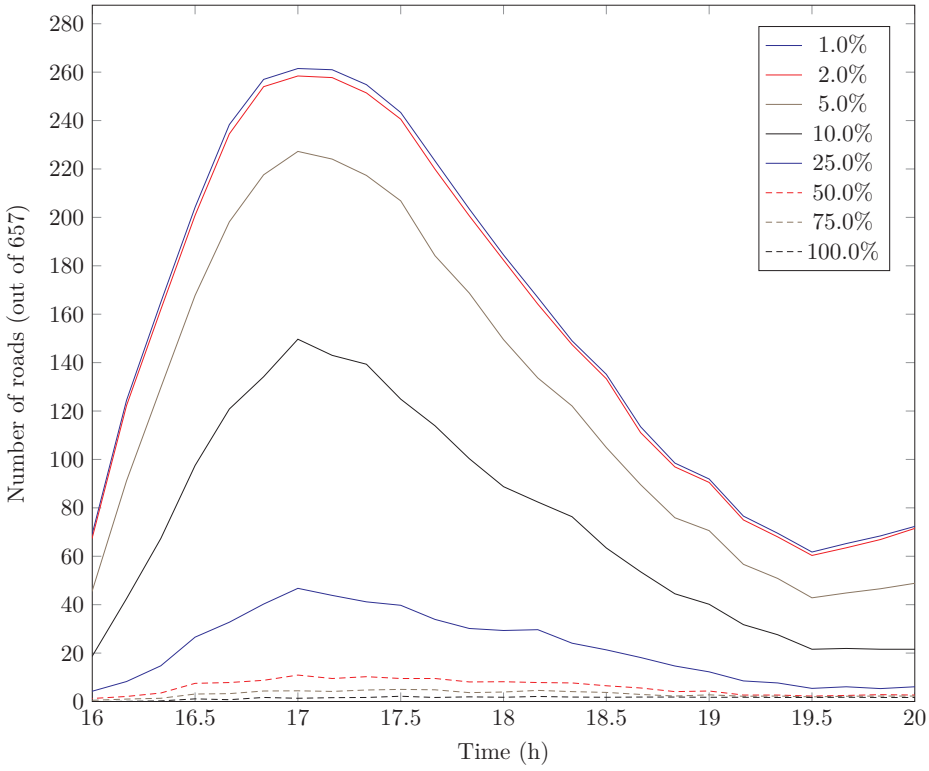


Figure 11: Road capacity over time of Simulation 1 on day 1 during afternoon rush hour

6.3 Without directed agents

Simulation 1 and its results exposed above should be put into perspective with Simulation 2, a simulation on the same network but where all 20.000 agents are free. Similar figures as for the previous simulation are present: Figure 12, 13 and Table 5 show respectively the travel times, distances and raw statistics of Simulation 2 while Figures 14, 15 and 16 represent the roads' capacity usage.

First of all we shall look at the similarities between Simulation 1 and 2. The distance covered by the (free) agents in this second simulation and the first one is unsurprisingly the same.
The stability of travel times over the week of simulation also has the same variance as the free agents had in Simulation 1 for the same reasons as exposed in the previous section.
The roads' capacity usage are also almost identical during rush hours. Only small differences in the number of lightly encumbered streets can be observed.

Secondly comes the differences between the two simulations with the most important one: the global average travel time and that of free agents only. In the second simulation the increase in congestion leads to an average 320 seconds (20%) longer travel time for free agents and an overall average 240 seconds (14.5%) increase. This comes from the fact that the 2000 directed agents that were present in Simulation 1 are in this simulation part of the free agents and as such use the absolute shortest paths which leads to more congestion than was present beforehand.

	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	All
	Free	Free	Free	Free	Free	Free	Free	Free
Avg. TT	2369	1847	2037	2270	1790	2034	1827	2025
TT Dev.	4315	2966	3354	4098	2808	3587	2950	3433
Avg. Dist.	3640	3698	3670	3657	3683	3667	3662	3668
Dist. Dev.	1831	1889	1852	1847	1872	1850	1852	1856

Table 5: Mean and deviation of travel times and distances of Simulation 2

This observation is extremely positive in that it infers that the addition of a partial central authority has managed its expected goal: to decrease the total travel time (indicated here by a decrease of the global average travel

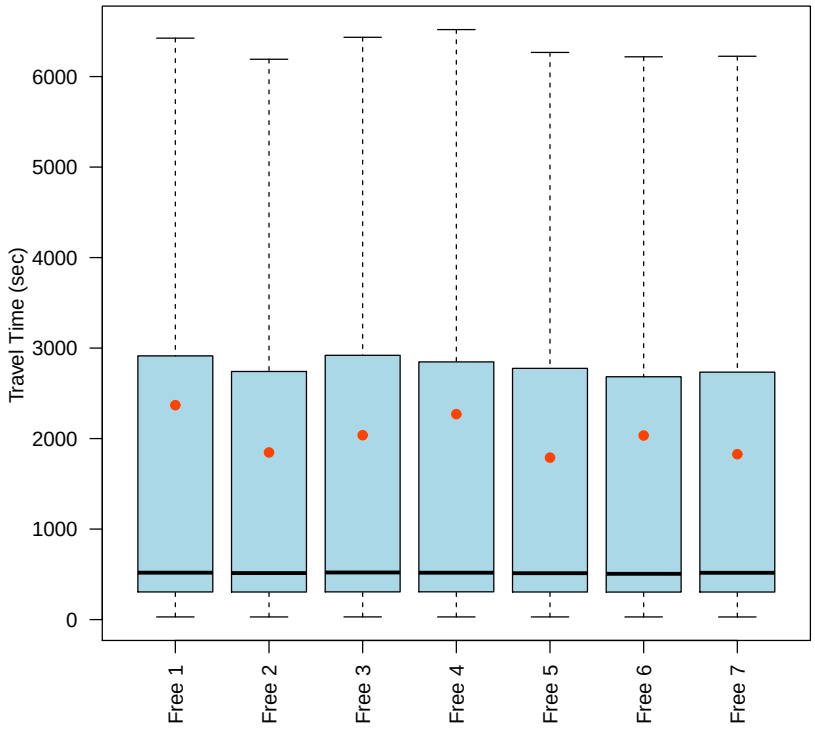


Figure 12: Travel times per day of Simulation 2

time). Indeed with only 10% of the population directed, it is possible to heavily reduce the travel time of the remaining 90% while keeping the cost of the directed drivers low enough as to also diminish the total cost.
The travel time experienced by directed agents in Simulation 1 is however greater than the one they would have if they were free agents by 25%. This also means that no reasonable drivers would decide to be directed instead of free, as it would increase his personal travel costs, both in terms of time and distances.

To come back to the roads' capacity usage being very similar between Simulation 1 and 2 despite a much longer average travel time, it comes from the fact that the few roads that are used over their total capacity are so even more in Simulation 2 ; the added free agents and those already present

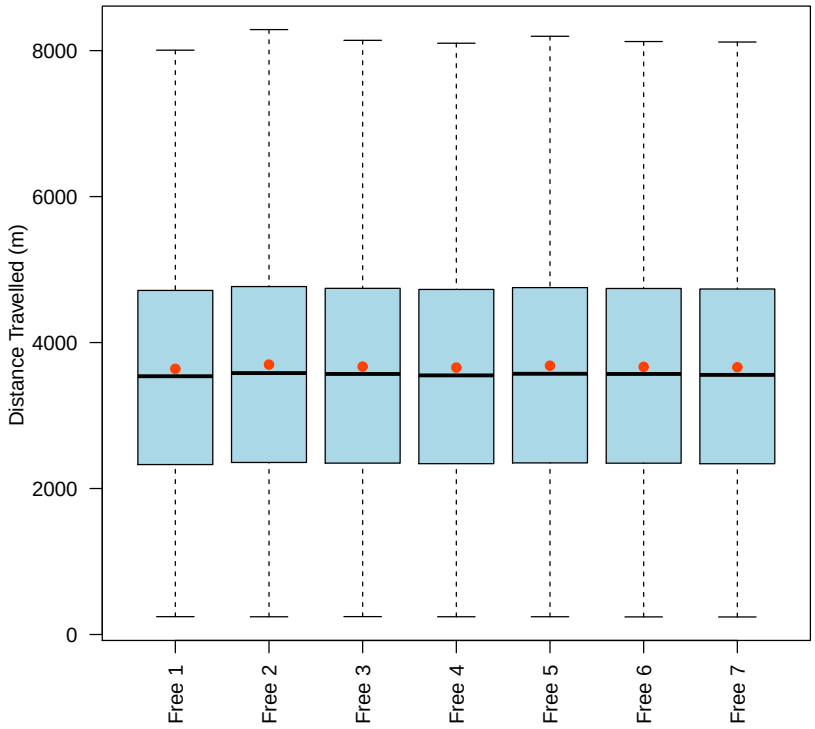


Figure 13: Distances per day of Simulation 2

before thus all suffer from worsened congestion.
The possibility for such capacity overuse comes from the lack of upper bound to the number of cars on a single road at a given time. This of course is unrealistic at a microscopic level as there should be a hard limit on the amount of car that can be on the same road at the same time. However on a macroscopic level the experienced travel time of the agents remains correct. All the traffic present on a single road segment should in reality be amassing on the previous roads due to a queuing effect of drivers trying to enter the over congested street. A more microscopic oriented implementation would have simulated this exact queuing behaviour ; however the difference in experienced travel times between such a microscopic simulation and the one

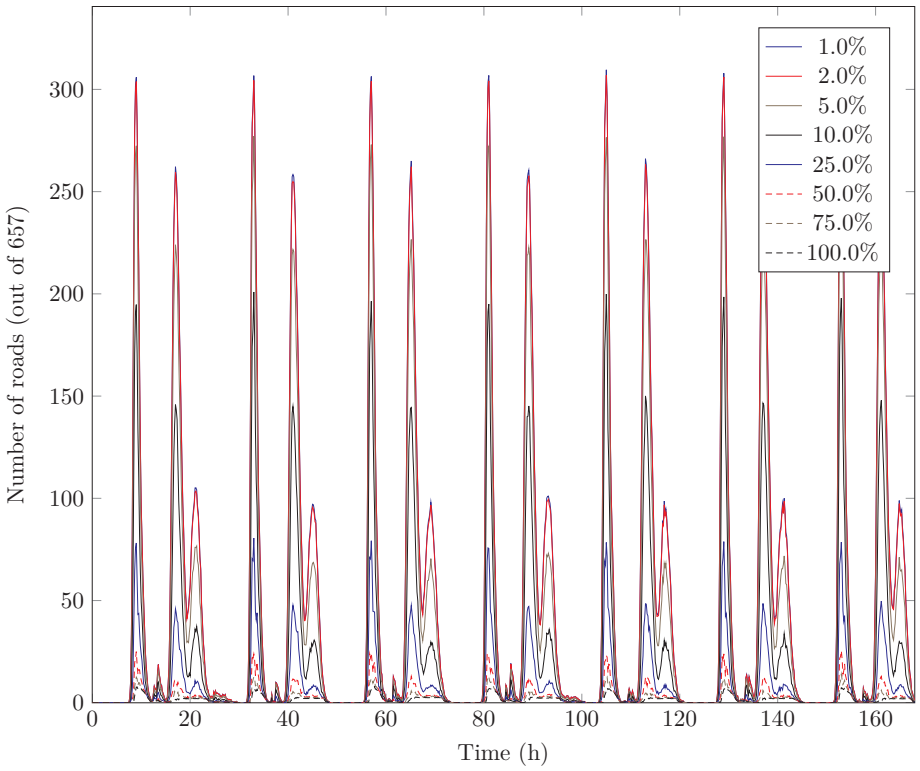


Figure 14: Road capacity over time of Simulation 2

performed here should be minimal on a daily scope[20].

[20]The travel times being recorded and exposed correspond to the total experienced by each motorist over a simulated day.

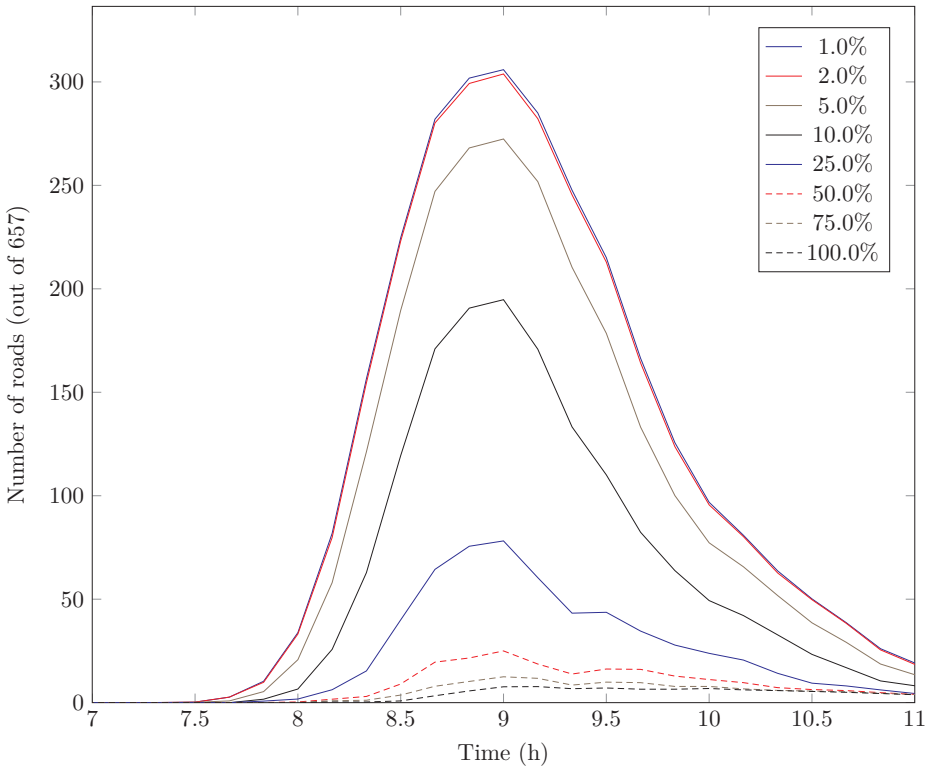


Figure 15: Road capacity over time of Simulation 2 during morning rush hour fo day 1

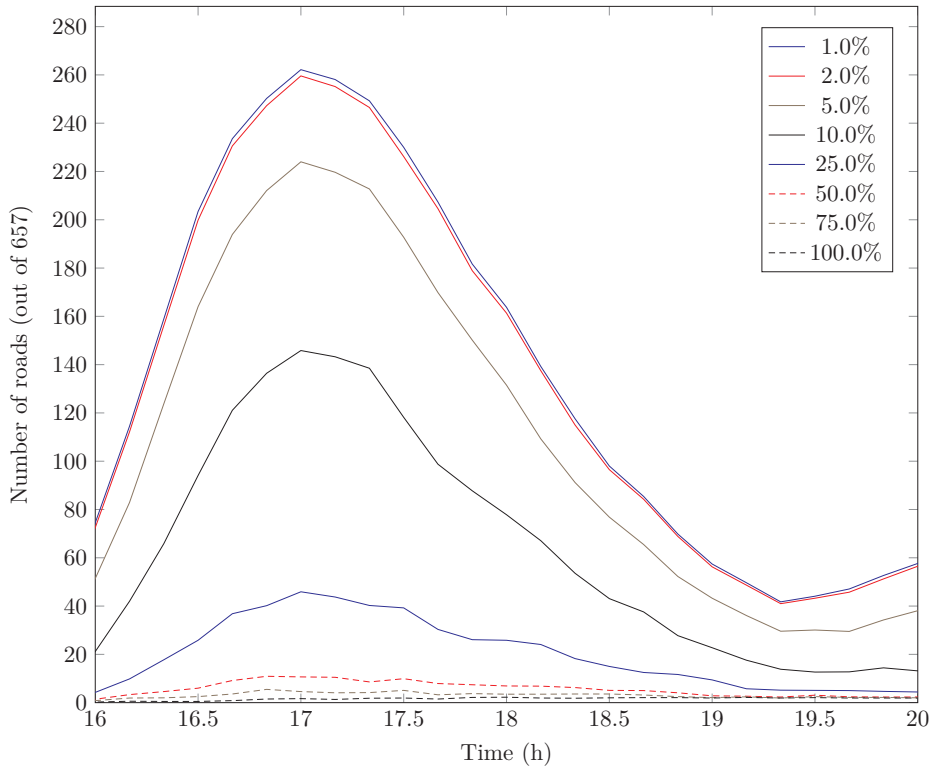


Figure 16: Road capacity over time of Simulation 2 during afternoon rush hour of day 1

6.4 Parameters optimization

Now that we have shown that it is possible to improve the global travel times by directing a portion of the motorists' population we shall try to optimize the parameters of the service provider.
The total number of possible parameters set-up being enormous due to the many learners, request handlers and their respective numerical parameters, the following optimization will be non exhaustive. The general procedure will be to optimize a given parameter while the others are set to their previously best found value when possible or to the default value that was already used by Simulation 1.
The metric of choice to compare the quality of the different parameters set will be the average travel time per agent type ; as indeed we have already shown that the travel distance of free agents does not depend on the directed agents whatsoever and that the real interest of this model is to improve the global time cost of the system which will be directly related to the travel times per agents type.

6.4.1 Other learners

The first analysed parameter is the choice of learning mechanism. Responsible for the quality of predictions used by the service provider to direct the agents correctly, this choice should have an impact on the quality of the directions dispensed.
The learners studied here will be denoted by the following abbreviations:

- *Simp.* is the simplest learner keeping only one average travel time for each road. The required count of observations on a given road for it to be fully learned is 100.
- *L. V.* is the last visited learner which keeps tracks of the last 5 recorded instances of directed agents having gone through a given road. The weights used by the prediction mechanism for each of the recorded entries are $0.3, 0.275, 0.2, 0.125$ and 0.1. The time certainty since the last recorded entry, to determine whether a road has been fully learned is set to 30 minutes.
- *Back.* is the back propagation learner with learning factor of 0.9 and a fully learned determinant of 5%.
- *Inter.* is the interval learner with 264 intervals per day. The certainty threshold for the amount of observations required to consider a road

fully road is set to 100.

Figure 17 is the visual comparison of the travel times between the different learners while Table 6 reports the numerical averages and standard deviations in travel times per agent type. From these we can see that all four learners give very similar results.

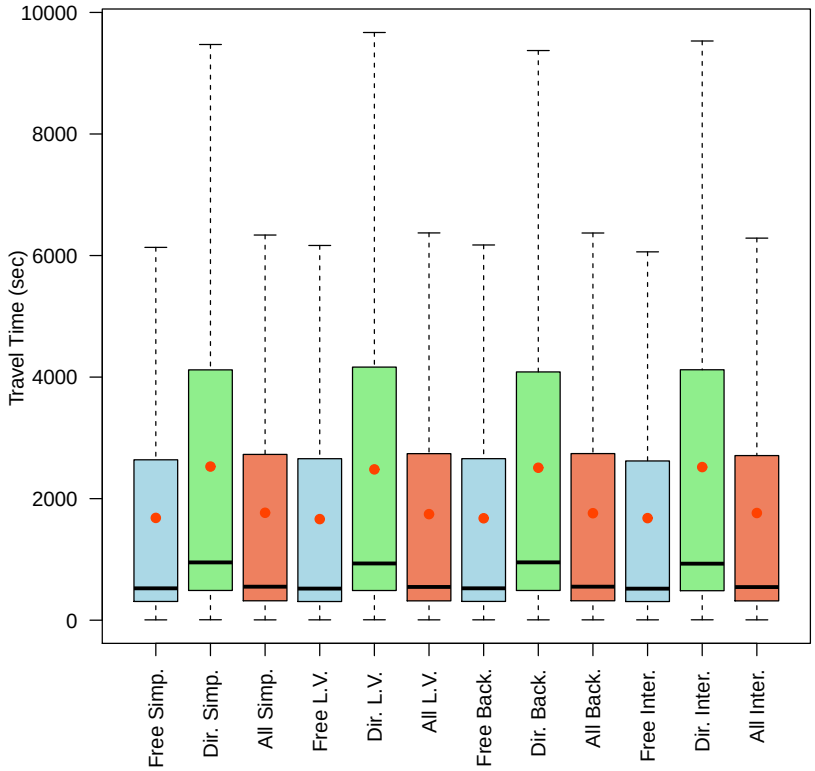


Figure 17: Comparison of travel times between learners

Indeed the perceived travel times with the simplified, back propagation and intervals learners are almost indiscernible in their quality. The last visited learner gives however slightly better results as it has around 15 seconds less (slightly less than 1%) on average for free agents, 35 seconds (around 1.25% improvement) less for directed agents and independently of agents type ends up with almost 18 seconds (1.1% improvement) less when compared with the other three learners.

	Simplified			Last Visited			Back Prop.			Intervals		
	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All
Avg. TT	1683	2528	1768	1663	2481	1745	1677	2508	1760	1680	2519	1764
TT Dev.	2223	2888	2312	2181	2852	2270	2207	2816	2289	2267	2926	2354

Table 6: Comparison of mean and deviation of travel times between learners

This leads to the conclusion that the high responsiveness of the last visited learning mechanism suits the problem at hand best, though only slightly better than the alternatives. Its capability to update its predictions throughout the day while not maintaining a very long memory of previous observations helps the learning of fast changing network conditions leading to more accurate predictions. This ends up helping to direct drivers away from congested road segments, as the decrease in free agent travel time indicates, and find them the shortest alternate path as suggested by the decrease in directed agents travel times.

6.4.2 Marginal costs

As the difference between learners was very small, we will try to compare them again, this time with the addition of marginal costs in the service provider path finding algorithm. The raw numerical results can be seen on Table 7 while the visualization of the simulation's output is present on Figure 18. Three observations can be made out of these results.
The first one being that for both the simplified learner and the back propagation one, the use of marginal costs does have any effect on the free agents travel time but has a negative one on the time cost of directed agents.
The second observation is about the last visited learner, previously considered the most efficient one. With marginal cost however it has become the worst possible set of parameters so far, as not only the directed agents take longer but it also negatively affects the free agents. It would seem marginal costs counter the positive aspects of the last visited learner by lessening its adaptability and decreasing the quality of its predictions when it comes to congested roads ; which results with directed agents adding to the traffic jams, slowing down both themselves and their free counterparts.

The final observation is also the only positive one. The intervals based learning method see its results improved by the addition of marginal costs, making it the only positive effect of marginal costs. Average travel times

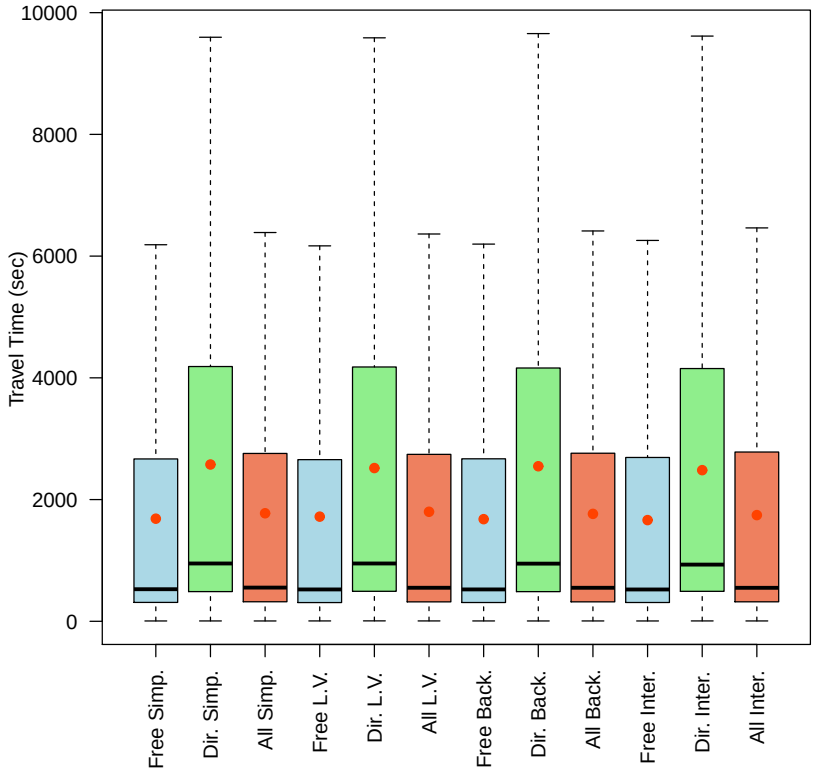


Figure 18: Comparison of travel times between learners with marginal costs

here being almost exactly the same compared to the last visited learner without marginal costs, the previously best found parameters set. However the interval learner can boast being the more stable of the two, as it has slightly smaller variance independently of agent type.

To conclude on the matter of marginal cost or no, we can state that it should not be used with any other learner than the intervals based one where it provides vast improvements.
For the following, we will keep both the intervals' learner with marginal costs and the last visited learner without said costs into consideration for further optimization.

	Simplified			Last Visited			Back Prop.			Intervals		
	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All
Avg. TT	1686	2576	1775	1720	2518	1799	1678	2547	1765	1663	2483	1745
TT Dev.	2178	2944	2282	2340	2820	2404	2169	2891	2267	2117	2803	2209

Table 7: Comparison of mean and deviation of travel times between learners with marginal costs

6.4.3 Exploration

Another parameter of the service provider is the exploration factor which so far was left at 0. Giving a positive value to this parameter would make directed agents be given random directions with a probability equal to the given value.
This addition is used in general to push greedy learning mechanism towards more exploration and less exploitation. Its benefits in our case remain to be analysed. To do so, simulations were performed for the two parameters sets identified above (intervals learner with marginal cost and last visited without) with an exploration factor set to 0.1. The results of these two simulations will then be compared with the ones observed previously, without the exploration behaviour.

Figure 19 allows us to clearly see that when using added exploration, the travel times of free agents does not seem affected but directed agents, on the other hand, end up spending more time on their daily commute. Indeed the average, the median and both shown quartiles are visibly higher for the exploring simulations than their non exploring counterpart.
Table 8 helps to confirm the previous observation by giving the actual number behind it: the directed agents see an increase of around 10% in travel times while free agents see an increase of 4% for the intervals learner and 3% for last visited.

	Intervals			Inter. expl.			Last Visited			L.V. expl.		
	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All	Free	Dir.	All
Avg. TT	1663	2483	1745	1730	2734	1831	1663	2481	1745	1716	2757	1820
TT Dev.	2117	2803	2209	2297	3054	2402	2181	2852	2270	2276	3161	2400

Table 8: Comparison of mean and deviation of travel times based on exploration

The addition of forced exploration does not help the central authority to di-

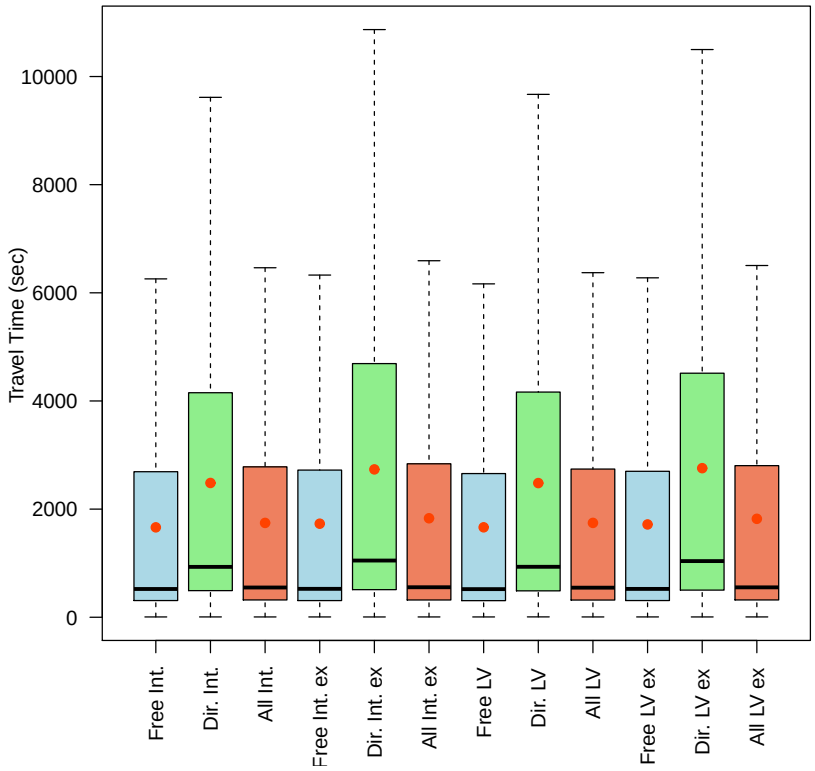


Figure 19: Comparison of travel times based on exploration factor.

rect accordingly its agents. The learning mechanisms with this many agents do not need any more exploration as they achieved their best learning capabilities of the network's state fast enough without it. As such the addition of randomly picked paths only makes directed drivers take suboptimal paths. Those not only increase their travel cost but also pushes them in the way of the non-directed motorists, slowing them down, whenever the random paths intersect with the absolute shortest ones.
It is however possible that the tested 10% added exploration is a too high value. The effect of a smaller percentile capable of improving the travel costs, would probably be negligible.

6.4.4 Request handler variation

The last remaining parameter to be changed in the service provider is the most important one: the choice of the algorithm in charge of computing the paths given to the directed agents. We have so far only used *normal A**[21] and found the combination of it with the *LV learner* without exploration or marginal cost to give the best results. In the following the use of *reduced A** with a depth of 5 will be discussed based on a comparison of this variation with the previously exposed one.

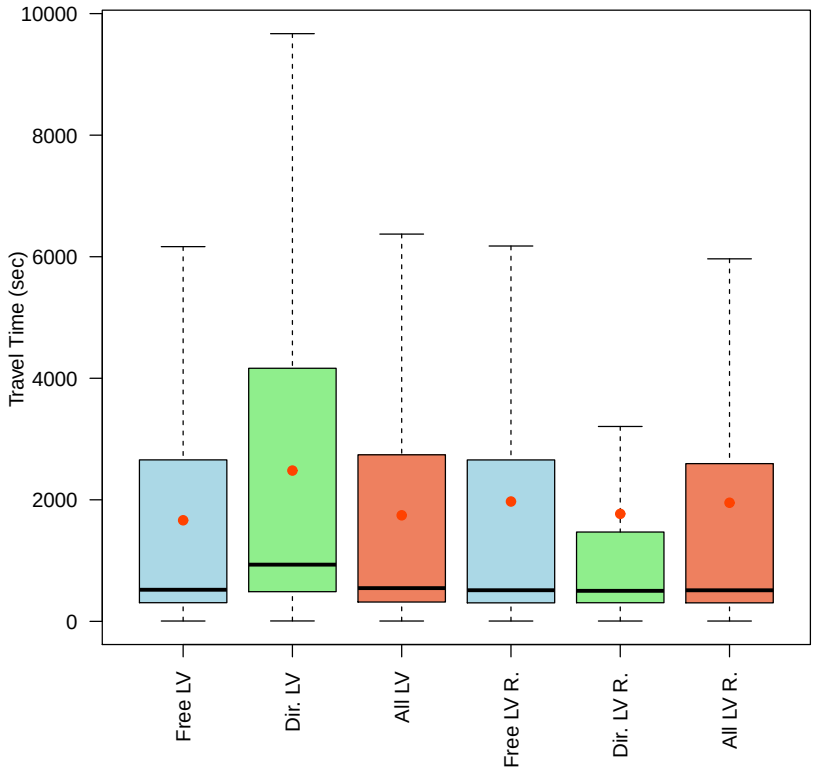


Figure 20: Comparison of travel times between A* and reduced A*.

The use of *reduced A** changes the resulting travel times dramatically which

[21] It is however true that variation in the use of marginal costs and the exploration behaviour will affect this algorithm behaviour.

is shown on Figure 20[22]. Indeed the directed agents seem to have their travel times decreased significantly. Furthermore the major bulk of recorded results is much more compact (based on the shorter interquartile range). Surprisingly the average travel time is above the 75% quartile, indicating that a small percentage of all the directed motorists have had longer commutes, but those were much longer than for the other travellers. Table 9 confirms this by the larger than usual standard deviation.

Another interesting feature can be deduced from these results. The use of *reduced A** improves vastly the travel times of directed agents however induces worse figures for free drivers and consequently worse global results. The reason behind this is that *reduced A** has a smaller scope: it only uses the learned information up to 5 links from the current agent position to compute its best path. Agents directed this way will thus not stray far from the absolute shortest paths. Past his scope, the algorithm bases itself on the absolute shortest travel times to complete the path which leads to it creating small detours around traffic jams while still staying generally close to the absolute shortest path up to the agent destination.

	LV			LV R.		
	Free	Dir.	All	Free	Dir.	All
Avg. TT	1663	2481	1745	1972	1769	1952
TT Dev.	2181	2852	2270	3376	3403	3379

Table 9: Comparison of mean and deviation of travel times between A* and reduced A*.

Directed agents having better results can be explained by the following: they will go through paths closer to the absolute shortest with no regards to the current, or currently learned, system state. This means that they will use overall better paths when the network is not congested. It however also means that they will encumber said paths more which is the reason behind the worse results for free agents that can be observed when compared with *normal A**. This last fact is also the reason behind the larger deviations of travel times ; a small portion of the controlled drivers end up stuck in traffic while the majority of them follow shorter paths than with *normal A**.

Even though the overall travel times simulated with *reduced A** are worse

[22]Boxes labelled with R use *reduced A**, the others do not.

than by the normal variant, this parameters set-up can still be considered the best so far. When compared with the results of Simulation 2, where no directed agents were present, we can still see a some improvement in global travel times (3.6% decrease) and for free agents (2.6% decrease) but this is also the first instance of improvement for directed agents with a 12.6% decrease of travel times. Considering such results better can be argued based on the fact that this is the only set of parameter where a reasonable agent would not revert back from being directed to being free ; as all previous simulations have had resulting travel times for directed agents worse than the ones they would have experienced if free.

Still using the data reported in Table 5 as reference we can also observe that the deviation of the results with *reduced A** is comparable with those from Simulation 2. This again suggests that the directed agents here follow paths closer to those of the free agents.

6.5 Directed agent ratio

We have now identified two very promising parameters sets: last visited learner without added exploration or marginal costs and *A** (denoted *LV* in the following) and the same using *reduced A** (abbreviated to *LVR*). To further analyse these two different simulation set-up, tests were performed with regard to the ratio of directed to free agents. The same input network as previously will be used with the same total of 20.000 agents but the number of directed ones will now vary. Of course, the base case of 0 directed agents was already shown and discussed as it was Simulation 2. Simulations were thus performed from by increasing steps of 1000 added agents (5% of the total population) from 1000 up to the maximum 20.000.

6.5.1 Last Visited & *A**

We can see from Figure 21 that quite clearly the travel times with *LV* get much worse the greater the directed to free agents' ratio is. This is especially true for the directed agents. The reason behind this is that the service provider has less and less information to learn about the network state, as it is itself responsible for more and more of the traffic. This leads to the idea that the *A** learner, which will avoid all form of congestion, directs its agents to avoid the paths that other directed agents have just taken, and as

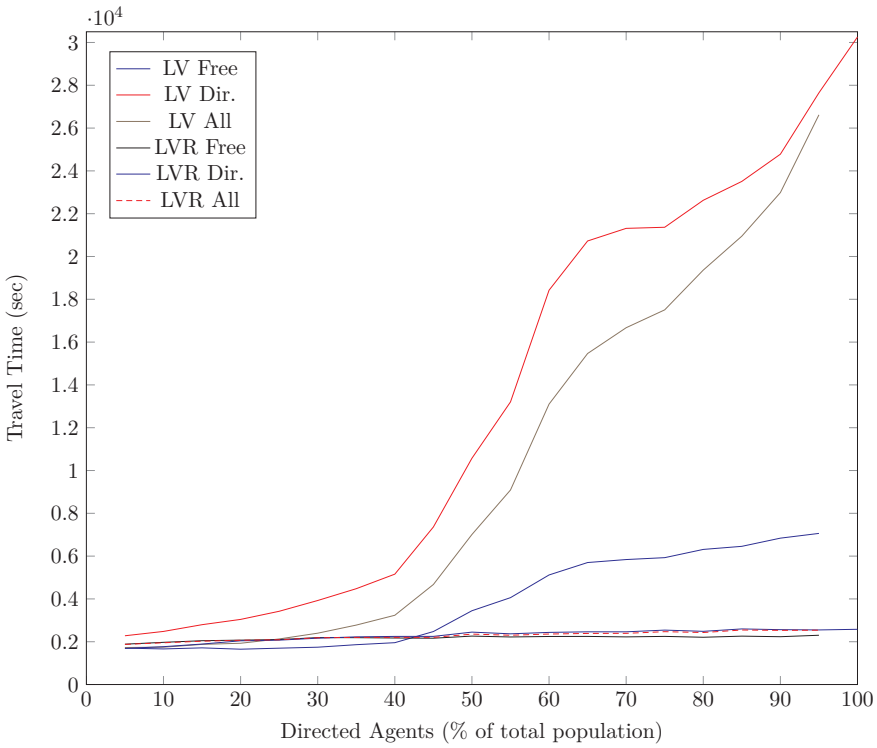


Figure 21: Travel time per ratio of directed agents

such to avoid each other. In turns, this creates more congestion, this time on the secondary paths, which have in general have less capacity. A much higher number of smaller roads thus ends up overused.

In fact, as seen on Figure 23 for the travel times and Figure 24 for the distances of the *LV* simulation with 95% directed agents, the service provider learns that the absolute shortest paths are congested on the first day of simulation. By lack of learned data roads segment it directs its agents there, creating the congestion, and learning about it as a result. After learning that the shortest paths are highly congested, the service provider will start exploring alternative paths but due to the fact that it controls the majority of agents, these alternative paths will also become part of the traffic jams. This can be easily seen on Figures 23 and 24 result visualization for the LV simulation with 19000 directed agents. It is shown there that both the travel times and distances have an extreme increase between day one and the following.

The congested state of the road network can also be observed on Figure 22, where the phenomenon described above can even be refined to the morning rush hour of day one, after which the directed agents seem to diverge more and more from the absolute shortest paths leading to a more continuous congestion of a greater part of the network. The service provider having learned that the shortest paths are congested it will direct agents on alternative paths. This leads to high congestion due to the amount of directed agents on these alternative paths. $A*$ will then start oscillating between more exploration (using even longer paths composed) and reverting back to using previously explored streets, as the last visited learner will consider these not fully learned if no agent goes through them for an elongated period.
The whole problem resides in the fact that the request handler directs too much traffic and will congest any roads that it has not explored yet almost instantly, leading to the overall high capacity usage of a much greater portion of roads than with less directed agents.

Both directed and free agents are affected by this. Indeed as more of the small roads become congested, the free agents that usually would have not had encountered any traffic jam, now do. Highly congested traffic still remains present in the zones where it always has.

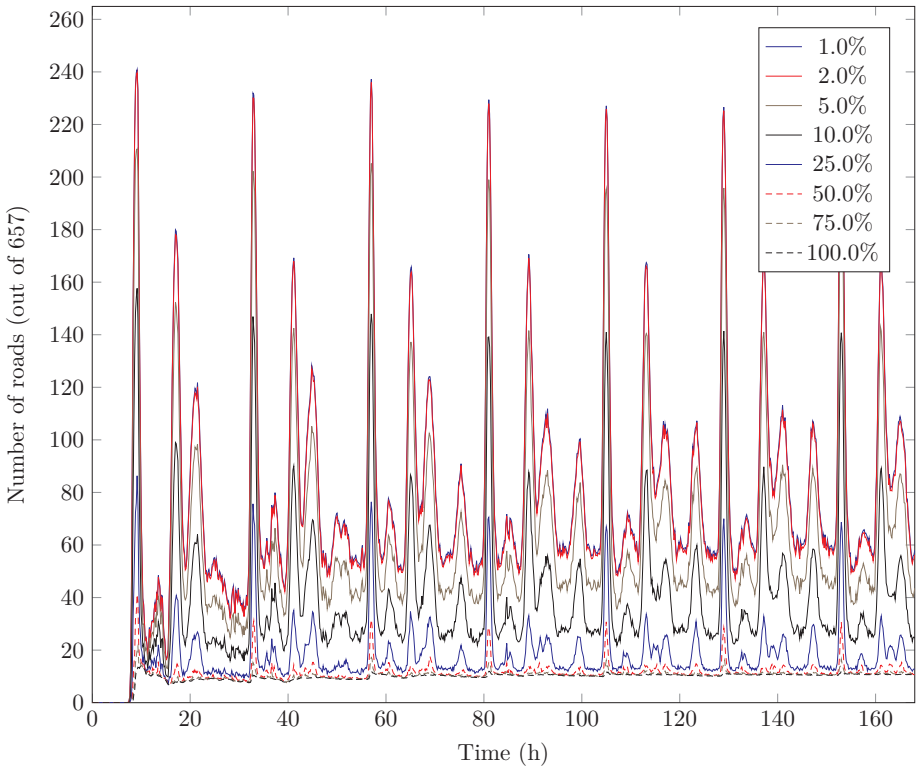


Figure 22: Road capacity usage over time with *LV* and 19000 directed agents

From a more general point of view, it can be observed that the effect of this problem gains more importance when the directed agents become the majority, with a massive increase in travel times between 45% and 50% of directed agents[23]. This is a soft threshold after which the service provider learns more about the results of its own decision scheme than about the uncontrolled traffic, as it was originally designed to, leading to the above mentioned problem.

6.5.2 Last Visited & *reduced A**

The phenomenon of over saturation of alternative paths is less prominent with *LVR* simulations due to the less exploratory nature of *reduced A**, as can be seen on Figure 21. Figure 25 shows that it is however still present in

[23]This is the case for *LV* and, to a lesser extent, for*LVR*

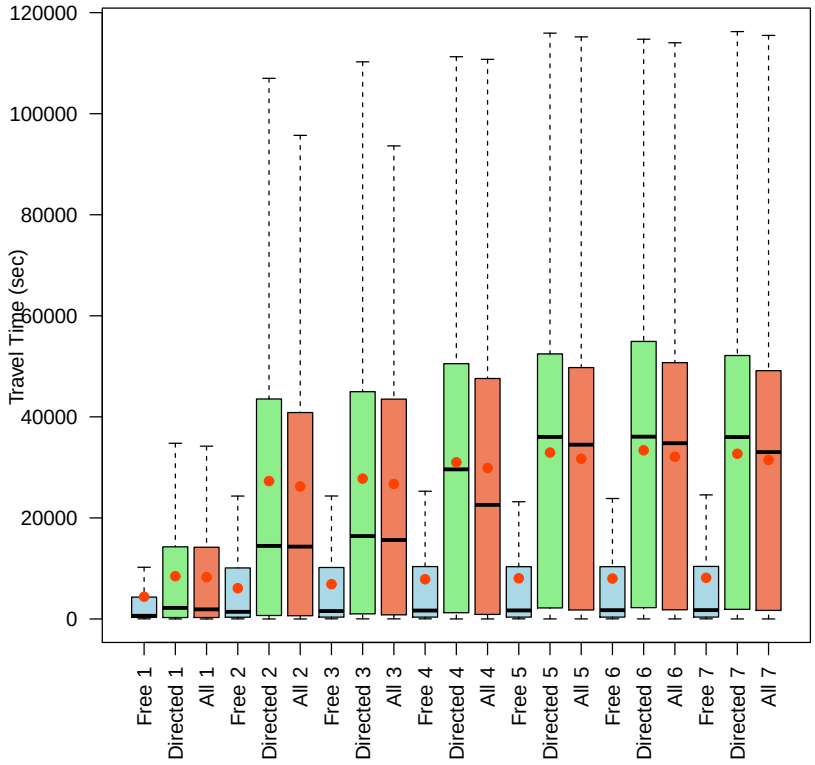


Figure 23: Travel Times through the week of simulation with LV and 19000 directed agents.

some form as the travel cost of all agents does increase with the proportion of directed agents.

This request handler does indeed tend not to direct any agent far from the shortest paths which minimises the negative effects exposed above. Indeed, *reduced* $A*$ will not overuse most alternative paths the same way $A*$ does due to its limited scope. Despite this, it will still create more traffic jams but only on roads that are close to the absolute shortest paths used by the free agents. Meaning that a lesser amount of commuters are directed on smaller roads and as such the increase of the levels of congestion is lessened. This can clearly be seen on Figures 26 and 27, the graphs representing the

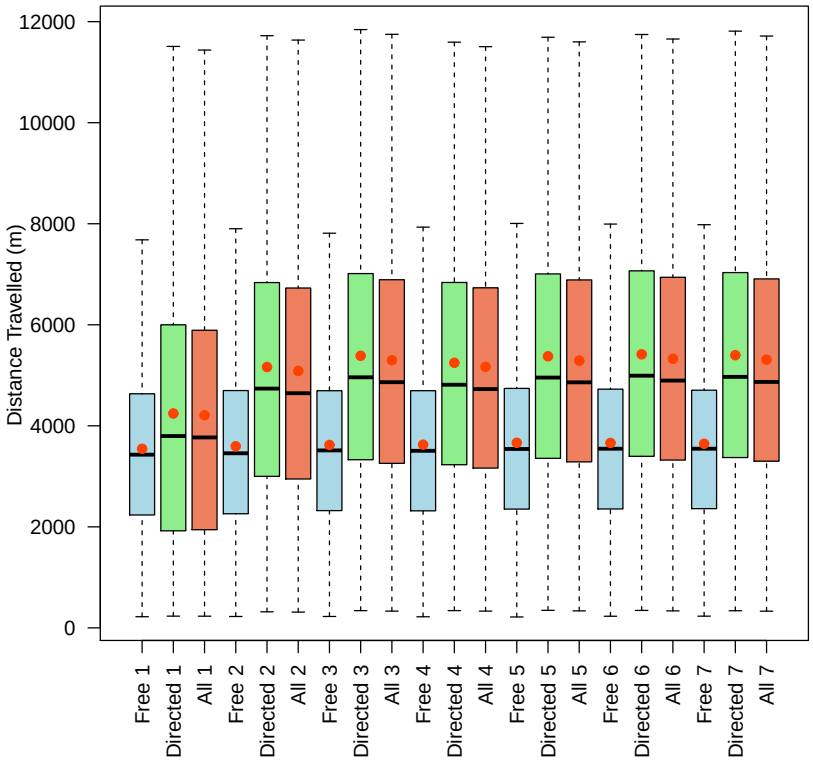


Figure 24: Distances through the week of simulation with LV and 19000 directed agents.

capacity usage of roads with *LVR* and 19000 agents ; which are much closer to the ones shown for Simulation 1 in Figures 9 and 10

6.5.3 Avoiding overuse of directed agents

The problem exposed above caused by the increase of directed agent proportion has an easy solution with the assumption that the service provider knows the optimal percentage of directed agents for the network. Indeed the request handler could simply, whenever it controls more agents than needed, decide to direct the overflow of motorists the same way free drivers would have behaved. The number of actually directed agent would then never be

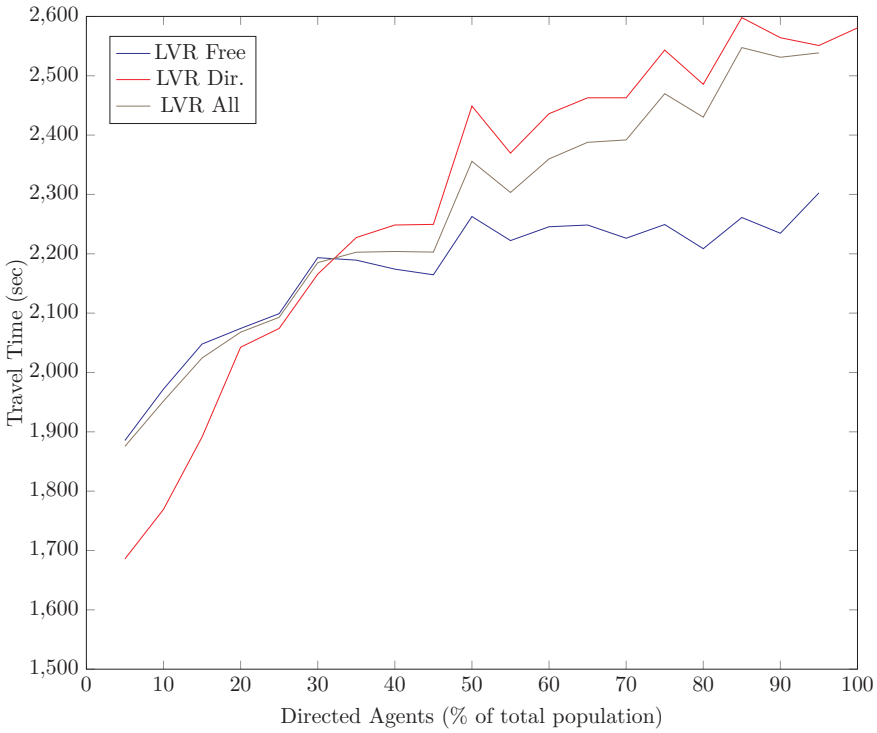


Figure 25: Travel time per ratio of directed agents

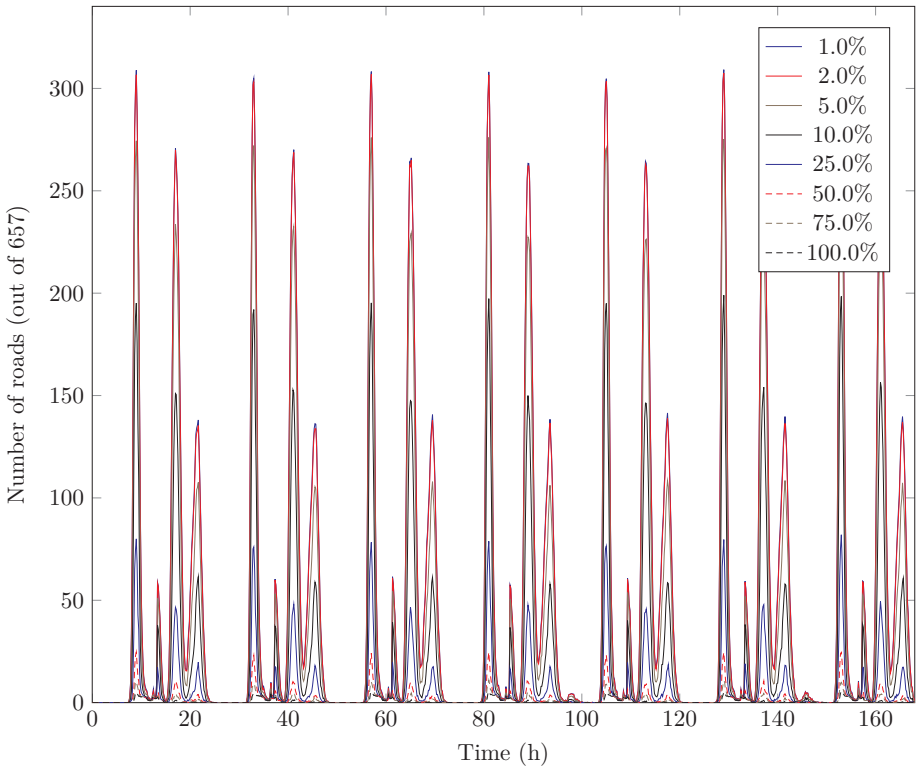


Figure 26: Road capacity usage over time with *LVR* and 19000 directed agents.

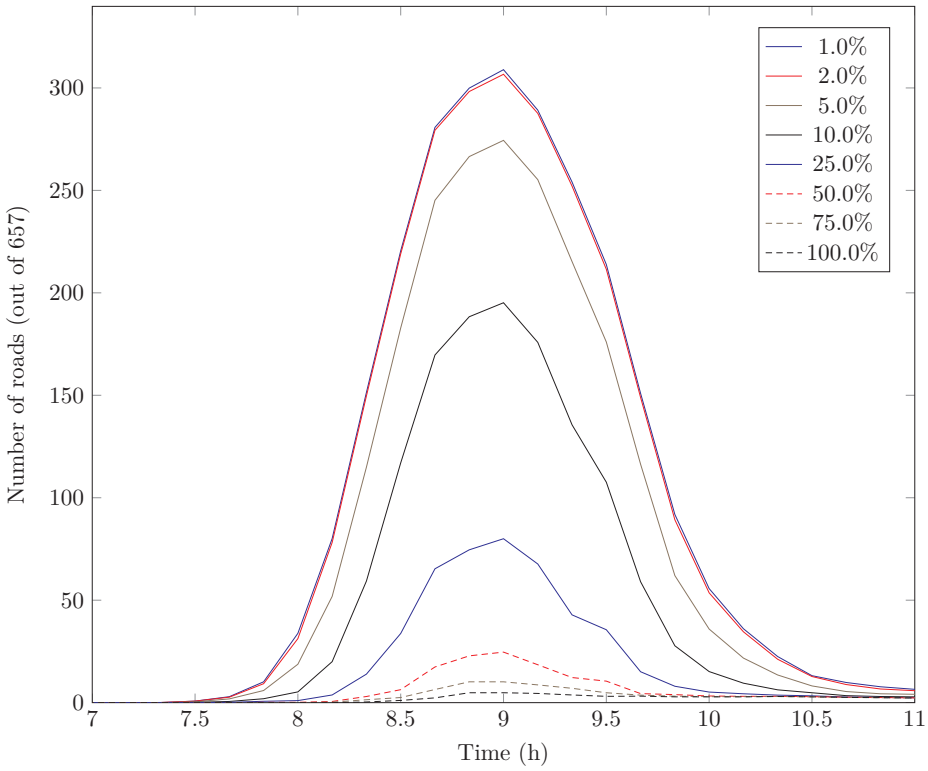


Figure 27: Road capacity usage over time with *LVR* and 19000 directed agents during morning rush hour of day 1.

more than the optimum. With this simple change, all results would be the same for more than the optimal 5 percent of directed travellers.

The problem of finding the optimal ratio of directed to free agents however still remains. A proposed variation of the service provider would see it start with a small amount of actually directed agents amongst its total available, given by the amount of directed agents of the simulation. It would then proceed by increasing the count of actually directed drivers until the increase of travel times by said motorists is perceived. As such, the slow increase of really directed commuters will proceed on a day to day basis.
Starting with a very small amount would guarantee to never have the extreme case of congestion exposed above.

The optimal percentage will be naturally dependent on the network but also on the total population. As indeed, if not enough population should be present to have any congestion in the first place, there should be no reason to have any directed drivers as they would only increase the global travel time of the system by taking longer routes. The method exposed here would also solve problem of varying amount of drivers on a network by dynamically adapting the amount of agents following the directions of the request handler.

With such changes put in place, the service provider should be able to optimize the ratio of (actually) directed agents to all form of free agents, whether real free agents or drivers directed to follow the same basic behaviour as free agents. The problem of overall congestion exposed previously should thus be eliminated.

6.5.4 Travel Times at the optimum

As previously discussed it could be possible to force the service provider to only ever have the optimum amount of directed agents in the system. This amount is, based on Figure 21 no more than 1000 for both *LV* and *LVR* simulations with a total of 20.000 agents.

Results with 1000 directed agents, the optimum value tested, can be found on Figure 28 and Table 10. The later also show the percentile improvement of travel time with regards to the results from Simulation 2, the simulation

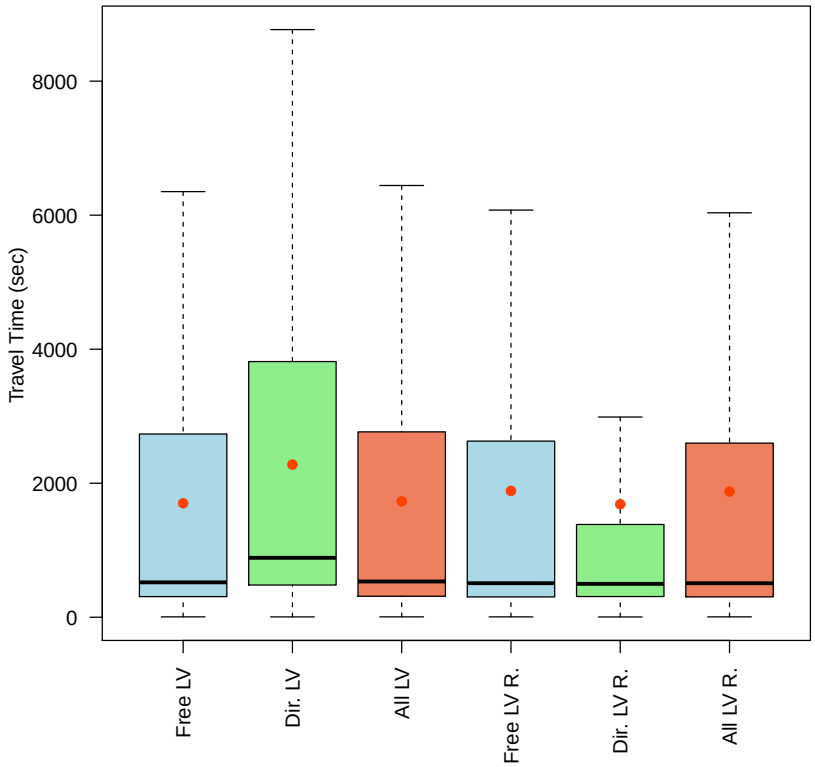


Figure 28: Comparison of travel times between A* and reduced A*.

free of any directed agents. These again reinforce the difference between *LV* and *LVR* already discussed in Section 6.4.4.
The quality of these results show that the designed service provider architecture can give promising results towards better overall travel times if sacrificing the directed drivers (*LV*) but also to a lesser extent improvements for all agent types (*LVR*).

	LV			LV R.		
	Free	Dir.	All	Free	Dir.	All
Avg. TT	1701	2279	1730	1885	1686	1875
Avg. TT decrease from Sim. 2	16%	-12.5%	14.6%	6.9%	16.7%	7.4%
TT Dev.	2297	2614	2317	3170	3191	3172

Table 10: Comparison of mean and deviation of travel times between A* and reduced A* with 1000 directed agents.

6.6 Informed agents

So far, only free and directed agents have been discussed. A third type of agents was however designed: informed agents. These agents, extending the behaviour of free agents, have access to the exact state of the system around there current position[24]. They then use *reduced A** to find their way through the network without being directed by a central authority.

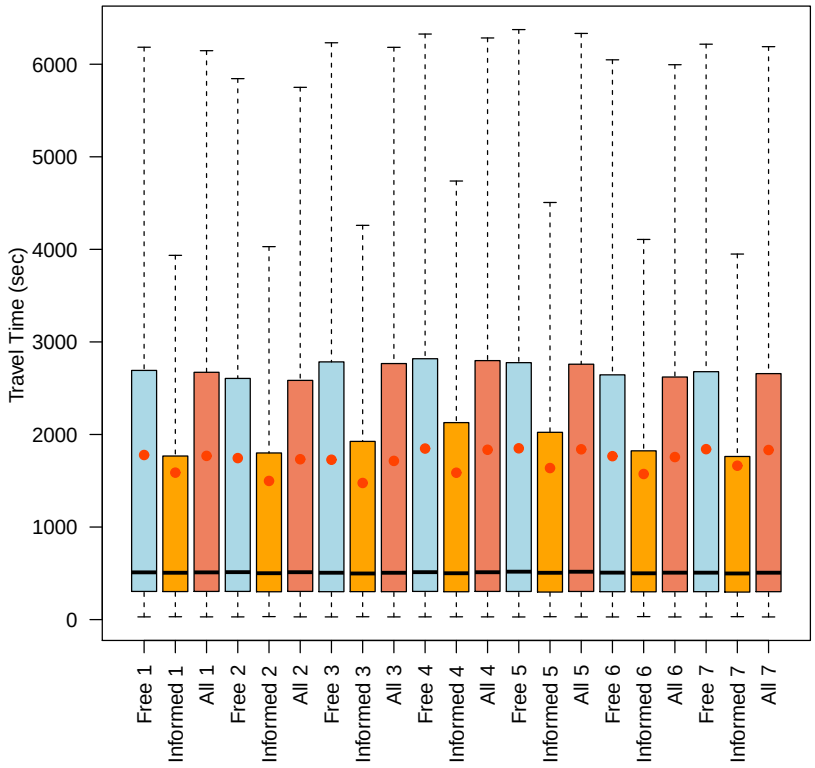


Figure 29: Travel times per agent type in informed simulation.

This access to perfect information can be seen as unrealistic when considering that real drivers would not have the perfect information but rather an

[24]The travel time on the roads connected to the node hey currently are in, or is at the end of their current street segment.

appreciation on the congestion level of the nearby roads. Informed agents are thus only used for theoretical purpose.
It is however interesting to note that this type of agent are equivalent to directed agent where the service provider were to use *reduced* $A*$ and the learner were to have access to perfect knowledge concerning the network's current state.
To showcase this set-up, a simulation was done with the optimal amount of directed agents exposed above as the number of informed drivers ; the 19000 others being free.

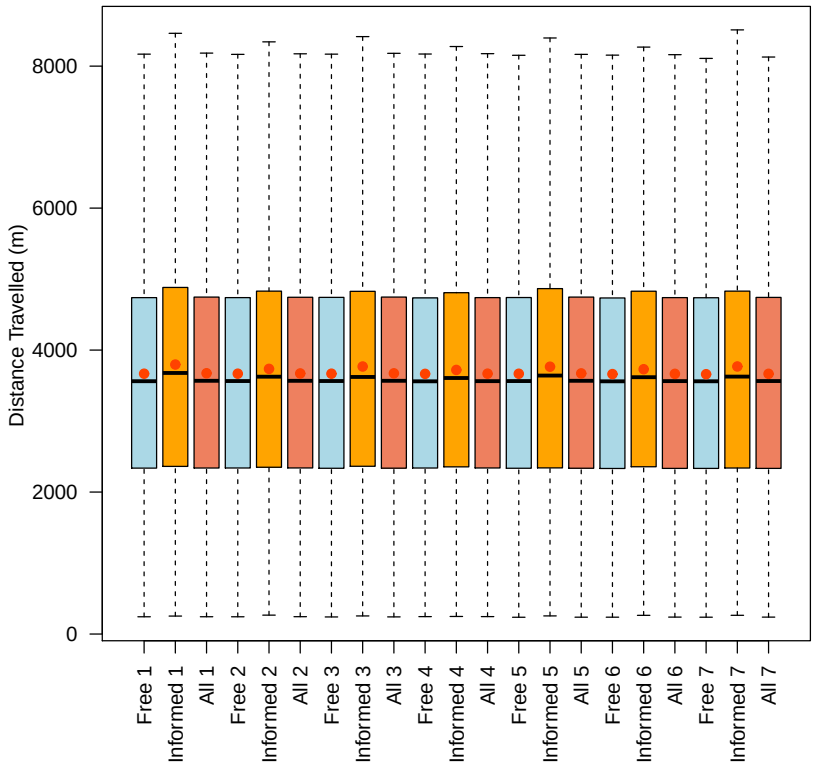


Figure 30: Distances per agent type in informed simulation.

Results of the informed simulation can be seen on Figure 29 for the travel times and Figure 30 for the distances. The raw numerical results are pre-

sented in Table 11. From these statistics we can conclude that informed drivers do indeed behave like directed agents with *reduced A** request handler. However it is interesting to note that the results for all agents type is even better here. Indeed the access to perfect information allows the informed agent to know exactly when taking a detour would be beneficent. This explains the shorter travel times of informed agents and by extension of the fact that they will more likely manage to avoid zone of extreme encumbrance, the improved free agents time costs.
We can also conclude that the learning dynamism used in previous sections with *reduced A**, the last visited learner, is not entirely accurate. If it was, the average travel times exposed in Table 10 for the *LVR* simulation should be better ; as they should be equal to those of informed agents.

	Day 1			Day 2			Day 3		
	Free	Inf.	All	Free	Inf.	All	Free	Inf.	All
Avg. TT	1778	1588	1769	1745	1498	1732	1727	1476	1715
TT Dev.	2762	2703	2760	2666	2392	2653	2532	2275	2521
Avg. Dist.	3667	3796	3674	3667	3736	3671	3667	3769	3672
Dist. Dev.	1859	2054	1870	1857	1953	1862	1861	2798	1919
	Day 4			Day 5			Day 6		
	Free	Inf.	All	Free	Inf.	All	Free	Inf.	All
Avg. TT	1848	1587	1835	1851	1637	1840	1766	1573	1756
TT Dev.	2883	2561	2869	2835	2712	2829	2776	2604	2768
Avg. Dist.	3665	3720	3668	3667	3766	3672	3662	3731	3665
Dist. Dev.	1856	1961	1862	1858	2039	1868	1857	1900	1859
	Day 7			Global					
	Free	Inf.	All	Free	Inf.	All			
Avg. TT	1841	1663	1832	1794	1575	1783			
TT Dev.	3025	2914	3020	2783	2594	2774			
Avg. Dist.	3660	3769	3666	3665	3755	3670			
Dist. Dev.	1851	2248	1873	1857	2136	1873			

Table 11: Mean and deviation of travel times and distances in informed simulation.

7 Conclusion

7.1 Overview

Throughout this master thesis the following three main topics have been covered:

- Summary of state-of-the-art techniques to solve the *TAP*. Done by going over a first general definition of the problem and its related concepts, a set of analytical methods and their own specialized constraints as well as an overview of simulation based methods and their different parameters. The idea of optimizing the total travel times further by changing the traffic flows through the use of a central authority, responsible for the path choices of part of the drivers' population, was also introduced.
- The design and implementation details of a traffic simulation having both microscopic, due to its agent oriented nature, and macroscopic aspects while maintaining high levels of realism. The idea of a central authority depending on both a learner and a request handler to direct part of the population of motorists towards better overall travel times was also introduced and detailed in this second part of the thesis. Variations of the simulation's service provider behaviour were also detailed. A total of four different learning algorithms[25] were presented and tested:
  - Simplified learner, where a single average travel time is learned for each road
  - Intervals based learner, where an average travel time is learned for each road and for each interval over a day
  - Last visited learner, where only a set number of the last perceived travel times are kept
  - Error backwards propagation, where the standard back propagation is used to learn the approximated travel time of each road

  Two distinct handlers, whose role was to use the prediction of the learner to compute optimized paths, were also introduced:
  - *A**, the classical shortest path heuristic

[25] A fifth one, the weighted composition of any of the others, is also shown.

  - Reduced $A*$, a slight variation initially designed to reduce the complexity of the simulation process

  With the possible addition of a forced exploration and use of marginal costs, the total number of handlers was brought up to eight.

- The result generation and analysis of the benchmarking of the different parameters sets, on a real life traffic network imported from *OpenStreetMap*, with a first goal of finding (exhaustively) the best combination of learner and handler when trying to optimize the total travel times.
  After finding two promising parameters sets[26], another benchmark was performed while this time varying the proportion of directed to free agents. This led to the discovery of a negative effect between the size of the directed population and the travel times and overall network congestion. As indeed increasing the former proved to increase the later. This lead to the idea of optimal direction ratio over a given network and total population size. Itself showed the way to a slight redesign of the service provider to force it to never exceed said optimal ratio.

7.2 Improvements & Extensions

Some potential changes to either the simulation process itself or the algorithm behind the central authority have already been mentioned. However those and many more alternative have been considered but not implemented nor tested. In the following such revisions will be outlined.

7.2.1 Realistic simulation

Changes to the core of the simulation would require all results and their analysis to be redone. However advancing the model towards more realism would be a true test of said results stability.

Queueing Effect In Section 6.3 the unrealistic fact that roads have no hard limit on the number of drivers on them at any time was showed in action. The counter measure to this was also already indicated: the addition of a queueing effect on roads too full such that any motorist trying to enter

[26] Last Visited learner with either $A*$ or *reduced* $A*$

a road at capacity would have to wait on its current position until a free spot is available.
This would give rise to a more realistic simulation on a microscopic level ; however adding complexity to the whole model. In the case of a theoretical analysis this might not be necessary, as long as the main metric of interest is not affected ; which in our case was true for the global travel times.

Free & informed agents' behaviour The decision making process of free agents can be said to be too simplified while the informed agents on the other hand are unrealistic due to their perfect knowledge. As such an enhancement of the simulation could be to redefine a new way for either of these two agents types[27] to choose their shortest paths.
This could be achieved by taking into account partial information of each driver's surrounding in its computations ; such as a general appreciation of the congestion levels of adjacent roads or a personal learning (and prediction) of the travel times along its daily route.

7.2.2 Refined Predictions

Clearly shown in Section 6.6, the accuracy of predictions will greatly influence the effect on both travel times of directed agents and by extension the population. As such improving the learning mechanism would always be beneficial to the general goal of influencing the system towards its global optimum.

Alternative learner Unexplored in this thesis, the idea of a learning mechanism that will store all experienced travel times of each directed agents could have interesting results. From the stored information, predictions of the future network states could be performed by applying general *big data* heuristics.
Such change would of course infer increase in both computational and space requirement of the simulation.

Monitoring Traffic More traditional means of traffic predictions could be linked with the one used here ; though only available in a real life scenario rather than in simulation. Indeed the traffic flow in most big cities is

[27] Or an entirely new type

already monitored to detect and inform of the presence of traffic jams. This information, if available, could either be used in addition or instead of the learner par of the service provider.

7.2.3 Towards global optimum

Self balancing directed agents proportion As already summarized, the concept of the optimal ratio of directed agents was introduced ; with it came a proposed redesign of the service provider such that the effective amount of motorists receiving direction would never surpass said ratio while tending to be equal towards it. Such changes were only proposed and never implemented or tested to see if the expected effect on travel times was exact. Such tests could be easily performed following the same template of benchmarking as here while also comparing the new results to the ones shown in Section 6.5.

Alternative handler A type of handler that was not investigated here, but could yield promising results, is one based on the concept of *k-shortest* paths. Such a handler would compute not just the one shortest path for each request it receives, but instead would find a given number, k, of shortest paths. The agent responsible for the request would then be directed to a randomly selected path amongst the ones created.
Such an algorithm could also be used to make free agents more realistic. As they could also choose amongst multiple absolute shortest paths instead of always following the same one.

This extension of the handler would however increase substantially the complexity of the simulation. On the other hand, if used for free agents, the bulk of the computations could be performed at a preprocessing step.

7.2.4 Central authority removal

A massive change to the service provider design here would be the removal of the centralized part of it. This could be done different ways:

- The removal of the request handler leading to only a centralized learning mechanism and agents capable of independently computing their own paths ; while still trying to attain better global travel times.

- The complete removal of the central authority by having both path computing and network status learning performed by each agent individually. This will lead to a more swarm intelligence oriented solution to our problem. This could in reality be allowed by new technologies, such as self-driving cars, making agents able to *communicate* any necessary information with each other without the need for centralization.

Such changes would not only lead to more testing but also an entire revision of the simulation process as in all cases the complexity required by the simulation of independent shortest path computations and the second case would induce the need to simulate communication between agents. Both will then lead to increase computational complexity and cost of the simulation - which might make it incompatible with other changes suggested previously.

8 References

[1] Adler, J. L., Satapathy, G., Manikonda, V., Bowles, B., and Blue, V. J. (2005). A multi-agent approach to cooperative traffic management and route guidance. *Transportation Research Part B: Methodological*, 39(4):297–318.

[2] Akçelik, R. (1991). Travel time functions for transport planning purposes: Davidson's function, its time dependent form and alternative travel time function. *Australian Road Research*, 21(3).

[3] Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT press.

[4] Bazzan, A. L. and Klügl, F. (2005). Case studies on the braess paradox: simulating route recommendation and learning in abstract and microscopic models. *Transportation Research Part C: Emerging Technologies*, 13(4):299–319.

[5] Bellei, G., Gentile, G., and Papola, N. (2005). A within-day dynamic traffic assignment model for urban road networks. *Transportation Research Part B: Methodological*, 39(1):1–29.

[6] Bowman, J. L. (1998). *The day activity schedule approach to travel demand analysis*. PhD thesis, Massachusetts Institute of Technology.

[7] Branston, D. (1976). Link capacity functions: A review. *Transportation Research*, 10(4):223–236.

[8] Breiman, L. and Lawrence, R. L. (1973). Time scales, fluctuations and constant flow periods in uni-directional traffic. *Transportation Research*, 7(1):77–105.

[9] Burmeister, B., Haddadi, A., and Matylis, G. (1997). Application of multi-agent systems in traffic and transportation. In *Software Engineering. IEE Proceedings-[see also Software, IEE Proceedings]*, volume 144, pages 51–60. IET.

[10] Carey, M. (1992). Nonconvexity of the dynamic traffic assignment problem. *Transportation Research Part B: Methodological*, 26(2):127–133.

[11] Carey, M. and Ge, Y. (2007). Retaining desirable properties in discretising a travel-time model. *Transportation Research Part B: Methodological*, 41(5):540–553.

[12] Carey, M. and Subrahmanian, E. (2000). An approach to modelling time-varying flows on congested networks. *Transportation Research Part*

B: Methodological, 34(3):157–183.

[13] Cetin, N., Nagel, K., Raney, B., and Voellmy, A. (2002). Large-scale multi-agent transportation simulations. *Computer Physics Communications*, 147(1):559–564.

[14] Chen, H.-K. and Hsueh, C.-F. (1998). A model and an algorithm for the dynamic user-optimal route choice problem. *Transportation Research Part B: Methodological*, 32(3):219–234.

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

[16] Dechter, R. and Pearl, J. (1985). Generalized best-first search strategies and the optimality of a*. *J. ACM*, 32(3):505–536.

[17] Dia, H. (2002). An agent-based approach to modelling driver route choice behaviour under the influence of real-time information. *Transportation Research Part C: Emerging Technologies*, 10(5):331–349.

[18] Florian, M., Mahut, M., and Tremblay, N. (2008). Application of a simulation-based dynamic traffic assignment model. *European Journal of Operational Research*, 189(3):1381–1392.

[19] Gary A. Davis, John Hourdos, H. X. T. M. (2010). Access to destinations: Travel time estimation on arterials. *Report 3 in the series Access to Destinations Study.*

[20] Huang, Z.-f., Ren, G., Lu, L.-l., and Cheng, Y. (2014). A modification of local path marginal cost on the dynamic traffic network. *Journal of Modern Transportation*, 22(1):12–19.

[21] Janson, B. N. (1991). Dynamic traffic assignment for urban road networks. *Transportation Research Part B: Methodological*, 25(2):143–161.

[22] Jayakrishnan, R., Tsai, W. K., and Chen, A. (1995). A dynamic traffic assignment model with traffic-flow relationships. *Transportation Research Part C: Emerging Technologies*, 3(1):51–72.

[23] Jin, W.-L. (2007). A dynamical system model of the traffic assignment problem. *Transportation Research Part B: Methodological*, 41(1):32–48.

[24] Klügl, F. and Bazzan, A. L. (2004). Route decision behaviour in a commuting scenario: Simple heuristics adaptation and effect of traffic forecast. *Journal of Artificial Societies and Social Simulation*, 7(1).

[25] Klügl, F. and Bazzan, A. L. (2012). Agent-based modeling and simulation. *AI Magazine*, 33(3):29.

[26] Knuth, D. E. (1977). A generalization of dijkstra's algorithm. *Information Processing Letters*, 6(1):1 – 5.

[27] Mahmassani, H. (2001). Dynamic network traffic assignment and simulation methodology for advanced system management applications. *Networks and Spatial Economics*, 1(3-4):267–292.

[28] Matloff, N. (2008). Introduction to discrete-event simulation and the simpy language. *Davis, CA. Dept of Computer Science. University of California at Davis. Retrieved on August*, 2:2009.

[29] Nagel, K., Flötteröd, G., et al. (2009). Agent-based traffic assignment: going from trips to behavioral travelers. In *12th International Conference on Travel Behaviour Research (IATBR), Jaipur*.

[30] Nannicini, G. (2010). Point-to-point shortest paths on dynamic time-dependent road networks. *4OR*, 8(3):327–330.

[31] Patriksson, P. (1994). *The traffic assignment problem: models and methods*.

[32] Peeta, S. and Ziliaskopoulos, A. K. (2001). Foundations of dynamic traffic assignment: The past, the present and the future. *Networks and Spatial Economics*, 1(3-4):233–265.

[33] Rodrigues Gomes, E. and Kowalczyk, R. (2009). Dynamic analysis of multiagent q-learning with ε-greedy exploration. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 369–376. ACM.

[34] Scerri, P., Vincent, R., and Mailler, R. (2006). *Coordination of large-scale multiagent systems*. Springer.

[35] Sheffi, Y. (1985). Urban transportation networks: equilibrium analysis with mathematical programming methods.

[36] Tong, C. and Wong, S. (2000). A predictive dynamic traffic assignment model in congested capacity-constrained road networks. *Transportation Research Part B: Methodological*, 34(8):625–644.

[37] United States. Bureau of Public Roads (1964). *Traffic assignment manual for application with a large, high speed computer*. U.S. Dept. of Commerce, Bureau of Public Roads, Office of Planning, Urban Planning Division.

[38] Wunder, M., Littman, M. L., and Babes, M. (2010). Classes of multiagent q-learning dynamics with epsilon-greedy exploration. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 1167–1174.

[39] Zeng, W. and Church, R. L. (2009). Finding shortest paths on real road networks: the case for a*. *International Journal of Geographical Information Science*, 23(4):531–543.