Masters in Informatics Engineering
Dissertation/Internship

Final Report

Cosmo

José António Capela Dias
jacdias@student.dei.uc.pt

Advisors:
Fernando Penousal Machado
João Oliveirinha
Date: July 12, 2012

FCTUC DEPARTAMENTO
DE ENGENHARIA INFORMÁTICA
FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Abstract

Due to the ubiquity of mobile devices nowadays, the diffusion of route planning mechanisms such as Global Positioning System (GPS) and the public's demand for information, a way to optimize traffic flow is needed. In order to study this issue, we will look at a city as a complex system and to each driver as an individual agent. The main objective of this dissertation is to influence the behavior of each driver by using an Ant Colony algorithm, in order to achieve maximum throughput in the road network and to minimize road congestion. We use the inherent selfishness of drivers to actually make everyone contribute to the greater good by constantly spreading updated network information which will distribute drivers more uniformly throughout the network. In this study we analyze different algorithms (such as an inverted Ant Colony Optimization) to try to achieve the proposed goal. Also, in order to achieve a more realistic driving behavior we introduce a set of features that are able to reproduce several aspects of a driver personality. The results, obtained on artificial and also real road networks, provided by a road traffic simulator (SUMO), show that it is indeed possible to improve a network's efficiency.

Keywords

ant colony optimization, collaborative route planning, information sharing, intelligent agents, swarm intelligence, traffic congestion, traffic information, transportation management

Contents

List of Figures

List of Tables

Chapter 1
Introduction

In the last few years we have observed a substantial expansion of the mobile devices market. Alongside with this growth, the diffusion of route planning mechanisms such as Global Positioning System (GPS) was a natural development. The increasing demand from the public for faster access to more detailed information will eventually lead to these mechanisms becoming ubiquitous and as necessary as the steering wheel itself. In terms of traffic, the selfishness that is inherent to driving is a known fact: people always want to get as fast as possible to their destination. This, allied with the growing importance of traffic jams due to the increase of CO_2 emissions makes this an issue that needs to be dealt with.

The main objective of this dissertation is to study how influencing the behavior of each driver would allow us to improve a network's efficiency - achieving maximum throughput in the road network and minimizing road congestion. Therefore, in order to study this issue and to perform simulations, we will look at the city as a complex system and to each driver as an individual agent.

By combining GPS systems with real time traffic information we have a means to distribute information about traffic amongst virtually all drivers. But how to avoid the previously referred selfishness and make everyone contribute to the greater good? The idea is to constantly spread updated network information to the drivers and to actually use their selfishness in order to distribute them more uniformly throughout the network. In this paper we will analyze different algorithms to try to achieve the proposed goal with special emphasis on the developed Ant Colony algorithm - *Inverted Ant Colony*. The results, provided by a road traffic simulator that reproduces the individual behavior of the agents, shows that a network's efficiency can be greatly improved by the use of this algorithm.

Another interesting aspect is the individual modeling of drivers. Current microscopic traffic simulators allows for an agent-based simulation, enabling us to control/influence their behavior. These tools already include important driving behaviors such as car following, overtaking and lane changing. However there is still room for improvement, especially regarding personality. Personality traits and emotions influence the way drivers behave and can consequently have an impact in the entire system in which the driver is involved. In order to achieve a more realistic portrayal of real life traffic we propose studying a set of personality features (such as distraction, stubbornness, irregularity and aggressiveness) that enable us to reproduce several aspects of a driver's personality by evaluating the impact that these personal features have on city transit.

In the previous year, a basis for this system was developed: a structure was built to communicate with the simulator and a few route planning algorithms were created. However it had efficiency limitations and was poorly tested.

In terms of structure, this dissertation is organized as follows: Chapter 2 consists of the State of the Art - a review of the literature that contributed to the developed work, in fields such as Traffic management systems, Ant colony optimization, Drivers' Personality and Traffic simulation. In Chapter 3 we will discuss Cosmo Platform Development process, which comprises the exploration of the SUMO simulator, understanding current platform, different testing phases, the implementation of Ant Colony Optimization algorithm and also the development of real life maps and traffic demand. Chapter 4 describes the experimental setup and shows the results, which will be analyzed in Chapter 5. Finally, Chapter 6 will give a final overview of the project and the developed work.

1.1 Planning

This chapter shows the planning referring to the past semester and the expected plan for the following semester. In Appendix A, there are Gantt charts referent to each semester's planning.

First Semester

The plan for the first semester was the following:

Task	Duration
State of the art	1 month
Understanding and experimenting with SUMO	1 month
Understanding and experimenting with simulation platform	1 month
Implementation of first collaborative routing algorithm	1 month
Experimentation	0.5 months
Improvement of first algorithm	0.5 months
Intermediate report	1 month

Second Semester

The plan for the second semester is the following:

Task	Duration
Pheromone variation history mechanism	1 month
Experimentation	0.5 months
Measuring CO_2 emissions	0.25 months
Improve route generation	1 month
~~Differentiated information for users~~	~~1 month~~
Personality parameters	1 month
Experimentation in real city map	1 month
Final report	1 month

The *"Differentiated information for users"* task appears struck through because it was not executed. It was initially more of an idea than an actual task and was not thoroughly discussed with the advisors for that reason. Given the various issues that surfaced while using the simulator with a real city map (which will be further explained in Chapter 3) there was not enough time to explore this idea, which was considered secondary, and therefore we decided not to carry out this task.

Chapter 2
State of the Art

Given that our goal is to study how influencing the behavior of each driver would allow us to improve a network's efficiency we needed firstly to understand which traffic management systems already existed. Then, an understanding of the original *Ant Colony* algorithm and the subsequent *Ant Colony* algorithms resulting from various modifications was required in order to develop an interesting and innovative *Ant Colony* algorithm. Also, since we intended to study the individual modeling of drivers, we needed to understand the most relevant features regarding driver personality. Finally, in order to put our plans into practice, it was necessary to evaluate current traffic simulation tools and to choose the most suitable option.

2.1 Traffic management systems

A car navigation system that provides all users with the same traffic information simultaneously will lead to an increase of the congestion in specific routes. Yamashita [1] developed a traffic flow model to predict future traffic load and in order to do so they divided roads into blocks, in order to measure car density in each of them. To search for the optimal path, drivers can use one of the following route choice mechanisms:

- *Shortest distance* (does not take congestion levels into consideration)
- *Shortest time* (based on current levels of congestion)
- *Route information sharing* (*RIS*): each time a driver reaches an intersection, his current position, destination and route to destination is transmitted to the route information server. The server then estimates future traffic congestion and sends the information regarding those estimations back to the drivers. Each vehicle analyzes that information and re-plans its route accordingly. This cycle is continuously repeated.

Their results show that the *RIS* has a greater performance than the Shortest distance or Shortest time algorithms and that it's efficiency increases with the percentage of drivers that use that system.

- In this paper they claim to have achieved a good performance but they rely on personal information from the drivers. This will not be taken into account in the Cosmo system given the fact that having access to this information might raise privacy issues, which some users of the system might not be comfortable with. Also, due to their system being based on cycles, there is a possibility for repetitive cycles in which each driver's change of plans affects other drivers' plans, which in turn affect the first one, and the cycle could go on indefinitely.

Adler [2] developed the *CTMRGS* (Cooperative Traffic Management and Route Guidance System), a cooperative, distributed multi-agent system that uses principled negotiation to improve dynamic routing and traffic management. Their goal was to achieve an "efficient reallocation of network capacity over time and space without seriously violating any individual user's preferences".

To measure driver's satisfaction they used a linear normalized weighted utility maximization model and they assumed that every vehicle has the ability and the equipment to learn, define and calibrate their route plans.

There are three different types of agents with independent goals and decision-making strategies in their system: the *Agents-IRANS* (who represent drivers that have a set of weights across the goals and a set of anchor points to define the normalization curve), the *Agents-ISP* (information service providers), the *Agents-TMC* (system operators). Given that this process is based on negotiations, there is a need to identify the boundaries of the negotiation space of each *Agent-IRANS* and for that purpose there were created two threshold parameters: path utility indifference threshold (paths with the same utility are equally preferable) and weight change tolerance threshold (boundaries over which the agent will allow his set of weights to be adjusted). These parameters will model their resistance to change paths during negotiations.

The *Agent-TMC* keeps track of the network status and with the data it collects it has the ability to predict the length of queues and departure rates. If the simulator supports traffic signals, it can choose to use a fixed-time phase control strategy or a dynamic phase control strategy. The *Agent-ISP* collects network information from the *Agent-TMC* and acts as a mediator for negotiations with *Agent-IRANS*, with the objective of maximizing the throughput of the network.

The negotiation process is as follows:

1) Pre-trip analysis and proposal by *Agent-IRANS* – the *Agent-IRANS* contacts *Agent-ISP* to schedule a route assignment and submits his set of preferred routes;
2) *Agent-ISP* analysis - *Agent-ISP* assesses the received data and determines whether to accept the route proposal or to make a counter-offer.
3) *Agent-ISP* counter-proposal - If the *Agent-ISP* rejects the initial route offer, it generates an alternative route choice;
4) *Agent-IRANS'* evaluation of the counter-proposal - If an agreement cannot be reached (choices do not satisfy its option space) the *Agent-IRANS* will leave route choice at the discretion of the driver.

They came to the conclusion that the *CTMRGS* negotiation process allocates drivers more evenly over the network (improving performance) without compromising drivers' satisfaction.

- In this work, they developed an interesting negotiation system that enhanced the efficiency of the network. On the other hand, they admit that the utilized model may be too simplistic and that further testing should be executed.

Paruchuri [3] intended to show the viability of applying multi-agent simulation for unorganized traffic, by modeling the behavior of drivers to mimic human behavior. Each driver is given a set of characteristics, like free will speed or free will braking power, which will model their behavior as being cautious, normal, and aggressive.

Their model consists of a centralized agent and a blackboard concept. The roads can be thought of as a blackboard where each agent continuously updates it. Every sector and junction is modeled as an agent who senses the vehicles passing through. Each driver has a set of macro (final destination) and micro goals (decision at each point of time) but the best micro goals might not lead to the macro goal's optimal plan.

An interesting feature of their work is the existence of overtaking, which is influenced by confidence and rush factors. The confidence factor is a function that takes into account the current speed of the vehicle, the free will speed and the expected speeds of the other vehicles involved in the overtake. The rush factor represents the urgency to achieve a goal.

There is also the occurrence of the flow phenomena, in which a flow of vehicles follows the direction of the dominating vehicle until another dominating vehicle, whose goal is interrupted because of this flow of vehicles, interrupts it and establishes a flow in its own direction.

They managed to achieve a realistic behavior resulting from agent interactions. Another result achieved was the fact that, as aggressiveness increases beyond certain percentage, the number of overtakes in general decreases. They believe the reason for this fact is that as drivers become more aggressive, not only does their tendency to overtake increase, but also the tendency to allow other vehicles to perform an overtake decreases.

- Here they developed a system that gives drivers a personality, having different characteristics among them, and also implemented an overtaking system. This allowed them to achieve a more realistic interaction between agents. However, the computational costs involved in this simulation are not very clear in their paper.

Oh [4] proposed a dynamic route search algorithm that is based on genetic algorithms and that is able to dynamically find alternative routes in unexpected events of system malfunctioning or traffic slow-downs due to accidents.

In their system, every vehicle communicates traffic data with neighboring navigation terminals in order to update its traffic database. After receiving updated information, it is constantly interested in finding better partial routes and to do so it re-evaluates its route by applying the Genetic Algorithm (GA).

In the implemented algorithm, named *Alternative Route Search* (*ARS*), chromosomes have variable length and each gene represents a specific point in the map. Their crossover operator exchanges partial chromosomes by choosing two crossover points. The mutation operator changes random sub-paths within the current path. This may result in invalid chromosomes, so they developed a repair function to fix these faulty chromosomes.

Each vehicle has two weight lists: one consisting of original weights (based on the observed average travel time) and one with newly observed weights, the updated list. By constantly checking if there are new messages concerning the commuting time of the links, it constructs its updated list ordered by timestamps and trimmed to N most recent elements due to memory capacity issues. Each vehicle collects data on commuting time and transmits it to others every time it passes them by. Rerouting is only performed at specific time intervals to avoid the deterioration of the service quality.

To experiment their algorithm they built virtual roads and three-way and four-way intersections. In addition, they used traffic control signals with proper settings. They also defined a degree of saturation (the number of vehicles divided by the maximum number of vehicles on a link) that will cause the reduction of the overall traffic speed with the increase in vehicle density. They compared the results of their system to those of simply applying the algorithm *Dijkstra* to find the shortest route and to a pseudo *Transport Protocol Experts Group* (*TPEG*) algorithm where the data on entire roads is updated every five minutes. They also took into consideration two factors: adaptability (by creating accidents) and the survivability (by disabling vehicles and by observing how each system affects the others) of each algorithm.

The results show that despite not being very efficient at the beginning of the simulation (since there is little information propagated among systems) the *ARS* obtained a better performance than other algorithms as the simulation continues in terms of route finding capability and also in terms of adaptability and survivability.

- In this paper, they developed an ingenious decentralized way to communicate information among drivers combined with genetic algorithms and achieved good results. However the authors assume access to data regarding average commute time for every link on the map and that all the data is pre-computed before the beginning of the simulation.

Sadek [5] created a discrete time, nonlinear, non-convex, dynamic assignment model that does not explicitly impose capacity constraints on the arcs' flow.

Using nonlinear regression analysis, they define a relationship between the flow (in vehicles per hour per lane) and the density (in vehicles per mile per lane). They also define the objective function and its constraints: state equations, conservation equations, initial conditions, upper bounds, and non-negativity.

In terms of the *Genetic Algorithm* (*GA*), they state that "real-world problems cannot be handled with binary representations and binary operators" and so they make use of appropriate data structures (such as floating point representations) and specific genetic operators. Regarding constraint handling, they followed a hybrid approach combining features from the following three approaches:

- creation of a feasible population by using appropriate data structures and specially designed genetic operators;
- use of a penalty function approach (fitness function results are decreased if a violation of constraints occurs);
- use of repair algorithms that restores the legality of an individual.

The developed *GA* uses a real-value vector representation for each chromosome. A potential solution would be: $u = (u_1, u_2, u_3, u_4, ..., u_{45})$, corresponding to the control variables, ($d_{0,1}$, $d_{0,3}$, $d_{0,4}$, $d_{1,1}$, $d_{1,3}$, ..., $d_{14,4}$). To initialize the population they determine the upper and lower bounds for each control variable (calculated by means of a combination of the values assigned to the control variables in previous time intervals) and then attribute a random number within these bounds to that variable. The constraints that are part of the control vector are treated as a feasible population, while the other constraints were handled using a penalty function approach. As a selection method they used the roulette wheel process. The mutation operator makes sure that the values of solution vector are always within a dynamic range. The crossover operator is a linear combination of two vectors that also checks the validity of that chromosome and repairs it in case it is illegal.

Their results show that their *GA* algorithm allows the relaxation of many of the assumptions that were needed to solve the problem analytically, that it achieves good solution quality in a reasonable amount of execution time even in large problems and that in comparison to Microsoft Excel Solver, an NLP tool, their program was much faster and achieved comparable results. However, it is necessary to run a set of experiments beforehand in order to obtain appropriate values for the control parameters. It also showed that no significant improvement was achieved beyond 300 generations.

- This paper describes an interesting use of genetic algorithms in dynamic traffic assignment, achieving good results in a reasonable amount of time. However their solution has a minor limitation, which was the simplicity of the network, given that all links were assumed to consist of two lanes per direction.

The works described in this sub-chapter allow a deeper understanding of traffic route searching and planning and show various perspectives that might be taken into consideration on how to tackle this issue.

2.2 Ant colony optimization

Dorigo [6] proposed a definition for the *Ant Colony Optimization* (*ACO*) meta-heuristic. *ACO* algorithms consists of using a population of ants to collectively solve an optimization problem and can be used to discover minimum cost paths through a certain graph, while respecting specific constraints.

Let G = (C, L) be the graph related to a certain optimization problem. C represents the set of components that exist in the graph and L is the set of connections between these components. The solutions to that optimization problem can be defined as feasible paths on the graph G. While searching for the shortest path, ants deposit pheromone in the traversed route. This leaves pheromone trails that encode a long-term memory regarding the search process. The arcs of the graph might also have a heuristic value that results from a priori information or from run-time feedback.

They define the properties that the ants of the colony should have such as a memory (to evaluate the current solution or to retrace the path backward), termination conditions and being able to move in its feasible neighborhood. The ants' probabilistic decision rule depends on its memory, the problem's constraints and also the ant-routing table (a local data structure with pheromone trails and heuristic values). There are two types of pheromone updates: online step-by-step update (updated by the ant when it moves through an arc) and online delayed update (updating the pheromone trails after finding a solution and retracing the path backwards).

Additionally there are two other processes of updating pheromone trails: daemon actions and pheromone evaporation. Daemon actions are optional and might be used to perform actions that individual ants cannot do. Pheromone evaporation consists of decreasing the intensity of the pheromone trails over time in order to avoid convergence to sub-optimal areas and to explore new areas of the graph.

In this paper they applied this *ACO* meta-heuristic to two specific problems: the *traveling salesman problem* (*TSP*) and adaptive routing in communications networks. In order to solve the *TSP* problem they applied an *Ant System* (*AS*), which consists of a number *M* of ants positioned in parallel on m cities. When all ants have completed a tour (this happens synchronously because during each iteration each ant adds a new city to the path that is being constructed) they re-trace the tour backwards and increase the intensity of the pheromone trail using their memory, which is also used to avoid visiting the same city twice. In *AS*, pheromone evaporation occurs when all the ants have completed their tours and no daemon actions are performed. The amount of pheromone deposited is proportional to the quality of the produced solutions. The probability of choosing a certain connection depends on the heuristic value of that connection. Finally there are two parameters α and β that are used in the ant-routing table and that determine the importance given to distance and pheromone values. If $\alpha = 0$, the closest cities are more likely to be selected and if on the contrary, $\beta = 0$, paths with higher pheromone values will be chosen, which may lead to the rapid emergence of a stagnation.

The problem of routing in communications networks is building and using routing tables to direct data traffic and maximize network performance. The ant-routing tables are bi-dimensional given that the node to which a data packet entering a generic node should be forwarded depends on the packet destination node. They developed the *AntNet* algorithm to solve this problem. In this algorithm the heuristic value of an arc is a function of the length of the queue of the link connecting two neighboring nodes. The other aspects are similar to those of the previous algorithm.

There are no results in this paper given that it consists of a formal description of the *Ant Colony Optimization* meta-heuristic and the class of problems to which it can be applied. In the next sub-chapters we will review some of the practical applications that this algorithm has had.

Vehicle Routing Problem

There have been several works and different approaches addressing the *Vehicle Routing Problem (VRP)*. It can be defined as the problem of minimizing time and costs of the distribution of goods performed by a fleet of vehicles, from a depot to a set of customers.

Yu [7] proposed a strategy to update pheromone levels, called *ant-weight* strategy. In the *ant-weight* strategy, the quantity of the global pheromone increment of each route is related to the total length of the solution. On the other hand, local pheromone increment in each link is based on the contribution of that link to the solution. They also make use of mutation operator that performs random customer exchanges.

Bell [8] suggested two modifications to the *ACO* in order to allow the search of the multiple routes of the *VRP*: use of local exchange and of a candidate list. The local exchange procedure uses the 2-opt heuristic (in which all possible pairwise exchanges of visited locations are tested) in order to improve individual routes. The other improvement strategy is the use of a candidate list for selecting the next location in a vehicle route. Each location is allocated to a candidate list based on the distance to all the other elements in the location set. In terms of pheromones, they simulate natural evaporation through reducing the amount of pheromone on all visited arcs (local updating). Global updating consists on adding pheromone to all of the arcs included in the best route solution. Negulescu [9] present a somewhat similar idea by introducing an *Elitist Ant System (EAS)* in which each ant that finds a better solution has the chance to deposit more pheromone.

Donati [10] addressed the *Time Dependent Vehicle Routing Problem* by suggesting new time dependent local search procedures (such as Customer Relocation, Customer Exchange or Branch Exchange) and the use of two Ant colony systems (*ACS*): *ACS-TIME* – a colony that has the objective of minimizing the total length of the solution; *ACS-VEI* – a colony that attempts to find a feasible solution with a lower number of tours than the best solution found so far. They also claim that, in order to attain a faster construction of the solution, it is useful to introduce a set of neighbors for each customer given that in an optimized solution there will never be trips between distant locations.

Gambardella [11] made use of two ant colonies: one colony minimizes the number of vehicles and the other minimizes the traveled distances. The cooperation between colonies is executed by the exchange of information through pheromone updating. This approach makes the vehicle routing problem closer to the traditional traveling salesman problem.

Zabala [12] proposed the use of the *Ant Colony System with a Local Search* algorithm to solve the *Vehicle Routing Problem with Capacity and Time Windows*, in order to minimize the cost (traveled distance) of the delivery routes. In the beginning, ants are placed in the central depot and start moving to unvisited nodes by applying exploitation and exploration, without violating the capacity restriction and time windows. Global and local pheromone updates are performed along the tours. Choosing the next state is determined by a function that depends on pheromone levels, distance and also two random variables that determine the relative importance of pheromone vs. the distance and exploitation vs. exploration. To improve the solutions, a local search procedure is implemented using the CROSS-exchange operator (that interchanges consecutive customer segments between two different routes) after all the solutions were generated. The solutions that are within a percentage above the best solution found, form part of the Local Search process. They also used the Variance Analysis (*ANOVA*) statistical test (which is used to study the relationship between a dependent variable and independent variables) in order to reach more reliable conclusions. In terms of parameters, they came to the conclusion that the best results are obtained when the pheromone level and the distance between nodes have the same level of importance, there is a medium global pheromone evaporation level and there is low local evaporation during the construction of the routes. Also, when finalizing the

construction of the routes, a higher evaporation level should be applied in order to avoid being trapped in local optimums.

These works are focused on a problem somewhat different from the one which we tackle in this paper but their solutions use interesting modifications to the *ACO* algorithm and allowed for a better understanding of the Ant Colony mechanisms and how important the tuning of parameters is in order to achieve a greater performance.

Traffic Signal Timings

The *ACO* algorithm has also been used to optimize traffic signal timings. Bullnheimer [13] makes use of *ACO* to manage signal setting parameters like number of phases, cycle length or effective green times. The results achieved by He [14] show that ACA is a simple and feasible method for signal timing optimization problems.

Once again, these papers focus on a different problem from the one that we will tackle but still it demonstrates the wide range of usefulness of the *ACO* algorithm.

Traffic Prediction

Claes [15] used a combination of the *ACO* algorithm with link travel time predictions (based on current traffic situations, historic data and real-time information provided by vehicles) to find routes that reduce travel times. Also they present a new parameter, *heuristic importance*, that balances heuristic and pheromone values. The introduced algorithm does not require global pheromone updates and assumes that pheromones from different vehicles do not interact. Instead, they only use local updates performed by exploration or primer ants. To create solutions each primer ant is sent out over each route before exploration ants and will locally increase the pheromone level of all edges. This pheromone increase will guide the exploration ants to the statistically optimal solution.

Ando [16] proposed a traffic prediction method that employs a pheromone mechanism and in which each car is regarded as a social insect that deposits multi-semantics of pheromone on the basis of sensed traffic information. They use the idea of alarm pheromone: a leading bee informs the subsequent bees of any danger using alarm pheromone so that other bees can avoid it. They suggest the combination of three different pheromone flavors: a *Basic Traffic Pheromone* in which speed represents congestion rate; a *Braking Pheromone* in which the deposited amount depends on the number of times its brakes are applied; a *Distance Pheromone* that unlike the previous two this acts as a pheromone of attraction and that informs following vehicles about the possibility of a decrease in congestion. This pheromone flavor however assumes that vehicles are "equipped with a millimeter-wave radar". As transition functions they make use of a common evaporation mechanism (pheromone levels decreases over time) and implemented a new mechanism to represent pheromone propagation that represents the characteristic of pheromone trails spreading throughout the surrounding environment. The results of their experiments (which used real traffic data) confirm the applicability of their method to short-term traffic prediction. However, their implementations use an Intelligent Transportation Systems (ITS) infrastructure called the probe-car system that only exists in Japan.

Our aim is to create a cooperative system that could be used in real-time. Claes [15] assumed access to historic data in order to make travel time predictions. Also, their proposed use of primer and exploration ants means that for each route, the path will be traversed twice. Finally, their system is not cooperative given that pheromones from different vehicles don't interact and consequently the other vehicles in the network do not benefit from the information brought back by that particular ant. Ando [16] propose a cooperative system that makes short-term predictions of traffic. There are some similarities between their concept and ours but we do not

intend to actually predict traffic. Traffic prediction is a very complex process given the instability of traffic: even a short-term prediction can be rendered useless if, per example, an accident occurs. So we decided that our focus was mainly dealing with real time information.

Other approaches

Another known problem is the *Railroad Blocking Problem (RBP),* which is addressed by Yue [17], which consists in minimizing "the total operating costs of delivering the traffic on the railway network while satisfying the resource and capacity constraints at the stations and the priority constraints for shipments". In this paper they apply the *ACO* algorithm in a similar fashion to how they would solve the *TSP* problem.

Another application is to create routing algorithms for mobile ad-hoc networks (MANET) which are a collection of mobile nodes that communicate over radio. Gunes [18] propose a new *ACO* algorithm called *Ant colony based Routing Algorithm* (ARA) that consists of three phases: the *route discovery phase* (use of a forward ant and a backward ant to create new routes), the *route maintenance phase* (responsible for route improvement during communication) and the *route failure handling phase* (routing failures are very common in mobile ad-hoc networks).

Another cooperative *ACO* approached is explored by Claes [19]. The *ACO* procedure allows the knowledge of previous found solutions to be embedded into the graph. Other ants then use this knowledge to guide them. This indirect means of communication and coordination is called *stigmergy* and is explored in that paper. The most desirable routes are those that lead to locations that are near their destination. To achieve this, locations are grouped based on the region they are in. However, maintaining the additional data needed for the region specific information presents an overhead compared to the original algorithm.

Garro [20] proposed evolving some parameters of the *ACO* algorithm through a genetic algorithm (*ACO-GA*). They introduce a new transition rule that overcomes the limitation of knowing the distance between two cities that is based on the fact that real ants choose a path solely based in the levels of pheromone. So, they use a *GAS* to evolve the parameters α, β and γ from the transition rule. In terms of the genetic algorithm they make use of a fitness function and a crossover operator. The fitness function is based on two kinds of ants (workers and explorers). A worker ant reaches its destination and is carrying food. The best ant worker is that one that has more food. An explorer ant does not possess food and is searching for its destination. The crossover operator randomly combines the α, β and γ parameters of the best workers to generate new offspring.

Robinson [21] explored Pharaoh's ant colony that comprises two types of trail pheromones: *attractive* and *repellent*. Experiments have previously shown that Pharaoh's ants use both types of pheromone. Their results show that with low traffic flow, pheromone decay overwhelms pheromone deposition indicating that small colonies of Pharaoh's ants cannot establish organized foraging trails. On the other hand, their model also predicts that above a certain threshold, increasing the ant flow will not increase foraging success.

These papers offer some interesting concepts such as worker and explorer ants, forward and backward ants and the use of attractive and repellent pheromones.

2.3 Drivers' Personality

As previously stated, current state-of-art simulation frameworks either do not model the driver's particular behavior or have a somewhat simplistic approach regarding its modeling. The following papers study some of the most relevant features needed in order to reproduce a more realistic driver personality.

The work of Vaa [24] is focused on the lack of understanding of human cognition in the current state of the art and the need for a deeper understanding of risk compensation. Older theories affirm that drivers have a "target level of risk". However, the author of this paper believes that this concept does not grasp the varied dynamics of thinking and feeling and should not be regarded as a number, but as a feeling. So it should be replaced with "target feeling" concept. In conclusion they state that by combining risk monitoring and target feeling *«the development of driver behaviour models can be put back "on the right track"*».

Simulating driver behavior

Ehlert [25] proposed a model based on reactive driving agents that can control a simulated vehicle and perform tactical-level driving. They utilized the *SHIVA (Simulated Highways for Intelligent Vehicle Algorithms)* simulator that models highway traffic. The developed agents combine traditional and reactive methods to execute their tasks but the emphasis is on the latter given that the response time is important. For every agent, sensor information is stored in the memory and models a temporary representation of the world. Each agent follows behavior rules that range from road following to respecting traffic lights and to performing a car-following behavior. All the behavior rules are influenced by the subsequent behavior parameters: speed, gap acceptance and rate of acceleration or deceleration. They experimented with a careful driver (with a low preferred speed, reasonably large gap acceptance, and a low preferred rate of deceleration) and a young aggressive driver in order to show that their driving agents exhibits human-like driving behavior and are capable of modeling different driving styles.

Demir [26] suggested a model to create a realistic urban traffic environment with hazardous situations in order to allow novice drivers to practice in a realistic environment. The tool they used was the *TRAFIKENT* driving simulator, which is used for driver training. They implemented different driving styles to create categories of urban drivers (e.g. private car, taxi, bus driver; slow, normal or fast driver) and for each of those drivers they implemented a behavior model that consists of two abstraction layers: *Decision Making Layer* (tactical level tasks such as determining right of way or lane changing) and *Decision Implementation* (operational level tasks such as car following or speed adaptation). They have also implemented a mechanism to simulate driver errors and violations such as following too closely (tailgating) or mistakes in yielding right of way. They present results that validate their behavioral model being able to emulate various driving styles for different categories of drivers.

Aggressiveness

Tasca [27] performed a review of existing literature on aggressive driving and suggests that *«a more precise definition of aggressive driving would focus on deliberate and willful driving behaviors that while not intended to physically harm another road user shows disregard for their safety and well-being.»* and that such behaviors *«are motivated by impatience, annoyance, hostility and/or an attempt to save time»*. They state that in attitudes and behaviors the gender effects are negligible but there are substantial age-related differences. The conclusions they present are that the following factors appear to influence the likelihood of aggressive driving behavior: being young, male, in a traffic situation that confers anonymity, generally disposed to sensation-seeking, in an angry mood (likely due to events

unrelated to traffic situation), having the belief that one possesses superior driving skills and finally, unexpected traffic congestions.

Laagland [28] describes how aggressive driver behavior can be modeled. They acknowledge that in current driver behavior models there is an important factor missing: emotion. The most influential emotion is aggression which can be defined with this formal description: *«A driving behaviour is aggressive if it is deliberate, likely to increase the risk of collision and is motivated by impatience, annoyance, hostility and/or an attempt to save time».* They present a series of aggressive behaviors (cutting, tailgating, etc) and categories that contribute to aggressive driving: situational and/or environmental conditions, personality or dispositional factors and demographic variables. Also, according to a study they state that driving in rush-hour traffic did not correlate with driver aggression and that aggressive driving only occurred if the congestion was unexpected. They propose that the aggressiveness of vehicles can be represented by attributing weights for personality parameters (stressful drivers, high and low aggression drivers etc), and by varying the age factors and the anonymity levels.

Compliance with traffic guidance

Dia [29] intended to study the individual driver behavior under the influence of real-time traffic information. In order to do that they used the data from a behavioral survey of drivers, conducted on a congested commuting corridor, to define for each individual driver a set of preferences, perceptions, goals and personal characteristics. Using a *Belief Desire Intention (BDI)* agent framework and microscopic traffic simulation model (*Quadstone*), they developed cognitive agents that possess a mental state composed by the following mental elements: *beliefs* (representation of current state of the agent's internal and external world), *capabilities* (executing actions), *commitments* (agreement to attempt a particular action at a particular time if the necessary pre-conditions are verified) and *behavioral rules* (which match the set of possible responses against the current environment). In this model, each driver is assigned aggressiveness, awareness, gender, age and familiarity with network.

Gao [30] explored the driver's route choice behavior under guidance information with a combination of *decision field theory* (*DFT*) and *Bayesian* theory and developed a model that describes a driver's propensity to comply with received guidance information. They state that in human's decision-making process there is a threshold parameter that regulates the trade-off between the decision-making speed and quality (cautious drivers tend to use higher thresholds and impetuous drivers use lower values - leading to shorter deliberation times that result in insufficient data processing). They also state that route criteria can attribute more importance to total distance or total time required to complete that route. In conclusion, they suggest that the following factors critically affect drivers' response to guidance information: the confidence level of guidance information, travel experience, inherent route preference, decision-making speed/quality and route choice criteria.

2.4 Traffic simulation

Simulators are part of the category of analytical tools that are used to analyze various types of data, such as traffic flows, financial transactions or pathogens spreading disease through a population. They make it possible to assess the effects of various changes to the environment without altering the real world making a cheap and safe way to predict the effectiveness of the proposed modifications.

The sort of simulation in which we are interested in this project is the traffic microscopic simulation that represents the process of creating a model of a certain network, observing and executing interactions between the agents of the system, which as previously referred, are an

autonomous entity with individual characteristics, goals and decision making mechanisms that can greatly vary within a population.

Chen and Cheng [22] analyzed a wide range of agent-based traffic simulations systems and divided them into five different categories:

- agent-based traffic control and management system architecture and platforms;
- agent-based systems for roadway transportation;
- agent-based systems for air-traffic control and management;
- agent-based systems for railway transportation;
- multi-agent traffic modeling and simulation.

Given the characteristics of this project, the adequate category is the multi-agent traffic modeling and simulation. In this paper they refer two open source agent-based traffic simulators:

- *Multi-Agent Transport Simulation Toolkit* (*MATSIM*), a toolbox for the implementation of large-scale agent-based transport simulations and is composed of several individual modules that can be combined or used stand-alone. It allows for demand-modeling, traffic flow simulation and to iteratively run simulations.

- *Simulation of Urban Mobility* (*SUMO*), a portable microscopic road traffic-simulation package that offers the possibility to simulate how a given traffic demand moves through large road networks.

MATSIM

During the previous year, this tool was analyzed and it was concluded that it did not fulfill the needs of Cosmo given that both its daily plan based simulation and its iterative demand optimization process added unnecessary complexity. In this project we need a simulator that allows us to improve individual trips, where agents can make decisions in real time and not through a process of trial and error.

SUMO

Simulation of Urban MObility (*SUMO*) is an open source, portable, microscopic road traffic simulator conceived to deal with large road networks and it was developed by employees of the Institute of Transportation Systems at the German Aerospace Center.

It was studied in detail by Krajzewicz [23] who describe it as multi-modal, meaning that there can be various types of transportation vehicles besides passenger cars. It is space continuous and time discrete in which every time step has the duration of one second. Also, in each step the vehicle's speed is adapted to the speed of the leading vehicle in a way that yields to a collision-free system. It also offers support for traffic lights implementation.

In conclusion, this review shows that traffic simulation is an area that has been studied thoroughly and that can be approached by multiple perspectives. The presented works illustrate different approaches on how to tackle the traffic route searching and planning subject. Also, the *ACO* sub-chapter offers some interesting variations of this algorithm. We believe that by combining some of the different aspects of these works we can create a distributed algorithm that pro-actively tries to disperse traffic in real-time city, while avoiding privacy concerns. Regarding driver personality, there is still much room for improvement concerning the modeling of drivers' particular behavior.

Chapter 3
Cosmo Platform Development

Having completed an initial literature survey it was time to initiate the development process. In this section we will describe the different stages of the development process of the Cosmo platform.

3.1 Exploration of SUMO

The first step of the process was to understand the traffic simulator *SUMO* (on which the COSMO platform was built on) and its structure. There are three main modules in the SUMO package:

- *SUMO*, which reads the input information, processes the simulation, gathers results and produces output files. It also has an optional graphical interface called SUMO-GUI;
- *NETCONVERT*, a tool to simplify the creation of *SUMO* networks that can be made, using this module, from a list of edges. It reads the input data, computes the input for SUMO and writes the results into various output formats, such as XML, CSV or VISUM-networks. It is also responsible for creating traffic light phases;
- *DUAROUTER*, a command line application that, given the departure time, origin and destination, computes the routes through the network itself using the *Dijkstra* routing algorithm.

As input data, *SUMO* needs three main files: "rou.xml", "nod.xml" and "edg.xml". The file "rou.xml" represents the traffic demand and includes information about all the agents involved in this simulation and their characteristics (departing time, maximum acceleration, maximum deceleration, driving skill, vehicle length and color) and route (list of edges).

In terms of outputs, there are different types available such as:

- a raw output that contains all the edges and all the lanes along with the vehicles driving on them for every time step, which results in a considerable large amount of data;

- log-files created by simulated detectors (a simulation of induct loops with the ability to compute the flow, average velocity on the lane, among other values) are written using the CSV format. This data can be aggregated for specified time intervals which may be configured by the user.

This tool also possesses a *Traffic Control Interface* (*TraCI*) that uses a TCP based client/server architecture to provide access to *SUMO*, which acts as a server that is started with additional command-line options. After connecting to a specific TCP port it is possible to send messages that may contain instructions to influence the behavior of the system or may receive information about the ongoing simulation.

Finally, this simulator offers a way to measure pollutant emission based on the *Handbook of Emission Factors for Road Transport (HBEFA)* database. According to *HBEFA*'s website[31], it was "*originally developed on behalf of the Environmental Protection Agencies of Germany, Switzerland and Austria*" and is now also supported by Sweden, Norway, France and the *JRC (European Research Center of the European Commission)*. It provides emission factors per traffic activity, i.e., it offers a way of measuring CO_2 emissions and fuel consumption, among other pollutant factors, for various vehicle categories (such as, passenger cars, light duty vehicles, heavy duty vehicles, buses, coaches and motorcycles - in this work, only passenger cars will be taken into account), being suitable for a wide variety of traffic situations.

3.2 Understanding basic traffic definitions

We will now present a few definitions of some basic traffic concepts.

According to the Free Dictionary [32], traffic congestion "is a condition on road networks that occurs as use increases, and is characterized by slower speeds, longer trip times, and increased vehicular queuing. (…) When traffic demand is great enough that the interaction between vehicles slows the speed of the traffic stream, congestion is incurred. As demand approaches the capacity of a road (or of the intersections along the road), extreme traffic congestion sets in. When vehicles are fully stopped for periods of time, this is colloquially known as a traffic jam." Another important definition is traffic flow: "Flow rates are collected directly through point measurements, and by definition require measurement over time."

In order to measure traffic congestion there are two alternatives: by calculating density or by calculating occupancy. Also according to [32], density is "the average number of vehicles that occupy one mile or one kilometer of road space, expressed in vehicles per mile or per kilometer." According to the Transportation Research Board Special Report on Traffic Flow Theory [33] "occupancy can be measured only over a short section (…), with presence detectors, and does not make sense over a long section.".

In one of the algorithms developed in the previous year, as a measure of an edge's congestion we considered the occupancy value returned by *SUMO* for each edge. According to the *SUMO* user documentation [34], this value consists of the percentage of time a detector (generated by *SUMO*) was occupied by a vehicle. However, according to [33] "density can be measured only along a length. If only point measurements are available, density needs to be calculated, either from occupancy or from speed and flow. Speeds within a lane are relatively constant during uncongested flow. Hence the estimation of density from occupancy measurements is probably reasonable during those traffic conditions, but not during congested conditions. ".

In conclusion, according to these sources, the occupancy factor alone will not be a realistic measure of congestion. This topic will be further discussed in section 3.6 - Implementation of Ant Colony Optimization algorithm.

3.3 Understanding current platform

There are five main modules in the platform developed in the previous year:

- *cosmoAgent*: contains all the information and actions relative to each individual agent;
- *cosmoConstants*: contains global constants that are used by the other modules;
- *cosmoController*: is the main class. It is responsible for parsing the network file, creating the network and the population, communicating with SUMO and saving the simulation results to file;
- *cosmoNetwork*: contains the relevant information of the network in question: the layout, the distances and time step occupancies;
- *cosmoPopulation*: serves as an intermediate between the *Controller* and the *Agents* and is responsible for creating the agents, parsing the route files and attributing routes to the agents and for communicating.

As previously said, the communication with *SUMO* is made through its *Traffic Control Interface* (*TraCI*). The following figure shows the communication process:

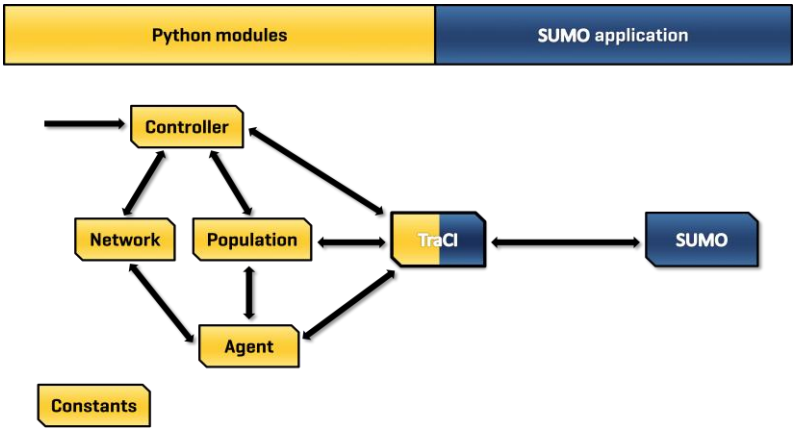

Figure 1 - Communication Process

Regarding algorithms, two relevant algorithms were developed:

- *Shortest Path*: using the *Dijkstra* algorithm, the shortest path was calculated using solely distance values;
- *Shortest Time*: using a variation of the *Dijkstra* algorithm, the shortest path was calculated using distance values but also takes into account the occupancy values of the previous time step in each edge.

Another two algorithms were developed: an *Error Insertion* algorithm that randomly introduced error into the weights of the edges and a *Road Blocking* algorithm which made edges, that were occupied beyond a pre-defined percentage, unavailable to drivers.

The *Error Insertion* mechanism combined the one developed in the *Shortest Time* algorithm with a random tweaking of the occupancy parameters. It used the actual occupancy status but inserted a random value into that percentage in order to distribute traffic more evenly. This would make traffic in non-congested zones more chaotic given that routes will be constantly changing even if there was no significant change in traffic and would lead drivers to traverse longer distances and not the optimal paths. This would create dissatisfaction among drivers and would eventually lead them to no longer trust this system and therefore quit using it.

The *Road Blocking* mechanism was not a realistic approach given that the basis for this mechanism was prohibiting vehicles to choose a determined road that in real life could not be physically achieved: roads could not be blocked by the developed system. Also, it could make routes impossible if there was only one possible path and it was blocked.

In conclusion, after analyzing the concept of the algorithms and results obtained in the last year, we decided not to further pursue the development of these algorithms.

Finally, a mechanism to simulate accidents was developed in the previous year but it was not taken into account in the work developed. Nevertheless, it may be included in further developments in order to portray a more realistic traffic environment.

3.4 First Testing Phase

One of the flaws of the work done in the previous year was the lack of extensive testing. Due to this, it was concluded that the system and the developed algorithms did not achieve significant improvement over the default behavior.

Our experiments, executed with larger traffic loads (five and ten thousand vehicles), show that the developed system achieves indeed a considerable improvement over the default behavior.

One of the major difficulties during this initial testing phase of the simulation process was its execution time. Through a few optimizations, like caching path values and minimizing route calculation by only allowing re-calculation near intersections, an improvement was achieved but, given that a continuous communication with *SUMO* is not very scalable, it was a minor improvement.

Testing with heavier traffic loads was performed and the results obtained with the *Shortest Time* were very good regarding the duration of the trips. However, the values of mean distance traveled were lower than those of the default behavior which made them incompatible with the other results, given that this algorithm will make vehicles avoid congested edges and choose longer but less occupied edges, likely resulting in longer paths. The results obtained with the *Shortest Path* were poorer but this "mean distance traveled phenomena" did not occur.

3.5 Second Testing Phase

After some research, we came to the conclusion that the reason for the faulty results was the fact that *SUMO* only accounted for the length of the vehicle's final route state. Given that the network status is constantly updated during a simulation, each vehicle's route might be re-calculated several times. In the *Shortest Time* algorithm, each vehicle's route is likely to be altered, which greatly diminishes the mean distance traveled results, given that the final route will correspond to the partial route defined by the last re-calculation.

For example, initially, a vehicle has the following route: X1, X2, X3, X4, X5. But, when this vehicle arrives in location X3, his route re-calculation tells him that the best path to get to X5 is: X3, X6, X5. The vehicle's route is then updated in *SUMO* with only the X3, X6, X5, causing loss of information about its actual traveled route.

In order to obtain the actual results of the traveled distance, the traveled route is stored in each agent and at the end of the simulation the traveled distance values are calculated according to that route and not to the values returned by *SUMO*.

3.6 Implementation of Ant Colony Optimization algorithm

The main problem with the *Shortest Time* algorithm is the fact that it assumes a very simplistic model and is not accurate in predicting traffic congestion. In order to more effectively spread vehicles throughout the network, the idea to implement an *Ant Colony Optimization* (*ACO*) algorithm emerged.

This well-known algorithm is used to find optimal paths in a graph: each ant lays down pheromone trails and other ants are likely to follow the trail (instead of travelling at random) reinforcing it. Over time, pheromone trails starts to evaporate, thus reducing its attractive force.

But our goal is not to find the shortest distance path to a destination but a shortest time path. So our implementation will reverse the *ACO* logic and will make pheromone trails repel incoming vehicles.

This algorithm has an important advantage over the *Shortest Time* algorithm: it also takes into account the speed at which the vehicles move through a certain edge. Given that, in each time step vehicles deposit pheromone in the current edge, if they traverse it fast they will deposit less pheromone. On the other hand, if there is congestion in that edge it will take much longer to traverse, greatly increasing the pheromone trail. The occupancy factor in itself is not a realistic indication of the existence of congestion given that an edge can have an occupancy of 50%, but

if all the vehicles in it are moving at a high speed then there is little or no congestion. The inverted ACO combines occupancy and speed factors, giving a better portray of the real congestion status of the network.

3.7 Initial ACO Testing Phase

After having completed the implementation of the first inverted *ACO* algorithm prototype, in order to adjust the pheromone parameters (global and local evaporation and pheromone deposit) accordingly, experimentation was needed.

The initial phase of this testing process was composed basically by trial and error experiments with the referred parameters. Afterwards, there was experimentation with only global evaporation and the results were not very satisfactory.

To improve performance, local evaporation was added to the algorithm. This local evaporation consists on each vehicle withdrawing pheromone after leaving a certain edge. This withdrawal corresponds to the minimum amount that a vehicle would deposit if traveling through that edge at full speed. This way, if there is no congestion in a given edge, the vehicle will be able to go through it rapidly and will erase it's presence after leaving the edge, therefore updating pheromones to a more realistic level. On the other hand, if there is already congestion and the vehicle is forced to slowly travel through that edge, the amount deposited will be far superior to the minimum amount deposit, leaving a trail of its presence there. In short, here is a piece of pseudocode that illustrates this local evaporation process:

```
if agent.willLeaveEdge(edge):
    edge.withdrawPheromone(edge.getMinimumTravelTime()*PHEROMONE_AMOUNT)
else:
    edge.depositPheromone(PHEROMONE_AMOUNT)
```

Another important factor in evaluating current congestion is understanding the pheromone variation trend: if the pheromone level in one edge decreased in the last N time steps, than it is likely that the number of cars in that edge has diminished and consequently so has the congestion; on the other hand, an edge might currently have a low pheromone level although the tendency of the last N time steps might indicate an increasing demand for that edge. In order to take that information into account, we also added a pheromone variation history mechanism that adds a short history of pheromone levels in each edge, allowing a better understanding of the variation of traffic volume and a more efficient evaluation of traffic congestion.

3.8 Real Map ACO Testing Phase

The previous testing phase refers to a proof of concept, in which we ran experiments in two different simple traffic networks, *Lattice* and *Radial and Ring* maps (which will be presented in the next chapter). Prompted by these results, we intended to conduct a series of experiments using a real city map and realistic traffic information.

In order to be able to run tests in a real city, we firstly needed to obtain the map of a city, compatible with *SUMO*. So, we began searching for suitable solutions. The *SUMO* module *Netconvert* offers a way to import digital road networks from various sources (such as *VISUM*, *OpenDrive* or *OpenStreetMap* (*OMS*)) and creates networks usable by the other *SUMO* modules.

The search for *VISUM* or *OpenDrive* networks was unsuccessful given that only basic sample networks were found. On the other hand, to obtain maps in the *OMS* format we just needed to access the *OMS* website and export the network. So, we opted for the *OMS* format. However,

the conversion process of an *OMS* map to a *SUMO* network presents its challenges given that the conversion is sometimes faulty. We will return to this subject in the next sections.

Given the results of this search process, there were three available solutions: the city of Cologne, the city of Eichstäett, or importing a network. The city of Cologne offered thorough but massive network and route information, making it impossible to experiment with it in a reasonable amount of time. The city of Eichstäett offered a more lightweight network to experiment with. Finally, through *OpenStreetMap* we created a network of the city of Coimbra. The usage of the Eichstäett and Coimbra maps will be explored in the following sections.

Eichstäett map

According to the user documentation [35], the available map of the german city of Eichstäett (already in a *SUMO* format) was converted from the *OpenStreetMap* format and all unnecessary details filtered out. They also state that *"All roads were verified to have the right highway type, speed limit and one-way attribute. All traffic lights of cars (not all pedestrian lights) were also verified."* so it was ready to be used in SUMO simulations.

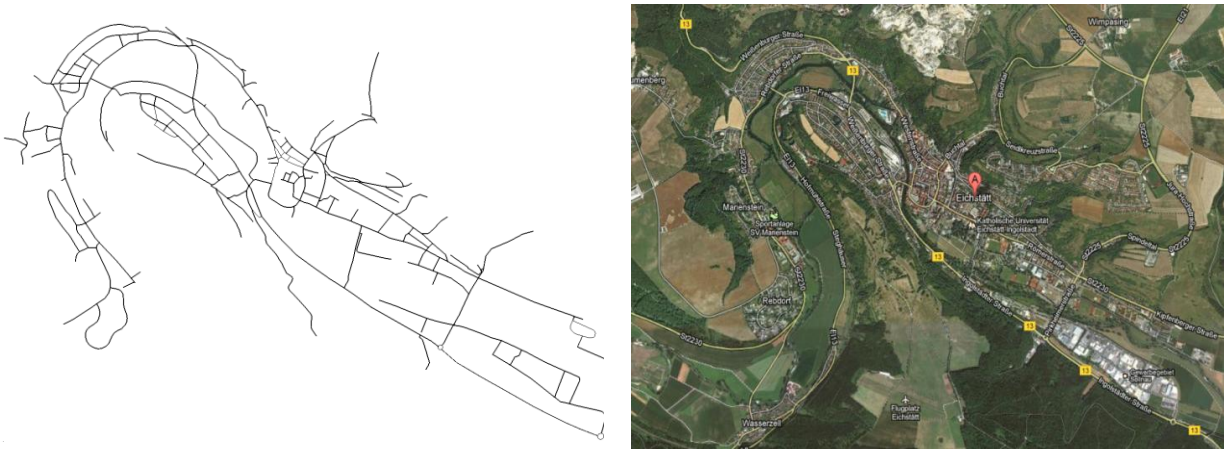

Figure 2 - Eichstäett map

A set of routes, of approximately 10.000 drivers, was created and experimented with. However these routes present a serious fault: they were randomly generated and did not offer a realistic portray of the traffic in that city. It's useless to experiment with a realistic map if no realistic traffic will traverse it so it was necessary to obtain real traffic information. Fortunately, an Origin-Destination matrix regarding the city of Coimbra was available.

Coimbra Map

In order to be able to run tests in Coimbra, we firstly needed to obtain a map of the city, compatible with *SUMO*. The *SUMO* module *Netconvert* offers a way to import digital road networks from various sources (such as *VISUM* or *OpenStreetMap*) and creates networks usable by the other *SUMO* modules.

As previously referred, no *VISUM* or *OpenDrive* networks of Coimbra were found so we opted for the *OpenStreetMap* format. By accessing the *OpenStreetMap* website we can search for the desired city and easily export that data into an *XML* file.

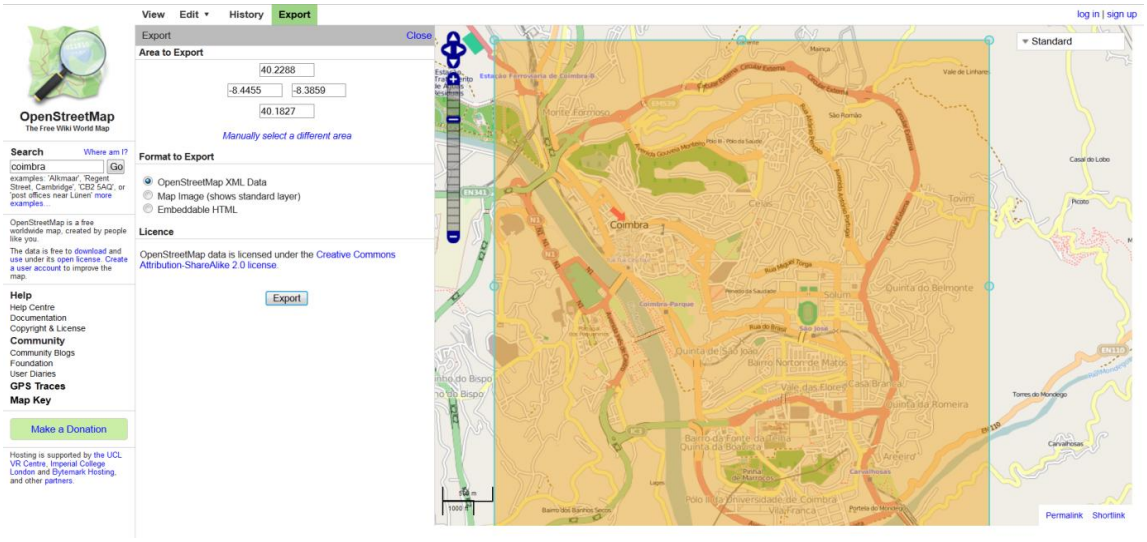

Figure 3 - OpenStreetMap website

With the resulting *XML* file (with an .osm file extension) we can convert it into a *SUMO* network (with the .net.xml file extension) with the following command:

netconvert --osm map.osm -o map.net.xml

After this process, we experimented with the generated network in order to evaluate the outcome of this conversion. We found out that there were some errors in the generated *SUMO* network such as the conversion of pedestrian roads into vehicle roads. In order to filter out the pedestrian roads, and also to correct some other minor errors, the *JOSM* tool was used, which is an *OpenStreetMap* editor written in *Java*. The current version of *SUMO* offers the option of filtering out certain types of roads while converting a map, but at the time this option was unavailable. Another problem that we found, and corrected, was that the *Netconvert* module, by default, adds a turnaround possibility for almost every edge, making the network map unrealistic and somewhat confusing:

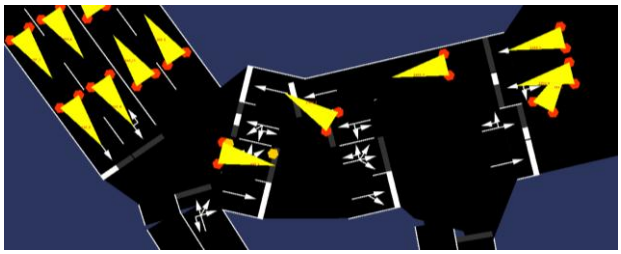

Figure 4 - Edges with unrealistic turnarounds

Then we created a route set and using the *SUMO GUI* we identified the zones responsible for creating the more relevant bottlenecks. These areas contained several problems that range from an incorrect number of lanes to faulty connections between edges.

Having corrected the previously referred faults, we believed the map to be ready for the testing process.

Coimbra Routes

With a corrected version of the map, it was time to generate the correspondent traffic demand. To complement the real city map we needed realistic traffic information. The work by Dr. Álvaro Seco and Nuno Norte Pinto [36] provided this information in the shape of an *Origin-Destination* (*OD*) matrix regarding the city of Coimbra. An *OD* matrix consists of a table that

matches trips' origins and destinations and also displays the number of trips going from each origin to each destination.

In order to create the traffic demand based on this *OD* matrix, we first needed to analyze its data. This matrix was divided into several zones as can be seen in Figure 5 and these zones referred to inner and outer city traffic.

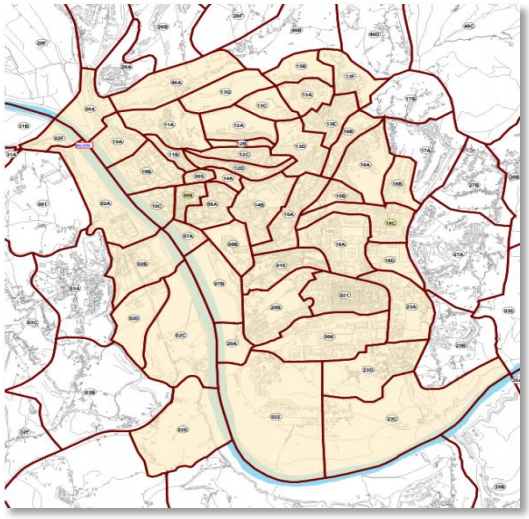

Figure 5 - Coimbra OD matrix zones

There were numerous zones referred to outer city traffic so, given that in this project we were mostly interested in studying traffic within a city, we decided to group these several outer city zones into 7 zones, which match the existing city entrances: North (IC2), Northeast, East, Southeast, Southwest (Europa Bridge), West (Santa Clara Bridge) and Northwest (Açude Bridge).

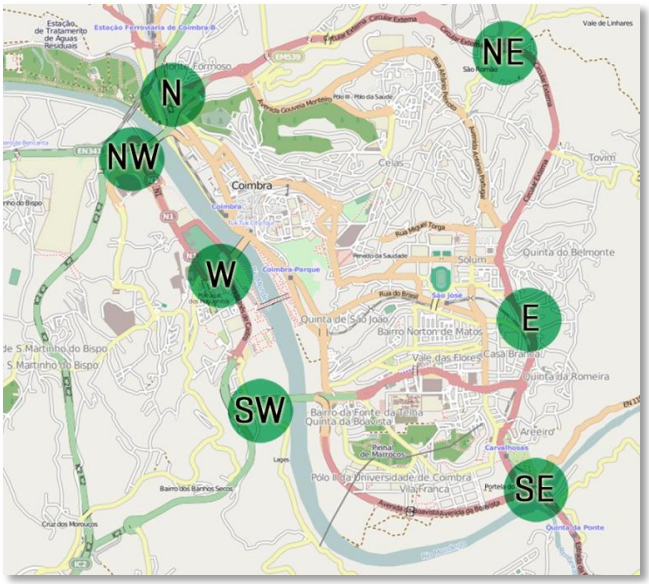

Figure 6 - Coimbra city entrances

The data in this matrix comprises 60.000 trips and corresponds to the morning period (7h30 to 10h30) with the duration of three hours. To make the testing process feasible, given the considerable map size, we decided to use 10.000 trips with an insertion period of one hour, i.e., vehicles are inserted at uniform time intervals during one hour. To make simulation environment more realistic, we inserted 2.000 vehicles at the beginning of the simulation (in the first time step) while the rest of the vehicles were inserted during the simulation.

In SUMO, route definitions are as follows:

```
<vehicle id = "0" type = "type1" depart = "0" color = "1,0,0">
    <route edges = "id_beginning id_middle id_end"/>
</vehicle>
```

This means that each trip is defined as a collection of the id of the edges that comprise that trip. The main difficulty was establishing the match between the *OD* matrix's zones and the ids present in the *SUMO* network. Firstly we experimented with two id values per zones but as a result the entrance of cars in the network was very slow. This is due to the fact that *SUMO* inserts vehicles in a pre-determined position in each edge and does not insert another vehicle unless there is space for that insertion. This means that the simulator would have to wait until there was enough room to insert each new vehicle, greatly increasing the simulation time. It also means that it would not be possible to achieve the desired effect given that this fact significantly delayed the insertion of vehicles and making it impossible for some vehicles to be inserted within the previously defined time slots.

In order to overcome this limitation, more id values were added to the trip generation process. It was not possible to define a constant number of ids per zone given that the number of roads in each zone was not uniform. However an average amount of approximately 7 id values was defined per zone.

This id search process lead to significant time consumption given that most zones defined in the OD matrix do not exactly match street names and so these id values were manually defined. However, it allowed for a much smoother insertion of vehicles.

Coimbra Map Problems

After experimenting with the *OD* routes we concluded that the results were still not satisfactory. An experiment with 10.000 vehicles lasted 27.000 seconds, i.e., approximately 7 hours, which is a very large and unrealistic amount of time.

We came to the conclusion that this was due to problems with the simulator and also with the utilized network. So, we started to study how to further improve the quality of the network in order to obtain more realistic results. The following figures show some of the problems we encountered:

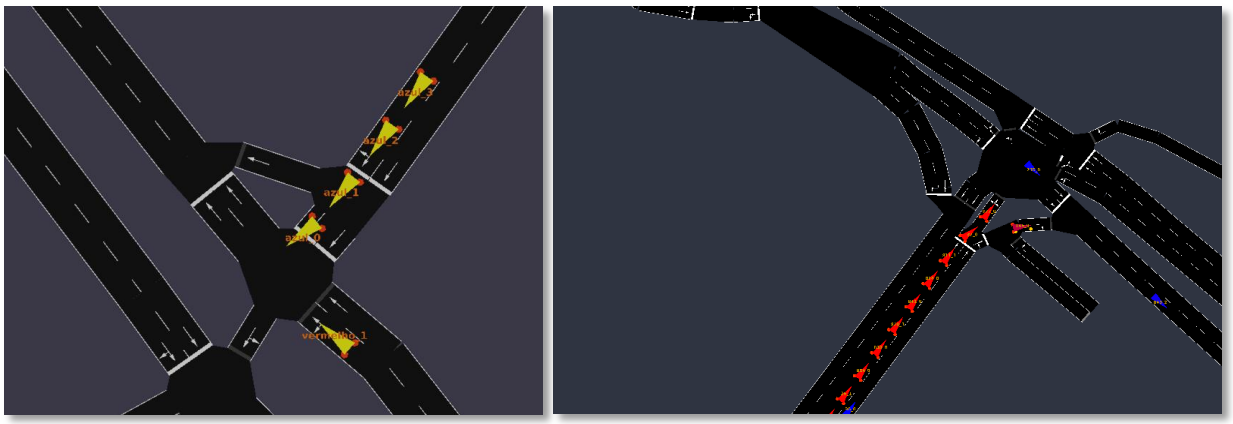

Figure 7 - Hesitation situations

In the leftmost image of Figure 7 we show an example of halted vehicles, each queue apparently blocking each other. In the rightmost image we see the red vehicles giving total priority to the blue vehicles, waiting until there was no blue vehicle left in order to advance. It appears that *SUMO* has some issues when it comes to detecting whether or not the leader is occupying the junction and therefore blocking the passage. In Figure 8 we can see that, although the vehicle on

the right lane (highlighted in green) wants to go forward and there does not seem to be any obstacle in front of him, it does not advance. However it considers that the other vehicle is in its way and therefore will only move once the other vehicle has also moved, causing an unrealistic and unnecessary queue.

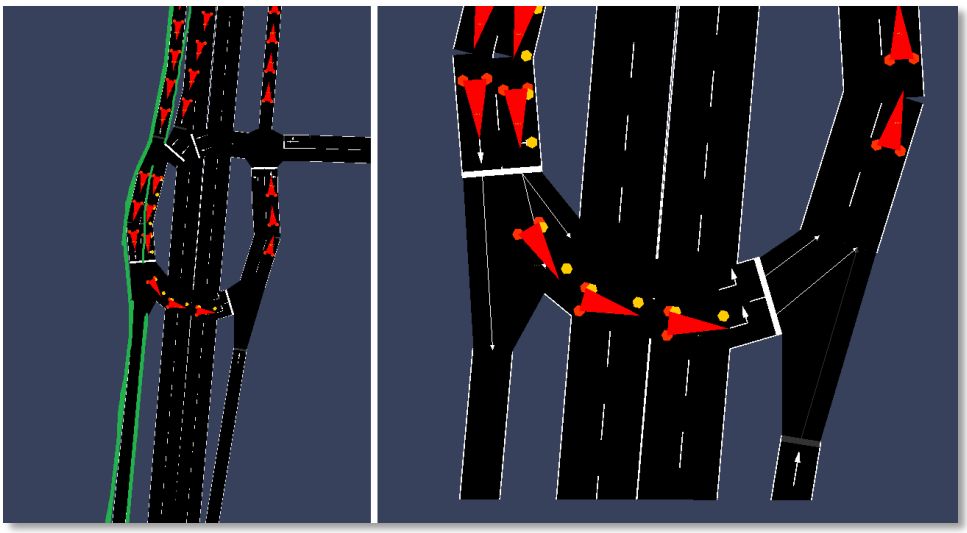

Figure 8 - Junction problems

We noticed that some of these zones lacked traffic lights - that were somehow lost in the conversion. Therefore, with the intent of reducing the "hesitation" that appears to occur in Figure 7, we added traffic lights in the areas that were most affected by congestion: Portagem, Avenida Fernão de Magalhães and Casa do Sal. The addition of traffic lights appears to improve the ordering of traffic, reducing the hesitation phenomena. However, while solving one problem it creates another - due to the addition of traffic lights in some junctions, vehicles are often stopped at intersections:

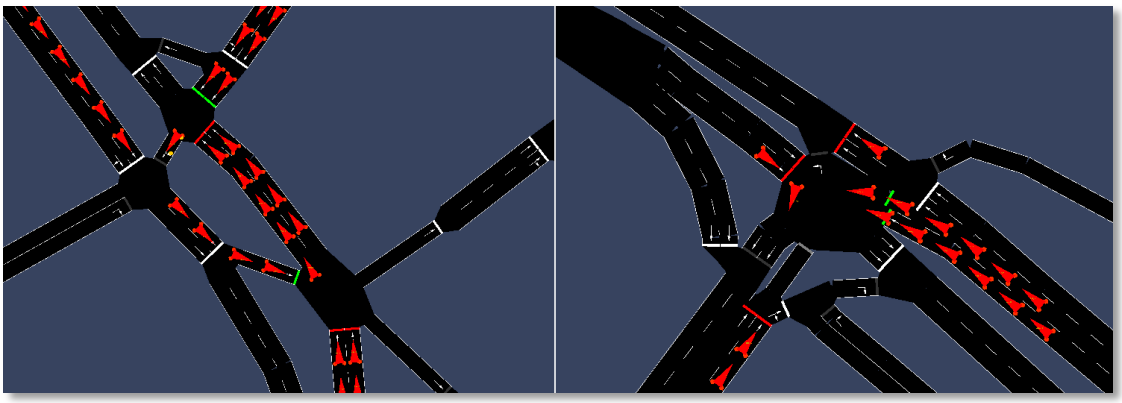

Figure 9 - Traffic light problems

SUMO is a collision-free simulator and therefore security is paramount. So its vehicles require a large gap to competing vehicles in order to decide to move forward as a few experimentations demonstrated. But this highly defensive attitude does not justify all the problems that we encountered. Some situations, like vehicles only using one lane when two or more lanes are available, may be justified with map errors and the consequent flaws in the way vehicles perceive the network.

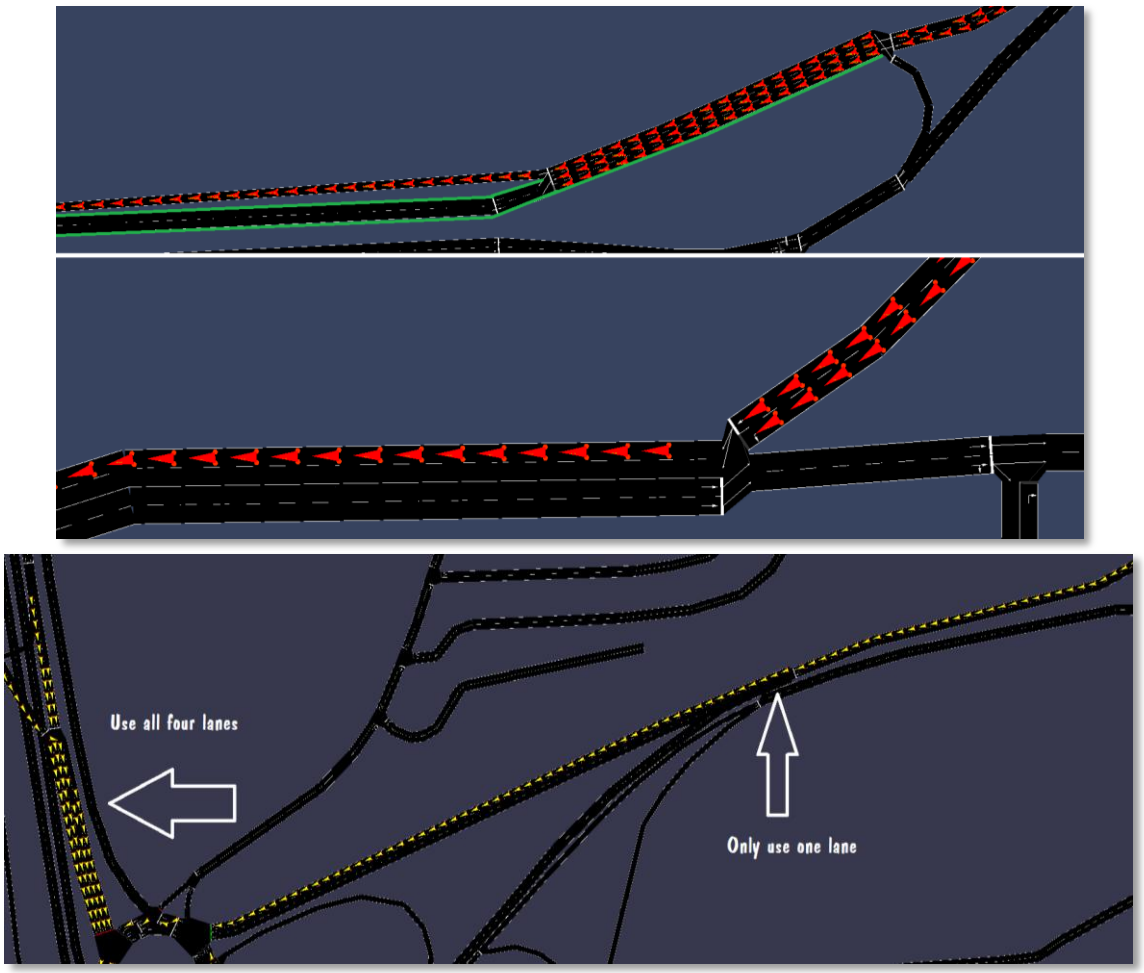

Figure 10 - Inadequate use of lanes

Finally, we tried to communicate with *SUMO's* developers in order to understand the cause of these situations. Firstly we informed them of one of those hesitation situations which was acknowledged by them as a probable bug. However *SUMO* is an open-source project and the feedback was not immediate. So, while we did not receive feedback, in order to understand the nature of the previously referred errors we analyzed the simulator source code. However, the code is extensive (several thousands of code lines) so we narrowed down the search to the part that affects edge/lane priority and defined the right of way. In a section of the code relative to checking *"traffic on next junctions"* we found an error - a conditional expression in which the code is equal to both the if and else clauses. We informed the *SUMO* developers about this bug in the hope that it would solve the hesitation phenomena.

In conclusion, we consider there to be three main problems with the current state of the simulator:

- *Map conversion*: leads to inadequate use of lanes;
- *Definition of the right of way*: in proximity to junctions it requires a large space between vehicles in order to advance;
- *Entering junctions*: vehicles sometimes get stuck on a junction if the target lane is full, causing widespread jams and thus increasing the average travel time.

As we are dealing with heavy traffic, these problems create a domino effect that generates enormous queues. The encountered problems have been reported and some were acknowledged as bugs, which they will try to solve in the next version of the simulator.

Given that certain issues, such as the occurrence of vehicles being stopped at intersections, obstructing the passage of other vehicles, appear to be more common in the most recent versions of *SUMO*, we opted for using a previous version of *SUMO* in which that bug is not as significant.

3.9 Personality features

The *SUMO* simulator already has the following aspects hard-coded: stopping at traffic lights, switching lanes, overtaking and applying traffic rules. So these driving parameters will not be tampered with. Also, *SUMO* uses a collision-free model so no traffic accidents will be considered in this work.

The implementation of the following features intended to create a more realistic portrayal of the diversity of driver behaviors in a given city. These features can be divided into two groups: the stubbornness feature assumes the existence of an information service that provides drivers with information about traffic congestion; the aggressiveness, distraction and irregularity features do not make that assumption.

Distraction

This feature represents situations in which a person is not experienced/familiar with the environment or is dealing with a high cognitive load (talking on a cell phone for example), therefore being distracted, and consequently takes one or many wrong turns along its proposed route.

The familiarity of a driver with the network in which he is traveling in obviously affects the amount of wrong turns that one can perform along a certain path. A person who is very familiar with a certain route is much less prone to make a mistake along the way than a person who is traveling through that path for the first time.

According to the *World Health Organization* (*WHO*) [37] a percentage between 1% and 7% of drivers have been observed using mobile phones in several European countries while driving. They also report that in the United Kingdom *"45% of drivers reported text messaging while driving"* and that in the United States *"27% of American adults report having sent or read text messages while driving"*. One of the obvious implications of this are driving accidents: *WHO* [37] refers that *"in Spain, an estimated 37% of road traffic crashes in 2008 were related to driver distraction"*. In this paper, as previously referred, we will not tackle the driving accidents problem but this statistic shows the importance of driver distraction. The influence of distraction, combined with familiarity with the network, will be studied in a route selection point of view: how distraction levels contribute to making mistakes along a certain route.

Stubbornness

This trait refers to the unwillingness of an agent to accept the proposed route. The rejection of this proposal might be caused by having little confidence on the system or a driver's belief to be able to better understand the current state of traffic and therefore to being capable of defining a more suitable route.

Gao [30] suggests that aggressive drivers were less prone to act in accordance with received guidance information. However, no significant proof of this correlation was found so we will assume that stubbornness and aggression are independent.

Irregularity

The concept of *Driver imperfection* refers to the inability of a driver to maintain a constant velocity, causing fluctuations in speed that affect the vehicles that are following him. The car-following model used in *SUMO* was developed by Krauß [38]. In it, each driver computes a safe velocity, in order to being able to brake fast enough and not colliding with the leader. Also, there is a "randomization step" in which a random amount that is uniformly distributed between 0 and *σ*accel* (where *σ* refers to *driver imperfection* and *accel* refers to a vehicle's acceleration) is subtracted to that safe velocity.

Aggressiveness

Dukes [39] claims that aggressive driving is a growing concern. They state that "64% of Americans believed that drivers were driving much less courteously and safely than five years ago." thus being an important aspect of driving behavior.

The implemented feature refers to the definition of various types of drivers according to age, gender and temper. The work of Tasca [27] initially suggests that "gender effects are negligible but there are substantial age-related differences". Afterwards they state that the factors that increase the probability of aggressive behavior are "*being young, male, in a traffic situation which confers anonymity, generally disposed to sensation-seeking, being in an angry mood (likely due to events unrelated to traffic situation), the belief that one possesses superior driving skills and finally unexpected traffic congestions.*". From these stated parameters, we will not consider the anonymity factor nor the occurrence of unexpected traffic congestions.

According to Wickens [40], driver aggression is greater for males (38.5%) than for females (32.9%). Younger drivers (from 18 to 34 years of age) reported the highest occurrence of perpetrated driver aggression (47.3% for females, 54.5% for males), while the oldest drivers (above 55 years of age) reported the lowest rates of driver aggression: 15.1% for females and 20.9% for males.

With the previously stated data in mind we decided to create the following driver types:

- courteous young male, courteous young female, aggressive young male, aggressive young female, courteous middle-aged male, courteous middle-aged female, aggressive middle-aged male, aggressive middle-aged female, courteous elder male, courteous elder female, aggressive elder male and aggressive elder female.

Each driver type will be assigned a specific value for minimum gap acceptance, driver's reaction time, acceleration and deceleration rates and desired speed. The minimum gap acceptance parameter allows us to simulate the tailgating behavior (following someone too closely) by defining low gap acceptance values. The tailgating phenomena is, according to Björklund [41], the driving situation that provokes most irritation.

According to Holland [42] one of the personality factors that influence driver behavior is *Locus of control* (*LOC*). Drivers with internal *LOC "perceive outcomes to be dependent on their own skill, efforts or behaviour"* that enables them to be more responsive than externally oriented drivers, which take fewer precautions to prevent road accidents. One consequence of internal *LOC* might be a more risky driving style, caused by the driver's belief in being able to avoid an accident using their own

skills. Given this information, the driver's reaction time will be used to simulate the influence of *LOC* and also to simulate elderly people's slower reaction capability. However, there does not seem to be a consensus as to what are the actual values of driver reaction time given that it depends on various factor such as gender, age, alertness or driving experience. Davis [43], Mehmood [44], McGehee [45] and Triggs [46] present different values, ranging approximately from 0.7 seconds to 2.3 seconds. Mehmood [44] also states that reaction time increases with age and that females have larger reaction times. The reaction time values used in this experiment take this data into account. Both acceleration and deceleration rates and desired speed will be a result of the aggressiveness of the driver and also of his responsiveness.

Tasca [27] considers gender effects to be almost negligible. On the other hand, Holland [42] affirms that *"women have more external LOC than men"*. Our belief regarding this matter is closer to the opinion presented by Tasca [27], so, we considered gender effects to be almost negligible by only slightly altering parameters between male and female drivers, attributing somewhat more aggressive parameters to male drivers.

Regarding age, in young drivers we defined a greater percentage of aggressive drivers, in middle-aged drivers a more balanced percentage and in elderly drivers a small percentage of aggressive drivers. Being disposed to sensation-seeking and believing to possess superior driving skills are also implicitly taken into account in the cautious/aggressive driver ratio.

SUMO's standard parameters' values are based on the work of Krauß [38] and are defined as follows:

accel="2.6" decel="4.5" minGap="2.5" maxSpeed="70" tau="1.0"

(where "*accel*" corresponds to the vehicle's acceleration, "*decel*" refers to the vehicle's acceleration, "*minGap*" corresponds to the minimum gap acceptance, "*maxSpeed*" refers to the vehicle's maximum speed and "*tau*" corresponds to a driver's reaction time).

Based on this, in order to simulate the previously referred driver types, we altered the parameters in the following manner:

type		**acceleration** (m^2/s)	**deceleration** (m^2/s)	**sigma**	**maxSpeed** (m/s)	***minGap*** *(m)*	**tau** (s)
Young	courteous male	2,5	4,5	0,5	23	*2,5*	1
	courteous female	2,4	4,4	0,5	23	*2,5*	1
	aggressive male	3,1	5,5	0,4	33	*1,2*	1
	aggressive female	3	5,4	0,4	33	*1,3*	1
Middle-aged	courteous male	2,4	4,1	0,6	21	*2,5*	1,5
	courteous female	2,3	4	0,6	21	*2,5*	1,5
	aggressive male	2,9	5	0,5	28	*1,6*	1,3
	aggressive female	2,7	4,9	0,5	28	*1,7*	1,4
Elder	courteous male	2,3	3,8	0,7	19	*2,5*	1,9
	courteous female	2,2	3,7	0,7	19	*2,5*	2
	aggressive male	2,6	4,5	0,6	25	*2*	1,7
	aggressive female	2,4	4,4	0,6	25	*2,1*	1,8

Table 1 – Aggressiveness: driver types

It should be noted that these are merely tentative values - more attention was given to the difference of parameter values between driver types than to the parameter values themselves.

However, the fact that the *minGap* parameter is only configurable in the most recent versions of *SUMO* (which currently have that previously referred bug regarding the entering of vehicles into junctions) causes widespread jams and produces unrealistic results. So, as previously stated, we opted for using a previous version of *SUMO* that does not offer the configuration of the *minGap* parameter but in which that bug is not present.

Chapter 4
Results and Discussion

In this chapter we will present and analyze the experimental results. The objectives of these experiments are to evaluate the performance of the proposed algorithms and to assess the influence of the personality parameters in a network's performance. In order to achieve those goals, the tests were executed in artificial and real life maps.

The first part of the testing process used the already existing algorithms of *Shortest Distance* and *Shortest Time* and consisted in the simulation of two sets of routes, for 5.000 and 10.000 drivers. These routes were executed in two different networks: *Lattice* and *Radial and ring*. These networks are illustrated in the following figures:

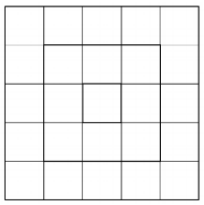

Figure 11 - Lattice network

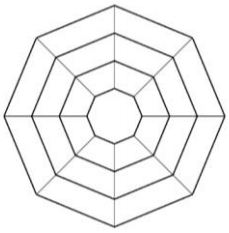

Figure 12 - Radial and Ring network

For each amount of drivers, 5.000 and 10.000, we generated 5 drivers per step/second. Also, for each of those amounts, we varied the percentage of active users of our system, testing with the following percentages: 0%, 25%, 75%. Each of these variations was simulated 10 times in order to enable averaging of the results and consequently the production of reliable data..

The second part of the testing process compared the already existing algorithm *Shortest Time* with the implemented inverted *ACO* algorithm. The simulation process is similar to that of the first part being executed with the same variations in total driver amounts and generated drivers per second in the same networks (*Lattice* and *Radial and ring*).

There are two main differences from the previous part of the testing process:

- to study the initial viability and appeal of the system, more attention was given to lower percentages of users: in order to be attractive and to convince people to adhere to it, the system should offer advantages even with low usage percentage. So we executed experiments with the following user percentages: 0%, 10%, 25%, 75% and 100% - each of these variations was simulated 30 times.

- testing with a real life map (*Coimbra* network) with real traffic data: these tests were executed only 10 times, due to the project's time restrictions.

Figure 13 - Coimbra network

The third part of the testing process is dedicated to studying the effects of the personality features and consists of experiments executed in *Lattice*, *Radial and ring* and *Coimbra* networks. The tests referred to in this last testing part were simulated 10 times.

The numerical results of all experiments are available in Appendix B.

4.1 Shortest Distance vs Shortest Time

Firstly we will compare the *Shortest Distance* (*SD*) and *Shortest Time* (*ST*) algorithms on the *Radial and Ring* and *Lattice* maps and we will analyze the results of each algorithm in terms of *average trip duration* and *average route length*.

Please note that in all the presented line charts, the dotted lines refer to average values plus and minus the standard deviation.

Radial and Ring

We will firstly present results regarding experiments with 5000 vehicles and afterwards with 10000 vehicles.

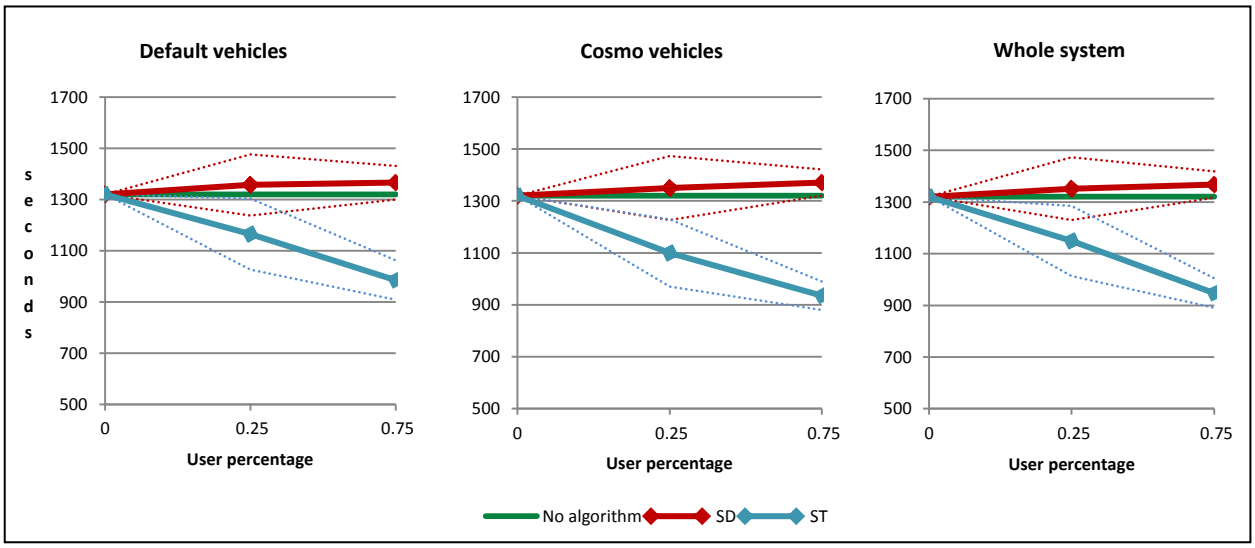

Figure 14 – Radial and Ring (5000 vehicles): average trip duration

In Figure 14 we can observe that the *SD* does not manage to reduce trip duration and actually achieves worse performance compared to the default behavior. The lack of improvement is caused by the trip generation process already attributing optimal or quasi-optimal paths. The performance deterioration can be explained by the surplus of vehicles whose optimized route goes through the center of the map, leaving the outer edges unoccupied. However, the performance did not deteriorate as much as expected given that this map's topology creates the conditions for the occurrence of congestion in the core of the map. On the other hand, *ST* algorithm is able to reduce the average duration of trips. By distributing the vehicles using the *ST* algorithm around the network it also allows the default vehicles to shorten their trip duration, greatly improving the network's performance as a whole.

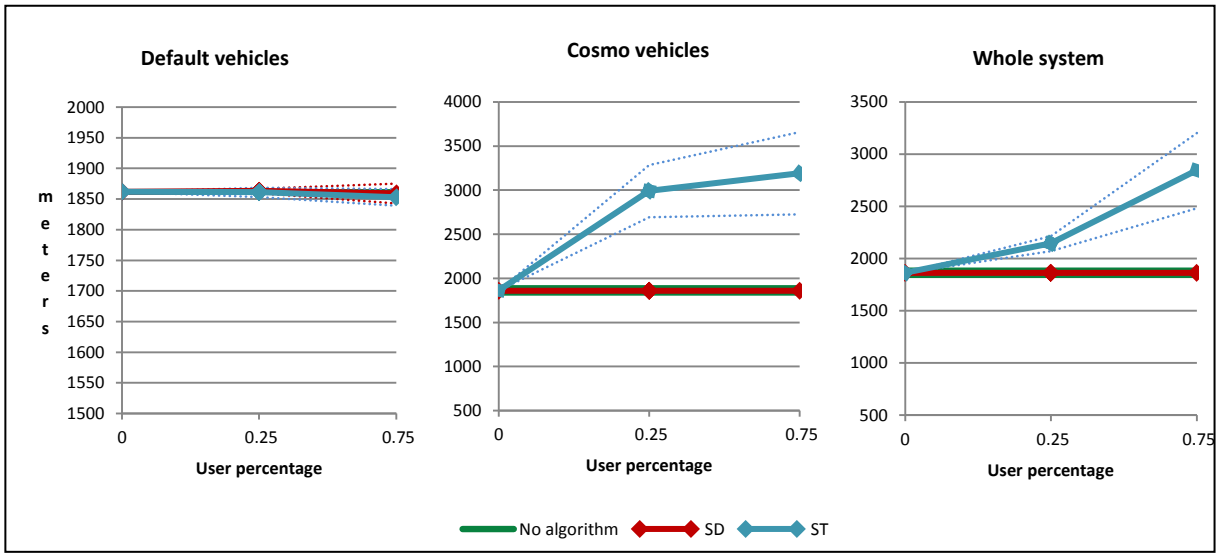

Figure 15 – Radial and Ring (5000 vehicles): effect on route length

In terms of distance (Figure 15), the amount traveled by the vehicles using the *ST* algorithm is considerably higher than of those using the *SD* algorithm. The reason for this is that the *ST* vehicles will have the tendency to avoid the shortest paths because those will be more used and consequently more congested.

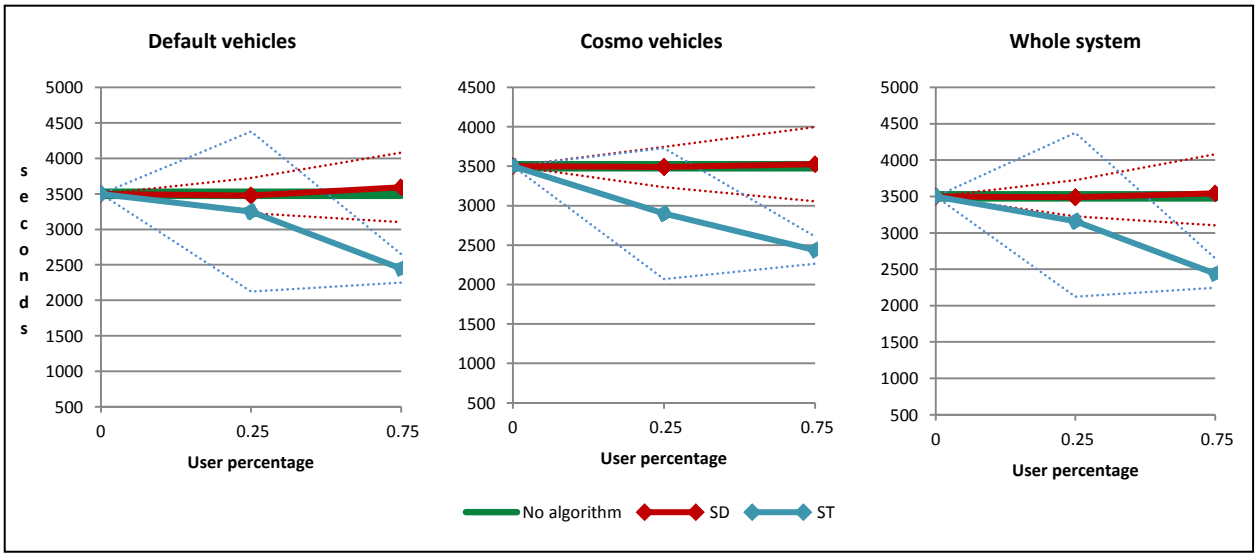

Figure 16 – Radial and Ring (10000 vehicles): effect on trip duration

In Figure 16 we observe that the differences in average trip duration are maintained with the increase in number of vehicles. Regarding traveled distance, it is roughly equivalent to the results obtained with 5.000 vehicles: vehicles using the *ST* algorithm traverse a longer path than the ones using the *SD* algorithm, as seen in Figure 17.

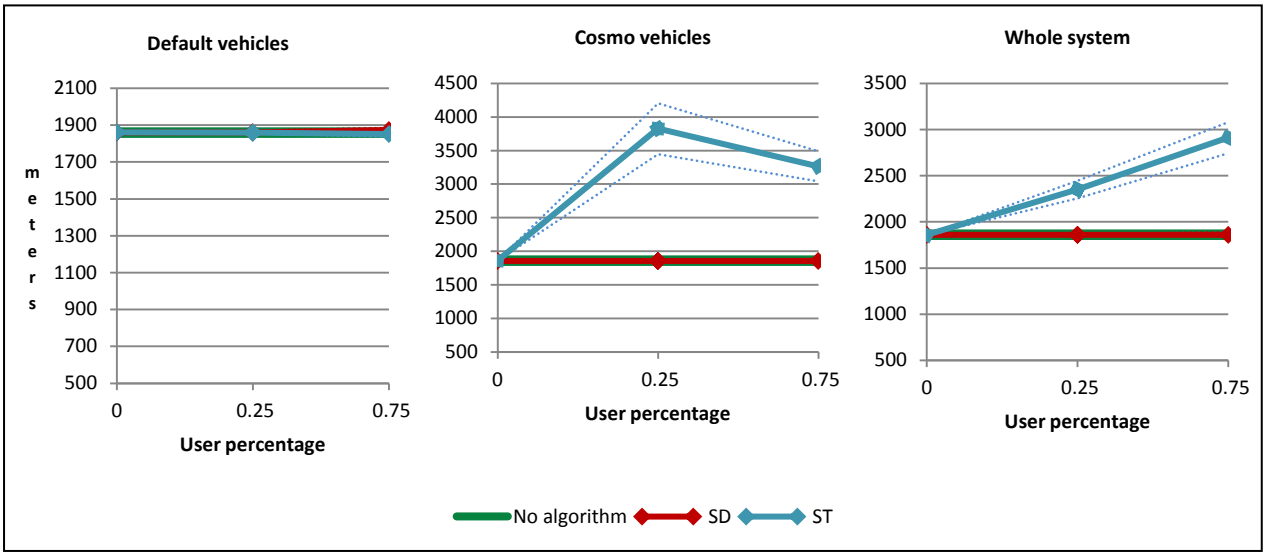

Figure 17 – Radial and Ring (10000 vehicles): effect on traveled route length

With these results, we can conclude that in this map, significant average travel duration reduction can be achieved using the *Shortest Time* algorithm, given that it distributes traffic throughout the network more evenly. We also conclude that the *Shortest Distance* is not adequate for this map's topology.

Lattice

In the previous section, the results showed that, for the *Radial and Ring* map, the *Shortest Time* algorithm distributes traffic throughout the network more evenly, achieving greater performance. We also conclude that the *Shortest Distance* is not an adequate solution for that map's topology. Now, we will present results regarding the Lattice map, which has a different topology:

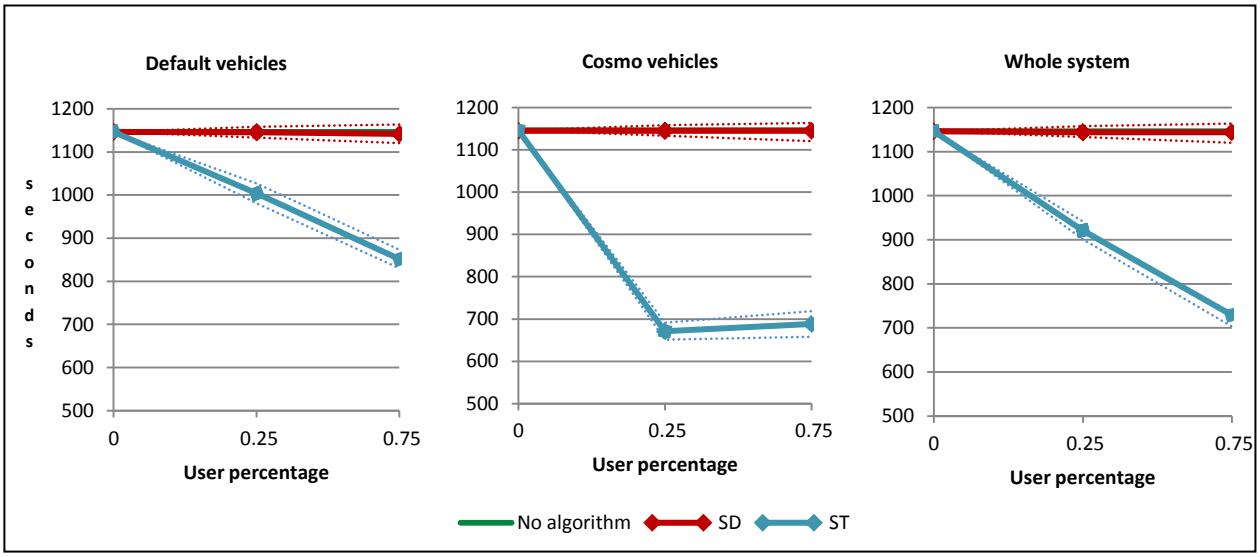

Figure 18 – Lattice (5000 vehicles): effect on trip duration

In Figure 18 we can observe that the *SD* algorithm achieves no significant reduction of the duration of trips given that there may be various paths with equal distance to a certain destination. On the other hand, *ST* algorithm manages to reduce the average duration of trips. As in the *Radial and Ring* map, this reduction also allows the default vehicles to shorten their trip duration, greatly increasing the network's performance as a whole. However, in this map, we can observe the duration of the trips of the *ST* users does not decrease as the user percentage increases. This may be due to the fact that the distribution of vehicles in the network stabilized and could not be optimized beyond that point.

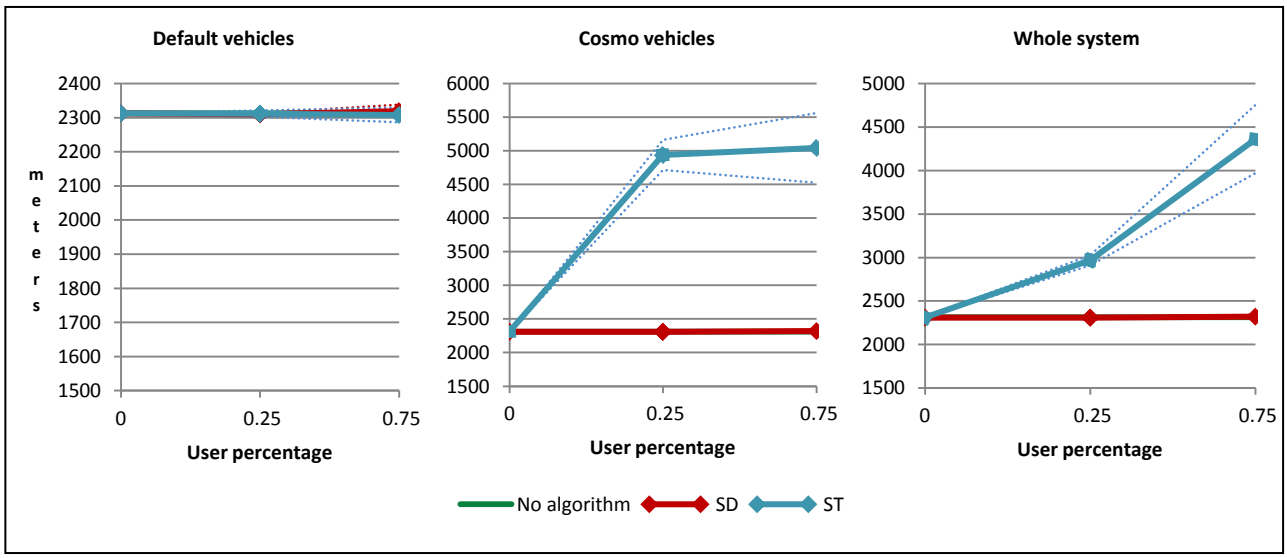

Figure 19 – Lattice (5000 vehicles): effect on traveled route length

In terms of distance (Figure 19), the results are also similar to those obtained in *Radial and Ring* map, which means that the amount traveled by the vehicles using the *ST* algorithm is considerably higher than of those using the *SD* algorithm. We can observe in the chart referent to "Cosmo vehicles", the same phenomenon that occurred in the *Radial and Ring* map. As the user percentage increases, the vehicles are more evenly distributed and are not so obliged to going through the outer paths of the map.

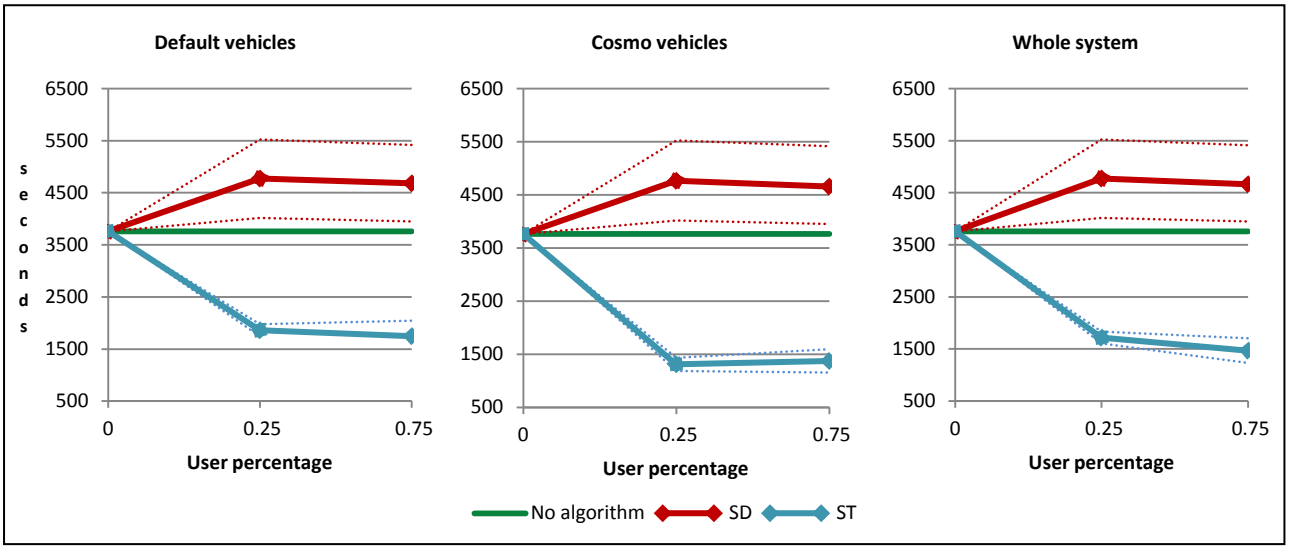

Figure 20 – Lattice (10000 vehicles): effect on trip duration

Figure 20 shows that the differences in average trip duration are accentuated with the increase in number of vehicles and that the *SD* algorithm's performance deteriorates greatly, achieving worse results than those of the default vehicles. This may be caused by attributing to various vehicles the same path, increasing congestion. Given that there may be several equally optimal paths to a destination, the *Dijkstra* will consistently choose the same path. In accordance to what we have seen in the chart referent to "Cosmo vehicles", we can observe that the duration of the trips of the *ST* users does not decrease as the user percentage increases - initially it decreases nut after reaching a 25% user percentage it tends to stagnate. This may indicate that the network is saturated and cannot be optimized beyond that point.

Regarding traveled distance, it is roughly equivalent to results obtained with 5000 vehicles, as seen below in Figure 21: with the rise of user percentage, *ST* vehicles' route length tends to stabilize.

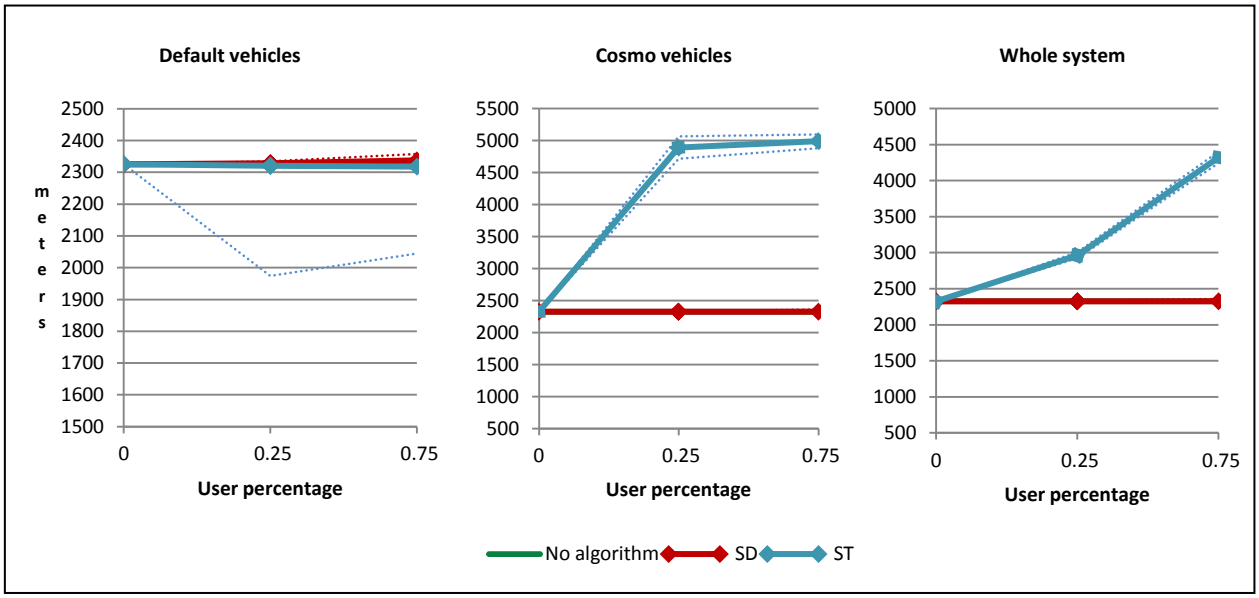

Figure 21 – Lattice (10000 vehicles): effect on traveled route length

By analyzing these results, we can conclude that in this map, significant average travel duration reduction can be achieved using the *Shortest Time* algorithm given that it distributes traffic throughout the network more evenly. We also conclude that the *Shortest Distance* is not adequate for this map's topology.

4.2 Shortest Time vs Ant Colony Optimization

As expected, *Shortest Time* (*ST*) obtained better results in the previous section. So, next we will compare the ST and *Inverted Ant Colony Optimization* (*IACO*) algorithms on the *Radial and Ring* and *Lattice* maps and we will analyze the results of each algorithm in terms of *average trip duration*, *average route length* and *pollutant emissions*.

Radial and Ring

Like in the previous section, we will firstly present results regarding experiments with 5000 vehicles and afterwards with 10000 vehicles.

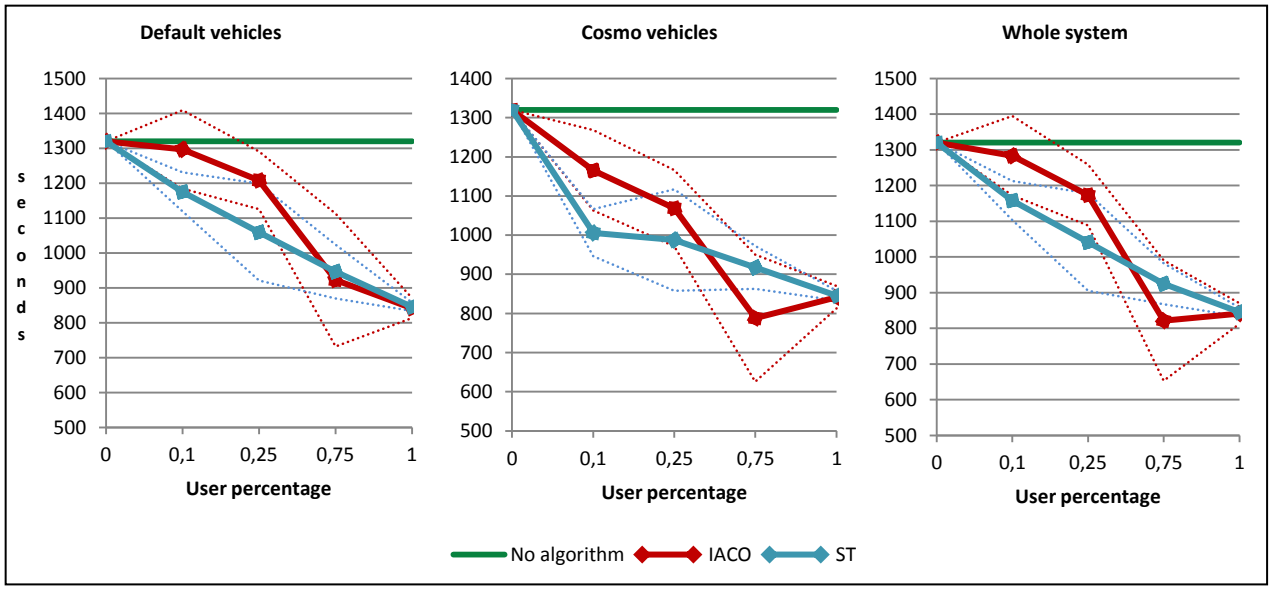

Figure 22 – Radial and Ring (5000 vehicles): effect on trip duration

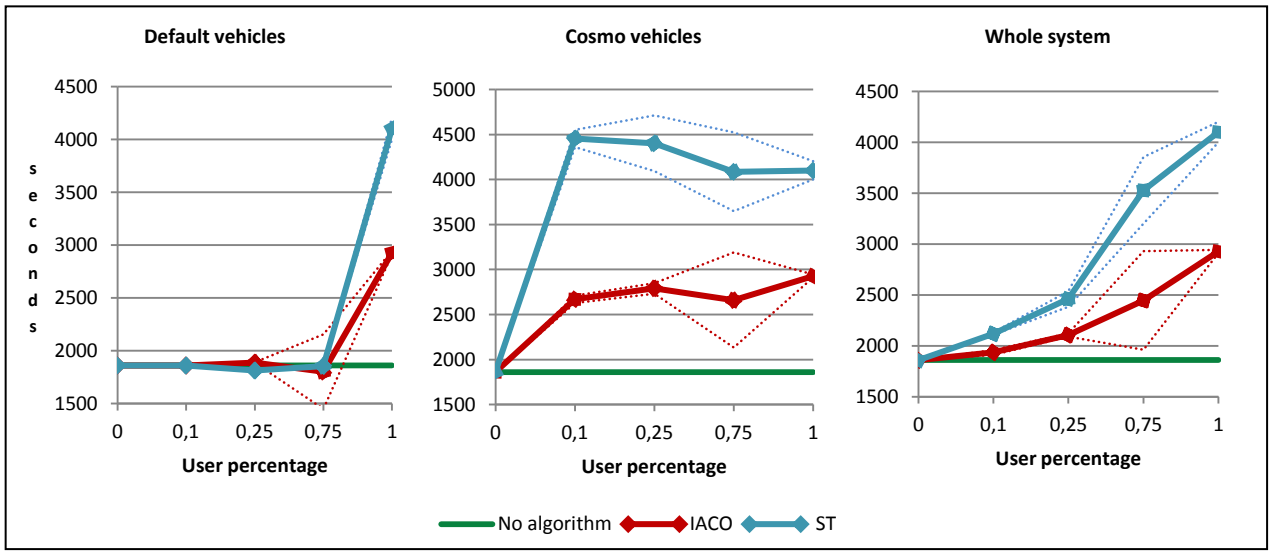

Figure 23 – Radial and Ring (5000 vehicles): effect on traveled route length

In Figures 22 and 23 we can observe that the *IACO* has a slow start in terms of trip duration performance, which is due to the lack of pheromone information that occurs with low user percentages. However, as the user percentage increases it achieves better results than the *ST* algorithm, always managing to maintain a significantly smaller route length. On the other hand,

with a 100% user percentage, *IACO's* trip duration efficiency appears to deteriorate, achieving a roughly equivalent performance to the *ST* algorithm.

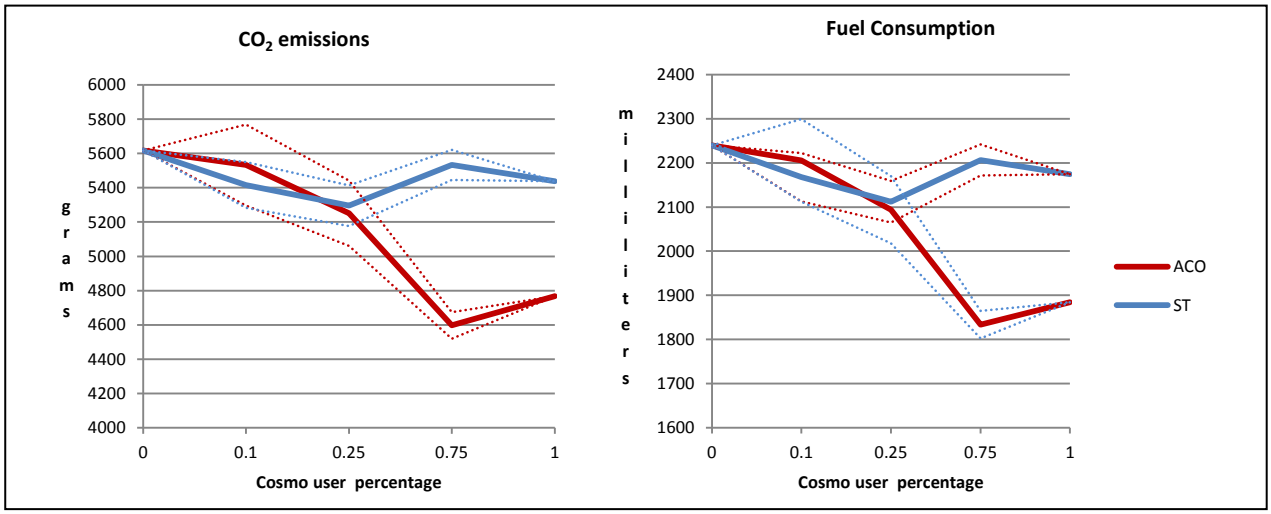

Figure 24 – Radial and Ring (5000 vehicles): CO_2 emissions and fuel consumption

Regarding pollutant emissions (Figure 24), we can observe that *IACO* attains lower fuel consumption, and consequently lower CO_2 emissions, for all tested user percentages. This also indicates that the *IACO* obtains a better equilibrium between trip duration and route length.

Given that "Fuel consumption" and "CO_2 emissions" charts are proportional, it should be noted that from now on we will only show the charts concerning the latter.

With a higher traffic load (Figures 25 and 26) *IACO* appears to improve and it achieves better performance in duration and route length, even for a 25% user percentage. This might be due to the fact that, even with a low user percentage, it detects heavy traffic in the center of the map and manages to avoid it. Once again, with very high user percentages, *IACO*'s performance seems to stagnate in terms of trip duration.

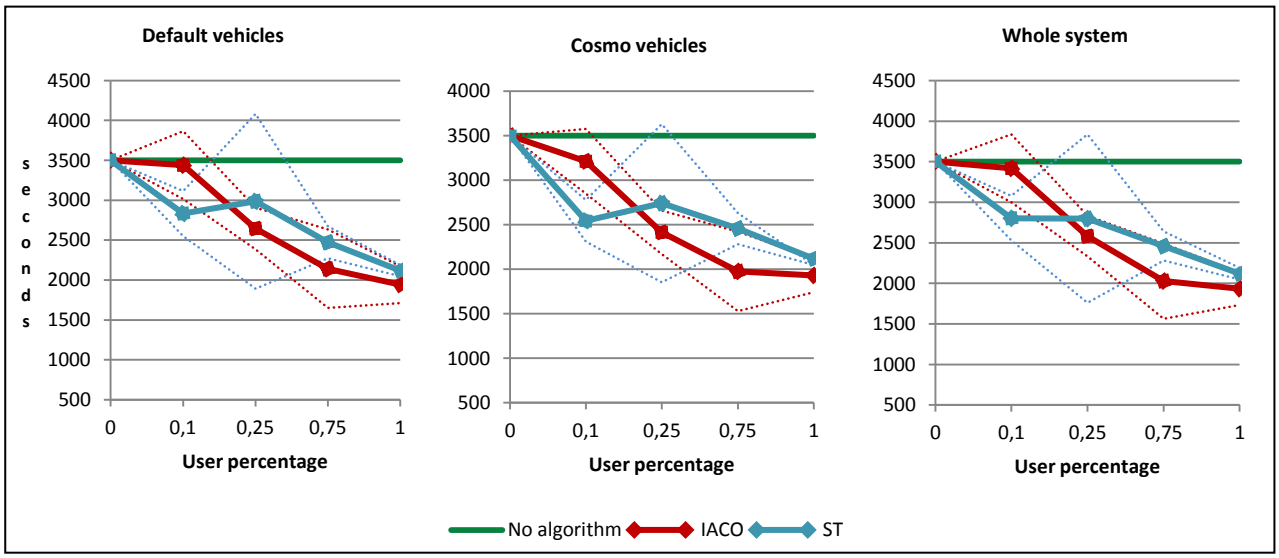

Figure 25 – Radial and Ring (10000 vehicles): effect on trip duration

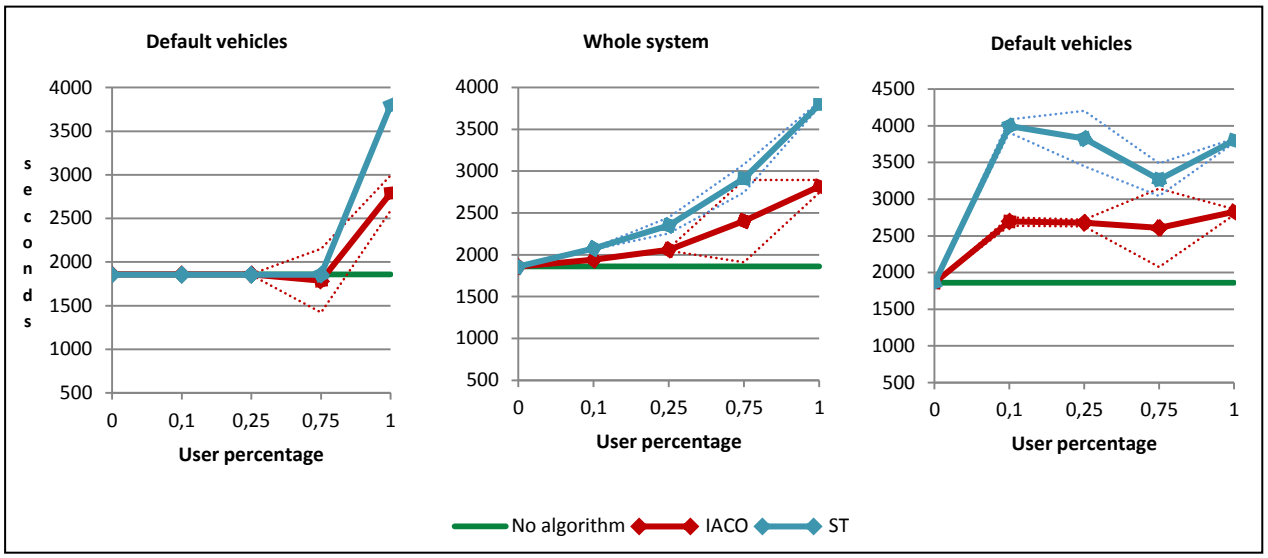

Figure 26 – Radial and Ring (10000 vehicles): effect on traveled route length

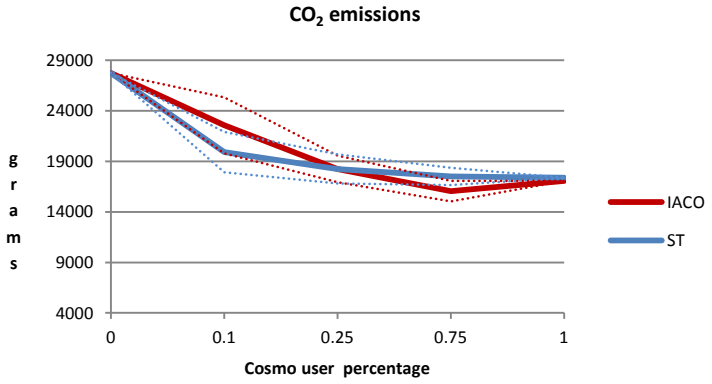

Figure 27 – Radial and Ring (10000 vehicles): CO_2 emissions

Regarding pollutant emissions (Figure 27), we can observe that for low user percentages, *IACO,* despite obtaining better performance in duration and route length, causes the emission of more CO_2 than the *ST* algorithm. However, as user percentage rises it reaches a better performance in terms of CO_2 emissions. With a high user percentage, we can also observe that the performance of both algorithms worsens, suggesting again that, for this map, the system becomes saturated for a very large percentage of users.

With these results, we can conclude that in this map, significant average travel duration reduction can be achieved using the *Inverted Ant Colony Optimization* algorithm, being an adequate alternative for this map's topology.

Lattice

In the previous section the results showed that for the *Radial and Ring* map the *Inverted Ant Colony Optimization* algorithm is an adequate option for that map's topology, achieving a greater average travel duration reduction than the *ST* algorithm. Now, we will present results regarding the *Lattice* map, which has a different topology:

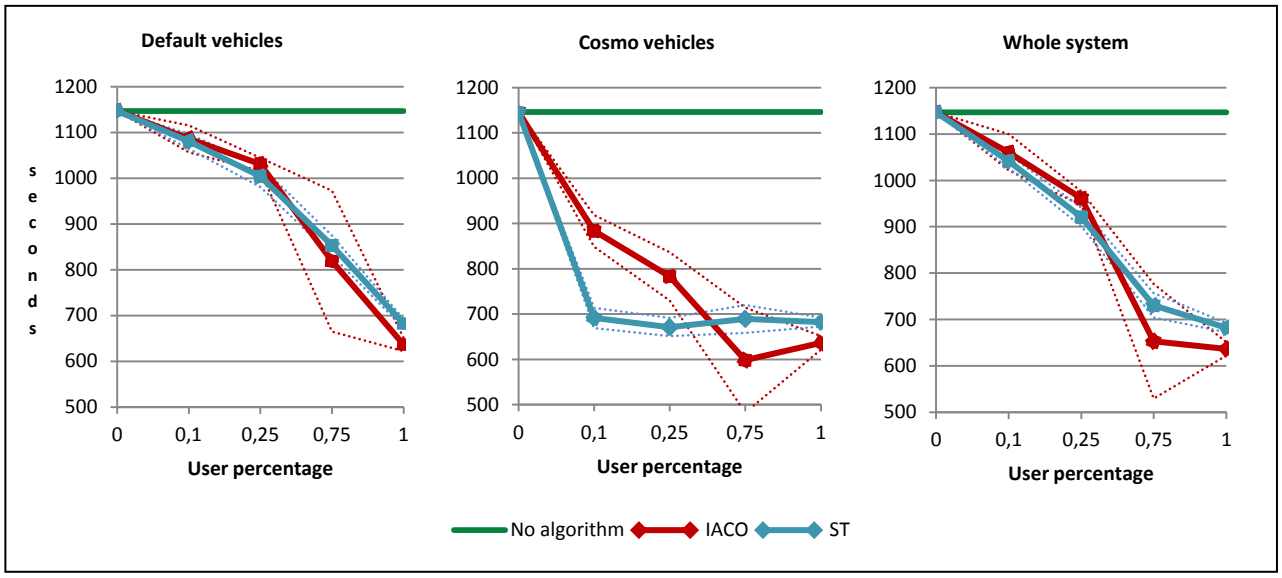

Figure 28 – Lattice (5000 vehicles): effect on trip duration

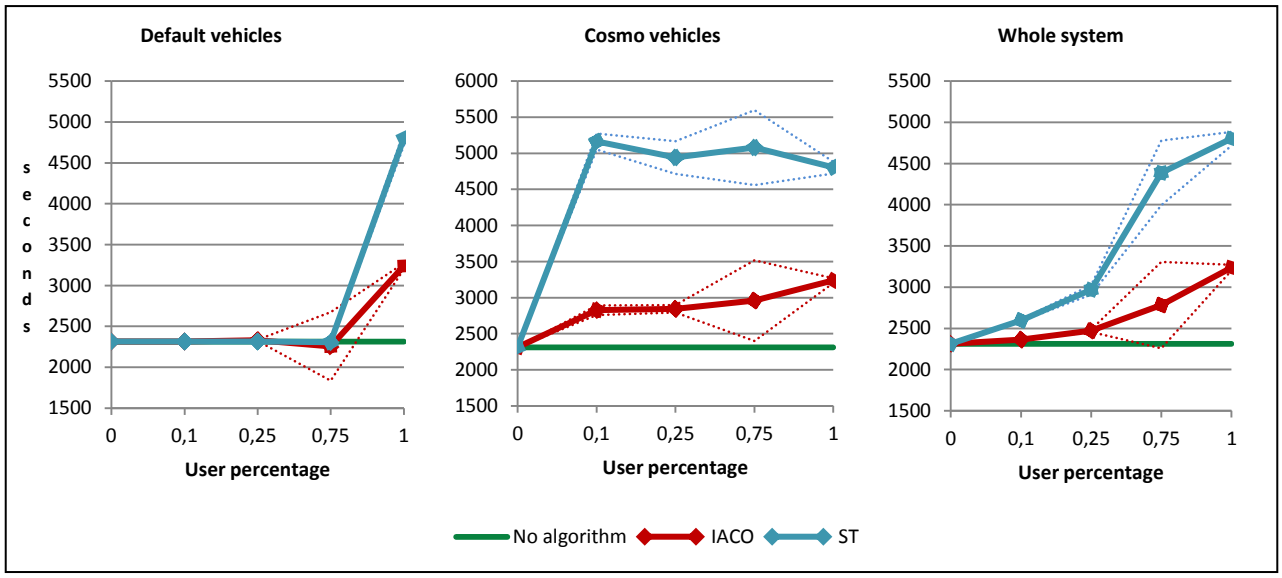

Figure 29 – Lattice (5000 vehicles): effect on traveled route length

Figures 28 and 29 show that the *IACO* algorithm has a slower start than the *ST* algorithm, but it achieves greater trip duration reduction for high user percentages, managing to do so in shorter trip distances.

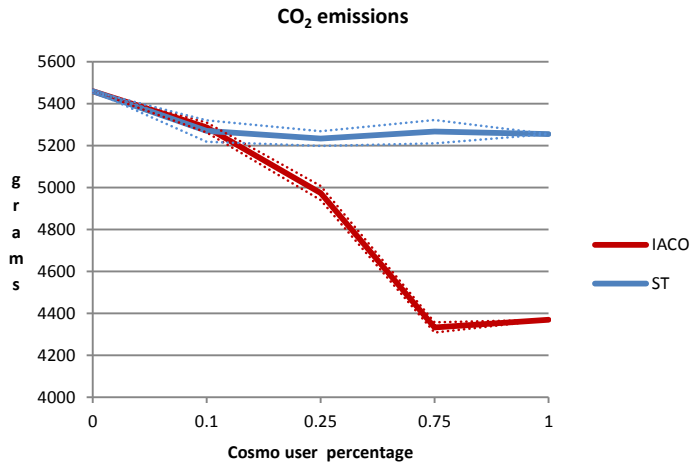

Figure 30 – Lattice (5000 vehicles): CO_2 emissions

Concerning CO_2 emissions, the results show that *IACO* attains lower fuel consumption, and consequently lower CO_2 emissions, with the exception of the 10% user percentage. This indicates that *IACO* distributes traffic more efficiently, attaining shorter trip durations and route lengths than its *ST* counterpart.

In Figures 31 and 32, we can observe that with the increase in total number of vehicles the *IACO* algorithm has a slower start and only manages to surpass the *ST* algorithm when the user percentage reaches a high level. This may be due to the fact that, unlike in the *Radial and Ring* map, there is no central point in the network and so pheromone information is more disperse.

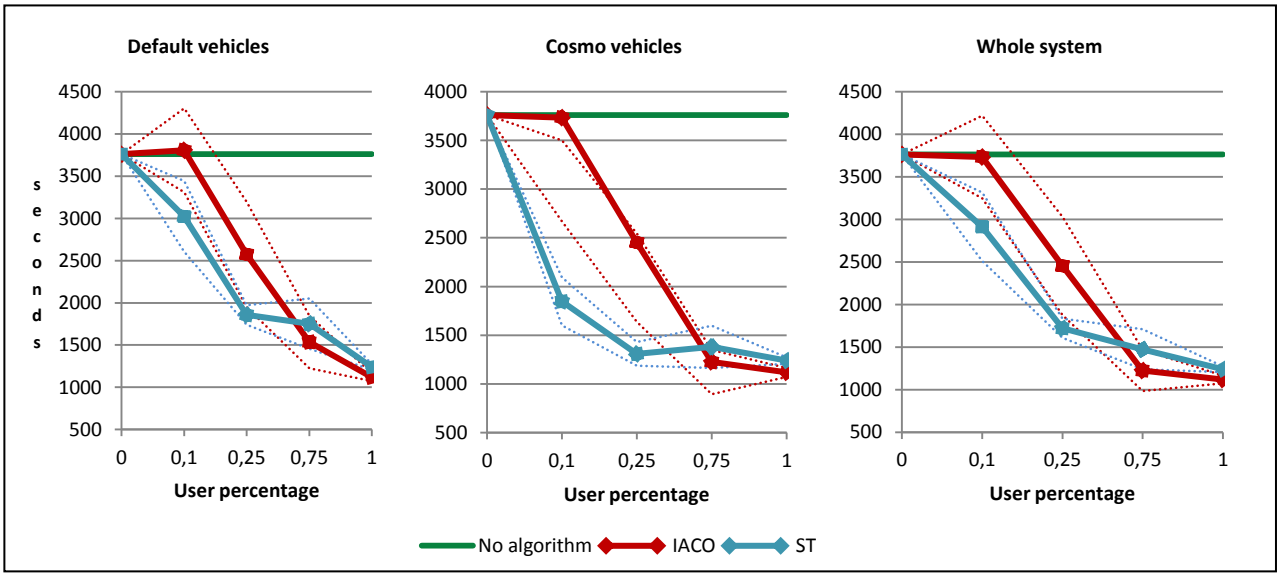

Figure 31 – Lattice (10000 vehicles): effect on trip duration

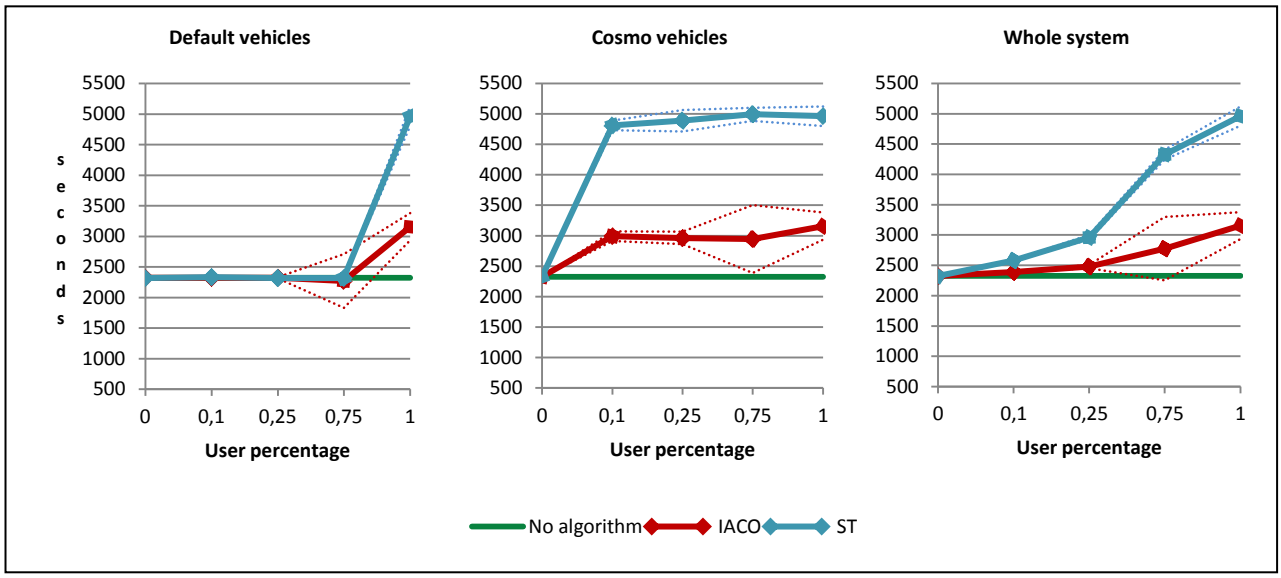

Figure 32 – Lattice (10000 vehicles): effect on traveled route length

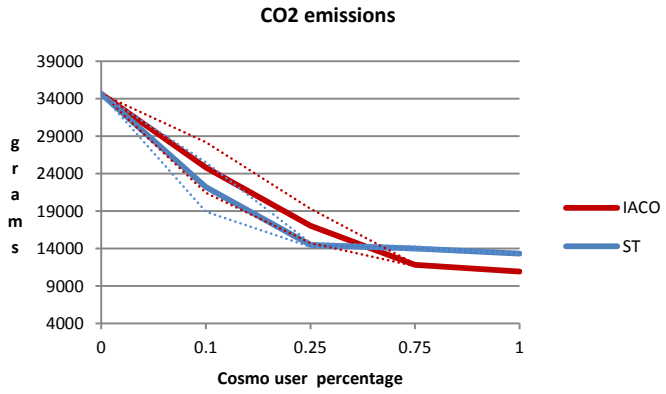

Figure 33 – Lattice (10000 vehicles): CO_2 emissions

With regard to CO_2 emissions, we can observe that for low user percentages, *IACO* obtains worse performance in CO_2 emissions than the *ST* algorithm - similarly to what happens regarding trip duration and route length results. However, as user percentage rises it manages to reach a higher reduction of CO_2 emissions.

By analyzing these results, we can conclude that in this map, significant average travel duration reduction can be achieved using the *Inverted Ant Colony Optimization* algorithm, being an adequate alternative for this map's topology.

Coimbra

In the previous sections we displayed results for *Radial and Ring* and *Lattice* map. Those results demonstrate that, for high user percentages, the *Inverted Ant Colony Optimization* algorithm achieves a greater performance (concerning *average trip duration*, *average route length* and *pollutant emissions*) than its *ST* counterpart.

Having executed experiments in the artificial maps, we will now present results regarding a real life map - *Coimbra* map - that provides a more realistic topology. The following figures refer to experiments with 10.000 vehicles:

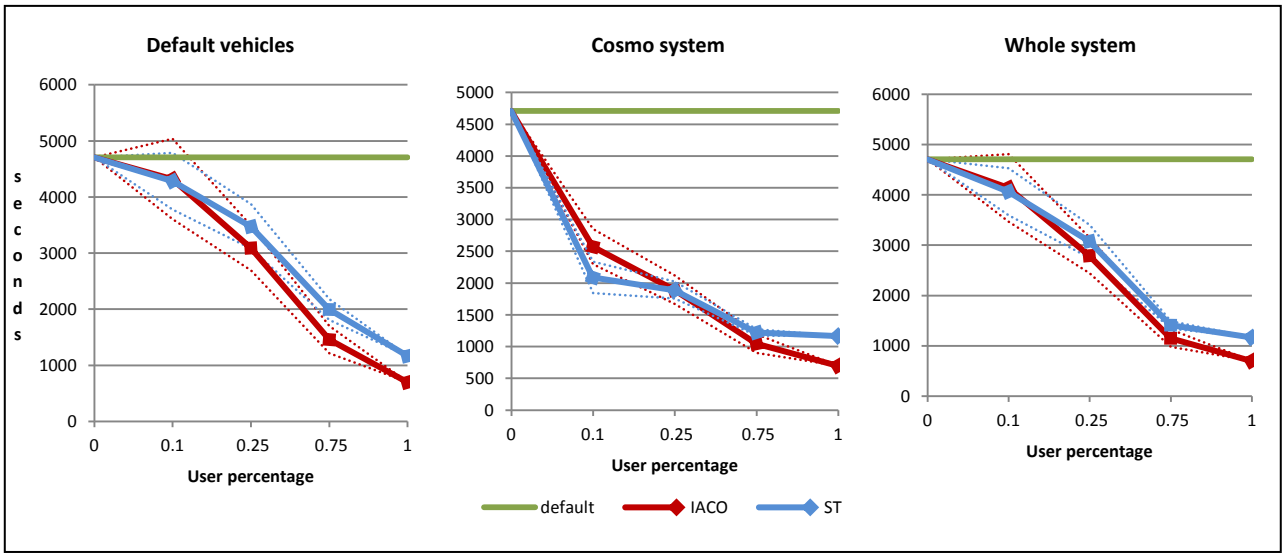

Figure 34 – Coimbra (10000 vehicles): effect on trip duration

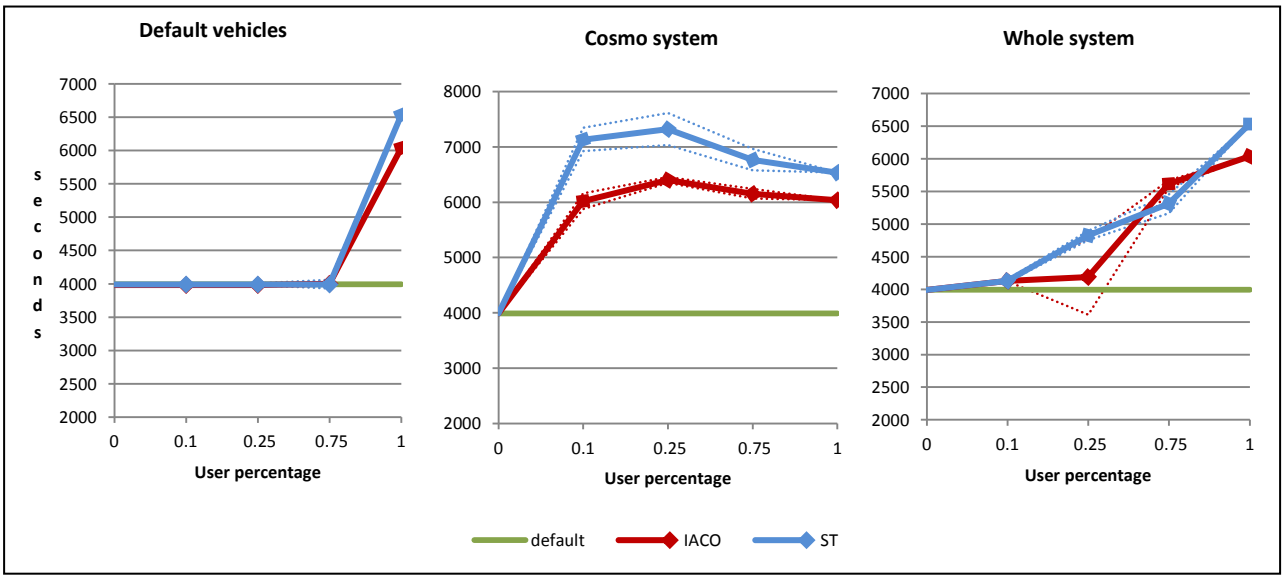

Figure 35 – Coimbra (10000 vehicles): effect on average traveled distance

Figures 34 and 35 show that the *IACO* algorithm has, as expected due to the lack of pheromone information, a slower start than the *ST* algorithm, which has access to occupancy data regarding the entire network. Nevertheless, even with a low user percentage (10%) the use of the *IACO* algorithm already offers significant advantage in trip duration.

As the user percentage increases, so does the performance of each of the algorithms. However, as the *IACO* algorithm reaches a 25% user percentage it already achieves a greater duration reduction than the *ST* counterpart, and continues to do so for greater user percentages. In terms of distance, *IACO* manages to maintain a shorter route length than the *ST* algorithm for every user percentage tested, suggesting that the *IACO* algorithm manages to attain a more precise vision of the actual state of the network.

The following figure displays the pollutant emissions referent to this experiment:

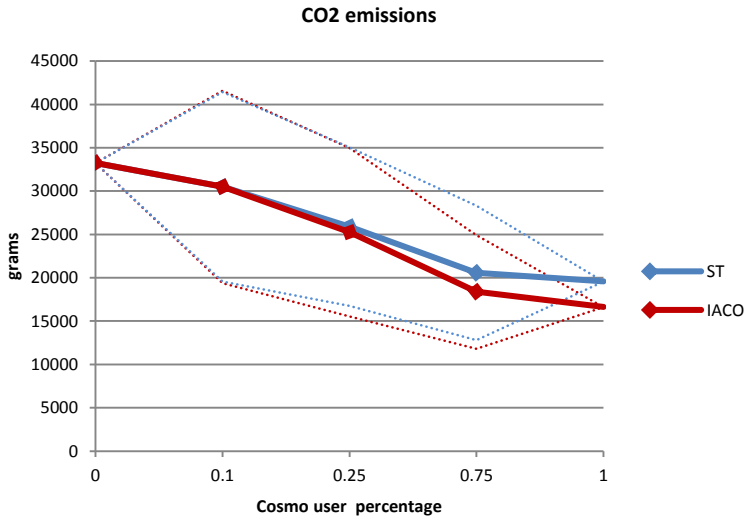

Figure 36 – Coimbra (10000 vehicles): pollutant emissions

Figure 36 demonstrates that both algorithms can attain a major decrease in pollutant emissions. Moreover, the *IACO* algorithm is capable of achieving a 50% reduction on CO_2 emissions and fuel consumption.

In conclusion, the *IACO* algorithm is based on a collaborative approach and tries to disperse traffic in real-time in order to enhance a network's traffic throughput. It relies heavily on the information provided by the users, i.e., the pheromone that they deposit.

Given its nature, with low user percentages, the information that is available about the state of the network in the *IACO* algorithm is somewhat fuzzy and incomplete and therefore, its performance is worse than that of the *ST* algorithm. However, even with low user percentage it is still advantageous given that it offers a considerable reduction in trip duration. This benefit should be appealing enough in order to convince people to adhere to the *IACO* system.

With the rise of the user percentage, the data regarding the network's state becomes more accurate and complete, leading to a greater performance than its *ST* counterpart. As a side effect (and one which we consider important) of this improvement in the network's efficiency, it is possible to obtain a significant decrease in pollutant emissions.

4.3 Personality features

In the previous sections we analyzed the performance of the *IACO* and *ST* algorithms for different networks. Now, we will analyze the effects that the proposed personality features have on those networks and also how they may affect the different algorithms.

Given that our aim to study the overall effects that these features will have in city traffic, the results shown in the following sections refer to the average travel time in the given scenario. For example, we do not intend to study the number of wrong turns that a distracted user makes along the way but rather the effects that those wrong turns will produce in the overall traffic.

We will compare the results obtained with the developed personality features to those of a default SUMO population (i.e. a collection of vehicles with standard values) on the *Lattice*, *Radial and Ring* and *Coimbra* maps. We experimented with 5.000 vehicles and 10.000 vehicles to enable the occurrence of some congestion on the referred networks. The developed features were at first tested separately and afterwards tested simultaneously.

Distraction

Before beginning the simulation, we attributed a random value to each driver to represent the distraction and familiarity parameters. These random values follow a Gaussian distribution with mean value $\mu = 1$ and variance $\sigma^2 = 0{,}25$. A distracted user who is also unfamiliar with the network will have a have a distorted view of the network - we simulate this distorted view by tampering with each driver's perception of the network, altering the weights that a driver associates with each edge. The rejection process is done by defining a threshold value for each of these parameters:

```
if (distraction > distractionThreshold) and (familiarity < familiarityThreshold):
   for edge in graph:
      for sucessor in graph[edge]:
          distance = graph[edge][sucessor]
          graph[edge][sucessor] = distance + random.uniform(-distance, distance)
```

These random values follow a Gaussian distribution with mean value $\mu = 1$ and variance $\sigma^2 = 0{,}25$ so:

- value < 1 corresponds to a 50% probability
- value > 0.75 corresponds to a 84,4% probability
- value > 1 corresponds to a 50% probability
- value > 1.25 corresponds to a 15,6% probability

Given this, we decided to experiment with three different values for *distraction threshold* and one for the *familiarity threshold*:

- *high distraction*: *distractionThreshold* > 0.75 and *familiarityThreshold* < 1
- *medium distraction*: *distractionThreshold* > 1 and *familiarityThreshold* < 1
- *low distraction*: *distractionThreshold* > 1.25 and *familiarityThreshold* < 1

We decided to only allow drivers who are not very familiar with the network to take a wrong path, and therefore only 50% at most can make wrong decisions. Then we varied the *distractionThreshold* to simulate different amounts of distracted drivers. In terms of percentages, the three scenarios present the following probabilities:

- *high distraction*: 42,2%
- *medium distraction*: 25%
- *low distraction*: 7,8%

The following two figures refer to experiments with these three scenarios and the assumption that every driver uses the *IACO* system:

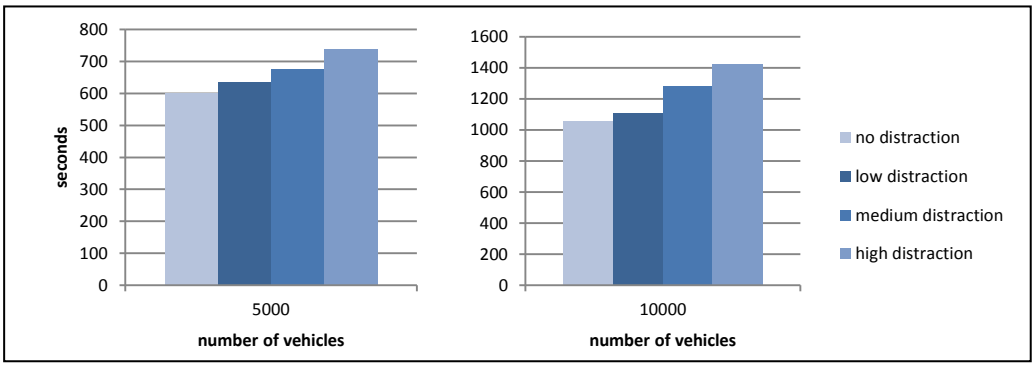

Figure 37 – Distraction: effect on trip duration with IACO algorithm (Lattice map)

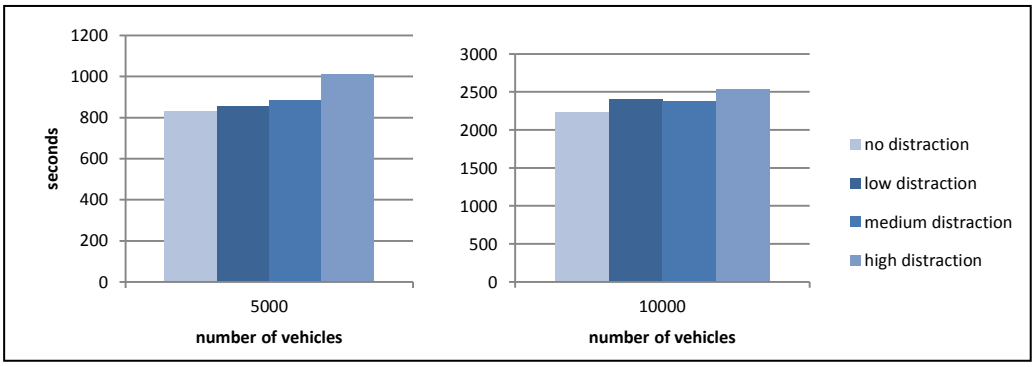

Figure 38 – Distraction: IACO algorithm with different user percentages (Radial and Ring map)

In Figure 37 we observe that in the *Lattice* map, distraction is always harmful to overall network performance. However, in Figure 38 we can observe that in the *Radial and Ring* map - which is more prone to congestion occurrence given its topology - when heavy congestion occurs, a medium distraction is the least harmful of the three levels in terms of average travel time.

We also experimented varying the user percentage in the *Lattice* scenarios, in which the distraction effects seem to be more consistent:

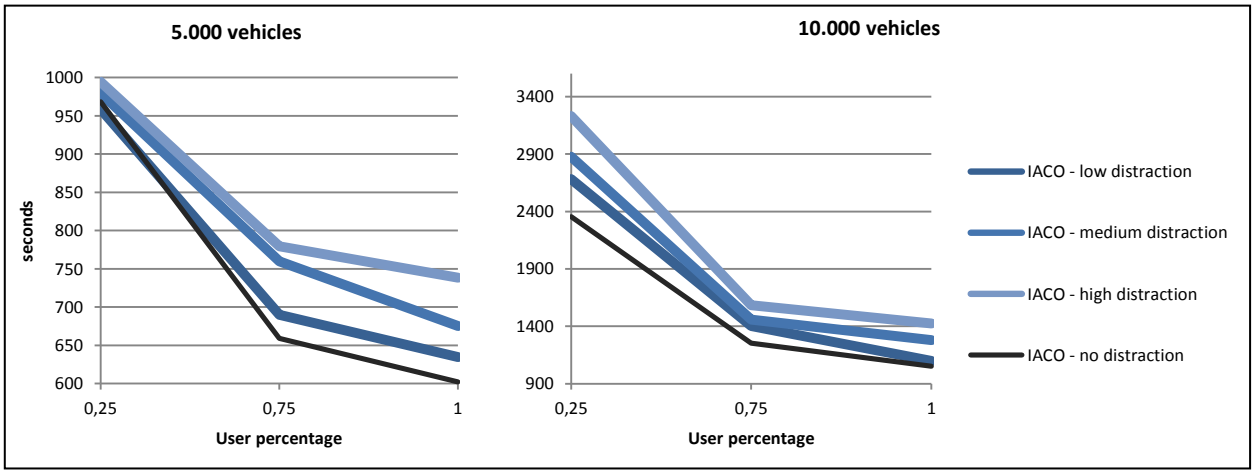

Figure 39 – Distraction: effect on trip duration with IACO algorithm (Radial and Ring map)

In Figure 39 we can see that distraction always has a detrimental effect on performance. It appears to be particularly harmful with a high traffic load and low user percentage - in that scenario, the few drivers that are using the system, are evaluating the network incorrectly and are

therefore not always choosing the correct path, greatly diminishing the system's accuracy. However, as user percentage rises, this effect seems to be mitigated.

The distraction parameter can only have negative effects on the network's performance given that a distracted user who is also unfamiliar with the network will have a distorted view of the network. Therefore individual trips will be longer, unnecessarily increasing the amount of vehicles on the road at any given time. The results show that distraction is always harmful to overall network performance. However, when heavy traffic traverses a more congestion prone map, the fact that some drivers do not take the optimal route might disperse traffic and improve performance.

Stubbornness

The work of Bonsall [47] suggests that there are numerous factors that affect the credibility of received guidance information, including "the extent to which it is corroborated by, or in conflict with, local evidence about the alternatives", a "drivers' familiarity with the local network" and a "the drivers' predisposition to accept advice". Given this information we defined the following driver parameters:

- *experience*: the drivers' familiarity with the local network
- *stubbornness*: the drivers' predisposition to accept advice

Similarly to the distraction feature, prior to the start of simulation, we attributed random values to each driver to represent the stubbornness and experience parameters. Bonsall [47] also states that experienced drivers are more likely to reject advice. These random values follow a Gaussian distribution with mean value $\mu = 1$ and variance $\sigma^2 = 0{,}25$. We simulate the rejection process by defining a threshold value for each of these parameters:

```
if stubbornness < stubbornnessThreshold and experience < experienceThreshold:
  route = newRoute
```

Chen [48] introduces another factor that influences route choice: the effects of a subjects' experience - an estimation of system accuracy by their temporal and spatial experiences. However, these parameters result from a sequence of travels along the same path that will enable drivers to gain knowledge about how the network operates. Our work does not support this evolution given that it pertains to a single run in which driver's choices are modified in real time.

These random values follow a Gaussian distribution with mean value $\mu = 1$ and variance $\sigma^2 = 0{,}25$. So we defined three different acceptance values:

- *high acceptance*: *stubbornnessThreshold* < 1.25 and *experienceThreshold* < 1.25
- *medium acceptance*: *stubbornnessThreshold* < 1and *experienceThreshold* < 1
- *low acceptance*: *stubbornnessThreshold* < 0.75 and *experienceThreshold* < 0.75

These acceptance values have the following probabilities:

- *high acceptance*: 71,2%
- *medium acceptance*: 25%
- *low acceptance*: 2,4%

Figures 40 and 41 refer to experiments with these three scenarios:

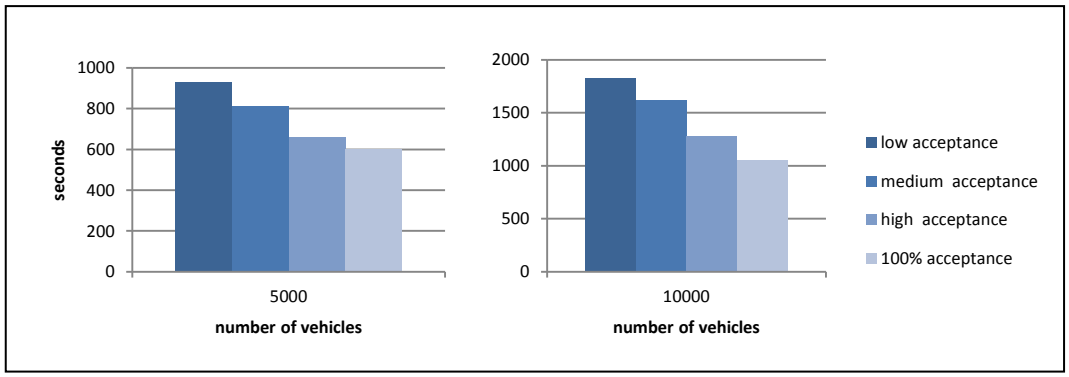

Figure 40 – Stubbornness: effect on trip duration with IACO algorithm (Lattice map)

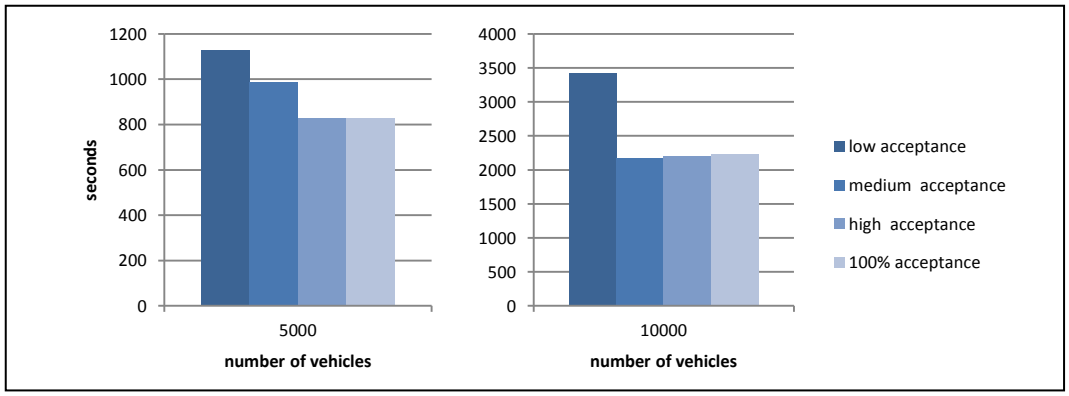

Figure 41 – Stubbornness: effect on trip duration with IACO algorithm (Radial and Ring map)

In Figure 40 we observe that in the *Lattice* map, as the acceptance levels increase, so does the system's performance. However, Figure 41 shows that in the *Radial and Ring* map (more prone to congestion), when heavy congestion occurs, the rejection of some of the proposed routes might even be advantageous in terms of average travel time.

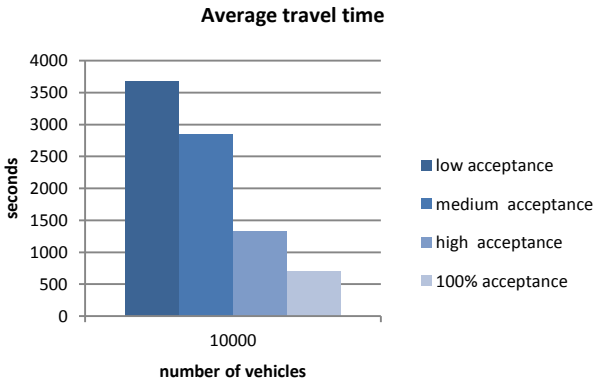

Figure 42 – Stubbornness: effect on trip duration with IACO algorithm (Coimbra map)

In Figure 42, the results of experiments in the *Coimbra* map show similarity to those with the *Lattice* map - there is no specific bottleneck location in the map, unlike in the *Radial and Ring* map, so disregarding the proposed route does not appear to be a beneficial option.

The previous experiments were executed using the *IACO* algorithm. We decided to also experiment the *ST* algorithm in these same scenarios, in order to analyze if the effects of the stubbornness feature were similar:

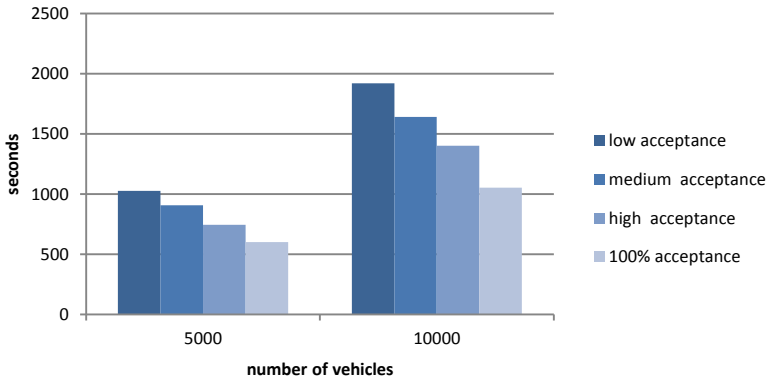

Figure 43 – Stubbornness: effect on trip duration with ST algorithm (Lattice map)

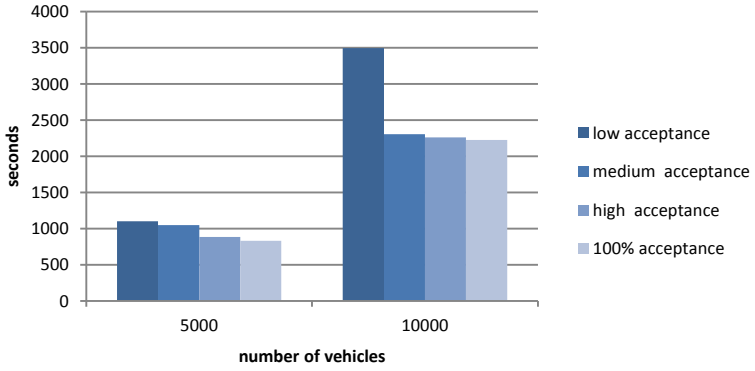

Figure 44 – Stubbornness: effect on trip duration with ST algorithm (Radial and Ring map)

In Figures 43 and 44, the results confirm that there is a similarity between the performance of the *IACO* algorithm (Figure 40 and 41) and the *ST* algorithm regarding the stubbornness parameter. The main difference is the fact that the rejection of some of the proposed routes appears never to be advantageous in terms of average travel time, not even in the *Radial and Ring* map with heavy congestion.

We also experimented varying the user percentage in these scenarios:

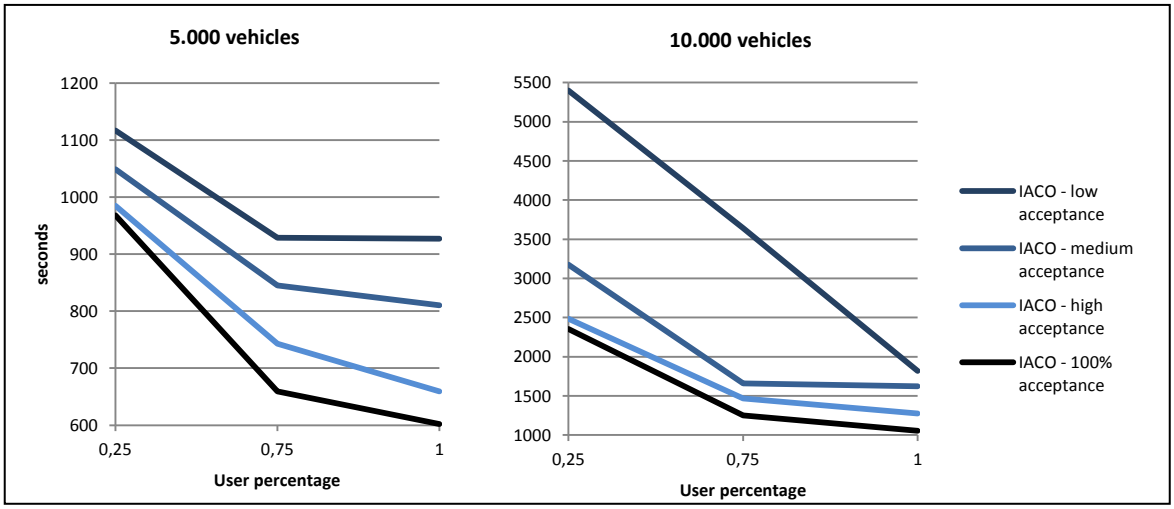

Figure 45 – Stubbornness: effect on trip duration with IACO algorithm (Lattice map)

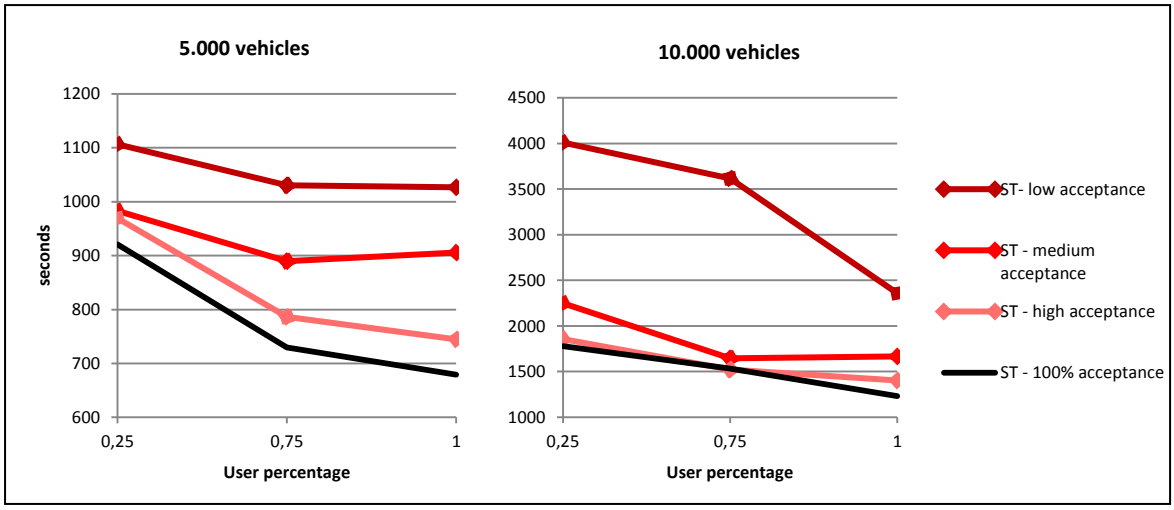

Figure 46 – Stubbornness: effect on trip duration with ST algorithm (Lattice map)

Figures 45 and 46 demonstrate that, in both algorithms, the overall performance of the network increases with the acceptance level of the system. They also show that, for both algorithms, the efficiency deteriorates for high user percentages - it seems to stagnate in most cases.

The stubbornness parameter can produce a negative impact on the network's performance given that rejecting the service provider's proposed route is essentially the same as not using the system, making individual trips longer and consequently deteriorating the network's throughput. The results show precisely that, demonstrating how the overall performance of the network increases with the acceptance level. However, and similarly to what happens with the distraction parameter, when heavy traffic traverses a more congestion prone map, the fact that some drivers reject the proposed route might disperse traffic and improve performance.

Irregularity

According to Krauß [38], the structure of the model dynamics, such as overreactions (where drivers deliberately slow down to velocities lower than necessary) or reduced outflow from jams, "is mediated exclusively by the fluctuations that are introduced ad hoc through the randomization step. If these fluctuations are eliminated, none of the properties of traffic flow is modeled correctly anymore.".
Experimentations were performed for high ($\sigma = 0{,}8$), default ($\sigma = 0{,}5$), and low ($\sigma = 0{,}2$) values of σ and are shown in the following figure:

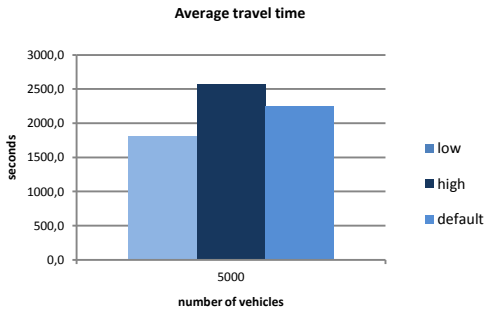

Figure 47 – Effects of irregularity

As we can observe in Figure 47, a greater *driver imperfection* leads to a worse performance in terms of travel time. On the other hand, a low value of *driver imperfection* allows for a significant decrease in the average travel time.

Regarding the irregularity parameter, we can observe in Figure 47 that this parameter greatly influences the average travel time. However, and as previously referred, significant modifications to these values produce an unrealistic vehicle behavior. So, it was decided to only slightly alter this parameter's values. We also decided to incorporate this parameter into the definition of the driver population in order to differentiate precise drivers from those who are more error prone, while maintaining the validity of the properties of the traffic flow model.

Aggressiveness

To experiment the created driver types we executed separate simulations for each age group. A study by the United States Department of Transportation [49] based on data from 2007, shows that there are no significant differences in driver population as far as gender is concerned. So, we established the following population distributions:

	Probability		
Type	**Young Population**	**Middle-aged Population**	**Elder Population**
courteous male	20%	30%	40%
courteous female	25%	35%	45%
aggressive male	30%	20%	10%
aggressive female	25%	15%	5%

Table 2 – Driver populations: type distribution

The study by the United States Department of Transportation [48] also reveals that, on a population of over two hundred million American drivers, the age distribution is approximately as presented next:

from 16 to 39 years of age: ~=42%
from 40 to 64 years of age: ~=45%
above 64 years of age: ~=13%

We believe that the distribution of this population is somewhat universal and therefore it can be used to portray the Portuguese driving population. Therefore, and in order to perform the next testing phase (experimenting all the driver types simultaneously), we established the following population distribution:

	Probability			
Type	**Young**	**Middle-aged**	**Elder**	**Total**
courteous male	8%	12%	8%	28%
courteous female	10%	14%	9%	33%
aggressive male	12%	8%	2%	22%
aggressive female	10%	6%	1%	17%
Total	40%	40%	20%	100%

Table 3 – Aggressiveness: mixed population distribution

The following figures show a comparison between population containing these age groups and a default SUMO population for 5.000 and 10.000 vehicles:

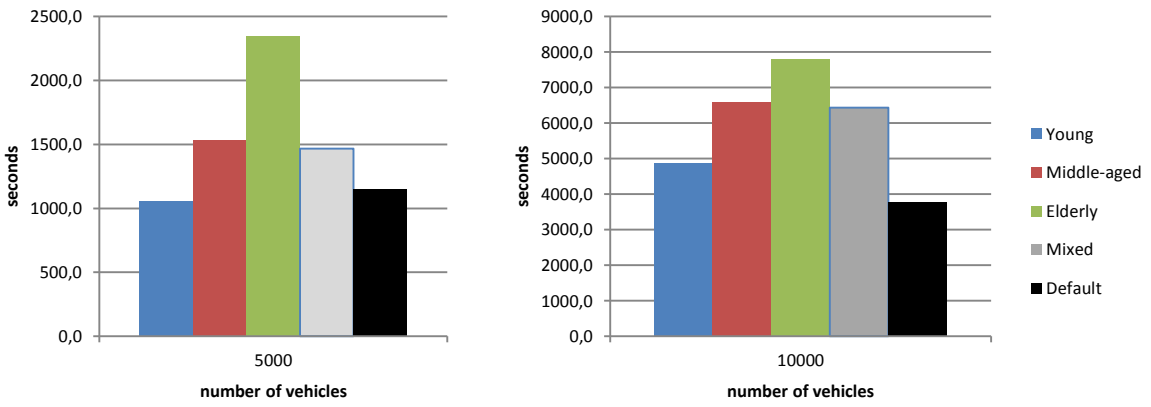

Figure 48 – Aggressiveness: effect on trip duration (Lattice map)

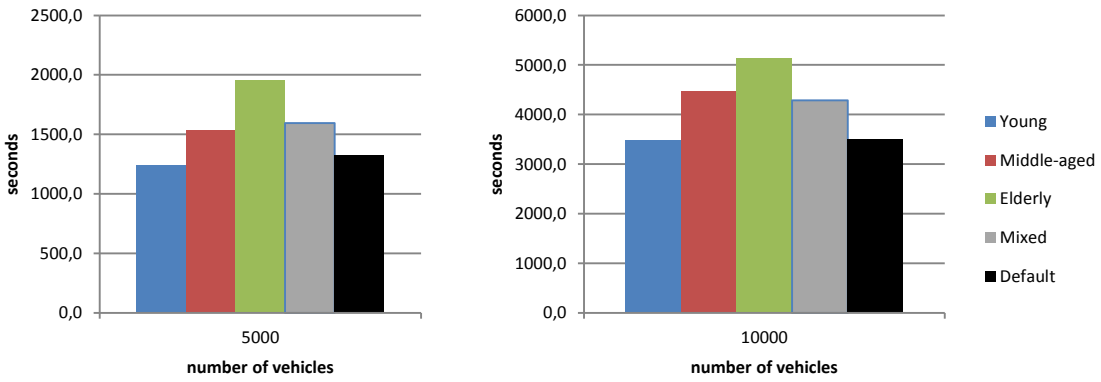

Figure 49 – Aggressiveness: effect on trip duration (Radial and Ring map)

In Figures 48 and 49 we can observe that the faster and more aggressive driving style performed by young drivers achieves a better performance that other alternatives, in both maps. The groups that integrate middle-aged or elderly drivers in its population obtain worse results due to their poorer acceleration capabilities, lower maximum speed and higher reaction time. The elderly group attains the worst results for both maps, given their slower approach. As the traffic load increases the difference in performance shortens, although young drivers still achieve better results than other population alternatives.

Regarding the *Lattice* map, we can observe that for a smaller traffic load, the young population achieves a slightly better performance that the default population. However, when the traffic load increases its performance deteriorates, obtaining poorer results than its default counterpart. This might be due to the fact that the more aggressive driving style performed by young drivers only makes them reach the congested areas sooner, consequently increasing congestion and waiting times.

In relation to *Radial and Ring* map, and similarly to the *Lattice* map, we can observe that for a smaller traffic load, the young population achieves a slightly better performance that the default population. With the increase in vehicles in the network, young driver's higher speed and acceleration do not seem to make any difference, suggesting that congestion is so great that there is not enough space available for these parameters to have a positive influence. On the other hand, the other driver populations approximately maintain their performance.

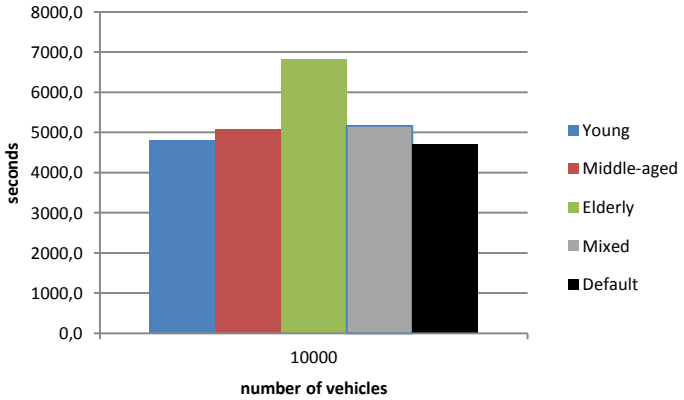

Figure 50 – Aggressiveness: effect on trip duration (Coimbra map)

We also experimented with the *Coimbra* map, and as we can see in Figure 50, the results are similar to those obtained with the artificial networks, *Lattice* and *Radial and Ring* maps. However, the difference between the performance of driver types is not so pronounced, except for the elderly group.

The following figures show a comparison between the same populations as before but now using the *IACO* algorithm:

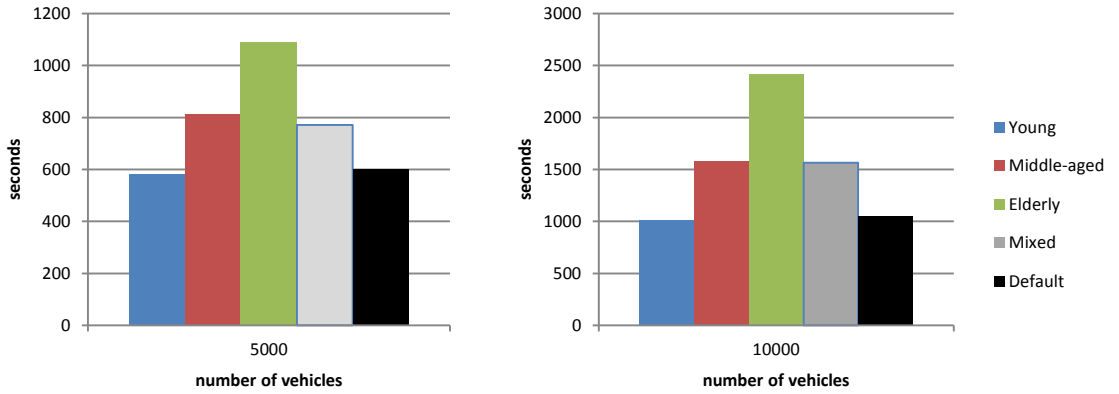

Figure 51 – Aggressiveness: effect on trip duration with IACO algorithm (Lattice map)

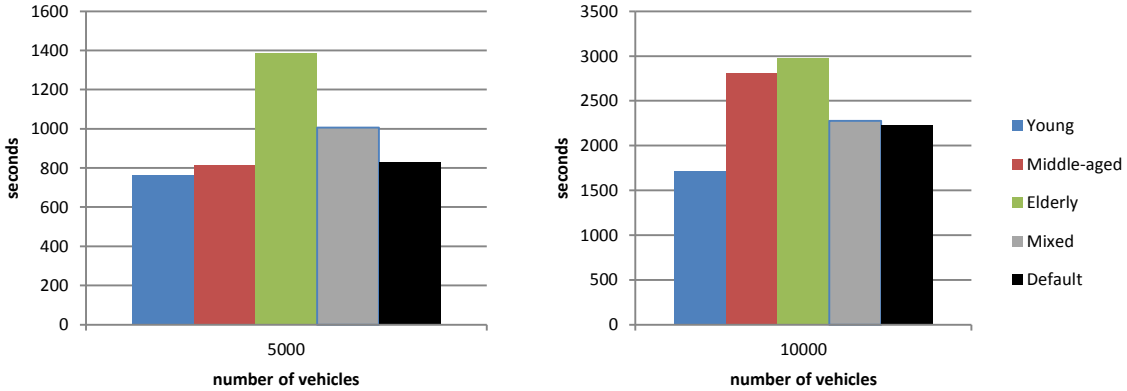

Figure 52 – Aggressiveness: effect on trip duration with IACO algorithm (Radial and Ring map)

In Figures 51 and 52 we can observe that, similarly to the previous experiments, the young drivers achieve a better performance than other alternatives in both maps and that the elderly group attains the worst results in both maps.

The main difference resides in the fact that in *Radial and Ring* map, the young population obtains a slightly better performance that the default population, especially with a higher traffic load. This suggests that now, unlike in the previous scenario, there is enough space available for these parameters to have a positive influence. Given that the *IACO* distributes vehicles more evenly through the network, congestion is diminished and therefore there is room for greater acceleration and speed. Also in the *Radial and Ring* map, it should be noted that there seems to be an irregularity regarding the *middle-aged* group - with 5000 vehicles, its performance is similar to that of the *young* group and with 10000 vehicles it is very similar to the *elderly* group. We believe that with further testing, the results obtained in this map would become more normalized.

Regarding the *Lattice* map, we can also observe a great improvement in the performance of the young population with a high traffic load - it now achieves results equivalent to the default population, while previously it had achieved inferior performance.

In Figure 53 we can observe that, similarly to the previous experiments with artificial networks, in the *Coimbra* map young drivers achieve a better performance than other alternatives in both maps and that the elderly group attains the worst results in both maps.

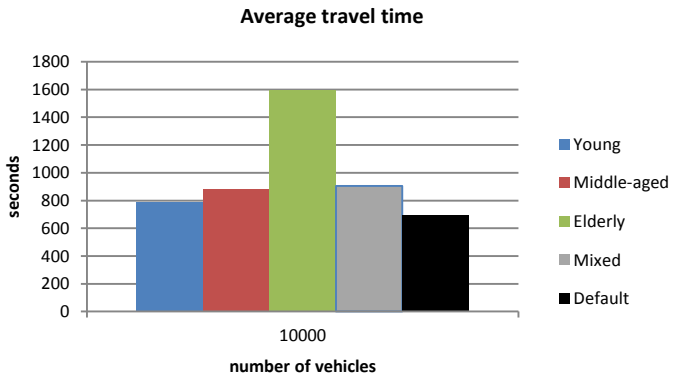

Figure 53 – Aggressiveness: effect on trip duration with IACO algorithm (Coimbra map)

The aggressiveness feature shows the influence of differentiating between age groups in the network performance and how it must be taken into account to develop a more realistic driver population within a simulation.

Chapter 5
Conclusions

This document consists on a description of the development process of traffic flow optimization algorithms and the analysis of their consequences and individual performance. These algorithms were applied to a road traffic micro-simulator named SUMO, which served as a test bed for this multi-agent system, allowing real-time communication with a given simulation.

The testing results show that there are benefits in using whether the *Shortest Time* or the *Inverted Ant Colony Optimization*. Average trip duration is globally reduced, benefiting not only the users of the developed algorithms but also for those vehicles who do not make use of the system. Both algorithms attempt to disperse traffic in real-time in order to enhance a network's traffic throughput. Consequently, fuel consumption and pollutant emissions are greatly reduced.

Analyzing the *Shortest Time* and the *Inverted Ant Colony Optimization* usage results more closely, we can observe that both algorithms achieve fairly similar results. However, being a collaborative algorithm in nature, the *Inverted Ant Colony Optimization* algorithm relies heavily on the pheromone data provided by the users. As the user percentage increases, the data regarding the network's state becomes more accurate and complete and so its best results are attained when the user percentage is higher, managing to outperform *Shortest Time* algorithm in that situation.

We developed features to simulate various driving factors such as driver distraction (that can, per example, be caused by mobile phones), driver aggressiveness (which depends on age, gender and temper) and driver irregularity (that allows us to differentiate good from bad drivers). Also, with the growing demand and diffusion of route planning mechanisms such as Global Positioning System, we developed a feature that simulates a driver's stubbornness, i.e., his/hers unwillingness to accept the proposed route.

In order to achieve a more credible model that substantiated the obtained results, we experimented the developed set of personality parameters in a real life map with real traffic information, along with the developed algorithms. However, due to the current state of the used simulator and its previously referred issues, the degree of realism obtained in the presented results is not totally clear. On the other hand, in this process we found solace in the hope that that the problems we found (which have been reported to the simulator's developers) will be corrected in the future version.

In short, our main contributions are:

- comprehensive review of fields such as: existing traffic management systems; various uses and different implementations of the *Ant colony optimization* algorithm; the development of drivers' personality;

- implementation and experimentation of a distributed *Inverted Ant Colony* algorithm that pro-actively tries to distribute possible routes for real-time city demand - the experimentation was supported by real traffic data from an OD matrix;

- introduction of a set of features that are able to reproduce several aspects of a driver's personality and the study of the impact that these personal features have on city transit.

As a result of this work, we submitted the abstracts of two articles for the *World Conference on Transport Research 2013 Rio* conference.

Regarding future work, there are various possibilities such as: further improving the quality of the real-world network and calibrating real-life scenarios; improving the existing algorithms by fine-tuning their parameters or by designing new ones; development of a more accurate driver personality modeling.

In conclusion, the developed work allowed us to achieve a greater degree of realism, both in terms of driver behavior and experimentation with real-life scenarios. Also, the developed algorithms, which would combine GPS systems with real time traffic information, would be able to disperse traffic in real-time and enhance a network's traffic throughput, further improving the GPS usefulness. This leads to a significant reduction of fuel consumption and CO_2 emissions, meaning that the use of these algorithms allows drivers to save money in fuel and is also eco-friendly.

References

[1] Yamashita, T., Izumi, K., Kurumatani, K. & Nakashima, H. (2005). Smooth traffic flow with a cooperative car navigation system. *Proceedings of the first international joint conference on Autonomous Agents and Multi-Agent Systems*, 478-485. ACM, New York

[2] Adler, J., Satapathy, G., Manikonda, V., Bowles, B. & Blue, V. (2005). A multi-agent approach to cooperative traffic management and route guidance. *Transportation Research Part B: Methodological*, 39(4), 297-318. doi:10.1016/j.trb.2004.03.005

[3] Paruchuri, P., Pullalarevu, A. R. & Karlapalem, K. (2002). Multi agent simulation of unorganized traffic. *Proceedings of the first international joint conference on Autonomous Agents and Multi-Agent Systems*, 176-183. ACM, New York

[4] Oh, B., Na, Y., Yang, J., Park, S., Nang, J. & Kim, J. (2010) Genetic Algorithm-based Dynamic Vehicle Route Search using Car-to-Car Communication. *Advances in Electrical and Computer Engineering*, 10(4); 81 - 86; doi:10.4316/AECE.2010.04013

[5] Sadek, A. W., Smith, B.L. & Demetsky, M. J. (1997). Dynamic traffic assignment: genetic algorithms approach. *Transportation Research Record No. 1588*. 95-103

[6] Dorigo, M. & Di Caro, G. (1997). Ant colony optimization: a new meta-heuristic. *Proceedings of the Congress on Evolutionary Computation, Vol. 2 (June-September 1999)*. 1470-1477. doi:10.1109/CEC.1999.782657

[7] Yu, B., Yang, Z., & Yao, B. (2009). An improved ant colony optimization for vehicle routing problem. *European Journal Of Operational Research*, *196*(1), 171-176. Elsevier B.V. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S0377221708002373

[8] Bell, J., & Mcmullen, P. (2004). Ant colony optimization techniques for the vehicle routing problem. *Advanced Engineering Informatics*, *18*(1), 41-48. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S1474034604000060

[9] Negulescu, S. C., Kifor, C. V., & Oprean, C. (2008). Ant Colony Solving Multiple Constraints Problem: Vehicle Route Allocation. *Communications*, *III*(4), 366-373.

[10] Donati, A., Montemanni, R., Casagrande, N., Rizzoli, A., & Gambardella, L. (2008). Time dependent vehicle routing problem with a multi ant colony system. *European Journal Of Operational Research*, *185*(3), 1174-1191. Elsevier. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S0377221706006345

[11] Gambardella, L. M., Taillard, É., & Agazzi, G. (1999). MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows. (D. Corne, M. Dorigo, & F. Glover, Eds.)*New Ideas in Optimization*, (IDSIA-06-99), 63-76. McGraw-Hill. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.45.5381

[12] Zabala, C., A. & Torres, J. F. (2006). Implementation of the ant colony system with local search for the vehicle routing problem with capacity and time windows (CVRPTW). *Third International Conference on Production Research – Americas' Region 2006*

[13] Bullnheimer, B., Hartl, R. F., & Strauss, C. (1999). An improved ant system algorithm for the vehicle routing problem. *Annals of Operations Research*, *89*(POM-10/97), 319-328. Springer. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.49.1415

[14] He, J., & Hou, Z. (2012). Ant colony algorithm for traffic signal timing optimization. *Advances in Engineering Software*, *43*(1), 14-18. Elsevier Ltd. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S0965997811002419

[15] Claes, R., & Holvoet, T. (2011). Ant Colony Optimization applied to Route Planning Using Link Travel Time Predictions. *2011 IEEE International Symposium on Parallel Distributed Processing Workshops* (pp. 358-365).

[16] Ando, Y., Fukazawa, Y., Masutani, O., Iwasaki, H., & Honiden, S. (2006). Performance of pheromone model for predicting traffic congestion. *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems AAMAS 06*, 73. ACM Press. Retrieved from http://portal.acm.org/citation.cfm?doid=1160633.1160642

[17] Yue, Y., Zhou, L., Yue, Q., & Fan, Z., (2011) Multi-route railroad blocking problem by improved model and ant colony algorithm in real world, *Computers & Industrial Engineering, Volume 60, Issue 1, February 2011*, 34-42. Retrieved from http://www.sciencedirect.com/science/article/pii/S0360835210002615)

[18] Gunes, M., Sorges, U., & Bouazizi, I. (2002). ARA-the ant-colony based routing algorithm for MANETs. *Proceedings International Conference on Parallel Processing Workshop*, *34*, 79-85. IEEE Comput. Soc. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1039715

[19] Claes, R. & Holvoet, T., (2012). Cooperative ant colony optimization in traffic route calculations. *Practical Applications of Agents and Multiagent Systems*

[20] Garro, B., Sossa, H., & Vazquez, R. (2007) Evolving ant colony system for optimizing path planning in mobile robots. *Electronics, Robotics and Automotive Mechanics Conference (CERMA 2007),* 444-449. doi: http://doi.ieeecomputersociety.org/10.1109/CERMA.2007.60

[21] Robinson, E. J. H., Ratnieks, F. L. W., & Holcombe, M. (2008). An agent-based model to investigate the roles of attractive and repellent pheromones in ant decision making during foraging. *Journal of Theoretical Biology*, *255*(2), 250-8. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18778716

[22] Chen, B. & Cheng, H. H. (2010). A Review of the Applications of Agent Technology in Traffic and Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, 11(2), 485-497. doi:10.1109/TITS.2010.2048313

[23] Krajzewicz, D., Hertkorn, G., Wagner, P. & Rössel, C. (2002). An example of microscopic car models validation using the open source traffic simulation SUMO. *Proceedings of Simulation in Industry 14th European Simulation Symposium*, 318–322

[24] Vaa, T. (2001). Cognition And Emotion In Driver Behaviour Models: Some Critical Viewpoints. *Road user characteristics with emphasis on life-styles, quality of life and safety - Proceedings of 14th ICTCT Workshop*

[25] Ehlert, P. A. M., & Rothkrantz, L. J. M. (2001). Microscopic traffic simulation with reactive driving agents. *ITSC 2001 2001 IEEE Intelligent Transportation Systems Proceedings Cat No01TH8585*, *40*(5), 860-865. Ieee. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=948773

[26] Demir, M., Çavuşoğlu, A. (2012). A new driver behavior model to create realistic urban traffic environment. *Transportation Research Part F: Traffic Psychology and Behaviour*, *Vol. 15, No. 3*. 289-296, doi:10.1016/j.trf.2012.01.004

[27] Tasca, L. (2000). A Review Of The Literature On Aggressive Driving Research. *Aggressive Driving Issues Conference.* Retrieved from: http://www.stopandgo.org/research/aggressive/tasca.pdf

[28] Laagland, J. (2005). How To Model Aggressive Behavior In Traffic simulation. *Aggressive Behavior.* Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.66.2976

[29] 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,*

Volume 10, Issues 5–6, October–December 2002. 331-349. doi: http://dx.doi.org/10.1016/S0968-090X(02)00025-6

[30] Gao, F., & Wang, M. (2010). Route Choice Behavior Model with Guidance Information. *Journal of Transportation Systems Engineering and Information Technology*, *10*(6), 64-69. China Association for Science and Technology. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S1570667209600726

[31] http://www.hbefa.net/e/index.html

[32] http://encyclopedia2.thefreedictionary.com/

[33] Transportation Research Board Special Report on Traffic Flow Theory, 2, 5-11, available at: http://www.fhwa.dot.gov/publications/research/operations/tft/chap2.pdf

[34] SUMO user documentation, available at: http://sourceforge.net/apps/mediawiki/sumo/index.php?title=TraCI/Induction_Loop_Value_Retrieval

[35] SUMO user documentation, available at: http://sumo.sourceforge.net/doc/current/docs/userdoc/Tutorials/OSMActivityGen/eichstaett.osm.html

[36] Seco, Á., Pinto, N. N. (2002) Matriz OD Coimbra 2002 - Justificação da Metodologia e Apresentação

[37] World Health Organization. (2011). Mobile phone use - a growing problem of driver distraction. Retrieved from: http://www.who.int/violence_injury_prevention/publications/road_traffic/distracted_driving/en/index.html

[38] Krauß, S. (1998). Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics. *Thesis and dissertation*, (319), 115. Forschungsbericht, Deutsches Zentrum für Luft- und Raumfahrt; zugl. Diss., Universität Köln. Retrieved from http://elib.dlr.de/8380

[39] Dukes, R. L., Clayton, S. L., Jenkins, L. T., Miller, T. L., & Rodgers, S. E. (2001). Effects of aggressive driving and driver characteristics on road rage. *The Social Science Journal*, *38*(2), 323–331. Elsevier. Retrieved from: http://linkinghub.elsevier.com/retrieve/pii/S0362331901001173

[40] Wickens, C. M., Mann, R. E., Stoduto, G., Butters, J. E., Ialomiteanu, A., & Smart, R. G. (2012). Does gender moderate the relationship between driver aggression and its risk factors? *Accident Analysis Prevention*, *45*, 10-18. Retrieved from http://dx.doi.org/10.1016/j.aap.2011.11.013

[41] Björklund, G. M. (2008). Driver irritation and aggressive behaviour. *Accident analysis and prevention*, *40*(3), 1069-1077. Retrieved from http://www.ncbi.nlm.nih.gov/pubmed/18460375

[42] Holland, C., Geraghty, J., & Shah, K. (2010). Differential moderating effect of locus of control on effect of driving experience in young male and female drivers. *Personality and Individual Differences*, *48*(7), 821-826. Elsevier. Retrieved from http://www.elsevier.com/wps/find/journaldescription.cws_home/603/description#description

[43] Davis, L. (2003). Modifications of the optimal velocity traffic model to include delay due to driver reaction time. *Physica A: Statistical Mechanics and its Applications*, *319*, 557-567. Retrieved from http://linkinghub.elsevier.com/retrieve/pii/S0378437102014577

[44] Mehmood, A., & Easa, S. M. (2009). Modeling Reaction Time in Car-Following Behaviour Based on Human Factors. *Engineering and Technology*, *3*(2), 93-101.

[45] McGehee, D. V., Mazzae, E. N., & Baldwin, S. G. H. (2000). Driver reaction time in crash avoidance research: Validation of a driving simulator study on a test track. *Human Factors and Ergonomics Society Annual Meeting Proceedings* (Vol. 44, p. 320–323). Human Factors and Ergonomics Society. Retrieved from http://www.ingentaconnect.com/content/hfes/hfproc/2000/00000044/00000020/art00026

[46] Triggs, T. J. & Harris, W. G & Monash University. Human Factors Group (1982). *Reaction time of drivers to road stimuli.* Human Factors Group, Dept. of Psychology, Monash University, Melbourne, Australia

[47] Bonsall, P. W., & Joint, M. (1991). Driver compliance with route guidance advice: The evidence and its implications. *Vehicle Navigation and Information Systems Conference 1991*. Retrieved from http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1623611

[48] Chen, W., & P. Jovanis, P. (1997). Analysis of a Driver En-Route Guidance Compliance and Driver Learning with ATIS Using a Travel Simulation Experiment. *Institute of Transportation Studies, University of California, Davis, Research Report UCD-ITS-RR-97-12*

[49] United States Department of Transportation. (2007). *Distribution of licensed drivers by sex and percentage in each age group and relation to population*. Retrieved from http://www.fhwa.dot.gov/policyinformation/statistics/2007/pdf/dl20.pdf

Appendix A

First Semester Chart:

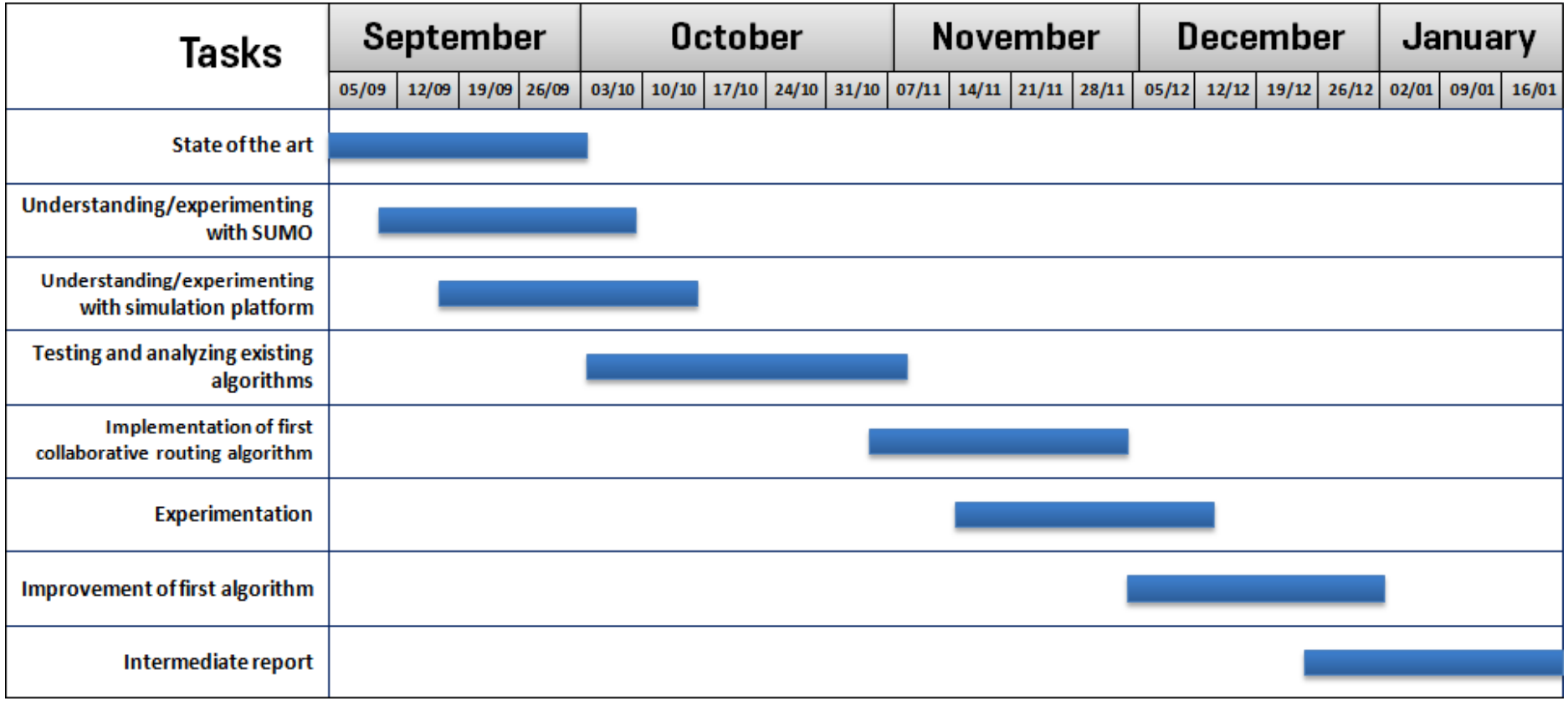

Figure 54 – First Semester Gantt Chart

Second Semester Chart (prediction):

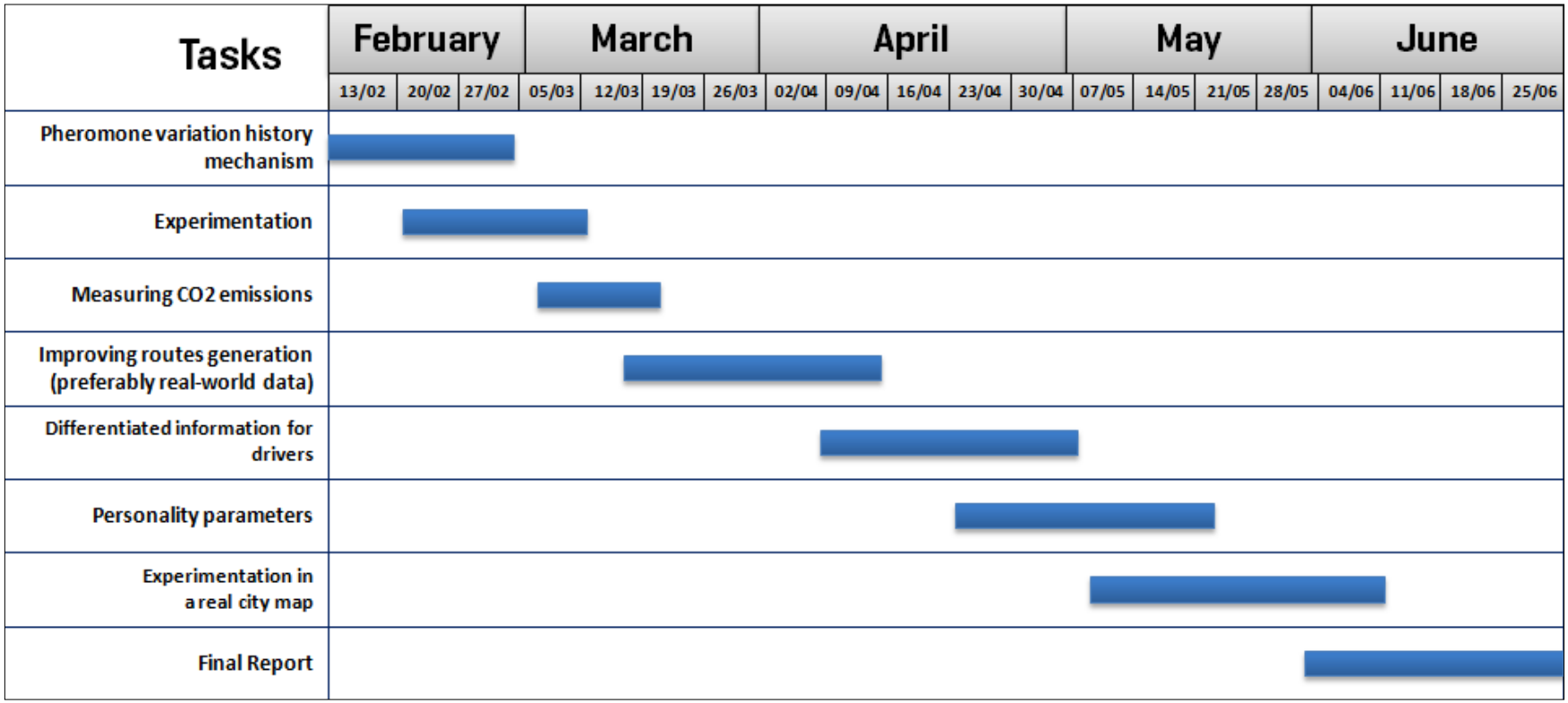

Figure 55 – Second Semester Gantt Chart (prediction)

Second Semester Chart (actual chart):

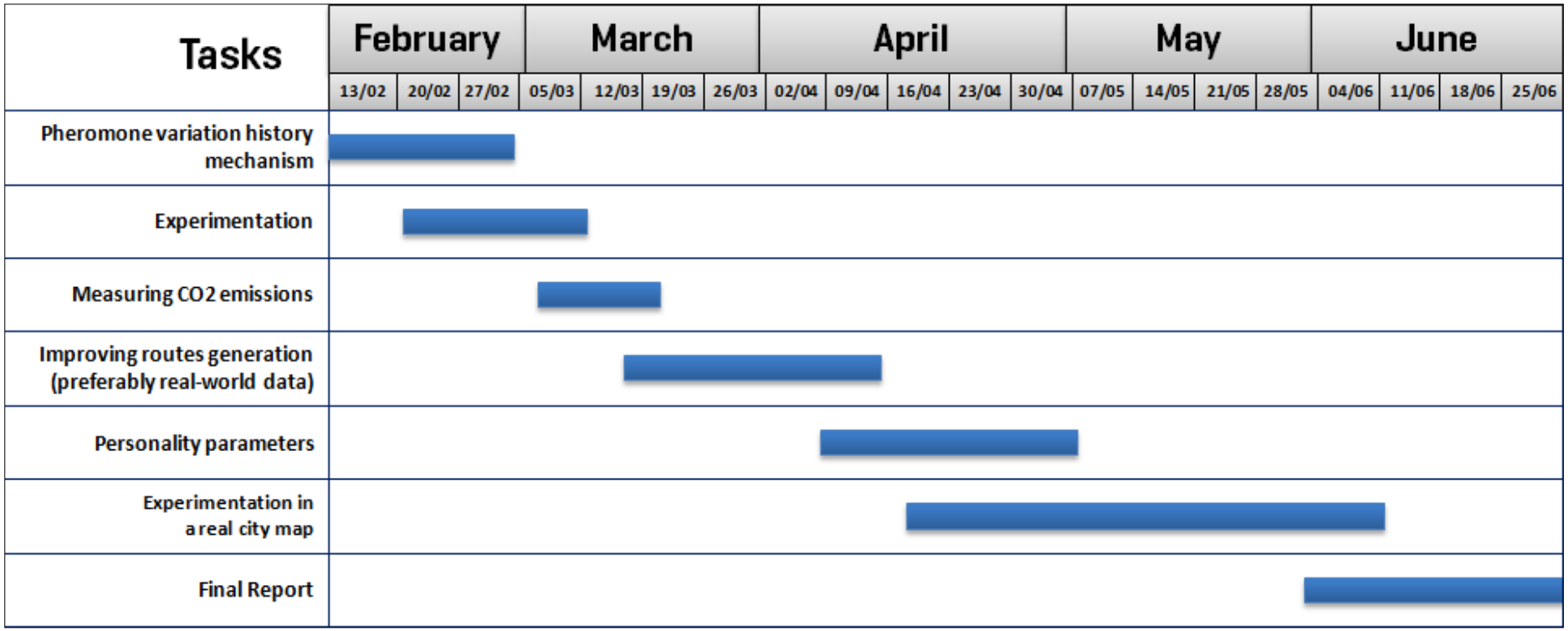

Figure 56 – Second Semester Gantt Chart (actual chart)

Appendix B

	5.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength stdDevs	10.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength stdDevs
LATTICE	*25% SD*	total	1144	2312	13	0	*25% SD*	total	4771	2330	756	4
		cosmo	1145	2319	19	11		cosmo	4764	2334	768	18
		default	1146	2310	12	4		default	4773	2329	753	6
	25% ST	total	924	2546	20	99	*25% ST*	total	1777	2522	112	34
		cosmo	673	3289	20	332		cosmo	1366	3135	123	148
		default	1008	2311	23	10		default	1914	2318	116	5
	75% SD	total	1145	2312	6	1	*75% SD*	total	4664	2329	726	3
		cosmo	1146	2309	10	6		cosmo	4658	2327	727	7
		default	1142	2320	22	18		default	4683	2337	732	21
	75% ST	total	738	2923	26	345	*75% ST*	total	1532	3073	237	362
		cosmo	694	3128	30	464		cosmo	1424	3344	217	491
		default	868	2310	21	22		default	1855	2318	298	18
	default	total	1147	2312	-	-	*default*	total	3761	2325	-	-

Table 4 – Shortest Distance vs Shortest Time: Lattice map results

RADIAL AND RING	5.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength stdDevs	10.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength stdDevs
	25% SD	total	1351	1861	121	0	*25% SD*	total	3490	1865	238	10
		cosmo	1350	1857	124	11		cosmo	3492	1861	256	14
		default	1357	1863	119	4		default	3477	1863	249	8
	25% ST	total	1149	2131	136	73	*25% ST*	total	3162	2047	1048	118
		cosmo	1100	2999	130	324		cosmo	2900	2615	830	470
		default	1166	1861	139	8		default	3249	1858	1128	3
	75% SD	total	1368	1861	50	1	*75% SD*	total	3543	1860	475	1
		cosmo	1371	1861	50	6		cosmo	3527	1855	471	5
		default	1366	1859	66	16		default	3592	1873	489	12
	75% ST	total	947	2841	57	360	*75% ST*	total	2440	2678	180	299
		cosmo	935	3191	55	467		cosmo	2437	3097	174	400
		default	985	1853	77	14		default	2448	1851	200	14
	default	total	1320	1862	-	-	*default*	total	3497	1860	-	-

Table 5 – Shortest Distance vs Shortest Time: Radial and ring map results

	5.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength StdDevs	10.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength StdDevs
Lattice	10% ACO	total	3735	2392	485	7	10% ACO	total	1061	2363	39	6
		cosmo	3085	2992	415	77		cosmo	884	2826	35	66
		default	3807	2325	497	8		default	1086	2312	29	5
	10% ST	total	2913	2581	402	7	10% ST	total	1042	2598	15	9
		cosmo	1847	4810	246	80		cosmo	691	5162	22	110
		default	3023	2333	420	3		default	1080	2313	16	5
	25% ACO	total	2453	2482	577	26	25% ACO	total	963	2473	15	16
		cosmo	2092	2962	453	103		cosmo	786	2844	53	51
		default	2573	2322	623	11		default	1031	2329	14	8
	25% ST	total	1720	2963	112	42	25% ST	total	921	2969	20	57
		cosmo	1309	4890	123	177		cosmo	671	4940	20	225
		default	1857	2320	116	5		default	1004	2312	23	10
	75% ACO	total	1225	2777	243	523	75% ACO	total	653	2782	124	522
		cosmo	1121	2945	227	558		cosmo	598	2959	116	559
		default	1535	2271	310	440		default	819	2254	154	419
	75% ST	total	1474	4326	237	82	75% ST	total	730	4385	26	392
		cosmo	1381	4995	217	108		cosmo	689	5078	30	520
		default	1754	2318	298	18		default	853	2305	21	22
	100% ACO	total	1119	3157	46	222	100% ACO	total	637	3237	15	34
		cosmo	1119	3157	46	222		cosmo	637	3237	15	34
		default	1119	3157	46	222		default	637	3237	15	34
	100% ST	total	1238	4983	*34*	*159*	100% ST	total	683	4800	*10*	*79*
		cosmo	1238	4983	*34*	*159*		cosmo	683	4800	*10*	*79*
		default	1238	4983	*34*	*159*		default	683	4800	*10*	*79*
	default	total	*3761*	*2325*			default	total	*1147*	*2313*		

Table 6 – Shortest Time vs Ant Colony Optimization: Radial and ring map results

	5.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength StdDevs	10.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength StdDevs
Radial and Ring	10% ACO	total	3330	1941	*402*	*5*	10% ACO	total	1293	1937	*108*	*22*
		cosmo	3129	2714	*358*	*56*		cosmo	1174	2682	*100*	*59*
		default	3352	1859	*407*	*2*		default	1306	1861	*109*	*5*
	10% ST	total	2804	2074	*274*	*9*	10% ST	total	1158	2121	55	11
		cosmo	2544	3995	*233*	*89*		cosmo	1006	4456	60	96
		default	2833	1861	*282*	*3*		default	1175	1861	56	4
	25% ACO	total	2583	2061	*250*	*12*	25% ACO	total	1167	2104	*85*	*16*
		cosmo	2415	2675	*245*	*46*		cosmo	1061	2789	*97*	*59*
		default	2646	1857	*261*	*4*		default	1202	1885	*83*	*4*
	25% ST	total	2798	2350	*1039*	*95*	25% ST	total	1042	2462	*136*	*78*
		cosmo	2741	3825	*888*	*378*		cosmo	988	4404	*130*	*309*
		default	2986	1859	*1095*	*5*		default	1060	1815	*139*	*8*
	75% ACO	total	2045	2392	*450*	*480*	75% ACO	total	824	2450	*163*	*467*
		cosmo	1993	2593	*433*	*527*		cosmo	790	2665	*157*	*510*
		default	2164	1791	*478*	*351*		default	926	1806	*185*	*342*
	75% ST	total	2460	2913	*180*	*168*	75% ST	total	925	3530	*57*	*328*
		cosmo	2456	3265	*174*	*224*		cosmo	917	4088	*55*	*437*
		default	2473	1855	*200*	*14*		default	947	1855	*77*	*14*
	100% ACO	total	1943	2823	*192*	*74*	100% ACO	total	840	2928	*29*	*16*
		cosmo	1940	2830	*183*	*38*		cosmo	840	2928	*29*	*16*
		default	1951	2800	*221*	*194*		default	840	2928	*29*	*16*
	100% ST	total	2120	3805	*55*	*40*	100% ST	total	847	4107	*12*	*96*
		cosmo	2120	3805	*55*	*40*		cosmo	847	4107	*12*	*96*
		default	2120	3805	*55*	*40*		default	847	4107	*12*	*96*
	default	total	*3497*	*1860*			default	total	*1320*	*1862*		

Table 7 – Shortest Time vs Ant Colony Optimization: Lattice map results

	10.000 vehicles		duration avg (s)	routeLength avg (m)	duration stdDev	routeLength StdDevs
Coimbra	*10% ACO*	total	4099	4194	670	12
		cosmo	2554	6029	278	144
		default	4276	3990	716	7
	10% ST	total	4064	4309	469	21
		cosmo	2087	7134	247	210
		default	4283	3996	505	7
	25% ACO	total	2856	4285	342	546
		cosmo	1952	6398	231	58
		default	3156	3978	384	19
	25% ST	total	2982	4834	279	64
		cosmo	1867	7365	115	272
		default	3354	3991	338	13
	75% ACO	total	1093	5604	158	56
		cosmo	1005	6145	133	81
		default	1356	3983	234	38
	75% ST	total	1416	6031	65	145
		cosmo	1230	6713	44	197
		default	1979	3985	159	58
	100% ACO	total	727	6066	34	47
			727	6066	34	47
			727	6066	34	47
	100% ST	total	1074	6568	68	36
			1074	6568	68	36
			1074	6568	68	36
	default	total	*4709*	*3992*		

Table 8 – Shortest Time vs Ant Colony Optimization: Coimbra map results

			5.000 vehicles				10.000 vehicles			
	Algorithm	User percentage	CO2 emissions avg (g)	fuel consumption avg (ml)	CO2 emissions stdDev	fuel consumption stdDev	CO2 emissions avg (g)	fuel consumption avg (ml)	CO2 emissions stdDev	fuel consumption stdDev
Lattice	ACO	10%	5286	2108	24	10	24801	9888	3380	1348
		25%	4974	1983	34	14	17014	6780	2281	909
		75%	4333	1728	24	10	11791	4699	151	60
		100%	4370	1743	37641	15	10928	4344	96710	19
	ST	10%	5270	2107	51	11	22196	8849	3276	1307
		25%	5234	2087	35	14	14442	5755	259	103
		75%	5267	2100	56	22	14004	5594	244	92
		100%	5255	2092	30750	29	13306	5357	120388	55
	default	0%	5459	2177	-	-	34637	13813	-	-
Radial and Ring	ACO	10%	5533	2206	237	93	22559	8993	2760	1101
		25%	5251	2094	191	76	18248	7261	1302	518
		75%	4597	1833	78	31	16060	6399	1016	404
		100%	4769	1884	62147	4	17075	6860	195717	68
	ST	10%	5416	2168	133	55	19913	7938	2005	799
		25%	5296	2112	119	47	18237	7269	1440	574
		75%	5533	2206	88	35	17498	6975	849	337
		100%	5439	2174	52703	28	17372	7043	487022	175
	default	0%	5618	2240	-	-	27750	11066	-	-
Coimbra	ACO	10%	-	-	-	-	30473	12155	11092	1088
		25%	-	-	-	-	25252	10050	9711	601
		75%	-	-	-	-	18369	7316	6551	341
		100%	-	-	-	-	16782	6676	308853	121
	ST	10%	-	-	-	-	30478	12143	10928	11067
		25%	-	-	-	-	25887	10382	9149	490
		75%	-	-	-	-	20550	8147	7762	103
		100%	-	-	-	-	19209	7796	314703	114
	default	0%	-	-	-	-	33227	13253	-	-

Table 9 – Shortest Time vs Ant Colony Optimization: CO2 emissions

Inverted Ant Colony algorithm							
map		Lattice					
number of vehicles		5.000			10.000		
user percentage		0,25	0,75	1	0,25	0,75	1
low distraction	average time (s)	959	690	634	2660	1404	1106
	average distance (m)	2970	3140	3109	2899	3135	3193
	stddev time	11,11	8,26	17,65	166,96	22,58	32,97
	stddev distance	49,03	27,41	77,38	416,46	21,94	43,99
medium distraction	average time (s)	981	760	675	2875	1461	1279
	average distance (m)	3016	3097	3142	3035	3142	3209
	stddev time	7,69	13,23	12,34	182,65	36,95	31,95
	stddev distance	39,72	70,25	60,80	15,81	16,08	34,83
high distraction	average time (s)	996	782	738	3192	1574	1424
	average distance (m)	2968	3112	3132	3041	3147	3192
	stddev time	11,01	30,89	9,64	315,07	149,92	46,58
	stddev distance	61,64	23,97	50,13	12,16	29,84	38,85
no distraction	time (m)	*968*	*659*	*602*	*2354*	*1253*	*1053*
	distance (s)	*2945*	*2907*	*2957*	*2959*	*2905*	*2924*

Table 10 – Distraction feature: IACO with different user percentages in Lattice map

		Inverted Ant Colony algorithm			
map		Lattice		Radial and Ring	
number of vehicles		5000	10000	5000	10000
low distraction	average time (s)	634	1106	850	2410
	average distance (m)	3109	3195	2873	2714
	stddev time	17,65	18,44	39,86	164,50
	stddev distance	77,38	37,98	45,94	44,76
medium distraction	average time (s)	675	1279	885	2398
	average distance (m)	3142	3209	2881	2739
	stddev time	12,34	31,95	20,10	209,83
	stddev distance	60,80	34,83	41,82	39,92
high distraction	average time (s)	738	1424	1010	2534
	average distance (m)	3132	3192	2889	2743
	stddev time	9,64	46,58	28,61	265,55
	stddev distance	50,13	38,85	34,01	49,17
no distraction	time (m)	*602*	*1053*	*829*	*2223*
	distance (s)	*3157*	*3224*	*2811*	*2677*

Table 11 – Distraction feature: effect in different maps

		Inverted Ant Colony algorithm					No algorithm				
map		Lattice		Radial and Ring		Coimbra	Lattice		Radial and Ring		Coimbra
number of vehicles		5000	10000	5000	10000	10000	5000	10000	5000	10000	10000
Young	average time (s)	581	1020	759	1724	734	1052	4865	1238	3482	4803
	average distance (m)	2192	2236	1933	2029	6061	2310	2331	1861	1860	3992
	stddev time	8,84	27,34	9,32	63,04	48,26	9,14	783,57	65,23	496,28	49,45
	stddev distance	2,26	8,35	5,17	5,75	1,62	0,16	3,32	0,41	1,99	0,34
Middle-aged	average time (s)	818	1578	1030	2698	877	1533	6592	1568	4474	5080
	average distance (m)	2199	2244	1941	1974	6042	2309	2336	1861	1860	3991
	stddev time	12,50	39,55	20,51	271,29	39,64	62,75	607,05	63,12	446,79	91,91
	stddev distance	6,88	2,38	3,56	27,57	35,01	0,43	3,55	0,31	1,09	0,11
Elderly	average time (s)	1098	2417	1382	3369	1582	2348	7790	1950	5136	6835
	average distance (m)	2230	2253	1953	1971	6075	2313	2341	1861	1861	3991
	stddev time	363,08	109,97	21,61	246,09	311,76	2659,22	971,86	145,45	517,45	576,64
	stddev distance	23,48	4,23	21,61	32,18	41,73	13,98	4,66	145,45	1,50	0,14
Mixed	average time (s)	765	1567	1009	2858	943	1466	6427	1596	4288	5164
	average distance (m)	2181	2241	1940	1983	6065	2309	2337	1861	1861	3991
	stddev time	374,33	40,75	12,38	293,59	53,00	2539,28	730,12	85,55	547,03	342,87
	stddev distance	19,93	4,66	3,67	24,09	27,67	13,63	3,60	0,49	1,74	0,26
Default	time (m)	602	1053	829	2223	698	1147	3761	1320	3497	4709
	distance (s)	2457	2524	1811	1677	6040	2312	2325	1862	1860	3992

Table 12 – Aggressiveness feature results

		Inverted Ant Colony algorithm						Shortest Time algorithm					
map		Lattice											
number of vehicles		5.000			10.000			5.000			10.000		
user percentage		0,25	0,75	1	0,25	0,75	1	0,25	0,75	1	0,25	0,75	1
low acceptance	average time (s)	1114	936	927	5229	3641	1820	1125	1009	1004	4009	3617	2358
	average distance (m)	2277	2143	2181	2302	2215	2050	2358	2389	2458	2346	2360	2397
	stddev time	28,38	53,35	100,43	674,13	1520,50	30,86	55,12	96,52	99,43	1281,36	1485,10	1032,22
	stddev distance	33,33	14,95	92,87	43,74	136,98	22,83	53,88	70,09	64,03	8,89	49,75	73,91
medium acceptance	average time (s)	1047	845	810	3157	1661	1623	978	886	903	2255	1639	1665
	average distance (m)	2259	2007	2220	2266	2224	2285	2356	2411	2408	2352	2258	2380
	stddev time	31,52	20,93	35,21	22,80	22,80	214,78	14,30	51,02	45,58	47,73	47,73	591,84
	stddev distance	8,90	190,61	31,94	99,45	99,45	93,43	12,46	45,15	41,24	118,30	118,30	52,67
high acceptance	average time (s)	980	758	659	2525	1469	1278	972	787	752	1865	1530	1402
	average distance (m)	2257	2283	2367	2337	2395	2426	2365	2397	2413	2365	2423	2399
	stddev time	17,11	23,87	15,94	74,29	26,83	55,87	25,82	19,22	41,10	62,40	54,37	733,89
	stddev distance	26,27	28,87	36,07	21,73	12,23	32,52	8,77	18,56	32,02	10,01	14,00	53,74
100% acceptance	time (m)	968	659	602	2354	1253	1053	921	729	679	1777	1532	1230
	distance (s)	2445	2907	2457	2459	2905	2524	2411	2425	2419	2522	3073	2399

Table 13 – Stubbornness feature: IACO and ST with different user percentages in Lattice map

		Inverted Ant Colony algorithm					**Shortest Time** algorithm			
map		**Lattice**		**Radial and Ring**		**Coimbra**	**Lattice**		**Radial and Ring**	
number of vehicles		**5000**	**10000**	**5000**	**10000**	**10000**	**5000**	**10000**	**5000**	**10000**
low acceptance	**average time (s)**	927	1820	1127	3417	3675	1004	2225	1083	3448
	average distance (m)	2181	2050	1778	1801	5656	2458	2413	1931	1833
	stddev time	100,43	30,86	129,40	775,86	146,82	99,43	942,54	60,55	493,66
	stddev distance	92,87	22,83	64,12	69,16	19,73	64,03	54,64	29,27	33,92
medium acceptance	**average time (s)**	810	1623	984	2178	2855	910	1664	1044	2348
	average distance (m)	2220	2285	1838	1767	5755	2404	2385	1915	1870
	stddev time	35,21	214,78	64,38	139,95	83,82	36,75	167,89	60,15	101,25
	stddev distance	31,94	93,43	31,92	10,96	55,34	41,08	19,45	21,47	31,41
high acceptance	**average time (s)**	659	1278	828	2195	1325	742	1405	888	2274
	average distance (m)	2367	2426	1969	1915	6065	2401	2389	1848	1707
	stddev time	15,94	55,87	21,21	181,09	42,60	39,15	42,98	22,19	30,09
	stddev distance	36,07	32,52	43,51	24,72	5,15	15,19	29,61	5,90	20,16
100% acceptance	**time (m)**	602	1053	829	2223	698	602	1053	829	2223
	distance (s)	2457	2524	1811	1677	6040	2457	2524	1811	1677

Table 14 – Stubbornness feature: IACO and ST in different maps