STUDIES OF TRAFFIC SITUATIONS USING CELLULAR AUTOMATA

Arto Hämäläinen

Laboratory of Physics
Helsinki University of Technology
Espoo, Finland

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Department of Engineering Physics and Mathematics, Helsinki University of Technology for public examination and debate in Auditorium E at Helsinki University of Technology (Espoo, Finland) on the 9th of October, 2006, at 13 o'clock.

Dissertations of Laboratory of Physics, Helsinki University of Technology
ISSN 1455-1802

Dissertation 142 (2006):
Arto Hämäläinen: Studies of Traffic Situations Using Cellular Automata
ISBN 951-22-8368-9 (print)
ISBN 951-22-8369-7 (electronic)

OTAMEDIA OY
ESPOO 2006

Abstract

The growing volumes of vehicular traffic necessitate new solutions to traffic problems. When changes to the road network are planned and traffic control systems are set up, it is usually not possible to test in advance the resulting effects in the real world. In such situations simulation models can be of great help.

Traffic simulation models can roughly be divided into macroscopic and microscopic ones. When macroscopic models examine the dependencies between traffic flow, traffic volume, and average velocities, microscopic models investigate the movements of individual vehicles. Microscopic models for vehicular traffic include car-following and particle hopping models — the latter are usually implemented as cellular automata (CA). Despite the simplicity of the CA models, the most important features of traffic flow can be modelled with them.

From the physical point of view, vehicular traffic can be seen as a nonequilibrium many-particle system with the flow of traffic fluctuating between different dynamical phases.

This work deals with simulation of vehicular traffic in urban and freeway environments. The behaviour of simple CA models is studied in connection with providing indicators — queue lengths at traffic signals, average velocities on different routes, and flow rates — describing traffic situations in a road network. The knowledge of traffic situations is important for traffic management and control, which aim in maintaining road traffic fluent, comfortable, and free of accidents and excess pollution effects. — The results also suggest how well the measuring procedure functions in generating the above-mentioned indicator values.

Road traffic is studied in different simulation test environments. The urban cases comprise two- and five-intersection areas with multiple lanes and traffic signal controlling. The freeway area corresponds to a stretch of a two-lane freeway with several on- and off-ramps. The vehicle arrivals are generated on a random basis, or they are gathered from measurements on real locations. The signal status changes use fixed period lengths. The performance of the CA models is examined against the functioning of a more refined simulation model that uses car-following approach. The simulation periods are typically from two to five hours in real time.

The results show that CA simulation usually produces traffic situation indicator values well when compared to those of the reference model, especially when averages over several runs are calculated. Separate runs may generate dissimilar outcome, which can create problems when used in on-line simulations. The results can be improved by adjusting the measuring technique (the detection of vehicles inside the network, for example).

The CA traffic models are also used to study the statistics of queue formation at some obstacle on a one-lane road. The probability for the effective queue length area seems to obey exponential decrease with increasing length values. The queue lengths grow rapidly with higher flow values.

Tiivistelmä

Ajoneuvoliikenteen kasvavat määrät vaativat uusia ratkaisuja liikenneongelmiin. Kun suunnitellaan muutoksia tieverkkoon ja asennetaan liikenteen hallintajärjestelmiä, ei aiheutuvia vaikutuksia yleensä pystytä etukäteen kokeilemaan reaalimaailmassa. Tällaisessa tilanteessa simulointimallit voivat olla suureksi avuksi.

Liikenteen simulointimallit voidaan karkeasti jakaa makroskooppisiin ja mikroskooppisiin. Kun makroskooppiset mallit tutkivat liikennevuon, liikennemäärien ja keskinopeuksien välisiä riippuvuuksia, mikroskooppiset mallit tutkivat yksittäisten ajoneuvojen liikkeitä. Ajoneuvoliikenteen mikroskooppisiin malleihin kuuluvat ajoneuvoseurantamallit ja hyppymallit, joista jälkimmäiset yleensä toteutetaan soluautomaateilla (SA). Yksinkertaisuudestaan huolimatta jälkimmäisillä voidaan mallintaa tärkeimmät liikennevuon piirteet.

Fysikaalisessa mielessä ajoneuvoliikennettä voidaan pitää epätasapainoisena monihiukkassysteeminä, ja liikennevuo voi vaihdella eri dynaamisten faasien välillä.

Tämä työ liittyy ajoneuvoliikenteen simulointiin kaupunki- ja moottoritieympäristöissä. Yksinkertaisten SA-mallien käyttäytymistä tutkitaan liikennetilanteiden — jononpituudet liikennevaloissa, keskinopeudet eri kulkureiteillä ja liikennevuon tiheydet — ilmaisijoina tieverkoissa. Liikennetilanteiden tuntemus on tärkeää liikenteen hallinnan ja säännöstelyn kannalta, näillä tähdätään siihen, että liikenne pysyisi sujuvana ja miellyttävänä, sekä siihen, että vältyttäisiin onnettomuuksilta ja ylimääräisiltä saasteilta. — Tulokset antavat lisäksi viitteitä siitä, kuinka hyvin käytetty mittaustapa toimii yllämainittujen ilmaisijoiden tuottajana.

Maantieliikennettä tarkastellaan erilaisissa simulointiympäristöissä. Kaupunkiliikennetapaukset käsittävät monikaistaiset ja liikennevaloin ohjatut kahden ja viiden risteyksen alueet. Moottoritiealue käsittää jakson kaksikaistaista moottoritietä, jolla on useampia liittymä- ja poistumaramppeja. Ajoneuvojen saapumiset luodaan tapahtuviksi satunnaisin välein tai ne otetaan aidoissa ympäristöissä toteutuneista mittauksista. Liikennevalojen ohjauksessa käytetään kiinteää jaksotusta. SA-mallien toimintaa tarkastellaan vertailuin yksityiskohtaisemman, ajoneuvoseurantaan perustuvan simulointimallin toimintaan. Simulointiajat ovat tyypillisesti kahdesta viiteen tuntia reaaliajassa.

Tulokset osoittavat, että SA-simulointi yleensä tuottaa liikennetilanteen ilmaisuarvoja vertailumalliin nähden pätevästi, etenkin laskettaessa useamman ajon keskiarvoja. Yksittäiset ajot saattavat aikaansaada toisistaan eroavia tuloksia, mikä voi olla ongelma ajantasaisessa simuloinnissa. Tuloksia voi parantaa säätämällä mittaustekniikkaa (esim. ajoneuvomäärien mittaus verkon sisällä).

SA-malleja käytetään myös tutkittaessa jonojen muodostumisen tilastollisuutta yksikaistaisella tiellä sijaitsevan esteen seurauksena. Toteutuneen jonoalueen pituuden todennäköisyys näyttää olevan eksponentiaalisesti vähenevä pituusarvon kasvaessa. Jonojen pituudet kasvavat nopeasti korkeammilla vuoarvoilla.

Preface

This thesis has been prepared in the Laboratory of Physics at the Helsinki University of Technology during the years 2001-2006.

I would like to express my deepest gratitude to my instructor and supervisor Academy Professor Risto Nieminen and to my instructor Docent Iisakki Kosonen (from the Laboratory of Transportation Engineering at the Helsinki University of Technology) for their efficient guidance. I also wish to thank Professor Matti Pursula (from the Laboratory of Transportation Engineering) in helping to choose the subject for the thesis and Docent Mikko Alava for his help concerning the literary background.

For good working facilities and pleasant working atmosphere I want to thank the personnel of the Laboratory of Physics. Furthermore, I am grateful to the personnel of the Laboratory of Transportation Engineering for useful information concerning traffic engineering.

I am most appreciative for the financial support given to this work by the Finnish Cultural Foundation and the Jenny and Antti Wihuri Foundation.

Also, I would like to thank my relatives and friends for their support.

Espoo, September 2006

Arto Hämäläinen

Contents

Author's contribution

The author has performed all the programming, simulations, and the interpretation of the results reported in his thesis. He is also responsible for writing the text of the thesis.

Chapter 1

Introduction

1.1 Problems in vehicular traffic

In modern societies the need for people to move between separate places at least during working days is rather a rule than an exception. Our homes are usually situated in different locations than the jobs, and so are shops and other premises related to services with a tendency towards fewer and larger units easily causing longer journeys for more and more people to reach them. Also, many facilities for entertainment and spare time require travelling even large distances. Although the means for moving are various, most of the everyday trips and journeys are performed on roads and railways. Issues of this study are solely related to normal road traffic done by vehicles, and for simplicity only passenger cars are used in the studied environments.

The growing of traffic volumes demands new solutions from traffic engineering and traffic science in order to be able to accommodate to the changing requirements. Building new and safe routes for traffic is expensive, and in many areas of the world there is already lack of space. Although this concerns primarily urbanised areas, the lining out of new, multi-lane highways may be a difficult problem even through thinly populated areas. In cities even widening of existing streets may be impossible because of the local housing stock. Consequently, often the only option is to intensify the use of the existing road networks.

More efficient usage of the existing road infrastructure is related to the management and controlling of traffic flows, not forgetting safety and convenience for the people travelling there. Hardly anyone wants to spend time in vehicle queues if another route, even a longer one, is available that might reduce the total travelling time. Nowadays, the management of traffic in urbanised areas and on heavily used long-distance routes is becoming more important. This is also closely related to informing ordinary road users about traffic issues both in advance and during the journeys.

The management of traffic including the gathering and making available the information concerning it has lately become more easy because of the development of modern technology. Although many principals and instruments used in acquisition of traffic data have been in use for a long time, the transferring and processing of that information would not be possible without fast data transmission and computers. Also, personal cellular phones and communicators and miniature, built-in computers in vehicles are gaining popularity in the usage of data exchange related to traffic.

The controlling and management of traffic covers the whole extent of the road network, not just local areas. Events and circumstances — constituting the traffic situation — on a certain portion of a road may differ considerably from those in other parts of it. However, the incidents in a certain area may soon create consequences into other parts of the road as well. The knowledge of the real time traffic situations, at least the difficult ones, in different parts of the network would be useful to other travellers and to the traffic management in all.

Getting information from various parts of the road network requires investments in different locations. Since the network often covers vast areas, the acquiring and maintenance of the equipment easily becomes costly, and making the number of necessary installations as low as possible would be desirable. Of course, even more resources have to be spent when planning and building new roads or making changes to the existing ones. For both of these purposes, research work that aims into finding favorable but sufficient solutions is of high importance.

The research of traffic uses various types of models in finding solutions to the existing problems. Some of these models use simulation of road traffic to get a collective picture of the probable effects caused by the planned changes to the network. Models of this type can also be used in connection with traffic management systems to reduce the number of flow measurement spots in the network.

Models require theory to base the model on. Theories concerning traffic flow include fluid-dynamical and gas-kinetic approaches on the macroscopic side and car-following theory on the microscopic side, all of which use continuous time and space. Cellular automata models also belong to the group of microscopic models, but they use discrete time and space in the calculations. The choice of the used model may be difficult; trade-off between simplicity and reality of the model is usually guided by the specific situation where the model is used in.

In the case of simulation models, one factor effecting the choice of the model is the speed of it. When the model is used in connection with planning, the temporal requirements for getting simulation results are usually not very tight. However, when the model is used in traffic management, it is desirable that the simulation describes the present situation of the traffic in the network; often it may also be helpful if the model can predict upcoming events by 'simulation into the future'. The need for real time simulation is getting more important in the traffic management of today.

Traffic data are normally gathered with instruments giving local information concerning vehicle existence and (potentially) velocity at certain positions at distinct moments in the network. However, what often is needed is the density and velocity distribution as a function of position along the network roads to get a general view of the current traffic. A good candidate to fill this need is a microscopic traffic model (like a cellular automaton), which generates vehicle traffic in its own image of the network according to the detector data acquired from real traffic. The model parameters are subject to adjustments during simulation since the vehicle amounts, turning rates at intersections, and velocities, for example, may differ from those that the model predicts. The less parameters the model contains the easier it is to handle them and the calculations related do not consume too much simulation time.

This study in mainly related to the management of traffic. It can be seen as one part of the so-called DigiTraffic research program that aims in producing advanced telematics services concerning road traffic [40, 76]. These services can be used in management and controlling of traffic as well as producing information that is available for ordinary road users. The program pursues a comprehensive modelling system of road traffic collecting information from different sources and using various calculation models (simulation, neural networks, and fuzzy logic) to produce parameters describing traffic. These can be used as a basis for producing telematics services and enhancement of traffic management and safety. Another purpose for the program is to promote the development of traffic modelling.

Other related programs are FITS and its follower AINO. The FITS program (Finnish Research and Development Programme on ITS Infrastructures and Services) aimed in developing public and private ITS (Intelligent Transport System) or transport telematics services and the related information infrastructures [77]. The purpose of the program was to improve the prerequisites for services and networking necessary for ITS and the emergence of services related to transport telematics needed by private persons, authorities, and companies. The FITS program is continued by AINO, whose goal is to develop the collection, management, and utilisation of real time information in transport systems [75]. It aims, for example, in increasing the attractiveness of public transport by providing real time information and route guidance, improving the management of deliveries, and in developing the cost-effective, real time monitoring of transport network status information. — Earlier works related to the subject can be found, for example, in [16] and [10].

1.2 Viewpoint

The viewpoint to traffic issues used in this study is partly from the generic physical side of the problems and partly from the traffic engineering side. The collective features of traffic are often studied by traffic scientists who are dealing with universal features found in traffic, like phases of traffic flow and changes of these phases, usually involving large amounts of vehicles. These features have a lot in common with other physical phenomena found in gases, liquids, and granular solid materials. This side of traffic is examined with the help of statistical physics. On the other hand, management of traffic issues in the infrastructure is handled by traffic engineering including the planning of the road networks for vehicles and pedestrians and the designing of traffic controlling and management systems (the use of traffic signs and signalisation, routing, and informing).

Both of the above-mentioned categories are usually studied with models since traffic projects are often expensive and substantial in volume. Also, experiments in real environments to support planning are practically impossible to arrange. The functioning of traffic models has to be tied somehow to real traffic, and this is done by adjusting the model parameters so that some characteristics of the model traffic match as well as possible the corresponding ones measured from real traffic. This study uses models to examine traffic problems and the connections to real traffic are left fairly loose.

The issues attacked in this study are related to both generic features of traffic and the engineering side of it. In focus is one type of simulation models used with vehicular traffic surveys: cellular automata. These models are used here to examine queue building as a generic feature in front of some obstacle (like red signal), and secondly, the models are tested for reproducing traffic situations in intersection and freeway environments against a reference.

The cellular automata models in question are extremely simple simulation models, which has necessitated the removal of many fine-featured aspects from the behaviour of the driver-vehicle units including most of those related to human behaviour. These simple models enable, though, quick run times, which is a very useful feature when the vehicles quantities become large.

1.3 Purpose of this study

This study deals with modelling of vehicular traffic by simulation with cellular automata models. The performance of these models is surveyed by assessing the validity of a few traffic parameters produced by them. These parameters can be used to illustrate the prevailing traffic situations in localised regions, which again can serve as data for description of the state of traffic on larger scales.

The chosen models use microscopic approach to traffic modelling so that movements of individual vehicles are distinguished in the process. The use of microscopic models appears more natural in situations that involve local information, especially urban streets and intersections, where vehicle quantities may occasionally become very low and the use of more coarse models might be problematic.

One purpose of the present work is to systematically study the applicability of the chosen models to produce reliable traffic parameter values related to traffic situations. The goodness of these values is estimated by comparing them to the values produced by another, more sophisticated microscopic simulation model. More genuine reference material would be the data gathered from real traffic, but the setting up of the test areas in natural environments is usually difficult and the data acquisition may be expensive.

Another purpose of the study is to examine the systematics of measurement and modelling. Traffic data can be collected by different methods — photographing, detector data, probe cars, etc. — depending on the special purpose. The gathered data usually give information concerning vehicle positions in the network and vehicles' velocities. The means for traffic data acquisition selected for this study are detectors (of vehicle positions) and indicators of traffic signal status changes. The detectors and indicators are situated in the reference model, and the data are gathered over different time periods. Later, the studied models receive the collected information and try to generate traffic situations in their own world that fit the measured data. How successful they are in this attempt depends mainly on the placement of the detectors and the model characteristics and parameters.

The traffic situation indicators used in the study are particularly queue lengths at traffic signals and average vehicle velocities. Traffic flow information is studied in some cases from this perspective, too. The results will also illuminate the applicability of the chosen measurement procedure (detectors, signal data) in producing these indicators.

The attainable speed of the simulation process is of great concern when on-line simulation is required. Nowadays, the traffic management and control often needs particularly up-to-date information from various parts of the road network. Cellular automata models are known to be fast enough for on-line simulations, and so the results of the study will give indications on whether they are otherwise suitable for producing valid traffic situation data valuable also for this purpose.

The study also explores queue building at some obstacle from the statistical point of view as an example of generic features of traffic. The chosen models are particularly suitable for this kind of tasks since they require large amounts of unrelated measurements that easily consume a great deal of simulation time when carried out by some other model types (not to mention getting the data from real traffic). The queue building is examined as a function of the intensity of traffic flow by varying the modelling circumstances.

1.4 Summary of contents

The study is principally divided into five chapters. **Chapter 1** is related to general issues of the study: common problems met in vehicular traffic, the viewpoint to traffic matters used here, and the purpose and scope of this study.

Chapter 2 discusses concepts of vehicular traffic and phases found in traffic flow outside urban areas — the viewpoint is more on the statistical physics side of traffic. Different ways to model vehicular traffic are presented in brief. Cellular automata models are covered in more detail, and their use is reported in diverse modelling environments.

In **chapter 3** the methods of the study are introduced: the problem setting, methodology, the programs used, and the test arrangements. The simulator code built for the study realising the application of the cellular automata models is described on a general level, and so are the supporting programs used in traffic generation and processing of the results. The behaviour of the vehicles of the simulator is not compared to real traffic but to the traffic of another, sophisticated simulation program, which thoroughly models vehicle traffic in the true world. A brief description of this reference simulator is also included. Furthermore, the chapter reports the building of the model areas in the two simulators, some of the parameters used, and the traffic generation (vehicles and signalisation) for the simulations.

The test runs of the study use different kinds of model areas. Generic feature runs are performed on a long, one-lane road with a signal head at the end of it. The signalisation is used for cutting the traffic flow to build queues. The queue formation is examined on a statistical basis by using a few cellular automata models and varying the flow and the red signal length.

Urban traffic is studied in two model areas of the two simulators by comparing maximum queue lengths at traffic signals, average velocities on separate routes of the networks, and the traffic flow at a few detectors placed on some lanes of the case environments.

Freeway traffic is approached via a two-lane (in the same direction) road section including some on- and off-ramps: the average velocities of the model vehicles are compared between the two simulators.

Chapter 4 reports the results reached in this study. At the beginning of the chapter, the calibration of the model mostly used in the simulations is described and references to measurements in real traffic are made. Next, the measurements for the fundamental diagram for two of the CA models used in the study are described.

Of the results of the study, the simulation runs for generic features of traffic are firstly summed up. The queue building on a single lane at a traffic signal seems to obey exponential decrease as a function of queue length with the exponent depending on the flow rate and on the model used. The queue lengths found at a certain probability level (P(50%)) increase quickly when the traffic flow values get high suggesting a phase transition in the flow.

In simulations of urban traffic, the queue lengths at traffic signals of the cellular automata model are found to agree well with the reference traffic system when dealing with conditions outside congestion times. When the traffic is heavy and the queue lengths at signal heads grow, cellular automata results show significant variation on separate runs although the starting conditions are equal; the differences easily get large particularly on one-lane links. Average values taken over several runs, on the contrary, usually coincide well with the reference values. One factor that often deteriorates the results is routing: if the vehicle flow values are corrected after intersections, the queue length values are seen to improve. Comparisons of the average velocities between the two simulators behave much the same way as the queue lengths.

In urban intersection areas, the traffic situation indicator values (queue lengths, average velocities) match quite well the values of the reference model especially when averages over several runs can be calculated. The results for one particular run may, however, in many cases differ substantially from the reference, therefore on-line simulation may give false results.

Simulations in a freeway environment show the dependency of the cellular automata results on randomness at times of congestion when a simple model is used, although as a whole, the model seems to perform better than in the urban intersection cases. Even better results are reached by using a model variation that takes into account the density of traffic in front a vehicle.

The traffic situation indicator values (as the average velocity) in the freeway case seem to match the reference values even better than in the urban traffic case. Especially, if the shortcomings of the basic cellular automata models can be avoided, the method is well applicable to on-line simulation, too.

Chapter 5 draws together conclusions from the simulations of this study.

Chapter 2

Survey of Traffic Models

2.1 Models for vehicular traffic

To study vehicular traffic otherwise than from empirical observations, it needs to be modelled in some way. Although passenger car traffic is not much older than about one hundred years, traffic research has begun as early as in the 1930's. Vehicular traffic has been modelled with different methods. Some methods have in their time been more popular than others, but still many of them are being used today. The two sections of this chapter briefly sum up concepts used in describing vehicular traffic and some of the different modelling methods that are used with special emphasis on cellular automata models.

2.1.1 Concepts and empirical findings

Questions of traffic are covered by traffic science and traffic engineering. While traffic engineering deals with matters like planning and constructing traffic network systems and environments and questions related to traffic controlling, traffic science tries to find generic laws and dependencies possibly prevailing in traffic systems.

From the physical point of view a traffic system can be considered to be a nonequilibrium many-body system being able to exchange energy and particles (vehicles, for example) with the surroundings. The behaviour of this kind of systems is not governed by such general principles (laws of thermodynamics and statistical mechanics) as closed systems in equilibrium are (gases, liquids, and solids) [20]. The units of traffic systems are self-driven, moving with the aid of their own energy sources as distinct from driven particles whose movements are influenced by forces from outside (pressure, gravitation, electrical forces). Furthermore, the behaviour of traffic units is controlled by human decisions, which are usually ignored in studies of traffic, though (this also applies to the present study).

In nonequilibrium systems similar phenomena can be found as in closed, equilibrium systems. Systems that are far from equilibrium have dynamical phases corresponding to stable phases of equilibrium systems (gaseous, liquid, and solid states). These phases can change to each other in time or stay stable for long periods of time.

Dynamical phases are one thing that traffic science is interested in because nonequilibrium systems are theoretically not as well understood as equilibrium systems are [11]. Other important questions involved with nonequilibrium traffic system are: fluctuations near steady states, how steady states are reached when the initial state of the system is not steady, can self-organised criticality be found in traffic, are the phase transition points characterised with long-range correlations and scaling exponents of power laws.

Traffic phenomena can be found in one, two, or three dimensions. Vehicular traffic is usually considered in one dimension restricted to one lane at a time with vehicles occasionally turning to diverging directions or changing to parallel lanes. City traffic can yet also be seen as two-dimensional, as composed of grids of streets. Pedestrian traffic is often seen as two-dimensional. In this text only one-dimensional vehicular traffic is considered, allowing lane-changing.

With traffic it is not possible to make similar controlled, repeated experiments as when studying most other physical problems: organising changed conditions (new roads, lanes, ramps, traffic lights) can be very expensive and time consuming. Also, modifying road conditions for testing purposes in established environments can be very hazardous from the point of view of the road users. Thus, one way to learn about traffic matters is to collect empirical information of it by passive observations. The tools for this are primarily induction loops, infrared detectors, photographing, and the use of probe vehicles (moving among 'normal' vehicles and collecting and conveying information from the field). The most expressive way would probably be aerial (video) photographing, but this method is expensive and difficult to arrange. Video photographing from ground stations is much cheaper, and can be useful concerning restricted parts of a road network.

Induction loops are the most widely used devices in monitoring vehicular traffic. One loop gives information primarily about the time instance a vehicle hits it and the time the vehicle spends in the loop area; two successive loops can measure the vehicle velocity, too. From these data the flow rate, average vehicle velocity, vehicle length, time headway to the vehicle in front or behind, etc. can be calculated [45, 26]. The average velocity and flow values are expressed for time intervals lasting typically for a few minutes. Induction loops are used either as sources of information for local signal controlling or as devices for collecting network-wide data about traffic flow and velocities for monitoring and controlling purposes. In the latter case, the loop information needs to be transferred elsewhere for processing.

Since the detection conditions outdoors may vary considerably (light, moisture/rain, snow/ice), the equipment has to be able to cope with different circumstances. The volumes in vehicular traffic change substantially depending on the time of day, week, and year. Moreover, road surface conditions, curvatures and slopes, and the existence of on- and off-ramps affect the results. Also, the possibility of false signals from measurements has to be taken into account when using the collected information. It is therefore important that particular attention is to be paid to the time and place when empirical observations take place.

Besides observations gathered from real locations, traffic problems can be studied with simulation [57], and this method is often used in connection with traffic planning and controlling. When planning new routes, broadening old roads, and making changes to the allowed volumes of traffic, it is usually not possible to test in advance the planned changes in the real world. In such situations the use of simulation models provides a way to get a picture of the probable effects. Simulation models can also be used as a helping device for traffic management purposes, i.e. finding out what procedures in controlling techniques (e.g. traffic signals, signs, route guidance) are appropriate when trying to avoid situations such as the formation of a complete capacity breakdown [38]. Provided the simulation model is equipped with on-line updating of the vehicle flows at appropriate spots of the modelled area, animation of the vehicles on a computer display can give a valuable, real time image of the events in different parts of the studied area.

Important concepts related to vehicular traffic are flow, density, (average) velocity, gap, and time and distance headways [44]. Of these items velocity is the most easily understood, and it is usually expressed in kilometres per hour (km/h). Often it is practical to know the average velocity of vehicles on some road stretch because it describes the fluency of traffic.

The (traffic) *density* is defined as the number of vehicles on a certain stretch of road measured at a certain time instance, and it is expressed as vehicles per kilometre (veh/km). The *flow* is a macroscopic characteristic of traffic, but traffic scientists often use in the sense of the measurable quantity better referred to as *flow rate* or *flux*, which correspond to the expected number of vehicles in a time interval passing a certain site on the road. In this text flow usually refers to flow rate, and the meaning should be evident from the context. The flow is expressed in vehicles per hour (veh/h). — As mentioned before, the density and flow may vary considerably depending on the time instance when the measurement is performed and on the spot of road where the measurement takes place. The density is ordinarily quite difficult to measure especially on a longer road stretch (except by aerial photographing). Instead, the flow is easy to measure but it gives only local information.

Time headway, which is often abbreviated to headway, is the time interval between two successive vehicles passing a point, and it is measured, for example, from the time the front bumper of a leading vehicle hits the detector to the time the front bumper of the vehicle behind hits it. Time headway is usually stated in seconds. *Distance headway* (or space headway or spacing), on the other hand, refers to the spatial distance between two consecutive vehicles, again measured, for example, from the front bumper of the first vehicle to the front bumper of the vehicle behind. Of these two headway items, the time headway and respectively the flow rate are more often used in traffic engineering.

Measured traffic items can be used for monitoring, controlling, and making statistics. Of these three purposes, the controlling of traffic is becoming more and more important at least in urban areas and also on intensively used highways and freeways. Usually the aim of traffic controlling is to avoid traffic jams, but already heavy congestion arouses irritation among drivers and enhances pollution problems when low velocities are used for longer periods of time. Traffic controlling can be implemented by telematic solutions (changing speed limits and route guidance, for example). Information about traffic can contain reporting of road conditions, of roadworks (reduction of the number of usable lanes), and of present traffic volumes on different road sections, and making prognoses based on statistical information.

One of the key issues in controlling traffic is the recognition of the prevailing traffic situation in the area of interest. The traffic situation comprises knowledge about the flow, density, vehicle velocities, travelling times, and the number of stops the vehicles suffer on the link in question, especially density and average vehicle velocities are often used in this connection.

Information revealing the traffic situation on a link is usually collected with induction loops (time instances of vehicle passing and vehicle velocities). However, only a dense placement of detectors can give a reliable picture of the velocities on a link because they may vary considerably even on a short stretch of the road. If merely coarse placing of detectors is possible, average velocity information on the link can be obtained by identifying the vehicles at consecutive detector spots by licence number (photographing) or by induction profiles of vehicles (or vehicle groups) — also identification of cellular phones moving with vehicles between areas of base stations has been tested [46]. Still, the velocity variation problem inside the link remains.

Of the empirical dependencies in traffic science, the following are widely used: the fundamental diagram (flow rate versus density) and (average) velocity of vehicles versus density. These two are depicted in Fig. 2.1 in qualitative forms. The fundamental diagram describes the connection between density and flow rate on the road. When the density is low, that is, vehicles are far from each other, the flow increases linearly with increasing density. When the density reaches certain value, vehicles start to 'interact' with each other, drivers become cautious and lower their velocities to maintain a safe distance to the vehicle ahead. The lowering of velocities causes the flow to decrease. As the density still increases, vehicle velocities get lower and finally a point is reached when traffic is completely jammed and the flow rate drops to zero. — The diagram (b) in Fig. 2.1 shows the qualitative velocity-dependence of vehicles on the density.

In the theory of traffic flow, it is supposed that the average flow rate ($q(\rho)$) is related to the density (ρ) and the average velocity of vehicles ($V(\rho)$) as [28]

$$q(\rho) = \rho V(\rho). \tag{2.1}$$

The fundamental diagram in Fig. 2.1 is only qualitative and refers to a situation with vehicles moving with equal distances to each other and with the same velocity. In real traffic vehicles often have at least slightly differing velocities and distances and smaller or larger clusters of vehicles appear. The measured values of flow are usually received as averages from several short-term detection periods (typically from one to five minutes), so the plot one gets consists of dots more or less near to each other forming a 2-dimensional area in the figure — the plot resembles the qualitative picture only at low and very high density values.

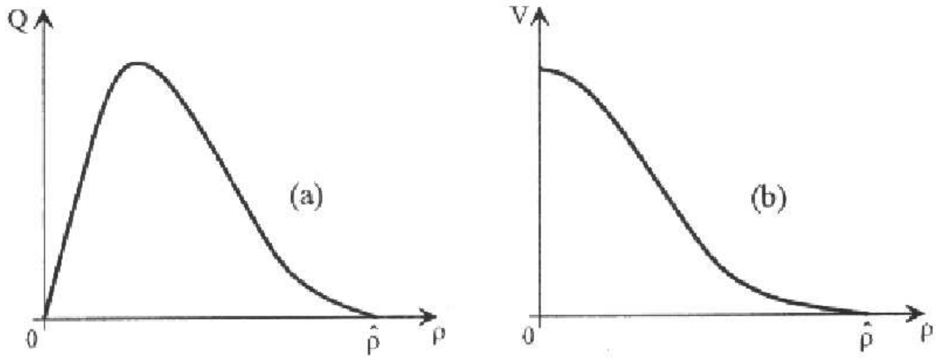


Figure 2.1: Qualitative forms (a) of the fundamental diagram (Q = flow, ρ = density, V = velocity) and (b) of the related dependence of the "maximal and out of danger" velocity of vehicles on the density. The term $\hat{\rho}$ is the maximal possible density of vehicles in traffic flow. Figure from Ref. [28].

The form of the fundamental diagram has been studied extensively and several shapes have been suggested for it [19]. Quite accepted is the conception that there is a discontinuity in the flow rate values at intermediate densities representing hysteresis. When starting from low values of density, the flow rate increases practically linearly with increasing density until, beyond a certain density value, the flow rate diminishes discontinuously and fluctuates in the area of intermediate densities. But when the examination is started from high density values towards low density ones, the flow does not get high and the free flow part of the diagram is reached using a different route than when moving in the other direction. In other words, around high flow rate values the fundamental diagram exhibits metastable states where the flow rate does not uniquely depend on the density. Even with a mean value for the flow rate at a given density, the actual form of the fundamental diagram for the free flow and high-density regimes is not very thoroughly known [11]. In Fig. 2.2 is depicted an empirical plot of the fundamental diagram from local measurements. Solid lines and arrows show transference between experimental points, which are average values gathered from one minute intervals.

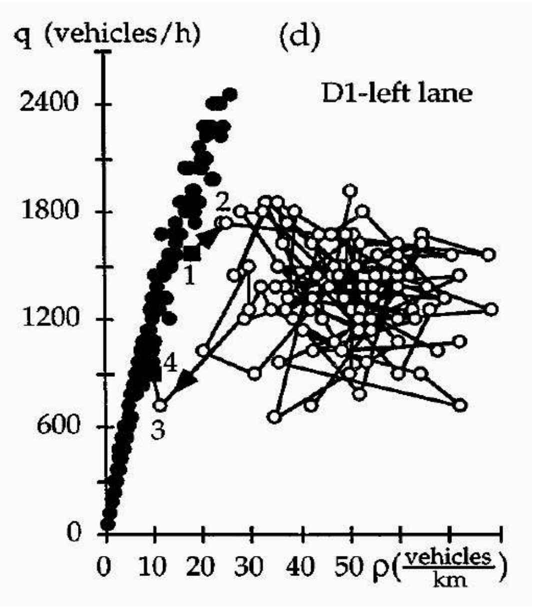


Figure 2.2: Experimental fundamental diagram: complex patterns of the flow-density relationship (q = flow, ρ = density). Figure from Ref. [33].

Other empirical observations of traffic flow contain time and distance headways. The time headway distributions depend largely on the road type, traffic phase, and the current density of vehicles on the road. The highway observations in [51] show that in free flow the distribution has a two-peak structure in different density regimes (ranging from 5 to 20 veh/km) with the first peak at about 0.8 seconds, which is approximately the reaction time of drivers (Fig. 2.3). This value is quite low and corresponds to risky driving behaviour leading to very high flow values in the metastable area of the fundamental diagram. The other peak is around 1.8 seconds, which corresponds to a safer and typical value near the recommended time headway of about 2 seconds. In congested traffic the 1.8 second peak is the typical value and very low headway values may disappear altogether.

The average velocity chosen by a driver depends on both the traffic phase and the distance headway to the vehicle ahead. The average velocity of vehicles with large distance headways in the congested phase (synchronised traffic) is significantly lower than that for the free flow phase, and it also stays on about the same level even for relatively large headways. The short distance headways which can be found in free flow traffic are possible when drivers anticipate the behaviour of the vehicles ahead, whereas in congested traffic this anticipation is less important [51].

The main dynamical phases of traffic flow are the *free flow phase* and the *congested phase*. In the free flow phase vehicles can move freely choosing their velocity as they wish in the framework of the prevailing speed limit. Often that choice is the maximum allowed velocity. In this phase vehicles practically do not interact with each other: all vehicles are so far from each other that different velocities do not cause interactions or passing may be accomplished without difficulty. The free flow phase refers to the linear, low-density part of the fundamental diagram in Fig. 2.1.

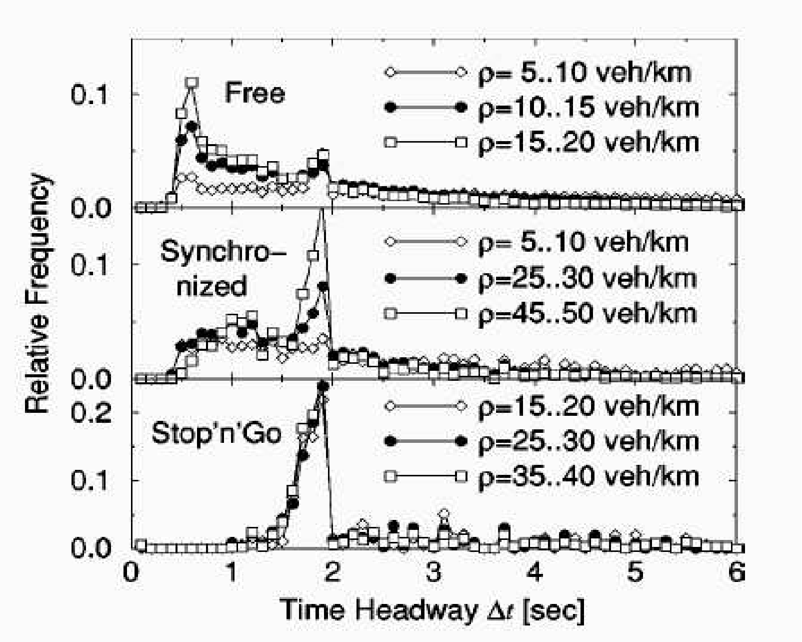


Figure 2.3: Time headway distribution for different density regimes, measured data from a German highway. Figure from Ref. [51].

The congested phase seems to be much more complicated than the free flow phase, and it has been considered to consist of several phases [51]. Congested traffic covers a large, two-dimensional area in the flow-density graph with an especially wide range of density values. Part of this area with high flow values may also overlap with values of the free flow phase.

The characteristics of the congested phase are under discussion, but this phase has often been divided into two separate phases: *synchronised traffic* and *wide moving jams* [26, 27], the latter is also often called *stop-and-go traffic*. The use of the terms is diverse depending on the researcher, but stop-and-go traffic rather depicts a sequence of short, moving traffic jams, inside which the velocity of the vehicles is low — even reaching the value of zero. Meanwhile, wide moving jams refer to longer jams with the widths of the jam fronts being much shorter than the width of the jam and the vehicle velocities being zero inside the jam, the jam yet propagates as a structure on the road. Synchronised traffic occurs usually at higher densities than the free flow phase, but flow rates do not reach as high values as in free flow —

the velocities, also, are lower. In the synchronised phase, strong correlations usually exist between vehicle velocities on different lanes of a highway. Furthermore, there is a tendency of velocity synchronisation between vehicles on the same lane of the road. Different states are also seen inside synchronised traffic: stationary and homogeneous flow with both the average velocity of vehicles and the flow being stationary over long periods of time, states with vehicles' velocity being quite stationary but the density changing somewhat in time, and states with the average velocity and the flow changing abruptly. These states are referred to with different names [32, 68]. Synchronised traffic is often met in connection with bottlenecks of different sorts (on-ramps, reductions in the number of lanes, etc.).

Synchronised states with varying average velocity and flow values often contain oscillating traffic features and stop-and-go waves with sequences of jams and free traffic in between them. Stop-and-go waves are characterised as being narrow moving jams and rather belonging to the phase of synchronised traffic than to wide moving jams [27].

Wide moving jams are characterised as having widths considerably larger than the widths of the upstream and downstream fronts of it and the velocity of the vehicles being zero inside it [27]. The jam moves upstream as new vehicles join it at the upstream front and others leave it at the downstream end. In empirical studies several other characteristics of jams have been observed in the case of wide jams if there are no hindrances in the outflow and when free flow is formed in the outflow [28]: a jam can move through a highway for a long time keeping its form and main parameters, the mean values of the parameters of the downstream front (the velocity of the front, the flow out of the jam, the density of vehicles in the outflow) are nearly the same for different wide jams, the flow out of a jam can be considerably lower than the flow reached in free traffic, and an almost stationary moving jam can exist on a highway.

The most examined phenomena in vehicular traffic in highway and city areas are congested traffic and the formation of jams. Reasons for congested flow and resulting jams are various: road capacity limits, intersections, traffic signals in urban areas, occasional limitations in lane usage (construction, accidents), influence of ramps, etc. Also, sometimes jams appear as if from nowhere and later disappear although no obvious reason for them is to be found ('phantom jams' [30, 31]).

The phase changes in vehicular traffic have been intensively studied lately. It is believed at the moment that free flow does not usually change directly to the phase of a wide moving jam, but transition occurs first to synchronised flow [27]. The reason for the free flow to synchronised transition is often some obstacle or bottleneck on the road (for example, vehicle flow from an on-ramp). In the synchronised phase (in the pinch region upstream of a bottleneck), moving jams travelling upstream can then appear and they possibly merge to form wide moving jams. Both the transitions (free flow to synchronised flow and synchronised flow to wide moving jams) are considered to be first order transitions. However, the transitions between different states in the synchronised flow are rather of second order.

Theories of vehicular traffic can roughly be divided into three separate groups: traffic flow theory, car-following theory, and particle hopping models [47]. Another way of grouping is to divide them in macroscopic and microscopic categories. — In macroscopic theories traffic is seen from the viewpoint of large scales and movements of individual vehicles cannot be seen separately. In the microscopic view, the movements of single vehicles are examined.

The *traffic flow theory* tries to find dependencies between traffic flow, traffic density, and (average) velocities of vehicles, which are interconnected through the relation flow = density times velocity. The traffic flow theory belongs to the group of macroscopic theories, and it contains fluid-dynamical and (gas-)kinetic theories of vehicular traffic, although kinetic theories can be seen to belong somewhere between the macroscopic and the microscopic views because the description of traffic starts from individual vehicles.

The *car-following theory* examines traffic from the microscopic point of view, from the behaviour of single vehicles. The actions of a vehicle (or the driver) are considered to be a response to different incentives including the driver's own will, actions of other (particularly leading) vehicles, distance headway, etc.

Particle-hopping models of vehicular traffic deal with individual vehicles whose dynamics is primarily stochastic. These models are usually implemented as *cellular automata*, and the viewpoint is again microscopic.

2.1.2 Different approaches to modelling

In the macroscopic **fluid-dynamical theories** of vehicular traffic, it is assumed that — when the flow is considerably heavy and only collective features of the flow are treated — traffic can be considered as a stream of compressible fluid (in one dimension). Then, the flowing of traffic can be described in a similar manner as movement of fluid in the hydrodynamic theory.

The theory is based on the conservation of the vehicle number (excluding on- and off-ramps in the first approximation). If the density of vehicles on the road is symbolised by $\rho(x;t)$ and the flow of traffic by $q(x;t)$ (with $q(x;t) = \rho(x;t)V(x;t)$ where $V(x;t)$ is the average velocity), an equation of continuity can be established between these quantities:

$$\frac{\partial \rho}{\partial t} + \frac{\partial q}{\partial x} = 0. \tag{2.2}$$

Because this equation has two unknowns, it is not possible to solve it without further information. In an early theory (of Lighthill and Whitham from 1955) it is assumed that flow is determined primarily by the local density $\rho(x;t)$ so that the vehicle velocity adapts instantaneously to the surrounding density without any relaxation time [47]. The solutions are so-called kinematic waves. In this theory density discontinuities (shocks) can appear as separating regions of high and low densities of vehicles. The theory cannot, however, explain moving localised clusters in traffic or stop-and-go waves.

In later fluid-dynamical theories (second-order macroscopic models), the assumption of the immediate adaptation of the vehicle velocity to changes in the traffic situation is replaced with more realistic dependencies. The continuity equation is accompanied with a Navier-Stokes-like equation of velocity changes involving terms for velocity relaxation to a 'maximal and out of danger' average velocity (which depends on the density), for anticipation about future traffic events, and for the adjustment of the velocity of a vehicle to that of the others.

If the road contains source and sink terms (on- and off-ramps), the continuity equation (Eq. 2.2) has to be modified so that the right hand side contains the appropriate terms. — The above-mentioned equations can also be used for two-lane highways by taking into account the lane-changing vehicles in the source and sink terms.

The **(gas-)kinetic theories**, too, belong to the group of traffic flow theories. Gas-kinetic theories can be considered as mesoscopic theories, and they act as a bridge between macroscopic and microscopic (car-following and particle hopping) models. — In the gas-kinetic theories vehicles are treated like molecules in one-dimensional gas.

In the kinetic theory of gases, the gas molecules are described by a distribution function for the number of molecules in a volume element at some point in space with momentum around some value at a certain time. The time-evolution of this distribution is described with the Boltzmann equation [11].

The earliest version of the gas-kinetic theory of traffic [56] is based on an equation resembling the Boltzmann equation. The distribution function f represents the probability of a vehicle being in a certain road element with a velocity around some value at a certain time. The theory suggests that, if there were no interactions between vehicles, there would exist an ideal, desired distribution to which f tries to relax. The distribution f changes with time either by relaxation (if f differs from f_{des} for the same x and v, there is a desire to return to the ideal distribution) or because of interactions between vehicles. The process can be described by a kinetic equation:

$$\frac{\partial f}{\partial t} + v\,\frac{\partial f}{\partial x} = \left(\frac{\partial f}{\partial t}\right)_{rel} + \left(\frac{\partial f}{\partial t}\right)_{coll}. \tag{2.3}$$

The term $\left(\frac{\partial f}{\partial t}\right)_{rel}$ refers to the relaxation to f_{des} and the term $\left(\frac{\partial f}{\partial t}\right)_{coll}$ is the interaction term.

The shortcomings of this theory (discontinuous velocity jumps, the desired velocity distribution not taking into account different driver profiles) were tried to be corrected in the Paveri-Fontana theory [54]. In this theory it is suggested that each driver has an individual desired velocity to which the actual velocity tries to relax when there are no interactions with other vehicles. The phase-space distribution is a function of yet another variable, individual desired velocity, and alterations are made especially to the relaxation term. Later, other modifications to the model have also been suggested involving alterations particularly to this term [20].

Gas-kinetic methods have also been applied to multilane highways [21] and to some extent also to two-dimensional traffic.

In the **car-following theories**, the point of view to vehicular traffic is microscopic. The behaviour of every vehicle is separately described by an equation of motion. The behaviour of different models within the theory is based on the reactions of a follower vehicle to the actions of the vehicle in front of it. The reactions depend particularly on the vehicle velocity, on the vehicle's distance to the leading vehicle, or on the velocity difference between these two vehicles.

In an early car-following model (from the nineteen fifties [55]), the *follow-the-leader* model, a vehicle's reactions are brought about by the velocity difference to the next vehicle ahead. The acceleration of vehicle n is given as [55, 11]

$$\ddot{x}_n(t) = \frac{1}{\tau}\left[\dot{x}_{n+1}(t) - \dot{x}_n(t)\right], \tag{2.4}$$

where $n+1$ refers to the leading vehicle, τ is a parameter (the inverse of which can be seen as a sensitivity coefficient). In this model it is assumed that a vehicle tries to maintain the same velocity as the vehicle in front of it. The relation can be derived from the assumptions that the distance headway to the vehicle ahead should grow when the velocity gets higher and that a vehicle must have a safe distance to the leading vehicle.

A modification to this model is the use of a response time lag [7, 11] — the follower vehicle reacts to the actions of the leading vehicle after a certain time T as:

$$\ddot{x}_n(t+T) = S\left[\dot{x}_{n+1}(t) - \dot{x}_n(t)\right], \tag{2.5}$$

where S is a sensitivity constant. The time lag constant T consists of three things [7]: follower driver's perception time (observation of the change in the leading vehicle's acceleration), driver's response time, and the response time of the vehicle.

This model has been further modified to take into account the influence of the distance headway to the sensitivity coefficient [17, 11] or to take into consideration not only the actions of the leading vehicle but also the actions of the next nearest vehicle ahead [11].

One class of the car-following models are the so-called *optimal velocity models*. In these models the driver tends to adjust the velocity of the vehicle to an optimal velocity, which depends on the distance headway to the vehicle ahead (in contradiction to adjusting it to the velocity of the leading vehicle) [11, 2]. The dependency can be expressed as

$$\ddot{x}_n(t) = \frac{1}{\tau}\left[V^{opt}(\Delta x_n(t)) - \dot{x}_n(t)\right], \tag{2.6}$$

where $\Delta x_n(t) = x_{n+1}(t) - x_n(t)$. The term $V^{opt}(\Delta x_n)$ must have an upper value when the distance headway goes to infinity, several different expressions have been suggested for this function [11]. Since the expression now involves also distances instead of only velocities, second order differential equations are needed, and the analytical calculations become much more difficult.

Also several other models of the car-following type have been introduced over the years [20]. One example is the *intelligent driver model*, which tries to describe the behaviour of drivers more closely [69, 70]. The acceleration for vehicle n is described by

$$\ddot{x}_n(t) = a\left[1 - \left(\frac{\dot{x}_n(t)}{v_0}\right)^{\delta} - \left(\frac{s^*}{s_\alpha(t)}\right)^2\right]. \tag{2.7}$$

Here the term $a\left[1 - \left(\frac{\dot{x}_n(t)}{v_0}\right)^{\delta}\right]$ describes acceleration on a free road and $-a\left(\frac{s^*}{s_\alpha(t)}\right)^2$ describes braking deceleration. The term a is the maximum acceleration, v_0 is the desired velocity, δ is a constant with a value typically from 1 to 5, and $s_\alpha(t)$ is the distance headway to the leading vehicle discounting the vehicle length. The term s^* is the “effective desired distance” depending on, for example, the velocity of the vehicle, the velocity difference to the leading vehicle, the minimum “jam distance” to the leading, standing vehicle, the safe time headway in congested but moving traffic, and the ’normal’ deceleration. The term ’intelligent’ refers to braking strategy: drivers brake harder in critical situations than normally.

2.2 Cellular automata models

2.2.1 Basics of cellular automata

In this subsection some general features of cellular automata (CA) models are discussed although the calculation procedures for CA traffic models often differ from the treatment of standard CA.

The basic features of a cellular automaton can briefly be expressed as follows [9]:

- a regular lattice of cells in a d-dimensional space is set up for the system
- a set of Boolean variables is attached to each site giving the state of the cell at each time step
- a rule describing the time evolution of the states is defined

The rule is the same for all cells and it is applied to the cells in parallel (simultaneously). Normally the new state is a function of the previous time step only, but sometimes it is necessary to take into account the states of (some) earlier time steps, too. In simulation more memory is then needed for storing older states of the cells.

Because the CA rules are usually local, updating the state of a certain cell requires the knowledge of the states of the cells in the vicinity of it. This vicinity is called **neighbourhood**. Usually this neighbourhood is very restricted containing only a few cells (including, for example, 4 or 8 nearest neighbours in the two-dimensional case), otherwise the rule may become too complex.

Because in practice the simulation of a CA rule requires that the lattice cannot be infinite, certain **boundary conditions** have to be applied at the lattice edges. One possibility for managing the boundary cells is to apply a special rule for these sites, a rule that is different from the rule applied to 'normal' cells in the lattice. In this method it is also possible to use different rules at different boundaries. Another possibility is to define virtual cells beyond the boundaries, and the updating rule remains the same for all cells. — In the often used case of periodic boundary conditions, the one-dimensional lattice forms a ring.

The usual way to update the cells in a lattice is parallel (synchronous, simultaneous) updating. In some cases also other updating methods are used (e.g. random sequential).

Cellular automata are divided into two categories: **deterministic** and **probabilistic** (also called **stochastic**) [1]. The definition at the beginning of this subsection refers to a deterministic CA, and it bears the fact that the time evolution of the states of the CA is always the same if the starting configuration is the same. With probabilistic CA the new value of a cell state in updating depends on the values of the cells in the neighbourhood and on a pre-determined probability to get a certain new value. Thus, starting from a particular configuration, a probabilistic CA may evolve into totally different configurations in separate simulation runs.

In computer simulations the updating of the cell states may be implemented in different ways. The most straightforward way is to program the rule as it is and calculate the value every time step for every cell anew. In some cases it may be advantageous to make a **lookup table** in which the outcome value is calculated for every possible neighbourhood configuration in advance. — In **multispin coding** the lattice sites are packed in bits of a computer word and the rule of the automaton is then applied by bitwise logical operations — in this way many sites (depending on the computer word length) can be handled in one or a few computer instructions.

It seems that all one-dimensional CA (perhaps also others) fall into four distinct universality classes [74]. These classes characterise the attractors in the evolution of cellular automata configurations. Attractors are a family of certain states to which the trajectories in the configuration space evolve after many time steps having started from almost any initial state. Different classes can be characterised as follows:

– *class 1*: after a finite number of time steps, almost all initial states evolve to a unique homogeneous state, in which all sites have the same value (a 'limit point')

– *class 2*: a cellular automaton of this class generates in most cases separated simple structures from particular (typically short) initial site value sequences (the automaton serves as a 'filter'), the structures generated are either stable or periodic

– *class 3*: evolution from almost all initial states leads to chaotic, aperiodic patterns ('strange attractors')

– *class 4*: in most cases all sites attain value 0 after a finite number of time steps, but in few cases stable or periodic structures are formed, which persist for an infinite time — in some cases also propagating structures are formed.

Statistical quantities characterising cellular automata behaviour [73] are in simple cases the densities of sites or blocks of sites with particular values. Other quantities include the calculation of entropy and the use of correlation functions (describing the interdependence of the values of separate sites). Furthermore, power spectra or Fourier transforms can be used as statistical measures of CA configurations.

Cellular automata have a strong similarity with complex systems although their elementary dynamics can be quite simple [9]. The behaviour of a system may depend only little on the details of the interactions between its elementary components. The complex behaviour of the macroscopic world coming from collective behaviour rather than from some specific characteristics of microscopic interactions may well justify the simplification of the microscopic laws connected to the phenomenon if these laws are not relevant at the macroscopic level of observation. In this kind of situations cellular automata type simplifications may be very desirable.

2.2.2 CA models for single lane

The microscopic rules of CA may give an intuitive approach to modelling a physical phenomenon compared to modelling in more customary ways, for example, by differential equations. The modelling by CA requires that the rules governing the updating take into account the essential aspects of the phenomenon but in the simplest possible form. — Because the CA rules often use Boolean operations, the updating can be accomplished exactly and no truncations of numbers are necessary contrary to manipulating floating point numbers when solving differential equations. Also, a CA approach may be natural when complicated boundary conditions are present.

Cellular automata models for vehicular traffic are sometimes categorised as belonging to a group called particle-hopping models. Models in this group usually form a one-dimensional lattice of finite length. In this lattice every site may or may not contain one and only one particle (which in connection with traffic corresponds to a vehicle). The particles 'hop' from one site to another (empty) site, usually in one direction. The sites between the old and the new positions must be empty.

In CA models the time and space are discrete, and accordingly, velocity, acceleration, and deceleration have discrete values. The movements of the particles in particle-hopping models are governed by local transition (update) rules, which usually contain stochasticity. The particles are often identical and the same transition rules apply to all of them.

The update order specifies the way the transition rules are applied to particles of the model. In *parallel updating* all the particles are updated during one time step. In *random sequential updating* one particle is chosen at random and its state is updated. After that a new particle is picked and updated — the new particle can be the one updated just previously. Random sequential updating is usually applied with particle-hopping models whereas cellular automata models for vehicular traffic use parallel updating. The use of a specific updating method can have a dramatic effect on the dynamics of the model [47].

In vehicular traffic studies, the particle-hopping model with parallel update order is formally a cellular automaton [47]. The earliest use of cellular automata in association with traffic studies is from the 1950's by Gerlough. Later the idea was developed by Cremer and Ludwig [13]. In the 1990's two separate CA models [49, 6] started a major interest in questions of traffic among statistical physicists. These CA traffic models are one- or two-dimensional. The other of these models describes traffic in one- or multi-lane roads, and the other may be used in connection with city areas where a two-dimensional grid describes the street network.

In the one-dimensional CA for traffic, a (one-lane) road is imagined to be divided into short stretches, cells, 'through' which the vehicles move when they advance on the road. The length of one cell is about the length one vehicle occupies when it stands in a jam (the length of vehicle plus some space in front and behind) — often 7.5 metres is used for the length of the stretch of a real road. The velocity of the vehicle determines how many cells it advances during one time step. Acceleration and deceleration also happen instantaneously between time steps. — The state of one cell is primarily comprised of two quantities: an item showing whether a cell contains a vehicle (or not) and an item showing the velocity of the vehicle is if there is one in the cell. — In the early models, the velocity had often only binary values (stopped or moving), but nowadays, at least in models describing traffic outside urban areas, the velocity has more than two values. The length of the cell, maximum velocity, and the time step length are interconnected to give a proper picture of reality.

The update rule determines what will happen to a vehicle during the present time step. The rule tries to make the movement of vehicles as natural as possible, yet it is advantageous to make the rule simple enough so that an updating step would not take too long in computer time. When parallel update order is used, the 'state' of the road, i.e. the positions and velocities of all vehicles, are stored at the end of the time step. From these values a new set of positions and velocities for every vehicle is then calculated during the next step and stored in a temporary array during the updating so that they do not mix with the previous values.

The update rule for one time step usually consists of a few (sub)steps to calculate the new velocity and the new position of every vehicle. A possible change in the velocity is determined on the basis of the present velocity and the interaction with the next vehicle ahead. Stochasticity in velocity changes (usually in the form of deceleration) is used to implement driver behaviour for temporary fluctuations in the velocity. The new position is finally calculated with the help of the new velocity. The order of the substeps is important to achieve the desired effects.

There are nowadays many different CA rules to be used in traffic studies, but not one single rule with which all the features of vehicular traffic could be modelled [11]. Common to these rules is that they are fast to run even with workstation type computers. Also, because parallel updating is used, the programs can be designed to run on a parallel computer and very high simulation speeds can be reached in this way.

With computer simulations it is possible to get sufficiently accurate quantitative results when analytical results cannot be reached [11]. Several analytical calculations for CA models have been made, but they are mostly restricted to situations when velocity maximum is low, like 1 or 2 [61, 66, 65, 63, 64]. Order parameter and correlation function calculations can be found in [15, 8].

Much of the interest in cellular automata usage for traffic simulations dates back to the beginning of the 1990's when Kai Nagel and Michael Schreckenberg introduced their cellular automata model for freeway traffic [49]. The benefits of the model are its simplicity, suitability for computer simulations (also for parallel computing), and quick updating of simulation steps even for larger simulation arrays. Yet many features of the real vehicular traffic flow can be captured with it.

The original Nagel-Schreckenberg model (= NaSch model) consists of a one-dimensional array of L sites with open or periodic boundary conditions. Every site of the array may be occupied with one and only one vehicle or it may be empty. Every vehicle has a velocity of integer value between 0 and v_{max}. The vehicles move in one direction from array site to another obeying an update rule, which consists of the following four (sub-)steps [49]:

1) **Acceleration:** if the velocity v of a vehicle is lower than v_{max} and if the distance to the next vehicle ahead is larger than $v + 1$, the velocity is advanced by one $[v \rightarrow v + 1]$

2) **Slowing down (due to other vehicles):** if a vehicle at site i sees the next vehicle at site $i + j$ (with $j \leq v$), it reduces its speed to $j - 1$ $[v \rightarrow j - 1]$

3) **Randomisation:** with probability p, the velocity of each vehicle (if greater than zero) is decreased by one $[v \rightarrow v - 1]$

4) **Vehicle motion:** each vehicle is advanced v sites.

The four steps described are performed for all vehicles in parallel (inside one time step). The steps describe vehicular traffic on freeways quite realistically: drivers try to drive as fast as possible within the allowed speed limit (v_{max}) and slow down if they come too close to the vehicle ahead so that a crash can be avoided. The randomisation step takes into account three different properties of human driving [47]: fluctuations at maximum velocity, overreactions at braking, and noisy acceleration. It turns out that overreactions in braking are crucial for spontaneous jam formation [62]. — The order of the steps is important, too. If, for example, the order of steps 2 and 3 is changed, no overreactions can occur and consequently no spontaneous formation of jams. The parallel updating (instead of random sequential updating) takes into account the reaction time of drivers and may lead to a chain of overreactions. The jam formation mechanism cannot be modelled with the random sequential updating.

The NaSch model is a probabilistic cellular automaton, which in the case of $v_{max} = 1$ and the deterministic limit (randomisation probability $p = 0$) is equivalent to CA rule 184 in the Wolfram notation [72].

Two simulation configurations of the NaSch model arrays are seen in Fig. 2.4. Above the upper right corners of the cells, the corresponding state values are indicated for the cells as a consequence of the previous time step: the first item indicates whether the cell is occupied with a vehicle or not (1/0) and the second item indicates the velocity of the vehicle (if it exists).

The cell length in the model is usually in reality 7.5 metres and v_{max} is often 5 (cells per time step). The generally used time step length is one second. With these values the maximum velocity corresponds to 135 km/h in real traffic. The length of the array is typically a few thousands of cells at minimum. Typical values for the randomisation parameter p are $0.01 \rightarrow 0.50$.

The NaSch model is a basic model of vehicular traffic — all the four steps of the rule are necessary to reproduce the basic features of real traffic [62]. Simulations with the model show that it can reproduce jam formation on a highway, which is caused

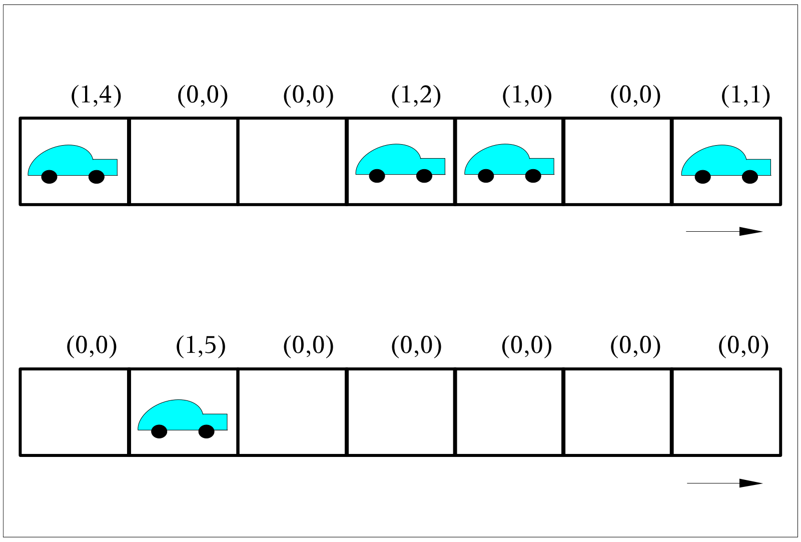


Figure 2.4: Two stretches of cell arrays of the NaSch model with the state values (occupation,velocity).

by vehicles' velocity fluctuations. The jams appear as clusters of vehicles producing start-and-stop waves on the road.

A particle-hopping model that is closely related to CA models for vehicular traffic is the (one-dimensional) **asymmetric simple exclusion process** (ASEP) [47, 62] whose particles interact by hard-core exclusion. Also this model uses a lattice, which is set up with sites empty or occupied with (usually) one particle. The rule for the behaviour of ASEP is: 1) choose randomly one particle, 2) if the site to the right of it is empty, the particle can move to it with a given probability or deterministically (particles may also be allowed to move in two directions). The boundary conditions for the model can be periodic or open, and the updating method sequential or parallel. One special case of the model is the totally asymmetric simple exclusion process (TASEP) where the movement is only in one direction, normally open boundaries and stochastic dynamics are used. The TASEP is a very simple prototype of driven, nonequilibrium systems. Fig. 2.5 presents the principle of the TASEP model: the parameter α defines the probability to insert a particle from the reservoir to the first cell of the lattice, the parameter β stands for the probability to move a particle from the last cell of the lattice, and a particle is moved to the next cell inside the lattice with probability q.

The TASEP model is closely related to the CA rule 184 (in Wolfram's notation) with maximum jump rate as one site per time step. The difference is the updating

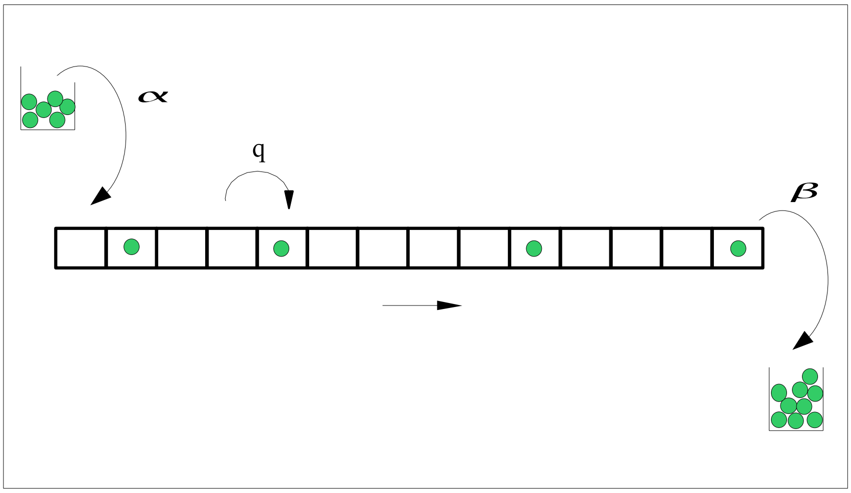


Figure 2.5: A schematic diagram of the TASEP model with open boundaries.

procedure: ASEP often uses random sequential updating. The TASEP model is well investigated in literature, and exact analytical results have been obtained for it. The phase diagram expressing the density profiles in the lattice when α and β are varied shows quite complex structure, and it is in principle similar for different updating schemes.

The phase diagram for TASEP can roughly be divided into three categories: the low-density phase (A), the high-density phase (B), and the phase with maximal current of particles in the lattice (C) [67, 11, 37]. The maximal-current phase C occurs when α and β exceed some q-dependent threshold value ($\alpha_c(p)$ and $\beta_c(p)$, respectively, $p = 1 - q$) with the current being independent of both α and β. In the low-density phase A (when $\alpha < \beta$ and $\alpha < \alpha_c(p)$), the current is low and dominated by α and it is independent of β. This phase can be divided into two subphases: for small β values the density at the right boundary gets higher values than elsewhere (builds up congestion) forming a phase AI, whereas for higher β values, the density profile shows slightly lower values at the right than in the other parts of the lattice, forming a phase AII. On the other hand, in the high-density phase (when $\beta < \alpha$ and $\beta < \beta_c(p)$), the behaviour of the system depends of the value of β and it is independent of α. This phase can also be divided into two categories with differences at the beginning of the lattice: for small α the density profile shows lower values than the rest of the lattice (phase BI), but for high values of α also the density is higher than for the rest of the lattice (phase BII). Explicit current and density values have been derived for different phases, which depict the situation also for the NaSch model with velocity maximum 1 (cell/time step) using parallel dynamics. The low-density and the high-density (sub)regions of the phase diagram can also be characterised by different types of domain walls: low-density–maximal-current, low-density–high-density, or maximal-current–high-density. In this case the lattice can be thought to represent a stretch of a road with known vehicle flows at the starting and ending points of it.

The phase transitions in the diagram can be connected to the correlation lengths ξ_α and ξ_β, which depend on the above-mentioned parameters α, β, and q. Transition from the low-density phase AII to maximal-current phase C is continuous with the correlation length ξ_α diverging. The change from the high-density phase BII to C is also continuous and the correlation ξ_β now diverges. The change from AI to AII is characterised by the ability of the right end of the lattice to remove the particles in the flow (depending on the value of β). Respectively, the change from BI to BII depends on the value of α to produce the density level at the left end with respect to the level in the rest of the lattice.

The change from the low-density phase AI to the high-density phase BI is of first order. The correlation lengths ξ_α and ξ_β remain finite, but a third length, $\xi^{-1} = |\xi_\alpha^{-1} - \xi_\beta^{-1}|$ diverges on the transition line.

The basic ASEP model corresponds to the noisy Burgers equation, which on the other hand, can be used as a fluid-dynamical description of vehicular traffic (the Lighthill-Whitham model with the Greenshields flow-density relationship).

The NaSch model CA using $v_{max} = 1$ can be related to a stochastic surface growth model when the surface is one-dimensional — the so-called single-step model [11]. The individual slopes upwards and downwards on the surface correspond respectively to unoccupied and occupied sites on the road lattice. The deposition of a new particle on the surface (in a local minimum site only) changes the slopes of two successive sites (down-slope → up-slope and vice versa), which is equivalent to a vehicle advancing one lattice site. The Kardar-Parisi-Zhang (KPZ) equation with noise is known to describe statistically such surface growth, and without noise the relaxation of an initially rough surface to a flat one [41].

The KPZ equation can be mapped onto the Burgers equation [11], which is a non-linear diffusion equation. On the other hand, with a simple approximation of the shape of the fundamental diagram ($q = v_{max}\rho(1-\rho)$, the Greenshields model), a fluid-dynamical description of vehicular traffic by the Burgers equation is reached. Thus a connection between CA models and fluid-dynamical theories exists.

Other models closely related to CA are **lattice-gas** models [59]. These models consist of systems of particles moving from site to site on a regular lattice; space, time, velocity, and density are discrete variables. The particles conserve momentum, which makes it possible to obtain hydrodynamic equations of motion for the particles.

The first fully discrete lattice-gas model of hydrodynamics was introduced in the 1970's. This model consists of identical particles moving on a square lattice, and the particle number and momentum are conserved in collision situations. The square lattice enables the model to be used only in limited applications, though.

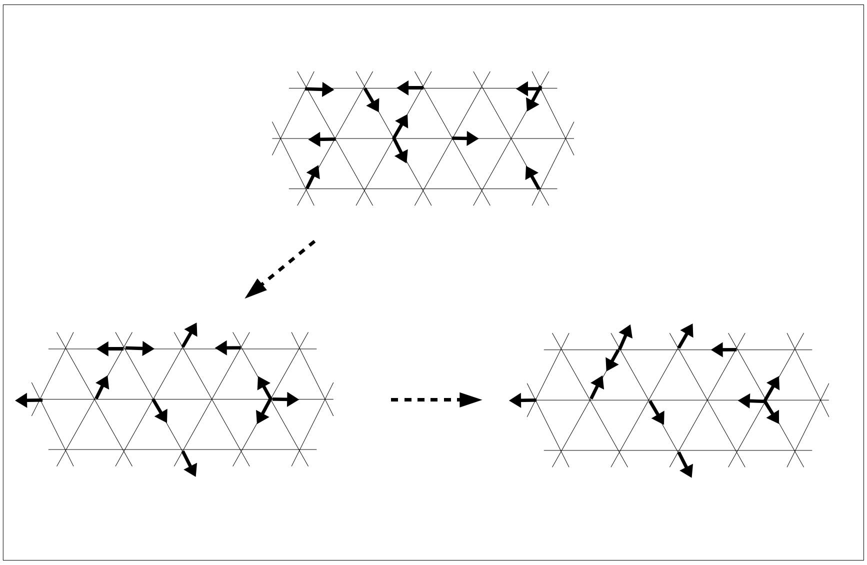


Figure 2.6: An example of the two stages of one discrete time step for a 2-dimensional FHP lattice-gas model.

Further development to the first discrete model was brought by the FHP model of Frisch, Hasslacher, and Pomeau in 1986. This model uses triangular lattice, which is sufficient for the needs of isotropic hydrodynamics.

Fig. 2.6 represents a lattice of the FHP model. An arrow in the lattice represents one particle moving with a unit speed in the given direction. The movements of the particles can occur along the lines connecting the lattice sites. Each lattice site can contain only one particle moving in the given direction. One discrete time step consists of two steps for the update of the lattice: first a particle moves to the next site in the direction of its movement, and secondly, if a collision occurs with another particle (other particles), a change in its direction may result. The collision rules are essential parameters of the model, and they may be applied via a probabilistic or a deterministic manner.

The FHP model has been intensively used for studying two-dimensional hydrodynamic flows especially in complex geometries. Also, the studied enviroment may involve particles of immiscible species or mixtures of gas and liquid. The model has later been developed to contain rest particles and three-dimensional cases.

The NaSch model is a minimal model, and it can reproduce the basic features of real freeway traffic (fluctuations in driving and jam formation) although the behaviour of vehicles has some nonphysical aspects (deceleration when approaching with a high velocity a slow or a standing vehicle). More diverse vehicle behaviour can be captured with small alterations to the basic model. By changing the acceleration for standing vehicles, the so-called *slow-to-start rules* are reached: the TT model (a generalisation of the model in [11]), the BJH model [4], and the VDR model [3]. The main empirically observed features of real traffic that have been captured with these modifications are metastable states and hysteresis effects visible in the fundamental diagram.

The slow-to-start models differ somewhat from each other, but they share the use of most of the NaSch model steps. Their acceleration steps differ from each other (and from the NaSch model): the braking percentage for standing vehicles is different (larger) than for the other vehicles. This usually causes the standing vehicles to start moving later than in the NaSch model. The slow-to-start rules describe the delay of vehicles in restart (after they have stopped in a jam because, for example, the driver does not notice the restart of the vehicle ahead). This causes the reduction of the outflow of vehicles from a jammed region and phase separation in flow.

The *cruise-control limit* of the NaSch model is a variation of the basic model and resembles the VDR model in the sense that the braking probability is velocity dependent. When vehicles move in this model with their maximum velocity v_{max} they just maintain their velocity. But the other, jammed vehicles move non-deterministically [48].

A cellular automata model suggested by Helbing and Schreckenberg, the *HS model*, is claimed to display also other properties of highway traffic flow than the models discussed so far. Among these properties are: the outflow from jams is independent of the initial conditions and it is, to a large extent, independent of the average surrounding traffic density. The outflow is also considerably smaller than the maximum flow [22].

The HS model has more parameters than the previous ones: different vehicle types, different velocity maxima for these vehicle types, the use of velocity dependent safe distances, and a velocity adaptation parameter for each vehicle type. — A significant difference with respect to the NaSch model is that, in the HS model rule steps, the new velocity for a vehicle is a weighted average of the velocity in the previous time step and the optimal velocity of the vehicle type. The vehicle-specific optimal velocity depends on the distance to the vehicle ahead. The adaptation parameter controls whether the new velocity is inclined to the velocity in the previous time step (great inertia to vehicle motion) or to the optimal velocity of the vehicle. The HS model is related to optimal velocity models, see Sec. 2.1.

In the past few years, CA models have been refined towards characteristics that take into account anticipation of events in traffic, and they also use finer spatial discretisation. The so-called *brake light model* [36] uses cell length of 1.5 metres, and it resembles the VDR model, but the braking parameter can have three different values depending on the vehicle's velocity and, if the vehicle ahead is within a horizon of interaction, the status of the braking lights of the vehicle in front (on/off). The brake lights and the vehicle's distance to the leading vehicle may also affect the acceleration and velocity adjustment of the vehicle.

A family of even more refined models, the *KKW models*, is introduced in [29]. The cell length in this case is again small, 0.5 metres, while the time step is 1 second as is normal with CA traffic models. The update rule consists of a deterministic and a stochastic part. In the deterministic part, the velocity value is first chosen between values 0, v_{max}, v_{safe}, and v_{des}. The term v_{safe} is connected to collision-free driving, and the term v_{des} is the desired velocity that is connected to acceleration (if the vehicle distances are large, normal acceleration is used, but if a vehicle is within a velocity dependent synchronisation distance to the vehicle ahead, it tries to adjust its velocity to that of the leading vehicle). In the stochastic part, a new velocity value for the vehicle is chosen from a few values, one of which contains stochasticity. Finally a normal movement step is included for the vehicle. The synchronisation distance is calculated is different ways in the separate versions of this model. Refining of the basic models leads to the requirement of using several new parameters in the rule steps.

Also, several other CA models using refined features have been introduced lately. Description of these can be found, for example, in [42, 5, 43, 24]. — In [35] different CA models of vehicular traffic are compared with respect to their ability to reproduce empirical features found in traffic: the fundamental diagram, large jams, outflow from jams, and generation of synchronised traffic regions.

The CA model mostly used in this study is the VDR model. The rule steps for the vehicle n of this model are:

1) $v_n(t+1) := \min(v_{max},\, v_n(t){+}1)$

2) $v_n(t+1) := \min(d_n(t)/\Delta t - 1,\, v_n(t+1))$

3) $v_n(t+1) := \max(0,\, v_n(t+1) - 1)$, with probability $p(v)$

4) $x_n(t+1) := x_n(t) \,+\, v_n(t+1) * \Delta t$

where $d_n(t)$ corresponds to the gap (the number of empty sites) in front of the vehicle n and the braking probability is calculated from

$$p(v) = \begin{cases} p_0, & \text{for } v = 0 \\ p, & \text{for } v > 0 \end{cases} .$$

2.2.3 Multi-lane, bidirectional, and city traffic

The simple cellular automata models developed for single-lane roads (with passing not allowed) give quite a limited picture of the reality of vehicular traffic. Usually there is a variety of vehicles on road having different maximum velocities which may be far less than the prevailing speed limit defined for the road. If this kind of situation was dealt with a CA model without passing allowed, it would end up with queues behind slower vehicles. Also, a large percentage of roads have two or more lanes in one direction and evidently a proper model for these cases would be useful.

Many models for multi-lane roads have indeed been developed from the basic NaSch model. The structure of these models is based on two things: forward movement on a single lane and rules governing the lane-change event. The update step is divided into two sub-steps [58]:

- the exchange of vehicles between two lanes is checked, and the vehicles wanting lane-change are moved sideways (but not advanced), this sub-step is performed in parallel for all vehicles
- independent one-lane updates for forward movement are made using normal one-lane update rules.

The sideways lane-changing is supposed to happen within one time step, which is usually one second in CA models. In reality the time is a bit longer. — If there are more than two lanes (in one direction), the conflict of vehicles trying to change from left and right to the same cell in the middle lane can be avoided by determining, for example, that all changes from left to right are allowed to be made during odd time steps and from right to left during even time steps.

There are several reasons for lane-changing but also different prerequisites that must be fulfilled before the change can be initiated [58, 50]. These can be expressed as 1) there must be incentive for the lane-change and 2) legal constraints must be fulfilled. Or 1) security issues have to be fulfilled, 2) legal constraints fulfilled, and 3) travel time minimisation is desired.

Legal constraints usually vary in different countries. In many European countries, the lane usage is governed mainly by two laws: the right lane has to be used by default and passing has to be done on the left. In the United States, on the other hand, the second of these laws is considerably relaxed — passing from right is not explicitly forbidden [50]. Travel time minimisation leads to a lane-change to the left when a slower vehicle in reached and there is more room on the left lane. — The lane-changing criteria can be either asymmetric or symmetric with respect to different lanes.

Security issues concerning lane-change contain the need for having enough space on the target lane relative to the position of the vehicle wanting the lane-change. The security criteria can be expressed in a CA rule by requiring that on the target lane the gap (the number of empty sites) in front of the vehicle's site (expressed in velocity units) is v (the velocity of the vehicle wanting the lane-change) and that the gap behind the site is v_{max}, which is the maximum velocity allowed in the CA in question.

The lane-changing incentive of a more 'European type' consists of changes from left to right and right to left and they are handled somewhat differently. A vehicle reaching a slower vehicle on the same lane has to change to the left lane for passing. Also, when passing is not allowed from right and there is a slower vehicle on the left lane, the vehicle on the right lane has to change lanes and get behind the slower vehicle. In both cases the incentive criterion for changing from right to left can be expressed as [50]

$$v_r \leq v \text{ .OR. } v_l \leq v,$$

where v_r (v_l) is the velocity of the vehicle in front, on the right (left) lane. This condition is taken within a certain lookahead distance parameter, and if there is no vehicle within this distance, the respective velocity (v_r or v_l) is set to ∞. Changing back to the right lane can be taken as a negation to the condition above:

$$v_r > v \text{ .AND. } v_l > v$$

signifying a constraint that the velocities of the vehicles ahead on both lanes be sufficiently large. This set for lane-changing constitutes an asymmetric rule. Several other rule sets can be found in [50].

Even if the necessary criteria for the lane-change are fulfilled, it may be advantageous to randomise the change event with a probability parameter to decrease the so-called ping-pong lane-changes [58].

In CA simulations of multi-lane traffic, it has been found that some generic effects can be pointed out [11]: the maximal performance of multi-lane systems is slightly increased in comparison to single-lane systems, slow vehicles lead to an alignment of velocities of different types of vehicles (with different maximum velocities) already at low densities, the 'lane inversion' phenomenon can be reproduced with CA (it has been observed on German highways that at low densities the right lane is more often used, but at densities close to the optimum flow, the left lane is more used than the right one).

The CA models for bidirectional traffic (two lanes, opposite directions) can be seen as an extension to multi-lane, or better, two-lane unidirectional traffic. The lane of the opposite direction can only be used occasionally for passing. The CA models for this purpose are again based on the NaSch model for one-lane traffic with additional rules set up to account for passing. The situation can be designed such that passing may be allowed on both lanes or only on the other one. Passing restrictions on either lane may also be applied to only certain parts of the road as is the situation on real roads.

Extra rules concerning bidirectional traffic may include [60]: 1) a vehicle must not decelerate while passing, if an oncoming vehicle is seen, the passing vehicle must return immediately to its own lane, 2) passing is not attempted unless the density in front of the vehicle to be passed is sufficiently low, 3) even if the conditions for passing are met, the changing of lanes is subject to probability, 4) a moving vehicle seeing oncoming traffic decelerates by one velocity unit to break the symmetry between the lanes (and prevent superjams). — The changing of lanes in the passing situation consists again of two parts: first the sideways movement and the forward movement after that.

According to simulations on a bidirectional model [60], there is little difference between the two-lane and one-lane models when the density on either or both lanes is large, but when the density on the passing lane is small, the flow on the home lane can be much higher than in a one-lane model. However, if the density on home lane is relatively small, the flow there may even be lower than in the one-lane model because oncoming vehicles hinder traffic on it.

At around the same time as the NaSch model was introduced, also another CA model, the model by Biham, Middleton, and Levine (= BML model) was published [6]. The BML model deals with city traffic although a direct analogy between the model and real traffic problems was not drawn. The model consists of a 2-dimensional square lattice, on which a simultaneous flow of objects can occur in two perpendicular directions. The original model consists of three variants, which differ by the use of stochasticity and occupation of cells.

A site in the square lattice of the BML model contains either an arrow pointing upwards (to north), an arrow pointing to the right (to east), or the site is empty. Lattice sites of the model can be considered to illustrate street crossings in cities and each time step to illustrate a period when the traffic lights remain green or red (in different directions). The right arrows are moved in even time steps to the right, and the up arrows are moved in odd time steps upwards; the allowed movement occasions depict traffic lights being green. An arrow cannot move to an occupied cell containing either a right arrow or an up arrow. The boundary condition is periodic in both directions. This constitutes the first variant of the model. Fig. 2.7 shows a typical dynamic configuration of the model (on the left) and a jammed situation (on the right).

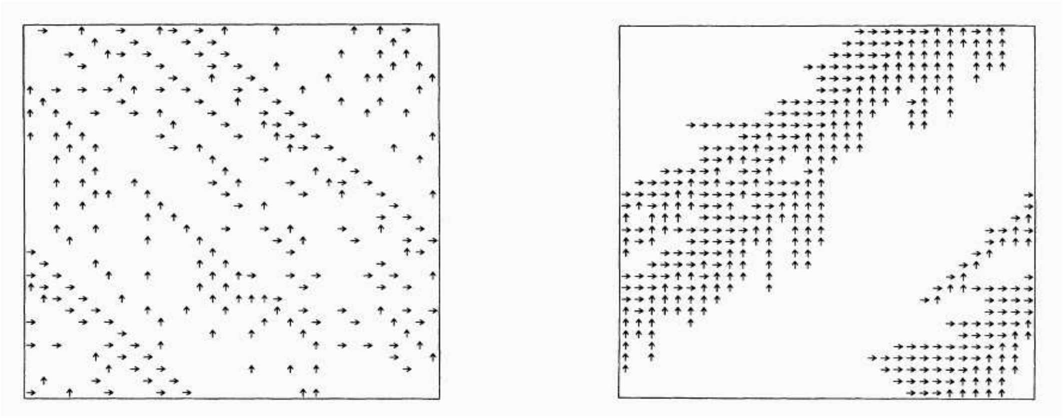


Figure 2.7: City model configuration example with a jammed situation on the right. Figures from Ref. [6].

The model described above has an update rule that is deterministic, and randomness is present only in the starting configuration of the arrows. In the simulations it has been noticed that — when starting from a random initial configuration — the system reaches an asymptotic state after a transient period. This asymptotic state can have two values which are separated by a sharp dynamical transition.

In the configuration below the transition (an example in Fig. 2.7 on the left), the system self-organises into separate rows of right and up arrows. These rows are along the north-west – south-east direction, all the right arrows can move on their 'own' time step and the same applies to the up arrows — the average velocity is then = 1. However, above the transition all the arrows get stuck and the average velocity is = 0 with very high probability.

A jamming transition separates the two states and it is induced by changing of the density, the occupation of the lattice sites. As the system size grows, the transition becomes sharper and the transition density decreases. Simulations demonstrate in this case that a first order phase transition takes place.

The original BML model has been varied over the years to describe the effect of faulty traffic lights, hindrances on streets, turning at crossings, etc. Lately, a unified model of the NaSch and the BML models has been introduced [12].

Whereas the vehicles hop from crossing to crossing in the city street in the BML model, there are separate sites between the crossings in the unified model and the vehicles move along these sites in the same manner as they move in the NaSch model. The traffic lights are synchronised so that they all remain green for the east-bound vehicles for a time interval T and at the same time they are all red for the north-bound vehicles. After time T they all turn to the opposite colour simultaneously. Again, no turning of the vehicles at crossings is allowed. When a light turns red at a crossing, the vehicles on the street stretch in front of it can keep on moving until they reach a stopped vehicle or the crossing.

It has been noticed in simulations [12] that in the unified model a phase transition from a free flowing dynamical phase to a completely jammed phase can take place in sufficiently large densities (the critical density depending on the length of the stretch between two crossings and the time period T). The mechanism for jam formation is different from the BML model, and it is necessary that the stochastic braking probability is not equal to zero.

Chapter 3

Methods

3.1 Problem setting

The objectives for this study are the following: on the one hand, to study the behaviour of CA simulation and to assess the utility of CA simulation in connection with traffic situations in urban road and freeway networks, and on the other hand, to study the laws and generic features of traffic, mainly congestion, with CA simulations.

The benefits of the CA models that can be utilised in urban vehicular traffic are already mentioned in earlier sections of this study: simple updating rules of the models together with a long time step length, which enable quick computer code using integer arithmetic operations, and yet quite natural average behaviour of the model vehicles. The speed of computer calculations becomes essential when vehicle quantities grow: in traffic studies the hours of interest are usually those of congestion, that is, times with a large number of vehicles. Moreover, simulation speed is also essential if real time simulation is required. — Whenever it becomes important to simulate realistic behaviour of single vehicles, sophisticated CA models are needed, but this again leads to longer calculation times.

CA models usually have only a few parameters, which often means easy calibration. Vehicle dynamics may be governed by one or two parameters for acceleration/deceleration. Additional parameters for model calibration may be connected to items like turning direction percentages at intersections or off-ramps, vehicle amounts entering the model area, and signalisation.

Other microscopic simulation models (of the car-following type) provide a detailed description of vehicle movement, which necessitates thorough simulation calculations and thereby leads to extra time consumption in the process. When the vehicle amounts get larger, real time simulation, let alone predictions of traffic into the future, may become impossible. Also, calibration may become a problem with a large amount of parameters to consider. An example of this kind of simulators is the HUTSIM program (see Sec. 3.4), which provides a detailed description of vehicle or pedestrian movement. The HUTSIM simulator is used in this study as a source of reference traffic.

The level of sophistication of the simulation model more or less determines the environment where it is most favorable to be used in. The applicability of the car-following type simulators, like HUTSIM, is good for situations when time is not so scarce, for example, traffic planning. CA simulation, on the contrary, is usually fast but lacks particular features of single vehicle movement, and thus it is better suited for the purposes of monitoring and controlling of traffic.

In practice traffic issues are typically measured with induction loops and infrared detectors. The data received from these devices are local and do not give a picture of the traffic in the areas between the detector spots. Microscopic modelling of traffic, for example CA simulation, can be used to process that raw traffic data from a telematic system and treat them into estimates of traffic situations. The traffic situation information can for its part be used as input for traffic management and information purposes.

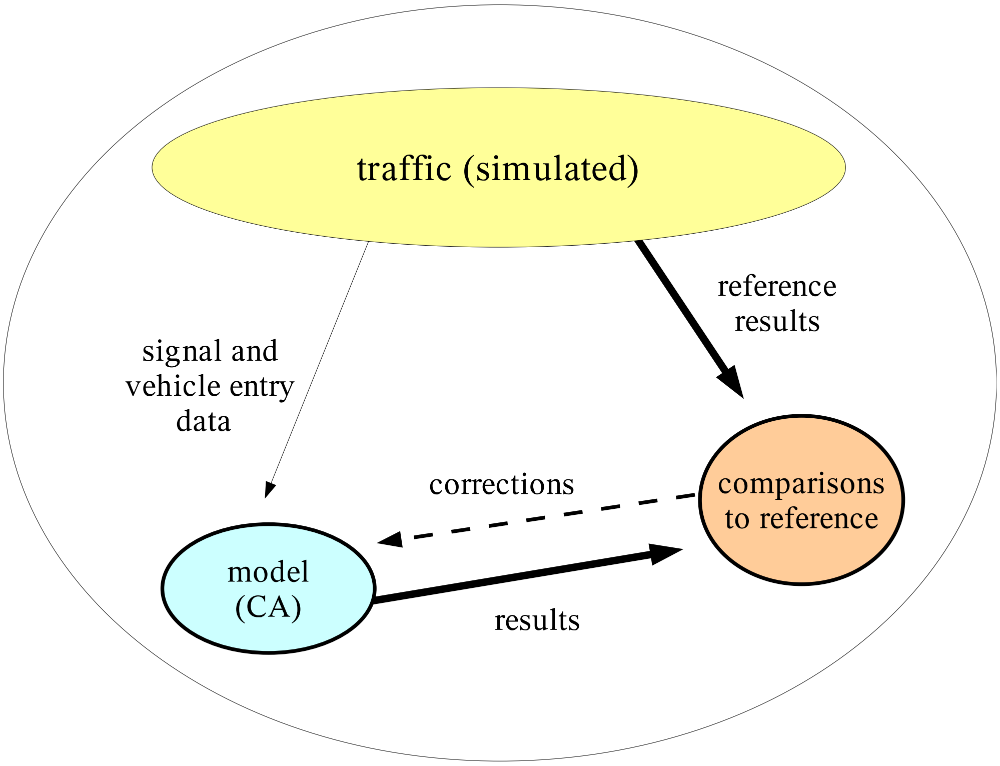


Figure 3.1: The research philosophy used for intersection and freeway studies.

The parameters describing traffic situations that can be produced by CA simulation include: queue lengths at intersections, average vehicle velocities, detector passing, delays, and the number of standstills of a vehicle during its journey. The validation of these simulation issues with respect to real traffic is difficult since repeated experiments with real traffic are usually expensive and very difficult to arrange. For that reason the CA simulation results are compared to the results of HUTSIM simulation, which is in this case considered to be 'real traffic'. HUTSIM implements very natural traffic activities for vehicles and pedestrians and enables the use of practically any kind of test environments. From these comparisons an absolutely direct connection to the link CA simulation – real traffic cannot be drawn. Additional reasons for the choice of the use of HUTSIM are that real measurement data were not available at the time the work was started and that HUTSIM is a validated simulation system reproducing the vehicle movements quite naturally (so the comparison to real traffic is not far-fetched). Also, one goal of the work is to study the systematics of measuring and modelling irrespective of how natural the reference data are.

Fig. 3.1 depicts the research philosophy that is used in this study in connection with intersection and freeway simulations. The traffic is generated in the HUTSIM program, and the reference results (queue lengths, average velocities, and detector passing) are constructed in it, too. Limited amount of traffic data (vehicle arrivals at model area boundaries and signal state changes) are fed to a CA simulation program, after which traffic is generated inside this simulator obeying the rules of the chosen CA model. The CA program then creates its own results describing the traffic situations. After the CA run is complete, comparisons are made between the results of both simulators. The CA model may then be corrected to get the CA results to better match the 'real' ones. These corrections may concern items like CA rule braking parameters and turning direction percentages.

The CA simulation behaviour in connection with traffic situations is studied in urban networks with two environments (two- and five-intersection areas) and in a freeway area involving several on- and off-ramps. Moreover, CA simulation is used in this study to find out whether any laws could be found in queue formation as a consequence of some obstacle like red traffic signal.

3.2 Basic methodology

Nowadays there are numerous, different CA models available for traffic simulation as discussed in the earlier chapters. At the time the work was started, the variety of CA models was not as rich as it is today, especially regarding models that use anticipation in calculating vehicle movements. Since it was expected for this study that a large number of vehicles is involved in simulations and that detailed vehicle simulation is unnecessary (especially in problems associated with generic features of traffic), a simple CA model was preferred. Although the NaSch model is considered to be the simplest of these models, the presence of numerous intersections in urban areas suggested the importance of realistic simulation in dissolving queues. For this purpose some of the slow-to-start models might be more suitable than the basic model. After some tests with simple CA models, the VDR model was chosen to be primarily used in this study. The VDR model has only two primary variable parameters making the calibration of the model simple. This choice was also supported by its use in many other studies found in the recent literature especially in situations when detailed vehicle behaviour is not required. Comparisons between the VDR and the other slow-to-start rules were not made.

Fig. 3.2 depicts the methods mainly used in this study in urban and freeway area networks for assessing CA simulator behaviour. The traffic network area models for both simulators are prepared as alike as possible; if both models are constructed from scratch, the congruence between them can be very good. If, as was the case in two of the models used, the HUTSIM models were already prepared elsewhere, the CA models to be constructed may have to compromise in some smaller details.

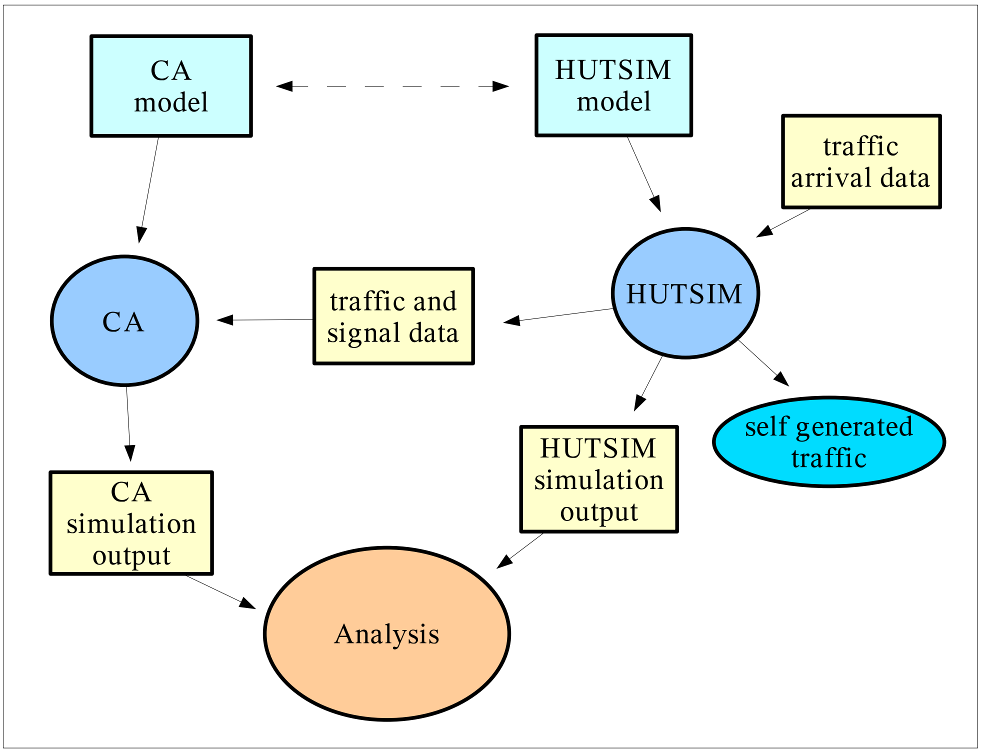


Figure 3.2: The normal program and data components used for studying CA simulator behaviour in comparison to HUTSIM traffic.

The CA simulation model is principally devised such that it gets the traffic data — vehicle arrivals to the model and signalisation — from outside, as data a file. This file is usually an output file from a HUTSIM run, and it contains the data (including time stamps) of vehicles passing over detectors located at the arrival spots on the borders of the simulation area. The signalisation data contain all the signal status changes with timing data. The traffic data conveyed via file correspond to information that can be gathered from detectors working in real time systems. Therefore the CA simulation arrangement might be connected to an on-line simulation environment in the future.

Both simulators produce files describing vehicle behaviour during the simulation run: information concerning vehicle origin (arrival) and destination (exit) spots, vehicle arrival and exit times, detector passing times, etc. From this data comparisons between vehicle behaviour in these simulators can be drawn. Also, both simulators provide visual animation of vehicle movement and signal status changes, which may help in the analysis.

The traffic data (signalisation and especially the arrival of vehicles) can be generated and used in a couple of ways. The HUTSIM program can produce arriving vehicles by itself given a desired flow level at the arrival spots (origins); the flow level can also be made to change with time simulating, for example, the rush hours. Another way is to use arrival files that already contain the description of arriving vehicles (type, desired velocity level, destination, and arrival time). An arrival file may be a result of field measurements with induction loop detectors or an output file from an earlier HUTSIM run.

In the two urban intersection cases of this study, the CA simulations are produced using the traffic and signal data created by HUTSIM, and the run results are compared to the HUTSIM run results. In the freeway case, the HUTSIM simulation runs are not needed since the simulation results for them are already available. The CA simulations are performed by using the arrival file directly as input to the CA simulator.

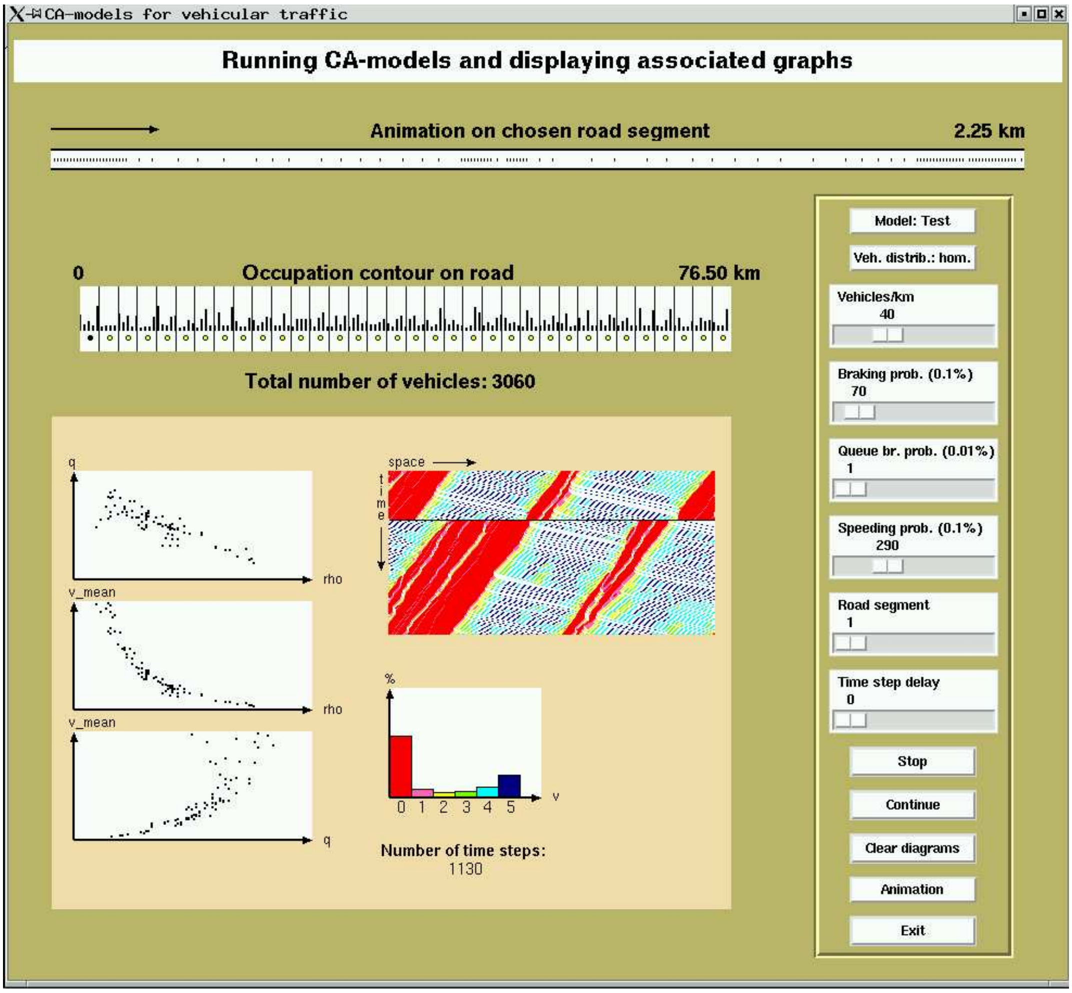
As regards simulations of generic traffic features, only CA simulations are used without any reference to HUTSIM traffic. The queue formation is studied in a stretch of road with a traffic signal head at the end of it. The traffic is generated with a separate program producing output of the arrival file type.

3.3 Overview of the CA simulator program

The traffic simulation program designed for this work is partly based on the testing simulator developed in [23]. In the testing program, the movement of a selected amount of vehicles is simulated on a simple one-lane stretch of road by different CA model rules. The behaviour of vehicles is visualised by animation and some graphs.

Fig. 3.3 shows one of the graphic interfaces used in [23] for studying different CA model rules. The display contains, for example, animation and a space-time plot of vehicle occupation for the chosen road stretch. Also, graphs between dependencies among flow, density, and average velocities are shown.

From this starting situation the program has been developed profoundly during this study. As the road network has become more complicated, calculations and decisions associated with all vehicles are more elaborate than in the testing simulator. Especially the need to use intersections, parallel lanes, traffic signals, and traffic input from outside caused deep changes of the program structures. The visual side of the present version of the simulator interface is now of a more assisting type since



CA-models for vehicular traffic

Running CA-models and displaying associated graphs

Total number of vehicles: 3060

Number of time steps: 1130

Model: Test

Veh. distrib.: hom.

Vehicles/km 40

Braking prob. (0.1%) 70

Queue br. prob. (0.01%) 1

Speeding prob. (0.1%) 290

Road segment 1

Time step delay 0

Figure 3.3: Interface of a testing simulator.

the characteristics of the simulated traffic are mainly inspected from the files output by the program. However, the basic division to the calculation part, realised as a C-coded program, and to the interface part, implemented by the scripting language Tcl/Tk, has remained.

The CA model rule that has mostly been used in this study is VDR, see Sec. 2.2. Vehicles follow the basic VDR rule in normal movement on street lanes, but modifications to this are made in several situations described in the coming sections (traffic signals, lane-changing, speed limits, yielding, etc.). Rough flow chart of the CA simulator can be seen in Fig. C.1 in Appendix C.

3.3.1 The simulator program

The programming platform for the work was chosen to be a normal workstation with the Linux operating system used in the laboratory. Although cellular automata models run usually quite fast, the largest traffic simulation solutions require parallel computing. However, from various literary sources and experience from [23], it was to be expected that the capacity of the workstation most probably would be sufficient for the present study. Besides, the speed of the model simulations was not a matter of primary interest even at the beginning.

The choice of the computing language was a rather delicate problem, which is more intensively considered in [23]. Because visualisation and animation of traffic in the model area was essential, conventional programming languages as such were not adequate. Since the Tcl/Tk scripting language was readily available for use on the computer and had from earlier experience proven to be quite flexible in this kind of use, it seemed a good choice.

One possibility might have been to use purely Tcl/Tk for programming. Yet it seemed in the early stages of the work that vehicle numbers may grow quite large, which would require much calculating leading to poor performance of the code. The solution was to use some conventional language together with the scripting code and split the load between these two: most of the time consuming calculations are made by a conventional language code.

There were practically two choices for the conventional language: Fortran and C. Although Fortran is usually considered as a code apt to mathematical calculations, requirements in this respect were not expected to be demanding. Since, in addition, the problem seemed to involve a lot of string manipulation, the C language was felt to be a better choice [34]. — As a programming style, the object oriented alternative (C++) probably would have been the best solution, but since I was not well acquainted with that technique, the conventional way still seemed adequate for this study.

Thus, the simulation program consists of two parts: a C-coded program for most of the calculations and a Tcl/Tk-coded part for the graphic interface, vehicle and traffic signal animations, and displaying counters. In Fig. 3.4 is depicted how the workload is divided between these two parts. The C-part of the program is discussed on the general level below.

A definition part and a small main program are located at the beginning of the C-program. All the other activities of the C-program are contained in separate functions, which are called either from the C-part or the Tcl/Tk-part.

The definitions in the C-program include the space allocations for the road elements — cells — and the data associated with the possible vehicle in each cell (velocity, temporary destination code, number of stops during the route, origin node code, and the starting time step). Also, the arrays for parallel lanes, traffic signal red states, routing information, speed limit pipes, and yield areas of pipes are allocated here, and so are the arrays for storing output data.

The main program is called after the definition part, and it first contains the reading of some run parameters from the console. Next, the code includes function calls by which the rest of the input parameters are read from various files. These files are called 'simulation-specific files' in this text (listed in *cafiles.dat* in Appendix A followed by the descriptions of these files). — A graphical presentation of different files used by the CA simulator program can be seen in Fig. C.2 in Appendix C.

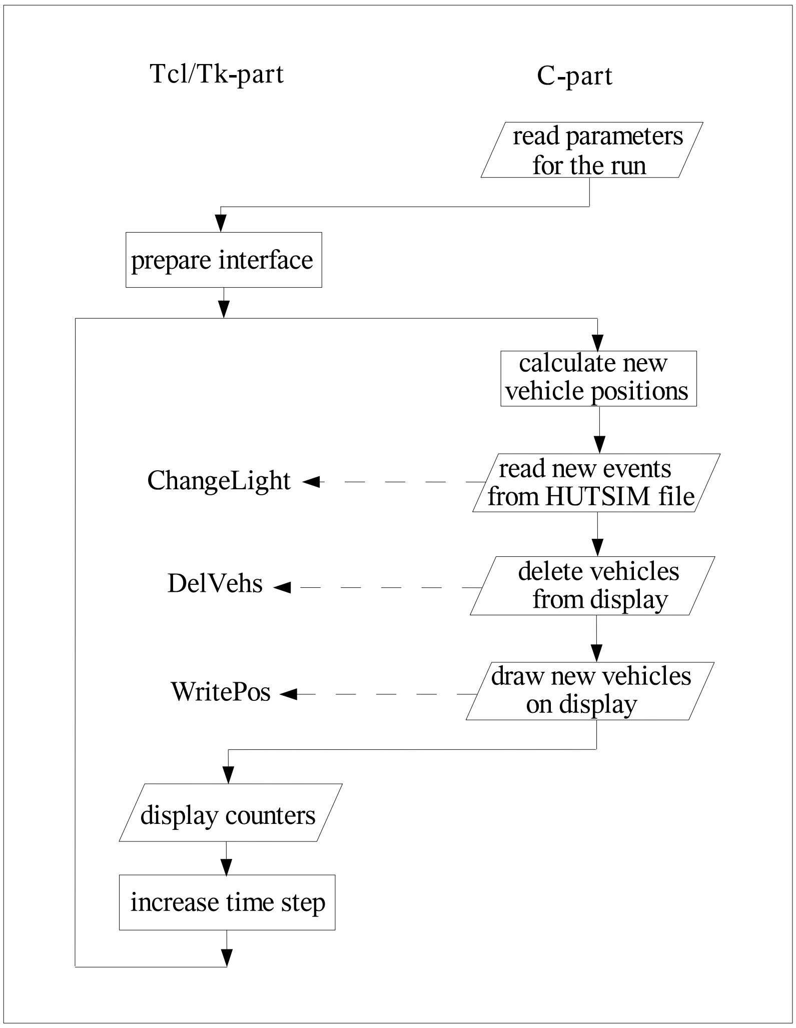

Tcl/Tk-part
C-part
read parameters for the run
prepare interface
calculate new vehicle positions
ChangeLight
read new events from HUTSIM file
DelVehs
delete vehicles from display
WritePos
draw new vehicles on display
display counters
increase time step

Figure 3.4: Workload between the C-program and the Tcl/Tk-script in the CA simulator.

At the end of the main program, the Tcl/Tk-initialisation function is called. Also, the links between the few mutual parameters in the two program parts are defined here: the used rule, values for the braking parameters, some control parameters, and counters. After these definitions the control is transferred to the Tcl/Tk-part.

The two important functions of the C-part are *getroadCmd* and *iterpipes*, through which most of the operations of the program are handled. They are both called during every time step. The former one calls numerous other functions in the C-part as new events occur (vehicle arrivals, signal state changes, and animation activation, see Fig. C.1). From this function also some procedures in the Tcl/Tk-part are called (see Fig. 3.4). The second function handles lane-change calls, vehicle movement during one time step, and detector update calls.

The internal data storage in the C-part is implemented by conventional arrays of various dimensions. Fig. C.3 in Appendix C shows most of the internal arrays used and also the input files from which the data are passed into these files at start-up time. The structure and items of separate internal arrays are not further discussed in this context, however, some of the array names are self-explanatory and the associated file names give hints of their contents (see Fig. C.3).

Most of the functions in the C-part are very small in size and simple in content, so they most probably can be used in case the program is developed further or if it is joined into some other software. — The list of the functions for the C-part can be found in Tables C.1–C.2 in Appendix C.

3.3.2 The user interface

The user interface is defined in the Tcl/Tk-part of the program. Tcl/Tk (for example [52, 71]) is a scripting language designed for easy use of graphical entities employing widgets like buttons, menus, scales, and scrollbars readily available for the user. On canvas areas one can define movable lines, arcs, ovals, polygons, rectangles, text, and image items. — The creation and moving of items on display requires some computer time; whenever quick animation on the screen is needed, performance problems may arise. However, in this study such situations were not encountered although even a large number of moving objects was present on the screen.

Examples of the interfaces used in this study can be seen in Figs. 3.17, 3.19, and 3.22 in Sec. 3.7. The actual animation area in these figures — lanes and signals of the intersections — is enclosed in a canvas rectangle, in which the objects are formed using lines, spheres, rectangles and text elements. — Other objects of the screen contain scales, pushbuttons, and menus (templates of which are supplied by the Tcl/Tk), which are drawn in rectangles of different shapes and sizes.

No special editor has been developed in this work for forming the interface display, so building up of the interface area is a bit tedious task requiring detailed definitions of road elements, signal heads, and vehicle routes in numerical form. With simple intersections this is not a problem, but requires a lot of work when the area is complicated. Since in this study the CA models to be constructed were to resemble the corresponding HUTSIM models, help for the construction could also be received from the HUTSIM side.

The interface building process is managed by forming detailed numerical locations of roads, lane lines, signal heads, and pipes in the network area on the screen. These locations are prepared beforehand with the help of a spreadsheet program, which offers some automation to the process. The information connected to these items is divided into four spreadsheet output files (road areas, lane lines, signal heads, and pipe positions) called 'interface-specific files' (see Fig. C.2). These files are read by the Tcl/Tk-part at the interface building time. The vehicle routes on the display are formed from pipe position data by calculating the position of every possible cell location in advance.

Every item (vehicle, coloured sphere of a signal head) on the canvas area that needs to be modified or moved during run time has to be attached with a tag so that it can be referred to from the Tcl/Tk-program. The movement of a vehicle on the screen during every time step is carried out by deleting the vehicle object from its old position with the help of its tag and creating it in a new position (instead of moving the object on the screen). The signal state changes are carried out by sending the new state code to the Tcl/Tk-part after it has been read from file.

The time step loop is run in the Tcl/Tk-part of the program, and a call to the C-part for incidents like new vehicle arrivals to the model, changes in traffic signal states, and calculation of changed vehicle positions is situated in this loop.

As a code the Tcl/Tk-part consists of definitions, procedures, and the program body. Procedures contain small tasks related to graphical operations on the display (like drawing of squares and lines), and they are called either from the Tcl/Tk-part itself or from the C-program. The body contains setting up the interface display. The main loop running the simulation iterations is also situated in this part of the program. — The list of the procedures in the Tcl/Tk-part can be found in Table C.3 in Appendix C.

In Secs. 3.3.3–3.3.6 some key aspects of the program are presented on a general level.

3.3.3 The road network

The road network is modelled as a collection of straight lane stretches of variable length called pipes. Since each pipe consists of cells, the pipe length in real space is limited to some multiple of the cell length. Fig. 3.5 shows an example of a road network (individual cells of a pipe are not marked in the figure).

In real space a pipe corresponds to one lane or some part of it. A vehicle is constrained to move to a certain direction in the pipe, and when it reaches the last cells of the pipe, it jumps to one of the first cells of the next pipe depending on its velocity. If there are more than one pipe following, the vehicle chooses the correct pipe according to its destination code and the routing information of the follower pipes.

Some of the pipes in the network may be defined as allowing lane-changing. In these pipes a vehicle may jump to an adjacent cell in the parallel pipe and continue its moving in it. Parallel pipes must be of equal length. The present program version does not allow more than two pipes belonging to the same parallel pipe set.

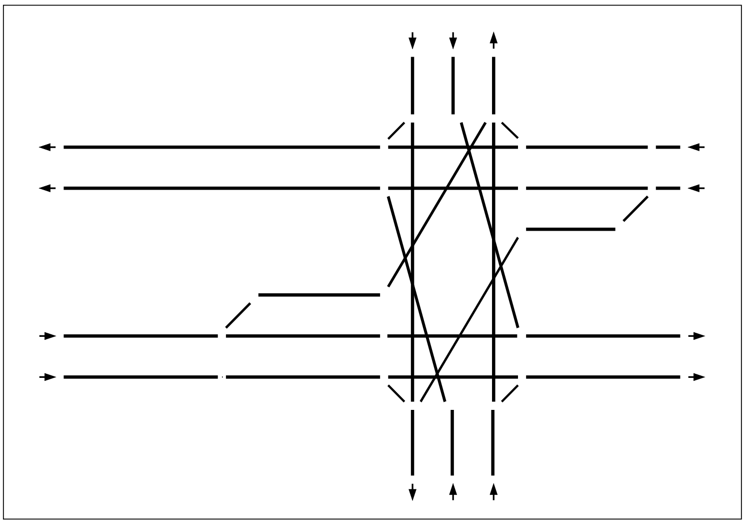


Figure 3.5: Pipe construction example.

The pipe length is limited by the array definition length: at the moment the maximum is 200 cells. The pipe length must be no shorter than the maximum velocity used by the model vehicles in the pipe.

3.3.4 Vehicle behaviour

In the CA simulator program, the data concerning vehicles is stored in the information related to pipes, which is contained in a three-dimensional array (*pipes*). The first dimension of this array corresponds to a specific pipe, second dimension to the cells of a pipe, and the third one to the characteristics of a cell. The most important characteristic of a cell reveals whether the cell contains a vehicle or not, and the other ones list the information concerning the vehicle. The vehicle information is composed of the items in Table 3.1.

–	velocity
–	(temporary) destination
–	flag for stops during the vehicle's travel
–	origin code
–	starting time step

Table 3.1: The stored vehicle information.

During every time step of a CA simulator run, each pipe of the network is treated in ascending numerical order. Within a pipe vehicles are searched starting from the first cell of the pipe up to the last one, and necessary actions are performed to each of the found vehicles. In normal movement the new velocity and the new position of the vehicle in the pipe (or in some of the possible follower pipes) is calculated with the chosen CA rule. The maximum reachable velocity is applied for the whole network, but the velocity can also be defined as being less than this by a pipe-specific value.

When reaching the last cells of a pipe, the program tests whether the next leap leads beyond the last cell of the present pipe. If this is the case, the first unoccupied cell of the follower pipe is chosen depending on how far the vehicle's velocity carries. In case the first cells of the follower pipe are not free, the vehicle remains in the last cell of the present pipe. If there is no follower pipe, the vehicle exits the model area. — The present program version does not allow vehicles to jump 'over' any pipes on its route.

When the deflection angle between two consecutive pipes is large, these pipes can be defined as a deflecting pair. In this case the vehicle always jumps to the first cell of the follower pipe in spite of its velocity. By this feature the velocity decrease in the turning is simulated.

Besides rule related issues, reasons that may cause a vehicle to stop on its journey are traffic signals and yielding. Traffic signals are always located at the end of a pipe, and in the case of red signal value, the vehicle remains in the last cell of the pipe until the signal turns to green. — When yielding, the cell where a vehicle stays put in can also be some other cell than the last one of the pipe.

Vehicles moving in a pipe that has a parallel, lane-changing pair, may change lanes. Within the CA simulator, the whole pipe or just part of it may be defined as a lane-changing area (see file *pipelen* in Appendix A). The lane-changing occurs in two phases within one time step: firstly, all the allowed lane-changes are made to the adjacent cell in the parallel pipe, and secondly, all the vehicles that have changed lanes advance from their new cells according to normal practice.

Lane-changing is necessary for routing purposes if a vehicle cannot reach its destination without taking the parallel lane. The rules used in lane-changing for routing purposes are defined in Table 3.2. The primary rule (the first row of item 4 in the table) implements an often used security criterion (see Sec. 2.2.3 and [50]). The second part (the last row of item 4) is a relaxation to the primary rule for standing vehicles.

It is also possible to change lanes because the adjacent lane is less occupied. This feature is managed by first selecting the candidates for lane-changing on one time step, and then the lane-changing itself is performed on the following time step. Conditions for selecting candidates include: both lanes must have direct access to the vehicle's destination, gap to the next vehicle must be smaller on the target lane than on the present one, and the preceding vehicle on the target lane must be moving. In the actual lane-changing event, the necessary conditions are items 1, 3, and the first part (row) of item 4 in Table 3.2, and the rule uses the same basic idea

1. vehicle on a lane-changing area
2. vehicle's destination accessible from target pipe with a smaller access code than from the current pipe (see Sec. 3.3.5)
3. vehicle's adjacent cell is empty
4. minusgap $\geq$ VMAX .AND. v $\leq$ plusgap
   .OR.
   v $<$ 1 .AND. minusgap $\geq$ VMAX–2 .AND. prob. cond.

(minusgap =	empty cells to the next following vehicle in the target pipe
plusgap =	empty cells to the next preceding vehicle in the target pipe
VMAX =	maximum velocity in the model
v =	vehicle's velocity
prob. cond. =	parameter controlled probability condition for lane-change)

Table 3.2: Lane-changing conditions for routing purposes.

of security as the lane-changing for routing purposes (yet being stricter since only the first part of item 4 is used).

Sometimes a dead-lock pair appears at the end of a lane-changing area. This situation is cleared by changing the two involved vehicles with one another, that is, by changing the characteristics of the pipe cells in question with one another.

For research purposes just the animation of vehicles is clearly not enough, but the movements of the vehicles have to be recorded for later examination. This is accomplished within the CA simulation program by recording to output files the times the vehicles pass detectors on chosen spots in the road network, the instants they leave the model area at the exit spots, or by measuring queue lengths at signal stop lines (at the time the signal turns from red to green). In connection with detector passing or exit times, also other items like arrival time and origin/destination codes are usually recorded.

Fig. 3.6 shows the internal entities in the CA program from which the data are transferred into output files. Some of the associated data are written to output files at the time the run finishes, some of the data are written to the files continuously during the run.

The queue lengths at signal posts are stored in 'quelengths', the contents of which are written to the end of the log at the finishing time of the run. The entity 'pipechains' is used in the queue length calculation, and it contains chains of codes for all those pipes that take part in the length calculation for a specific signal head.

If desired, vehicle exit data are written continuously during the run when a vehicle

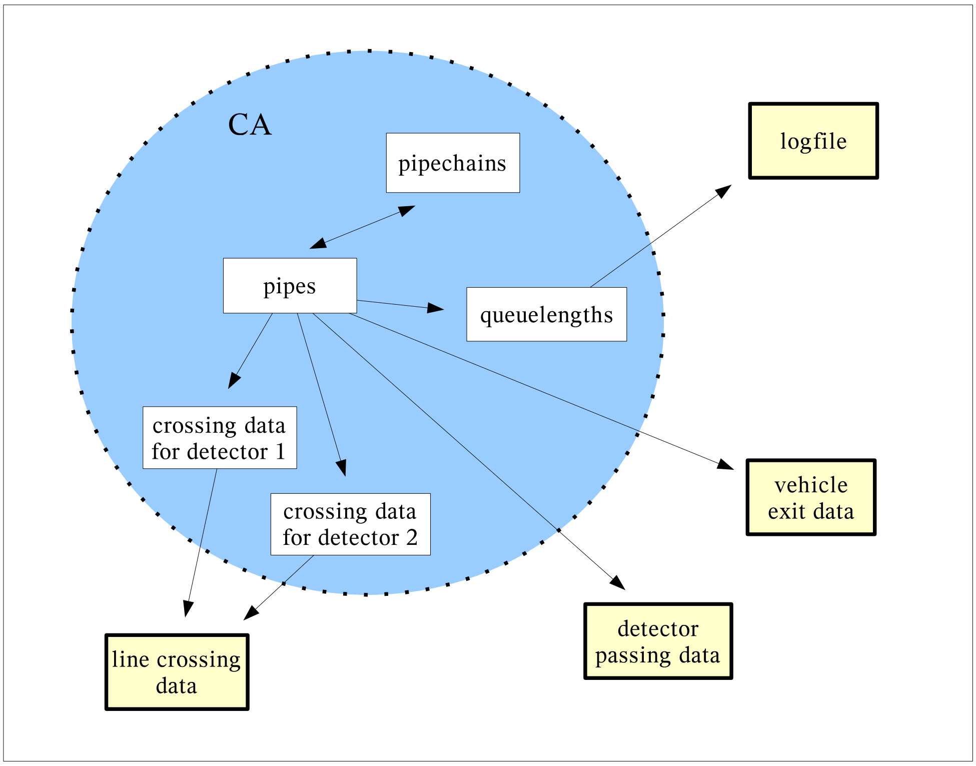


Figure 3.6: Storage and output of vehicle detection data.

moves out of the model area (output file *vehleave*). When necessary, detector passing data are also written continuously (output file *cadetsig*), whereas line crossing data are first written to internal tables, from which they are transferred to an output file ($detfile$). The descriptions of the output files can be found in the part 'Map for output files from CA', Appendix A.

3.3.5 Routing

Routing is needed to guide the vehicles of the model to one of the available destinations (exit spots) of the area. In real traffic the routing between the origin and the destination for a vehicle (origin-destination map) is seldom known in advance except for buses. So, when simulating traffic in some network, the vehicles have to be in some way given the route selection information that resembles the real events in the area. This can be accomplished by measuring the average turning percentages in all the intersections of the area in advance (taking into account the circumstances like time of the day/year, weather, incidents involving traffic) and using them for the vehicles in the simulator. If on-line simulation is used, the vehicle amounts can be monitored at intersections and the information conveyed to the model continuously: the turning rates are formed and changed according to real events in the modelled neighbourhood. In both cases several detectors per intersection are required.

Since in this study the traffic being simulated is HUTSIM traffic and the CA simulation is done after the HUTSIM run, the routes of the vehicles are already known before the CA simulation takes place. Hence, the CA routing is arranged by calculating the turning percentages with the help of HUTSIM generated routing or using the routing information of the arrival file.

When the original traffic is generated by HUTSIM's own mechanism, the HUTSIM vehicles get their route at generation time according to statistical basis from percentages given for each origin-destination pair from that origin. For CA vehicles these route percentages are changed into turning percentages (turn rates) at every intersection they can reach. At the origin spots these turning percentages for the next possible intersection are the same as they are for HUTSIM vehicles at the same spots. For a CA vehicle leaving an intersection, a new turning percentage concerning the next possible intersection is calculated from the HUTSIM routing data and the average flow related to the pipe in question (the average flow of vehicles entering the pipe and turning into certain direction in the next intersection divided by the total average flow to the pipe).

If arrival files are used, the HUTSIM route percentages can be calculated from the arrival file contents. Otherwise the CA calculation process is similar to the one described above.

The consequences of the differing routing distributions between the two simulators were not separately analysed. This could have been done by deliberately deviating the turning percentages from the calculated ones for the CA simulator. The runs with the extra detectors for the five-intersection case (Sec. 4.4.1) give some hints on the subject.

When a CA vehicle enters the network area, the final destination is usually not known. The vehicle is given a temporary destination code extending only beyond the next intersection. This means that the links between intersections need their own codes as temporary destinations. The temporary destination is given to the vehicle by drawing lots among the possible alternatives according to the turning percentages for the link. The same procedure applies to vehicles entering a new link after an intersection.

Besides the (temporary) destination, a vehicle moving in a network needs to know which pipes it can use on its route. This routing information is attached to each pipe, and it indicates which destinations of the model network are reachable from the pipe in question. The possibility to reach a certain destination is controlled by an access code. The codes are uncovered in Table 3.3. — The routing information of the network is arranged as an array *desti*, expressing the access code value attached to each possible destination for every pipe.

If a link is built of more than one pipe, the routing information must allow a vehicle to move through all of them. In case a pipe is followed by several pipes, the one with the lowest access code (>0) is chosen. If there are several follower pipes having direct access for the destination (code=1), the first one of them in the follower pipe list is chosen unless some of the others is less occupied.

code	explanation
0	destination is not accessible by this pipe
1	destination directly accessible by this pipe
2	destination accessible by one lane-change
9	destination accessible by two lane-changes

Table 3.3: Access codes used by the CA routing mechanism.

The routing code convention and usage method principal is similar to that in HUTSIM with some variation.

3.3.6 Traffic input and signalisation

The present version of the CA simulator program allows traffic and signalisation input only from file. The normal procedure is that a simulation period is run on HUTSIM and the generated output file ($detsigfile$) is used as input for a CA run.

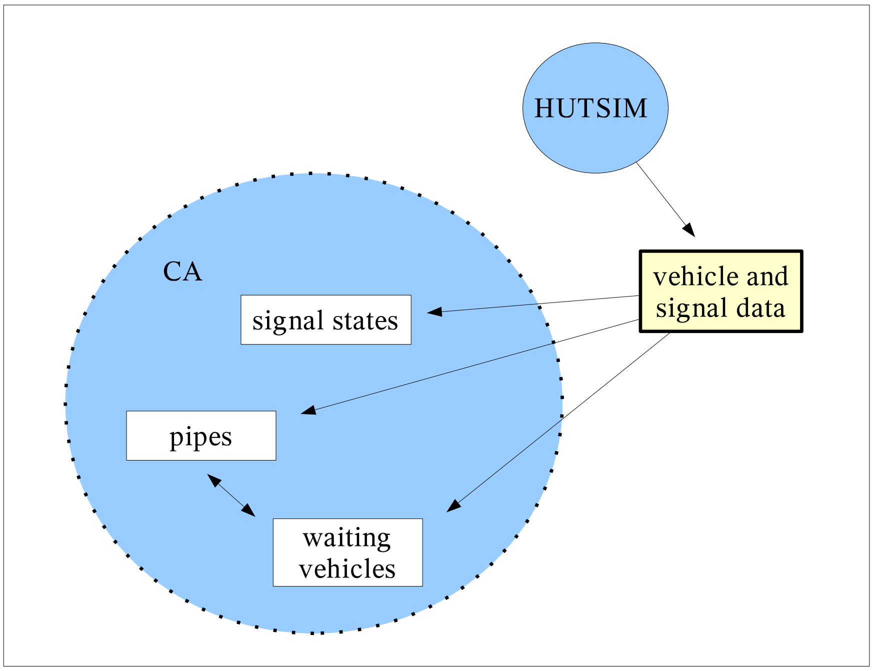


Figure 3.7: Vehicle production and signalisation management in CA from a HUTSIM file.

Fig. 3.7 describes the treatment of vehicle and signal data. When a vehicle entry is read from the file, the vehicle is placed at the first cell of the associated pipe at the origin node unless that cell is already occupied. If the cell is occupied, the new vehicle is placed in the entity 'waiting vehicles' waiting to be placed in the correct pipe and cell as soon as there is room for it. Signal data read from the input file are stored in the CA entity 'signal states' declaring for every signal head whether the signal light is red or not.

Accordingly, the input file contains two types of records. The first type stands for a vehicle at some of the detectors (usually at an origin spot of the model). The detect time and the detector code are conveyed. The second type corresponds to traffic signal records: the state change time, area and group codes of the post, and the new signal code are declared. The time unit in the input file is 0.01 seconds. An example and record description of the input file ($detsigfile$) can be seen in 'Map for an input file from HUTSIM', Appendix A.

A special version of the CA simulator (see 'Freeway area simulations', Sec. 4.5) reads the arrival file directly (normally used as an input file to HUTSIM). Since this file contains only vehicle arrivals and not any signalisation data, this method can be used only for unsignalised networks.

Yet another way of producing traffic and signalisation data is to use a separate program to generate them. This method is used with the general feature simulations of this study (see Sec. 4.2) since any HUTSIM runs are not involved in this case. A program called EVEGE was prepared to generate vehicle appearances at one origin of the model area at a predefined flow rate. The program also generates signal state changes to green and red with a chosen fixed period. The program EVEGE is described in Sec. 3.5.4.

3.3.7 Input and output files

The CA simulator uses both input and output files during a simulation run. The numerous input files contain descriptions about the model configuration, interface drawing, and the vehicle and signalisation input. The files used by the CA simulator are described in Appendix A.

The simulation run files define the model environment: general parameters, filenames, pipes, origin-destination information, traffic signals, detectors, etc. A simulator run also uses a few interface-specific files, which are not closely described here (road area, lane line, signal head, and pipe positions; see Fig. C.2 in Appendix C). These files contain numerical data for positioning graphic items on the interface display.

All the above-mentioned files are read by the CA simulator at the beginning of a simulation run. The program uses one more input file that is read during the run containing information about vehicles and signalisation in the form of a $detsigfile$ or an arrival file (see Sec. 3.3.6).

Output from the CA simulator includes four types of files: a log, a vehicle exit file, a detector crossing file, and a line crossing file. The log, produced by all CA simulation runs, contains information about the run in question: program name, run date, time step count, vehicle count, used parameter values, and the names of some of the used files. If the run input requests for a queue length listing, a table for the queues is written at the end of the log.

The different files used with the CA simulator are shown in a graphical form Fig. C.2 in Appendix C.

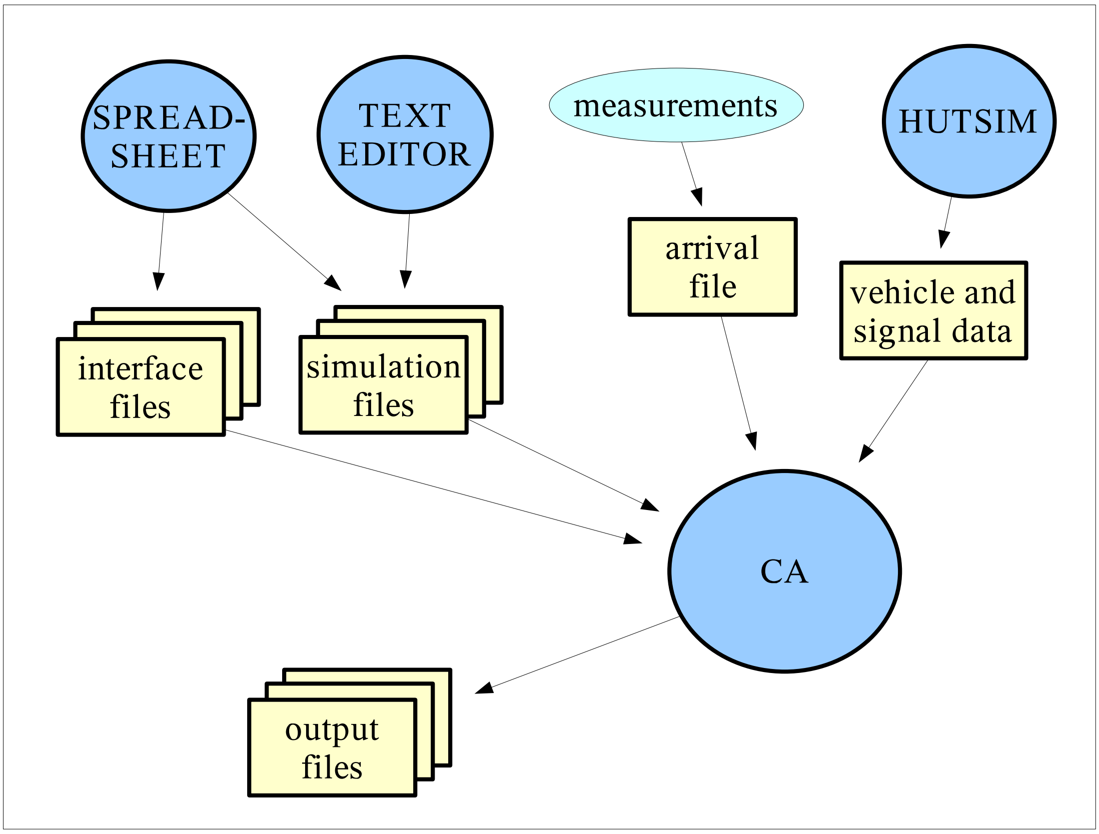


Figure 3.8: File production with the CA simulator.

Fig. 3.8 represents a rough graph of the ways different file groups are prepared and produced in connection with the CA simulator usage. The model-specific files (interface files and simulation files) are prepared either by text editors or as output files from spreadsheet programs. The vehicle and signal data file is usually an output file from HUTSIM, but it can also be produced by some text editor or by a custom-built program. The arrival file contains vehicle entries to the area, and it can be devised, for example, from site measurements.

3.4 HUTSIM — the reference simulation model

The reference traffic that CA simulation behaviour is compared to is created with the simulation program called HUTSIM (Traffic Signal Simulator) [38]. HUTSIM is a versatile traffic simulator program designed for microcomputer use. It has been mostly used for traffic simulations in urban neighbourhoods, and it can be used for different kinds of intersections with a large number of lanes and various signalisation capabilities both for vehicle and pedestrian traffic. The program has mostly been built in the Laboratory of Transportation Engineering at the Helsinki University of Technology.

The HUTSIM simulation process usually begins with the construction of the model area. The HUTSIM environment has a powerful editor for model designing called HUTEDI (Graphic Editor), with which it is easy to build the model road network and install all the necessary items belonging to normal vehicle and pedestrian traffic: roads (pipes), traffic signs, signals, signal controllers, and detectors.

The HUTEDI model construction begins by drawing on the display the roads as one-lane vehicle pipes, which are composed of sections of maximum 50 metres in real space. After the pipes are marked on the network, they are connected to each other according to the logic of vehicle movement, also necessary parallel connections for lane-changing are set up. The arrival spots (origins or traffic generators) and exit spots (destinations) are also marked (usually situated in the network borders), and they are connected to the appropriate links of the network. After the road network is accomplished, routing — the definition of acceptable vehicle routes in the network — has to be performed either manually or by using program's automatic routing mechanism.

When the road map is ready, the required traffic signs, signals, and detectors are placed on the model area. Signs and signals are connected to appropriate pipes so that vehicles moving in a certain pipe know which of them to obey. Signals also need a controller, which guides the changing of the signal status values. Detectors are placed at appropriate spots in pipes and connected to them (a detector may also have to be connected to a signal controller).

The HUTEDI model building process produces as output a configuration file of the model. This file contains all the above-mentioned information about the model that has been entered in the building phase. The configuration file is then later used as an input file for the HUTSIM program in the actual simulation run. Fig. 3.9 shows the files that are often used in connection with HUTSIM and how they are produced.

After the model layout is complete, the vehicle (and the potential pedestrian) input method has to be organised. For this there are mainly two possibilities: self-generation of traffic or use of a prepared arrival file, which contains all the necessary information (time, spot, etc.) about arriving traffic. In the self-generation case, the average flow of vehicles and the distribution of percentages to reach each available destination are defined for every generator. Also, it is possible to define an external traffic file that controls (and changes) the flow at separate origins during simulation. HUTSIM has also other vehicle generation modes, but only the ones mentioned are used in this study. — An arrival file can be an output from a previous HUTSIM run, or it can be gathered from detector measurements in real road environments.

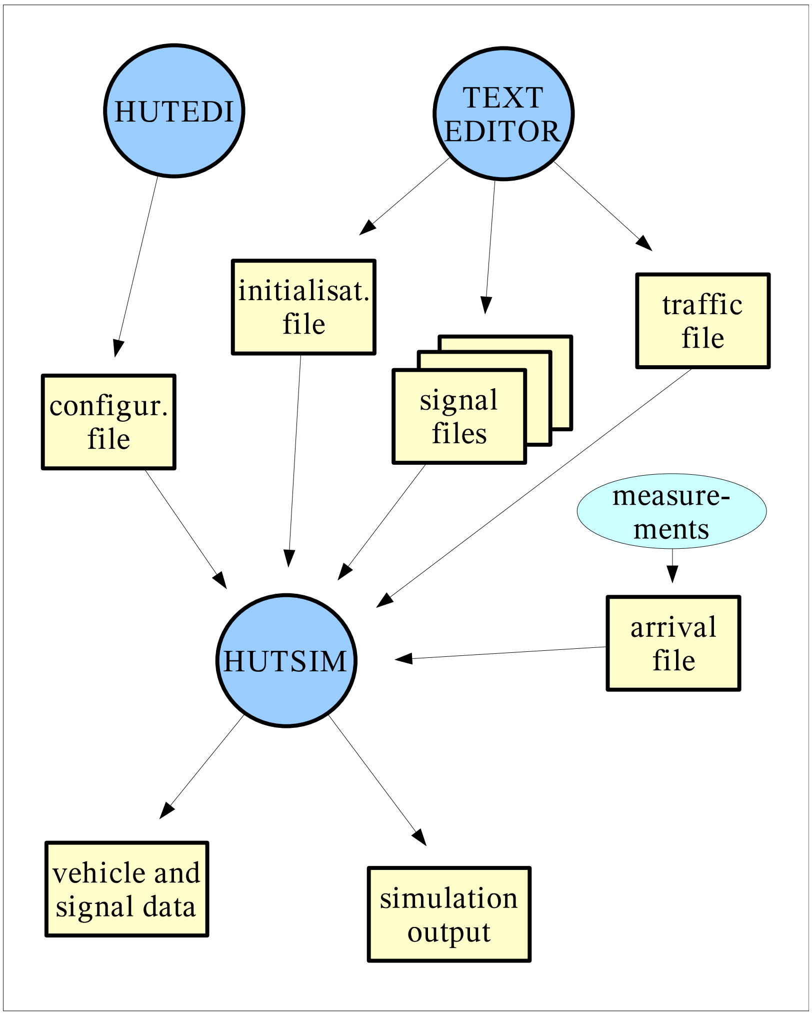


Figure 3.9: Files associated with HUTSIM model construction and simulations in connection with this study.

When an arrival file is used, the data contain the generator number the vehicle is created at and the code of the destination to which it will travel in the area. (In this case the generators do not need turning percentages for the vehicles they create).

If signalisation is needed, it has to be set up, too. The main modes for signalisation are fixed stage and phase ring controlling; in this study only fixed stage controllers are used. The fixed stage signalisation defines the lengths and structures of signal phases, and these remain the same during the whole run. For fixed stages it is necessary to prepare separate signal files for every signal area (covering usually one intersection); the files contain information about period length, stage length, and the order of each signal value for every signal group belonging to the area.

Besides the possibly needed signal and traffic files, a HUTSIM run also needs an initialisation file that contains general information necessary for the simulation. This information contains, for example, the names of the input and output files, the traffic generation mode, the simulation time, a seed number, the time step length, vehicle type distributions, and various parameters concerning the behaviour of different types of vehicles.

The vehicle dynamics used in HUTSIM is based on the so-called car-following principle [38], which assumes the key input variable of interaction between two successive vehicles to be speed according to Eq. 3.1.

$$a_{n+1}(t+t_r) = \frac{\alpha_{t,m}\left[v_{n+1}(t+t_r)\right]^m}{\left[x_n(t)-x_{n+1}(t)\right]^l}\left[v_n(t)-v_{n+1}(t)\right], \tag{3.1}$$

where		
	$a =$	acceleration/deceleration rate
	$v =$	vehicle speed
	$x =$	vehicle position
	$\alpha =$	sensitivity coefficient
	$t_r =$	reaction time
	$m, l =$	constants
	$n =$	leading vehicle
	$n+1 =$	following vehicle

The car-following approach has to be accompanied by other ways of modelling in situations of free driving, yielding, traffic signals, speed limits, etc. Also, a special freeway modelling method has been developed for HUTSIM [38].

Besides on-line animation (on various scales) of the on-going traffic during simulation, HUTSIM has versatile possibilities of monitoring single events for individual objects like vehicles, pipes, signals, and detectors. Additional windows can be used for quick reporting of the simulation events in real time. Also, an option for a written file report at the end of the simulation run is available.

HUTSIM offers several output file types to be selected from: log file (contains all events in the model), arrival file (information about vehicles produced in the model), export file (as the log file but in different form), delay file (information about vehicles passing detectors or leaving the model area), flow file (data from output detectors), GRINT file (signal function information), detsig file (data of vehicles passing detectors and changes in signal values). In connection with the present simulations, only delay file and detsig file (in the text these are referred to as $delayfile$ and $detsigfile$, respectively) of the output files are used. The file format of $detsigfile$ can be seen under the part 'Map for an input file from HUTSIM' in Appendix A.

The $delayfile$ has two types of records: vehicle records (vehicle exit time, the code of the output detector identifier, vehicle identifier number, vehicle type, generator number, destination leg number, etc.) and signal records (signal group and area numbers, cycle length, maximum queue, etc.) The $delayfile$ is used in this study as a reference file for comparing CA simulation output results concerning queue lengths at signal posts, average vehicle velocities on different routes of the model, and vehicle passing over output detectors. — These two file types ($detsigfile$ and $delayfile$) cannot be produced by one single HUTSIM run but by two successive runs, which necessitates keeping the input conditions identical for both runs.

Some of the HUTSIM models used in the present study have been developed from scratch with HUTEDI, some models have been received as complete from the Laboratory of Transportation Engineering. The vehicle and signal material is partly generated by the HUTSIM program and partly received as arrival files. The HUTSIM runs are made as part of this study in all cases except the freeway case, in which the CA results are compared directly to the results prepared earlier.

The HUTSIM runs are performed on Windows NT Workstation version 4.0. The program versions used are: HUTSIM 5.3721 and HUTEDI 5.07.

3.5 Additional programs

3.5.1 Listing of queue length comparisons

The QCMP program (Queue CoMParisons) is used in comparing the queue length values for CA and HUTSIM runs. The QCMP run is made separately for each CA simulation, and it outputs a comparison list for the queue lengths at every signal post for both CA and HUTSIM data.

If the queue length listing is desired for a CA run, then a chart is written to the end of the log expressing the queue lengths for the run at each signal post classified in the ascending order by signal area (1–5) and by signal group (1–12) for every signal period. The queue length is given by the number of vehicles placed at consecutive non-empty cells as counted from the starting cell of the queue (the first cell before the signal post). The counting comprises at maximum all the pipes defined as belonging to the counting process for an individual signal post. In the case of diverging lanes — for example, one lane for the straight going vehicles and the other one for the left turning vehicles — the counting may involve common pipes, but this is usually not a problem because the signals are in different phases. — The queue length is calculated and stored on the time step when the signal turns from red to green.

The HUTSIM queue lengths can be obtained from the $delayfile$ containing also other information besides the lengths. The values are picked from the signal lines for each signal area and group number.

```
  QUEUE LENGTHS FOR CA AND HUTSIM RUNS:    18.08.-04 11:58

  SIGNAL         1          2          3          4          5          6          7
  AREA           1          1          1          1          1          1          1
  SIMUL.      CA   HS    CA   HS    CA   HS    CA   HS    CA   HS    CA   HS    CA   HS
 ----------------------------------------------------------------------------------------
       1       0    0     0    0     0    0     0    0     3    3     2    1     0    0
       2       4    4     0    0     0    0     0    0     6    6     5    5     0    0
       3       9   10     1    0     0    0     0    0     4    4     5    5     0    0
       4       7    7     1    0     0    0     0    0     2    1     3    3     0    0
       5       7    6     1    2     0    0     0    0     2    3     2    2     0    0
       6       7    7     0    2     0    0     0    0     7    7     4    4     0    0
       7       7    5     0    0     0    0     0    0     6    6     4    4     0    0
       8       3    2     0    1     0    0     0    0     6    6     3    3     0    0
       9      17   16     1    0     0    0     0    0     3    3     6    6     0    0
      10      15   14     0    0     0    0     0    0     5    5     3    3     0    0
```

Table 3.4: Example of a QCMP output list.

The QCMP run reads the above-mentioned files and produces a list file of the comparison. Table 3.4 shows an example of (part of) the list. The list can then used as input for drawing figures by some visualisation program.

3.5.2 Listing of average velocity comparisons

The program MVCMP (Mean Velocity CoMParisons) produces output containing comparisons between average velocities for vehicles that have travelled certain routes in the CA and HUTSIM models.

When average velocity calculations are needed for a CA run, the program is asked to write in the associated output file arrival and exit data for every vehicle during the run to be later used in the velocity calculations. The data contain the following items: the time step when exiting the network, the origin node, the destination node when exiting the network, and the arrival time step when entering the network.

Since the route lengths are not stored for vehicle journeys on the CA side, they must be given in a separate (*routelens*) file. The route must not be ambiguous between an origin and a destination. (The distance along a CA route is always the same for every vehicle since lane-changing does not have any effect on it.) The velocity comparisons are made for routes specified in program input.

The arrival and exit data for HUTSIM vehicles can be obtained from vehicle lines in the *delayfile*. Besides the arrival and exit times, these lines also contain the distances the vehicles have passed on their journey.

The velocity comparisons are made for vehicles that exit the model during predetermined time classes calculated from the start of the simulation. The time class length is usually 5–10 minutes, and it is given in the input parameter file.

```
VEHICLE MEAN SPEED VS TIME INTERVAL:   01.10.-04 10:34

         route           11  ->   9     51 ->  3     72 ->  4
          time class        CA   HS    CA   HS    CA    HS

          1:<=   5          9.3   7.6  7.1  6.3  0.0   0.0
          2:<=  10          8.5   8.5  8.3  8.0  0.0   0.0
          3:<=  15          7.9   8.1  8.0  7.1  0.0   0.0
          4:<=  20          9.0   8.0  7.0  7.1  6.7   7.7
          5:<=  25          8.3   8.5  8.5  7.4  8.4   0.0
          6:<=  30          8.2   8.9  8.3  8.1  0.0  12.4
          7:<=  35          9.1   8.6  7.3  6.8  0.0  13.3
          8:<=  40          8.4   8.3  7.8  8.2  0.0   0.0
          9:<=  45          9.7  10.0  7.6  7.1  6.5   0.0
         10:<=  50          8.0   7.9  7.4  7.3  0.0   0.0
```

Table 3.5: Example of an MVCMP output list.

The output list of the run contains a line for each time class giving the average velocity in m/s (calculated as the sum of the route lengths for all vehicles divided by the sum of the travel times for all vehicles exiting during the specified time class) on every desired route for both CA and HUTSIM models. — Table 3.5 shows an example of the comparison list of this program.

3.5.3 Listing of vehicles passing detectors

The program DCMP (Detector crossing CoMParisons) resembles QCMP, but the output contains comparisons for CA and HUTSIM runs between the numbers of vehicles that pass detectors in a pre-defined period of time.

When a detector passing comparison is desired, the CA run is commanded to write in an output file one line of data for every vehicle passing a detector placed somewhere in the simulation area. The data output line contains, for example, the detector code and the time of passing.

For HUTSIM vehicles the necessary data are written into the *detsigfile* among other information, and they can later be sorted out from there by the detector code.

The DCMP run uses as input the above mentioned files and a few parameter files containing information about the detectors used and the length of the detection time during every traffic signal period.

The output list of the run contains a line for each signal period giving the number of vehicles that passed the associated detector in the time period examined. Because the detectors give the number of vehicles for only one lane, a sum for parallel lanes is also given in case it is needed. — Table 3.6 shows an example of the comparison list.

```
CROSSING DETECTORS FOR CA AND HUTSIM RUNS:                     11.10.-04 14:14

DETECTOR   1 (81)     2 (82)                    1 (71)     2 (72)
AREA          1          1                         2          2
SIMUL.    CA    HS    CA    HS    CAs   HSs    CA    HS    CA    HS    CAs   HSs
-------------------------------------------------------------------------------
   1       7     3     4     4     11     7     4     4     0     0      4     4
   2       8    12    15     9     23    21     9    10     5     4     14    14
   3       7    11     8     3     15    14    11    13     5     6     16    19
   4       7     5     8     7     15    12     8    11     8     4     16    15
   5       6    10     5     5     11    15     7     9     3     4     10    13
   6       4     5     5     5      9    10     5     5     5     7     10    12
   7       8    10     9     5     17    15    10    11     4     4     14    15
   8      10    10    11    12     21    22    12    14     8    11     20    25
   9      11    12    17    13     28    25    12    12    10     9     22    21
  10      12    10    13    11     25    21    14    15    12    12     26    27
```

Table 3.6: Example of a DCMP output list.

3.5.4 Traffic incident generator

When a *detsigfile* produced by HUTSIM containing vehicle arrival and signal state change information is not available for a CA model, the traffic incidents (vehicle arrivals and signal status changes) have to be created by some other method. This procedure is needed in connection with the general feature simulations since HUTSIM runs are not performed in this case. As the need is to get vehicle and signalisation input that can easily be varied, a separate program was decided to be constructed. This program was named as EVEGE (EVEnt GEnerator).

The input material for the general feature runs requires vehicle arrivals at random occurrence with a desired average flow lasting for an appropriate period of time. It also requires signal status changes in turn to the green and to the red with a steady and adjustable temporal length for each. The vehicle and signal data are to be written into an output file using the same format as the *detsigfile* has so that it can be read by the CA simulator in the normal way.

Fig. 3.10 shows in the upper part most of the input parameters required by the EVEGE program at run time. The flow and generation time items are on the left hand side: the desired flow rate (given as vehicles per hour) and the start and end times for vehicle generation (given as 1/100 seconds). The items related to signal periods are on the right hand side: the lengths of both the green and red states and the starting time of the first red state (given as 1/100 seconds). Also, the minimum time interval (headway) between consecutive vehicles is given in the input to the program.

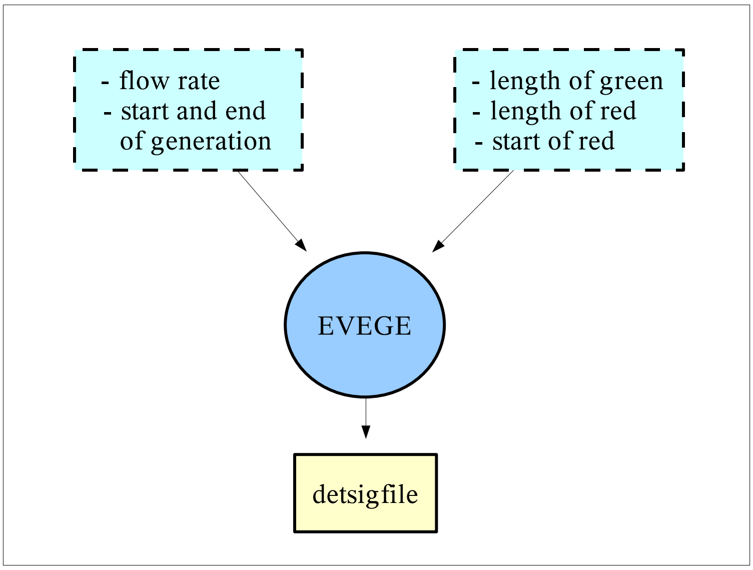


Figure 3.10: Producing a traffic input file with the program EVEGE.

The random occurrence of vehicles at the generation spot can be described by the Poisson distribution, which gives the probability that a certain amount of discrete events (like vehicle arrivals) occur during a certain time when the average rate of these events is known. With traffic studies the distribution is usually shifted by a small amount of time since there exists some interval (minimum headway) during which the next vehicle cannot emerge. This method for the vehicle arrival generation is similar to that used in HUTSIM, and the probability for vehicle appearance can be expressed as [38, 18]:

$$P(h \geq t) = e^{-(t-\tau)/(T-\tau)}, \tag{3.2}$$

where $\tau =$ the minimum allowable headway
$T =$ the average headway
$t =$ time

At run time the program also reads as input the name of the output file and a seed number used by the random number generator, so it is possible to produce output files that differ from each other although the rest of the input values (especially the flow rate) are identical. The EVEGE program generates vehicles only to one origin and signal status changes for one signal head because it is enough for the present purpose.

The output file from EVEGE has the type of *detsigfile* (see 'Map for an input file from HUTSIM' in Appendix A), but the vehicle lines in the file have the arrival file format since the generic feature runs use a similar CA simulator version that is applied with the freeway runs.

3.6 Output analysis

The simulation output from both CA and HUTSIM simulators is analysed essentially as shown in Fig. 3.11.

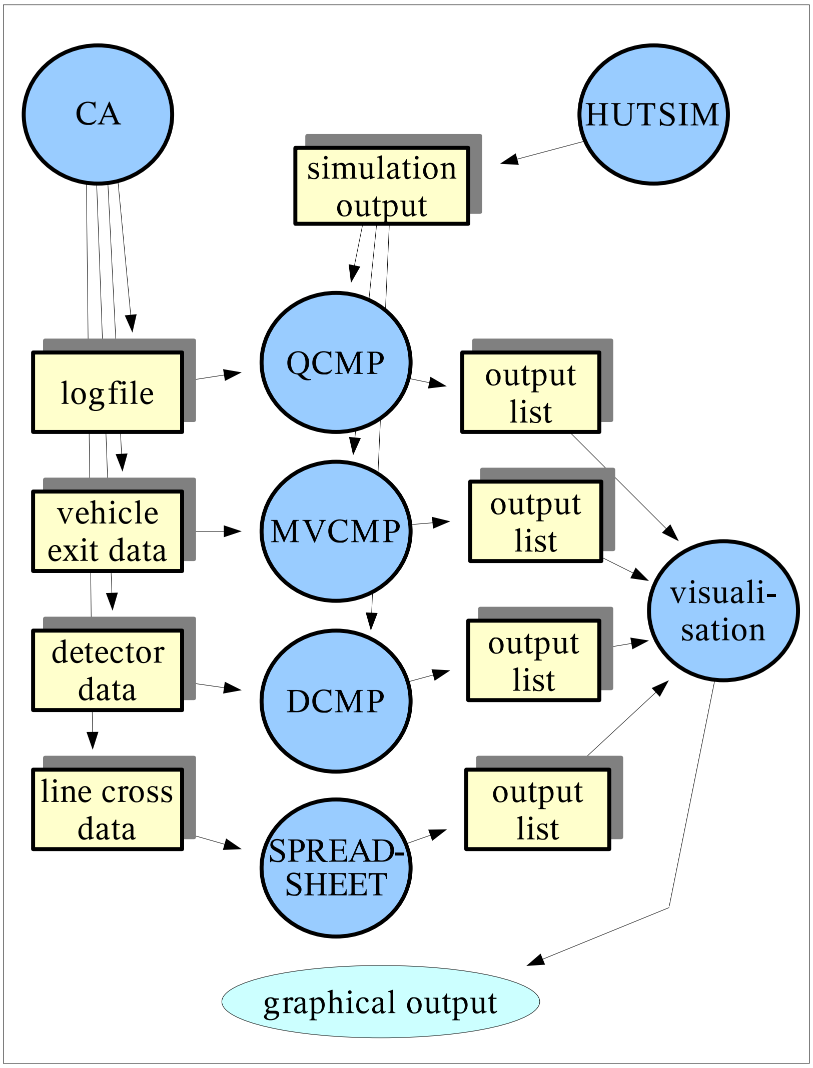


Figure 3.11: Management of simulation output data.

Output from the queue length data (*logfile*), vehicle exit data (*vehleave*), and detector passing data (*cadetsig*) is entered to the appropriate program processing comparison lists with the simulation output from HUTSIM (*delayfile* or *detsigfile*). In the case of line crossing data (calibration of the model, *detfile*), the data are separately processed for CA and HUTSIM. For the general feature simulations, the treatment of the queue lengths is handled by gathering the length information in a separate log during the run and processing it with a spreadsheet program (only CA simulations are used). This procedure is not marked in the figure, but it is equivalent to the case of line crossing data.

The additional programs — QCMP, MVCMP, and DCMP — produce text output lists displaying corresponding items (queue lengths, average velocities, and vehicle counts) for both simulators at given signal periods or time intervals. For easier assessment of the differences in results for both simulators, the results are transformed from output lists into graphical form by some visualisation program (XMGRACE is mostly used in this study). In some cases the graphs are drawn straight from the lists and in some cases average values are first calculated with a spreadsheet program.

3.7 Test arrangements

3.7.1 Traffic generation

As mentioned in Sec. 3.4, the traffic for a HUTSIM run is brought about by self-generation during the run or by reading it from an input file (arrival file). Both methods are used in this study.

Self-generation of traffic means that a HUTSIM run produces vehicles randomly at origin nodes by some predetermined average frequency with a simple negative exponential time headway distribution, see [38]. The frequency is defined as a flow value, vehicles per hour. The flow can be determined separately for every origin node. The generation frequency remains the same throughout the whole run. — At every origin node, a vehicle also gets a destination code at generation time. This code determines the destination node that the generated vehicle tries to reach during the run. The destination is usually chosen at random from the permissible ones according to a probability distribution, which is determined at the time the origin node is set up. The distribution is implemented as different percentages of vehicles to reach separate destinations from the current origin.

Another possibility with the self-generation of vehicles is to use a so-called traffic file. This file determines for every origin node the average flow of vehicles as being valid from a specified time instant. The flow value can be changed to a different value as many times as necessary, therefore the influence of the peak hours, for example, can be simulated. The traffic file overrides the flow values defined for the origin nodes, but the destination distribution is still read from the origin node data.

Besides self-generation, the vehicles arriving to the HUTSIM model can also be read from an external arrival file. This file determines the arrival time instant of every vehicle at some of the origin nodes available in the model. Other items defined for an arriving vehicle include destination leg code, vehicle type, and the desired velocity. — An arrival file can be a result from induction loop detector measurements in a real environment, for example.

The above methods of traffic generation are used with the two- and five-intersection simulations in Secs. 4.3 and 4.4.

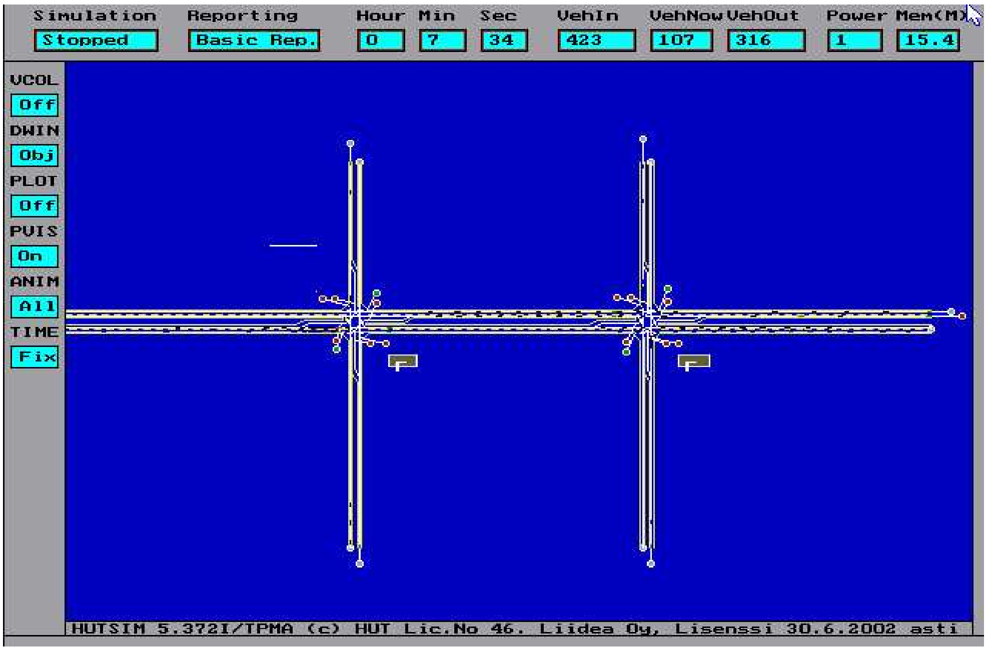
When the HUTSIM program is not needed for result comparisons with the CA simulator, the traffic incidents for CA (vehicle arrivals and signal state changes) must be produced by some other method. If an arrival file is available, it can also be read directly by the CA program, see 'Freeway area simulations' in Sec. 4.5. If not even an arrival file exists, the traffic incidents are generated with the EVEGE program described in Sec. 3.5.4. This program produces vehicle arrivals with the same basic idea that is used in HUTSIM with the desired flow as a major input parameter. If necessary, the program also generates red and green signals for one signal head using a fixed period. The vehicle arrivals and signal state changes are written into a file to be later used as input for a CA simulator run. This latter method of producing traffic incidents is used with general feature simulations in Sec. 4.2.

3.7.2 The HUTSIM test models

Since one of the primary goals for this study was to examine the behaviour of CA models in urban areas with low vehicle velocities, short links, and various intersections, it was decided to begin with a simple intersection area with links sufficiently long for reviewing jam formation. After some experimenting a model with two intersections was chosen.

The model was built from scratch for both HUTSIM and CA test environments. The links (legs) from the intersections to the boundaries of the area are of different lengths: 500 m, 300 m, 250 m, and 170 m in real space. The links between the two intersections are 300 m. Fig. 3.12 shows most of the HUTSIM model area. The horizontal road in the figure consists of two-lane links in both directions. The parallel lanes are mostly lane-changing areas — only those 50 m pipes that contain detectors do not allow lane-changing because vehicles changing lanes at detector positions may cause double detection. The detectors are located in the first pipes at the origin spots and in the horizontal pipes right after the intersections. The vertical roads have only one lane in both directions. In addition, there are short separate lanes at both intersections in all links for left turning vehicles. — Right-handed traffic is assumed in all the HUTSIM and CA models studied.

Fig. 3.13 shows a detailed picture of the intersection on the right as a HUTEDI view. Separate pipes are seen as straight lines between two small circles; the large



Simulation Stopped
Reporting Basic Rep.
Hour 0
Min 7
Sec 34
VehIn 423
VehNow 107
VehOut 316
Power 1
Mem(M) 15.4

VCOL Off
DWIN Obj
PLOT Off
PVIS On
ANIM All
TIME Fix

HUTSIM 5.372I/TPMA (c) HUT Lic.No 46. Liidea Oy, Lisenssi 30.6.2002 asti

Figure 3.12: Part of the 2-intersection simulation area for HUTSIM.

circles are traffic signal groups, a line connects the group head to the associated pipe or pipe pair. The two extra lines connecting parallel pipes are markings indicating the possibility of lane-changing between these pipes.

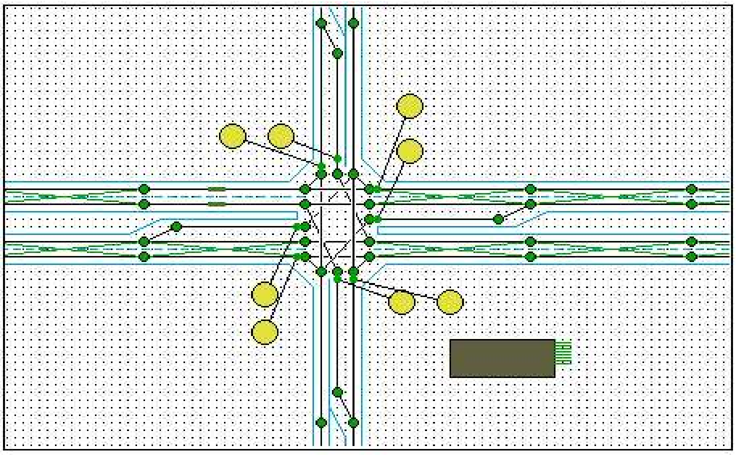
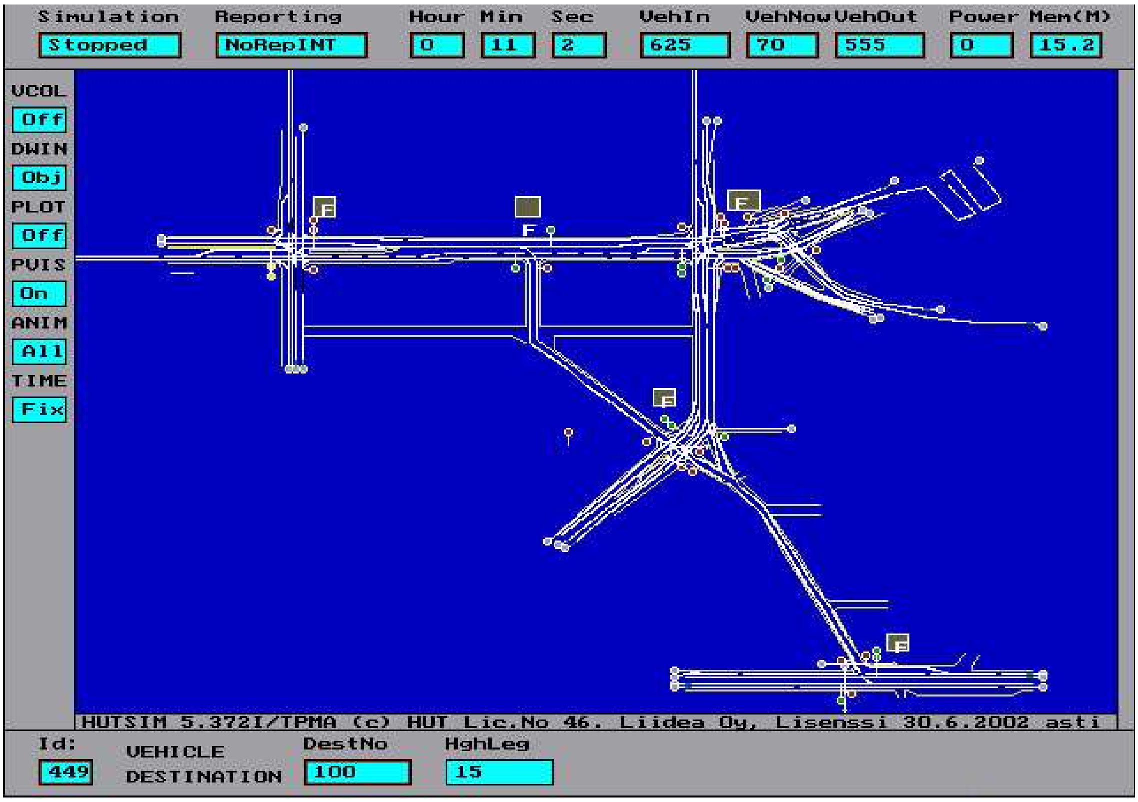


Figure 3.13: The intersection on the right of the 2-intersection area in a close-up as a HUTEDI display.

Detectors in the model are used for counting passing vehicles. Two of them can be seen in Fig. 3.13 as small rectangles on the two upper left lanes leaving the intersection. Similar detectors are located at the start of every incoming lane (not seen in the figure) on network boundaries. Detectors on incoming lanes are used for recording vehicle arrivals to be later used as input data for the CA simulation runs.

The signal controllers use fixed time control with 90 seconds cycle and own phase for the left turning lanes. The offset time between the intersections is 20 seconds. The signalisation is further described in Sec. 3.7.3.

The traffic input contains vehicles of one type (passenger cars) with a desired velocity of 50 km/h. The vehicles are created by random generation at the origin spots, and some of the generators use traffic files, which change the flow values during simulation to produce temporary jams in the intersection area. The routing percentages given to vehicles at arrival times are defined as four separate sets (later referred to as turn rate sets). For closer description of flow profiles and turn rate sets, see Sec. 3.7.3.



Simulation: Stopped
Reporting: NoRepINT
Hour: 0
Min: 11
Sec: 2
VehIn: 625
VehNow: 70
VehOut: 555
Power: 0
Mem(M): 15.2

VCOL: Off
DWIN: Obj
PLOT: Off
PVIS: On
ANIM: All
TIME: Fix

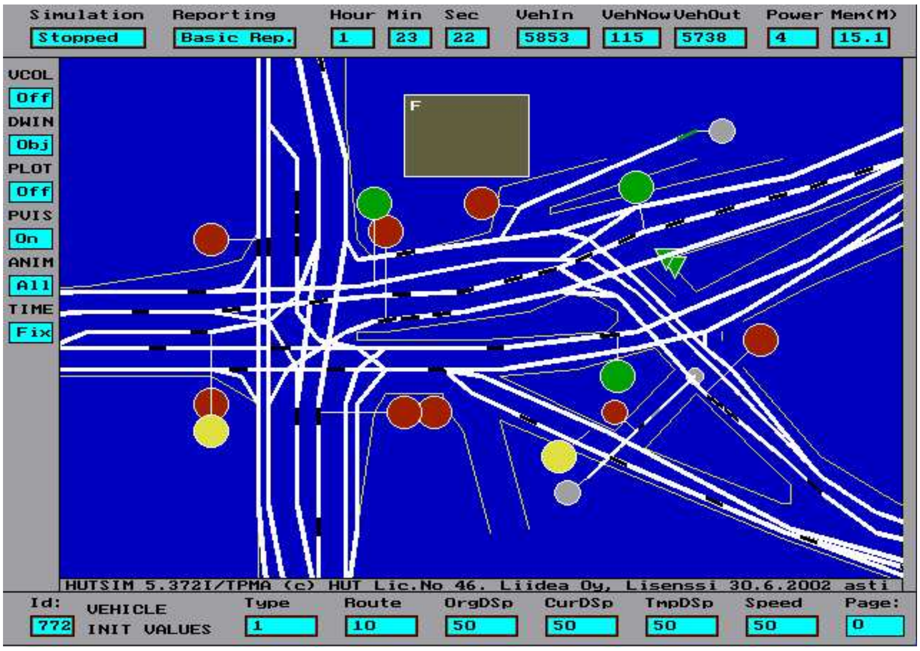
HUTSIM 5.372I/TPMA (c) HUT Lic.No 46. Liidea Oy, Lisenssi 30.6.2002 asti

Id: 449
VEHICLE DESTINATION
DestNo: 100
HghLeg: 15

Figure 3.14: The 5-intersection simulation area for HUTSIM.

Output from the HUTSIM runs comprises either a file containing signal and vehicle data (*detsigfile*) or a file containing simulation output for later analysis (*delayfile*). Since only one file output is possible from a run, the HUTSIM model is run twice for every simulation case in identical circumstances. The HUTSIM model for the two-intersection case was used for every four turn rate sets.

The next model to be considered is a five-intersection case from the city of Tampere in Finland. This area has been earlier used for research in the Laboratory of Trans-



```
Simulation   Reporting     Hour  Min  Sec   VehIn  VehNow VehOut  Power  Mem(M)
Stopped      Basic Rep.    1     23   22    5853   115    5738    4      15.1

VCOL  Off
DWIN  Obj
PLOT  Off
PVIS  On
ANIM  All
TIME  Fix

HUTSIM 5.372I/TPMA (c) HUT Lic.No 46. Liidea Oy, Lisenssi 30.6.2002 asti
Id:   VEHICLE        Type   Route   OrgDSp   CurDSp   TmpDSp   Speed   Page:
772   INIT VALUES    1      10      50       50       50       50      0
```

Figure 3.15: One detail (upper right intersection) of the 5-intersection area for HUTSIM.

portation Engineering, and it has been modelled there with HUTSIM. Hence the HUTSIM model for this area was not built in connection with this study, but this ready-made model was used with its configuration, initialisation, and signal files. The vehicle arrival material is from measurements at real locations of the original test area.

Minor changes were made to the configuration file of the original HUTSIM model for the purposes of this study: some unused detectors were removed, new detectors were added to the start of the incoming lanes, and also, some of the incoming pipes at network boundaries were altered not to allow lane-changing.

The five-intersection area of the HUTSIM model can be seen in Fig. 3.14. The area dimensions in real space are about 900 m times 600 m. Most of the links have two lanes in both directions, and many of them have separate lanes for turning left or right. The right lane in the main directions is usually defined as a bus lane.

The vehicle input and signalisation for this area was also received as complete with the model as one arrival file and five signal files. However, since the CA model can handle only vehicles of one size, the vehicle material in the arrival file was changed to contain only passenger cars (with the desired velocity of 50 km/h) before it was entered to the HUTSIM simulation. — The input material covers simulation time for one five hour period.

Fig. 3.15 shows a detailed view of one part of the model area: the upper right intersection in a close-up with separate lanes and the attachment of signal heads to them.

The five-intersection HUTSIM model was run twice in identical conditions to get the *detsigfile* (for input to the CA model) and the *delayfile* (to be used later for analysis).

Besides the runs described above, one modification of the five-intersection model was made to minimise the effect of the turning percentages on queue lengths and average velocities. This version of the model was otherwise identical to the original model, but new detectors were situated at the start of four 2-lane links; in this way the real amount of vehicles coming to these lanes can be reproduced more accurately in the CA runs. The locations of the new detectors are shown in Fig. 3.21 as a CA model view. The pipes with the new detectors were changed so that they do not allow lane-changing.

The runs with the extra detectors in the model were made in the same way and with the same input material as with the original model.

The third HUTSIM model to be studied is a 5.7 km long straight stretch of a 2-lane freeway with three on-ramps and two off-ramps. The model has been originally built in the Laboratory of Transportation Engineering as part of a study of traffic situations on links by different methods [46]. The model was not applied in the present study for HUTSIM runs, but the results of the runs in the above-mentioned work were used as reference.

The traffic for this test case is input from arrival files. This time the material was entered directly to the CA model since the HUTSIM runs were not made (and consequently the *detsigfile* was not available). The arrival files are described in Sec. 3.7.3.

The HUTSIM test model for the freeway contains one long, horizontal line of road with short on- and off-ramps, therefore a figure of the whole model is not included here. Fig. 3.16 shows one detail of the freeway HUTSIM model: the area of the first on-ramp as a HUTEDI view. — Most of the test area pipes allow lane-changing, only pipes with detectors deny it.

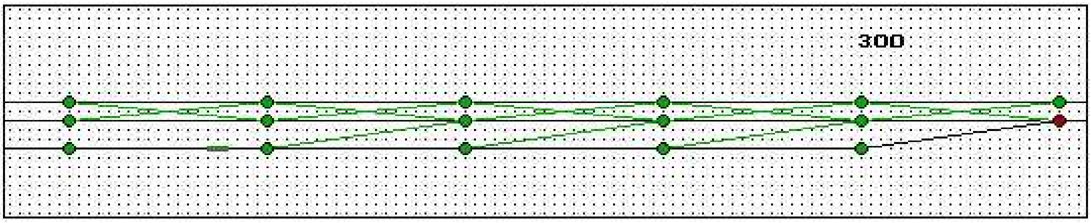


Figure 3.16: One detail — the first on-ramp — of the freeway model, a HUTEDI display.

The HUTSIM model was used mainly in [46] for measuring and assessing the average vehicle velocities in various ways on a road stretch of about 1.75 km starting at the area of the first on-ramp. Vehicles both travelling on the main road and entering from the first on-ramp were used for measurements. Some of the vehicles exit at the first off-ramp while others continue on the main road. The studied area is bounded by detectors whose placement for the corresponding CA model can be seen in Fig. 3.23.

The available vehicle input material consisted originally of several arrival files [46]. For this study only two of them were chosen, representing two somewhat similar flow data with one congestion peak in each, the temporal length of the material in both files is 4 hours 45 minutes in real time. Fig. 3.24 shows the flow profile for the main entry and the first on-ramp lanes as calculated from the corresponding arrival files. The routing for vehicles in these files is not the same for both, and the produced congestion is different for the two cases.

The reference material for the CA runs was received as tables of average velocities of vehicles on the inspected area. The average velocity values had been calculated from the HUTSIM output material, and also, they were assessed by different methods from detector output (these methods are described in detail in [46]).

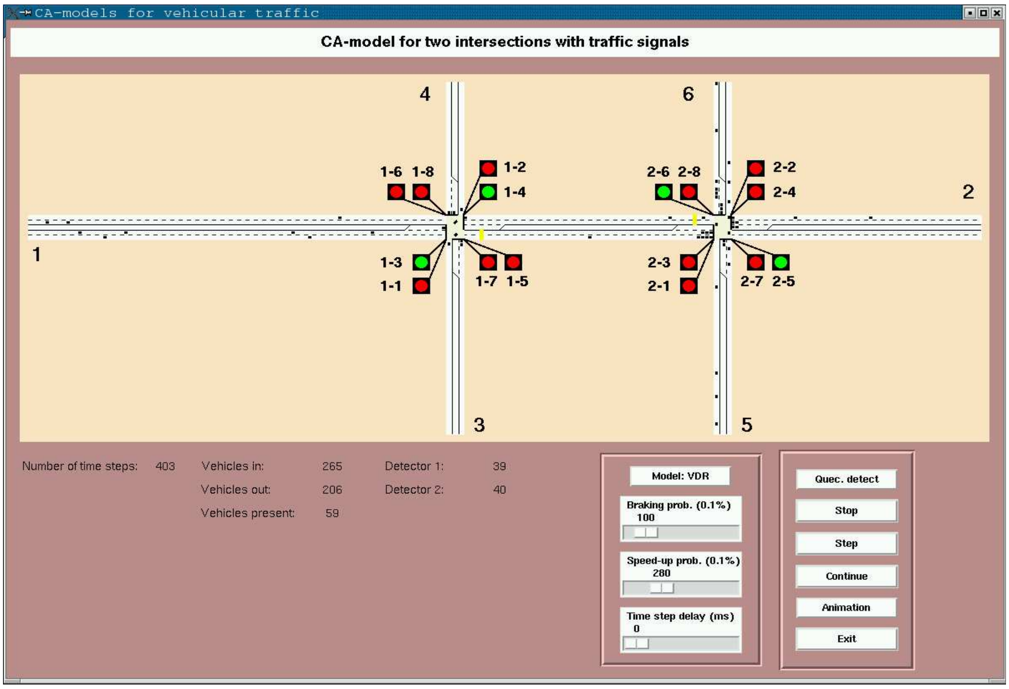
3.7.3 Construction of the CA test models

The CA models for the test areas, whose behaviour is compared to the corresponding HUTSIM areas, were built in physical characteristics as similar to the HUTSIM models as possible. Since the cell length was fixed to 5 metres (in real space), some justifications were required when the HUTSIM link length was not an exact multiple of 5, but in all the differences are only minor.

The features transferred (with possible modifications) from the HUTSIM models are: pipe lengths and positions, signal head positions, yield signs, detector positions, routing information, turning percentage information, and flow rates (static or varying). Some data items had to be added to these: a list of pipes that define new routing after a vehicle leaves an intersection, information concerning vehicle dynamics (deflecting pipe pairs), and pipe chains information (needed when calculating queue lengths). All of the features on the HUTSIM side are given in the configuration file whereas on the CA side they are stored in separate files constructed from the original road plan: pipe naming, lane-changing areas, follower pipes, and information connected to the interface (simulation area on the display).

Many of the different files required by a CA program run (file contents presented in Sec. 3.3.7) were prepared with a spreadsheet program, which automated the process considerably and made it possible to check the consistency of the model.

The CA model for the two-intersection area is depicted in Fig. 3.17. The actual simulation area contains markings indicating the origin (1: origins for lanes 11/12, 2: origins for lanes 21/22, 3: origin for lane 31, 4: origin for lane 41, 5: origin for lane 51, and 6: origin for lane 61) and destination spots (the same markings are used as for the origins). Markings for the traffic signal heads are also shown: the



CA-models for vehicular traffic

CA-model for two intersections with traffic signals

Number of time steps: 403

Vehicles in: 265

Vehicles out: 206

Vehicles present: 59

Detector 1: 39

Detector 2: 40

Model: VDR

Braking prob. (0.1%) 100

Speed-up prob. (0.1%) 280

Time step delay (ms) 0

Figure 3.17: The 2-intersection area for CA.

left turning short lanes are controlled by signal heads 1–3, 1–4, 1–7, 1–8, 2–3, 2–4, 2–7, and 2–8; other heads control straight going and right turning lanes.

The green times for signals heads are listed in Table 3.7 for both intersections of the model.

green time (s)	signal heads
30	1-1, 1-2, 2-1, 2-2
11	1-3, 1-4, 2-3, 2-4
18	1-5, 1-6, 2-5, 2-6
7	1-7, 1-8, 2-7, 2-8

Table 3.7: Green times for signals in the 2-intersection area (values do not contain the additional 3 seconds of yellow allowing passing).

Fig. 3.18 shows the flow rates for vehicle input at the origins 1, 2,..., and 6. The flow rates apply separately to each lane connected to the origin (if there are several of them).

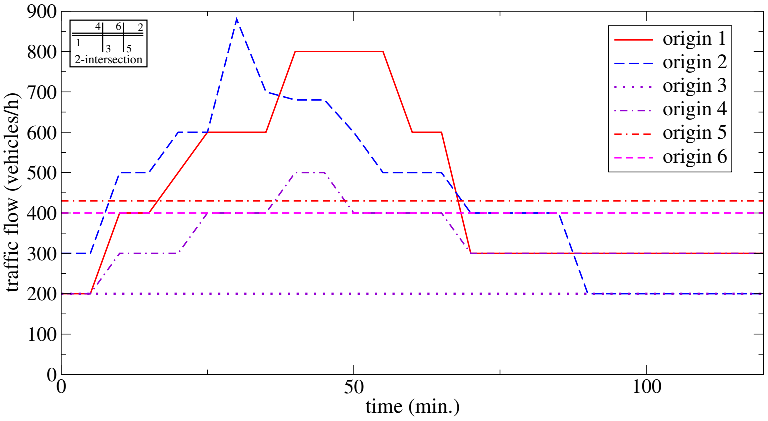


Figure 3.18: Flow profiles per lane for the 2-intersection simulations as vehicles/hour.

The turning percentages in the two-intersection case for different turn rate sets are depicted in Table 3.8. The values on rows in the table describe the percentages for vehicles from the origin (given by the code on left) to reach the corresponding destination (given by the column header). Value '–2' means no access to the destination.

	set 1						**set 2**					
	destination											
origin	1	2	3	4	5	6	1	2	3	4	5	6
11	-2	100	0	0	0	0	-2	83	10	2	5	0
12	-2	100	0	0	0	0	-2	90	0	8	0	2
21	100	-2	0	0	0	0	85	-2	0	3	0	12
22	100	-2	0	0	0	0	87	-2	4	2	7	0
31	0	100	-2	0	0	0	12	85	-2	0	3	0
41	0	0	100	-2	0	0	10	5	85	-2	0	0
51	0	0	0	0	-2	100	5	12	3	3	-2	77
61	0	0	0	0	100	-2	10	5	0	2	83	-2

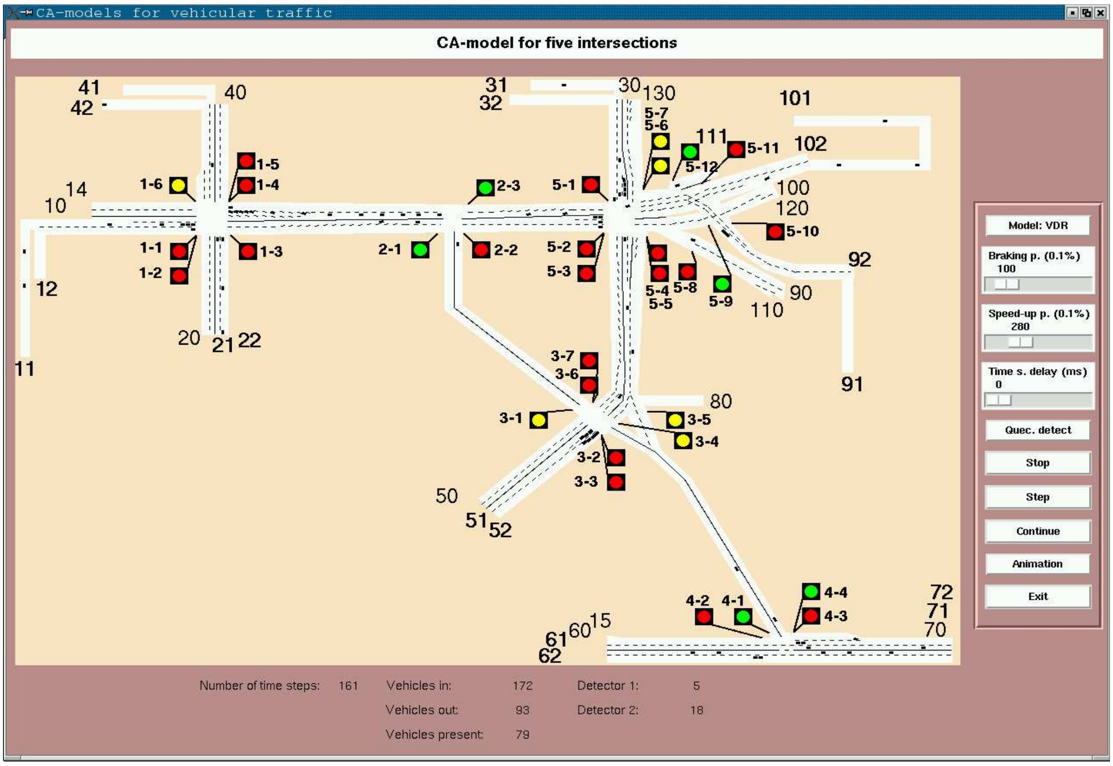
	set 3						**set 4**					
	destination											
origin	1	2	3	4	5	6	1	2	3	4	5	6
11	-2	70	10	5	15	0	-2	50	20	7	20	3
12	-2	72	4	15	0	9	-2	53	6	20	3	18
21	70	-2	3	15	0	12	51	-2	4	24	1	20
22	72	-2	8	5	15	0	50	-2	13	10	22	5
31	12	70	-2	8	8	2	21	54	-2	10	12	3
41	10	13	70	-2	2	5	15	20	50	-2	5	10
51	5	14	5	6	-2	70	10	18	8	12	-2	52
61	20	10	0	2	68	-2	25	15	4	8	48	-2

Table 3.8: Turning percentages for different destinations in the 2-intersection area: turn rate sets 1, 2, 3, and 4.

The turn rates start from the simple case when vehicles follow straight routes: 1 → 2, 2 → 1, 4 → 3, etc., with no vehicles turning to side directions. The only exception is origin 3 from which all vehicles turn right in the first intersection. As the turn rate set number increases, deviations from the main direction grow larger. In the set 4 the main direction share is no longer more than about 50 %.

The CA program also requires new turning percentages for those vehicles that have crossed an intersection and are heading towards a new intersection. At the moment the program uses a system in which these middle-area percentages are calculated beforehand from the corresponding HUTSIM route selection percentages (given at the origins) using the average flow rates for different lanes (see Sec. 3.3.5).

The CA model area for the five-intersection case was prepared in the same way as the two-intersection case. The only difference was that the HUTSIM model was received as already complete from the Laboratory of Transportation Engineering. Since the cell length on the CA side is fixed, modifications to the CA pipe lengths were needed with this network. Nevertheless, the link and route lengths could be kept quite close to the original ones and the deviations are only of the order of a few metres on the whole lengths of the routes. The angles of the lanes have also been kept as close to the HUTSIM model as possible, so the network appearance is very similar to the original one.



Braking p. (0.1%) 100

Speed-up p. (0.1%) 280

Time s. delay (ms) 0

Number of time steps: 161 Vehicles in: 172 Detector 1: 5

Vehicles out: 93 Detector 2: 18

Vehicles present: 79

Figure 3.19: The 5-intersection area for CA.

In Fig. 3.19 is depicted the CA simulator interface for the five-intersection area. The display contains marking of the origins, destinations, and signal heads. Signal heads use codes 1-1, 1-2,..., and 5-12. Purely left turning lanes are regulated by the heads 1-1, 1-4, 3-2, 3-6, 5-2, and 5-6. Signal heads 4-4 and 5-5 are for the right turning traffic only.

A few lanes in the figure are diverted at the start to get the display area more compact, but this does not change the vehicle behaviour at the lane corners as compared to straight stretches (lanes 11, 12, 31, 32, 41, 42, 91, and 101).

Labels for the origin spots of different lanes are marked with numbers from 11 to 102: for origin 1 lanes 11 and 12, for origin 2 lanes 21 and 22, etc. Included are also destination labels (10 and 14, 20, 30 and 130, 40, 50, 60 and 15, 70, 80, 90 and 110, 100 and 120) because some of them are not evident from the context.

The green times for signal heads are listed in Table 3.9 for the intersections of this model.

green time (s)	signal heads	green time (s)	signal heads
8	1-1	25	4-1
27	1-2	55	4-2, 4-3
14	1-3	66	4-4
13	1-4	15	5-1, 5-10
34	1-5	5	5-2
16	1-6	23	5-3
64	2-1, 2-3	18	5-4, 5-5
16	2-2	14	5-6
16	3-1, 3-4	30	5-7
8	3-2	65	5-8
31	3-3	66	5-9
43	3-5	26	5-11
23	3-6	3	5-12
44	3-7		
17	3-8		

Table 3.9: Green times for signal heads in the 5-intersection area (values do not contain the additional 3 seconds of yellow allowing passing).

As in the case of the two-intersection area, the turning percentages are needed for CA vehicles entering the model area and new percentages for vehicles in areas between intersections, too. The calculation is again performed beforehand. — For vehicles entering the model, the percentages are calculated from the original arrival file, which gives the origin-destination (route) data for every vehicle. The percentage value for turning into certain direction in the first intersection after arrival is calculated as a partial amount of vehicles taking that direction of all the vehicles arriving from that origin. For vehicles moving in the areas between intersections, the percentages for turning in the next intersection are calculated from the average flow to the lane in question and the original routing data of those vehicles entering the lane; the method is the same as with the two-intersection case. The routing percentages in the five-intersection case are shown in Table 3.10.

	destination													
origin	10	20	30	40	50	60	70	80	90	100	110	120	130	14	15
11	0	2	7	14	3	0	3	0	26	45	0	0	0	0	0
12	0	31	0	0	5	0	11	0	0	0	19	30	4	0	0
21	47	0	0	49	0	0	0	0	0	0	0	0	0	4	0
22	0	0	3	57	4	0	0	0	11	25	0	0	0	0	0
31	10	2	0	1	42	1	8	0	24	12	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
41	0	51	4	0	2	0	2	0	14	27	0	0	0	0	0
42	28	68	0	0	0	0	0	0	0	0	0	0	0	4	0
51	4	1	93	1	0	0	0	0	0	0	0	0	0	1	0
52	0	0	0	0	0	0	6	26	25	43	0	0	0	0	0
61	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
62	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
71	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0
72	6	0	10	1	6	69	0	1	2	3	0	0	0	1	1
91	48	11	16	5	13	0	3	0	0	4	0	0	0	0	0
92	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
101	49	12	14	5	16	1	3	0	0	0	0	0	0	0	0
102	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
111	69	0	31	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.10: Routing percentages for different destinations in the 5-intersection case.

The vehicle data for this model consist of one input file (arrival file), which comprises vehicle arrivals from real measurements in the test area. The recorded period is 3 hours 45 minutes long, and the flow varies considerably on different routes during that time. Fig. 3.20 describes the flow changes at some of the input spots on the test area (from origin 1 to origin 10). A given flow value in the figure is an estimate calculated from the arrival file as the number of vehicles appearing at the input detector during a five minute period. The value is then expanded to vehicle counts per hour during the specified period.

The five-intersection area is also studied with an arrangement in which the CA vehicles are created according to their real incidence in the HUTSIM run for some lanes (in areas between the intersections). The idea was to correct the amounts of vehicles arriving to some selected lanes as compared to the amount values given by the turn rate assessment (which is always subject to statistical deviations). This experiment was accomplished by placing detectors at the starting points of eight lanes of the HUTSIM model and recording the HUTSIM vehicles that passed these detectors. This information was then written among other items in the *detsigfile*.

After crossing an intersection and before entering a lane with the extra detector, the CA vehicles on these lanes are deleted and new ones created at the same time instances as the HUTSIM vehicles hit the equivalent detectors.

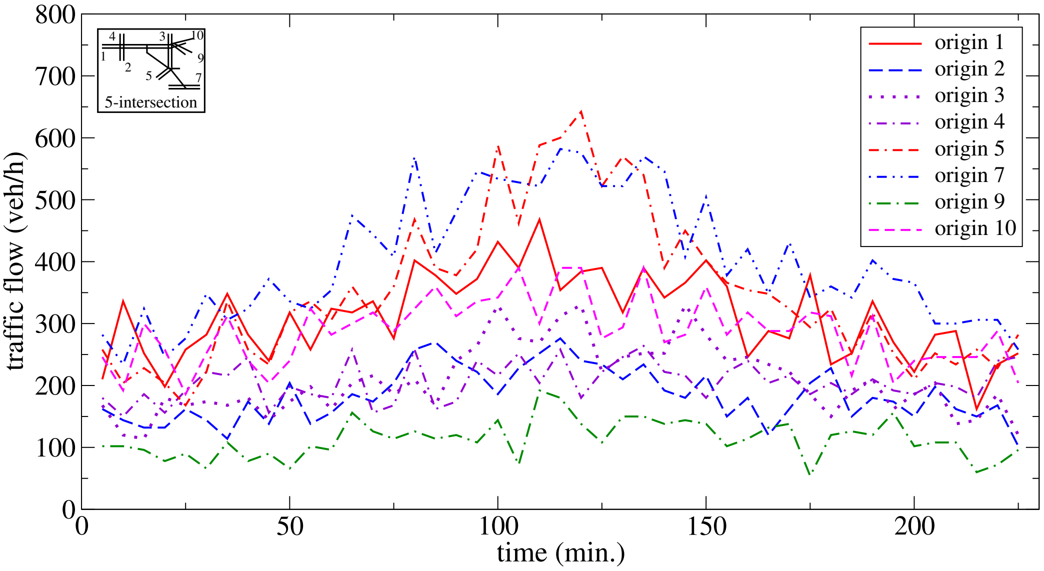


Figure 3.20: Traffic flow data for the 5-intersection area.

Besides the correct time instance to start the corresponding link, also the real amounts of vehicles for the lane can thereby be reproduced in the CA simulation, which can lead to more precise queue lengths at the next signal posts on the associated links. The turning percentage data used for these runs are the same as in the normal model. — The extra detectors for the CA model are marked in Fig. 3.21.

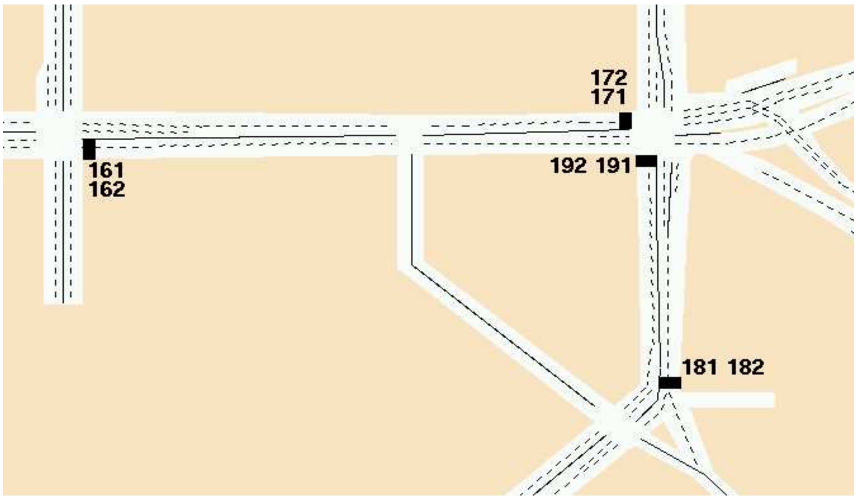
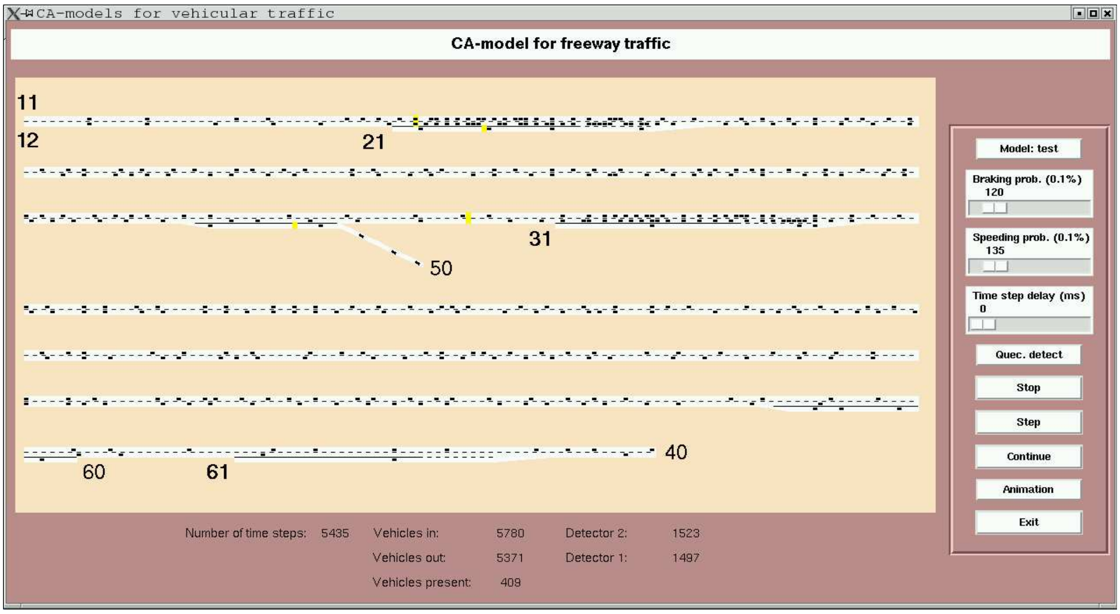


Figure 3.21: Placement of the extra detectors in the 5-intersection case.

The removal and addition of vehicles at the sites of the extra detectors can be expected to have no significant disturbance on the flow of traffic. This is because the vehicles are created in the detector spots with maximum velocity (3 cells/time step) that equals the probable velocity they would have even when leaving the previous traffic signal from a standstill. Furthermore, the detector spots do not involve strongly deviating pipe pairs that would require a low velocity start at the spot.

The third network case is a freeway model comprising a stretch of a two-lane freeway with on- and off-ramps. This model has no intersections and no signalisation and the form of the road is a straight line.

Fig. 3.22 shows the freeway CA model. Again, the original HUTSIM model for this area was not constructed as part of this study, but it was received as such from the Laboratory of Transportation Engineering (prepared as part of [46]).


CA-models for vehicular traffic

CA-model for freeway traffic

Model: test

Braking prob. (0.1%) 120

Speeding prob. (0.1%) 135

Time step delay (ms) 0

Number of time steps: 5435
Vehicles in: 5780
Vehicles out: 5371
Vehicles present: 409
Detector 2: 1523
Detector 1: 1497

Figure 3.22: Interface of the CA freeway model.

In this study the HUTSIM model was only used as help to construct the CA model. The corresponding CA model was built somewhat different in visual appearance by cutting the freeway stretch into shorter pieces and placing them side by side on the display area for easier inspection of the traffic. Other reason for this solution is that the CA program does not have any zooming capabilities, so the vehicles would have been ’lost in between the screen pixels’ on a road of this length. The cutting points of the road do not affect the dynamics of the vehicles.

The CA simulator's lack of dealing with three parallel lanes caused a problem at the on-ramps of the studied area. The changes to the code itself would have been so tedious that this way of action was given up, and the on-ramp lanes were changed to sequences allowing lane-changes only on two adjacent lanes at a time. Since the on-ramp lanes are quite long, this arrangement seems not to create large differences between the two simulators' behaviour.

Only average velocities of vehicles are studied for the freeway model. The input material consists of two arrival files, which both cover an about five hour period of vehicle traffic on a real freeway stretch. The HUTSIM runs were performed in the Laboratory of Transportation Engineering in connection with [46] using arrival files from measurements mentioned above. Since the average velocity values for those runs were also received from the laboratory, the HUTSIM runs were not repeated in connection with this study.

The CA runs for this model area use the same input material (the arrival files) as the HUTSIM runs have used. Because the CA simulation program does not read an arrival file as such, the input part of the program was changed so that it can read the vehicle arrivals directly from that file instead of using the normal *detsigfile* produced by a HUTSIM run.

The average velocity values are measured only on part of the 5.7 kilometre freeway. The stretch used for measurements is located between the first and the second ramp area (origins 21 and 31 in Fig. 3.22). The exact positions of the detectors (71, 72, 73, 81, 82, and 83) are marked in Fig. 3.23. The velocities of the vehicles on this area are strongly influenced by the traffic coming from the first on-ramp. At times when the flow values are high, the traffic from the second on-ramp (outside the stretch used for measurements) causes so much congestion that it affects the inspected area, too.

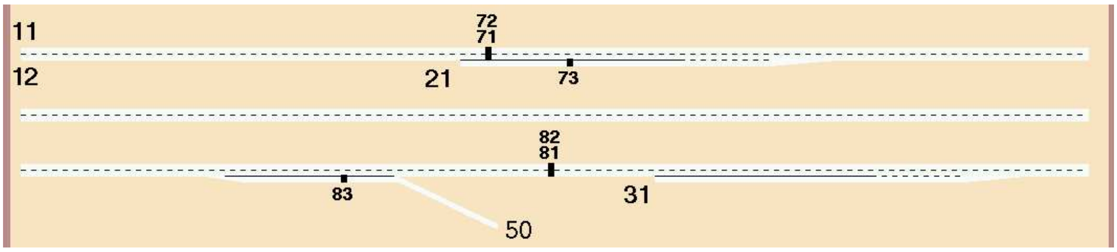


Figure 3.23: Detectors 71, 72, 73, 81, 82, and 83 on the freeway area.

The two arrival files (file 1 and file 2) have been prepared from measurements on the main direction of the road (corresponding to origins 11 and 12) on a freeway in Southern Finland, and the traffic from the on-ramps has been formed as percentages from the main direction flow: 38 % for ramp 1 (origin 21) and 31 % for ramps 2 and 3 (origins 31 and 61) [46]. The flows from origins 11, 12, and 21 as calculated from the arrival files for 15 minute periods can be seen in Fig. 3.24. Because the

CA simulator can handle only vehicles of one size, the vehicle types in the input material were changed to contain normal passenger cars only.

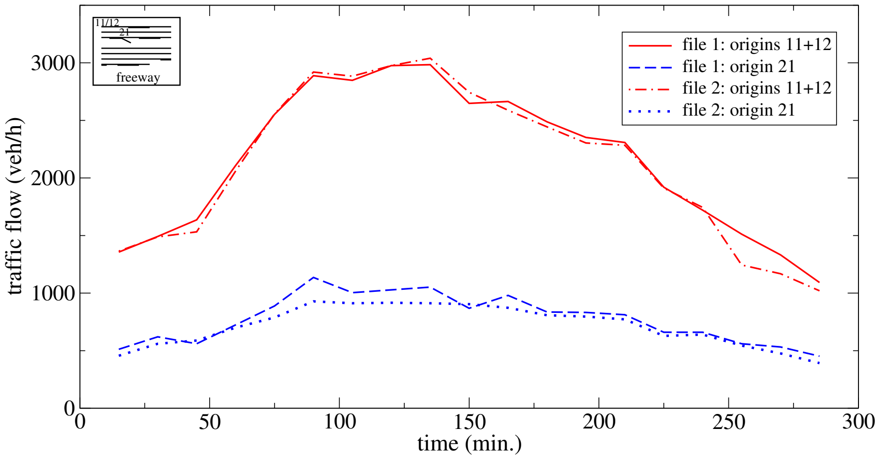


Figure 3.24: Vehicle flow from origins 11, 12, and 21 in the freeway model for the two arrival files.

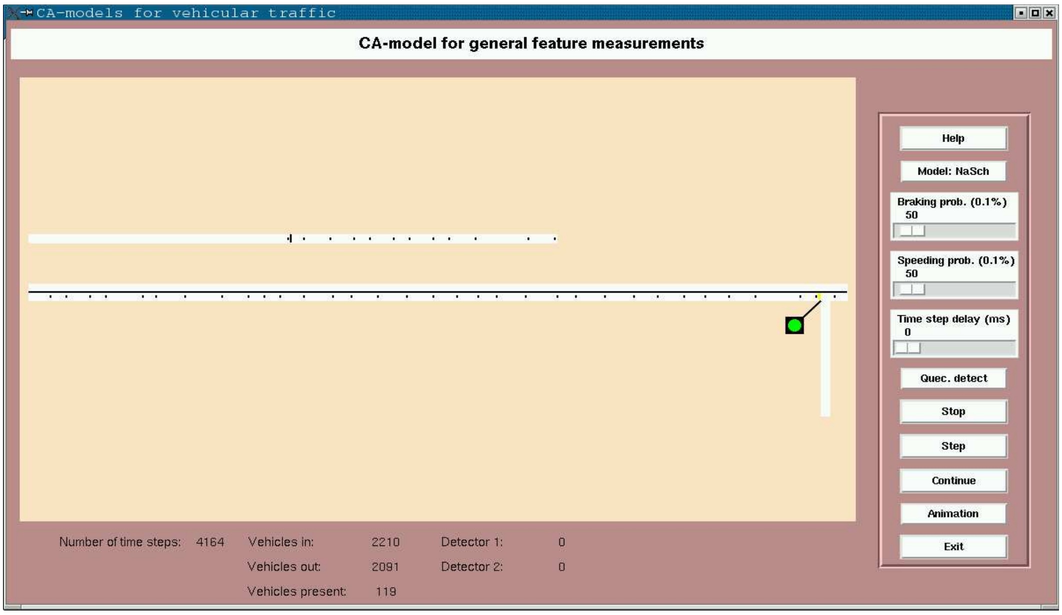
The origin-destination maps used in the HUTSIM runs in [46] can be seen in Table 3.11. These maps were used to calculate the turning percentages for the CA vehicles with the help of the average flows from each origin in the model (the procedure is practically the same as with the two earlier model areas). — The average flows were calculated from the arrival files as the total number of vehicles from an origin divided by the total simulation time (5 hours). In the percentage calculation, it was also assumed that after each on-ramp (where vehicles are given new percentages for behaving at the next off-ramp) both lanes are equally occupied.

	case 1			**case 2**		
	destination					
origin	4	5	6	4	5	6
11	70	15	15	60	20	20
12	70	15	15	60	20	20
21	80	5	15	70	10	20
31	95	0	5	90	0	10
61	100	0	0	100	0	0

Table 3.11: Origin-destination maps (as percentages) of the two input cases for the freeway area runs.

The generic feature runs are performed solely with the CA simulator, therefore it was necessary only to construct a CA model for this part of the study. The plan was to examine queue lengths or the range of influence of queue formation at traffic signals, which may serve as a pattern for queue building caused by other reasons, too. To keep the studied area simple, it was decided to use a one-lane road stretch that has one signal head at the end of it.

Fig. 3.25 depicts the arrangement of the model area for the generic feature runs. In fact, only part of the straight stretch used for measurements is seen in the middle of the figure since the whole length is five kilometres in real space (1000 cells) and it did not seem necessary to have it all visible. The road area in the centre part of the model coincides with the starting one third of the road, and the rest of the road up to the signal head is not seen. The vehicle movement occurs only on one lane of the area (from the left to the right in the figure), and the right turning lane located after the signal head is not used.


CA-models for vehicular traffic

CA-model for general feature measurements

Braking prob. (0.1%) 50

Speeding prob. (0.1%) 50

Time step delay (ms) 0

Number of time steps: 4164

Vehicles in: 2210

Vehicles out: 2091

Vehicles present: 119

Detector 1: 0

Detector 2: 0

Figure 3.25: Interface of the model for studying general features.

The upper shorter road stretch in the figure serves as a buffer area for feeding vehicles to the actual measurement area. The use of the buffer road is necessary since stop-and-go waves travel backwards on the road and easily suffocate the origin spot area when the flow values get heavier.

To get the buffer pipe area flexible for vehicle input, it was necessary to change the vehicle feed part of the simulator program code because a particular cell as a sole vehicle feed origin may not occasionally be enough in spite of the use of the buffer cells. The problem was solved by allowing several cells act as origin spots when new vehicles are fed frequently. In the present model case, the origin spot is the first free cell (minus a constant depending on the maximum velocity limit) before the middle part of the buffer pipe (the vertical line on the pipe in the figure). The buffer pipe length is 200 cells. If the vehicle input rate is very high, even the buffer pipe area may not be sufficient for new vehicles, which leads to a premature termination of the run. Although the buffer pipe and the measurement pipe are in different parts of the figure, the vehicles move directly from the former pipe to the latter one and the displacement point does not affect the dynamical movement of vehicles. However, the vehicles in the buffer pipe are not subject to normal breaking percentages of the used CA rule, but they move deterministically to reduce the effect of the stop-and-go waves near the feeding area.

The generic feature runs comprise measurements of the lengths of the queue influence area on successive signal periods. A measurement starts from the prevailing queue length in front of the signal head when the red signal turns to green (and the first vehicle starts to pull away) and continues during the next time steps until no new vehicles join the queue from the upstream end, that is to say, until the remains of the queue dissolve. The resulting length value is written to a log for later processing.

As the queue formation is studied here as a generic phenomenon, the queue lengths are examined from the statistical point of view, and a large amount of separate measurements is required. (In this connection the queue length is meant to cover the whole influence area of queue building although the shorter term is used.) Because the flow rate may affect the queue length distribution, the measuring covers flow values starting from 100 vehicles/h with intervals of 200 (veh/h) to as high values as possible (until the vehicle feeding limit is reached).

The minimum allowable headway (parameter τ in Eq. 3.2) for input file production is set to 1.3 seconds to avoid too frequent vehicle appearances at the origin area. The red signal temporal length is usually quite low (often 100 seconds), and the green length is about 10–30 times that of the red one so that the queues have time to dissolve properly before the next red starts to build a new queue.

The measurement procedure is organised by preparing 20 different input files (of the type $detsigfile$) for each flow value with a long enough simulation time giving a little over 260 queue incidents for each file. Altogether, 5220 queue values are measured for every flow value. The input files are produced with the EVEGE program (see Sec. 3.5.4) with separate seed numbers for the required 20 input files.

Although the CA model that is mostly used with the study is VDR, the freeway model is also simulated with a test CA model, which tries to improve some of the VDR's characteristics found somewhat disturbing in freeway traffic: lack of anticipating velocity drops caused by obstacles and low velocity movement in dense traffic. The test model used here is not a result of thorough development but more of a test type attempt for improvement.

The test model is basically a normal VDR model, but vehicle acceleration is prohibited in case occupation is dense in front — also deceleration is possible. This special action is taken if the gap to the next vehicle ahead is smaller than a velocity dependent parameter, reference gap (*refgap* in Eq. 3.3), where v is the velocity of the vehicle.

$$refgap = \sum_{i=1}^{v} i + 4, \quad (v > 0) \tag{3.3}$$

Acceleration changes depend on the density of vehicles ahead in a velocity dependent range (starting from the next vehicle ahead within the reference gap). Table 3.12 lists the possibilities for choosing the value for the braking parameter p and the ways the velocity is changed. The first one of these four that fits is chosen.

1. v < 1 .OR. gap > refgap: $p = p_0$, $v := v + 1$
2. 2*v > refocc+2: $p = 0.98$
3. v < VMAX .AND. 2*v < refocc+2: $p = p_0$, $v := v + 1$
4. otherwise v and p do not change

where refocc = MAX(sites,range) – vehicles in sites

Table 3.12: Acceleration for the CA test rule (in refocc comparisons v is expressed in steps).

In Table 3.12 the value for the 'range' is $2 * v$ and the 'vehicles in sites' sums the total number of vehicles found within this range. The present program version calculates the range only over (at maximum) two pipes, and if the range extends beyond that, merely the number of the sites available is used. The same applies to situations when the present pipe is the last one on the route.

3.7.4 Generic features of traffic

Generic features of traffic are studied with CA simulations by measuring the range of vehicles forced to stop because of queue building. According to the test arrangement (see Sec. 3.7.3), the total influenced area in the number of vehicles is measured on a single lane before a signal head.

The measuring of the queue (building influence area) length is requested in a CA simulation run with the parameter *queuemaxon* (=1) in the input file *generalpar*. The calculation starts at the moment the signal status is changed from red to green and continues until the queue dissolves. The dissolving point is defined as clearing of the cell that marked the maximum queue length during the previous time step. (Some congestion area may still be left around this cell, and this cluster may either dissolve or the congestion starts to move upstream). The resulting length values are printed on the workstation screen or to a file defined to collect the run time output (but not to the run-specific log).

Since the queue length distribution is to be examined statistically, a considerable number of length measurements are required for each flow rate value. For this purpose, 20 equally large traffic input files are generated for every flow value and the runs are performed in sequence for one flow level at a time. The queue length output from this set of runs is gathered in one log file that is further processed by a spreadsheet program.

The queue length results for one run set (one flow value) are transferred to a spreadsheet table, and the number of queues as a function of the length is counted. For each length value, the fraction of the number of those queue incidents with the length greater than or equal to this length (from all queue incidents) is then calculated and drawn to a graph using logarithmic scale. Examples of these graphs are seen in Sec. 4.2 and in Appendix B.

Measurements are carried out for various CA models and a couple of maximum velocities (VMAX = 3 and/or 5 cells/time step) are used. The flow values are varied (starting from 100 veh/h) for each model and the used VMAX. The CA model related general parameter values are the same ones as used with the intersection and freeway runs. The results for the generic feature runs are reported in Sec. 4.2.

3.7.5 Queue lengths

The comparison between the HUTSIM and CA simulators' behaviour concerning vehicle queue lengths at signal posts was carried out for the both two- and five-intersection cases. The measuring of the queue lengths is executed in both programs for all signal posts, but the results reported are a chosen selection of posts in the two case environments. The results for the two test areas are reported in Secs. 4.3.2 and 4.4.1.

The HUTSIM program is run twice with identical initialisation for every model because two output files are needed: $detsigfile$ to be used as input for CA vehicle and signalisation data and $delayfile$ for queue length comparisons. The $delayfile$ output is obtained from a HUTSIM run by simply selecting this file type as output type when starting the HUTSIM run (see file description in Sec. 3.4).

The queue length data gathering is requested with the *generalpar* parameter named *quelenon* (=1) for the CA simulator. The queue lengths are measured at the signal post every time the signal value changes from red to green. The length values are collected into a run-time array, and they are written at the end of the run to $logfile$. At the moment the measuring assumes fixed signal periods, and it starts when a chosen signal turns from red to green (definition of the chosen signal post is at the start of the *tlights* file).

The measured queue length values for both simulators can be compared by running the QCMP program (see Sec. 3.5.1). From the output list of QCMP, graphical comparison curves are drawn with a separate visualisation program (examples of figures in Secs. 4.3.2 and 4.4.1 and in Appendix B).

Since randomness plays a significant role in the CA results, the CA runs are performed in this study several times (usually 10) against one HUTSIM run using different seed numbers. Comparisons are then made between different CA runs and between an average queue length curve against the HUTSIM curve. — The number of the CA repetition runs in further discussed in Sec. 4.3.1. The HUTSIM runs also deviate from one another when different seed numbers are used, although the differences are much smaller than for CA runs. In this study only one HUTSIM reference run is always used reflecting the randomness met in real traffic situations.

3.7.6 Average velocities

In addition to queue length comparisons between the CA and HUTSIM simulators, another way to examine the behaviour of them is to measure average vehicle velocities on chosen routes of a case model. This kind of comparing was applied to all three test models described in Sec. 3.7.3. The raw data for velocity calculations in both simulators are gathered for all vehicles, but they are processed and reported only for vehicles on some chosen routes. Results for different model cases can be found in Secs. 4.3.3, 4.4.2, and 4.5.1.

The HUTSIM program outputs vehicle exit data necessary for these comparisons (the arrival time, the exit time, and the route length) to the *delayfile* when this output file type is requested. Again, the HUTSIM program is run twice for the two- and five-intersection cases (producing both *detsigfile* and *delayfile* outputs). In the freeway case, the HUTSIM simulation is not needed because the average velocity data are already available.

The CA program preliminary data collection is requested for the velocity calculations with the parameter *vehleaveon* (=1) in *generalpar*, causing vehicle exit information to be written to the file *vehleave* during the run (see file description in 'Maps for output files from CA' in Appendix A). The route lengths are not stored, and they have to be given separately at the comparing phase.

When the runs for the HUTSIM and CA simulators are done, the comparisons for velocities are performed with the MVCMP program (see Sec. 3.5.2) using the *delayfile* and the *vehleave* as input. In addition to these two files, the program needs input data about the selected routes, the length of the examined time periods during which the averages are taken, and the route lengths (on the CA side). In this study the time period was usually chosen to be 5 minutes. — The MVCMP program produces an output list reporting the average velocity for vehicles that exit the model within the time period in question for both simulators. The output list can then be used as an input for visualisation of comparisons (examples of figures in Secs. 4.3.3 and 4.4.2 and in Appendix B).

For assessing the influence of randomness, the CA runs were performed ten times using different seed numbers, and comparisons between separate CA runs and between the CA and HUTSIM runs are made as in the queue length case.

3.7.7 Traffic flow

Traffic flow measurements were carried out for the two- and five-intersection cases for both simulators. Evaluating the flow in different parts of the CA model and comparing it to the values of the HUTSIM model gives information about how well the route prediction (turning rate assessment) works and how the possible jams dissolve at intersections on the CA side. — For small networks as in this study, the route prediction can be expected to succeed well in any case because the detectors have to be placed at locations where vehicles have probably passed only one intersection since their arrival.

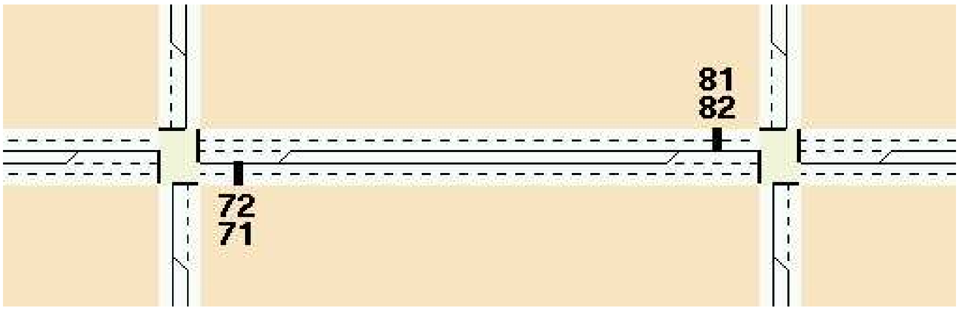


Figure 3.26: Placement of the flow detectors in the 2-intersection case.

The flow data for the HUTSIM runs are again collected from the *delayfile*, in which a record is written whenever a vehicle passes an output detector during the run.

The detector data storing in the CA simulation is activated with the parameter *cadetsigon* (=1) in the parameter file *generalpar*. The data are written to the file *cadetsig* during the run. The record format for this file is described in the part 'Map for output files from CA' in Appendix A.

The comparison list for the HUTSIM and CA detector passing events is created by the program DCMP with the files *delayfile* and *cadetsig* as input (the program description in Sec. 3.5.3). A flow value in the list is given as the number of vehicles passing a detector in a time interval corresponding to one signal period length.

The detectors are placed in the two-intersection case between the intersections on each lane, see Fig. 3.26.

In the five-intersection case, the flow detectors are placed in the upper right region of the model area on four lanes, see Fig. 3.27.

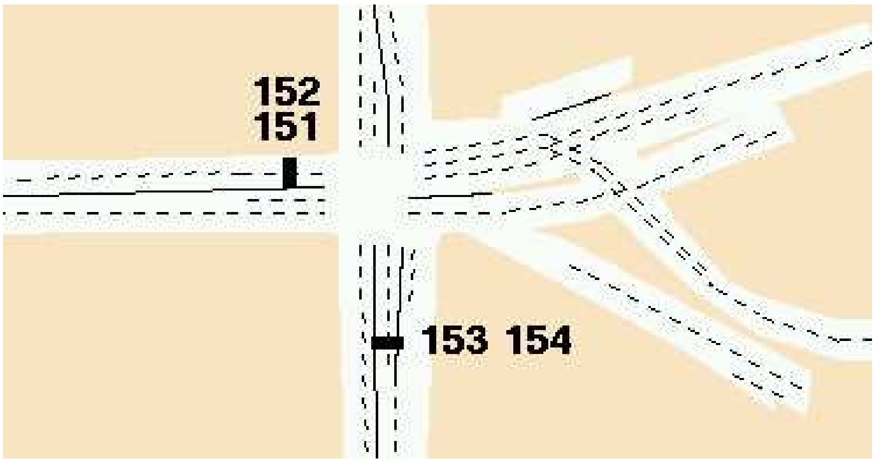


Figure 3.27: Placement of the flow detectors in the upper right region of the 5-intersection area.

For the two-intersection case, ten CA simulations were made against one HUTSIM run with different seed numbers using the turn rate set 3. Ten CA runs were also performed for the five-intersection case along with one run for HUTSIM.

The results of the flow measurements for both test areas are reported in Secs. 4.3.4 and 4.4.3.

Chapter 4

Simulation Results

4.1 Calibration

The CA model calibration does not seem to have been an issue of high importance formerly, but especially the braking parameters have received differing values. As the models have recently become more sophisticated, more weight has also been given to the calibration process. When this work was started, it seemed obvious that most of the runs will be related to urban intersection traffic. For this reason the calibration is also closely connected to vehicle movement in urban intersections.

Since the vehicle velocity in the CA model corresponds to some number of cells per time step, the velocity in real space depends on the cell and time step lengths. The time step length was chosen to be one second long because this value is normally used in connection with traffic CA models in question. The cell length in basic CA models is often 7.5 metres, but in this study it was chosen to be 5 metres. This value was decided on for comparisons to normal passenger car traffic in HUTSIM to be more realistic. (The vehicle length for passenger cars in HUTSIM queues is 5.07 metres on the average.)

The VDR model contains two parameters that influence vehicle movement: p_0 (braking probability for the present time step if the velocity was zero on the previous time step) and p (normal braking probability). In free flow a vehicle maintains the maximum velocity with the exception of decreasing it with one velocity step in case of randomisation depending on the value of the parameter p. So, the first thing after deciding the cell length is to try to adjust the free flow velocity as close to the velocity of HUTSIM vehicles as possible with this parameter. The target velocity of HUTSIM vehicles was defined to be 50 km/h. In CA simulations this would correspond to using the p value of 0.22 with the chosen cell and time step lengths. This value was, however, quite high causing slow dissolving of queues, so a smaller value for p was to be searched even if it was to lead to an average velocity higher than 50 km/h.

From literature references it could be noticed that with the VDR model the value for p is often less that 0.1. Making the value very low would have raised the average velocity of vehicles, so it was decided to use the value 0.100 for p. The chosen value makes the average maximum velocity of vehicles in free motion with 5 metre cell length, 1 second time step, and at maximum velocity of 3 (cells/time step) to be 52.2 km/h, which is yet quite close to the maximum velocity of HUTSIM vehicles.

4.1.1 Vehicle movement over stop line

After fixing the value of the braking probability, p, the only other parameter for vehicle movement with the VDR model is p_0, the braking probability, for those vehicles that are at rest (v=0) as a consequence of the previous updating step. The nature of the VDR model requires the value of this probability to be larger than the value for vehicles in movement.

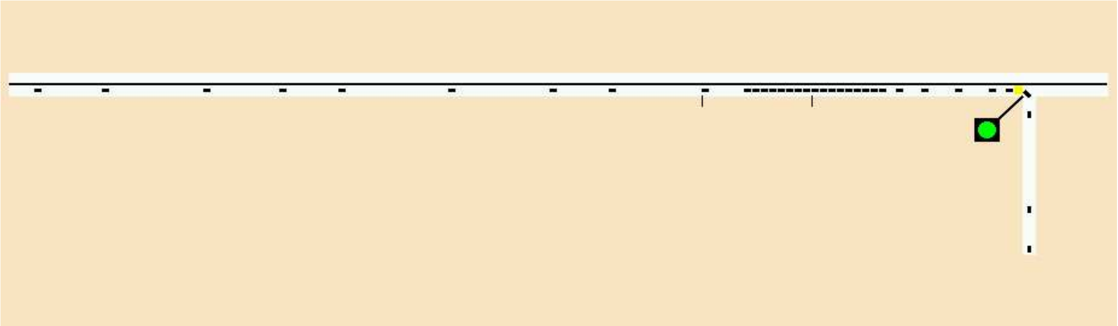


Figure 4.1: Test arrangement in vehicles crossing the stop line for CA.

The parameter p_0 determines how quickly a standing vehicle gets into motion, and accordingly, the pace a standing queue dissolves with. Therefore it can be adjusted, for instance, by measuring the velocity of queue dissolving in real traffic [36]. Since in this study comparisons were to be made to HUTSIM traffic and intersection traffic was mostly to be considered, it was decided to make the calibration of p_0 with the vehicle movement in a queue across the stop line at a traffic signal post after the signal turns from red to green. The movement was compared to HUTSIM vehicles' movement in similar circumstances.

The test arrangement for this comparison for both HUTSIM and CA vehicles was a one-lane road ending with a signal post, after which a straight going and right turning lanes follow (picture of the CA model in Fig. 4.1). The measurement tracks how many seconds have passed since some number of vehicles have crossed the stop line.

HUTSIM vehicles move in a very steady manner compared to CA vehicles, therefore

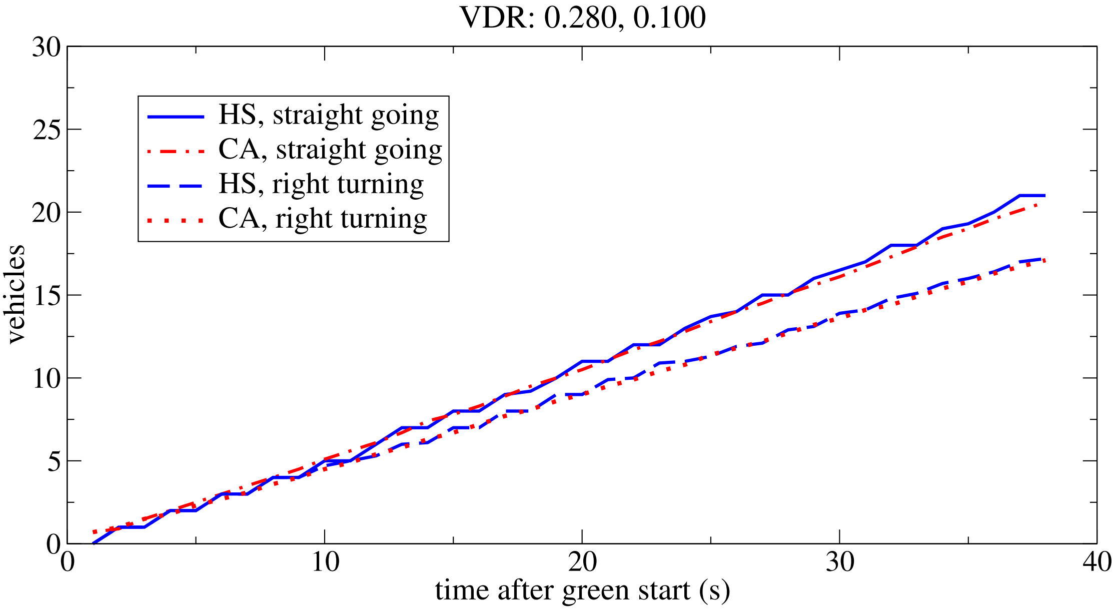


Figure 4.2: Comparison between vehicles in HUTSIM and CA crossing the signal stop line after the signal turns from red to green.

the HUTSIM results were calculated as an average of only 15 separate dissolving sequences (both straight and right turning cases) up to queue lengths of 38 vehicles.

In the CA measurements, the p_0 value was varied and the p value was kept at the rate mentioned earlier. The CA runs were carried out for a larger number of sequences for better statistics (one hundred runs with each p_0 value for both straight going and right turning cases). The results were then compared to the HUTSIM vehicle results, and the value $p_0 = 0.280$ that gave the best fitting was chosen to be used in later simulations.

The comparison curves for both simulators with the chosen CA braking probability values are shown in Fig. 4.2.

These values were used in all the VDR runs except for the freeway runs that involved higher maximum velocity (6 cells/time step) and a large input from an on-ramp. With the freeway runs, expecially the p_0 value had to be changed significantly to get a better throughput on the freeway. The parameter value choices are further explained in Sec. 4.5.1.

The test model parameter values were chosen without proper calibration to be close to the VDR freeway model values, see Sec. 4.5.1. The p value is the same as with the freeway VDR case and the p_0 value is a bit smaller (0.130) than for the freeway VDR case.

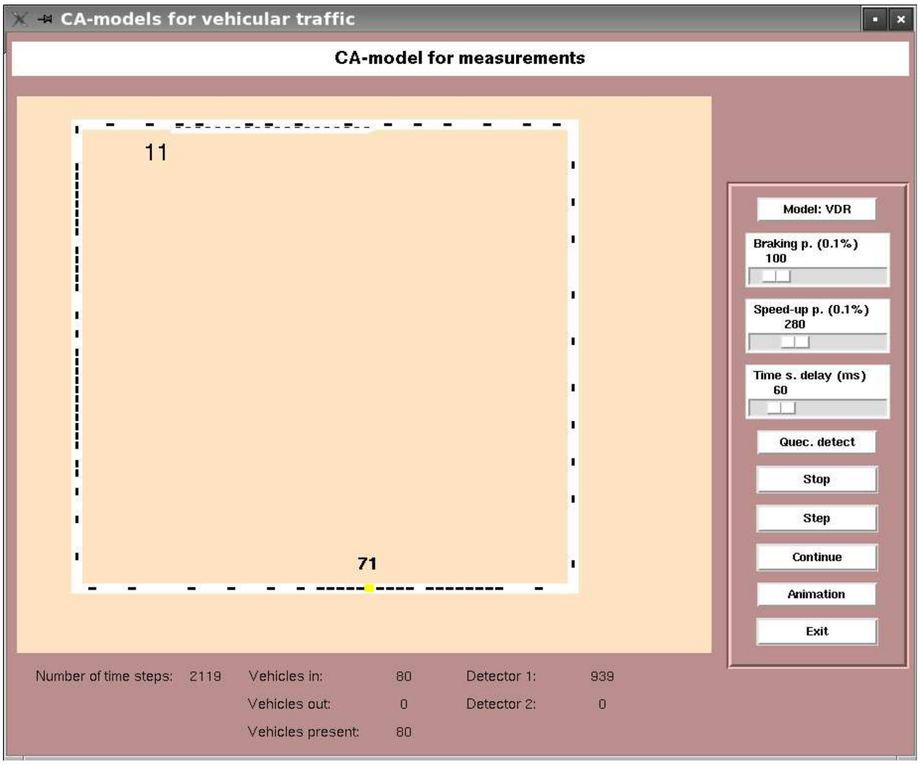
The only NaSch model runs are made in Sec. 4.2 where the selection of the very small p value (0.050) is explained.

4.1.2 Queue front line velocity

For the chosen normal VDR parameter values p and p_0, the velocity of the moving of a queue downstream front line when a queue dissolves was also checked. This was measured from the simulator runs on screen as an average of 20 sequences by checking the number of time steps (seconds) it took for the queue front line to move to the upstream direction over a road stretch of one kilometre. The value obtained was –13.1 km/h, which comes quite close to the value (≈ –15 km/h for a highway environment) in [36]. However, at traffic signals the velocity of the front line has been estimated to be ≈ –20 km/h [14].

4.1.3 The fundamental diagram

The fundamental diagram for different CA models has been extensively studied both theoretically and experimentally, see for example [49, 66, 3, 35]. It has been found that these models reproduce the empirically discovered shapes of the diagram quite well.



Braking p. (0.1%) 100

Speed-up p. (0.1%) 280

Time s. delay (ms) 60

```
Number of time steps:  2119   Vehicles in:        80    Detector 1:    939
                              Vehicles out:        0    Detector 2:      0
                              Vehicles present:   80
```

Figure 4.3: Test arrangement for the fundamental diagram measurements.

For this study the fundamental diagram was measured for two models used in the simulations: the VDR model with maximum velocity 3 (cells/time step) and the test model with maximum velocity 6 (cells/time step). Fig. 4.3 depicts the measurement arrangement: a closed one-lane road of square shape. The length of the road is one kilometre, and the corner areas do not change the vehicle dynamics. The vehicles are fed to the road from one on-ramp (from the origin 11 in the figure).

The vehicle movements are recorded at one detector (detector code 71 in the figure): the time of passing and the velocity of the vehicle. The density on the road takes values from 10 veh/200 cells (= 10 veh/kilometre) to the value of 190 veh/200 cells with the increments of 10 vehicles.

The flow and the average of velocities are measured versus density by taking five five-minute samples after all the vehicles for the respective density are fed into the system with a time sequence of at least 900 time steps to normalise the flow. The start of each sample is 1000 time steps since the previous one.

The flow versus density for the VDR model with maximum velocity of 3 cells/time step is seen in Fig. 4.4.

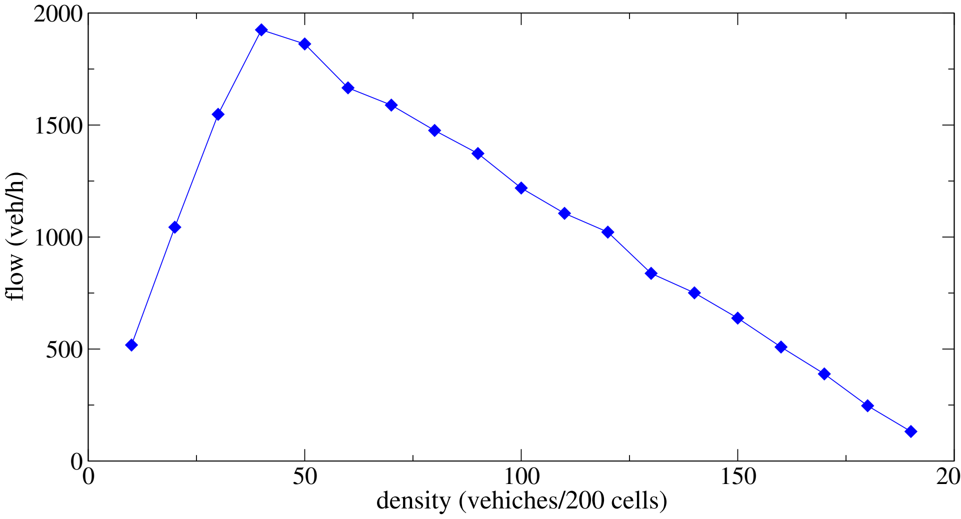


Figure 4.4: Flow versus density for the VDR model (VMAX=3).

The diagram starts with a linear part where the flow value grows quickly with the increasing density. The maximum flow value is reached with the density value of 40 veh/200 cells, and beyond that the flow starts to decrease quite slowly. The decreasing part is also fairly linear since the flow value is an average of several five-minute periods. If each five-minute (or shorter) period was marked separately in the figure, there would be large fluctuations in the flow values with the growing densities because moving jams affect the flow at the detector position.

The shape of the diagram is typical as compared to measurements of other CA models and also to measurements of real traffic.

The velocity diagram in Fig. 4.5 depicts the average of the velocity values for separate vehicles in the five samples of each density value. The velocity should drop monotonically with the increasing density, but some fluctuation is seen in the diagram. The velocity does not drop below 1 (cell/time step) because of the discrete nature of the model.

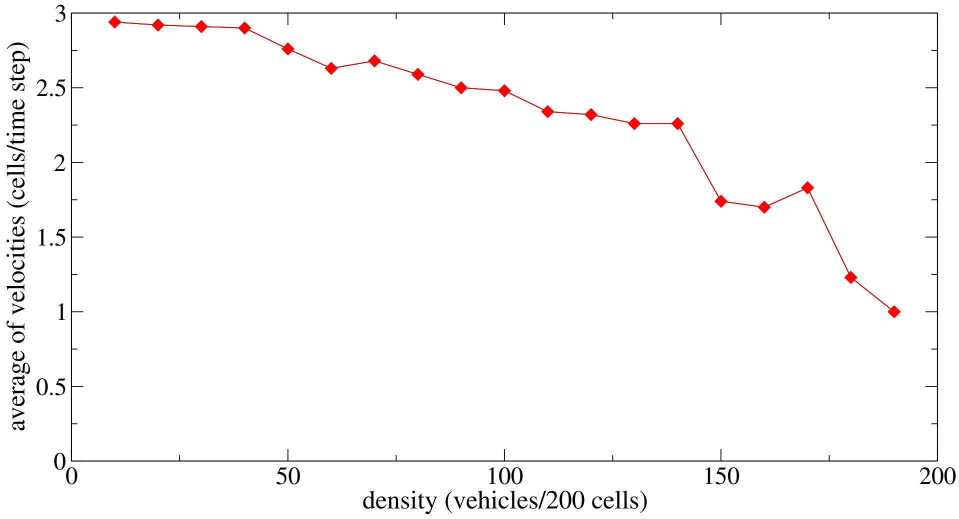


Figure 4.5: The average of velocities versus density for the VDR model (VMAX=3).

The respective figures for the test model with the velocity maximum of 6 cells/time step are seen in Figs. B.67–B.68 in Appendix C. The diagrams for the test model are quite similar to the ones for the VDR model, only this time the velocity curve lacks large fluctuations.

4.2 Generic feature simulations

The goal for generic feature runs in this study is to search for any regularities in queue building appearing in CA modelling of vehicular traffic. The tests were decided to be run on a long, one-lane stretch of road with a traffic signal head at the end. The queue building is examined statistically, which requires numerous measurements of similar queueing situations. A quick simulation model like CA is specially suitable for this kind of studies since the number of separate tests is large. The test arrangements are discussed in Secs. 3.7.3 and 3.7.4.

The runs were made using different CA models: NaSch, VDR, and the test model of this study. The velocity maximum values were 3 and 5 cells/time step. As the influence of flow was also to be examined, the runs used several flow values starting from 100 veh/h up to as high values as possible. The upper limit for the flow was reached at different rates when the origin point of the model could not supply higher flow values because of heavy congestion. This congestion at the origin area was formed by backwards flowing stop-and-go waves on the road. Another reason for the upper flow limit could be the queue length (area) reaching the measurement pipe length. The runs for general features are described in Table B.11 and the figures are in Figs. B.73–B.84 all in Appendix B. The last two runs in each model case may have the flow difference also other than the normally used 200 veh/h. A snapshot of one simulation situation is seen in Fig. 4.6.

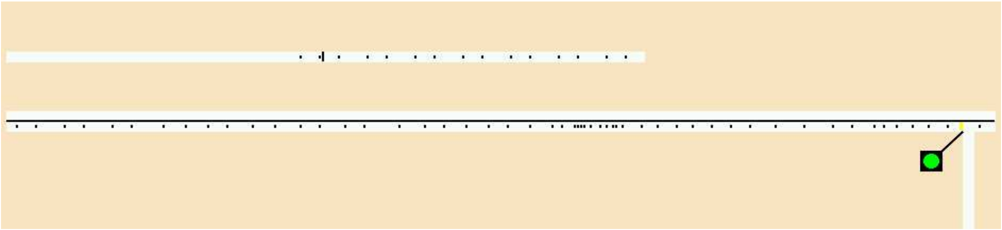


Figure 4.6: An example of the general feature simulations (model NaSch, flow=2500 veh/h).

The runs for generic features comprise measuring the length of the influence area of queue building before a traffic signal head. The queue starts to build when the signal turns to red, and it usually continues to build at the upstream side although the front area starts to dissolve with the signal turning to green — the queue starts to move. The measurement counts all the vehicles that come to a stop before the queue completely dissolves.

Simulation runs with the VDR model use maximum velocities 3 and 5 (cells/time step). The model parameters are the same as those used with the intersection and freeway runs (the braking probabilities described in Sec. 4.1). The used flow values range from 100 with increments of 200 up to 1700 (veh/h). Beyond 1700 (veh/h) the origin area easily suffered from overflow, and the last value measured was 1800 (veh/h).

The graphs for different flow values can be seen in Fig. B.73 in Appendix B for the velocity maximum 3 (cells/time step). Three of these graphs are also seen in Fig. 4.7.

The figures show quite similar behaviour for all flow values. The average influence area length grows when the flow gets higher and also the range of the attained lengths becomes wider. The average length values usually grow faster than evenly with increasing flow until the highest flow values (the separation between the last graph and the one nearest to it is minor because the flow difference for them is smaller than normally). Drawn on a logarithmic scale, most of the separate graph groups suggest power law or exponentially decreasing behaviour if the highest probability values in each group are counted out. The slope of the black line in the Fig. B.73 drawn separately as a reference line corresponds to the value −14.57. The slope does not change much with the flow value.

The runs for the VDR model also include a group of measurements with a lowered red signal phase time. The signal length was changed from 100 to 50 seconds and

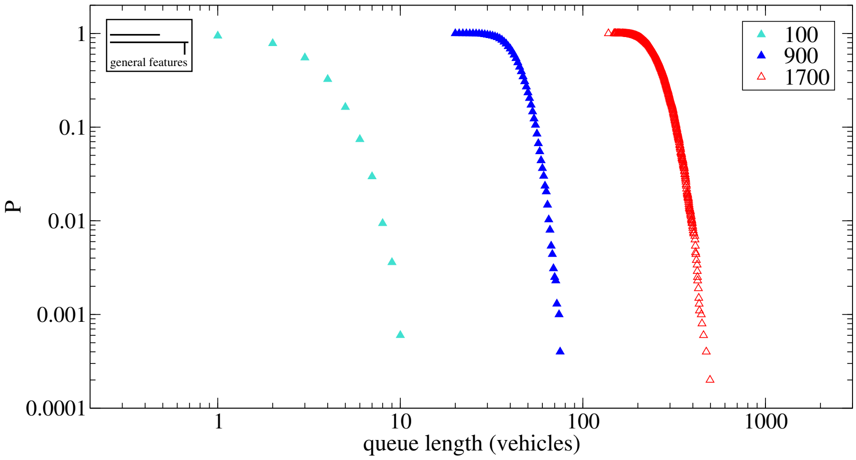


Figure 4.7: Probability for a queue length to exceed a specified value (VDR, VMAX=3, flow= 100, 900, and 1700 veh/h).

the green time was enhanced by the same amount to keep the overall period length equivalent to the previous runs. The velocity maximum is again 3 (cells/time step) and the simulation time is the same as before. The flow maximum could in this case be raised to the value of 1850 veh/h. The results for these runs are seen in Fig. B.74. The graph shapes are quite similar to each other with about even intervals (but on the logarithmic scale). The slope values are again practically equal among the graphs. As compared to the slope value for the runs with equal velocity maximum but longer red phase, the slope of the reference line is now smaller than before: –10.61 (corresponding to some flow values below 1850 veh/h). The reason for this can be expected to be the starting of the queue lengths from lower values (shorter red time) than in the previous case yet reaching quite high queue lengths.

The VDR model was also run with the maximum velocity value of 5 (cells/time step). Results for these runs are seen in Fig. B.75. The figure shows that the flow value could be raised a bit and the last measured value was 2000 veh/h. Higher flows were attainable because raising the maximum velocity induces lower density on the road (as compared to the density at similar flow but smaller maximum velocity) meaning less congestion at the origin area. The graphs for this velocity category look very much the same as the former ones, and slope for the reference line is now –18.79 (corresponding to the high flow values except for the last one).

The queue length influence area was also measured with the NaSch model. This model has not been used otherwise in this study, and the reason to be used here was to pursue higher flows than can be reached with the VDR model with the chosen parameter values. The NaSch model uses only one braking probability percentage and does not have slow-to-start behaviour, which causes less stop-and-go traffic at least when the flow values are low. The braking probability value was chosen deliberately low, as 0.050, although the often used values for the model lie between 0.100 and 0.500. By selecting a low braking percentage, the flow could be raised over 2000 veh/h. The maximum velocity was set to 5 cells/time step.

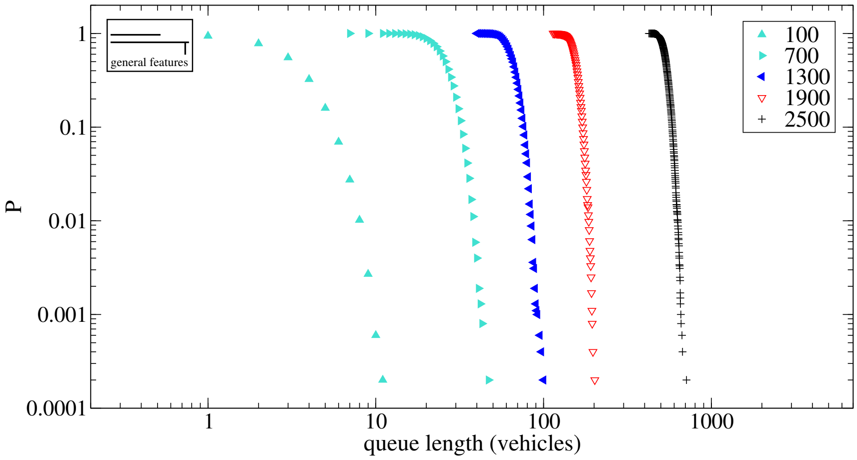


Figure 4.8: Probability for a queue length to exceed a specified value (NaSch, VMAX=5, flow= 100, 700, 1300, 1900, and 2500 veh/h).

The graphs for the NaSch model runs can be seen in Fig. B.76, and a selection of them are also depicted in Fig. 4.8. The largest flow value measured was as high as 2500 veh/h. These graphs are again quite similar to the ones in the cases studied above. The slope values are close to each other among the curves for different flow values, but the slope is now steeper than before gaining the value –32.44 (for the reference line in Fig. B.76). This can be explained with the absence of the slow-to-start behaviour causing the queues to dissolve more easily than in the earlier cases.

The general feature runs were also performed for the test model that is used in the freeway simulations (see description of the model in Sec. 3.7.3). The results could not be expected to be much different from the VDR case since the test model uses similar slow-to-start behaviour as the VDR does. Furthermore, its anticipation qualities for driving in dense traffic and queue building probably do not matter much in the present queueing circumstances (but the road may be more occupied upstream the queue area). — The model parameters used are the same as with the freeway runs (see Sec. 4.5) with the braking percentages as $p_0 = 0.130$ and $p = 0.120$. The velocity maximum of 5 (cells/time step) is used.

The test model graphs are depicted in Fig. B.77. The graphs for this model look quite the same as those for the VDR model using the same maximum velocity. As high length values as with the VDR runs are not reached now, and the slope for the graphs is somewhat larger; the slope of the reference line reaches the value –24.55, which means earlier dissolving of the formed queues. This can be understood by the lower p_0 value as compared to the corresponding VDR parameter.

To resolve whether the shapes of the graphs follow an exponential or a power law curve, some of the separate graphs of the VDR model case (with the longer red duration) and the NaSch model case are compared to calculated curves. For the exponentially decreasing case, the curve is of the form

$$P(\tau) = Ae^{-\alpha\tau}, \tag{4.1}$$

where $P(\tau)$ is the probability of finding a queue with the length τ, A and α are parameters. For the power law case, the curve is of the form

$$P(\tau) = B\tau^{-\beta}, \tag{4.2}$$

where B and β are parameters.

In Figs. B.78–B.79 are examples of the exponential fit and power law reference curves for some of the VDR model runs. The exponential fits are made with the XMGRACE program with weights on the higher queue lengths. The exponent values ($= \alpha$) for the three cases (flows 300, 1100, and 1800 veh/h) are 0.48, 0.14, and 0.02, respectively. The power law reference curves in Fig. B.79 are drawn with the exponent value ($= \beta$) of 14.57.

From the figures it can be seen that the exponential fits are better than the power law references. Also, fitting the power law in this case is somewhat unreliable since the range of the queue length values is quite low for separate curves.

Comparison between the exponential and power law fitting was also made for the NaSch model case. Figs. B.80–B.81 show curves for the exponential fits and power law references. The exponents ($= \alpha$) for the three flow values (300, 1300, and 2300 veh/h) are 0.60, 0.20, and 0.07, respectively. The power law reference exponent ($= \beta$) in Fig. B.81 is 32.44. The exponential fits seem to be better also in the NaSch model case.

The queue length dependence of the flow value at P(50%) is plotted for the VDR and NaSch model runs in Figs. B.82–B.83. The figures show that in both cases the queue length P(50%) values rise quite sharply after a steady rise at lower flow values. The sharp rise implies an anticipated phase transition in traffic from free flow to some other state. Fig. B.84 shows the queue length dependence at P(50%) of the flow rate on logarithmic scale for different rules, the slope of the reference line in the figure is 1.04.

4.3 Two-intersection area simulations

The CA version of the two-intersection test area can be seen in Fig. 3.17; the HUT-SIM version is in Fig. 3.12. Some tests were also run with an intersection version that did not contain separate left turning lanes. In these tests the results were pretty poor in cases where the turning percentages off the main route direction were larger than zero because the vehicles waiting to turn left blocked the way for others. The results of these runs are not presented here.

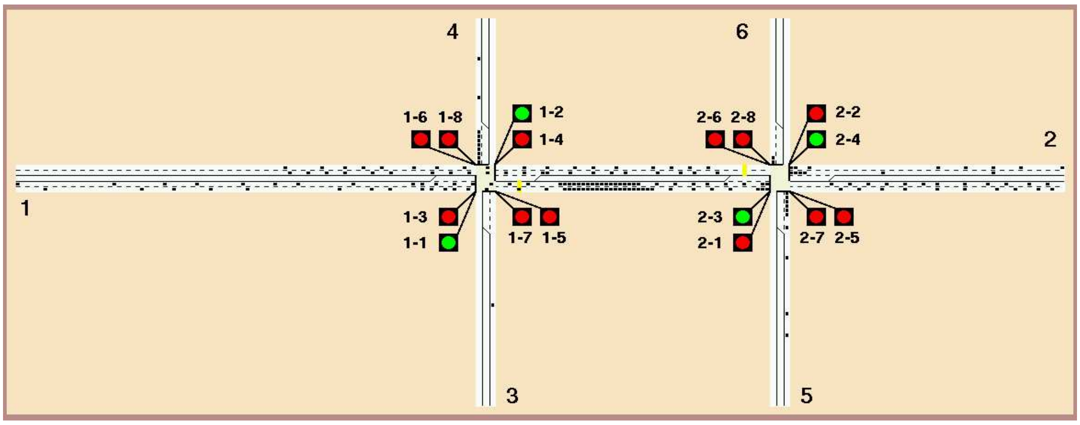


Figure 4.9: An example of the 2-intersection simulation situations.

Fig. 4.9 shows a snapshot of the two-intersection simulations. The tests for this intersection type contain runs for queue lengths measured at each signal post, runs for average velocities on different routes through the area, and some runs for traffic flow values measured at a few detectors on the area between the two intersections. Below are the results for these three test categories.

4.3.1 The repetition of the CA runs

Since randomness greatly affects the results of one separate CA simulation run, several repeated runs of a simulation event using different seed numbers give a more reliable picture of the model behaviour. In most of the CA simulation environments of this study, the number of the repetition runs was chosen — from general experience — to be ten.

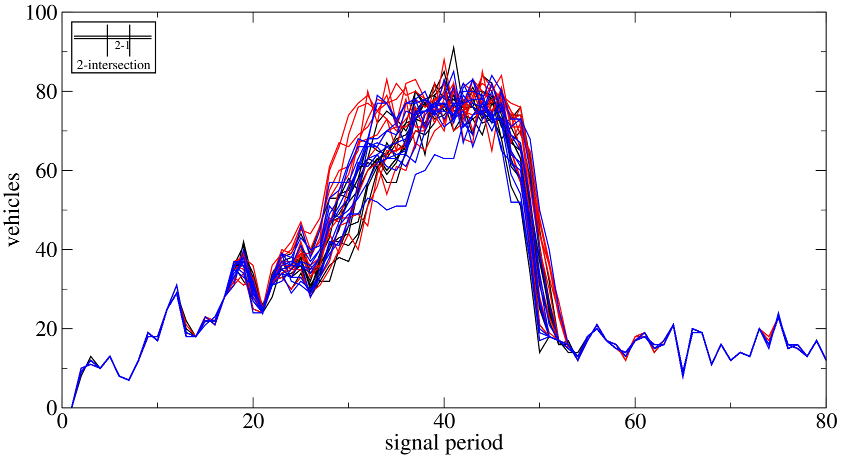


Figure 4.10: The results of three sets of ten CA runs for queue lengths at signal post 2-1, turn rate set 1.

Some cases for the two-intersection area runs were performed for additional twenty runs using different seed numbers. Fig. 4.10 shows the results of the queue length runs (with the turn rate set 1) at signal post 2-1 of the same ten runs (see Sec. 4.3.2) as in Fig. B.19 (in Appendix B) plus two sets of ten additional runs.

As compared to Fig. B.19, it can be seen that randomness somewhat enhances uncertainty in the area of the signal periods of about 30–45. For other signal period areas, already the ten first runs give practically the same results as all the thirty runs together.

In Fig. 4.11 are depicted the average CA queue length graphs of the three ten run sets with the HUTSIM result as a reference (the first ten run set is the same as in Fig. B.23). All the average graphs differ from the HUTSIM graph especially around the signal period 40, but they are otherwise quite close to it and to each other.

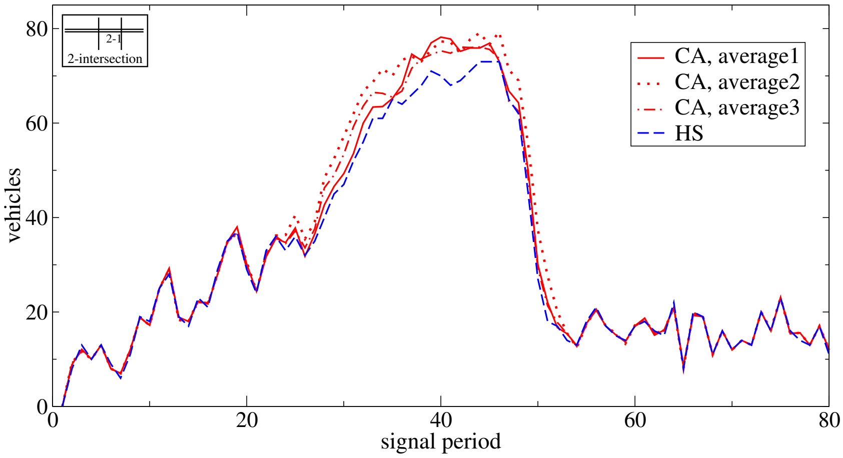


Figure 4.11: The average queue lengths of the three sets of ten CA runs at signal post 2-1, turn rate set 1.

Similar additional queue length runs were also performed for the signal post 1-6. The results for the three sets of ten runs are seen in Fig. B.1 in Appendix B. As compared to Fig. B.11 in Appendix B (containing the results of one of these sets), the variation between the curves gets somewhat larger in the signal period area where it already is significant in the first ten runs. The areas outside the signal period area of about 30–50 give quite similar results as Fig. B.11 does.

The average graphs for the three sets together with the HUTSIM result are seen in Fig. B.2 in Appendix B. The CA average graphs differ significantly from each other only in the signal period area of about 40–46.

Twenty additional simulations were also run for the average velocity measurements to see the influence of randomness on the results. Fig. 4.12 shows the results of the first ten CA runs of Fig. B.28 in Appendix B plus two additional sets of ten runs (with different seed numbers) for the route 11 → 2 (see Sec. 4.3.3).

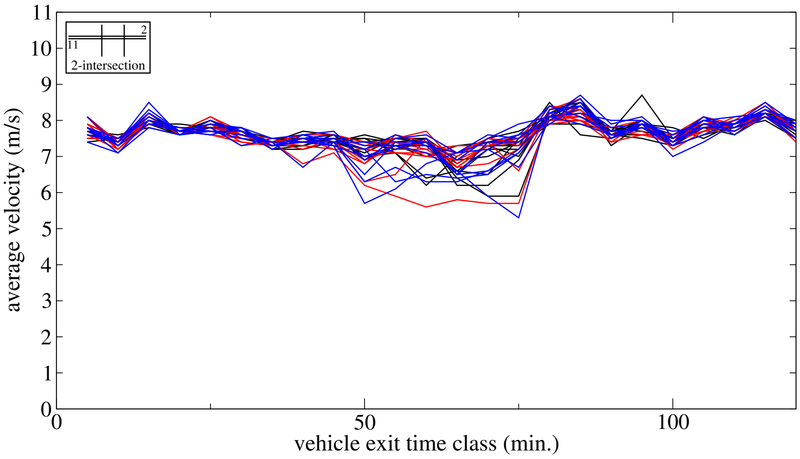


Figure 4.12: Average velocities for CA vehicles on route 11 → 2; three sets of ten runs, turn rate set 2.

The results for the thirty runs are very similar to the results of the first ten runs outside the signal period area of about 50–75, but the additional runs make differences between the graphs larger in the area between the signal periods mentioned.

The additional average velocity graphs were also measured and drawn for the route 31 → 2. The results of the three ten run sets are seen in Fig. 4.13 (the first ten are also seen in Fig. B.32 in Appendix B).

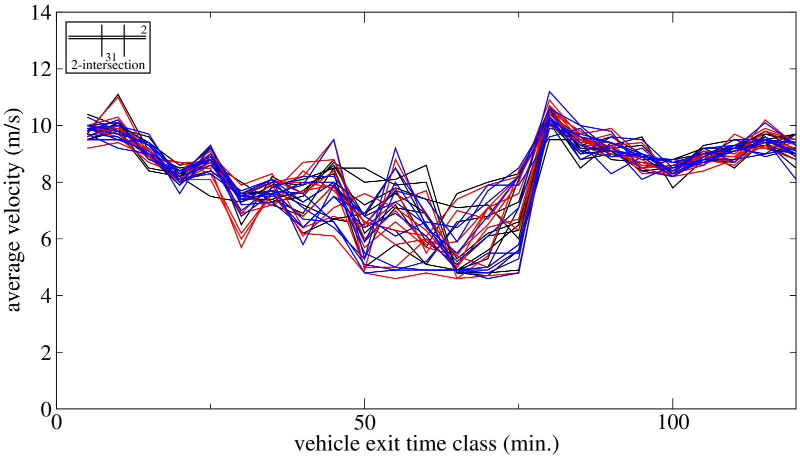


Figure 4.13: Average velocities for CA vehicles on route 31 → 2; three sets of ten runs, turn rate set 2.

In this case the additional runs do not much add the variation of the results.

From the above results it can be deduced that the choice of using ten CA runs for one simulation situation gives a quite reliable picture of the variation of the results caused by randomness.

4.3.2 Queue lengths

The queue lengths at different signal posts were measured on the CA side at the time the signal turns from red to green. On the HUTSIM side, the program gives the maximum queue lengths during a signal period, which amounts to about the same as the queue length values given by the CA runs. The measurements were carried out for every signal post marked as 1-1, 1-2,..., and 2-8 in Fig. 3.17. The signal codes and the turning directions associated with these signals are also shown in Fig. 4.14.

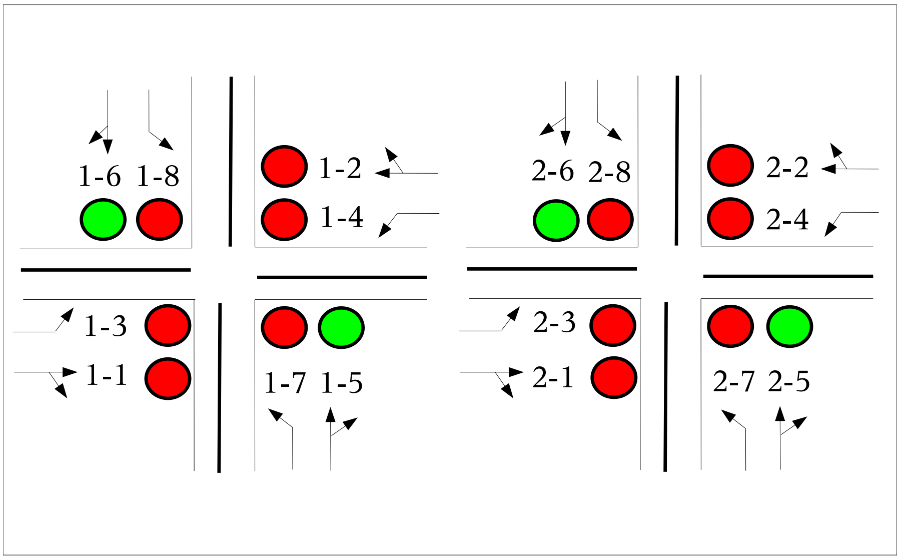


Figure 4.14: Traffic signal codes and the associated turning directions for the 2-intersection case.

The queue length values measured for both HUTSIM and CA runs are compared and presented as figures displaying the vehicle count as a function of the signal period number during the two hour simulation time (period length for the signals in this model is 90 seconds).

The QCMP program output list contains comparisons for all traffic signals of the model, but the figures presented here only cover comparisons for signal posts 1-1, 1-6, and 2-1. Reasons for choosing these posts include the purpose to test the behaviour of the CA model in a larger congestion generated at some route; the long lanes before signal 1-1 offer a good place for building jams with an appropriate input flow level. The signal 2-1 serves, for its part, as a suitable checking point for the traffic coming from 1-1. Also, the lane from origin 3 feeds most of its vehicles towards 2-1 enhancing the load on these lanes. Quite heavy congestion is also built at 2-2, but it does not qualitatively differ much from that at 1-1 and 2-1. The queues at 1-2 remain low because of the 'green wave' it forms with 2-2. The flows to the one-lane links leading to signals 1-5, 2-5, and 2-6 are steady and quite low whereas the flow from origin 4 is varied leading to heavier congestion for some time at signal 1-6. This is why 1-6 was chosen to represent these one-lane stretches. — Traffic on the left turning lanes was in most cases quite insignificant and oscillating (depending on the turning rates), and queue comparisons at these signals are not included here.

The CA runs were performed ten times towards one HUTSIM run using different seed numbers to see how much randomness affects the results. In Figs. B.3–B.26 in Appendix B are depicted the graphs for each signal post (1-1, 1-6, 2-1) for every four turn rate sets (1 – 4). For each signal post, there are firstly four figures displaying joint graphs for different seed numbers (one figure for each turn rate set). After that there are four figures where the average queue lengths of the associated ten CA runs are compared to the sole HUTSIM run, one figure for each turn rate set. Table B.1 in Appendix B gives file and parameter information concerning the HUTSIM and CA runs in question and Table B.2 connects the figures to these runs.

At lower flow values, the queue lengths given by the CA model generally match quite well the lengths given by the HUTSIM model. (One separate example is shown in Fig. 4.15 for signal post 1-6). Even in areas where the curves do not completely match, their shapes are similar. However, comparison between the CA runs using separate seed numbers reveals that large differences may exits. Already with the turn rate set 1, the results may differ considerably when jams are built up. Outside congestion areas the CA graphs are often almost congruent.

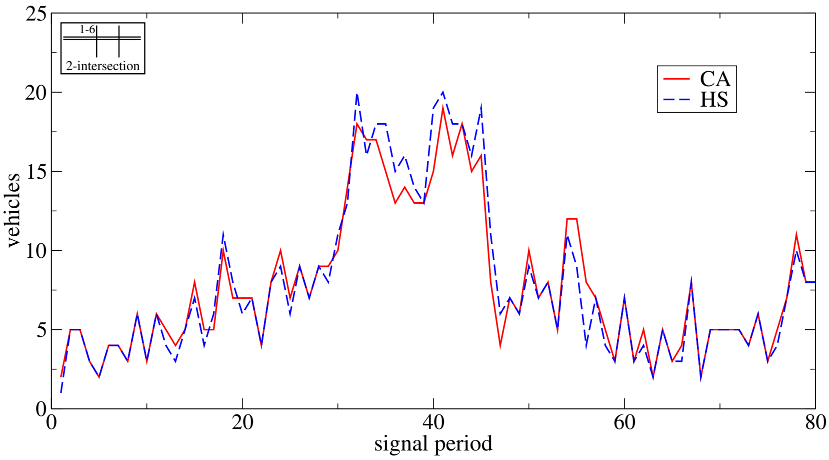


Figure 4.15: Queue lengths at signal head 1-6 (straight going and right turning). Turn rate set 1, CA run 1.

Usually the most significant reason for deviations is randomness — either in queue dissolving or in determining the turning direction for individual vehicles. In the former case of these two, the fluctuations in queue dissolving may pile up and cause temporarily extra long CA queues. These queues do not dissolve as quickly as the corresponding HUTSIM queues because more vehicles (than in the HUTSIM queues) have to wait for the next signal period to leave the intersection.

The longer the queue is (or the sum of the queues in the two-lane cases) at the end of the signal period the more probable it is that there are fluctuations in CA queue lengths. This can been seen in the two-lane cases in Figs. B.3–B.6 and B.19–B.22: the differences especially for queue lengths over 30 vehicles are severe. In the one-lane cases, the fluctuations start at much lower queue lengths, even at the lengths of 10 vehicles (signal post 1-6 in Figs. B.11–B.14).

Fig. 4.16 shows one separate example of the series in Fig. B.19 where even long queue lengths match quite well. The CA queue lengths often tend to longer values than the corresponding HUTSIM queue lengths, because the average velocity for the CA vehicles is a bit larger and the VDR model does not contain any mechanism for braking before the queue end is reached.

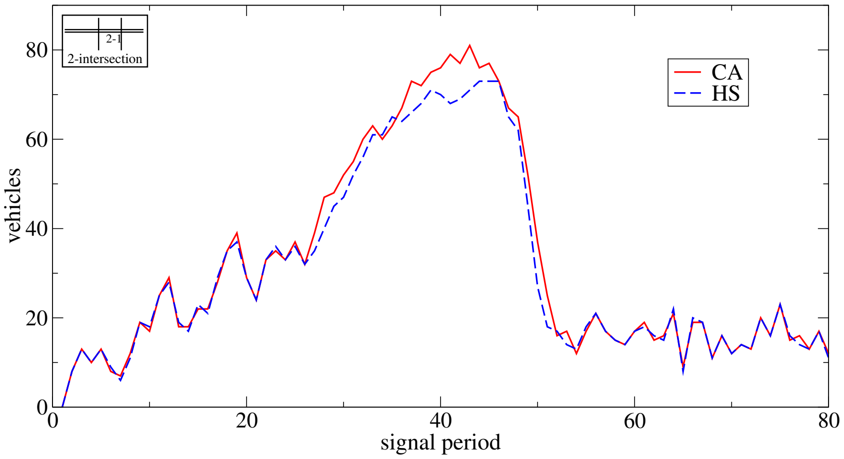


Figure 4.16: Queue lengths at 2-1 (straight going and right turning). Turn rate set 1, CA run 2.

Differences in queue lengths between the CA runs with different seed numbers are larger when the turn rate set number increases, i.e., when the turning rates at intersections to separate directions (off the main flow direction) become larger. This can be seen in all the comparison sets, Figs. B.3–B.6, B.11–B.14, and B.19–B.22. Differences already occur when the queue lengths are small.

For each of the three signal posts, the average of ten CA runs is compared to the HUTSIM run in Figs. B.7–B.10, B.15–B.18, and B.23–B.26, respectively. The average curves match (with a few exceptions) well at least at shorter queue lengths — also when the turn rates get larger. Especially the curve shapes are usually similar.

A few strange cases in the above-mentioned curves can be noticed, however. In Fig. B.7 the HUTSIM side builds up a strong peak starting at about the period number 38 whereas the average of the CA runs remains on a much lower level.

(One separate example of the ten CA runs is shown in Fig. 4.17.) Reasons for this difference are the high flow rate of vehicles still after the first peak at about period 30 and the amount of vehicles that pile up between the intersections (at signal post 2-1); the two lanes between the intersections are practically filled with vehicles on the CA side. In this kind of a situation, the HUTSIM vehicles reduce their velocity already before post 1-1 and build up queue before it. The CA vehicles, on the contrary, cross the first intersection and form a more dense queue, and less extra vehicles are left in the area before post 1-1 than in the HUTSIM case. Respectively, the queue lengths for the CA vehicles at post 2-1 are on the average much larger than for the HUTSIM vehicles around the signal period number 40 as can be seen in Fig. B.23.

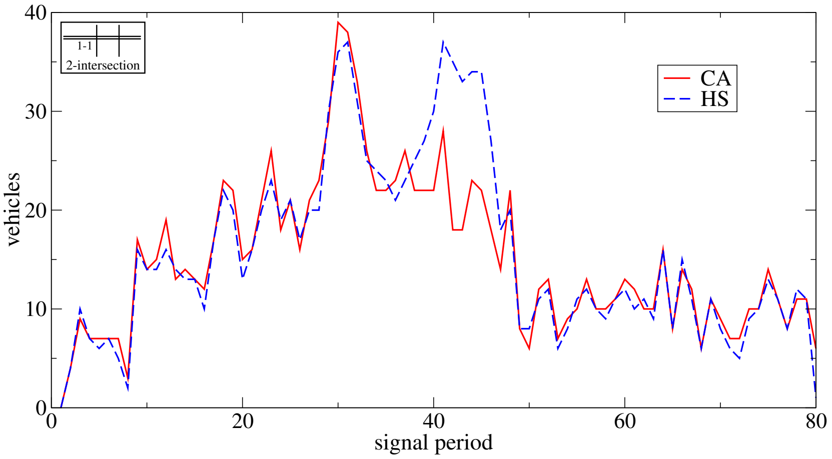


Figure 4.17: Queue lengths at 1-1 (straight going and right turning). Turn rate set 1, CA run 9.

Another large difference (for the turn rate set 3) can be seen in Fig. B.25 for post 2-1, where the CA runs produce much shorter queues than the HUTSIM side starting at about the signal period number 40. One of the individual cases of the ten runs is seen in Fig. 4.18.

The reason for the difference in these runs seems to be the quite high turning rates combined with the high flow rate of vehicles. The queue length at post 2-1 depends on the flow from origin 1, on the number of vehicles turning left or right at the first intersection from origin 1, on the number of vehicles coming from origins 3 and 4, and on the number of vehicles turning left for destination 6 at this intersection. While the contribution from the generators at origins 3 and 4 is low and quite stable, the main effect comes from statistical deviations in the number of vehicles leaving for destinations 3, 4, and 6. In the HUTSIM run, the number of vehicles for destinations 3, 4, and (especially) 6 was low during a few signal periods before the peak queue length value (building a queue at 2-1) whereas the number of CA vehicles leaving for those destinations was higher (checked for a few runs). When a long queue in the HUTSIM run has thereby been developed, it takes several signal

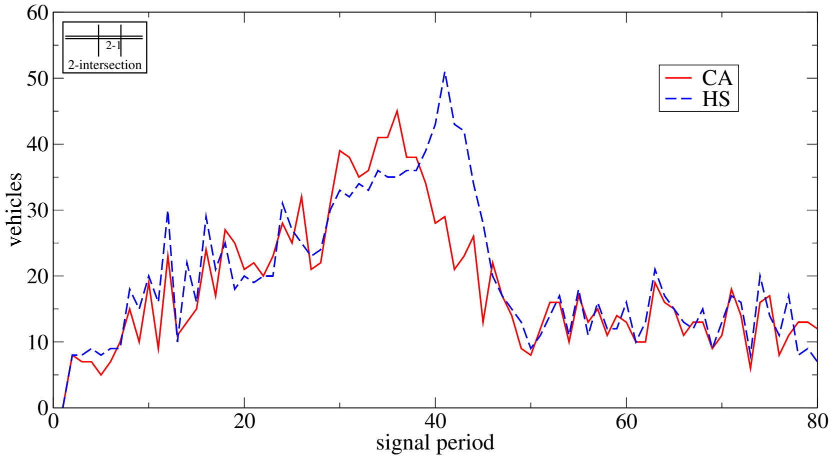


Figure 4.18: Queue lengths at 2-1 (straight going and right turning). Turn rate set 3, CA run 30.

periods before it gets dissolved. — A couple of the CA runs also build high peaks around period 40 as seen in Fig. B.21, but most of the values drag the average to a low level.

As regards the other signals on the main lanes (1-2, 1-5, 2-2, 2-5, and 2-6), the queue length variation in them between different CA runs is usually low for turn rate set 1 and outside congestion areas. When the turn rate set number grows, differences remain low for the signal 2-2 with two lanes (but are large at 1-2 because of 'green wave' ending) and for 1-5 with one lane because of low flow rate. In the other one-lane cases, 2-5 and 2-6, the differences between these runs are larger than for 1-5 probably because of higher flow rates. In times of congestion, variation between the CA runs is usually large at all posts.

The CA average values as compared to the HUTSIM values are usually good for 1-5 and 2-2 with all turn rate sets, but differences grow at congestion times for 2-5 and 2-6 when the turn rate set number is larger than 1 (the CA values usually stay on a quite low level).

4.3.3 Average velocities

As a second way of comparing the functioning of the HUTSIM and CA simulators, the measuring of average velocities on different routes in the two-intersection model was tested. The recording procedure of the program covers all vehicles, consequently allowing the examination of all routes afterwards, but only three of the more interesting ones are discussed here. The routes chosen are: 1 → 2 (with vehicles from origin lane 11 only), 3 → 2, and 4 → 3 (see Fig. 3.17). The route origins for these lanes are henceforward referred to with the origin lane names: 11, 31, and 41. The origin and destination codes for this intersection are repeated in Fig. 4.19.

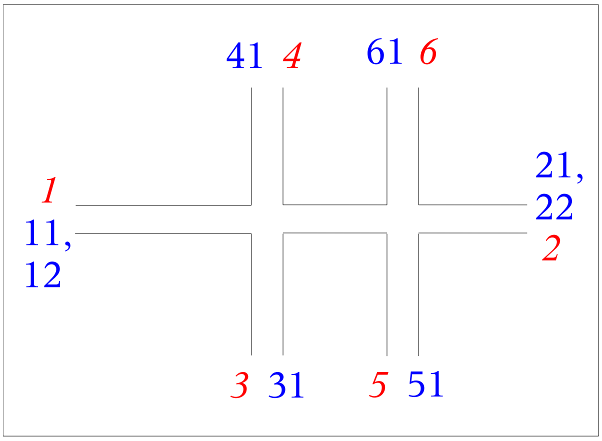


Figure 4.19: Origin and *destination* codes for the 2-intersection case.

The route 11 → 2 was chosen because it had some heavy congestion on it for part of the simulation time. The route 31 → 2 was interesting, because most of the vehicles from origin 3 used this route instead of the route 31 → 4 and most of the route length was often heavily occupied by vehicles from origin 1. The route 41 → 3 is an example of a short route with a lower, yet altering, flow level. — Long, cross-area routes like 51 → 4 were also considered, but the vehicle amounts remained so low that velocity comparisons are not reliable.

The HUTSIM runs were not performed again to measure the average velocities since the required information can be obtained from the *delfile* already available for all of the turn rate sets from the queue length runs. The CA runs were performed once more recording the vehicle information in the *vehleave* files. The runs were done in similar conditions as the queue length runs — only output files (*logfile* and *vehleave*) differ from these runs. Again, ten CA runs were made using different seed numbers for every four turn rate sets. The data concerning these runs can be seen in Table B.3 in Appendix B, and the associated figure information is included in Table B.4 of the same appendix. — The combined CA figures of the average velocities for different seed numbers now also contain the corresponding HUTSIM velocity graph because no additional, combined average velocities were calculated for the ten run sets (Figs. B.27–B.38).

Figs. B.27–B.30 show the average velocity curves on the route 11 → 2. The first thing to noticed from the pictures is that velocity values usually get more even as the turn rate set number grows. This is due to the lowering number of vehicles moving the whole route from start to end (the vehicle number from origin 4 grows a bit but it is not high anyway). With the turn rate set 1, a long-lasting congestion builds on the route before both signal posts, which can be seen as a significant decline of the average velocities between approximately the time classes 40–75 minutes. The velocity graphs for separate seed numbers are quite close to each other and also to the HUTSIM graph. The differences between the HUTSIM and the CA simulations present in the queue length graphs (at about signal period 40) cannot be seen now

— the apparent reason for is that the switchover of vehicles from one signal queue to the other (from 1-1 to 2-1) does not affect the average velocity on the route. Fig. 4.20 shows one example of the graphs for the turn rate set 1 with the flow profile curve from origin 1.

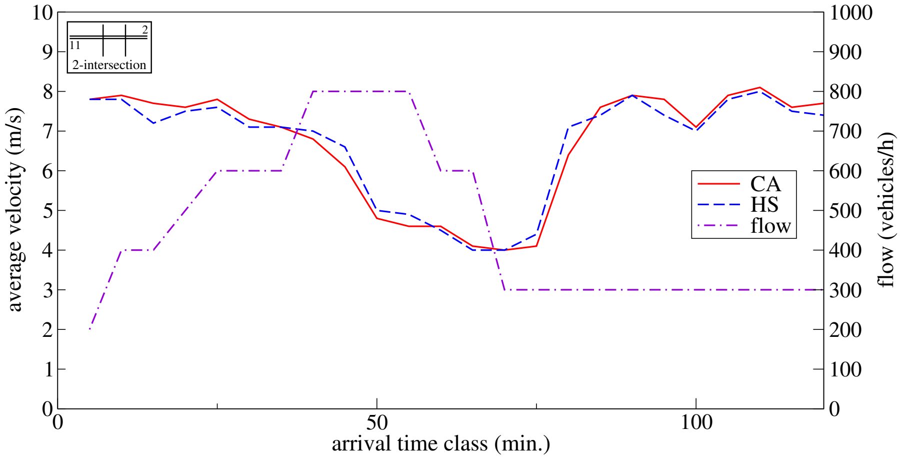


Figure 4.20: Average velocities on route 11 → 2 (from origin 1 to destination 2, flow profile from origin 1, CA run 49).

Regarding the other turn rate sets (Figs. B.28–B.30), the average velocities are quite smooth over the whole simulation period. The CA graphs show now a little more fluctuation with respect to each other. Yet they are quite congruent with the HUTSIM graph — the average velocities often are slightly higher than the HUTSIM values, which could be expected because of the somewhat higher velocity maximum attainable for the CA vehicles.

The next route to be examined is 31 → 2. Figs. B.31–B.34 show the graphs for separate seed numbers and the HUTSIM graph. The flow from origin 31 is steady and quite low with most of the vehicles heading to destination 2 also when the turn rate set number is larger than 1. The congestion between the intersections due to vehicles from origins 1 and 4 affects the velocities of the vehicles on this route.

For the turn rate set 1, the CA curves show clearly the velocity reduction caused by jams between the intersection area up to time classes at about 75 minutes. The CA velocities usually reach lower values than the HUTSIM vehicles — this is most probably caused by the more full lanes between the intersections for the CA vehicles than for the HUTSIM ones for reasons explained earlier in connection with queue lengths.

As the turn rate set number grows, the average velocities do not reach as low values as in the case of the turn rate set 1. There is also more fluctuation between the CA graphs in most parts of the examined time period. The vehicle amounts from origin 3 reaching destination 2 remain quite low as the turn rate set number grows (with the set 4, usually under ten vehicles per one five minute time class), so abrupt changes in the CA curves and discrepancies against the HUTSIM curve are common. Fig. 4.21 shows one example of the graphs for the turn rate set 3 with the flow profile from origin 1 marked in the figure.

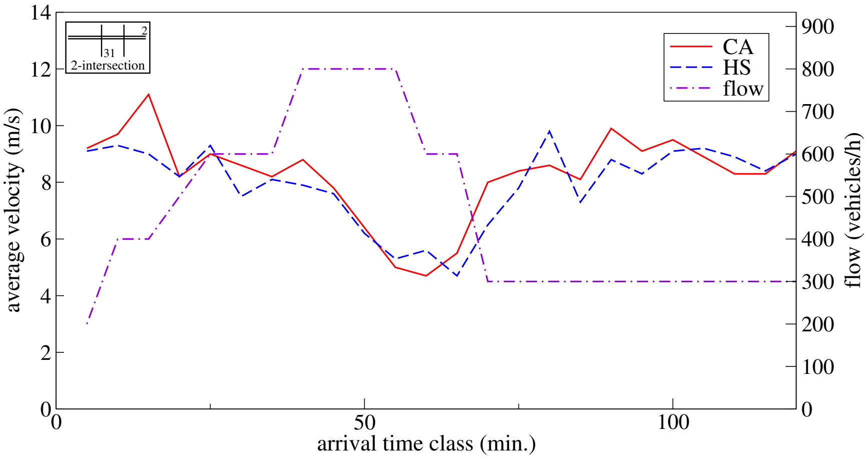


Figure 4.21: Average velocities on route 31 → 2 (from origin 3 to destination 2, turn rate set 3, flow profile graph from origin 1, CA run 61.)

The third route to be discussed is 41 → 3. This one-lane route has its own, varying flow profile from origin 41, as can be seen in Fig. 3.18. The average velocity CA graphs for different seed numbers and the HUTSIM graph are shown in Figs. B.35–B.38 in Appendix B.

In the case of the turn rate set 1 (Fig. B.35), the CA curves are quite close to each other outside the congestion area (between time classes about 50–75 minutes), and their shape is consistent with the HUTSIM curve. The congestion area brings variation to the average velocity levels although the curve shapes remain close to each other and to the HUTSIM graph. As a whole, the CA vehicles' velocities are again somewhat higher than those of the HUTSIM vehicles'.

Differences between the CA graphs in the other turn rate sets are larger practically at all parts of the examined time period. Furthermore, the congestion area clearly present with the turn rate set 1 fades away since the left turning lane starts to get more vehicles. In the congestion area still present in the case of the turn rate set 2 (Fig. B.36), the fluctuations in the vehicles' average velocity values are affected by CA statistics and by the temporary percentage with which the vehicles turn to the other directions.

Fig. 4.22 shows one example of the turn rate set 2 graphs with the flow profile from origin 41 marked on it. In this case the correspondence between the CA and HUTSIM results is quite good.

The velocity levels in the CA graphs for the turn rate sets 3 and 4 match quite well the levels in the corresponding HUTSIM graphs, but fluctuations in the CA curves

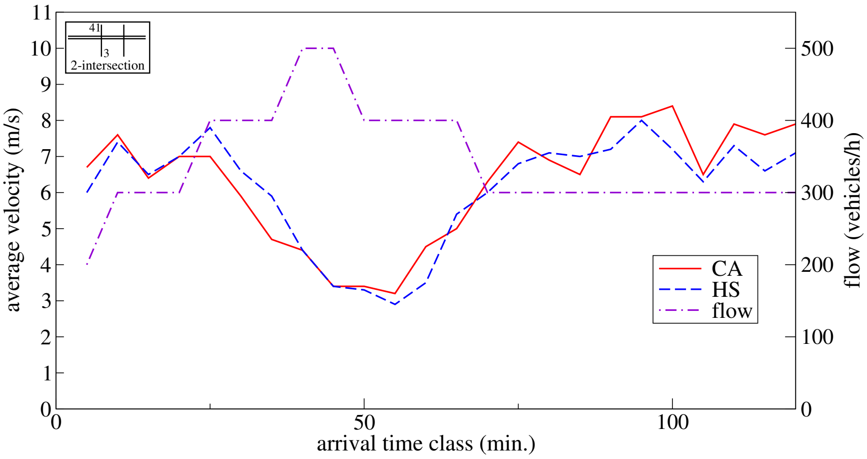


Figure 4.22: Average velocities on route 41 → 3 (from origin 4 to destination 3, turn rate set 2, flow profile from origin 4, CA run 60).

are already significant, which can also be due to the lowering number of vehicles reaching destination 3.

4.3.4 Traffic flow

The CA and HUTSIM simulators were also compared by running traffic flow tests for two detector configurations in the two-intersection case. A flow value is calculated as the number of vehicles crossing a detector site during the time of a traffic signal period. The sites for detectors 71+72 and 81+82 are shown in Fig. 3.26 in Sec. 3.7.7. The detector placement is repeated in Fig. 4.23.

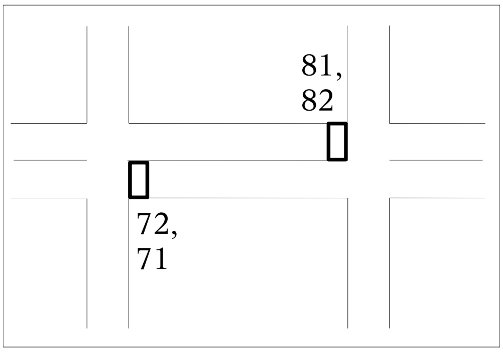


Figure 4.23: Flow detector placement.

The traffic flow runs were not planned to be detailed ones, the purpose was more to get a few examples of this detecting procedure. By limiting the detection time to some part of the signal period, it would be possible to get information about the vehicle amounts coming from certain directions to some of the post-intersection lanes.

Of the four turn rate sets used with the two-intersection case, the set number 3 was chosen, because the turning percentages were already quite large for it and fluctuations in the flow values for the two detectors could be anticipated. The CA simulation was again run for ten different seed numbers against one HUTSIM run. The CA run data are listed in Table B.5 (these runs correspond to the queue length and average velocity runs for the same turn rate set numbers).

Fig. 4.24 shows the aggregate number of vehicles crossing the detectors 81+82 during every signal period of 90 seconds for different CA runs. These graphs do not show large fluctuations among themselves at least when the vehicle quantities are low. The differences between runs at higher vehicle amounts are somewhat larger probably because statistics causes variation in queue dissolving before signal posts prior to these lanes.

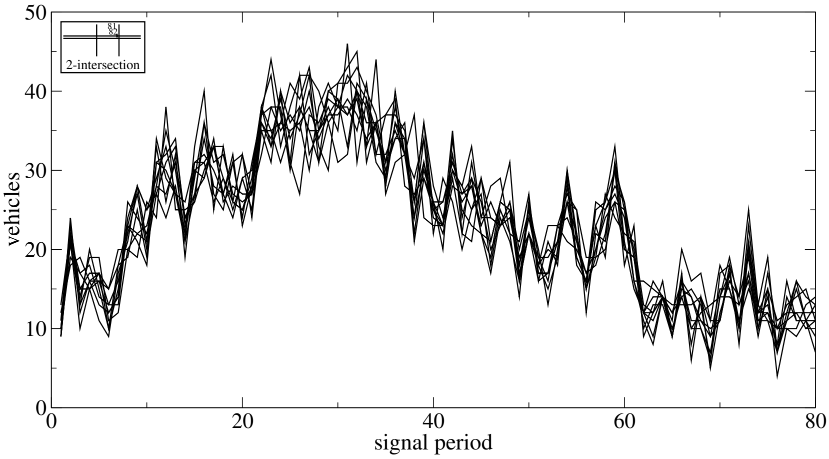


Figure 4.24: Vehicles crossing detectors 81+82, ten different seed numbers (turn rate set 3, CA runs 81–90).

In Fig. 4.25 are depicted the HUTSIM curve for this turn rate set and the CA curve whose vehicle values are calculated as averages from the ten CA runs. As can be seen, these curves match quite well even at higher vehicle amounts.

The results were also gathered from the above-mentioned runs for the detector pair 71+72. The joined figure for different seed numbers and the figure for the average CA graphs together with the corresponding HUTSIM graph are found in Figs. B.39–B.40 in Appendix B. Again, the CA curves differ from one another most at higher vehicle rates. The correspondence between the average CA graph and the HUTSIM graph is on about the same level as with the other detector pair.

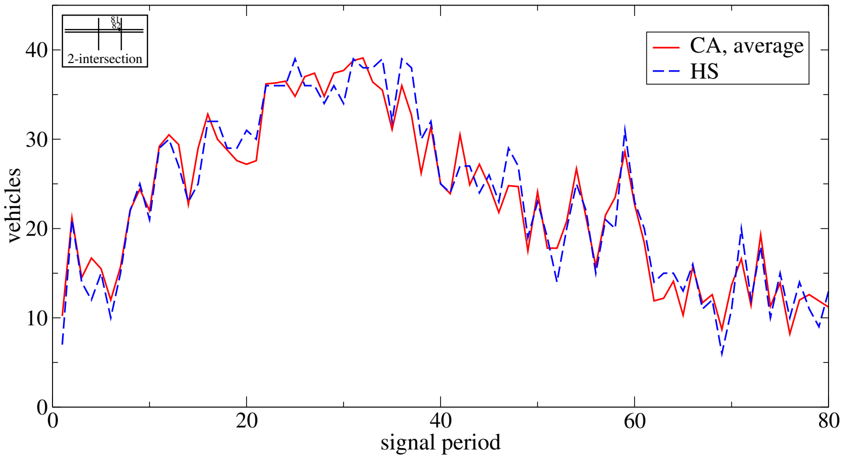


Figure 4.25: The average number of vehicles crossing detectors 81+82 (turn rate set 3, CA runs 81–90).

4.4 Five-intersection area simulations

The next example to be considered is the five-intersection case, the HUTSIM version is depicted in Fig. 3.14 and the CA version in Fig. 3.19.

The tests for this intersection type contain runs for queue lengths measured at each signal post, average velocities on some routes through the area, and a few runs for traffic flow values measured at detectors on the area between some of the intersections. Additionally, queue lengths were measured on some links with extra detectors installed to the models for correcting vehicle numbers.

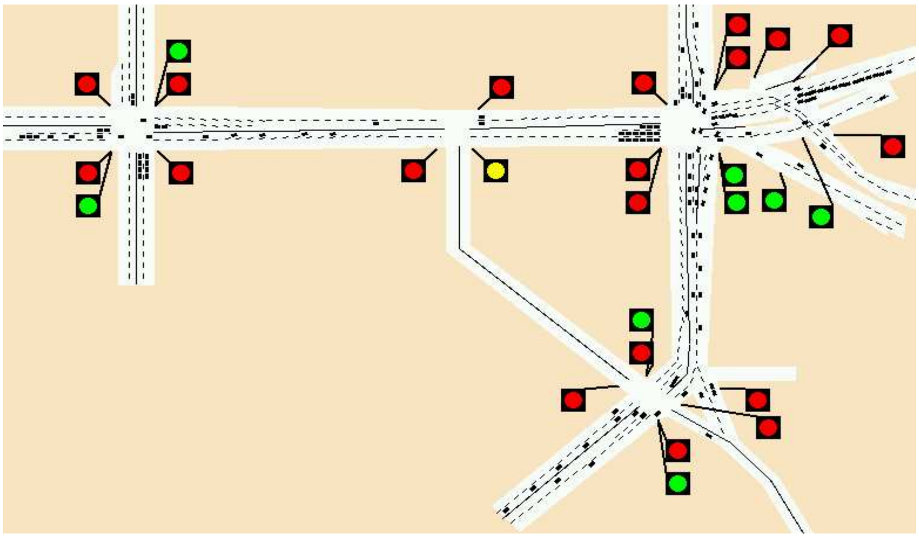


Figure 4.26: A situation example from the 5-intersection runs.

Fig. 4.26 shows a situation from part of the area in the five-intersection case.

4.4.1 Queue lengths

The queue lengths at different signal posts were measured in the same way as for the two-intersection case. The measurements were carried out for the five-intersection case for every signal marked as 1-1, 1-2,..., and 5-12 in Fig. 3.19. The signal codes and the turning directions associated with these signals are shown in more detail in Figs. 4.27–4.28.

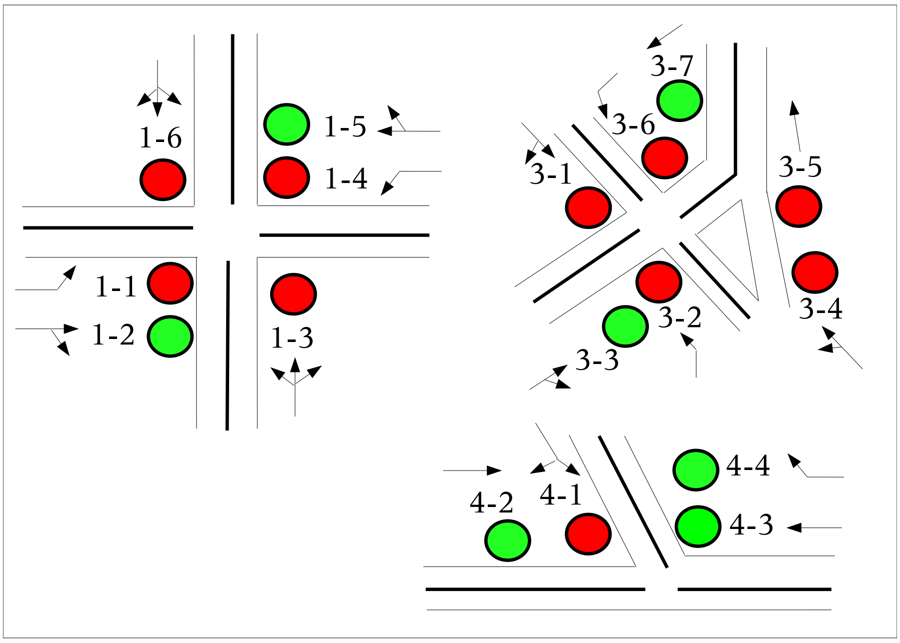


Figure 4.27: Traffic signal codes and the associated turning directions for the intersection on left and for the two lowest ones in Fig. 3.19.

The queue length values measured are compared and presented as figures displaying the vehicle count as a function of the signal period number during the 3 hour 45 minute simulation time (period length for the signals in this model is also 90 seconds). The figures presented here mostly contain comparisons at signal posts 1-2, 3-3, 5-1, and 5-11. These posts were chosen because they are situated on lanes of higher flow values. They also represent cases with a varying number of lanes connected to them, for example, all the vehicles coming from origins 31 and 32 obey signal 5-1, which exemplifies signals 1-3, 1-6, 4-2, etc. On the other hand, only part of the vehicles coming to 1-2 and 3-3 from origins 11, 12 and 51, 52, respectively, obey these signals. Some of the signals regulating traffic on the links between intersections are discussed at the end of this section in connection with extra detectors.

Ten CA runs were performed towards one HUTSIM run using different seed numbers. The results are depicted in Figs. B.41–B.48 in Appendix B for the signal posts mentioned above. For each signal post, there is one figure displaying joint graphs

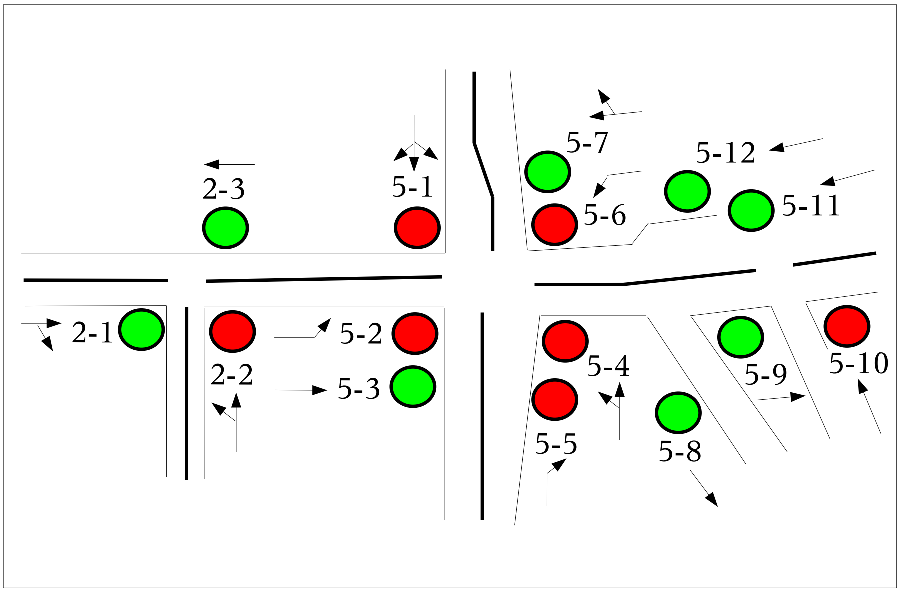


Figure 4.28: Traffic signal codes and the associated turning directions for the two intersections in the upper right part of Fig. 3.19.

for different seed numbers and one figure where the average queue lengths of the ten CA runs are compared to the corresponding HUTSIM run. Table B.6 in Appendix B gives file and parameter information for the runs, and Table B.7 connects the figures to these runs.

The average queue lengths given by the ten runs for the CA model generally match quite well the lengths given by the HUTSIM model. Also, the difference in CA length values between runs using different seed numbers are very small for those signals that regulate two parallel lanes — especially when the vehicle amount in the queues is less that about 15 (the queue lengths for post 1-2 are composed practically from one lane, the left one, because vehicle amounts on the right lane are very small).

The largest differences among the four comparison cases can be seen at the queues for signal post 1-2 (in Figs. B.41–B.42) starting at about the signal period 75. Here the HUTSIM values are usually larger than the CA values although the CA average curve shape follows the HUTSIM curve quite well. The reason for this discrepancy appeared to be a temporarily low percentage of HUTSIM vehicles turning left in this intersection that caused a long queue to be built; it took several signal periods for this queue to dissolve. In the corresponding CA runs, however, the high number of the left turning vehicles lowered the congestion level. — The CA variation between different runs is not significant in other areas of the time interval, and the correspondence of the average graph to the HUTSIM curve is good.

The figures for the queue lengths at 3-3, 5-1, and 5-11 show that the CA values are quite congruent in the runs with different seed numbers (especially 3-3 and 5-1). The good results can be explained by two facts: all of these cases are the first signals from respective origins in each link and have quite low vehicle amounts. Also, comparison of the CA averages to the HUTSIM graphs shows quite good agreement in each case.

In Figs. B.41–B.48 are depicted only a few of the 32 signal post queue length results that were measured in connection with this five-intersection model. Much of the length data not presented in these figures includes small queue lengths, which often makes the comparison unreliable especially in areas between intersections when randomness in determining the turning direction mixes up vehicle amounts. Also, differences in the model characteristics between VDR and HUTSIM affects queue building (maximum velocity not being exactly the same for both models and HUTSIM anticipation). Reasons for very short queue lengths subject to the influence of randomness are: low occupation on the lane (signal heads 1-4, 2-2, 3-4, 3-5, 3-6, 5-2, 5-6, and 5-12), the signal status changing to green while the queue is building up (signal heads 1-4, 2-1, 3-7, and 5-3), short duration of red signal status (5-8), short duration of red signal status with only the very remains of a queue in a green wave getting caught by it (2-3).

A few separate examples of the other queue length results are presented below (Figs. 4.29–4.31). They are related to the cases in which the vehicles have earlier passed at least one intersection, and they can thereby have largely differing vehicle amounts compared to the HUTSIM run. More examples of this type can be found in connection with the text describing the simulations using extra detectors (Figs. B.49, B.51, B.53, B.55, B.57, and 4.33), see below.

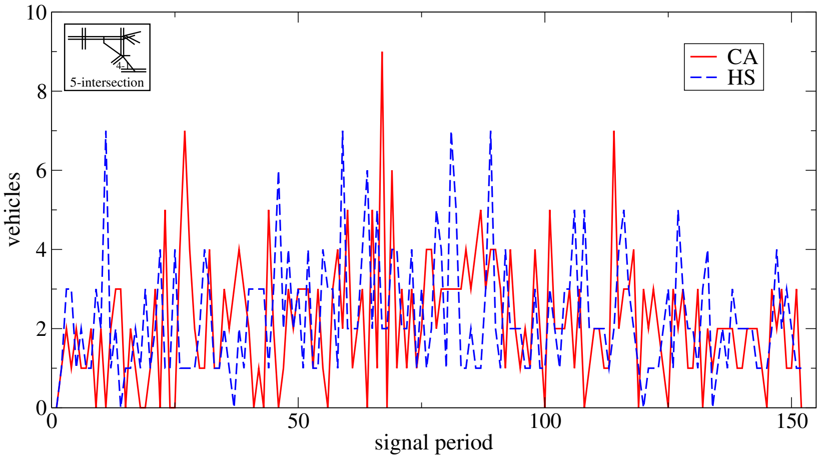


Figure 4.29: Queue lengths at 4-1 (right and left turning, CA run 93).

In Fig. 4.29 is an example of a queue length comparison at signal post 4-1 for one run. The queue length values are quite small because the flow rate on this link is low. The CA and HUTSIM curves differ considerably.

Queue lengths at signal post 1-5 can be seen in Fig. 4.30 for one run. In this case the CA queues are almost systematically longer than the HUTSIM queues.

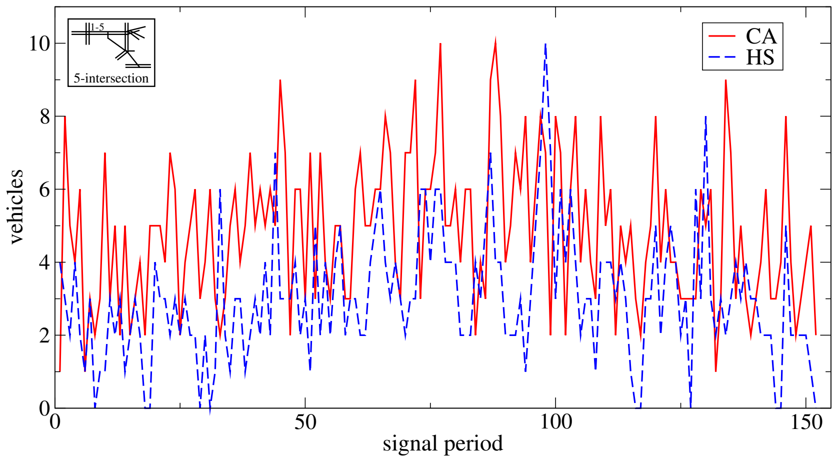


Figure 4.30: Queue lengths at 1-5 (straight going and right turning, CA run 91).

Reasons leading to this behaviour include the 'green wave' existing between the two intersections in the upper part of the model figure (into the left-bound direction). The first vehicles coming from the signal post 5-6 only just reach the signal 1-5 when it turns to green. The CA vehicles use a bit larger velocity on the link than the HUTSIM vehicles, and consequently they have a little more time to build up the queue. On the other hand, the HUTSIM vehicles are able to lower their velocity before they reach the queue end at the signal showing red and thereby do not arrive in time to build the queue.

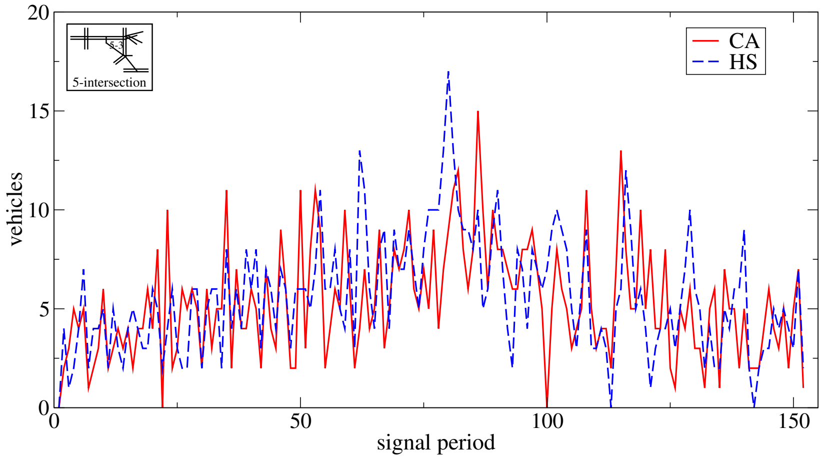


Figure 4.31: Queue lengths at 5-3 (straight going, CA run 97).

In Fig. 4.31 is a queue length comparison at signal post 5-3. This figure shows again differences between the two simulators although the curves are partly quite similar in shape. Turning rate statistics can now be considered as the main reason for queue length differences here.

The five-intersection area was also tested with the extra detectors placed at four pairs of lanes after vehicles exit an intersection. This arrangement is described in Sec. 3.7.3, and the placement of the extra detectors was seen in Fig. 3.21. The detector placement is still repeated in Fig. 4.32.

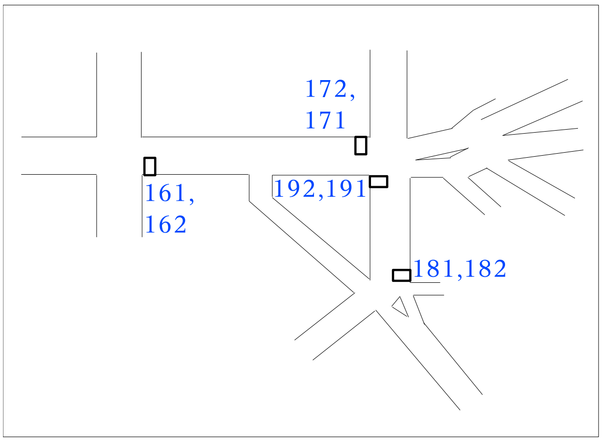


Figure 4.32: The placement of the extra detectors.

The purpose of the extra detectors is primarily to eliminate the differences for the CA and HUTSIM runs in vehicle amounts that take the direction in question after crossing the intersection. Also, the detectors enforce vehicles in both simulators to start the link at exactly the same time. The method should induce queue lengths at the next signals to be more equal between separate CA runs and as compared to the HUTSIM runs than without them.

Because of the placement of the extra detectors, the changes in queue lengths should especially be related to the signals 1-4, 1-5, 3-6, 3-7, 5-2, 5-3, 5-4, and 5-5.

The HUTSIM run with the extra detectors was made once with a new configuration file. The test arrangement is practically the same as with the run using normal detector positions — only some of the lane-changing areas are shorter than before, so the results can be compared to the previous runs. The CA simulation was run again with ten seed numbers; the run and figure data are listed in Tables B.6–B.7 in Appendix B (CA runs 101–110). Most of the figures related to these runs are in the same appendix (Figs. B.49–B.58).

The first figure pair, Figs. B.49–B.50, shows the queue lengths at signal 5-3 for CA runs with different seed numbers before and after installing the extra detectors. Although there is one intersection between the extra detectors and the studied signal, the queue lengths are much more uniform in the latter case indicating the effect of similar circumstances in vehicle flow at the start of the link (of course, a varying number of vehicles turning to the right at signal 2-1 causes differences in numbers reaching the signal 5-3). The influence of similar vehicle quantities (and entry times to the link) is even clearer for queues at signals 3-7 and 5-4, the latter case of which is portrayed in Figs. B.51–B.52. — In the signal 1-4 queues, the effect of the extra detectors is obscure: the queues are in any case very short because of the 'green wave' on the link between the signals 5-6 and 1-4.

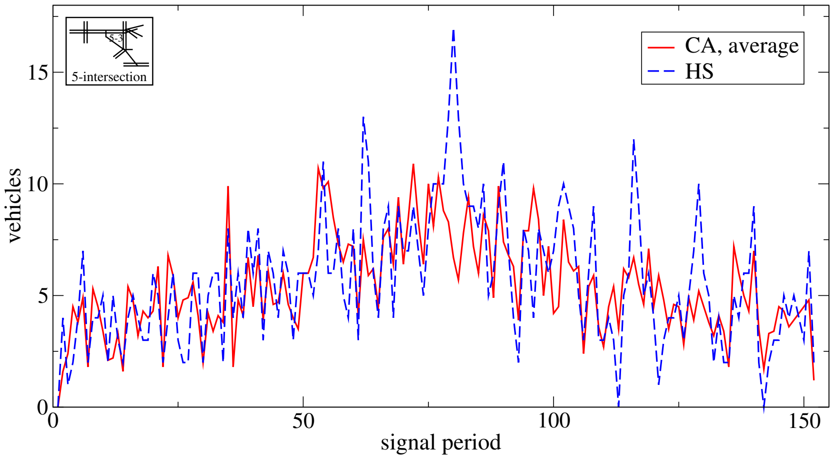


Figure 4.33: Queue lengths at 5-3 (straight going, CA runs 91–100).

The average CA queue lengths are also compared to the HUTSIM queue lengths at all of the eight selected signals. One of these cases, signal 5-3, is shown in Fig. 4.33 with the normal detector placing and the one with the extra detectors is in Fig. 4.34.

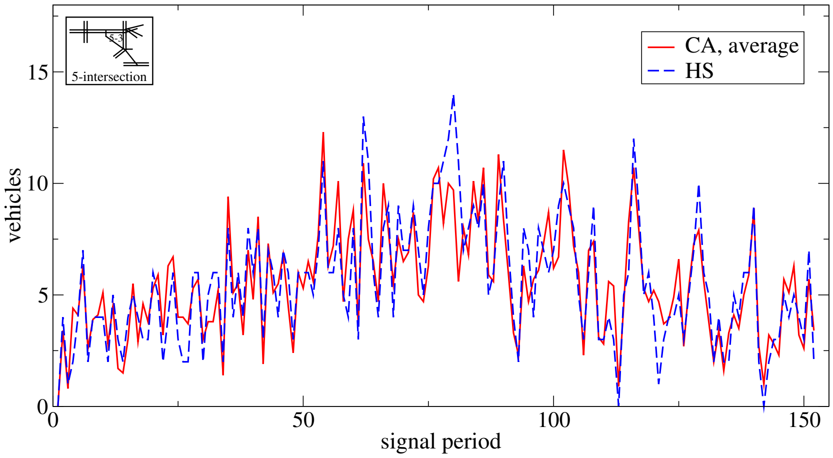


Figure 4.34: Queue lengths at 5-3 (straight going, CA runs 101–110, extra detectors).

The correspondence between the CA average graph and the HUTSIM graph is not very good in the normal case (Fig. 4.33) although the graph forms are partly quite similar. The CA average graph usually oscillates much less than the HUTSIM graph: the reason for this are the large differences between separate CA runs seen in the Fig. B.49, the averaging smoothes the CA curve. In the case with the extra detectors, the CA average graph is mostly very close to the HUTSIM graph also in the absolute queue length values as seen in Fig. 4.34. — Similar difference between the cases can be seen for the signal 3-7 queues (Figs. B.55–B.56).

In the case of the signal 1-4 (Figs. B.53–B.54), the extra detectors do not improve the CA average length curve much, perhaps the latter case curve oscillates a bit more suggesting more uniform results for different seed numbers.

As was seen in Fig. B.51, the CA run results at signal 5-4 for different seed numbers are quite non-uniform. This flattens the average CA graph of Fig. B.57 although the form of the average graph may in places be similar to the HUTSIM graph. The CA average lengths are normally 2 or 3 vehicles larger than the HUTSIM values, which is mostly due to two reasons. Firstly, most of the HUTSIM vehicles come from behind signal 3-2 and they use a slightly smaller velocity in the large intersection area. Secondly, the vehicles decelerate earlier than the CA vehicles before the signal 5-4, which is in red state as the queue from 3-2 reaches it.

The effect of the extra detectors can be seen in Fig. B.58: the CA average graph is very close to the HUTSIM graph in shape and also in absolute values. The queue lengths are still a little bit larger for CA vehicles because of the deceleration of the HUTSIM vehicles and the 'green wave'.

4.4.2 Average velocities

The average velocities on different routes were also measured for the five-intersection area. The aim in preparing these comparisons was to find long routes in different parts of the area. Three examples of the several possible routes were chosen for a closer review: 11 → 100, 51 → 30, and 72 → 30. Better choices instead of the route 72 → 30 would have been, for example, 72 → 20 or 72 → 40, but the vehicle amounts on these routes were too low. The measuring time even on the route 72 → 30 had to be increased to 10 minutes per period to get most of the vehicle amounts to exceed zero level (the flow from origin 7 is quite high as can be seen from Fig. 3.20, but most of the vehicles go to the destination 60). The origin and destination codes for this intersection are yet seen in Fig. 4.35.

The CA simulations were again run 10 times with different seed numbers against one HUTSIM run. The CA run data is listed in Tables B.8–B.9 in Appendix B (CA runs 111–120). The Figs. B.59–B.61 show the results for the CA runs with different seed numbers and the corresponding HUTSIM run.

On the route 11 → 100 the CA graphs show variation between themselves, mostly during the exit time classes 100–150 minutes. The reason for this can be expected to be randomness. The route is quite long (about 860 metres), and the acceleration from signal stops and the number of vehicles turning to other routes varies in different runs. Most of the graphs follow the shape of the HUTSIM graph although the CA average velocities are, as a whole, somewhat higher than those of the HUTSIM vehicles' (this is partly due to the larger CA maximum velocity).

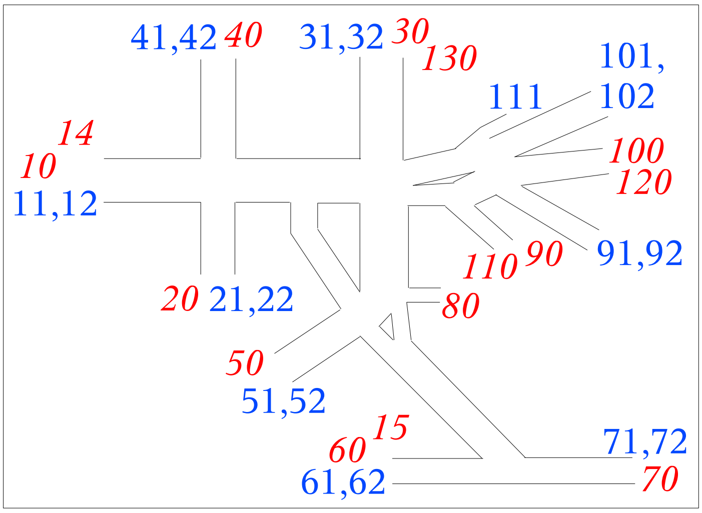


Figure 4.35: Origin and *destination* codes for the 5-intersection case.

Between the time classes about 110–140, the HUTSIM average velocities are usually much lower than the CA velocities, which is mostly due to the large queue lengths in the HUTSIM run seen in Fig. B.42. The associated congestion causes more vehicles in the HUTSIM simulation to stay at the 1-2 signal (and probably also at the 5-3 signal) over two periods than in the CA runs. One example of the separate runs is seen in Fig. 4.36.

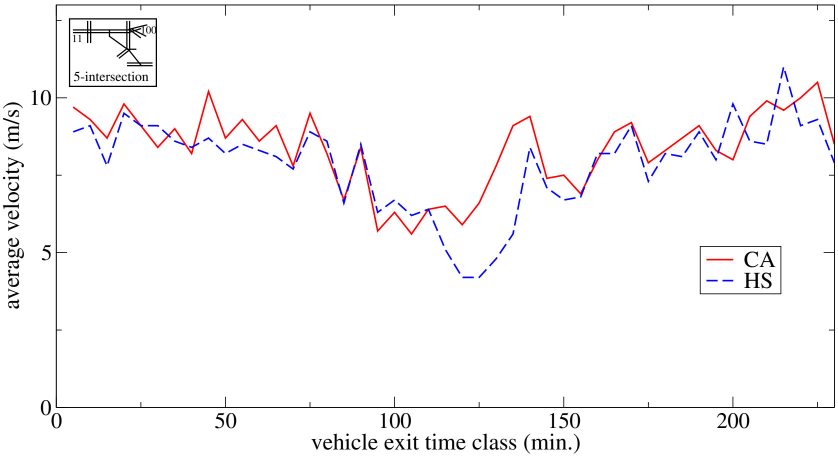


Figure 4.36: Average velocities on route 11 → 100 (CA run 112).

The results for the route 51 → 30 are in Fig. B.60. As can be seen, the different CA run graphs are fairly similar, and they are also quite consistent in shape with the HUTSIM velocity level graph. The route length (about 480 metres) is much less than for the 11 → 100 route, and the vehicles usually stop on the route only once at the signal 3-3. The CA graph shapes follow closely the HUTSIM graph shape, but the velocity levels are practically throughout somewhat higher for the CA vehicles (the HUTSIM vehicles use smaller velocities in the curvature of the road than the CA vehicles). — One separate example for the 51 → 30 runs is depicted in Fig. 4.37.

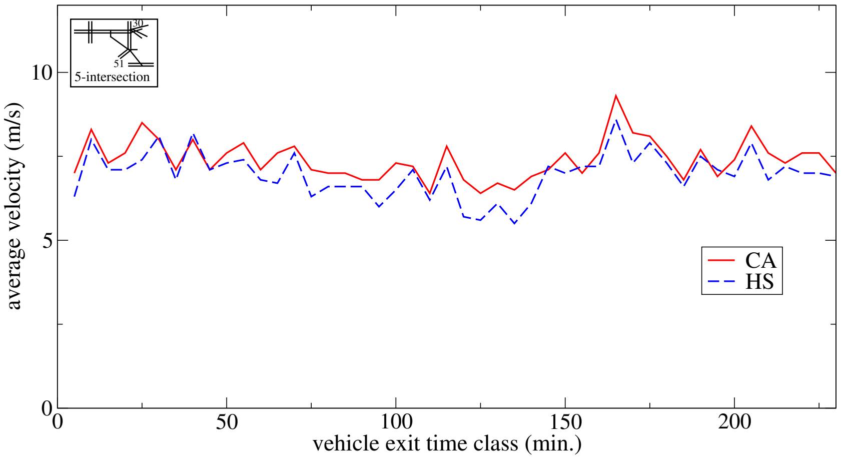


Figure 4.37: Average velocities on route 51 → 30 (CA run 120).

The third route 72 → 30 was not very frequently used, which led to several 5 minute observation periods to get zero vehicles to finish it. That is why the time class length was doubled to 10 minutes, and still the vehicle count was usually less than 10 vehicles per period. In Figs. B.61–B.62 in Appendix B are the CA and HUTSIM run results for this route and one separate example of the comparisons.

The small vehicle count, long route length (about 750 metres), and possible stops at the three signal posts on the route 72 → 30 cause variation between the CA graphs. The average velocities for the CA vehicles often seem to be higher than for the HUTSIM vehicles. In all, the velocities remain quite low because of the several signal stops although these vehicles are able to travel practically without any congestion on the route.

4.4.3 Traffic flow

Traffic flow comparisons between CA and HUTSIM for the five-intersection case involve measurements on two links with four detectors. The placement of the detectors was shown in Fig. 3.27 for the CA model and is repeated in Fig. 4.38. The number of vehicles passing the detectors was measured during traffic signal periods. The period length was 90 seconds.

Ten CA simulations were run with different seed numbers against one HUTSIM run. The run data are collected in the Table B.8 in Appendix B (runs 121–130). The CA simulation graphs are in the same appendix in Figs. B.63–B.64. The graphs show the summed number of vehicles for both detectors in the adjacent lanes for the ten CA runs, respectively.

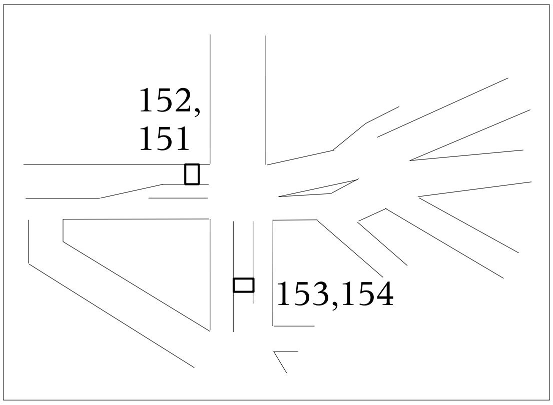


Figure 4.38: The flow detectors for the 5-intersection case.

The CA run graphs show some variation in the vehicle quantities between different runs as can be expected. The variation is larger for the detector pair 151+152 than for the pair 153+154. The obvious reason for this is that the detectors 151+152 get most of their vehicles from the origins 91, 92, 101, and 102 which give vehicles, besides to the destinations 10, 14, 20, and 40, also to the destinations 50, 60, and 70, so randomness changes the amounts between these destinations on different runs. Detectors 153+154 get most of their vehicles from the origin 51 that does not give many vehicles to other destinations than 30, therefore the flow is quite stable on different CA runs.

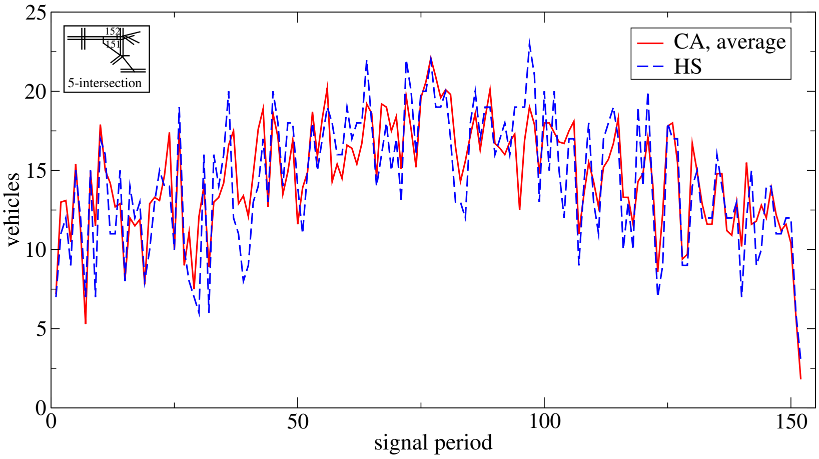


Figure 4.39: Average number of vehicles crossing detectors 151+152 (CA runs 111–120).

Fig. 4.39 displays the average graph for the above-mentioned ten CA runs (of Fig. B.63) and the associated HUTSIM graph. The average CA curve follows the HUTSIM curve quite well at least in shape although the amounts usually differ.

The average CA graph for the detector pair 153+154 can be seen in Fig. 4.40 together

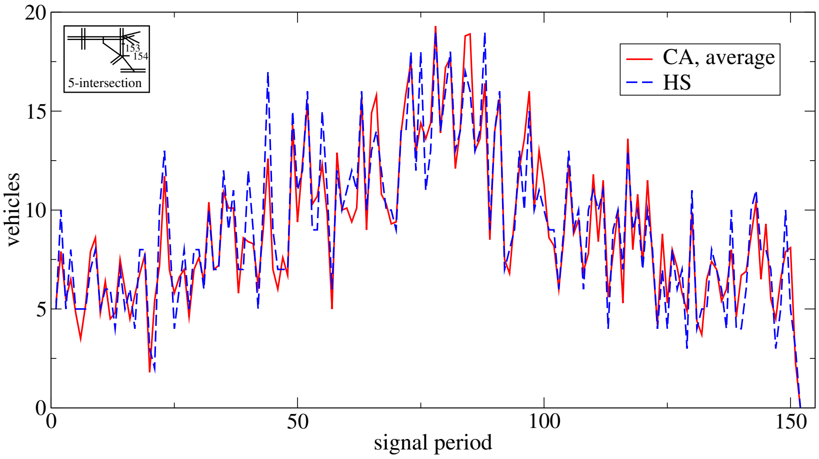


Figure 4.40: Average number of vehicles crossing detectors 153+154 (CA runs 111–120).

with the HUTSIM graph. In this case the resemblance of the CA graph shape with the HUTSIM graph is even better than for the previous detector pair. Also, the vehicle amount values are quite congruent with the HUTSIM vehicle amounts.

4.5 Freeway area simulations

4.5.1 Average velocities

The freeway environment was studied with the CA model (shown in Fig. 3.22) with two input files described in Sec. 3.7.3. Since the HUTSIM simulation results were already available, only the CA simulations were run. The origin, destination, and detector codes for the freeway case are still repeated in Fig. 4.41.

The freeway simulation situation differs a lot from the intersection environments considered earlier in this study. The freeway area contains neither normal intersections nor traffic signals, and the maximum allowed vehicle velocities are higher than in the urban models. The vehicle input material also contains large amounts of vehicles coming in from on-ramps without signal regulation, which easily causes congestion around the ramp areas. For reasons mentioned above, the program code and the run parameters had to be changed a little for the freeway runs.

The program code changes include the following two items: selection of candidates for lane-changing (see Sec. 3.3.4) covers also moving vehicles with velocity $\leq$ VMAX–1 and, in the actual lane-changing situation of the candidates, the minusgap need be $\geq$ VMAX–3 instead of VMAX, Table 3.2. For normal lane-changing the condition 4 in Table 3.2 is changed to: 'minusgap $\geq$ VMAX–3 .AND. v–2 $\leq$ plusgap .OR. v $<$ 1 .AND. prob. cond.'. Also, the value of p is used for the parameter p_0 for the first cell of an origin pipe to avoid feeding jams at origin spots during large flow rates.

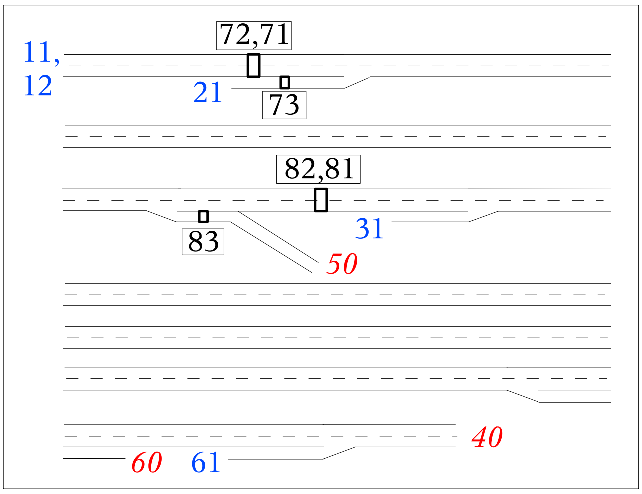


Figure 4.41: Origin and *destination* codes for the freeway case. Approximate detector positions and detector codes (in rectangles) are also marked.

Two of the parameters in *generalpar* (see Appendix A) were changed to make merging of traffic flows from on-ramps to be smoother: 'gap for probing lane-changing' → 25 and 'probability for lane-changing' → 0.95.

The cell length is kept as 5 metres in the freeway model, but the maximum velocity for vehicles is now 6 cells/time step. — Fig. 4.42 shows one situation of the freeway simulations.

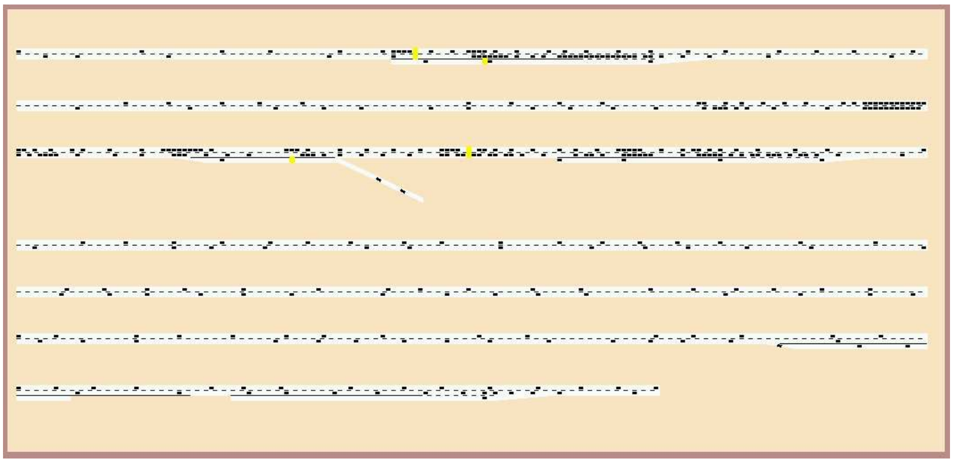


Figure 4.42: A snapshot of the simulations of the freeway case (model VDR).

The braking parameters used earlier with the VDR model cause too much congestion in the freeway case extending queues far in front of the first on-ramp (and outside the simulation area). This is because the value of p_0 affecting the start of the stopped vehicles was quite large in the intersection runs. The value of p_0 was changed to 0.160 and the value of p to 0.120, which together give a congestion profile between the first two on-ramps of the area to be quite similar to that in the HUTSIM simulation.

The larger p value also lowers the maximum velocity a bit for the CA vehicles (to about 105.8 km/h). — With the new parameter setting, the queue front line was checked as in Sec. 4.1.2, and it was found to be 15.2 km/h for the VDR model and 15.7 km/h for the test model, both values close to the value in [36].

The average velocities in the freeway case are calculated in a different manner than in the earlier cases of this study. This is because the reference values calculated in [46] use a procedure in which the average velocity on the examined link during a certain time sequence is a weighted average of the velocities of those vehicles that have travelled on the link during this sequence. The weighting is based on how much time (compared to the length of the time sequence) a vehicle spends on the link in question. The length of the time sequence is 0.5 minutes and the simulation time is 5 hours.

The CA simulations performed contain 10 runs with different seed numbers for both vehicle input files for the two CA rules used. The runs are listed in Table B.10 in Appendix B with CA run numbers 131–170.

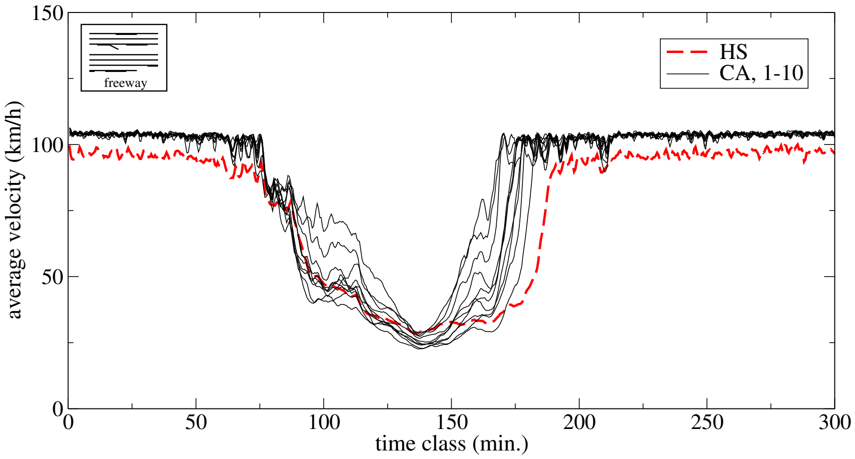


Figure 4.43: Average velocities on the freeway stretch (VDR model), file 1 (CA runs 131–140).

The runs for the VDR model (runs 131–140) with the input file 1 (IH1.ARR) are seen in Fig. 4.43 together with the HUTSIM result. Outside the congestion area, the velocities for both simulators are quite steady with the CA values being higher because of the CA cell and time step lengths and the maximum velocity value. In the congestion area (time classes between about 75–180), the starting point of lowering velocity values is quite similar for both simulators except for the differences in separate CA runs, which are due to statistical reasons. At the lowest velocity values, the shapes are often similar for both simulators, but the CA vehicles usually reach lower values than the HUTSIM vehicles. At the area of growing average velocities (time classes 150–180), the CA vehicles recover from congestion earlier than the HUTSIM vehicles and the differences between separate CA runs are large.

The reasons for the differences in CA runs are statistical, but the differences between the CA and HUTSIM results are a consequence of the distinct behaviour of vehicles in congestion areas: the HUTSIM vehicles are able to gradually reduce their velocity in front of a disturbance in flow, for example, a large amount of vehicles coming from an on-ramp. The CA vehicles, on the contrary, carry on practically at full velocity before they stop at a queue end or some other obstacle. Starting from a standstill takes some time, and the result is a stop-and-go flow that moves upstream. In the HUTSIM environment, the flow in congestion areas is uniform with all the vehicles having about the same velocity. — An example of the CA occupation profile in a snapshot from congestion can be seen in Fig. B.65 in Appendix B.

The vehicle type distribution is significantly different in the simulators since the CA side only has one type of vehicles, passenger cars, but HUTSIM has also larger vehicles: trucks (about 4%, length 7 m), buses (about 2%, length 12 m), and trailers (about 6%, length 18m). This makes the road more occupied and lowers somewhat vehicle velocities at least in congestion in the HUTSIM simulation. Moreover, before off-ramps the HUTSIM vehicles about to take the ramp lane reduce their velocity, which affects also vehicles behind moving straight thereby lowering the average velocities. The CA vehicles do not behave in this way although the ramp lane has the same speed limit as the HUTSIM model does. The vehicle amounts on the whole freeway area in the HUTSIM simulation are much higher than in the CA simulation during congestion (minutes 105–165): CA vehicles present on the area are about 70–75 % of HUTSIM vehicles (checked for the CA run 131). Furthermore, the higher occupation level extends to the last off-ramp in the HUTSIM simulation, whereas in the CA simulation only to the second on-ramp. All these reasons lead to the earlier recovery of the CA vehicles' average velocities.

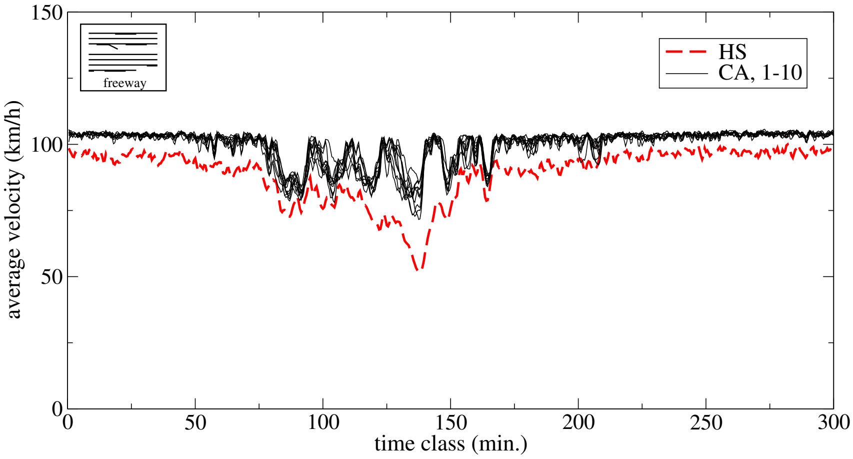


Figure 4.44: Average velocities on the freeway stretch (VDR model), file 2 (CA runs 141–150).

Run results with the input file 2 (IH2.ARR) are seen in Fig. 4.44 with the associated HUTSIM result. Although the input flow of vehicles is quite the same as with file

1, the congestion is much less heavy in the examined link because larger amounts of vehicles now exit via the first off-ramp. The input file 2 also contains different types of vehicles — the percentages are the same as with file 1 — but CA runs again use only vehicles of the passenger car type.

The shape of the CA graphs in this case follow in places quite well the HUTSIM graph, but the fluctuations are partly large and usually the CA average velocities remain higher. At intervals the CA vehicles also reach the maximum velocity level during the congestion period (time classes about 80–170), but the HUTSIM vehicles do not.

The same reasons for differences between the CA and HUTSIM results apply also now as with the file 1 case. Since the number of vehicles exiting through the first off-ramp is now larger, the congestion in the area of the second on-ramp is not as severe as previously; in the CA runs the lowering of velocities does not practically even extend to the area of the examined link, but the velocity decrease occurs only around the first on-ramp.

The freeway environment was simulated for both input files with the CA test model, too. These runs are listed in Table B.10 with the CA run numbers 151–170. The basic model parameters are the same that were used with the VDR model with the exception of the parameter p_0, for which the value 0.130 was used.

Fig. 4.45 shows the results for ten CA runs with different seed numbers and the HUTSIM result. This time the differences between CA runs are not as large as with the VDR model at the descending and ascending parts of the graphs. However, the CA vehicles' average velocities again recover earlier after the congestion at around time class 175 but as a whole later than with the VDR model.

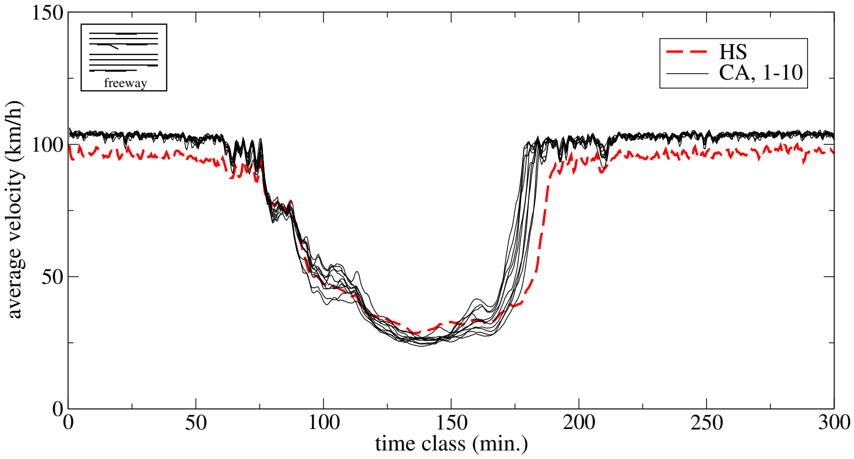


Figure 4.45: Average velocities on the freeway stretch (test model), file 1 (CA runs 151–160).

The road occupation with this model resembles the HUTSIM simulation more than that of the VDR model did. CA vehicles present on the whole area are about 94–

100% of the HUTSIM vehicles (checked for the CA run 151) during simulation time classes 105–165, and vehicles are more evenly distributed than in the VDR runs. The area between the second on-ramp and the last off-ramp is also more densely occupied than in the VDR run (see Figs. B.65–B.66 for occupation comparison).

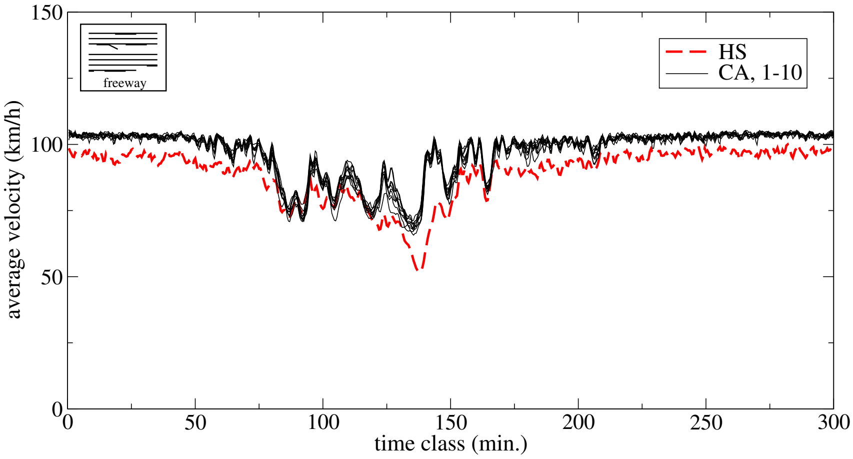


Figure 4.46: Average velocities on the freeway stretch (test model), file 2 (CA runs 161–170).

Simulation results for file 2 with the CA test model can be seen in Fig. 4.46. The differences between separate CA runs are now quite small. Furthermore, vehicles practically do not reach the maximum velocity level between the time classes 75–140 as often happened with the VDR model runs. Up to the time class 130, the correspondence to the HUTSIM results is also good with the exception of the highest velocities before the congestion area. Recovery from the lowest velocity values at time classes 140–150 occurs again somewhat more easily for the CA vehicles.

Errors to the comparison of the CA and HUTSIM results are again brought by the coarse discretisation of the time step and cell lengths in CA runs, which makes the CA vehicles' maximum velocity about 5.8 km/h higher than the HUTSIM vehicles have. This can be seen in all the comparison figures in the areas outside congestion times.

Differing vehicle types should not have any consequences to the average velocities when the traffic is not dense. In jams the same vehicle flows should mean thinner occupation for the CA vehicles and therefore higher velocities. The CA model parameter selection, however, was made on a rough basis, and it can easily obscure this kind of effects.

Chapter 5

Conclusions

5.1 Technical issues of program and interface development

The purpose of this work was to study the applicability of one type of microscopic simulation models, cellular automata, in producing indicator values identifying the prevailing traffic situations in road networks. Secondly, the study is related to the systematics of measurement and modelling in the sense that how well the chosen method of data collection suits in the production of these indicators. The test environments included both urban intersection areas and a freeway case, and the traffic situations were probed by queue lengths, average velocities, and flow rates. Also, the aim of the study was to use CA models to examine queue formation of vehicles as a statistical process at some obstacle.

Much effort was spent in developing the program that establishes the modelling environments and implements the chosen CA rules describing the vehicle movement. Already at the beginning of the work it was evident that the program was to be able to satisfy versatile requirements for modelling areas. Yet it did not seem necessary to build an all-purpose simulator used in any kinds of road traffic situations since the focus was to be expected to be in urban intersection cases and in the testing of the CA simulation of vehicle movement.

The generated program can be used to construct model areas of various sizes. Although the construction of a test structure is quite laborious because of the lack of an editor program, road lines of different angles and several lanes can be realised. A unit of road in the model is one pipe that constitutes one lane or part of it in the road structure. The solution is similar to the one in the reference model program, HUTSIM, and consequently it is possible to build the model areas quite similar in both simulators. Of course, the minimal length of CA road units, cells, prevents using very short stretches, which necessitates some adjustments especially in road deviations to get the structures to resemble one another as closely as possible. This was not eventually a problem in the studied cases since the differences remained minor, and the effects, for example, in average velocities cannot be noticed. Lane-changing is possible in the CA program, but the number of parallel changes is limited. This deficiency came up in one part of the five-intersection test area, but the problem could be evaded quite comfortably by fixing the lane-changing areas to a sequence.

The CA simulator program implements normal signalisation features, and yielding and speed limits are in use of features concerning traffic signs. A controlling system for traffic signals does not exist since the signal status changes are received from outside (input file). The program neither creates the vehicles by itself but reads the arrival instances for them from input file. If the vehicle entries are not available from reference program output or from field measurements, they can be produced by an additional program that was developed for the general feature runs.

In the development of a simulation program of this type, it is worth while taking into account several aspects. In the simulator of this study, the interface part was realised with a scripting language program and the actual calculation code with a conventional programming language. The choice was quite satisfactory since the used scripting code can provide expressive construction alternatives and animation possibilities for two-dimensional presentations adequate for the cases in question. If the interface and the rest of the program use different languages, communication is needed between the two parts; there is reason to make certain in advance that passing that information does not seriously affect the performance when the data volumes grow. Also, the moving of a large number of objects of the screen may seriously consume resources.

Another issue directly connected to the interface is the building of the test area. A proper editor for the construction of the model area usually becomes valuable as the structure gets complex. Furthermore, zooming capabilities are often handy both in the building phase and later during the actual runs.

Most of the calculations in this sort of simulators are associated with the vehicle movements and the decisions involved, which may induce large amounts of calculation operations. This can lead to capacity problems when the amount of vehicles grows and should be taken into account when choosing the programming platform. Cellular automata models are inherently suitable for parallel computing, therefore this aspect should also be kept in mind if large road networks are in question.

The choice of the cell length is an important one with a CA simulation model, and it usually has to be made at the early stages of the program design. The tendency with the present day models is to use a rather short cell, for example, 1.5 metres in real space. The short cell length improves the vehicle dynamics in acceleration and deceleration and enables more velocity levels. A drawback is that the vehicles have to be split over several cells, which complicates the calculations. The cell length often used with CA traffic models is 7.5 metres, which describes well an average vehicle length when also busses and different sorts of lorries and vans are considered along with normal passenger cars. — The cell length for the simulator of this study was chosen to be 5 metres. The choice is probably a good one, because the behaviour of individual vehicles is not very important in the simulations performed and at least the queue lengths are rather realistic compared to the HUTSIM queues since only passenger cars were used. The disadvantage is that other vehicle types were not included and the reliability of the results for the freeway case suffers from this lack.

For future developments a short cell length (1.0 – 1.5 metres, for example) may be a good choice since usually the need is to have different sorts of vehicles in the system (real measured data probably contain various vehicle types). With CA models the modelling of single vehicle movement can be expected to be more important in the future and this may easily become problematic with a long cell length. The need for a short cell length should be considered when the program development is started because it affects significantly many parts of the program.

5.2 CA simulation in the studied environments

One purpose of the study was to examine CA simulation behaviour as a modelling means. For this aim two distinctively different types of test areas were used: urban intersections and a freeway case.

5.2.1 The two-intersection area

The two-intersection test areas were built from scratch for both simulators, so it was possible to make them practically identical. The behaviour of traffic on this area could also be examined thoroughly since many sets of traffic input material were generated using different turning probabilities for the vehicles. Summary tables for the queue lengths, the average velocities, and the flow measurements for this intersection case are presented in Figs. B.69–B.70 in Appendix B.

Maybe the biggest problem in the CA simulation of this study seems to be the influence of randomness to the results. This is frequently seen particularly in the queue length measurements of the two-intersection area where two successive runs can give quite differing results for the length values and consequently dissimilar information about the traffic situation for the link. The effect is prominent when the turning percentages and/or the queue lengths grow. Also, if only one lane is in use, the differences can get large even at low queue lengths. If two lanes are in use, the length differences between runs can be quite small even with very long queues at least when turning percentages are low; obviously two parallel lanes even up the effects of randomness.

Comparisons of the graphs giving the average queue lengths to the corresponding HUTSIM graphs show quite satisfying congruence both for one- and two-lane links especially when the turning percentages are low. Some special deviations mentioned in Chap. 4 may occur because of routing reasons and the different behaviour of the models: the HUTSIM vehicles can anticipate congestion ahead and reduce velocity, which may transfer vehicles to links before the previous signal head. The used CA model is not capable of doing this, and it also makes the vehicles form a queue quickly and tightly in case of an obstacle. — When the turning percentages get large, the differences between the average CA graphs and the HUTSIM graphs get more significant, but they still usually remain much smaller than the differences between the separate CA runs.

For the average velocities of the two-intersection case, the CA results do not differ much from each other or from the HUTSIM results with the lowest turning rates sets and when the link is (partly of totally) a two-lane one. Also for the studied one-lane link, the results are quite good with these turning rates for most of the time with the exception of a period after the flow maximum, which causes large variations in the CA results. As soon as the turning percentages start to grow, variations in the CA results easily become remarkable, and there seems not to be much difference whether the flow is higher or lower. A link with two lanes (for the whole link length) seems to give much steadier CA behaviour also with larger turning percentages. The HUTSIM average graphs are placed among the different CA graphs, and usually any systematic deviation cannot be seen there.

The flow curves for the two links in the two-intersection case show quite uniform behaviour between different CA runs. At the largest vehicle amounts, the deviations grow somewhat, but as a whole the results are satisfactory. Any significant differences could not be seen either when comparing the average CA graph to the HUTSIM graph, and the vehicle amounts did not have any special influence in this case.

5.2.2 The five-intersection area

When passing to the five-intersection area, it can be noticed that about the same applies to this case too as did for the previous one. The difference is that vehicles may have stop more frequently at different signals although the proportions of the two areas do not differ much. Furthermore, there was only one traffic data file available for this intersection case so that various circumstances could not be tested. The summary tables for the queue lengths, the average velocities, and the flow measurements for the five-intersection case can be found in Figs. B.71–B.72 in Appendix B.

The queue lengths at the signal heads at the end of the incoming links behaved usually very neatly between different CA runs probably because the vehicle amounts remained quite low, often the congruence is excellent. The average CA graph also usually corresponds well to the HUTSIM result graph.

The queue lengths at the intermediate signal heads are more problematic. Reasons for this are the statistical effects in routing and, on longer links, the maximum velocity difference and anticipation effects between the simulators. If the vehicle amounts are small, the fluctuations in the graphs may be significant also between the average CA graph and the HUTSIM graph. — The use of the extra detectors fixing the vehicle number reaching a link leads to better results when the reason for the differences is routing.

The results for the average velocities and flow rates for the five-intersection case resemble the results for the area with two intersections. The low number on vehicles on some links lessens the value of the results, but for heavier used routes, the results agree quite well with HUTSIM results.

Regarding the purposes of the study (Sec. 1.3), the results for the discussed urban intersection areas show that the used CA model (VDR) suits quite well in producing the chosen traffic situation indicator values as compared to the reference especially for averages taken over several runs. On the other hand, separate CA runs may produce results that differ substantially from the reference, so the performance of the model in on-line simulation contexts can fail notably.

5.2.3 The freeway area

The first problem with the freeway runs is the cell length in the CA simulation, which causes the velocity to get only separate, fixed values. This means that the 100 km/h value of the HUTSIM vehicles cannot be reached (with the chosen CA cell and time step lengths). The effect is seen as a difference in the average velocity level between the results during the free flow stages. With the VDR model, the different CA runs produce very congruent graphs during free flow, but again in the congestion, the CA graphs diverse from one another for statistical reasons (particularly with the heavier congestion case). Also, the recovery from low average velocities occurs in the ten CA runs earlier than in the HUTSIM run. The deviations are caused by the essential difference in vehicle dynamics: the VDR model does not have a mechanism that would enable realistic driving in congestion (low velocity and comparable intervals between vehicles).

The freeway area case was also simulated with the test model, which uses anticipation in the rule steps. The effect, even though not significant, can be seen as a more natural moving in congestion and in the comparison of the average velocity graphs. For both of the test files, the CA graphs are more congruent and the deviations from the HUTSIM graph are smaller than with the VDR model. Consequently, on-line simulation could be expected to work well in the freeway case particularly for the CA test model.

In all, the ability of the VDR cellular automaton model to predict the values reflecting the traffic situations in both intersection and freeway areas is quite good at least on the average level as compared to the behaviour of the reference model. When CA simulation is used in real time situations, this may not be adequate. Remedies contain the decrease of the cell length, which has been carried out in the latest CA models in use. This concerns mostly situations where vehicles often accelerate from rest or drive in congestion. In free flow the large cell length does not matter if the maximum velocity or the separation between velocity levels is not a problem. Another obvious improvement, which has also been taken into use with the newest models, is some sort of anticipation mechanism useful especially in congestion driving. For the measurement procedure, additional detectors are needed to make corrections to vehicle amounts after intersections. Also, average velocity measurements in real traffic require some method for vehicle and/or route identification.

5.2.4 Queue building

The queue length behaviour for the general feature runs shows quite similar results in all the tested cases. The graphs suggest that the probability of finding queues as a function of queue length decreases exponentially at the higher end of the reached queue lengths. The exponent varies according to the flow rate value chosen.

The flow rate does not seem to cause much difference in the graph shapes with either of the models if the graphs for the lowest flow values are not counted; any qualitative change in the shape cannot be seen while the flow rate is growing with the exception of the highest one for some of them. The highest flows cause spontaneous stop-and-go waves that affect the whole measuring area and easily produce feeding problems for vehicle supply.

When the queue lengths found at a certain probability level (P(50%)) are plotted against flow, the lengths grow rapidly at higher flow values suggesting phase transition.

The CA models can be expected to describe the queue forming quite well at least at the beginning of the queue building when an obstacle is met although vehicle deceleration is much smoother in real traffic. As the queue starts to dissolve from the downstream end, the queue length often starts to reduce. When the queue is short enough, vehicles (or drivers) in real traffic will not easily join the queue end but slow down quite early to escape stopping. The models used (with the exception of the test model) do not have this ability, and the vehicles join the queue end. Therefore, in real traffic the maximum queue lengths can be expected to be shorter than the lengths given by these models.

The future use of the CA modelling in situations similar to those discussed here is an option worth while considering. The benefits of CA simulation are significant particularly when rapid, on-line simulation is needed. The choice of the model depends on the simulation case, but more refined models than those used here are probably beneficial since more realistic dynamics of single vehicles seems to be sought for nowadays both in urban and highway/freeway environments.

5.3 Measuring and modelling

The measurement technique for producing traffic situation indicators with the detectors only on the borders of the areas seems quite adequate in the two-intersection case when the turning percentages are small. When the turning rates grow, the vehicle numbers after intersections need adjustments, which can be carried out with additional detectors at these locations. As the intersection count increases, the importance of extra detectors is evident.

The queue lengths, average velocities, and flow values are easy to measure in the case of both simulator programs, but in real life the situation is more complicated. As for the case of queue lengths, several detectors may be required if the variation of the length is large. Furthermore, the synchronisation of the length measurement with signal states is needed. The flow values assessed as vehicle counts in the network are easy to measure with detectors placed in appropriate locations, but the average velocity evaluation is more difficult. Although local velocities can be gathered with suitable detectors, the average velocity values on certain routes require identification of the vehicle routes. This may not be easy to accomplish even with equipment supplementary to detector and signal data collection. Using of probe cars might be the best alternative in this case although the expenses may then easily grow. In real life it is also necessary to take into account possible measuring errors regardless of the method used.

The CA simulation in the examined freeway case seems to work better than in the urban intersection cases. The average velocity as an indicator of the traffic situation matches quite well with the reference values (excluding the points mentioned) at least with the CA test model. The measurement technique with detectors only at the starting and ending spots of the routes is adequate for simulators although it does not tell everything about the flow in between (stop-and-go waves for the VDR model). In real traffic vehicle identification for average velocity calculations is again necessary, but that is now easier than in urban environments because vehicles cannot usually take various routes in the intermediate parts of the studied area of this type.

5.4 Future development

The use of CA modelling presented in this study has mostly been involved in circumstances where the simulation results have been compared to traffic situation information produced by another simulator program. The reason for this is primarily that (on-line) traffic data was not available at the moment the work was started with the Laboratory of Transportation Engineering. Also, the reference data could easily be made to cover different environments and spots by the used manner.

The input traffic data used in the simulations were mostly supplied by the reference simulator, and they correspond to information that might have been received directly from induction loops and signalisation systems. The next step in the development of the CA simulator program could be connecting it to on-line received data. In the future also other methods for data acquisition than induction loops become important: infrared detectors and microwave radars and information from probe (floating) cars.

The present work involves the generation of some traffic parameters (queue lengths, average velocities, and flow values) by CA simulation to describe the situations in the real world traffic. The next step would be the controlling of traffic parameter values to produce desirable and/or acceptable levels of service by the traffic system. This might be accomplished by coupling the controlling of the parameter values to simulation and developing a function relationship between the traffic input and the resulting traffic situation response. These kind of considerations are not included in this work, but they might well be suited as a subject for some future studies.

Present projects at the Laboratory of Transportation Engineering include two areas around the city of Helsinki, from which on-line detector data are transferred to the laboratory for the use by, for example, simulation programs [25, 39, 53]. The first one is a highway area consisting of six intersections, and the other one is a city environment of over ten intersections with heavy traffic for large periods of the daily hours. The traffic of these areas is planned to be simulated with various models, one alternative of which is CA simulation since the size of the sites and the volume of traffic impose substantial demands for the on-line simulation. The distant goal is the simulation of the traffic in the whole area of the city. For this kind of purposes, CA simulation may offer a sufficiently fast and lightweight solution.

Bibliography

[1] Bagnoli F., “Cellular Automata”, preprint/cond-mat/9810012

[2] Bando M., Hasebe K., Nakayama A., Shibata A., Sugiyama Y., “Dynamical Model of Traffic Congestion and Numerical Simulation”, Phys. Rev. E **51** (1995) 1035

[3] Barlovic R., Santen L., Schadschneider A., Schreckenberg M., “Metastable States in Cellular Automata for Traffic Flow”, Eur. Phys. J. B **5** (1998) 793

[4] Benjamin S. C., Johnson N. F., Hui P. M., “Cellular Automata Models of Traffic Flow along a Highway Containing a Junction”, J. Phys. A: Math. Gen. **29** (1996) 3119

[5] Bham G. H., Benekohal R. F., “A High Fidelity Traffic Simulation Model Based on Cellular Automata and Car-Following Concepts”, Transpn. Res. Part C 12 (2004) 1

[6] Biham O., Middleton A. A., Levine D., “Self-Organization and a Dynamical Transition in Traffic-Flow Models”, Phys. Rev. A **46** (1992) R6124

[7] Chandler R. E., Herman R., Montroll E. W., “Traffic Dynamics: Studies in Car Following”, Op. Res. **6** (1958) 165

[8] Cheybani S., Kertész J., Schreckenberg M., “Correlation Functions in the Nagel-Schreckenberg Model”, in *Workshop on Traffic and Granular Flow ’97*, edited by Schreckenberg M. and Wolf D. E. (Springer-Verlag, Singapore, 1998)

[9] Chopard B., Droz M., *Cellular Automata Modeling of Physical Systems* (Cambridge University Press, Cambridge, UK, 1998)

[10] Chopard B., Dupuis A., Luthi P., in *Workshop on Traffic and Granular Flow ’97*, edited by Schreckenberg M. and Wolf D. E. (Springer-Verlag, Singapore, 1998)

[11] Chowdhury D., Santen L., Schadschneider A., “Statistical Physics of Vehicular Traffic and Some Related Systems”, Phys. Rep. **329** (2000) 199

[12] Chowdhury D., Schadschneider A., "Self-Organization of Traffic Jams in Cities: Effects of Stochastic Dynamics and Signal Periods", Phys. Rev. E **59** (1999) R1311

[13] Cremer M., Ludwig J., "A Fast Simulation Model for Traffic Flow on the Basis of Boolean Operations", Math. and Comp. in Simul. **28** (1986) 297

[14] Del Castillo J. M., Pintado P., Benitez F. G., "The Reaction Time of Drivers and the Stability of Traffic Flow", Transpn. Res. B **28B** (1994) 35

[15] Eisenblätter B., Santen L., Schadschneider A., Schreckenberg M., "Jamming Transition in a Cellular Automaton Model for Traffic Flow", Phys. Rev. E **57** (1998) 1309

[16] Esser J., Schreckenberg M., "City Congestion: Can Online-Simulations Solve the Problem?", in *Workshop on Traffic and Granular Flow '97*, edited by Schreckenberg M. and Wolf D. E. (Springer-Verlag, Singapore, 1998)

[17] Gazis D. C., Herman R., Potts R. B., "Car-Following Theory of Steady-State Traffic Flow", Op. Res. **7** (1959) 499

[18] Gerlough D. L., Huber M. J., *Traffic Flow Theory — A Monograph*, (Transportation Research Board, National Research Council, Special Report 165, Washington D.C., 1975)

[19] Hall F. L., Allen B. L., Gunter M. A., "Empirical Analysis of Freeway Flow-Density Relationships", Transpn. Res. A **20A** (1986) 197

[20] Helbing D., "Traffic and Related Self-Driven Many-Particle Systems", Rev. Mod. Phys. **73** (2001) 1067

[21] Helbing D., Greiner A., "Modeling and Simulation of Multilane Traffic Flow", Phys. Rev. E **55** (1997) 5498

[22] Helbing D., Schreckenberg M., "Cellular Automata Simulating Experimental Properties of Traffic Flow", Phys. Rev. E **59** (1999) R2505

[23] Hämäläinen A., *Traffic Simulations with Cellular Automata*, Helsinki University of Technology, Department of Engineering Physics and Mathematics, licentiate thesis (Otaniemi 2001)

[24] Jiang R., Wu Q-S., "Cellular Automata Models for Synchronized Traffic Flow", J. Phys. A: Math. Gen. **36** (2003) 381

[25] Jokinen R., Kosonen I. *Ajantasainen kaupunkiliikenneverkon mallinnus ja toimintakyvyn mittaus — Esiselvitys*, AINO-julkaisuja 2005, Liikenne- ja viestintäministeriö (2005, in Finnish)

[26] Kerner B. S., "Experimental Features of Self-Organization in Traffic Flow", Phys. Rev. Lett. **81** (1998) 3797

[27] Kerner B. S., "Three-Phase Traffic Theory and Highway Capacity", cond-mat/0211684 (2003)

[28] Kerner B. S., "Traffic Flow: Experiment and Theory", in *Workshop on Traffic and Granular Flow '97*, edited by Schreckenberg M. and Wolf D. E. (Springer-Verlag, Singapore, 1998)

[29] Kerner B. S., Klenov S. L., Wolf D. E., "Cellular Automata Approach to Three-Phase Traffic Theory", J. Phys. A: Math. Gen. **35** (2002) 9971

[30] Kerner B. S., Konhäuser P., "Cluster Effect in Initially Homogeneous Traffic Flow", Phys. Rev. E **48** (1993) R2335

[31] Kerner B. S., Konhäuser P., "Structure and Parameters of Clusters in Traffic Flow", Phys. Rev. E **50** (1994) 54

[32] Kerner B. S., Rehborn H., "Experimental Properties of Complexity in Traffic Flow", Phys. Rev. E **53** (1996) R4275

[33] Kerner B. S., Rehborn H., "Experimental Properties of Phase Transitions in Traffic Flow", Phys. Rev. Lett. **79** (1997) 4030

[34] Kernighan B. W., Ritchie D. M., Huru E., *C-Ohjelmointi* (Edita, IT Press, Helsinki, 2000, in Finnish)

[35] Knospe W., Santen L., Schadschneider A., Schreckenberg M., "An Empirical Test for Cellular Automaton Models of Traffic Flow", cond-mat/0404710v1 (2004)

[36] Knospe W., Santen L., Schadschneider A., Schreckenberg M., "Towards a Realistic Microscopic Description of Highway Traffic", J. Phys. A: Math. Gen **33** (2000) L477

[37] Kolomeisky A. B., Schütz G. M., Kolomeisky E. B., Straley J. S., "Phase diagram of one-dimensional driven lattice gases with open boundaries", J. Phys. A: Math. Gen **31** (1998) 6911

[38] Kosonen I., *HUTSIM — Urban Traffic Simulation and Control Model: Principles and Applications*, thesis, Helsinki University of Technology, Transportation Engineering, Publication 100 (Espoo, 1999)

[39] Kosonen I. "Multi-Agent Fuzzy Signal Control Based on Real-Time Simulation", Transpn. Res. C **11C** (2003) 389

[40] Kosonen I., Kulmala R., Hautala R., *DigiTraffic — Liikenteen mallinnus- ja palvelujärjestelmä*, FITS-julkaisuja 30/2004, Liikenne- ja viestintäministeriö (2004, in Finnish)

[41] Krug J., Spohn H., "Universality Classes for Deterministic Surface Growth", Phys. Rev. A **38** (1988) 4271

[42] Lárraga M. E., del Río J. A., Schadschneider A., "Platoon Formation in a Traffic Model with Variable Anticipation", cond-mat/0306531 (2003)

[43] Lee H. K., Barlovic R., Schreckenberg M., Kim D., "Mechanical Restriction versus Human Overreaction Triggering Congested Traffic States", cond-mat/0404315 (2004)

[44] Leutzbach W., *Introduction to the Theory of Traffic Flow* (Springer Verlag, Berlin, 1988)

[45] Luttinen R. T., *Statistical Analysis of Vehicle Time Headways*, thesis, Helsinki University of Technology, Transportation Engineering, Publication 87 (Otaniemi 1996)

[46] Mattila H., *Linkkikohtaisen liikennetilanteen ajantasainen arviointi*, master's thesis, Helsinki University of Technology, Transportation Engineering (Espoo, 2004, in Finnish)

[47] Nagel K., "Particle Hopping Models and Traffic Flow Theory", Phys. Rev. E **53** (1996) 4655

[48] Nagel K., Paczuski M., "Emergent Traffic Jams", Phys. Rev. E **51** (1995) 2909

[49] Nagel K., Schreckenberg M., "A Cellular Automaton Model for Freeway Traffic", J. Phys. I France **2** (1992) 2221

[50] Nagel K., Wolf D. E., Wagner P., Simon P., "Two-Lane Traffic Rules for Cellular Automata: A Systematic Approach", Phys. Rev. E **58** (1998) 1425

[51] Neubert L., Santen L., Schadschneider A., Schreckenberg M., "Single-Vehicle Data of Highway Traffic: A Statistical Analysis", Phys. Rev. E **60** (1999) 6480

[52] Ousterhout J. K., *Tcl and the Tk Toolkit* (Addison-Wesley, Reading, MA, 1994)

[53] Pahlman S., Kosonen I. *Ajantasaisen liikennetilannemallin tuottaminen floating car datan avulla — Esiselvitys*, AINO-julkaisuja 12/2005, Liikenne- ja viestintäministeriö (2005, in Finnish)

[54] Paveri-Fontana S. L., "On Boltzmann-Like Treatments for Traffic Flow: a Critical Review of the Basic Model and an Alternative Proposal for Dilute Traffic Analysis", Transpn. Res. **9** (1975) 225

[55] Pipes L. A., "An Operational Analysis of Traffic Dynamics", J. Appl. Phys. **24** (1953) 274

[56] Prigogine I., Andrews F. C., "A Boltzmann-Like Approach for Traffic Flow", Op. Res. **8** (1960) 789

[57] Pursula M., “Simulation of Traffic Systems — An Overview”, J. of Geographic Information and Decision Analysis, **3** (1999) 1 (electronic journal in the Internet)

[58] Rickert M., Nagel K., Schreckenberg M., Latour A., “Two Lane Traffic Simulations using Cellular Automata”, Physica A **231** (1996) 534

[59] Rothman D. H., Zaleski S., “Lattice-Gas Models of Phase Separation: Interfaces, Phase Transitions and Multiphase Flow“, Rev. Mod. Phys. **66** (1994) 1417

[60] Simon P. M., Gutowitz H. A., “Cellular Automaton Model for Bidirectional Traffic”, Phys. Rev. E **57** (1998) 2441

[61] Schadschneider A., “Analytical Approaches to Cellular Automata for Traffic Flow: Approximations and Exact Solutions”, in *Workshop on Traffic and Granular Flow ’97*, edited by Schreckenberg M. and Wolf D. E. (Springer-Verlag, Singapore, 1998)

[62] Schadschneider A., “Statistical Physics of Traffic Flow”, Physica A **285** (2000) 101

[63] Schadschneider A., “The Nagel-Schreckenberg Model Revisited”, Eur. Phys. J. B **10** (1999) 573

[64] Schadschneider A., Schreckenberg M., “Car-Oriented Mean-Field Theory for Traffic Flow Models”, J. Phys. A: Math. Gen. **30** (1997) L69

[65] Schadschneider A., Schreckenberg M., “Garden of Eden States in Traffic Models”, J. Phys. A: Math. Gen. **31** (1998) L225

[66] Schreckenberg M., Schadschneider A., Nagel K., Ito N., “Discrete Stochastic Models for Traffic Flow”, Phys. Rev. E **51** (1995) 2939

[67] Schütz G., Domany J., “Phase transitions in an exactly soluble one-dimensional exclusion process”, J. Stat. Phys. **72** (1993) 277

[68] Schönhof M., Helbing D., “Empirical Features of Congested Traffic States and Their Implications for Traffic Modeling”, cond-mat/0408138 (2004)

[69] Treiber M., Helbing D., “Explanation of Observed Features of Self-Organization in Traffic Flow”, preprint/cond-mat/9901239

[70] Treiber M., Hennecke A., Helbing D., “Congested Traffic States in Empirical Observations and Microscopic Simulations”, Phys. Rev. E **62** (2000) 1805

[71] Welch B. B., *Practical Programming in Tcl and Tk* (Prentice-Hall, Upper Sadle River, NJ, 1995)

[72] Wolfram S., “Statistical Mechanics of Cellular Automata”, Rev. Mod. Phys. **55** (1983) 601

[73] Wolfram S., “Twenty Problems in the Theory of Cellular Automata”, Physica Scripta **T9** (1985) 170

[74] Wolfram S., “Universality and Complexity in Cellular Automata”, Physica **10D** (1984)

[75] WWW-pages http://www.aino.info/indexe.html

[76] WWW-pages http://users.tkk.fi/∼ikosonen/DIGITRAFFIC/FIN/ (partly in Finnish)

[77] WWW-pages http://www.vtt.fi/rte/projects/fits/

Appendix A

File formats for input and output files

INPUT/OUTPUT FILE MAP FOR THE CA SIMULATOR:

cafiles.dat:

```
generalpar    genpar.dat       % general parameters for the run
pipelen       plen.prn         % lengths of pipes
hutdata       lvhi/ds3.dat     % vehicle arrivals and signal changes
turndire      turn.prn         % turning percentages at crossings
destin        desti.prn        % origin/destination map
selectpipes   sele.prn         % pipes after crossings
followpipes   folp.prn         % potential followers to a pipe
yieldpipes    yield.prn        % pipes with yielding duty
tlights       ptlight.prn      % signal data for pipes
deflects      deflp.prn        % deflecting data for pipes
speedlim      spedl.prn        % pipes with speed limits
detectors     det.dat          % detector position data
detfile       detfile/d1.dat   % file for counting detector crossing matrices
credetectors  credet.prn       % data for extra vehicle input pipes
quepipext     qpext.prn        % follower pipes for queue length counting
cadetsig      cadet/dx.dat     % file for detector crossings
vehleave      lt4.dat          % data for vehicles exiting model area
```

CA SIMULATOR RUN FILE CONTENTS:

generalpar:

```
5        % rule          – model rule for the run
3        % VMAX          – maximum velocity (cells/time step)
14       % convlcgap     – gap for probing lane-changing
0.3      % lnchprob      – probability for lane-changing
0.100    % brprob        – braking probability
0.280    % spprob        – braking probability after stop (VDR)
0        % quelenon      – flag for queue length recording
0        % cadetsigon    – flag for detector crossing recording
0        % vehleaveon    – flag for vehicle exit data recording
0        % queuemaxon    – flag for max queue length recording
```

pipelen:

```
283                      % number of pipes
  0  14   1  5  9        % 1. column: pipe
  1  14   0  5  9        % 2. column: pipe length
  2   6   3  0  5        % 3. column: potential parallel pipe (>–1)
  3   6   2  0  5        % 4. column: first parallel cell
  4   4  –1  0  0        % 5. column: last parallel cell
  5  20   1  0  0
  6  14  –1  0  0
  7  10   8  0  9
  8  10   7  0  9
  9   5  –1  0  0
 10  14  –1  0  0
 11  12  –1  0  0
      .
      .
      .
```

turndire:

```
46  0  0   0  .  .   0   0      % 1. column: pipe
47  7  3   8  .  .  18   7      % 2. column: number of turning dir.
48  7  3   6  .  .  18  12      % 3. column: 1. turning dir.
49  0  0   0  .  .   0   0      % 4. column: 1. turning %
50  0  0   0  .  .   0   0      % 5. column: 2. turning dir.
51  4  1  47  .  .   0   0      % 6. column: 2. turning %
    .                           % .
    .                           % .
    .                           % 15. column: 7. turning dir.
    .                           % 16. column: 7. turning %
```

destin:

```
 0  1  0  2  .  .  .  0  0      % 1. column: pipe
 1  2  0  1  .  .  .  0  0      % 2. column: access to destination 1
 2  1  0  0  .  .  .  0  0      % 3. column: access to destination 2
 3  2  0  0  .  .  .  0  0      % .
 4  0  0  1  .  .  .  0  0      % .
 5  1  0  0  .  .  .  0  0      % . (access codes:
 6  0  0  0  .  .  .  0  0      % .   0 – no access
 7  0  0  1  .  .  .  0  0      % .   1 – direct access
 8  1  0  0  .  .  .  0  0      % .   2 – access by one lane-change
 9  0  0  0  .  .  .  0  0      % .   9 – access by two lane-ch.)
    .                           % .
    .                           % .
15  0  0  1  .  .  .  0  0      % 20. column: access to dest. 19
16  1  1  0  .  .  .  0  0      % 21. column: access to dest. 20
17  0  0  0  .  .  .  1  0
18  0  0  1  .  .  .  0  0
19  0  0  0  .  .  .  0  1
20  0  0  2  .  .  .  0  0
    .
    .
    .
```

selectpipes:

```
69          % select pipe
79
80
167
202
232
.
.
.
```

followpipes:

```
0   1   2   –1   –1          % 1. column: pipe
1   2   3    4   –1          % 2. column: number of follower pipes
2   1   5   –1   –1          % 3. column: 1. follower pipe (> –1)
3   1   6   –1   –1          % 4. column: 2. follower pipe (> –1)
4   1   7   –1   –1          % 5. column: 3. follower pipe (> –1)
.
.
.
```

yieldpipes:

```
197   0   169   4   20       % 1. column: duty to yield pipe
      .                      % 2. column: stay cell on pipe
      .                      % 3. column: right of way pipe
      .                      % 4. column: max cell to cause yielding
      .                      %            (with minus sign for min
      .                      %             cell from end)
      .                      % 5. column: destination for yielding vehicle
```

tlights:

```
1    1        % start of phase for queue lengths: area+group
1    1        % start of phase for detector cross. matrices: area+group
0  999  0     % 1. column: pipe
1    1  2     % 2. column: signal area (<999)
2    1  2     % 3. column: signal group
3  999  0
4  999  0
     .
     .
     .
```

deflects:

```
20   23          % 1. column: pipe
20  258          % 2. column: deflecting follower pipe
26   27
36   35
 .
 .
 .
```

speedlim:

```
 8  2           % 1. column: pipe
11  2           % 2. column: speed limit (cells/time step)
26  2
33  2
 .
 .
 .
```

detectors:

```
  6   4   3   81   0   1      % 1. column: pipe
  7   4   3   82   0   1      % 2. column: downstream end of cell
  8   4   3   71   0   0      % 3. column: distance for detection calc.
  9   4   3   72   0   0      % 4. column: identifier
 12   4   3   73   0   0      % 5. column: offset
 15   4   3   74   0   0      % 6. column: detector type
      .
      .
      .
```

credetectors:

```
  51    21          % 1. column: pipe
  52    22          % 2. column: create detector identifier
 129   111
   .
   .
   .
```

quepipext:

```
 0   –1   –1   –1   –1   –1      % 1. column: pipe
 1    0   –1   –1   –1   –1      % 2. column: number of external pipes
 2   –1   –1   –1   –1   –1      %          (–1: no signal for the pipe)
 3   –1   –1   –1   –1   –1      % 3. column: 1. external pipe
 4   –1   –1   –1   –1   –1      % 4. column: 2. external pipe
 5    1    2   –1   –1   –1      % 5. column: 3. external pipe
 5   –1   –1   –1   –1   –1      % 5. column: 4. external pipe
      .
      .
      .
```

MAP FOR AN INPUT FILE FROM HUTSIM:

detsigfile:

```
 110  4   1  1  0        % 1. column: time (1/100 s)
 110  4   2  1  0        % 2. column: record type —
 110  4   1  2  0        %             4: signal record
 110  4   2  2  0        %             7: vehicle record
 270  7  51  1           % 3. column: for 4: signal group
 340  7  21  1           %            for 7: arrival detector code
 610  7  61  1           % 4. column: for 4: signal area
 910  7  12  1           %            for 7: vehicle type (=1)
1210  4   1  2  2        % 5. column: for 4: signal value
1210  4   2  2  2
1400  7  22  1
      .
      .
      .
```

MAP FOR OUTPUT FILES FROM CA:

vehleave:

```
50  51  6   2        % 1. column: exit time (s)
52  51  6  19        % 2. column: origin
55  61  5   6        % 3. column: destination
57  61  5  17        % 4. column: arrival time (s)
60  61  5  22
64  61  5  33
65  51  6  33
      .
      .
      .
```

cadetsig:

```
40  61   6  81        % 1. column: detection time (s)
43  61  22  81        % 2. column: origin
60  31  16  71        % 3. column: arrival time (s)
62  31  25  71        % 4. column: detector
64  31  34  71
66  31  41  71
77  22  14  82
78  21   3  81
79  22  20  82
     .
     .
     .
```

detfile:

```
 0  0  695  800  905  1010  1115    % 1. row, 2. col. → : start
 1  0    1    1    0     1     1    %      time (s)
 2  1    0    0    1     0     0    % 2. row, 1. col.: time step (s),
 3  0    1    0    0     1     0    %      2. col. → : vehicle
 4  1    0    1    1     0     1    %      passed stop line
 5  0    1    0    0     1     0
 6  1    0    1    0     0     1
 7  0    1    0    1     0     0
 8  1    0    1    0     0     1
 9  0    0    0    1     1     0
10  1    1    1    0     0     1
11  0    0    0    0     1     0
12  1    1    0    1     0     1
13  0    0    0    0     1     0
14  0    1    1    1     0     1
    .
    .
    .
```

Appendix B

The simulations performed

The next pages contain parameter and file information associated with the performed simulations runs and results of the runs in graph form. The descriptions of the runs presented in table form are followed by tables connecting these runs to the figures. HUTSIM is often abbreviated as HS in the figures. — Figs. B.1–B.2 belong to the testing of the influence of randomness in the CA runs, and they are not part of the actual runs.

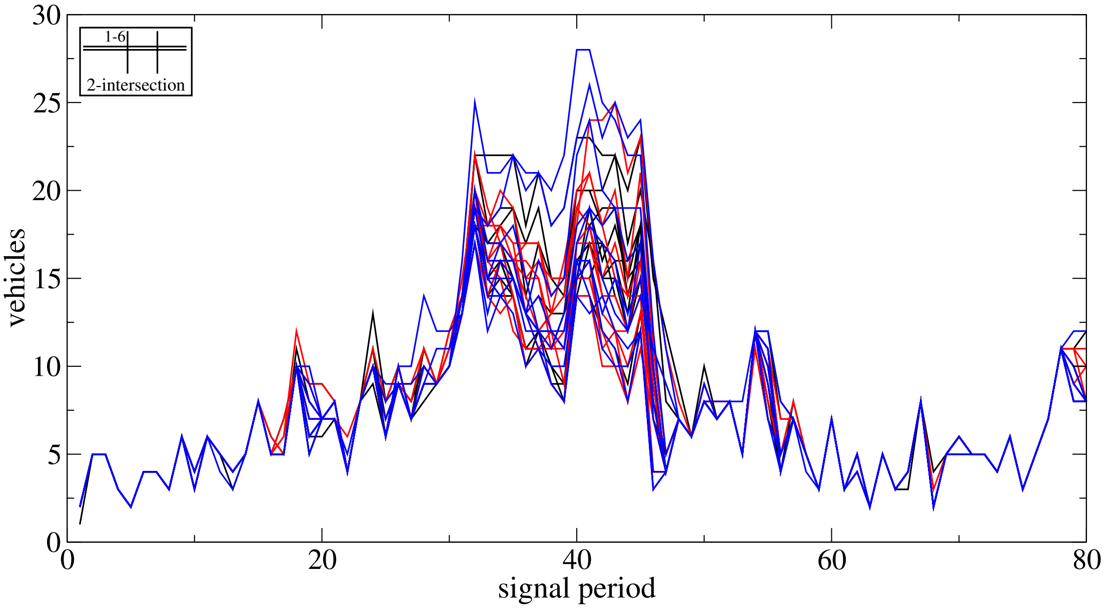


Figure B.1: The results of three sets of ten CA runs for queue lengths at signal post 1-6, turn rate set 1.

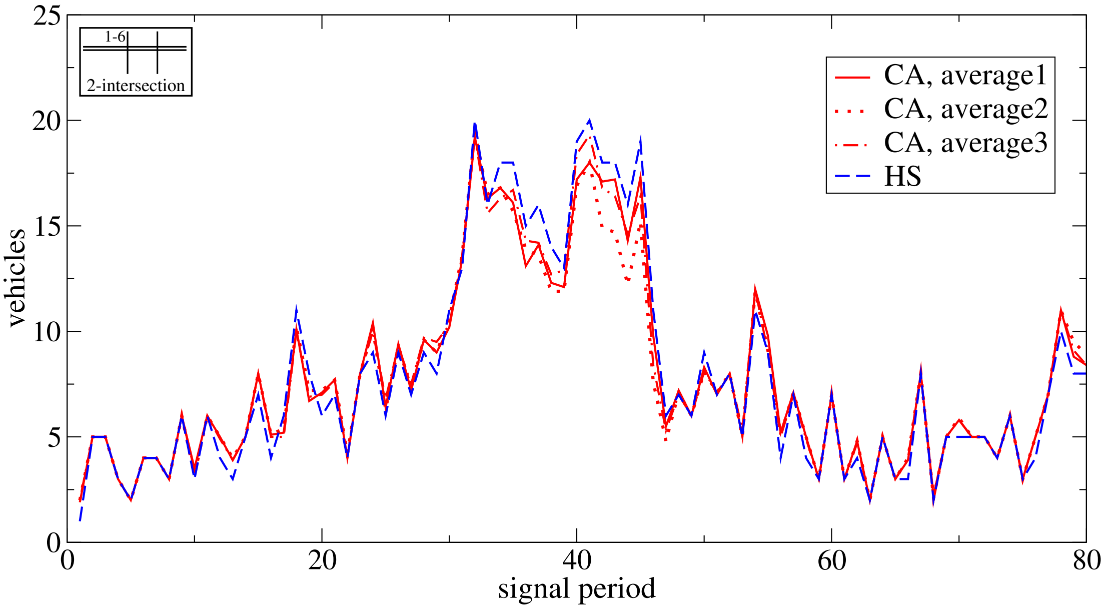


Figure B.2: The average queue lengths of the three sets of ten CA runs at signal post 1-6, turn rate set 1.

CA run number	HUTSIM configfile (.cnf)	HUTSIM delayfile (.del)	HUTSIM detsigfile (.dat)	CA turn rate set (.prn)	CA seed	CA logfile (.log)	CA genparfile (.dat)
1	cr21	hs21	ds21	cr2tu1	23356	rq2_1	genpar1
2	cr21	hs21	ds21	cr2tu1	2332	rq2_11	— “ —
3	cr21	hs21	ds21	cr2tu1	10112	rq2_12	— “ —
4	cr21	hs21	ds21	cr2tu1	9930	rq2_13	— “ —
5	cr21	hs21	ds21	cr2tu1	381	rq2_14	— “ —
6	cr21	hs21	ds21	cr2tu1	3354	rq2_15	— “ —
7	cr21	hs21	ds21	cr2tu1	12001	rq2_16	— “ —
8	cr21	hs21	ds21	cr2tu1	1020	rq2_17	— “ —
9	cr21	hs21	ds21	cr2tu1	25460	rq2_18	— “ —
10	cr21	hs21	ds21	cr2tu1	19500	rq2_19	— “ —
11	cr22	hs22	ds22	cr2tu2	23356	rq2_2	— “ —
12	cr22	hs22	ds22	cr2tu2	2332	rq2_21	— “ —
13	cr22	hs22	ds22	cr2tu2	10112	rq2_22	— “ —
14	cr22	hs22	ds22	cr2tu2	9930	rq2_23	— “ —
15	cr22	hs22	ds22	cr2tu2	381	rq2_24	— “ —
16	cr22	hs22	ds22	cr2tu2	3354	rq2_25	— “ —
17	cr22	hs22	ds22	cr2tu2	12001	rq2_26	— “ —
18	cr22	hs22	ds22	cr2tu2	1020	rq2_27	— “ —
19	cr22	hs22	ds22	cr2tu2	25460	rq2_28	— “ —
20	cr22	hs22	ds22	cr2tu2	19500	rq2_29	— “ —
21	cr23	hs23	ds23	cr2tu3	23356	rq2_3	— “ —
22	cr23	hs23	ds23	cr2tu3	2332	rq2_31	— “ —
23	cr23	hs23	ds23	cr2tu3	10112	rq2_32	— “ —
24	cr23	hs23	ds23	cr2tu3	9930	rq2_33	— “ —
25	cr23	hs23	ds23	cr2tu3	381	rq2_34	— “ —
26	cr23	hs23	ds23	cr2tu3	3354	rq2_35	— “ —
27	cr23	hs23	ds23	cr2tu3	12001	rq2_36	— “ —
28	cr23	hs23	ds23	cr2tu3	1020	rq2_37	— “ —
29	cr23	hs23	ds23	cr2tu3	25460	rq2_38	— “ —
30	cr23	hs23	ds23	cr2tu3	19500	rq2_39	— “ —
31	cr24	hs24	ds24	cr2tu4	23356	rq2_4	— “ —
32	cr24	hs24	ds24	cr2tu4	2332	rq2_41	— “ —
33	cr24	hs24	ds24	cr2tu4	10112	rq2_42	— “ —
34	cr24	hs24	ds24	cr2tu4	9930	rq2_43	— “ —
35	cr24	hs24	ds24	cr2tu4	381	rq2_44	— “ —
36	cr24	hs24	ds24	cr2tu4	3354	rq2_45	— “ —
37	cr24	hs24	ds24	cr2tu4	12001	rq2_46	— “ —
38	cr24	hs24	ds24	cr2tu4	1020	rq2_47	— “ —
39	cr24	hs24	ds24	cr2tu4	25460	rq2_48	— “ —
40	cr24	hs24	ds24	cr2tu4	19500	rq2_49	— “ —

Table B.1: Key for HUTSIM and CA runs, the 2-intersection case, queue lengths.

queue(s) at post	turn rate set	flow profile	figure	CA run numbers
1-1	1	origin 1	B.3	1 – 10
1-1	2	origin 1	B.4	11 – 20
1-1	3	origin 1	B.5	21 – 30
1-1	4	origin 1	B.6	31 – 40
1-1	1	origin 1	B.7	1 – 10
1-1	2	origin 1	B.8	11 – 20
1-1	3	origin 1	B.9	21 – 30
1-1	4	origin 1	B.10	31 – 40
1-6	1	origin 4	B.11	1 – 10
1-6	2	origin 4	B.12	11 – 20
1-6	3	origin 4	B.13	21 – 30
1-6	4	origin 4	B.14	31 – 40
1-6	1	origin 4	B.15	1 – 10
1-6	2	origin 4	B.16	11 – 20
1-6	3	origin 4	B.17	21 – 30
1-6	4	origin 4	B.18	31 – 40
2-1	1	origin 2	B.19	1 – 10
2-1	2	origin 2	B.20	11 – 20
2-1	3	origin 2	B.21	21 – 30
2-1	4	origin 2	B.22	31 – 40
2-1	1	origin 2	B.23	1 – 10
2-1	2	origin 2	B.24	11 – 20
2-1	3	origin 2	B.25	21 – 30
2-1	4	origin 2	B.26	31 – 40

Table B.2: Key for figures and CA runs, the 2-intersection case, queue lengths.

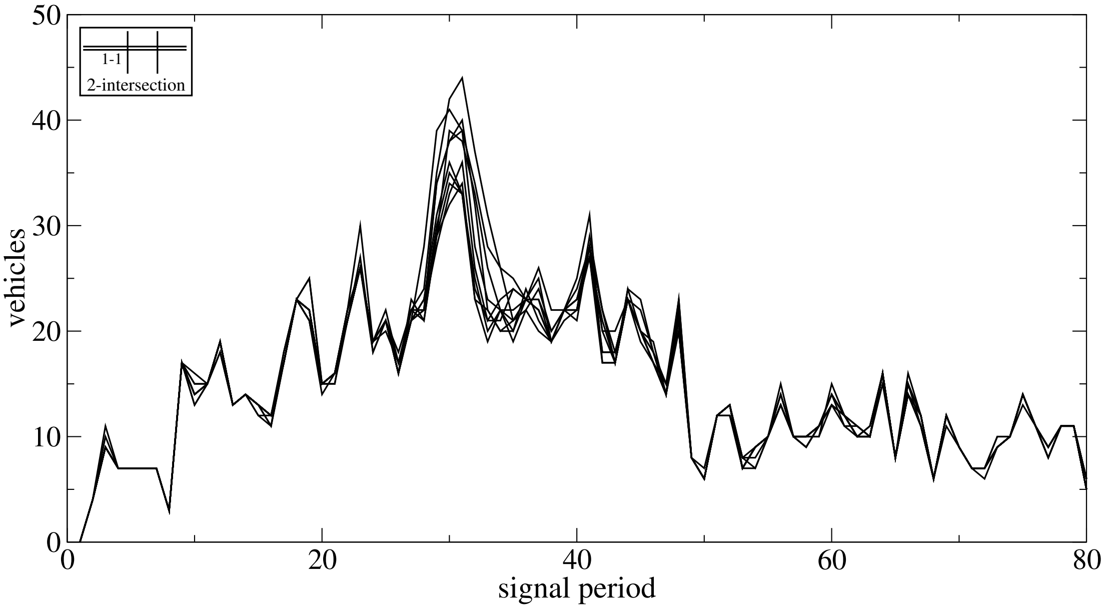


Figure B.3: CA queue lengths for the 2-intersection case at 1-1; 10 different seed numbers, turn rate set 1.

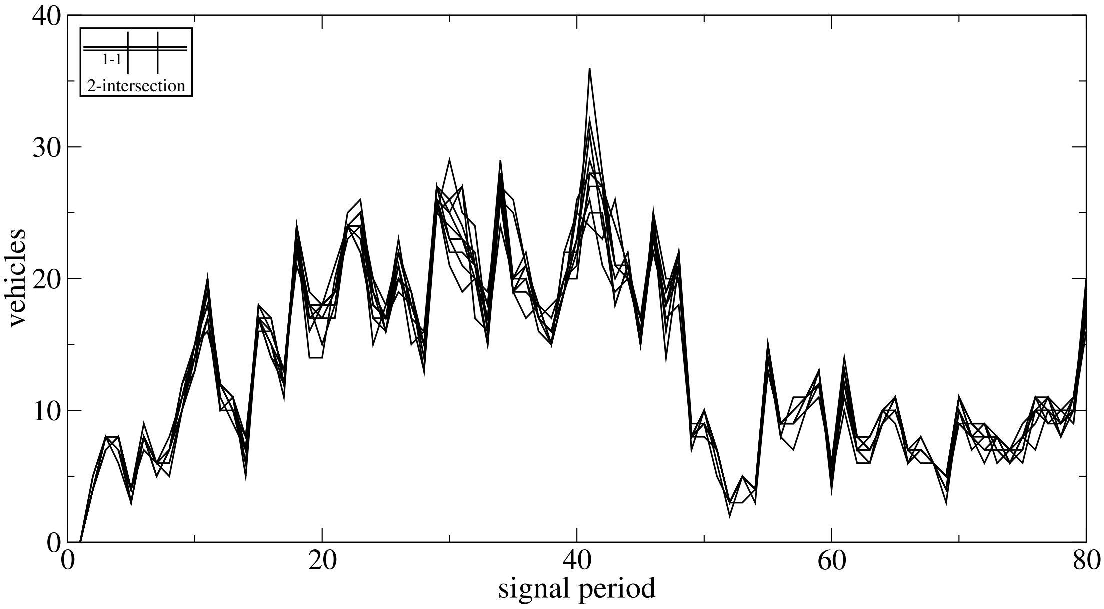


Figure B.4: CA queue lengths for the 2-intersection case at 1-1; 10 different seed numbers, turn rate set 2.

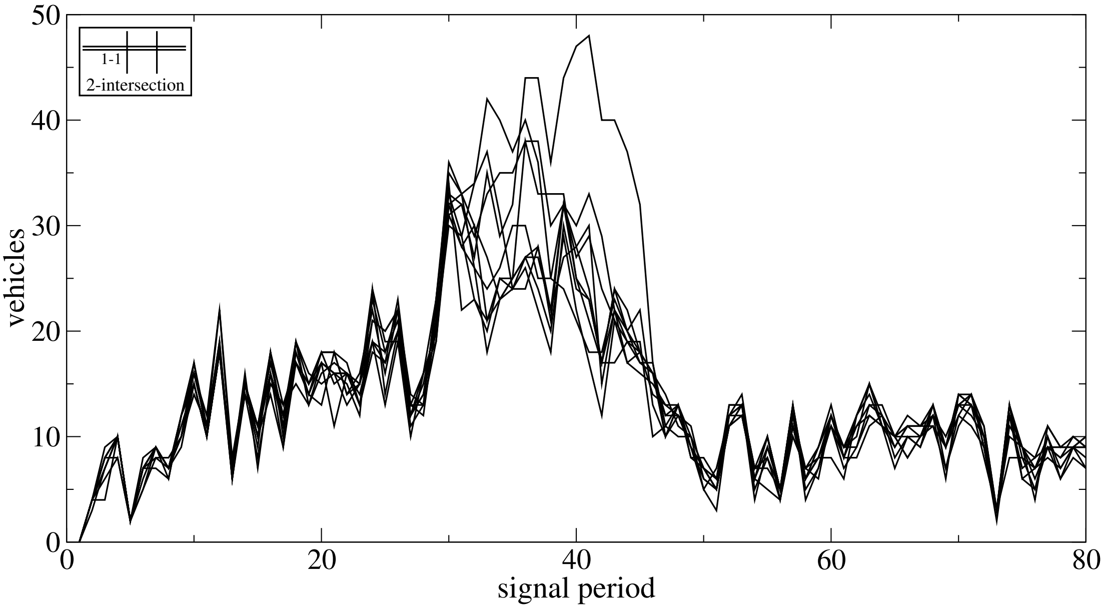


Figure B.5: CA queue lengths for the 2-intersection case at 1-1; 10 different seed numbers, turn rate set 3.

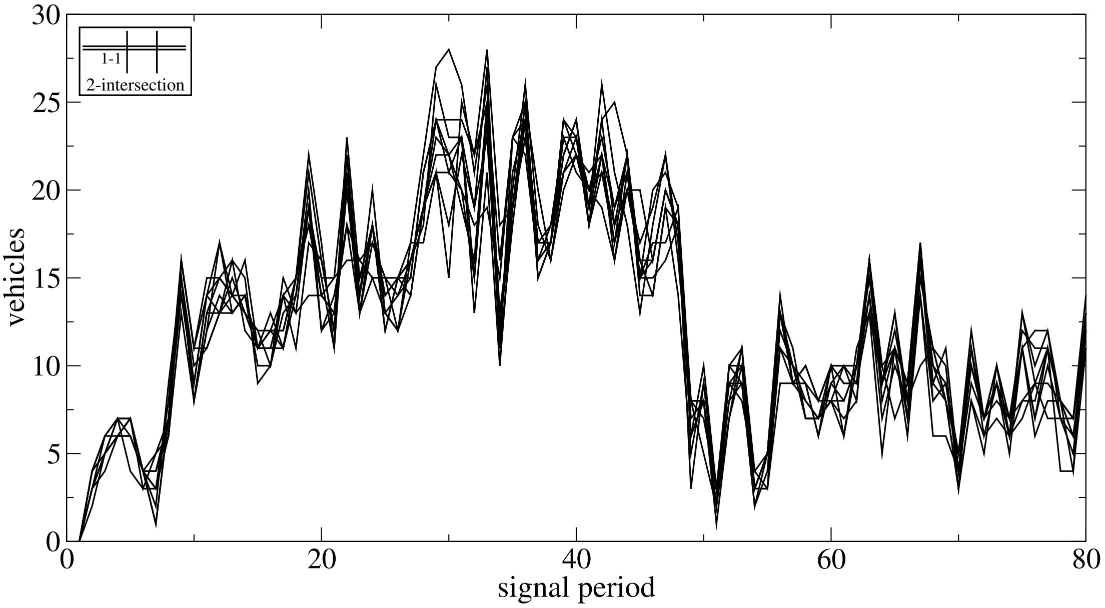


Figure B.6: CA queue lengths for the 2-intersection case at 1-1; 10 different seed numbers, turn rate set 4.

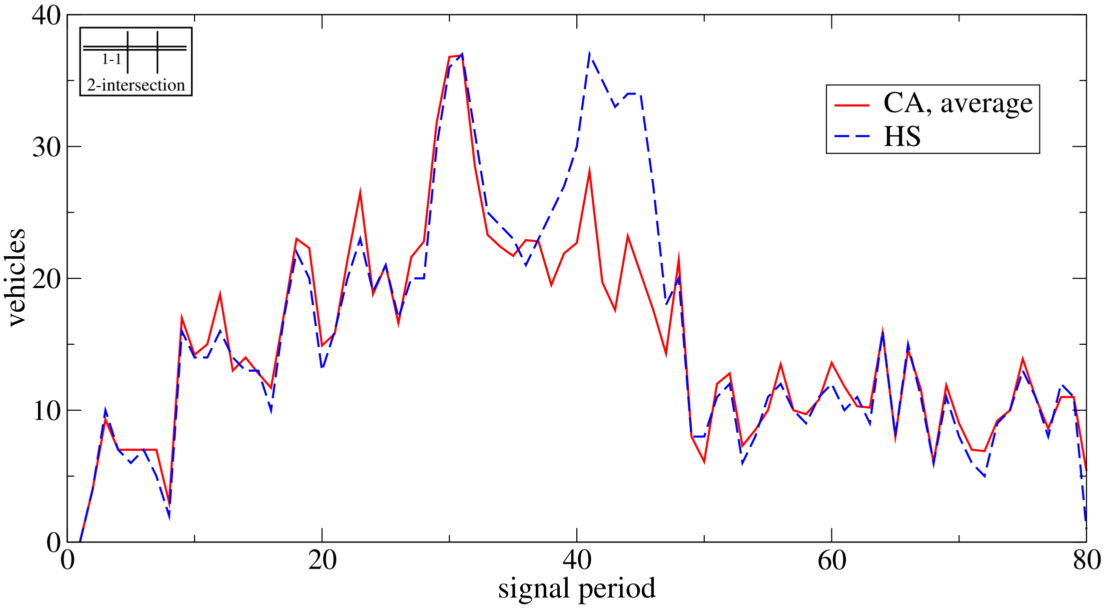


Figure B.7: Queue lengths for the 2-intersection case at 1-1; CA average of 10 different seed numbers, turn rate set 1.

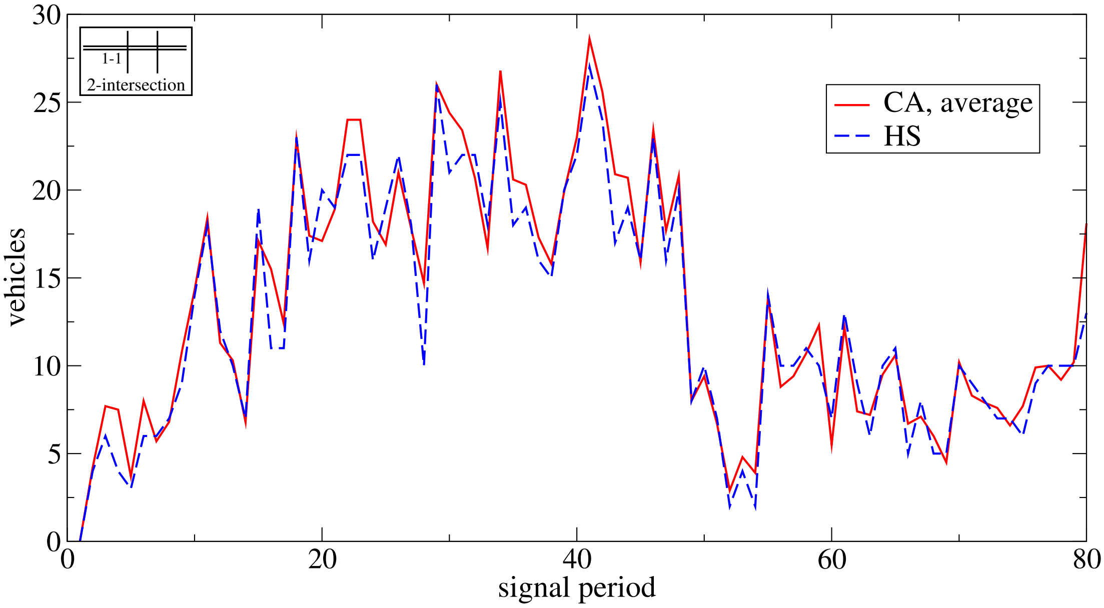


Figure B.8: Queue lengths for the 2-intersection case at 1-1; CA average of 10 different seed numbers, turn rate set 2.

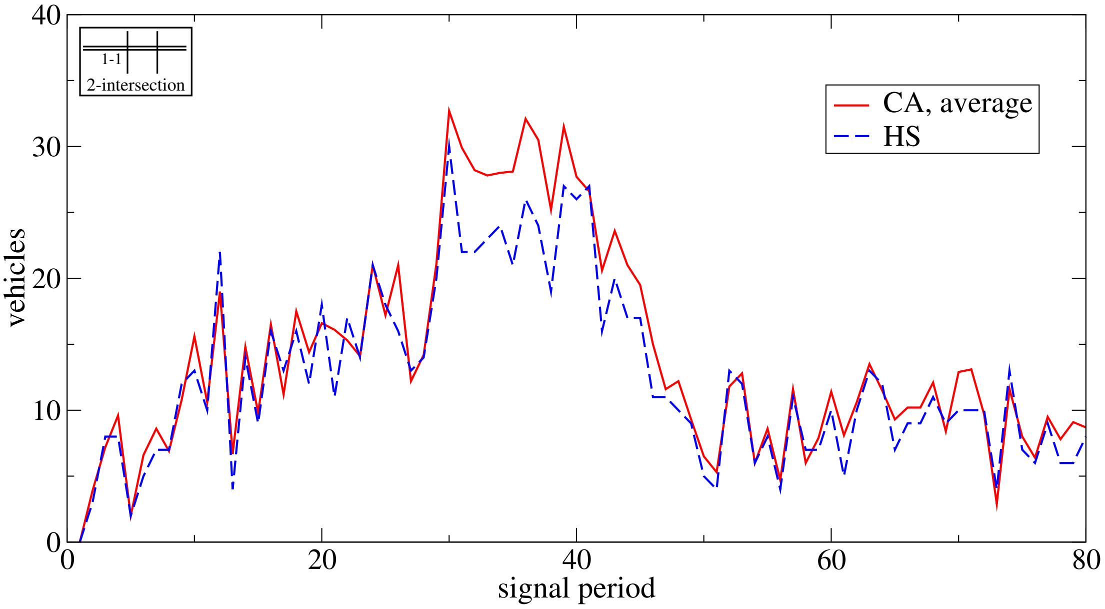


Figure B.9: Queue lengths for the 2-intersection case at 1-1; CA average of 10 different seed numbers, turn rate set 3.

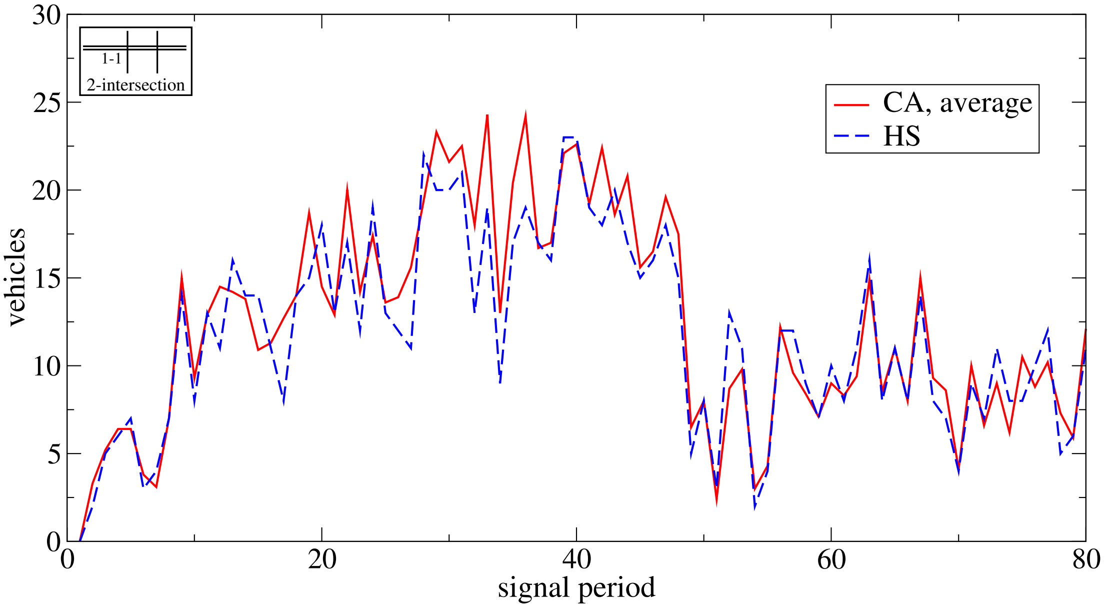


Figure B.10: Queue lengths for the 2-intersection case at 1-1; CA average of 10 different seed numbers, turn rate set 4.

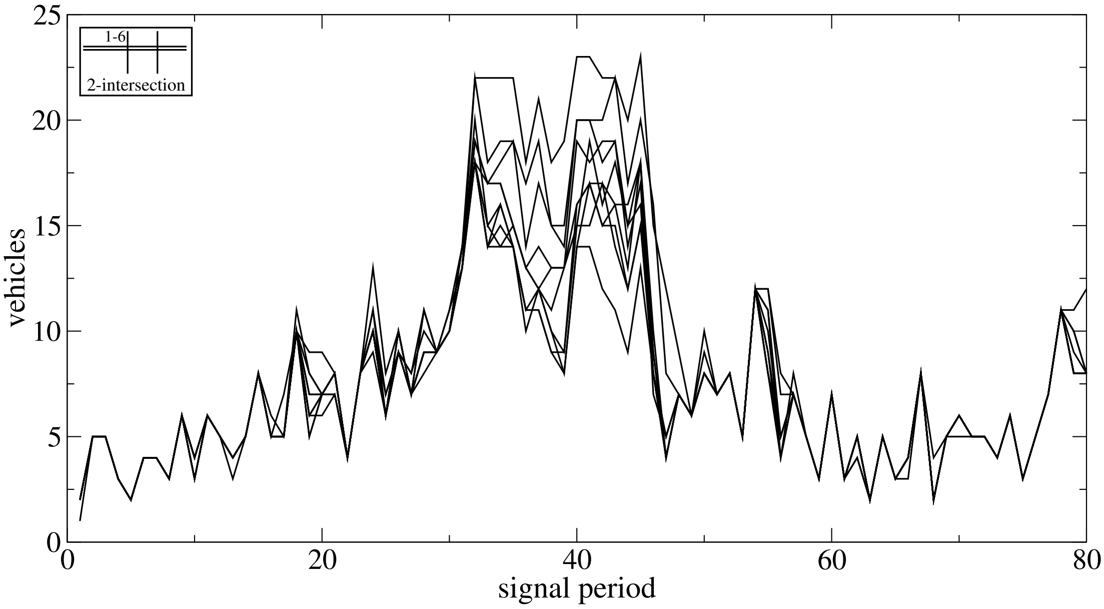


Figure B.11: CA queue lengths for the 2-intersection case at 1-6; 10 different seed numbers, turn rate set 1.

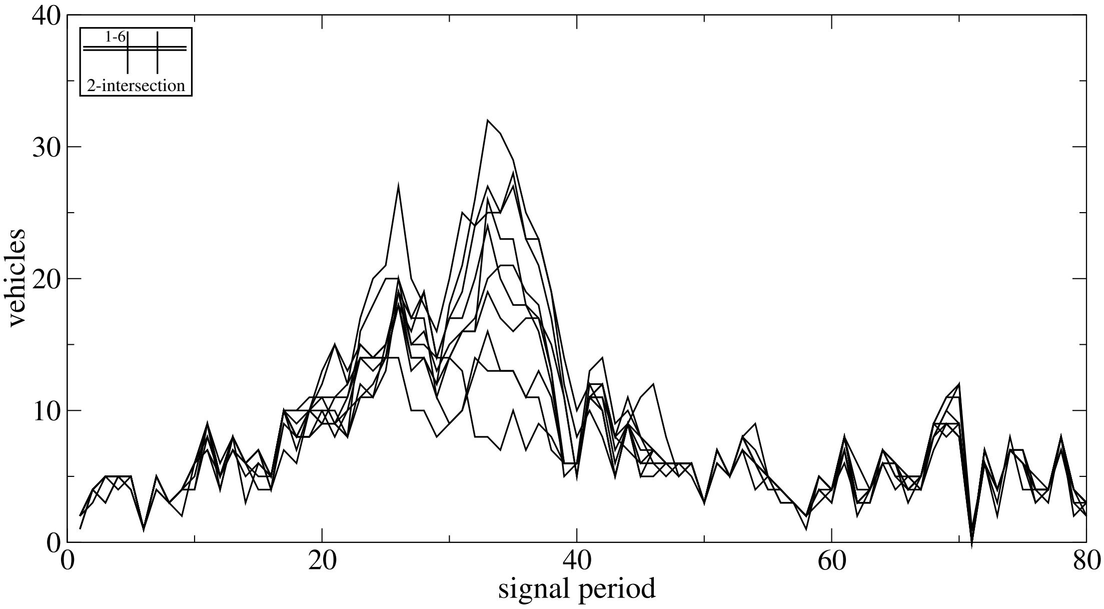


Figure B.12: CA queue lengths for the 2-intersection case at 1-6; 10 different seed numbers, turn rate set 2.

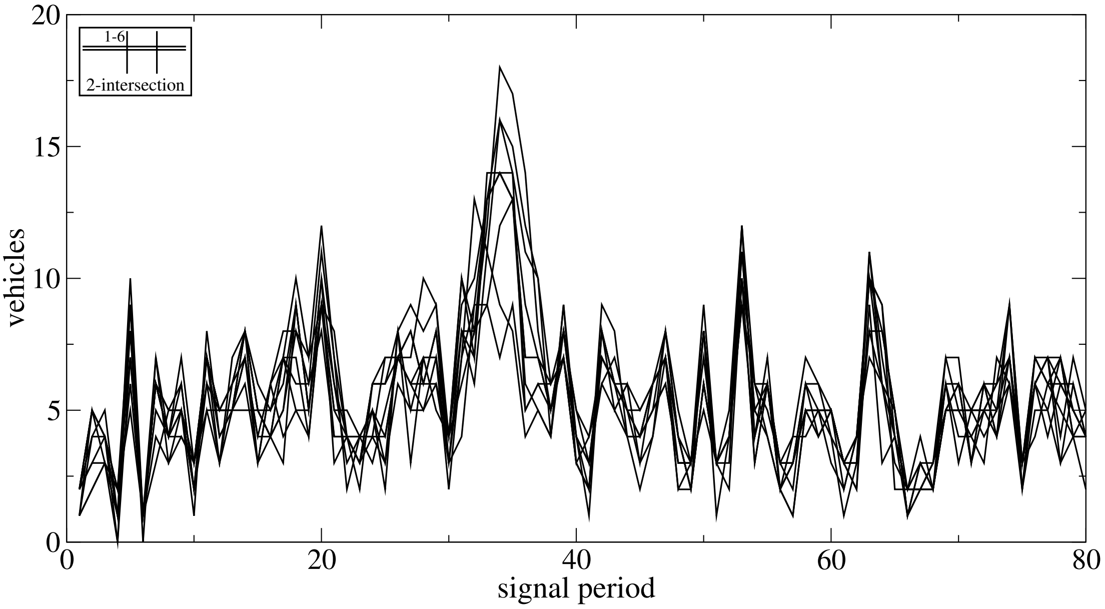


Figure B.13: CA queue lengths for the 2-intersection case at 1-6; 10 different seed numbers, turn rate set 3.

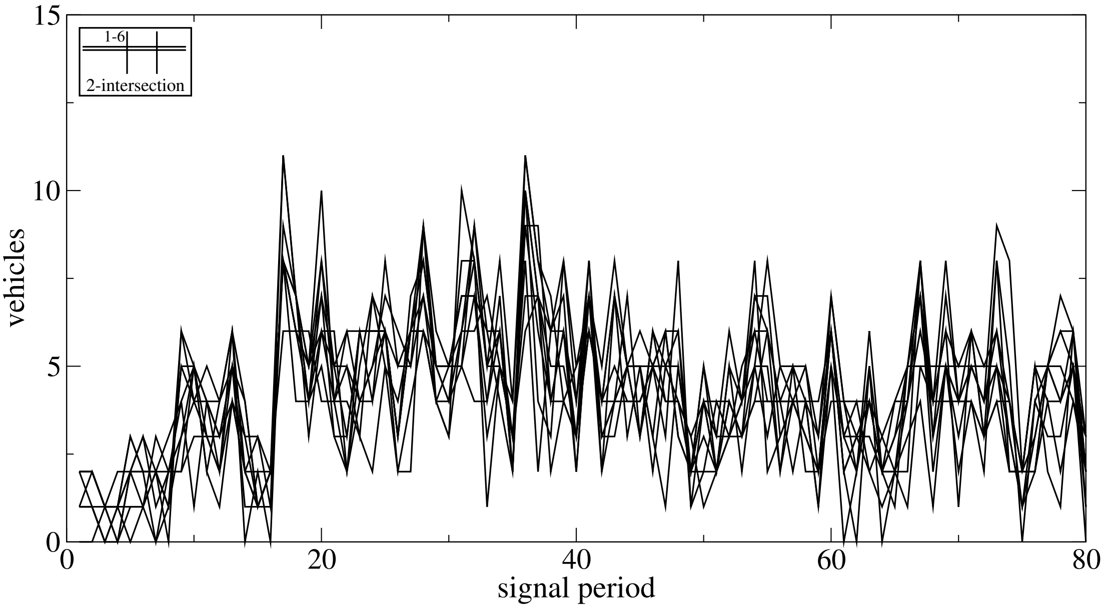


Figure B.14: CA queue lengths for the 2-intersection case at 1-6; 10 different seed numbers, turn rate set 4.

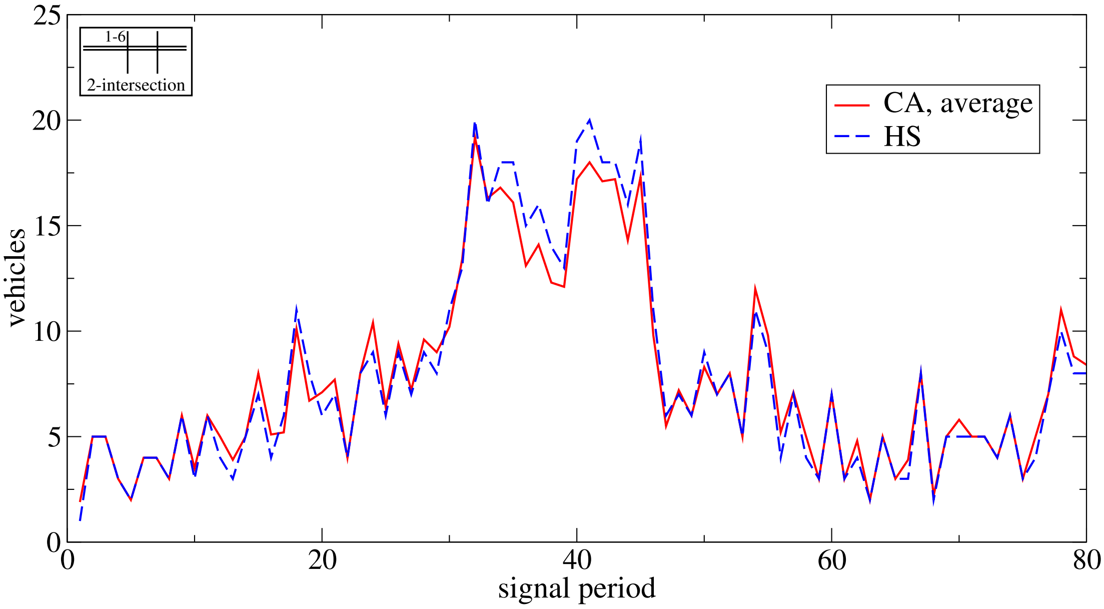


Figure B.15: Queue lengths for the 2-intersection case at 1-6; CA average of 10 different seed numbers, turn rate set 1.

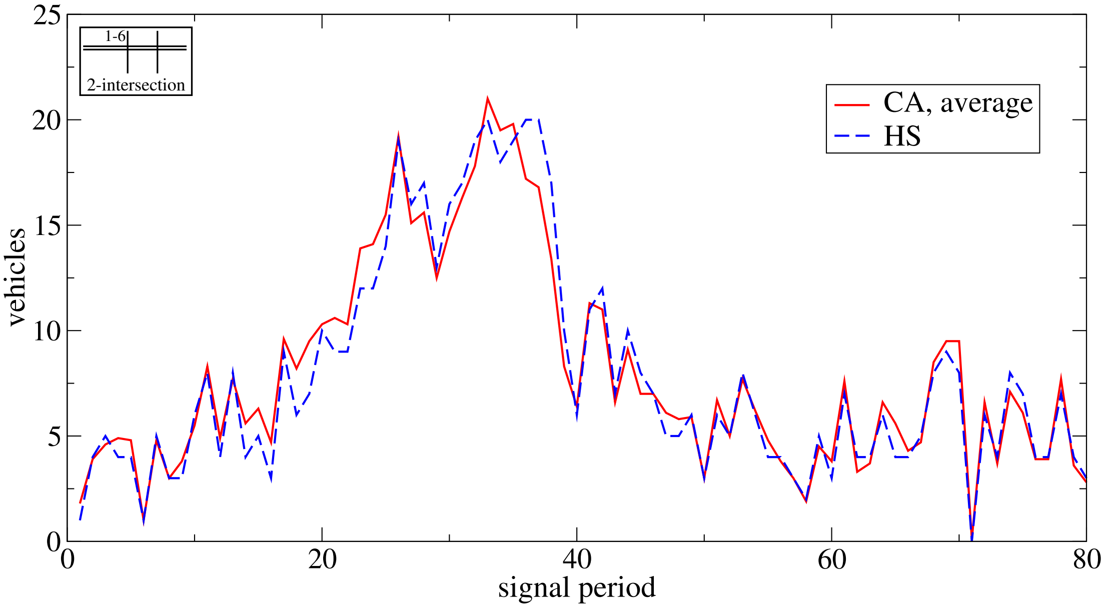


Figure B.16: Queue lengths for the 2-intersection case at 1-6; CA average of 10 different seed numbers, turn rate set 2.

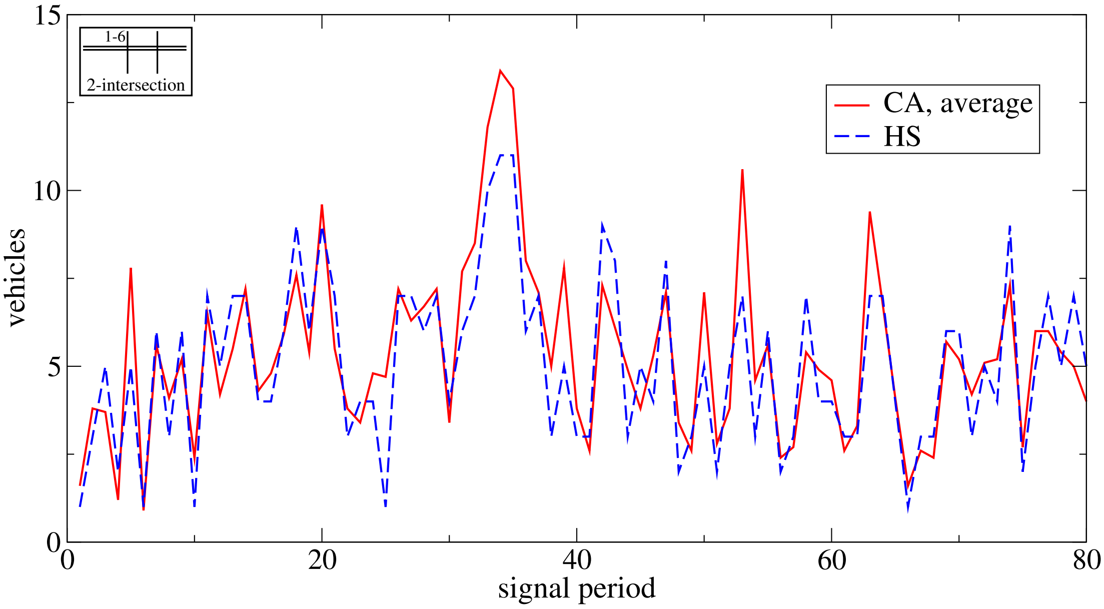


Figure B.17: Queue lengths for the 2-intersection case at 1-6; CA average of 10 different seed numbers, turn rate set 3.

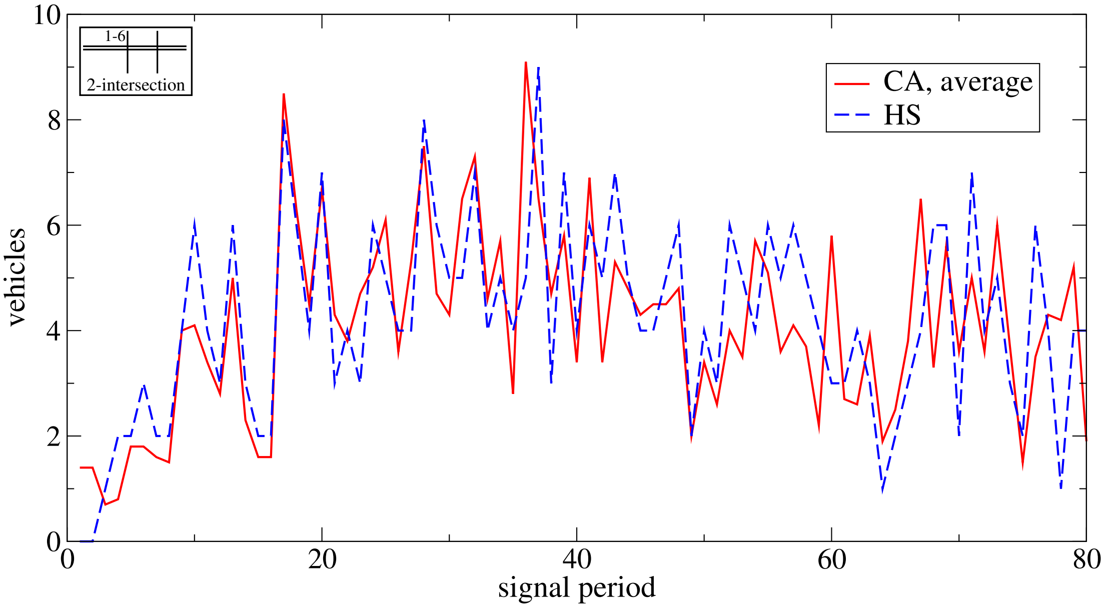


Figure B.18: Queue lengths for the 2-intersection case at 1-6; CA average of 10 different seed numbers, turn rate set 4.

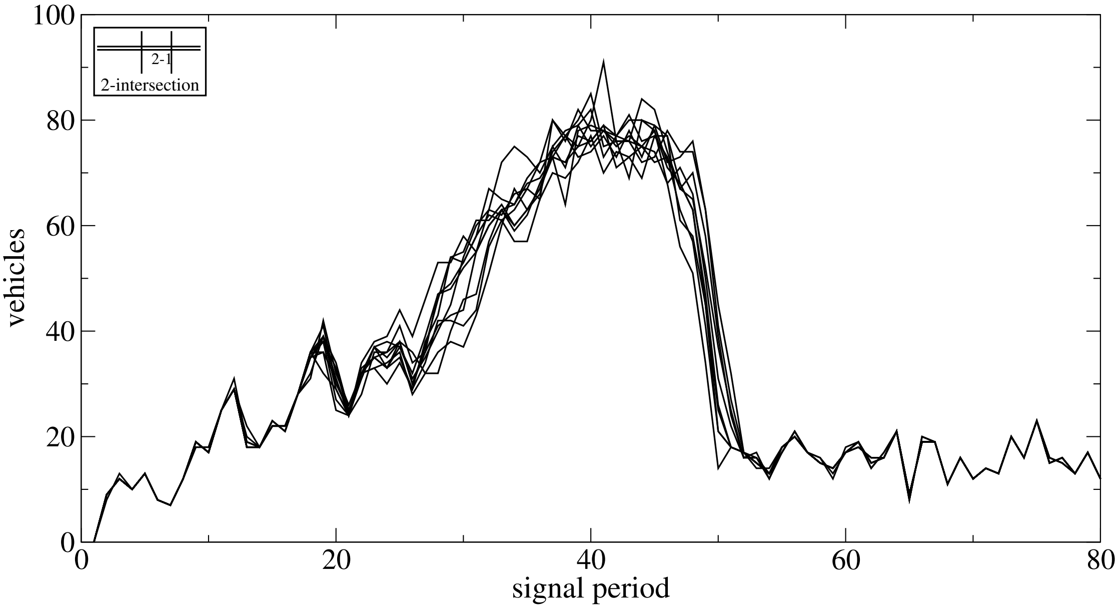


Figure B.19: CA queue lengths for the 2-intersection case at 2-1; 10 different seed numbers, turn rate set 1.

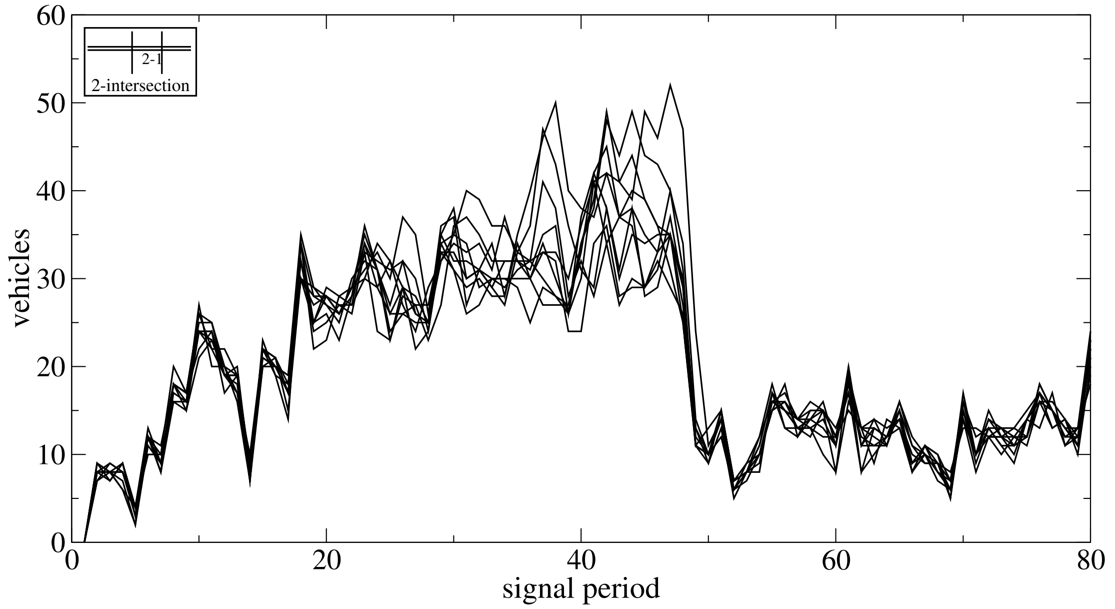


Figure B.20: CA queue lengths for the 2-intersection case at 2-1; 10 different seed numbers, turn rate set 2.

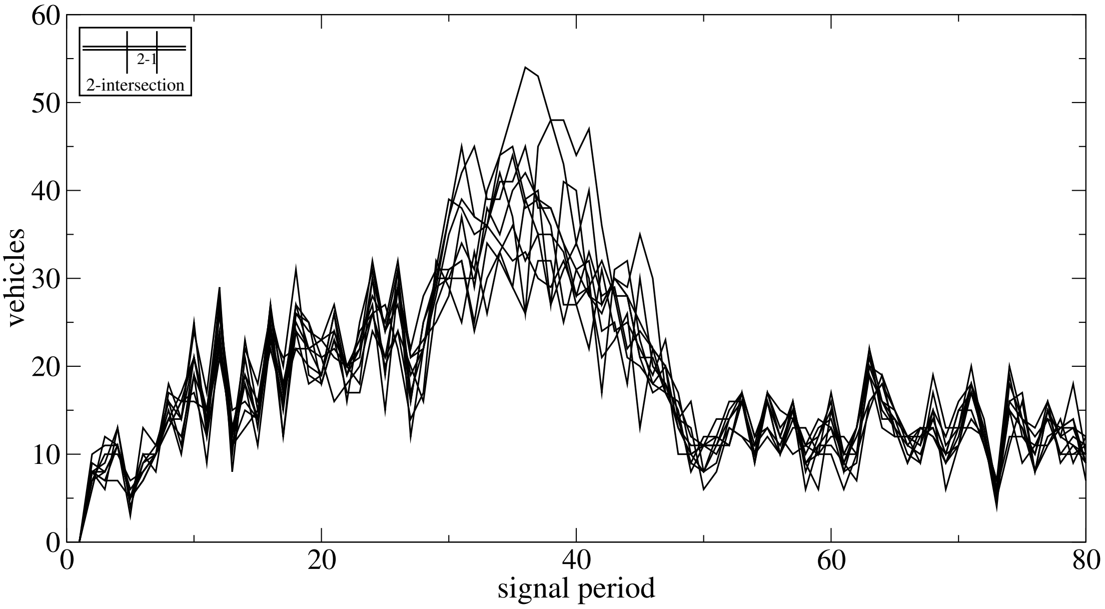


Figure B.21: CA queue lengths for the 2-intersection case at 2-1; 10 different seed numbers, turn rate set 3.

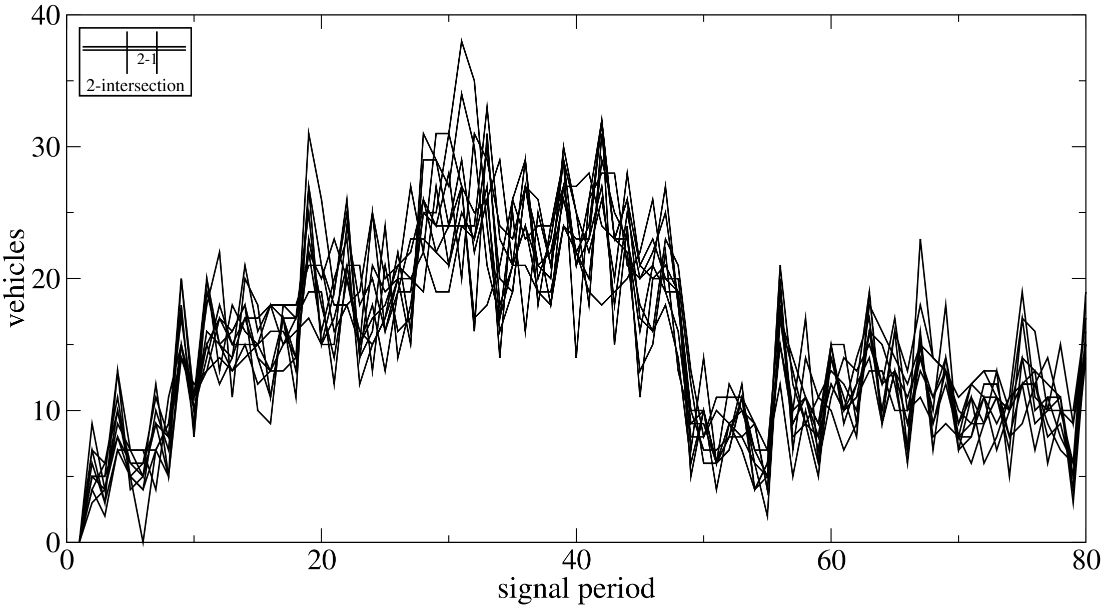


Figure B.22: CA queue lengths for the 2-intersection case at 2-1; 10 different seed numbers, turn rate set 4.

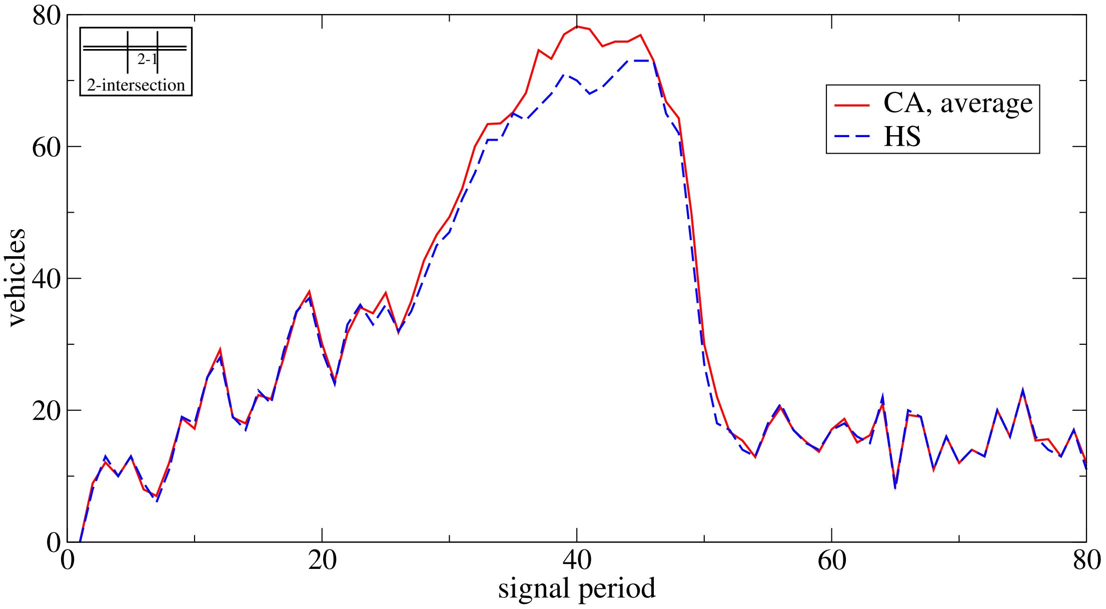


Figure B.23: Queue lengths for the 2-intersection case at 2-1; CA average of 10 different seed numbers, turn rate set 1.

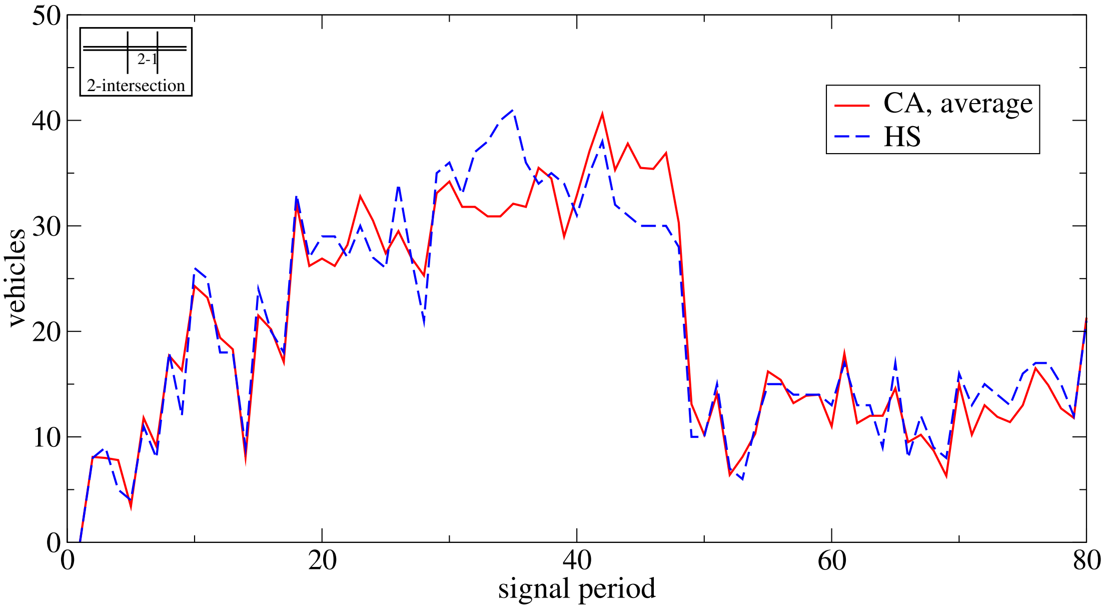


Figure B.24: Queue lengths for the 2-intersection case at 2-1; CA average of 10 different seed numbers, turn rate set 2.

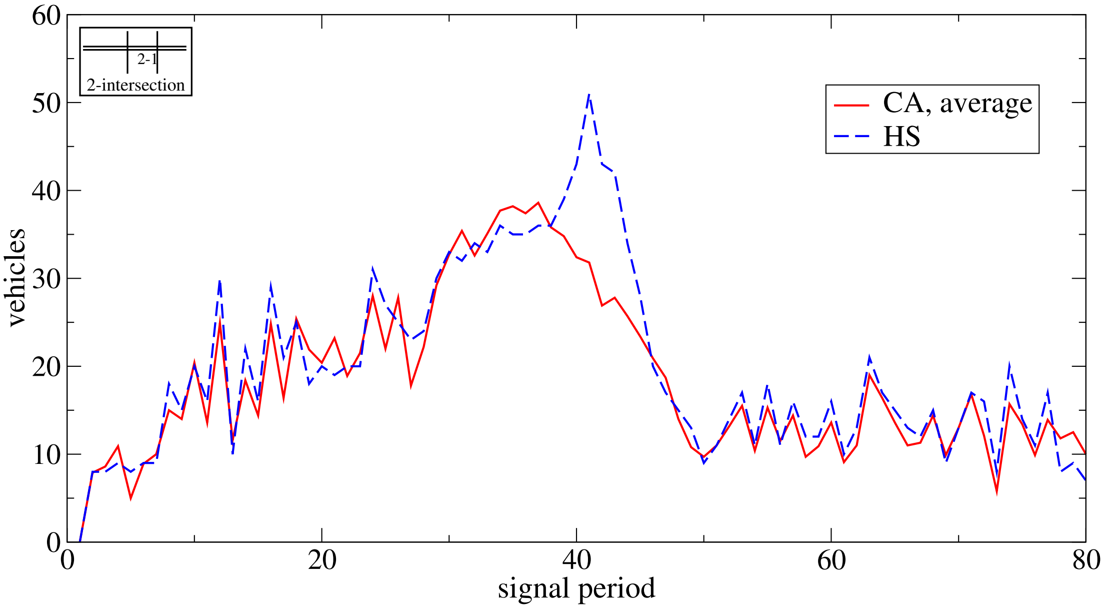


Figure B.25: Queue lengths for the 2-intersection case at 2-1; CA average of 10 different seed numbers, turn rate set 3.

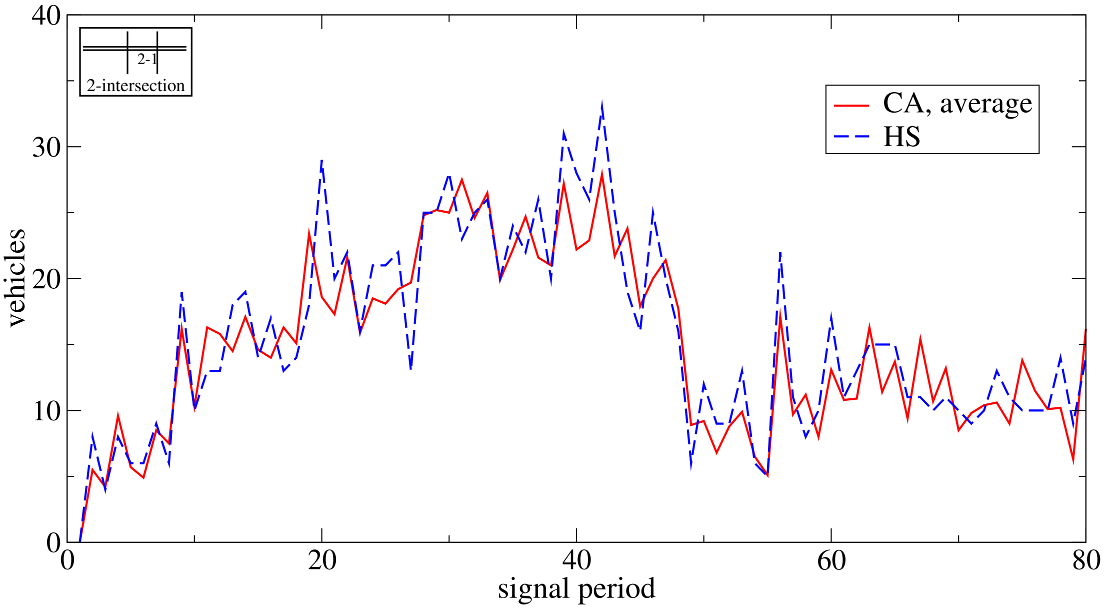


Figure B.26: Queue lengths for the 2-intersection case at 2-1; CA average of 10 different seed numbers, turn rate set 4.

CA run	HUTSIM	HUTSIM	HUTSIM	CA turn	CA seed	CA	CA
number	configfile	delayfile	detsigfile	rate set		logfile	genparfile
	(.cnf)	(.del)	(.dat)	(.prn)		(.log)	(.dat)
41	cr21	hs21	ds21	cr2tu1	23356	rm2_1	genpar1
42	cr21	hs21	ds21	cr2tu1	2332	rm2_11	— “ —
43	cr21	hs21	ds21	cr2tu1	10112	rm2_12	— “ —
44	cr21	hs21	ds21	cr2tu1	9930	rm2_13	— “ —
45	cr21	hs21	ds21	cr2tu1	381	rm2_14	— “ —
46	cr21	hs21	ds21	cr2tu1	3354	rm2_15	— “ —
47	cr21	hs21	ds21	cr2tu1	12001	rm2_16	— “ —
48	cr21	hs21	ds21	cr2tu1	1020	rm2_17	— “ —
49	cr21	hs21	ds21	cr2tu1	25460	rm2_18	— “ —
50	cr21	hs21	ds21	cr2tu1	19500	rm2_19	— “ —
51	cr22	hs22	ds22	cr2tu2	23356	rm2_2	— “ —
52	cr22	hs22	ds22	cr2tu2	2332	rm2_21	— “ —
53	cr22	hs22	ds22	cr2tu2	10112	rm2_22	— “ —
54	cr22	hs22	ds22	cr2tu2	9930	rm2_23	— “ —
55	cr22	hs22	ds22	cr2tu2	381	rm2_24	— “ —
56	cr22	hs22	ds22	cr2tu2	3354	rm2_25	— “ —
57	cr22	hs22	ds22	cr2tu2	12001	rm2_26	— “ —
58	cr22	hs22	ds22	cr2tu2	1020	rm2_27	— “ —
59	cr22	hs22	ds22	cr2tu2	25460	rm2_28	— “ —
60	cr22	hs22	ds22	cr2tu2	19500	rm2_29	— “ —
61	cr23	hs23	ds23	cr2tu3	23356	rm2_3	— “ —
62	cr23	hs23	ds23	cr2tu3	2332	rm2_31	— “ —
63	cr23	hs23	ds23	cr2tu3	10112	rm2_32	— “ —
64	cr23	hs23	ds23	cr2tu3	9930	rm2_33	— “ —
65	cr23	hs23	ds23	cr2tu3	381	rm2_34	— “ —
66	cr23	hs23	ds23	cr2tu3	3354	rm2_35	— “ —
67	cr23	hs23	ds23	cr2tu3	12001	rm2_36	— “ —
68	cr23	hs23	ds23	cr2tu3	1020	rm2_37	— “ —
69	cr23	hs23	ds23	cr2tu3	25460	rm2_38	— “ —
70	cr23	hs23	ds23	cr2tu3	19500	rm2_39	— “ —
71	cr24	hs24	ds24	cr2tu4	23356	rm2_4	— “ —
72	cr24	hs24	ds24	cr2tu4	2332	rm2_41	— “ —
73	cr24	hs24	ds24	cr2tu4	10112	rm2_42	— “ —
74	cr24	hs24	ds24	cr2tu4	9930	rm2_43	— “ —
75	cr24	hs24	ds24	cr2tu4	381	rm2_44	— “ —
76	cr24	hs24	ds24	cr2tu4	3354	rm2_45	— “ —
77	cr24	hs24	ds24	cr2tu4	12001	rm2_46	— “ —
78	cr24	hs24	ds24	cr2tu4	1020	rm2_47	— “ —
79	cr24	hs24	ds24	cr2tu4	25460	rm2_48	— “ —
80	cr24	hs24	ds24	cr2tu4	19500	rm2_49	— “ —

Table B.3: Key for HUTSIM and CA runs, the 2-intersection case, average velocities.

origin	desti- nation	turn rate set	flow profile	figure	CA run numbers
11	2	1	origin 1	B.27	41 – 50
11	2	2	origin 1	B.28	51 – 60
11	2	3	origin 1	B.29	61 – 70
11	2	4	origin 1	B.30	71 – 80
31	2	1	origin 3	B.31	41 – 50
31	2	2	origin 3	B.32	51 – 60
31	2	3	origin 3	B.33	61 – 70
31	2	4	origin 3	B.34	71 – 80
41	3	1	origin 4	B.35	41 – 50
41	3	2	origin 4	B.36	51 – 60
41	3	3	origin 4	B.37	61 – 70
41	3	4	origin 4	B.38	71 – 80

Table B.4: Key for figures and CA runs, the 2-intersection case, average velocities.

CA run number	HUTSIM configfile (.cnf)	HUTSIM delayfile (.del)	HUTSIM detsigfile (.dat)	CA turn rate set (.prn)	CA seed	CA logfile (.log)	CA genparfile (.dat)
81	cr23	hs23	ds23	cr2tu3	23356	rd2_3	genpar1
82	cr23	hs23	ds23	cr2tu3	2332	rd2_31	— “ —
83	cr23	hs23	ds23	cr2tu3	10112	rd2_32	— “ —
84	cr23	hs23	ds23	cr2tu3	9930	rd2_33	— “ —
85	cr23	hs23	ds23	cr2tu3	381	rd2_34	— “ —
86	cr23	hs23	ds23	cr2tu3	3354	rd2_35	— “ —
87	cr23	hs23	ds23	cr2tu3	12001	rd2_36	— “ —
88	cr23	hs23	ds23	cr2tu3	1020	rd2_37	— “ —
89	cr23	hs23	ds23	cr2tu3	25460	rd2_38	— “ —
90	cr23	hs23	ds23	cr2tu3	19500	rd2_39	— “ —

Table B.5: Key for HUTSIM and CA runs, the 2-intersection case, traffic flow.

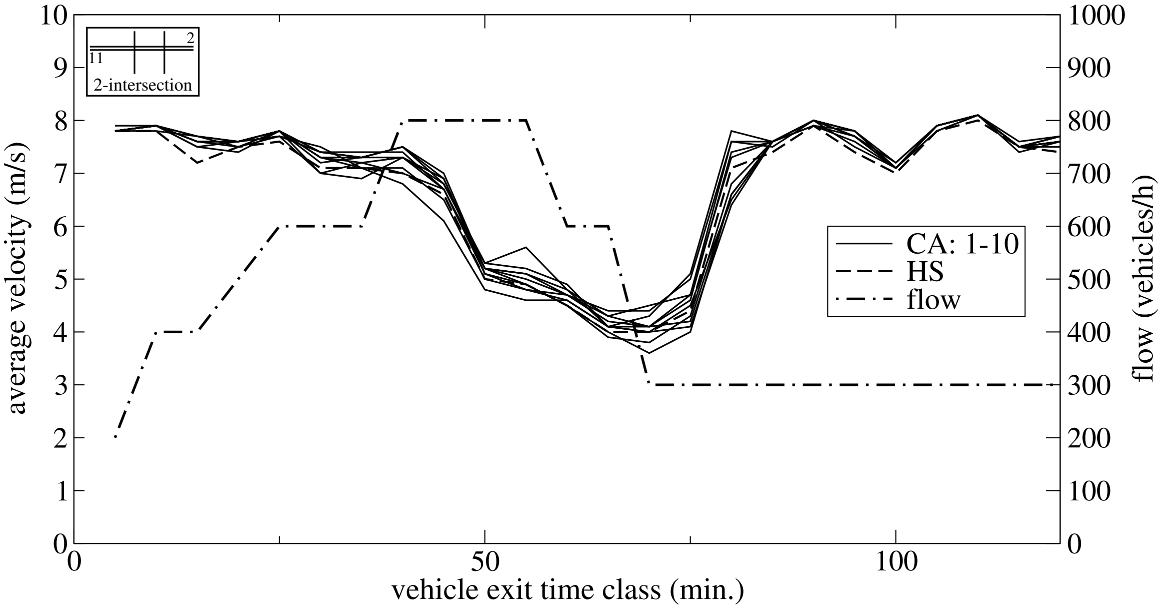


Figure B.27: Average velocities for CA vehicles on route 11 → 2; 10 different seed numbers for CA, turn rate set 1, flow profile from origin 1.

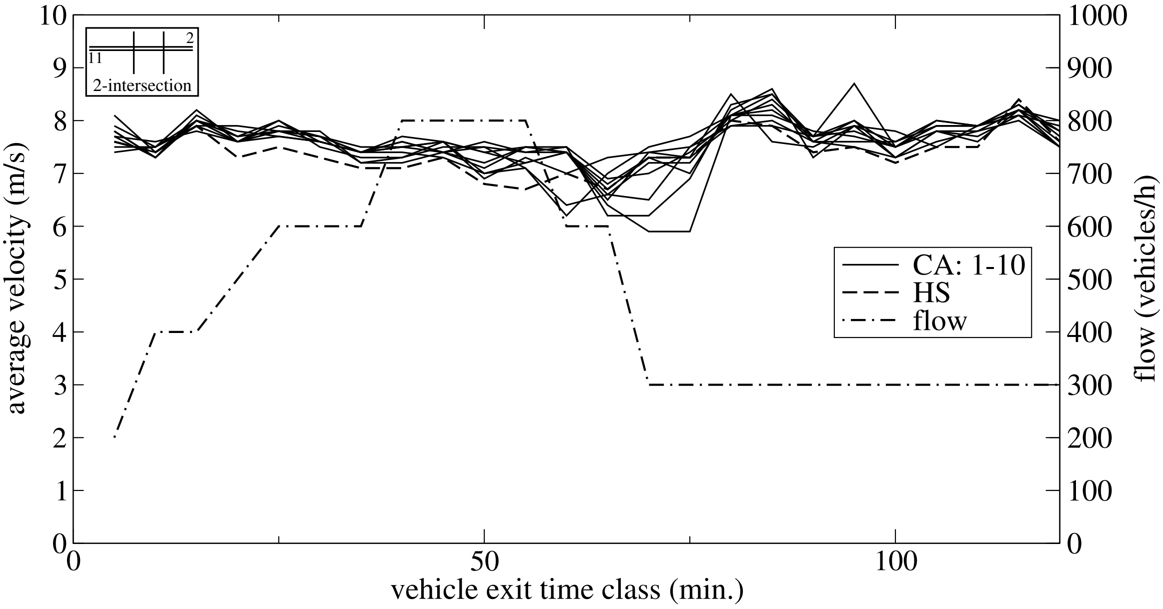


Figure B.28: Average velocities for CA vehicles on route 11 → 2; 10 different seed numbers for CA, turn rate set 2, flow profile from origin 1.

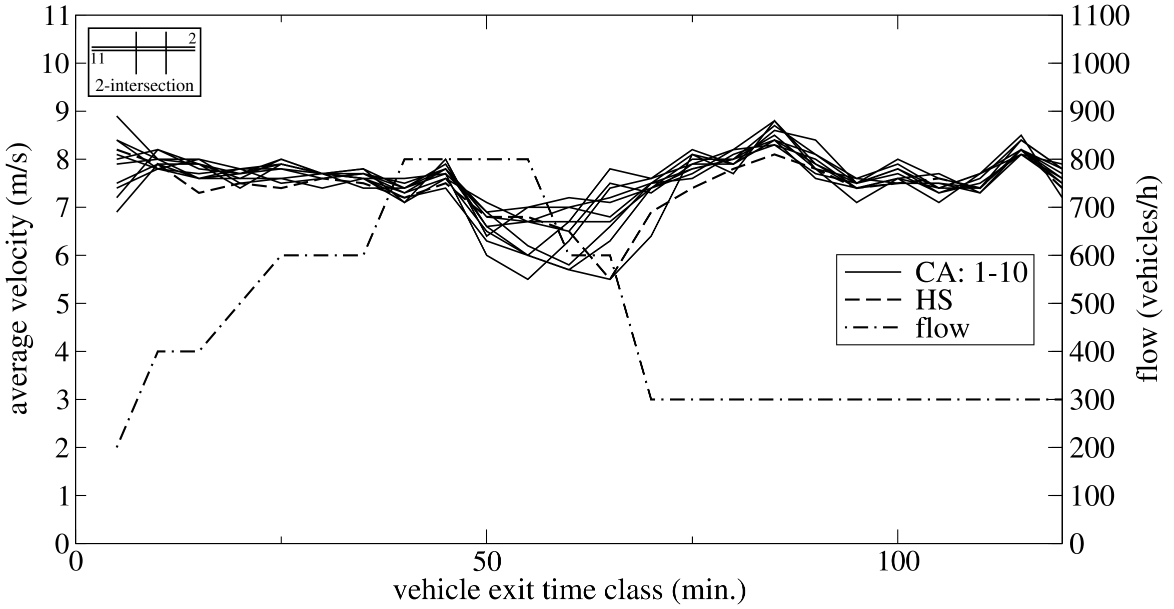


Figure B.29: Average velocities for CA vehicles on route 11 → 2; 10 different seed numbers for CA, turn rate set 3, flow profile from origin 1.

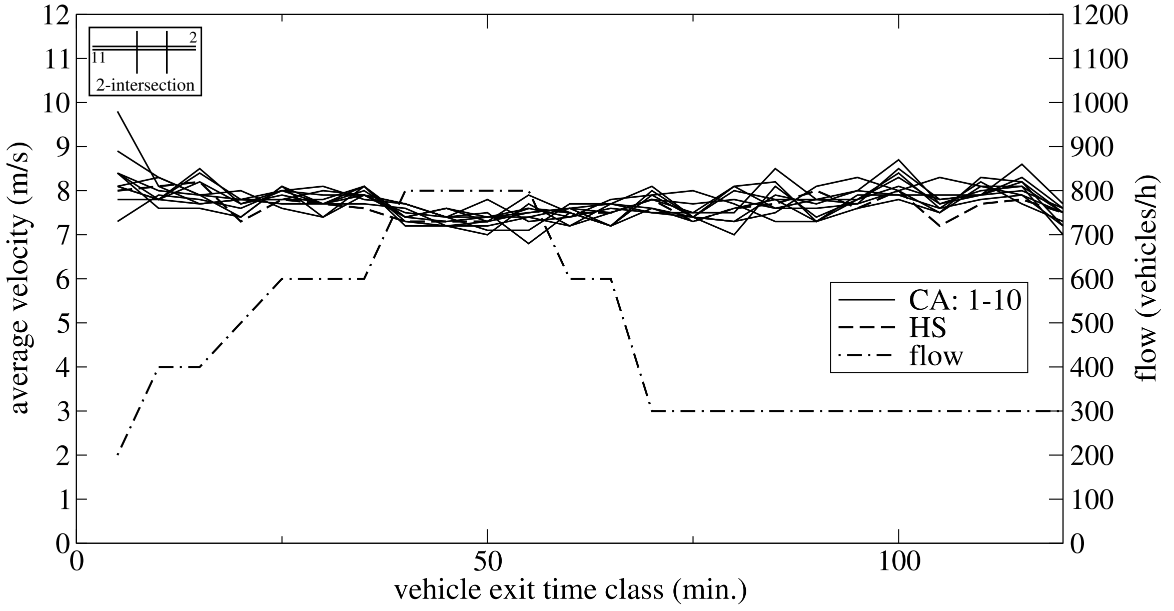


Figure B.30: Average velocities for CA vehicles on route 11 → 2; 10 different seed numbers for CA, turn rate set 4, flow profile from origin 1.

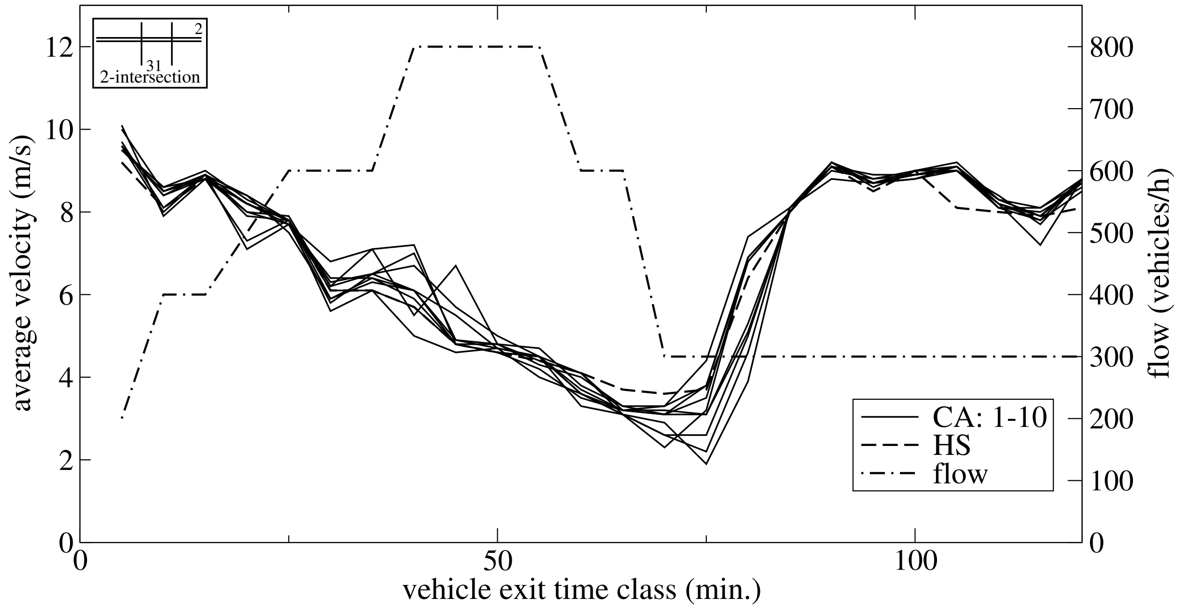


Figure B.31: Average velocities for CA vehicles on route 31 → 2; 10 different seed numbers for CA, turn rate set 1, flow profile from origin 1.

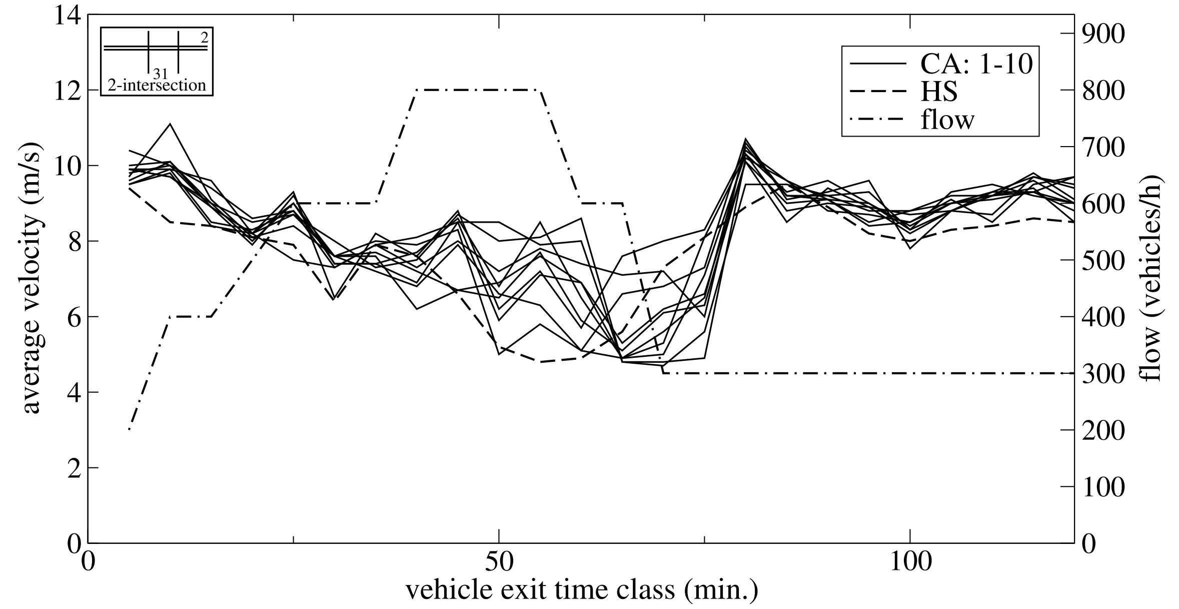


Figure B.32: Average velocities for CA vehicles on route 31 → 2; 10 different seed numbers for CA, turn rate set 2, flow profile from origin 1.

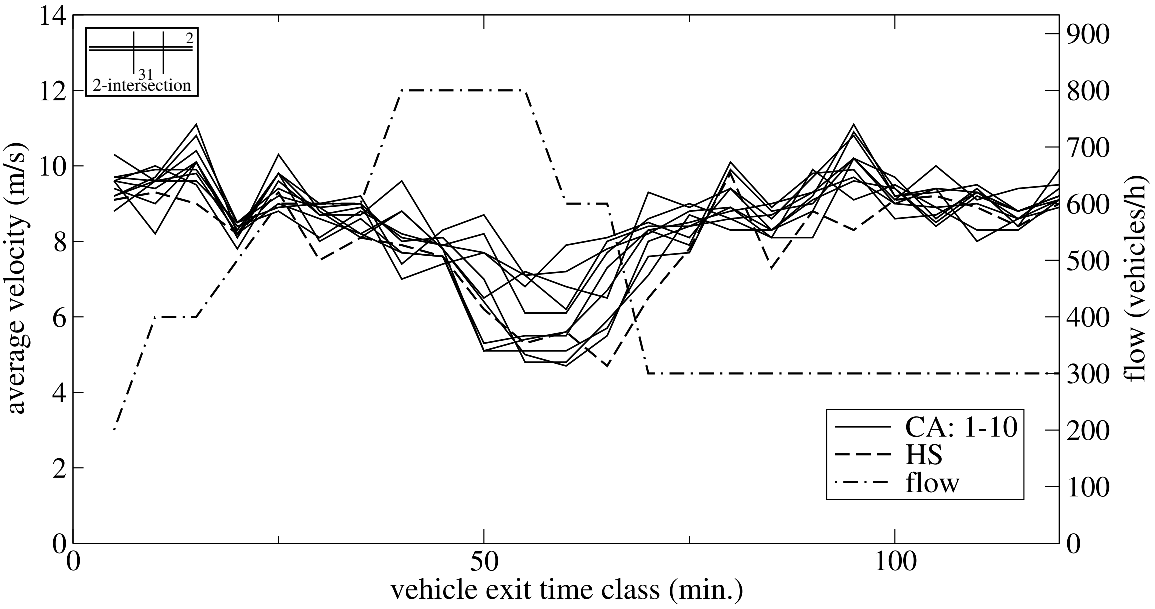


Figure B.33: Average velocities for CA vehicles on route 31 → 2; 10 different seed numbers for CA, turn rate set 3, flow profile from origin 1.

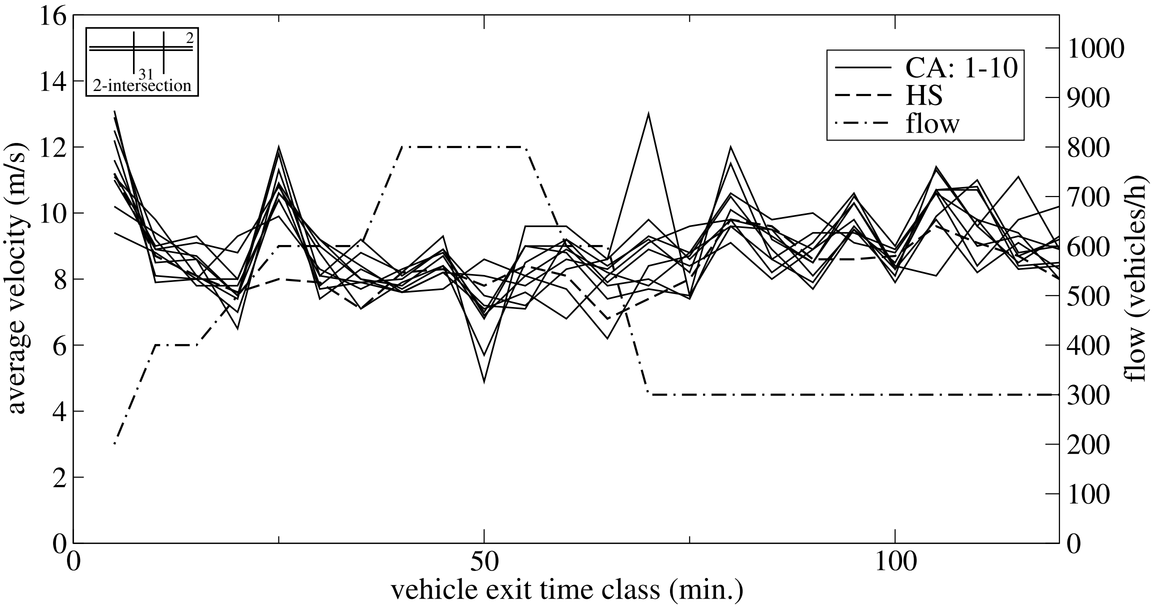


Figure B.34: Average velocities for CA vehicles on route 31 → 2; 10 different seed numbers for CA, turn rate set 4, flow profile from origin 1.

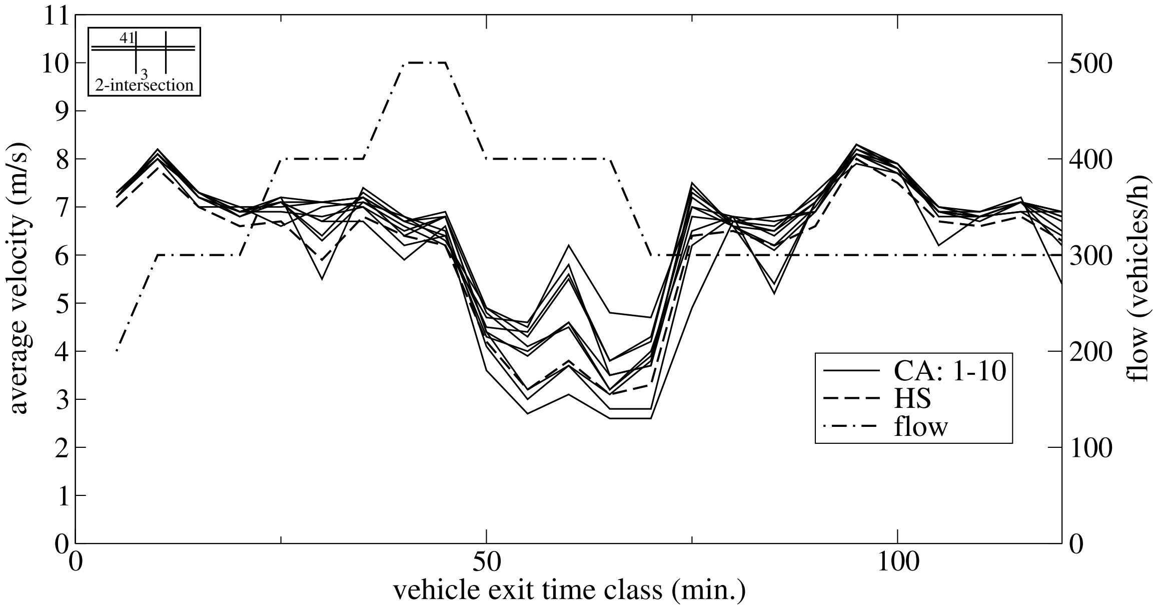


Figure B.35: Average velocities for CA vehicles on route 41 → 3; 10 different seed numbers for CA, turn rate set 1, flow profile from origin 4.

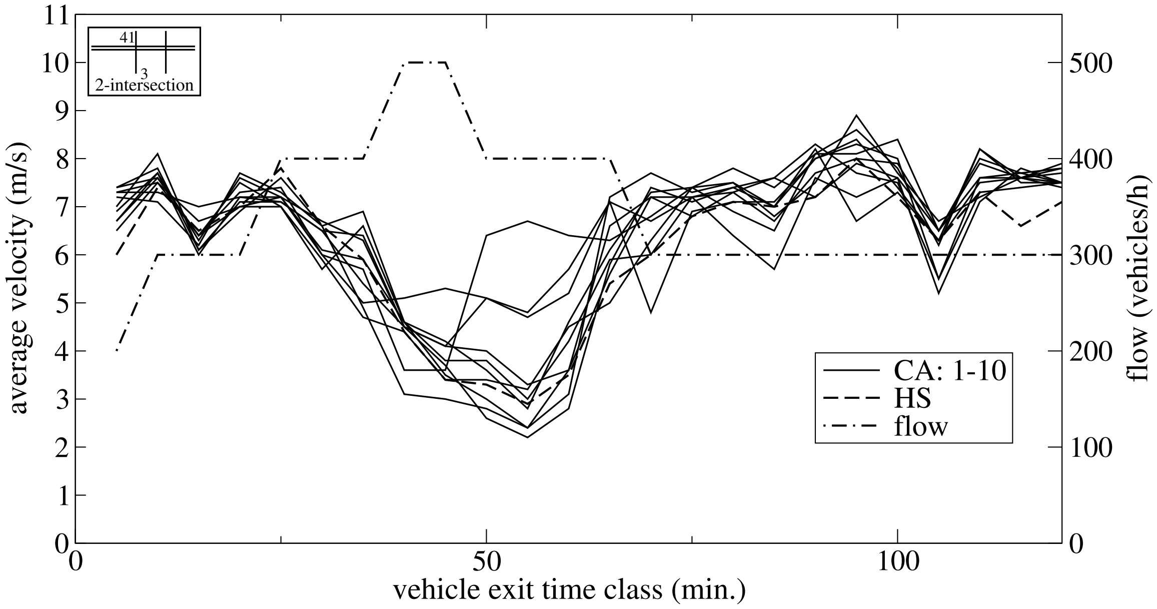


Figure B.36: Average velocities for CA vehicles on route 41 → 3; 10 different seed numbers for CA, turn rate set 2, flow profile from origin 4.

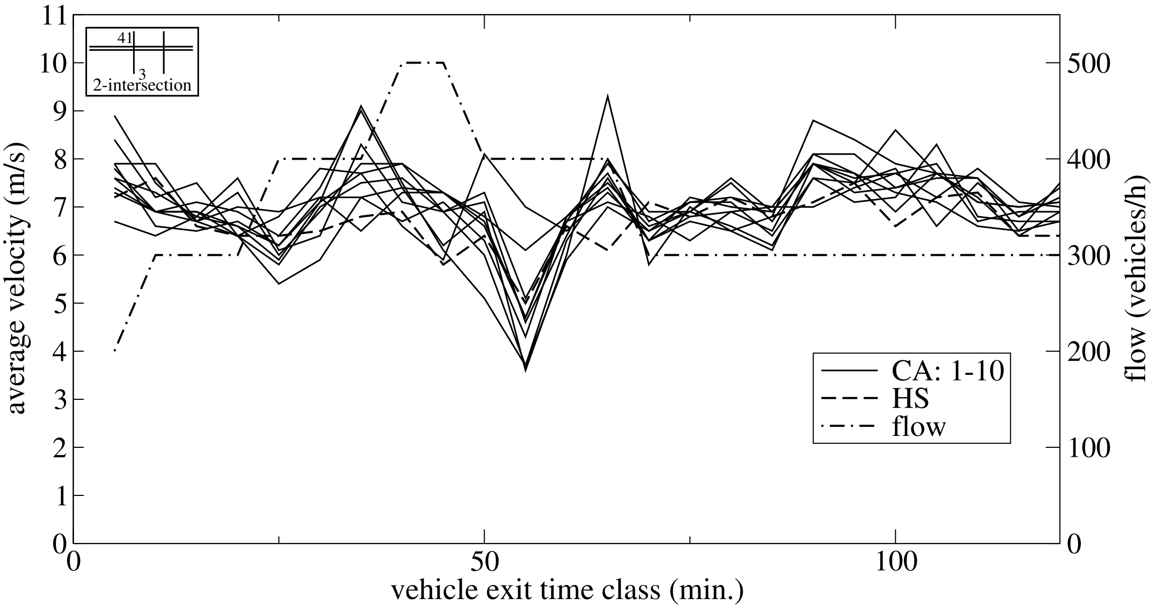


Figure B.37: Average velocities for CA vehicles on route 41 → 3; 10 different seed numbers for CA, turn rate set 3, flow profile from origin 4.

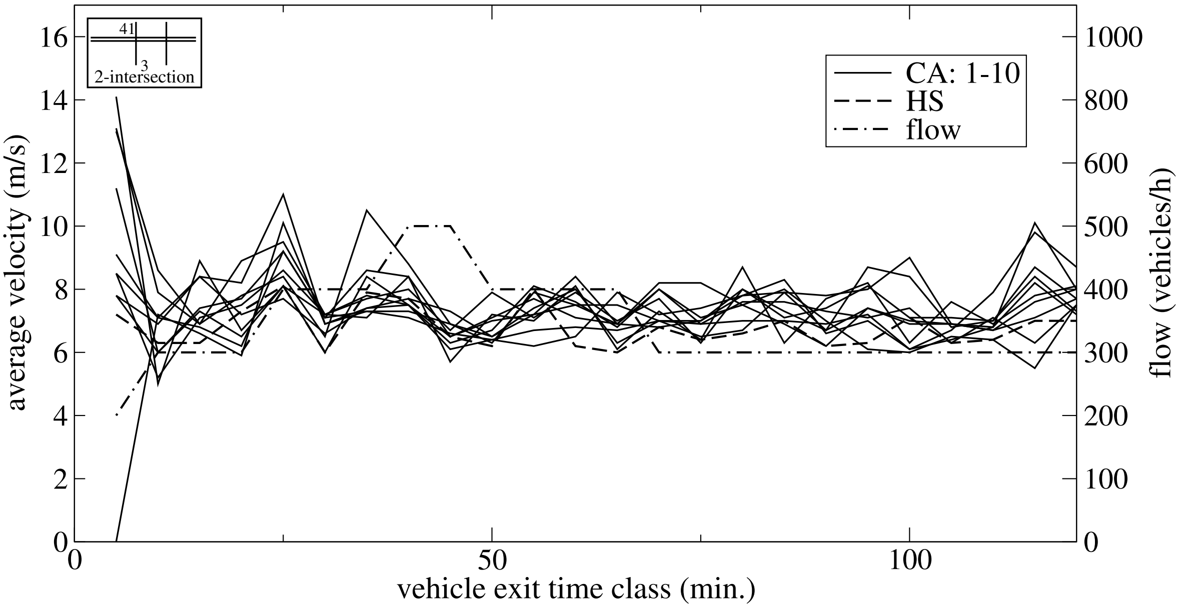


Figure B.38: Average velocities for CA vehicles on route 41 → 3; 10 different seed numbers for CA, turn rate set 4, flow profile from origin 4.

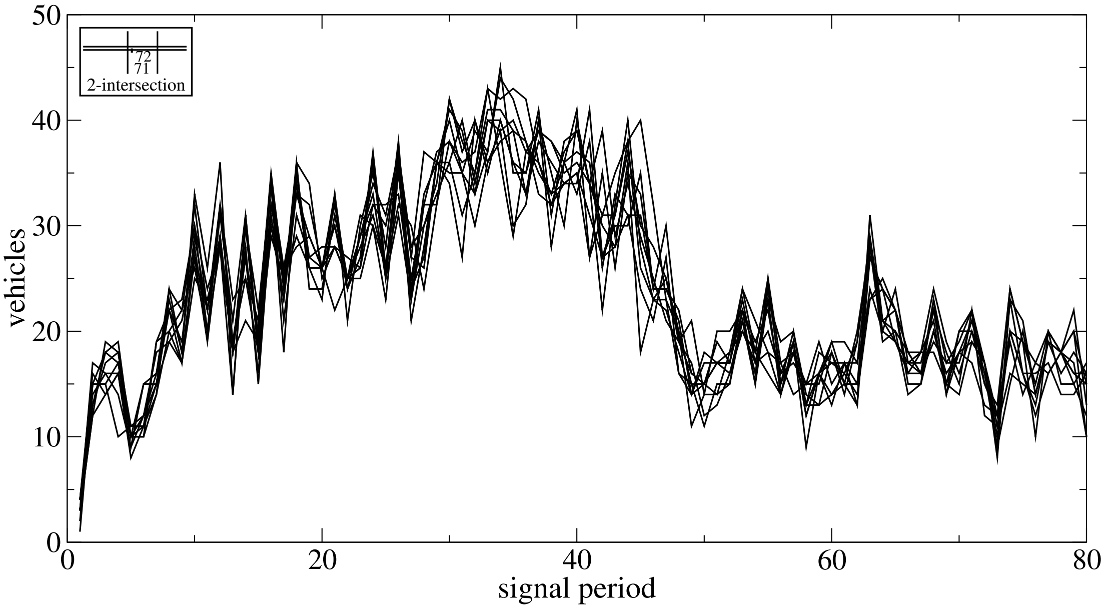


Figure B.39: Vehicles crossing detectors 71+72, different seed numbers, turn rate set 3 (CA runs 81–90).

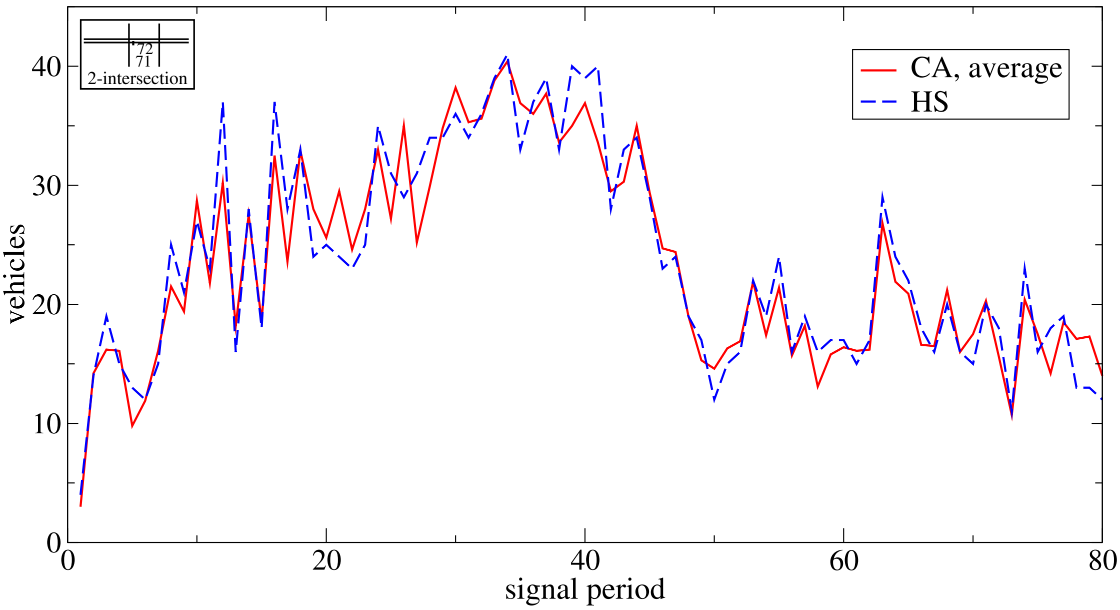


Figure B.40: Vehicles crossing detectors 71+72, CA average of 10 runs, turn rate set 3 (CA runs 81–90).

CA run number	HUTSIM configfile (.cnf)	HUTSIM delayfile (.del)	HUTSIM detsigfile (.dat)	CA turn rate set (.prn)	CA seed	CA logfile (.log)	CA genparfile (.dat)
91	tyk_i_b2	hs5	ds5	cr5tu	23356	rq5_1	genpar1
92	tyk_i_b2	hs5	ds5	cr5tu	2332	rq5_11	— “ —
93	tyk_i_b2	hs5	ds5	cr5tu	10112	rq5_12	— “ —
94	tyk_i_b2	hs5	ds5	cr5tu	9930	rq5_13	— “ —
95	tyk_i_b2	hs5	ds5	cr5tu	381	rq5_14	— “ —
96	tyk_i_b2	hs5	ds5	cr5tu	3354	rq5_15	— “ —
97	tyk_i_b2	hs5	ds5	cr5tu	12001	rq5_16	— “ —
98	tyk_i_b2	hs5	ds5	cr5tu	1020	rq5_17	— “ —
99	tyk_i_b2	hs5	ds5	cr5tu	25460	rq5_18	— “ —
100	tyk_i_b2	hs5	ds5	cr5tu	19500	rq5_19	— “ —
101	tyk_i_b3	hs5s	ds5s	cr5tu	23356	rqs5_1	genpar1
102	tyk_i_b3	hs5s	ds5s	cr5tu	2332	rqs5_11	— “ —
103	tyk_i_b3	hs5s	ds5s	cr5tu	10112	rqs5_12	— “ —
104	tyk_i_b3	hs5s	ds5s	cr5tu	9930	rqs5_13	— “ —
104	tyk_i_b3	hs5s	ds5s	cr5tu	381	rqs5_14	— “ —
106	tyk_i_b3	hs5s	ds5s	cr5tu	3354	rqs5_15	— “ —
107	tyk_i_b3	hs5s	ds5s	cr5tu	12001	rqs5_16	— “ —
108	tyk_i_b3	hs5s	ds5s	cr5tu	1020	rqs5_17	— “ —
109	tyk_i_b3	hs5s	ds5s	cr5tu	25460	rqs5_18	— “ —
110	tyk_i_b3	hs5s	ds5s	cr5tu	19500	rqs5_19	— “ —

Table B.6: Key for HUTSIM and CA runs, the 5-intersection case, queue lengths.

queue(s) at post	figure	CA run numbers	queue(s) at post	figure	CA run numbers
1-2	B.41	91 – 100	5-3	B.49	91 – 100
1-2	B.42	— “ —	5-3	B.50	101 – 110
3-3	B.43	— “ —	5-4	B.51	91 – 100
3-3	B.44	— “ —	5-4	B.52	101 – 110
5-1	B.45	— “ —	1-4	B.53	91 – 100
5-1	B.46	— “ —	1-4	B.54	101 – 110
5-11	B.47	— “ —	3-7	B.55	91 – 100
5-11	B.48	— “ —	3-7	B.56	101 – 110
			5-4	B.57	91 – 100
			5-4	B.58	101 – 110

Table B.7: Key for figures and CA runs, the 5-intersection case, queue lengths.

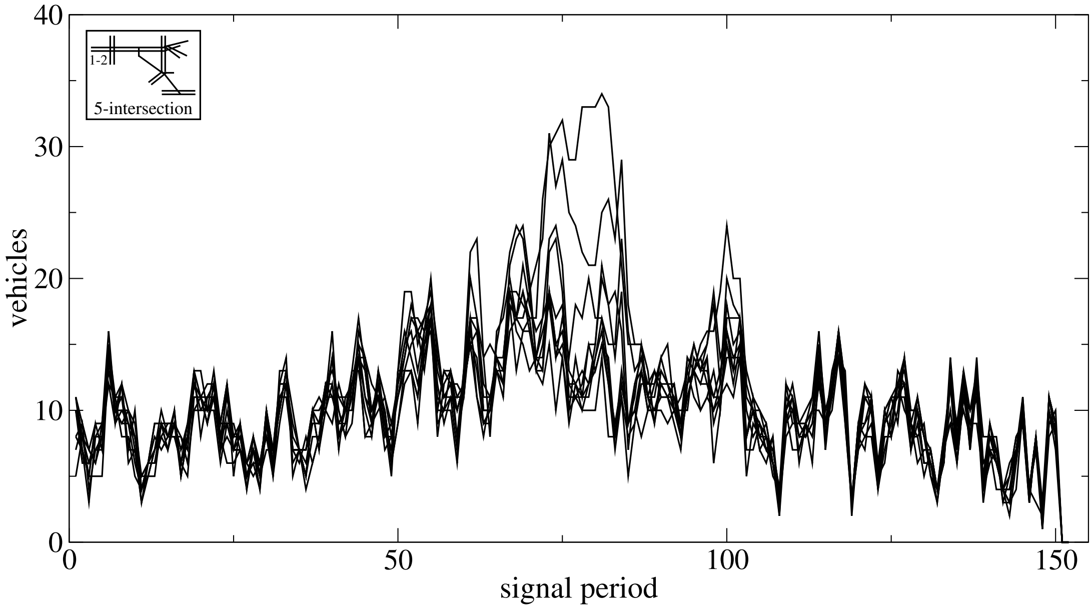


Figure B.41: CA queue lengths for the 5-intersection case at 1-2; 10 different seed numbers.

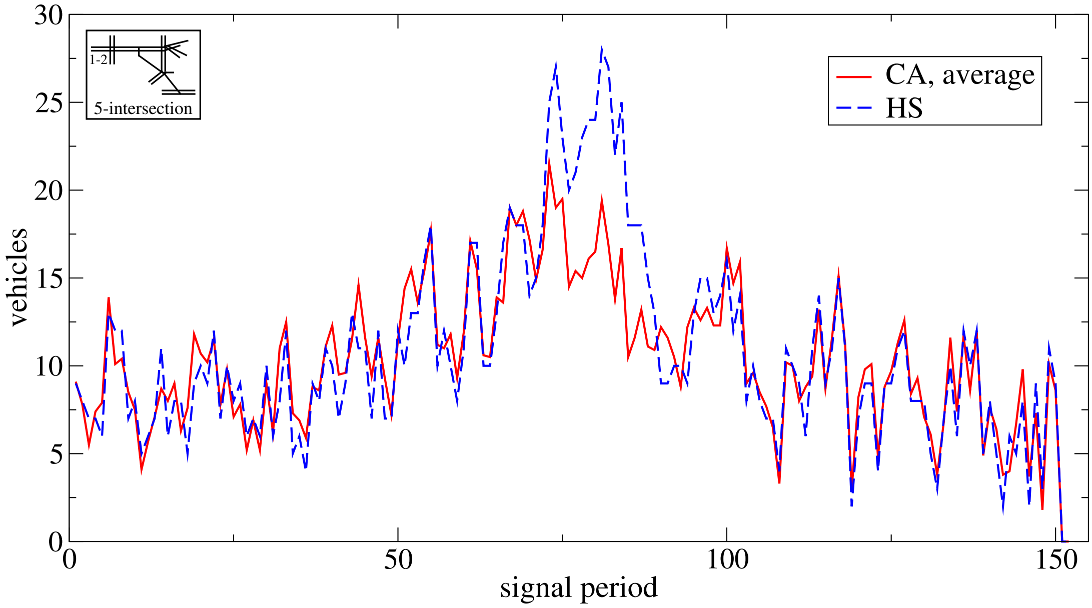


Figure B.42: Queue lengths for the 5-intersection case at 1-2; CA average of 10 different seed numbers.

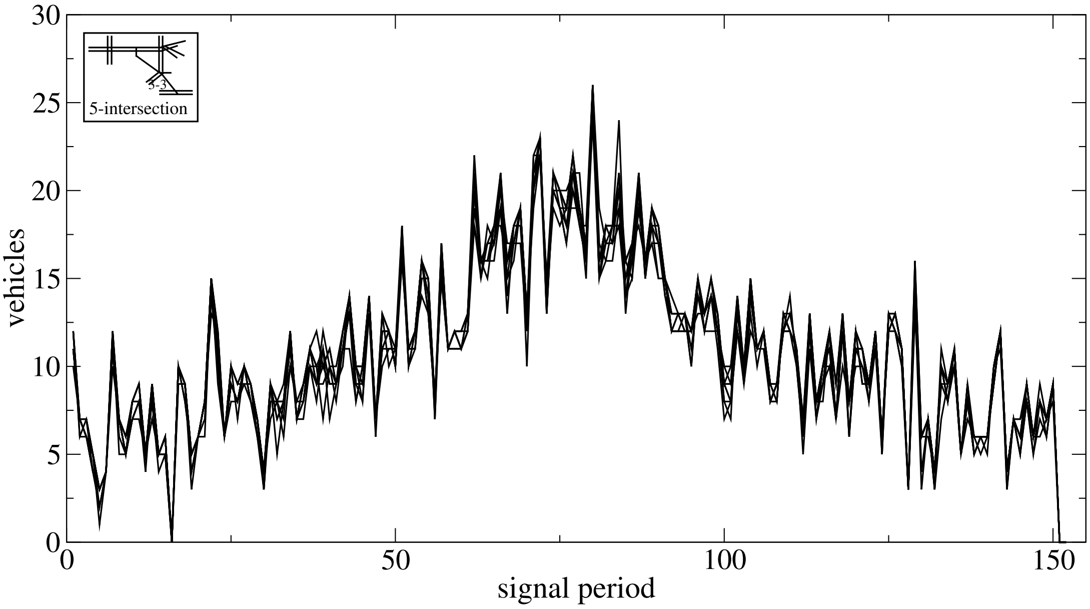


Figure B.43: CA queue lengths for the 5-intersection case at 3-3; 10 different seed numbers.

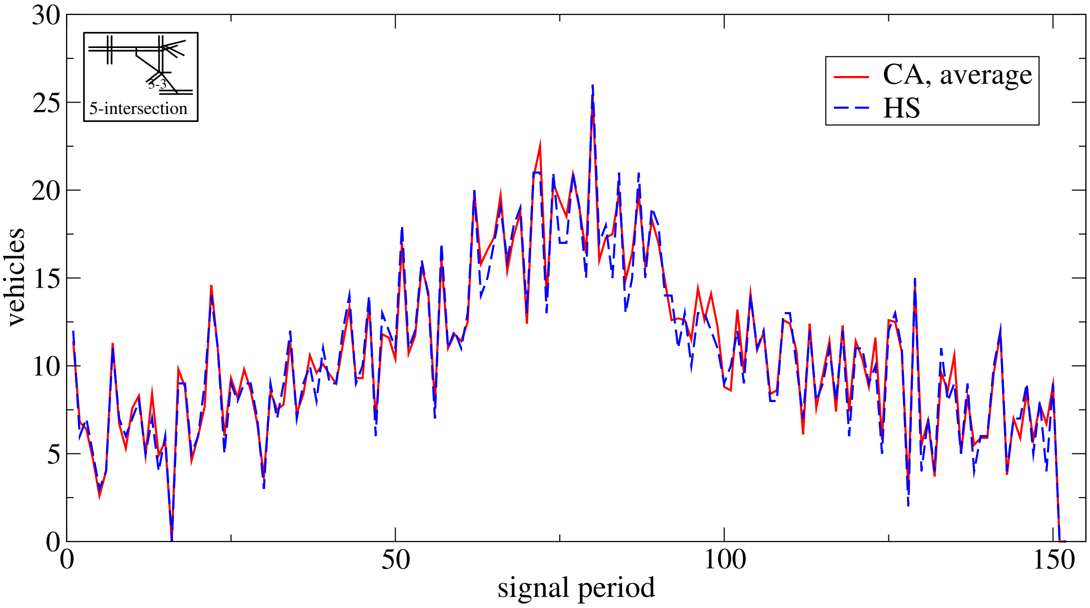


Figure B.44: Queue lengths for the 5-intersection case at 3-3; CA average of 10 different seed numbers.

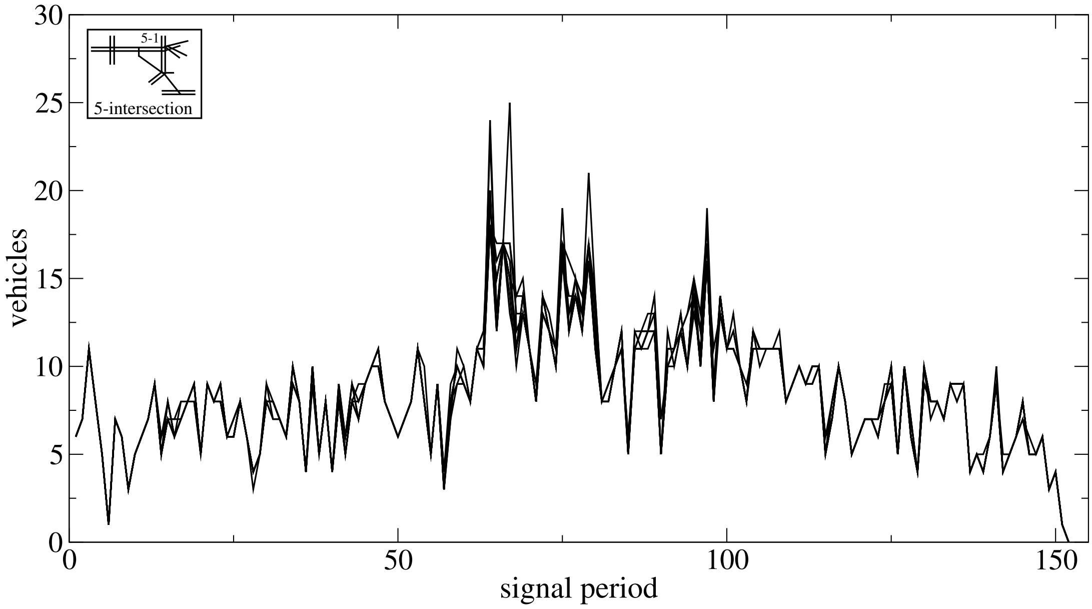


Figure B.45: CA queue lengths for the 5-intersection case at 5-1; 10 different seed numbers.

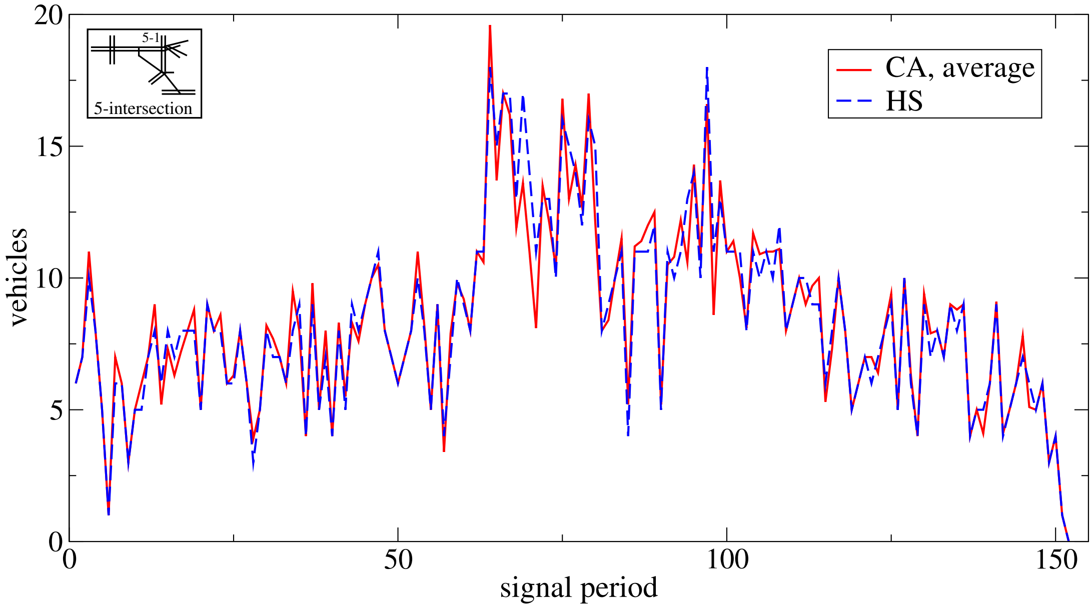


Figure B.46: Queue lengths for the 5-intersection case at 5-1; CA average of 10 different seed numbers.

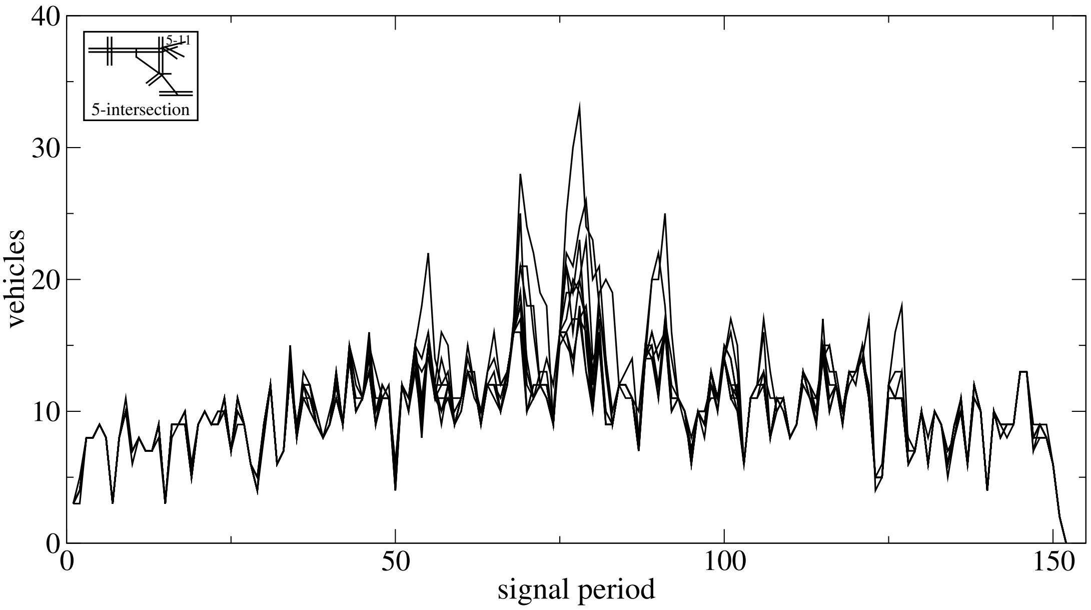


Figure B.47: CA queue lengths for the 5-intersection case at 5-11; 10 different seed numbers.

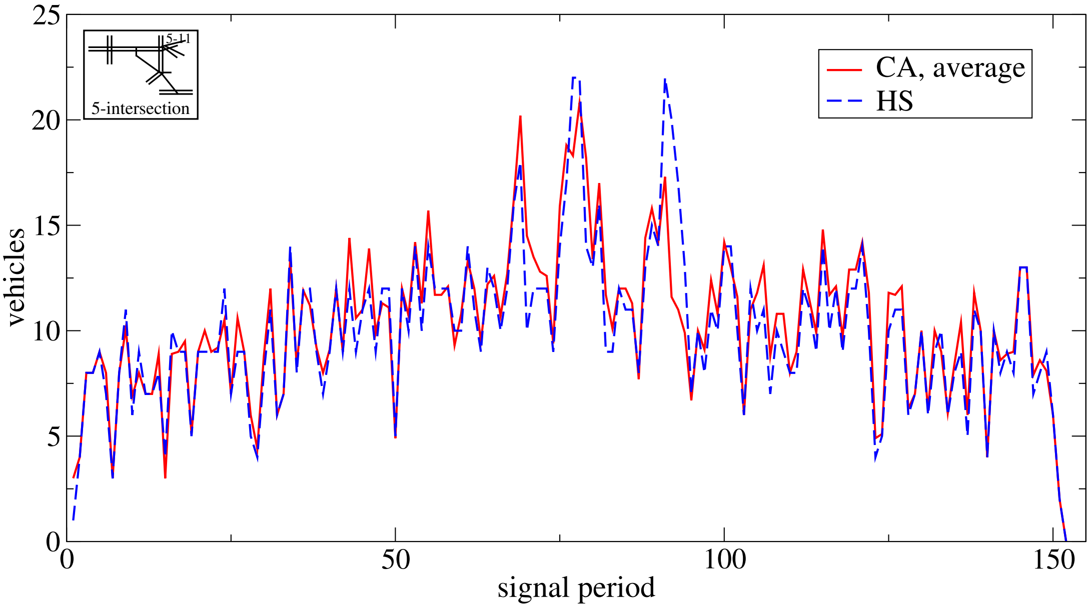


Figure B.48: Queue lengths for the 5-intersection case at 5-11; CA average of 10 different seed numbers.

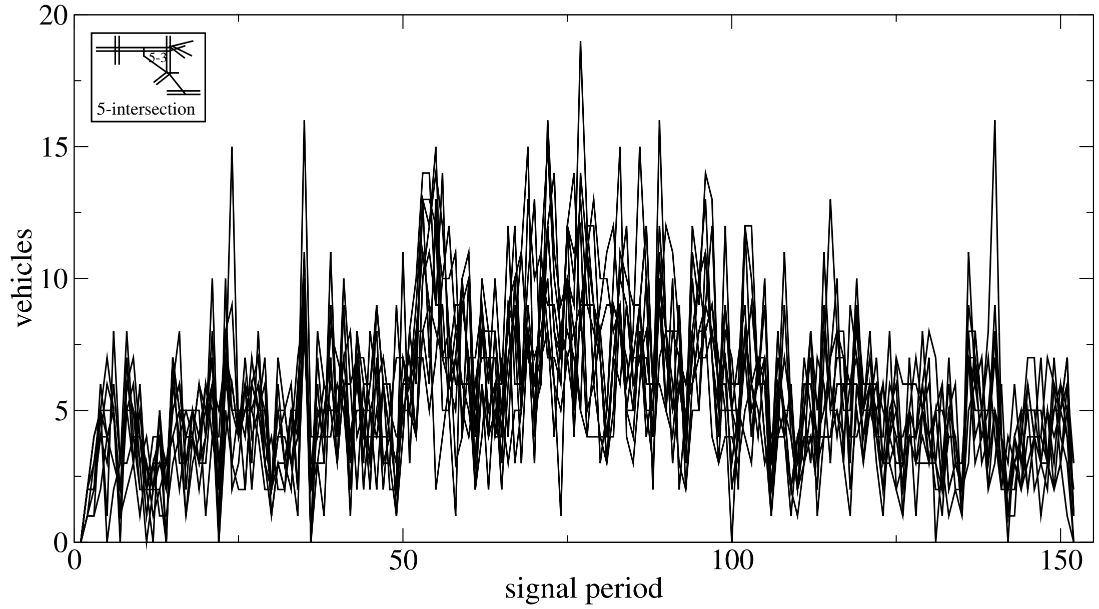


Figure B.49: Queue lengths for the 5-intersection case at 5-3; ten CA runs.

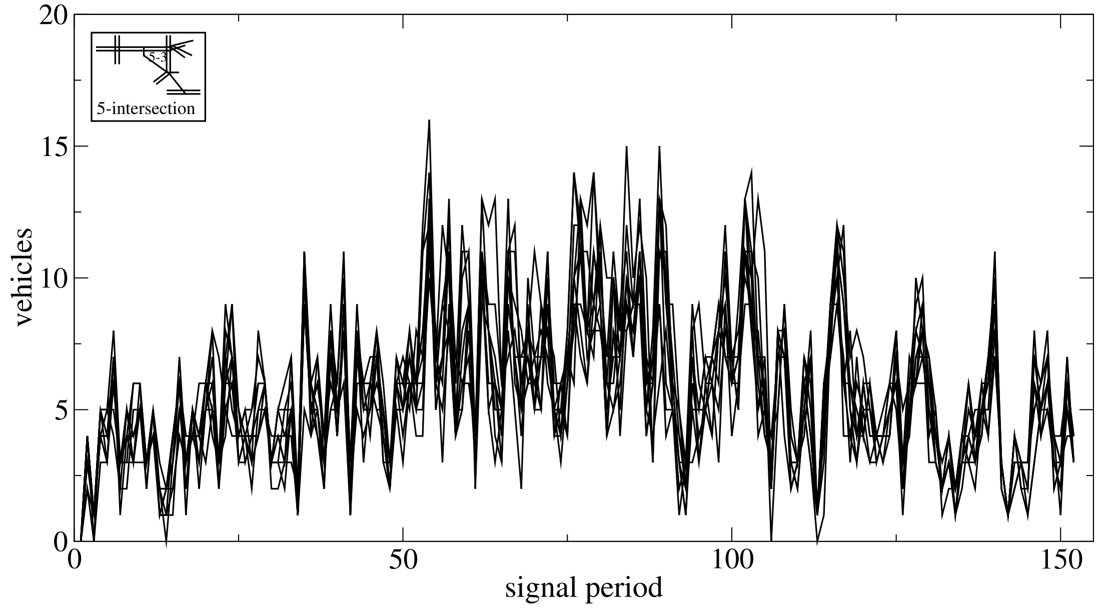


Figure B.50: Queue lengths for the 5-intersection case at 5-3; ten CA runs, extra detectors.

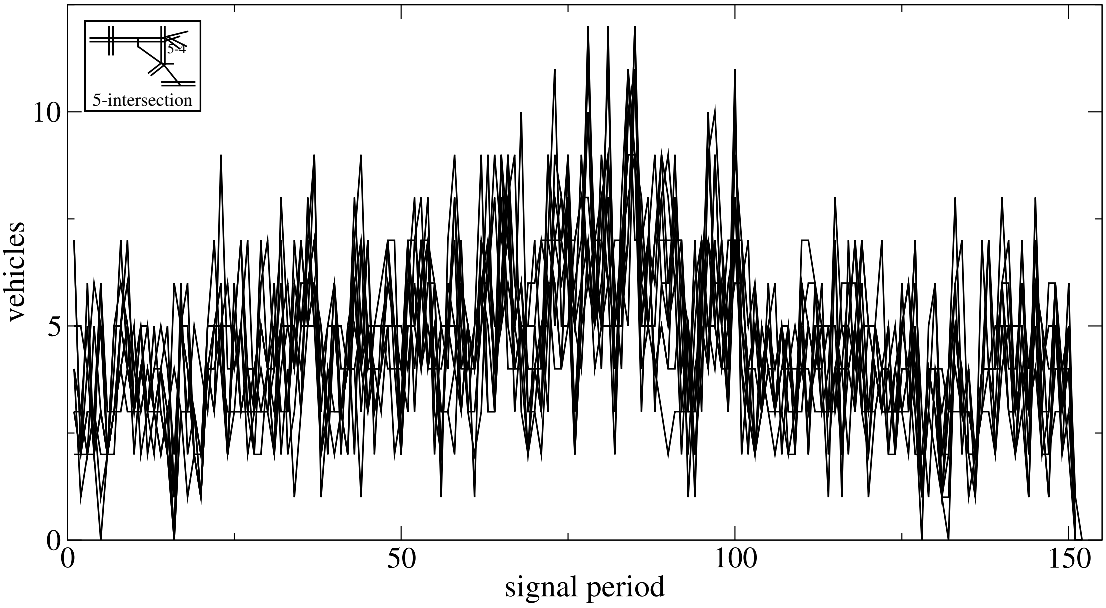


Figure B.51: Queue lengths for the 5-intersection case at 5-4; ten CA runs.

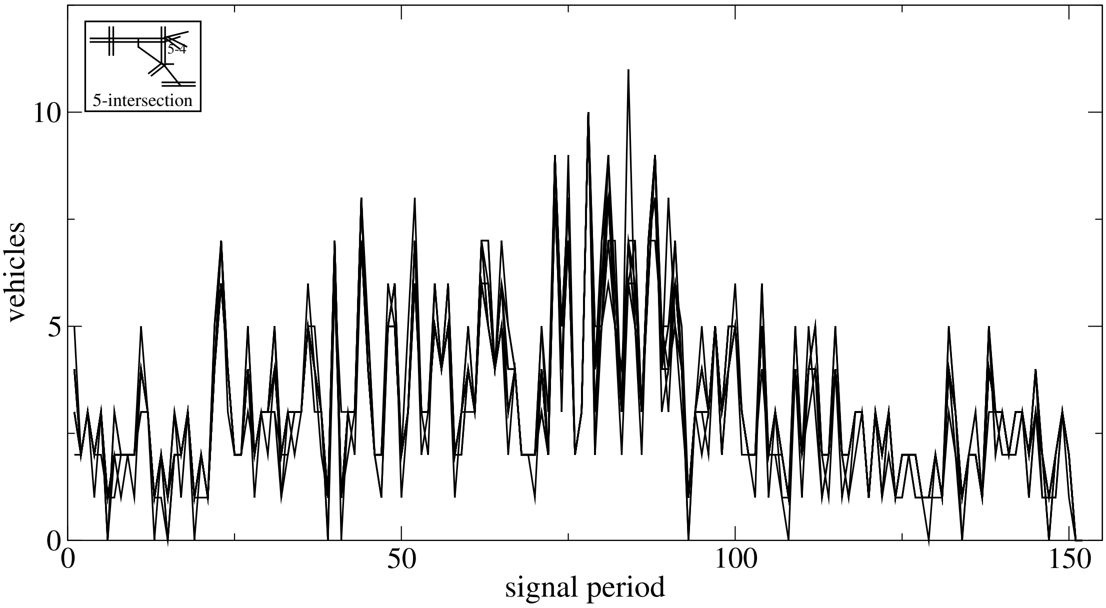


Figure B.52: Queue lengths for the 5-intersection case at 5-4; ten CA runs, extra detectors.

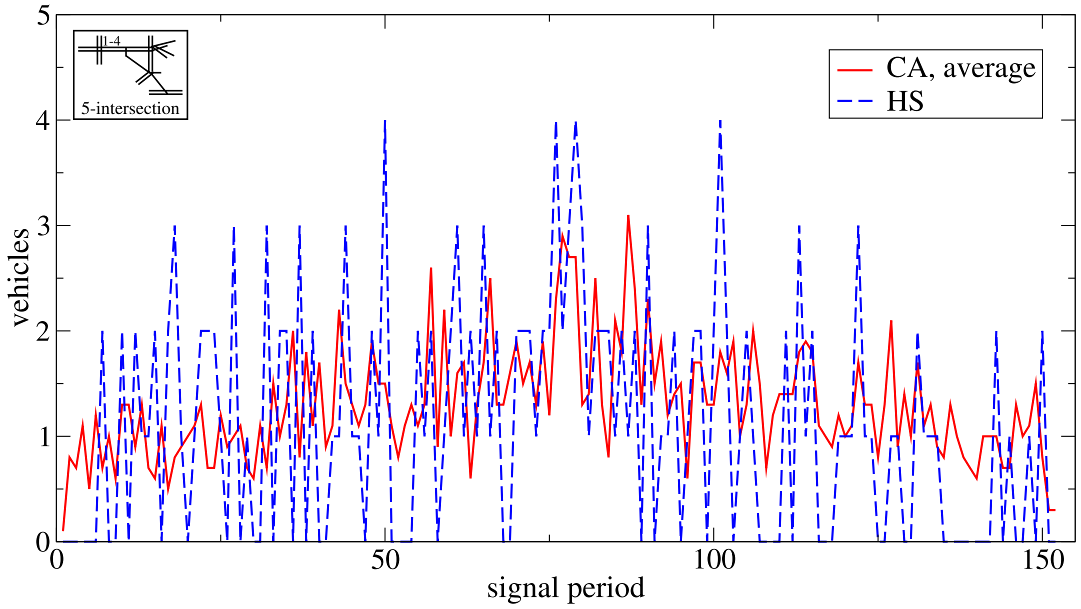


Figure B.53: Queue lengths for the 5-intersection case at 1-4; CA average of 10 different seed numbers.

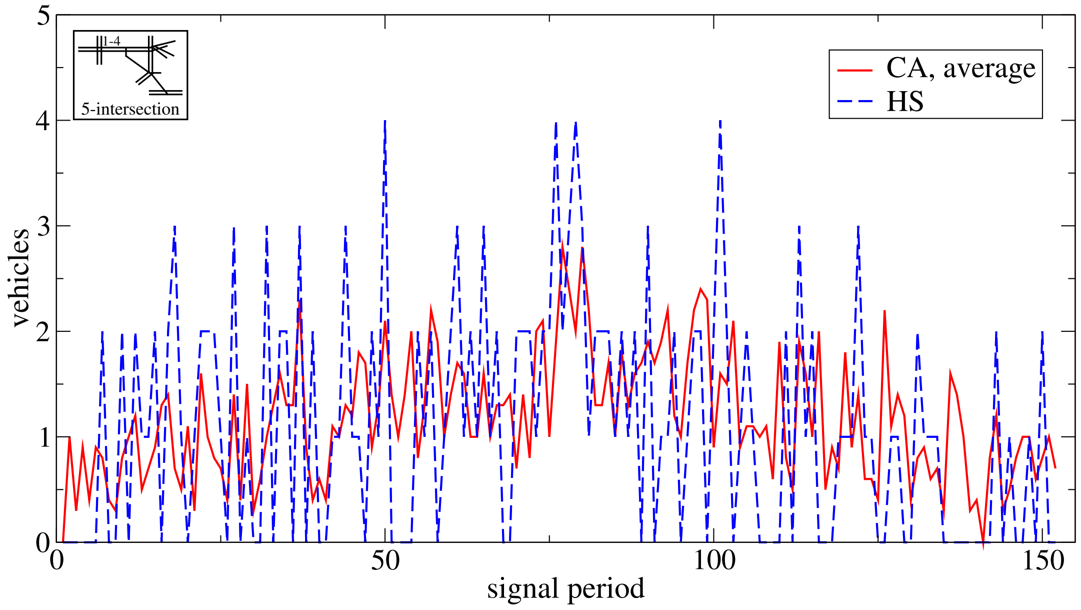


Figure B.54: Queue lengths for the 5-intersection case at 1-4; CA average of 10 different seed numbers, extra detectors.

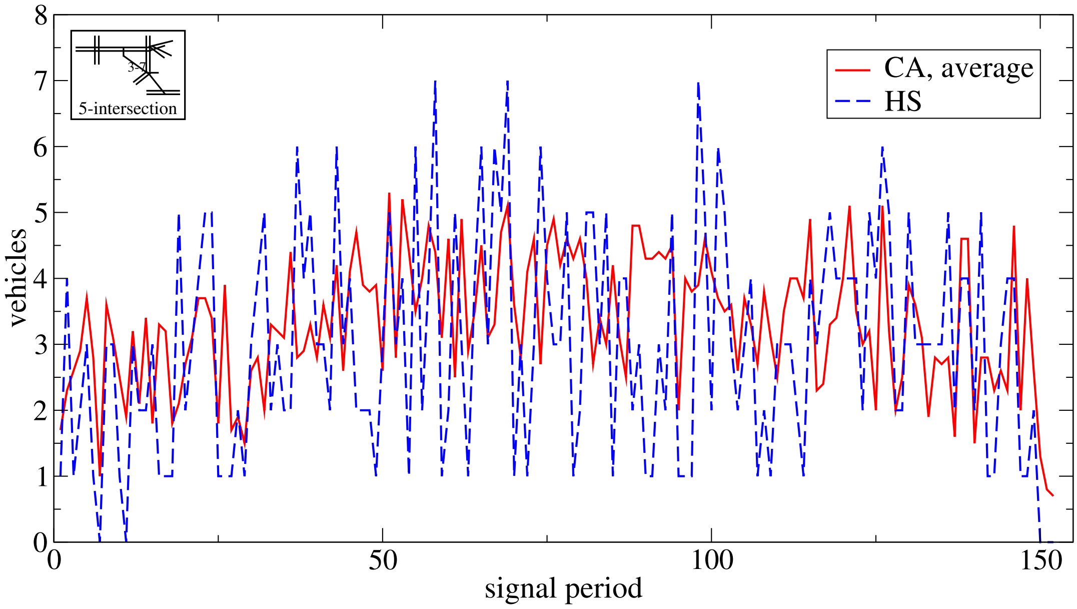


Figure B.55: Queue lengths for the 5-intersection case at 3-7; CA average of 10 different seed numbers.

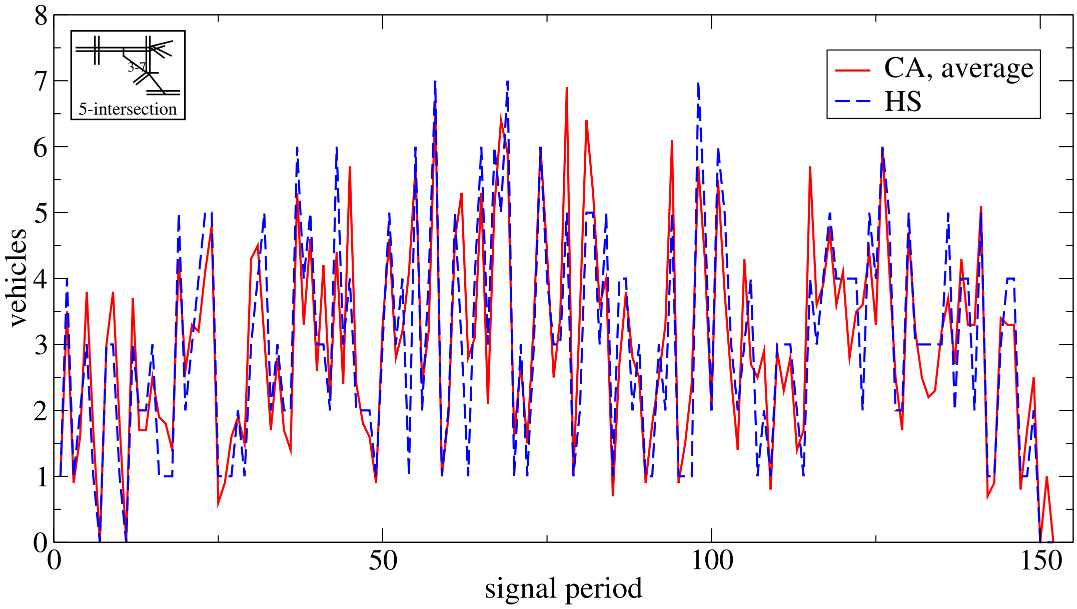


Figure B.56: Queue lengths for the 5-intersection case at 3-7; CA average of 10 different seed numbers, extra detectors.

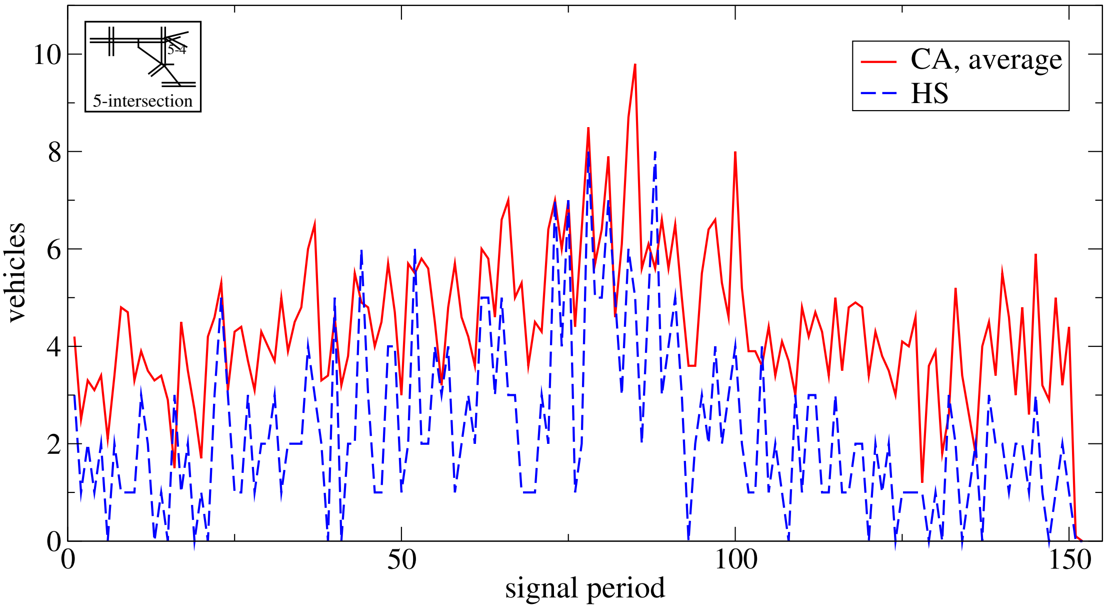


Figure B.57: Queue lengths for the 5-intersection case at 5-4; CA average of 10 different seed numbers.

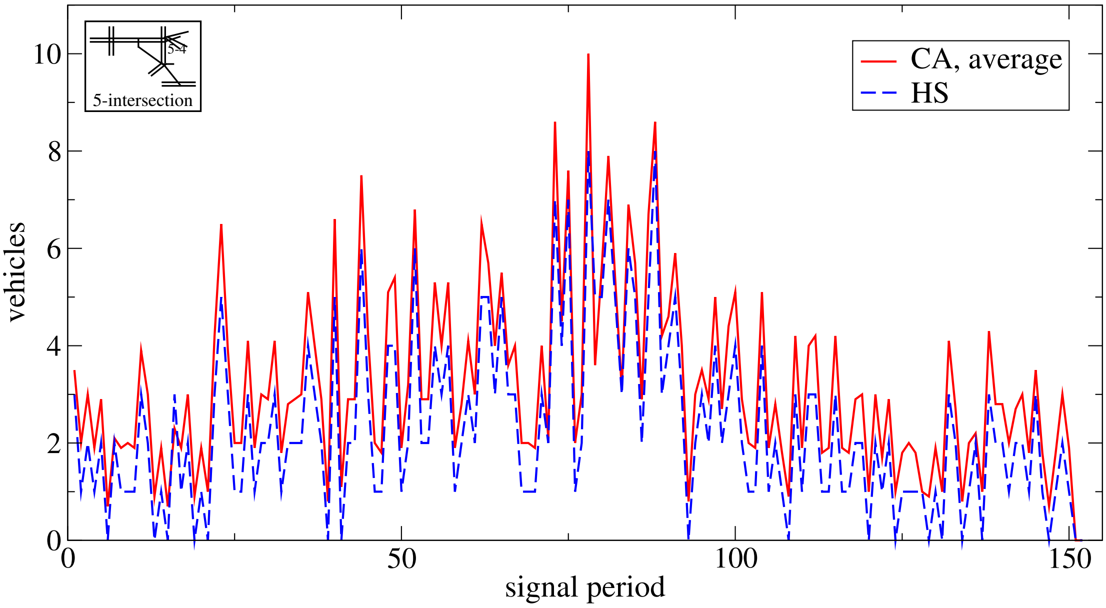


Figure B.58: Queue lengths for the 5-intersection case at 5-4; CA average of 10 different seed numbers, extra detectors.

CA run number	HUTSIM configfile (.cnf)	HUTSIM delayfile (.del)	HUTSIM detsigfile (.dat)	CA turn rate set (.prn)	CA seed	CA logfile (.log)	CA genparfile (.dat)
111	tyk_i_b2	hs5	ds5	cr5tu	23356	rm5_1	genpar1
112	tyk_i_b2	hs5	ds5	cr5tu	2332	rm5_11	— “ —
113	tyh_i_b2	hs5	ds5	cr5tu	10112	rm5_12	— “ —
114	tyk_i_b2	hs5	ds5	cr5tu	9930	rm5_13	— “ —
115	tyk_i_b2	hs5	ds5	cr5tu	381	rm5_14	— “ —
116	tyk_i_b2	hs5	ds5	cr5tu	3354	rm5_15	— “ —
117	tyk_i_b2	hs5	ds5	cr5tu	12001	rm5_16	— “ —
118	tyk_i_b2	hs5	ds5	cr5tu	1020	rm5_17	— “ —
119	tyk_i_b2	hs5	ds5	cr5tu	25460	rm5_18	— “ —
120	tyk_i_b2	hs5	ds5	cr5tu	19500	rm5_19	— “ —
121	tyk_i_b4		ds5d	cr5tu	23356	rd5_1	— “ —
122	tyk_i_b4		ds5d	cr5tu	2332	rd5_11	— “ —
123	tyh_i_b4		ds5d	cr5tu	10112	rd5_12	— “ —
124	tyk_i_b4		ds5d	cr5tu	9930	rd5_13	— “ —
125	tyk_i_b4		ds5d	cr5tu	381	rd5_14	— “ —
126	tyk_i_b4		ds5d	cr5tu	3354	rd5_15	— “ —
127	tyk_i_b4		ds5d	cr5tu	12001	rd5_16	— “ —
128	tyk_i_b4		ds5d	cr5tu	1020	rd5_17	— “ —
129	tyk_i_b4		ds5d	cr5tu	25460	rd5_18	— “ —
130	tyk_i_b4		ds5d	cr5tu	19500	rd5_19	— “ —

Table B.8: Key for HUTSIM and CA runs, the 5-intersection case, average velocities (111–120) and flow measurements (121–130).

origin	destination	detectors	figure	CA run number(s)
11	100		B.59	111 – 120
51	30		B.60	111 – 120
72	30		B.61	111 – 120
72	30		B.62	119
		151+152	B.63	121 – 130
		153+154	B.64	121 – 130

Table B.9: Key for figures and CA runs, the 5-intersection case, average velocities and flow measurements.

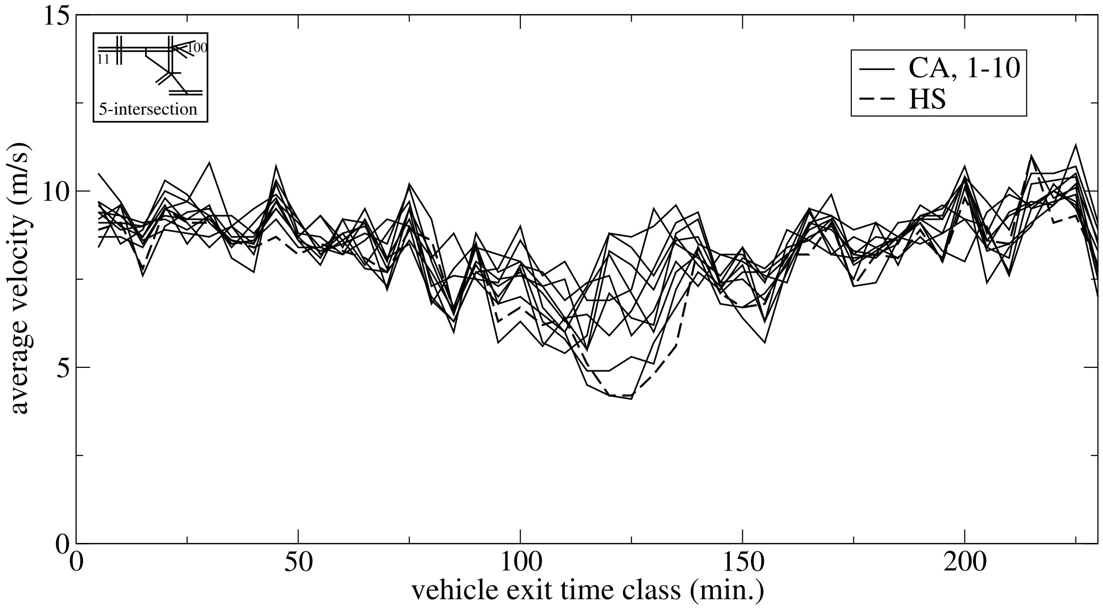


Figure B.59: Average velocity on route 11 → 100; 10 different seed numbers for CA vehicles.

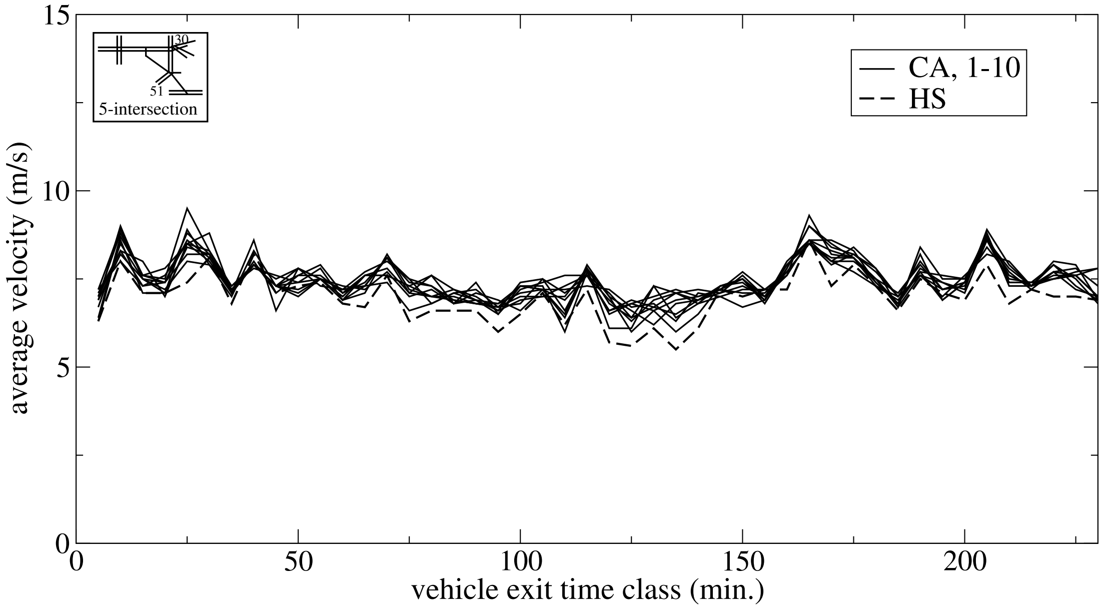


Figure B.60: Average velocity on route 51 → 30; 10 different seed numbers for CA vehicles.

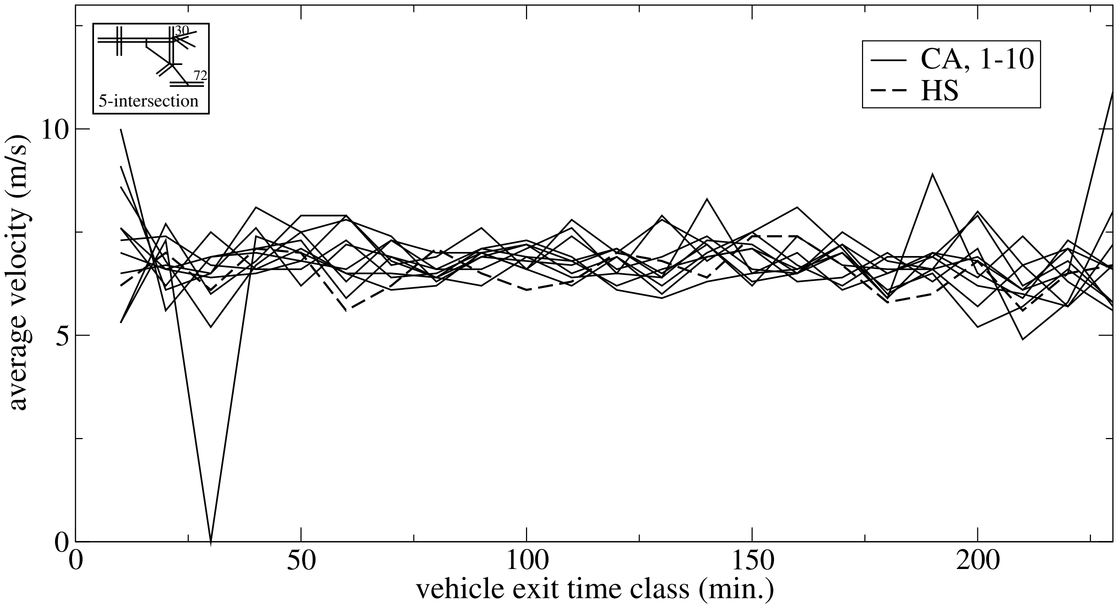


Figure B.61: Average velocity on route 72 → 30; 10 different seed numbers for CA vehicles.

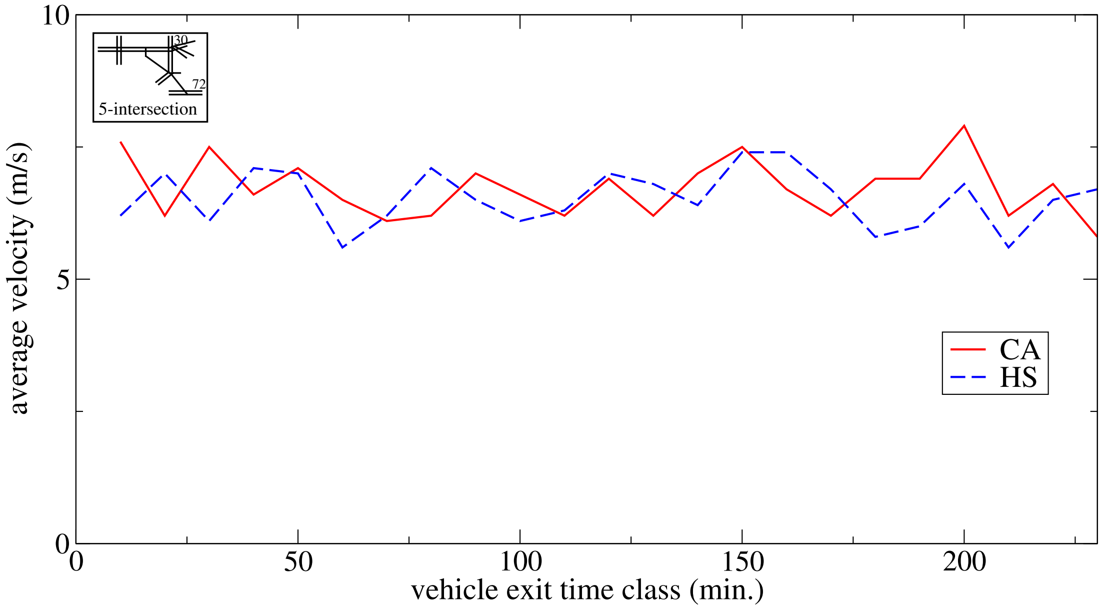


Figure B.62: Average velocity example on route 72 → 30; CA run 119.

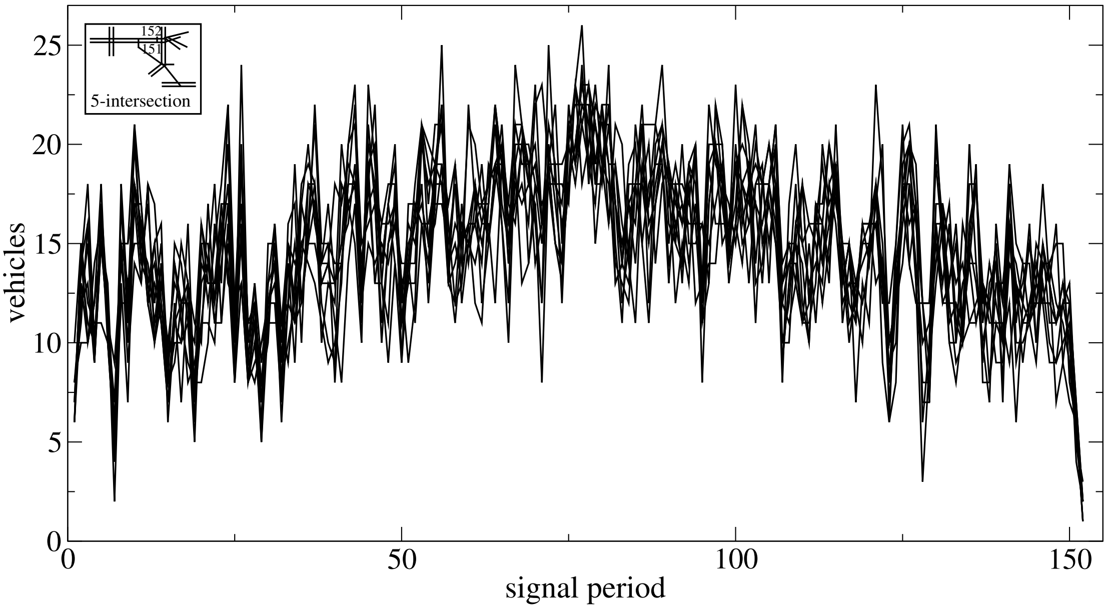


Figure B.63: Flow measurements at detectors 151+152; 10 different seed numbers for CA vehicles.

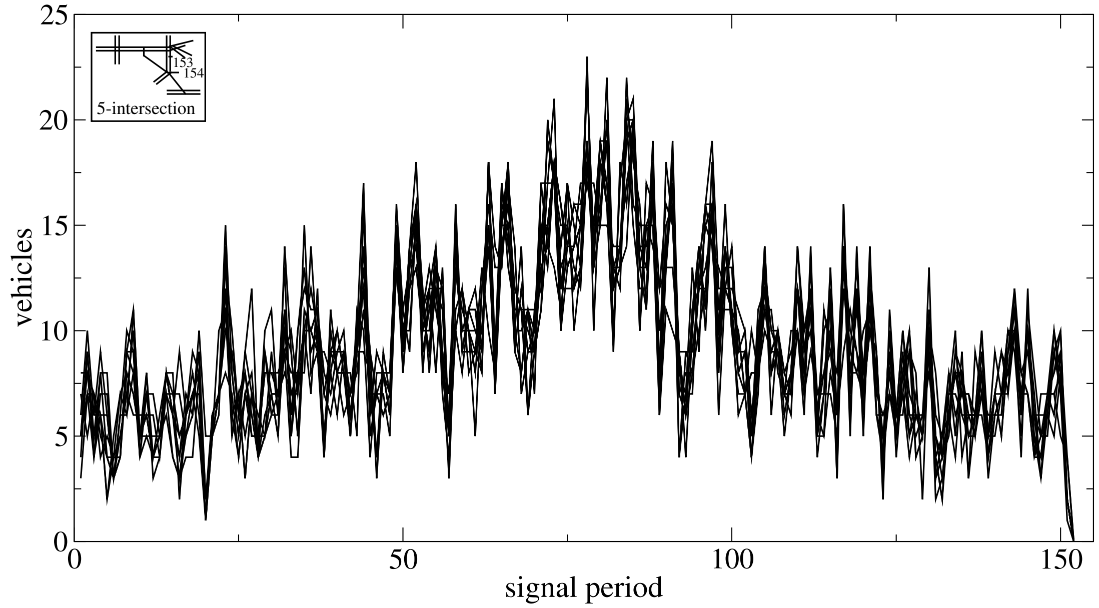
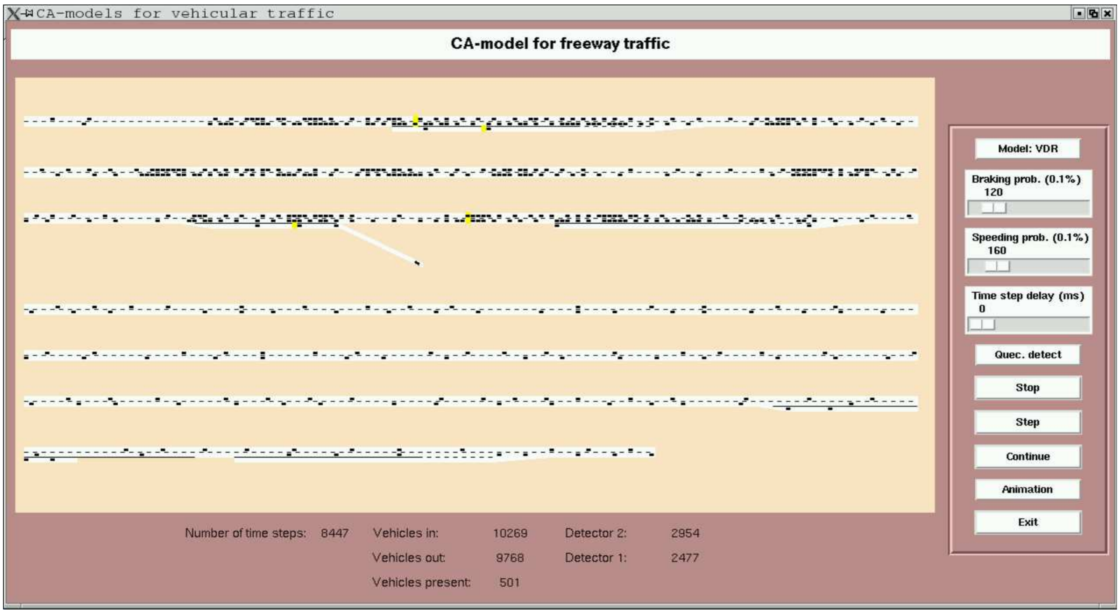


Figure B.64: Flow measurements at detectors 153+154; 10 different seed numbers for CA vehicles.

CA run number	HUTSIM arrival file (.ARR)	CA turn rate set (.prn)	CA seed	CA logfile (.log)	CA genparfile (.dat)
131	IH1	ihtu	23356	rf3_1	genpar3
132	IH1	ihtu	2332	rf3_11	— “ —
133	IH1	ihtu	10112	rf3_12	— “ —
134	IH1	ihtu	9930	rf3_13	— “ —
135	IH1	ihtu	381	rf3_14	— “ —
136	IH1	ihtu	3354	rf3_15	— “ —
137	IH1	ihtu	12001	rf3_16	— “ —
138	IH1	ihtu	1020	rf3_17	— “ —
139	IH1	ihtu	25460	rf3_18	— “ —
140	IH1	ihtu	19500	rf3_19	— “ —
141	IH2	ihtu2	23356	rf32_1	genpar3
142	IH2	ihtu2	2332	rf32_11	— “ —
143	IH2	ihtu2	10112	rf32_12	— “ —
144	IH2	ihtu2	9930	rf32_13	— “ —
145	IH2	ihtu2	381	rf32_14	— “ —
146	IH2	ihtu2	3354	rf32_15	— “ —
147	IH2	ihtu2	12001	rf32_16	— “ —
148	IH2	ihtu2	1020	rf32_17	— “ —
149	IH2	ihtu2	25460	rf32_18	— “ —
150	IH2	ihtu2	19500	rf32_19	— “ —
151	IH1	ihtu	23356	rf5_1	genpar2
152	IH1	ihtu	2332	rf5_11	— “ —
153	IH1	ihtu	10112	rf5_12	— “ —
154	IH1	ihtu	9930	rf5_13	— “ —
155	IH1	ihtu	381	rf5_14	— “ —
156	IH1	ihtu	3354	rf5_15	— “ —
157	IH1	ihtu	12001	rf5_16	— “ —
158	IH1	ihtu	1020	rf5_17	— “ —
159	IH1	ihtu	25460	rf5_18	— “ —
160	IH1	ihtu	19500	rf5_19	— “ —
161	IH2	ihtu2	23356	rf52_1	genpar2
162	IH2	ihtu2	2332	rf52_11	— “ —
163	IH2	ihtu2	10112	rf52_12	— “ —
164	IH2	ihtu2	9930	rf52_13	— “ —
165	IH2	ihtu2	381	rf52_14	— “ —
166	IH2	ihtu2	3354	rf52_15	— “ —
167	IH2	ihtu2	12001	rf52_16	— “ —
168	IH2	ihtu2	1020	rf52_17	— “ —
169	IH2	ihtu2	25460	rf52_18	— “ —
170	IH2	ihtu2	19500	rf52_19	— “ —

Table B.10: Key for CA runs, the freeway case, average velocities.



CA-models for vehicular traffic

CA-model for freeway traffic

Model: VDR

Braking prob. (0.1%) 120

Speeding prob. (0.1%) 160

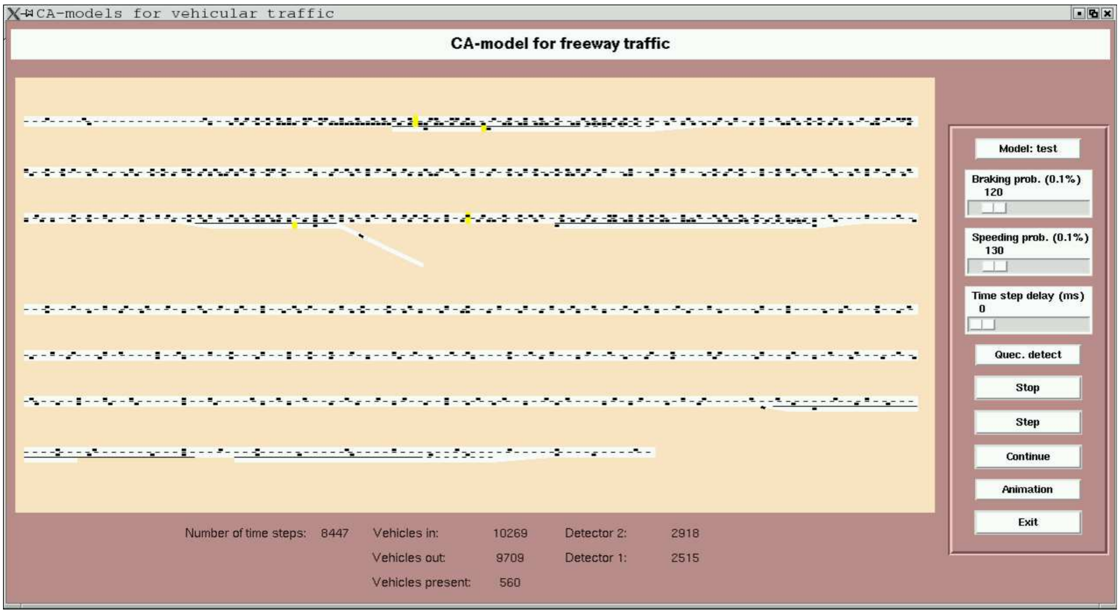
Time step delay (ms) 0

Number of time steps: 8447 Vehicles in: 10269 Detector 2: 2954

Vehicles out: 9768 Detector 1: 2477

Vehicles present: 501

Figure B.65: A snapshot from simulation (VDR, run 138).



CA-models for vehicular traffic

CA-model for freeway traffic

Model: test

Braking prob. (0.1%) 120

Speeding prob. (0.1%) 130

Time step delay (ms) 0

Number of time steps: 8447 Vehicles in: 10269 Detector 2: 2918

Vehicles out: 9709 Detector 1: 2515

Vehicles present: 560

Figure B.66: A snapshot from simulation (test model, run 151).

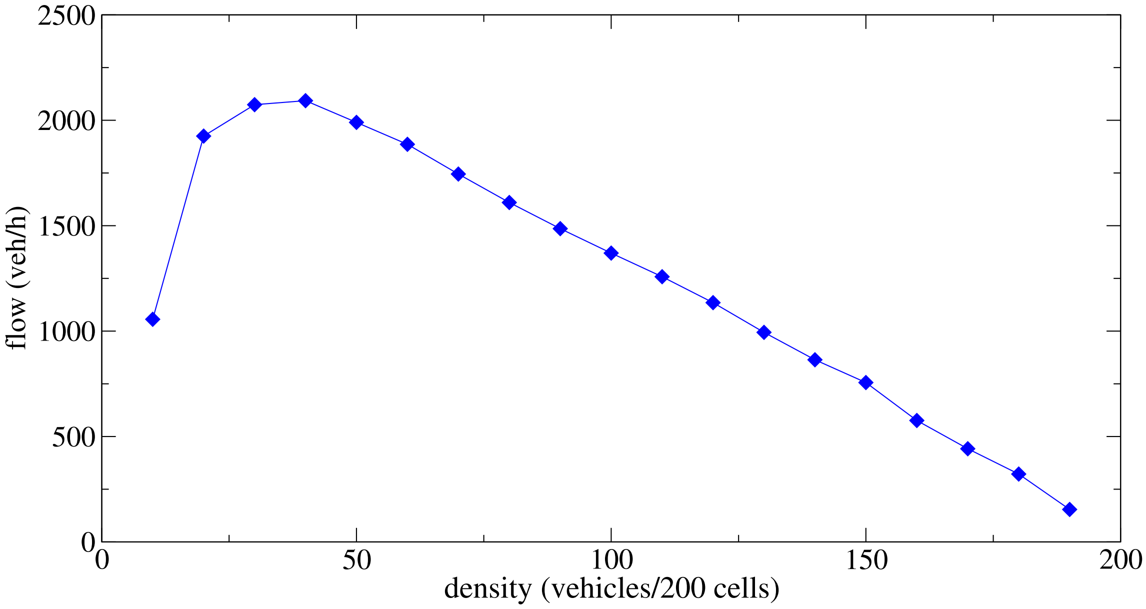


Figure B.67: Flow versus density for the test model.

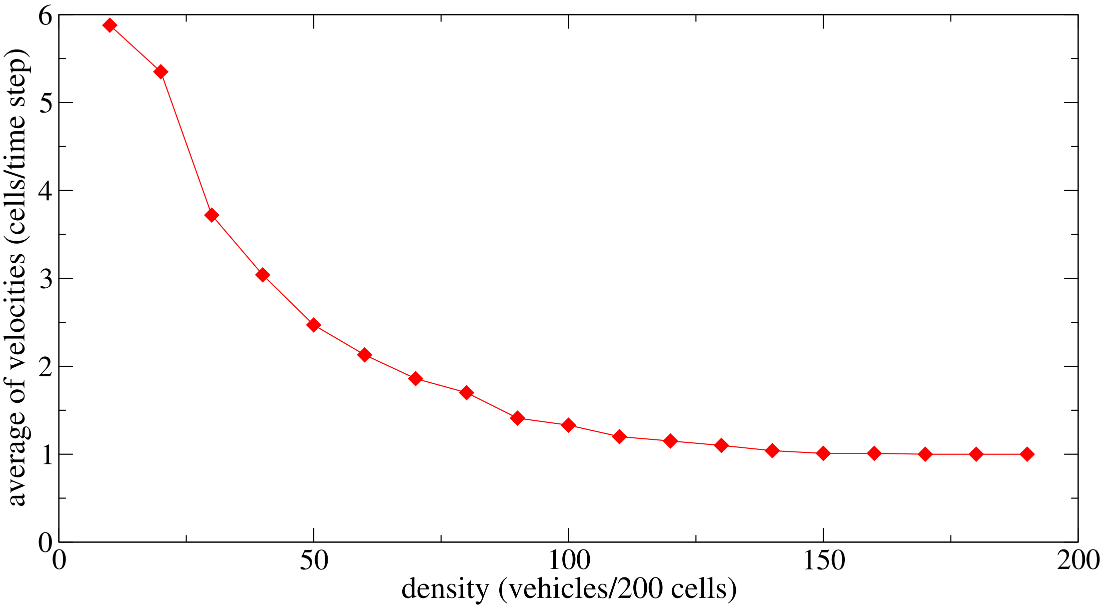


Figure B.68: Average of velocities versus density for the test model.

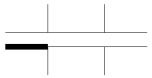
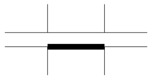
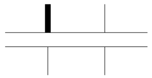
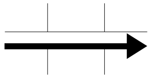
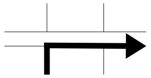
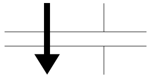
QUEUE LENGTHS			
2-intersection	 on two lanes, signal head 1-1	 on two lanes, signal head 2-1	 on one lane, signal head 1-6
turn rate 1	consistency between different CA runs usually very good, at congestion peaks quite good consistency between the CA average and the HS run usually very good	consistency between different CA runs often very good, at congestion peak reasonable (vehicle amounts at the peak area are very large) consistency between the CA average and the HS run usually very good, at congestion peak differences arise	consistency between different CA runs often very good, but at congestion peaks poor although the vehicle amounts are rather low consistency between the CA average and the HS run usually very good
turn rate 2	consistency between different CA runs quite good consistency between the CA average and the HS run is good	consistency between different CA runs quite good, at congestion area considerable differences arise consistency between the CA average and the HS result usually quite good	consistency between different CA runs quite good, but poor at peaks consistency between the CA average and the HS result usually quite good, also at congestion peaks
turn rate 3	consistency between different CA runs usually good, at congestion peaks differences may be significant consistency between the CA average and the HS run quite good, at congestion area differences arise	consistency between different CA runs reasonable, at congestion peak large differences arise consistency between the CA average and the HS run usually quite good	consistency between different CA runs reasonable, at a high congestion peak the differences may be significant consistency between the CA average and the HS run rather good, at peaks somewhat worse
turn rate 4	consistency between different CA runs usually quite good, at congestion peaks reasonable (but peak heights are lower than for other turn rate sets) consistency between the CA average and the HS run quite good	consistency between different CA runs not very good, not even outside peaks consistency between the CA average and the HS run reasonable	consistency between different CA runs quite poor although the vehicle amounts are low considerable differences between the CA average and the HS run graphs

Figure B.69: Summary of the queue length measurements for the 2-intersection case.

AVERAGE VELOCITIES			
2-intersection	 route 11 -> 2	 route 31 -> 2	 route 41 -> 3
turn rate 1	congruence between different CA runs quite good, even at the congestion peak area the graphs of different CA runs match quite well with the HS graph	CA runs give quite similar averige velocity values, only around the flow maximum differences arise (especially at recovery from congestion) congruence between the CA graphs and the HS graph is quite good, after the flow peak the CA values are lower	the CA run graphs match quite well with one another, but after the flow maximum large differences arise congruence between the CA graphs and the HS graph is usually good (with the exception of the area after the flow maximum)
turn rate 2	dispersion between different CA run graphs (especially after the flow peak) the CA runs give usually somewhat higher velocity values than the HS run does	the results for the CA runs differ from each other considerably in places (particularly around the flow peak) the CA values differ considerably from the HS values being usually higher	the CA run graphs differ considerably from each other especially after the flow maximum most of the CA runs give average velocities quite close to or a bit higher than the ones the HS run gives
turn rate 3	congruence between different CA runs quite good, around congestion area differences arise the CA runs give usually somewhat higher velocity values than the HS run, especially after the low maximum	differences between the CA values are considerable, also outside the flow maximum area the CA values are usually higher than the HS values	differences between the CA graphs are usually considerable, also outside the flow peak area the HS graph follows usually quite well the average level of the CA graphs
turn rate 4	the CA graphs differ somewhat from each other the HS graph blends well with the CA graphs	differences between the CA runs are somewhat large the CA runs give usually higher average velocity values than the HS run does	the differences between the CA runs are considerable the average level of the CA graphs again follows quite well the HS graph

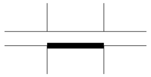
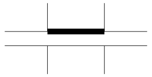
TRAFFIC FLOW		
2-intersection	 detectors 71+72	 detectors 81+82
turn rate 3	at higher vehicle amounts considerable deviations between different CA graphs, elsewhere congruence quite good the average CA graph follows quite well the HS graph	the results are quite similar to those of the other detector pair

Figure B.70: Summary of the average velocity and flow value measurements for the 2-intersection case.

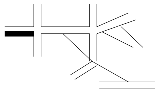
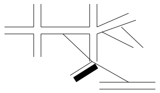
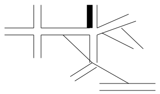
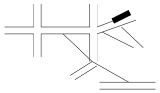
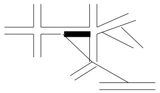
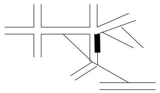
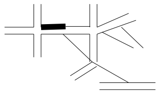
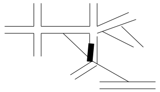
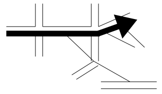
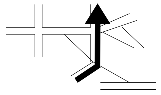
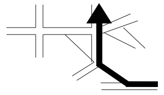
QUEUE LENGTHS				
5-intersection	 queues at 1-2	 queues at 3-3	 queues at 5-1	 queues at 5-11
	the CA graphs behave quite consistently, but large deviations at the congestion peak the average CA graph follows the HS graph well outside the congestion area	congruence of the CA graphs excellent consistency between the CA average graph and the HS graph excellent	congruence of the CA graphs usually excellent consistency between the CA average graph and the HS graph excellent	congruence of the CA graphs excellent except for some peaks consistency between the CA average graph and the HS graph usually very good
	 queues at 5-3	 queues at 5-4	 queues at 1-4	 queues at 3-7
	consistency between the CA graphs is rather poor, the extra detectors improve the results a bit the extra detectors improve the consistency between the average CA graph and the HS graph very much	consistency between the CA graphs is rather poor, the extra detectors improve the results very much the extra detectors improve the consistency between the average CA graph and the HS graph considerably	for several reasons the CA graphs differ much from each other, and the extra detectors do not improve the results the extra detectors do not improve the consistency between the average CA graph and the HS graph	the extra detectors improve the consistency of the CA graphs very much the extra detectors improve the consistency between the average CA graph and the HS graph very much

Figure B.71: Summary of the average queue length measurements for the 5-intersection case.

AVERAGE VELOCITIES			
5-intersection	 route 11 -> 100	 route 51 -> 30	 route 72 -> 30
	considerable differences between the CA graphs the CA graphs follow the HS graph usually reasonably well, but they often give higher velocity values than the HS graph	congruence of the CA graphs rather good consistency between the CA graphs and the HS graph is rather good although the CA velocity values are somewhat higher than the HS values	congruence of the CA graphs is rather poor (vehicle amounts are very low) consistency between the CA graphs and the HS graph is not good, but the velocity levels match quite well

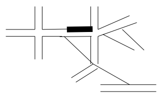
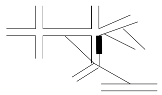
TRAFFIC FLOW		
5-intersection	 detectors 151+152	 detectors 153+154
	considerable differences between the separate CA graphs regardless of vehicle amounts the average CA graph corresponds reasonably well to the HS graph	the separate CA graphs are rather congruent both at lower and higher vehicle amounts the average CA graph corresponds quite well to the HS graph

Figure B.72: Summary of the average velocity and flow value measurements for the 5-intersection case.

CA run numbers	model	VMAX	flow (veh/h)	max queue logfile (.log)	logfiles (*=1-20) (.log)	CA genparfile
171 – 190	VDR	3	100	c3_100	c3_100_r*	genpar_3
191 – 210	VDR	3	300	c3_300	c3_300_r*	genpar_3
211 – 230	VDR	3	500	c3_500	c3_500_r*	genpar_3
231 – 250	VDR	3	700	c3_700	c3_700_r*	genpar_3
251 – 270	VDR	3	900	c3_900	c3_900_r*	genpar_3
271 – 290	VDR	3	1100	c3_1100	c3_1100_r*	genpar_3
291 – 310	VDR	3	1300	c3_1300	c3_1300_r*	genpar_3
311 – 330	VDR	3	1500	c3_1500	c3_1500_r*	genpar_3
331 – 350	VDR	3	1700	c3_1700c	c3_1700c_r*	genpar_3
351 – 370	VDR	3	1800	c3_1800d	c3_1800d_r*	genpar_3
371 – 390	VDR	3	100	c3h1_100	c3h1_100_r*	genpar_3h1
391 – 410	VDR	3	300	c3h1_300	c3h1_300_r*	genpar_3h1
411 – 430	VDR	3	500	c3h1_500	c3h1_500_r*	genpar_3h1
431 – 450	VDR	3	700	c3h1_700	c3h1_700_r*	genpar_3h1
451 – 470	VDR	3	900	c3h1_900	c3h1_900_r*	genpar_3h1
471 – 490	VDR	3	1100	c3h1_1100	c3h1_1100_r*	genpar_3h1
491 – 510	VDR	3	1300	c3h1_1300	c3h1_1300_r*	genpar_3h1
511 – 530	VDR	3	1500	c3h1_1500	c3h1_1500_r*	genpar_3h1
531 – 550	VDR	3	1700	c3h1_1700	c3h1_1700_r*	genpar_3h1
551 – 570	VDR	3	1850	c3h1_1850d	c3h1_1850d_r*	genpar_3h1
571 – 590	VDR	5	100	c35_100	c35_100_r*	genpar_35
591 – 610	VDR	5	300	c35_300	c35_300_r*	genpar_35
611 – 630	VDR	5	500	c35_500	c35_500_r*	genpar_35
631 – 650	VDR	5	700	c35_700	c35_700_r*	genpar_35
651 – 670	VDR	5	900	c35_900	c35_900_r*	genpar_35
671 – 690	VDR	5	1100	c35_1100	c35_1100_r*	genpar_35
691 – 710	VDR	5	1300	c35_1300	c35_1300_r*	genpar_35
711 – 730	VDR	5	1500	c35_1500	c35_1500_r*	genpar_35
731 – 750	VDR	5	1700	c35_1700	c35_1700_r*	genpar_35
751 – 770	VDR	5	1900	c35_1900c	c35_1900c_r*	genpar_35
771 – 790	VDR	5	2000	c35_2000d	c35_2000d_r*	genpar_35
791 – 810	NaSch	5	100	c15_100	c15_100_r*	genpar_15
811 – 830	NaSch	5	300	c15_300	c15_300_r*	genpar_15
831 – 850	NaSch	5	500	c15_500	c15_500_r*	genpar_15
851 – 870	NaSch	5	700	c15_700	c15_700_r*	genpar_15
871 – 890	NaSch	5	900	c15_900	c15_900_r*	genpar_15
891 – 910	NaSch	5	1100	c15_1100	c15_1100_r*	genpar_15
911 – 930	NaSch	5	1300	c15_1300	c15_1300_r*	genpar_15
931 – 950	NaSch	5	1500	c15_1500	c15_1500_r*	genpar_15
951 – 970	NaSch	5	1700	c15_1700	c15_1700_r*	genpar_15
971 – 990	NaSch	5	1900	c15_1900	c15_1900_r*	genpar_15
991 – 1010	NaSch	5	2100	c15_2100	c15_2100_r*	genpar_15
1011 – 1030	NaSch	5	2300	c15_2300c	c15_2300c_r*	genpar_15
1031 – 1050	NaSch	5	2500	c15_2500c	c15_2500c_r*	genpar_15
1051 – 1070	test	5	100	c55_100	c55_100_r*	genpar_55
1071 – 1090	test	5	300	c55_300	c55_300_r*	genpar_55
1091 – 1110	test	5	500	c55_500	c55_500_r*	genpar_55
1111 – 1130	test	5	700	c55_700	c55_700_r*	genpar_55
1131 – 1150	test	5	900	c55_900	c55_900_r*	genpar_55
1151 – 1170	test	5	1100	c55_1100	c55_1100_r*	genpar_55
1171 – 1190	test	5	1300	c55_1300	c55_1300_r*	genpar_55
1191 – 1210	test	5	1500	c55_1500	c55_1500_r*	genpar_55
1211 – 1230	test	5	1700	c55_1700	c55_1700_r*	genpar_55
1231 – 1250	test	5	1900	c55_1900	c55_1900_r*	genpar_55
1251 – 1270	test	5	2000	c55_2000d	c55_2000d_r*	genpar_55

Table B.11: Key for CA runs, general feature calculations.

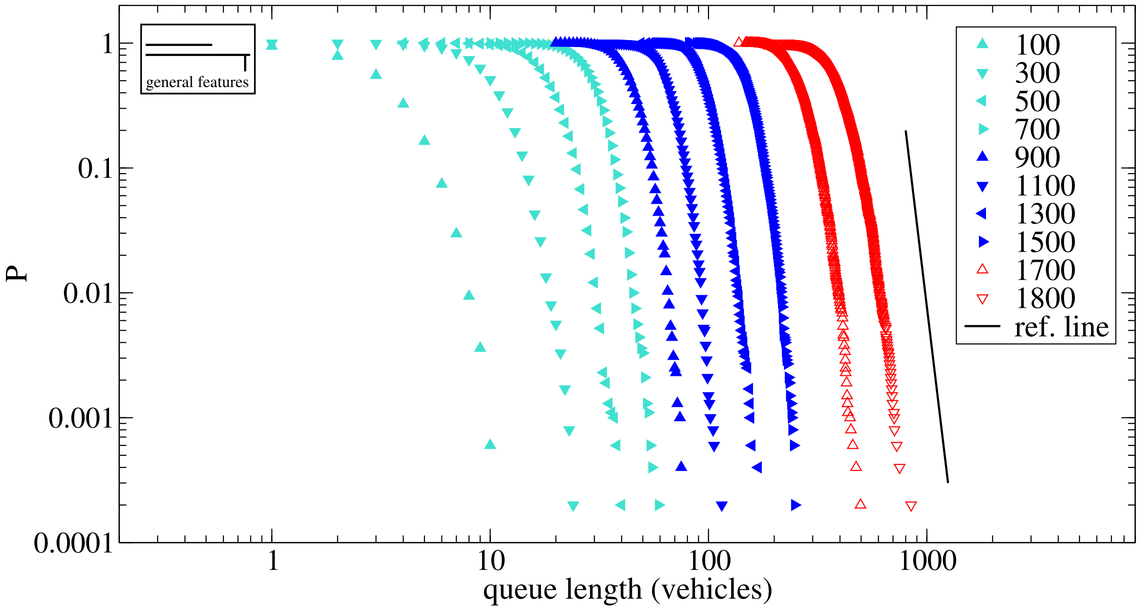


Figure B.73: Probability for a queue length to exceed a specified value (VDR, VMAX=3, flow=100–1800 veh/h, CA runs 171–370).

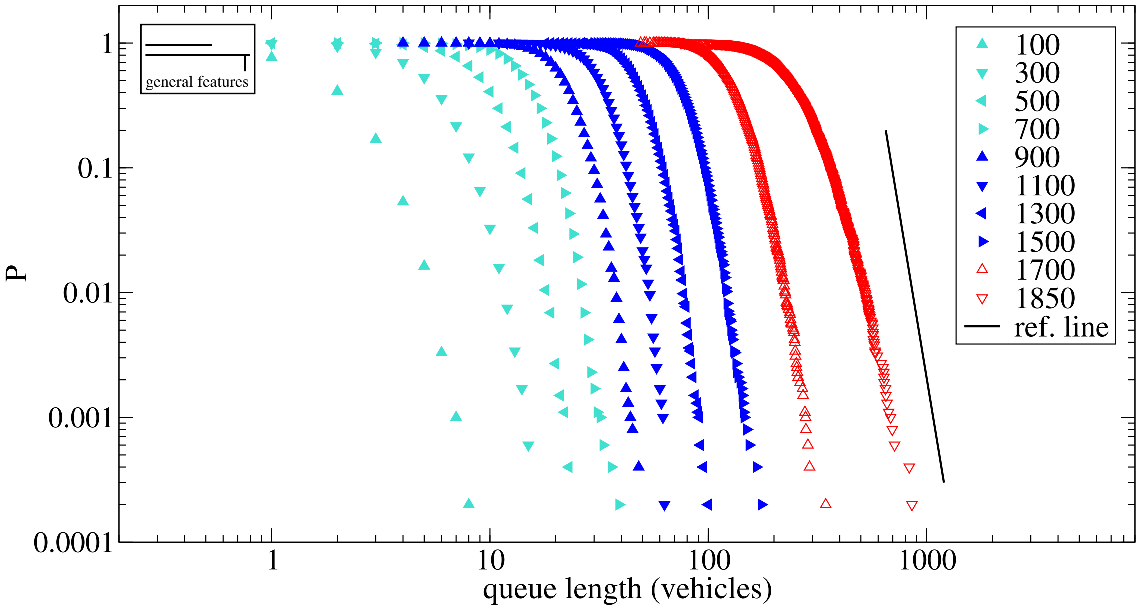


Figure B.74: Probability for a queue length to exceed a specified value (VDR, VMAX=3, flow=100–1850 veh/h, CA runs 371–570), red=50.

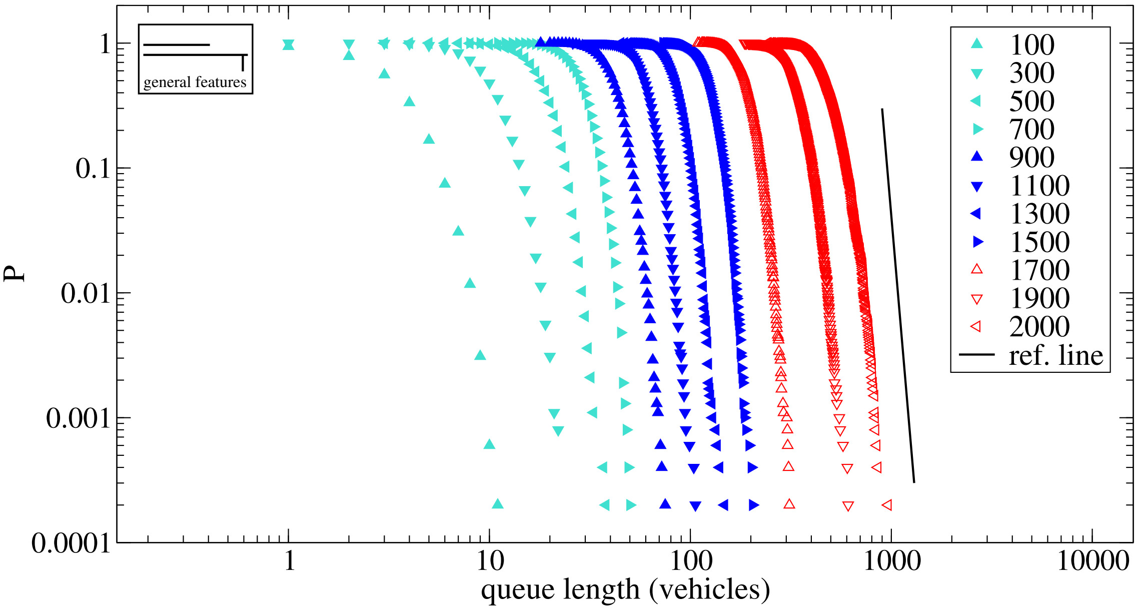


Figure B.75: Probability for a queue length to exceed a specified value (VDR, VMAX=5, flow=100–2000 veh/h, CA runs 571–790).

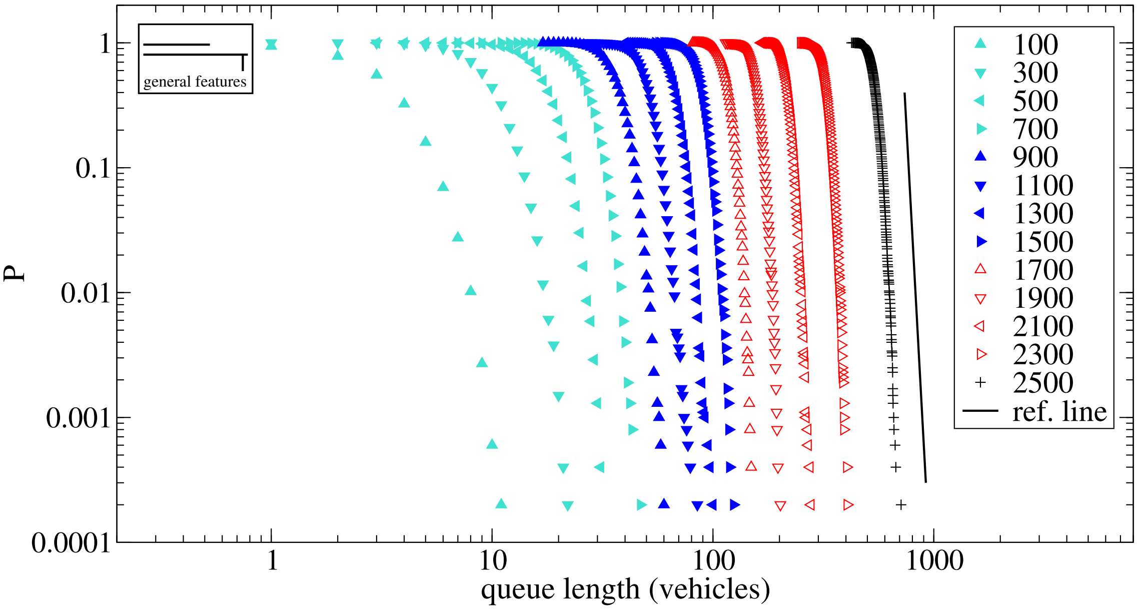


Figure B.76: Probability for a queue length to exceed a specified value (NaSch, VMAX=5, flow=100–2500 veh/h, CA runs 791–1050).

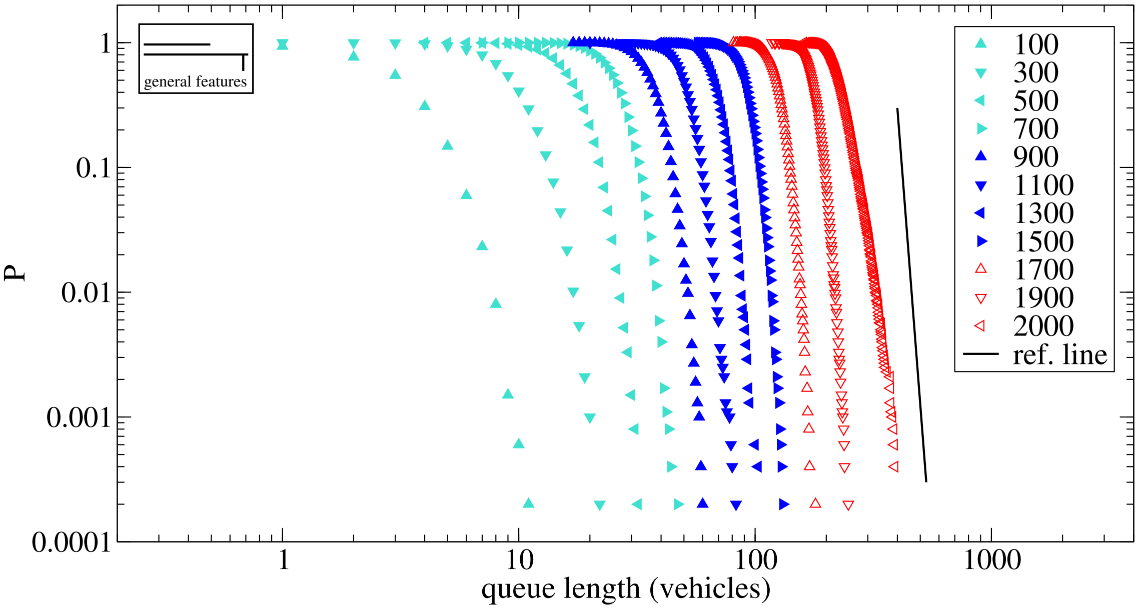


Figure B.77: Probability for a queue length to exceed a specified value (test model, VMAX=5, flow=100–2000 veh/h, CA runs 1051–1270).

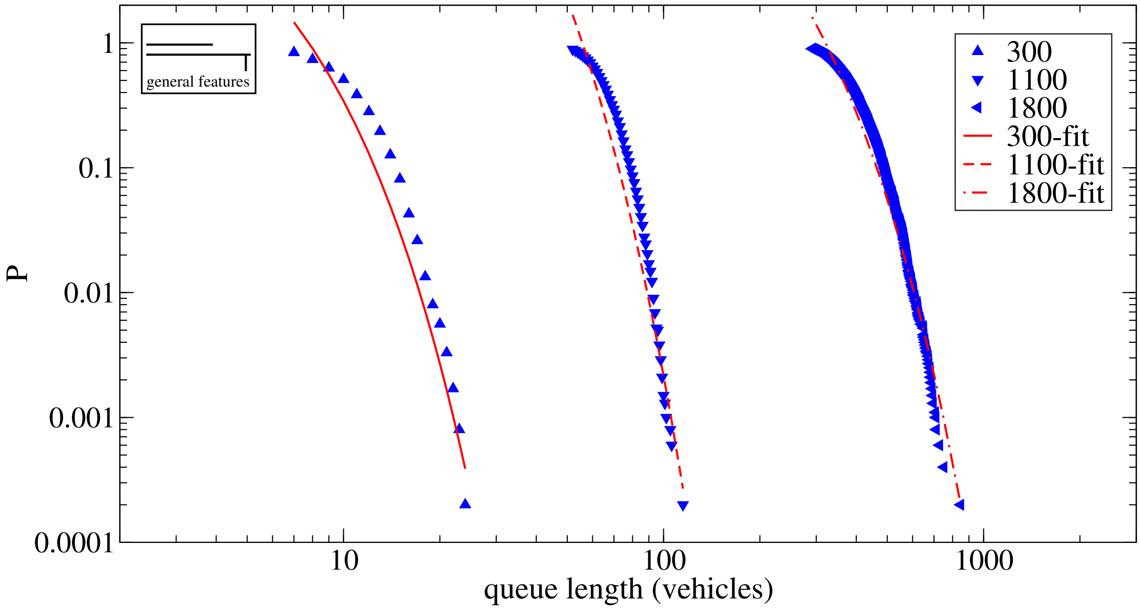


Figure B.78: Exponential fits for the VDR model (VMAX=3) runs.

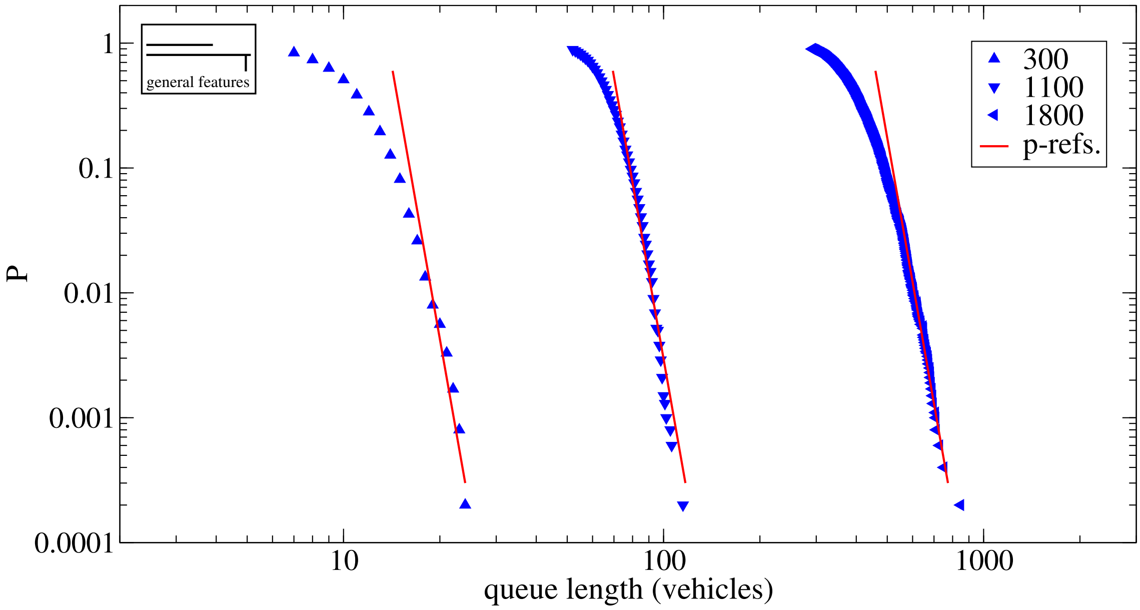


Figure B.79: Power law references for the VDR model (VMAX=3) runs.

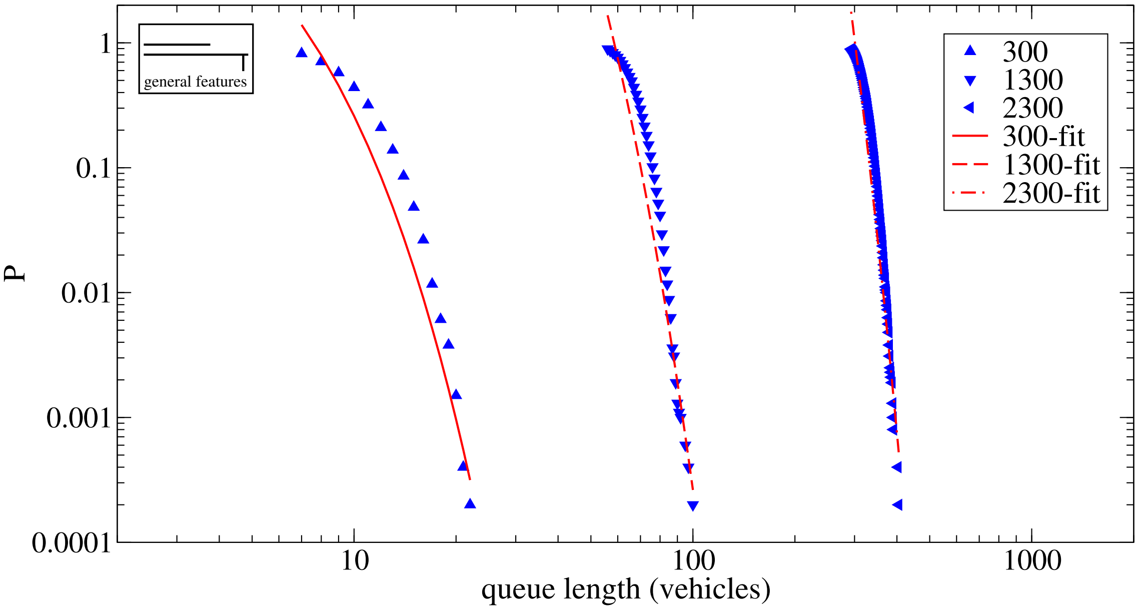


Figure B.80: Exponential fits for the NaSch model (VMAX=5) runs.

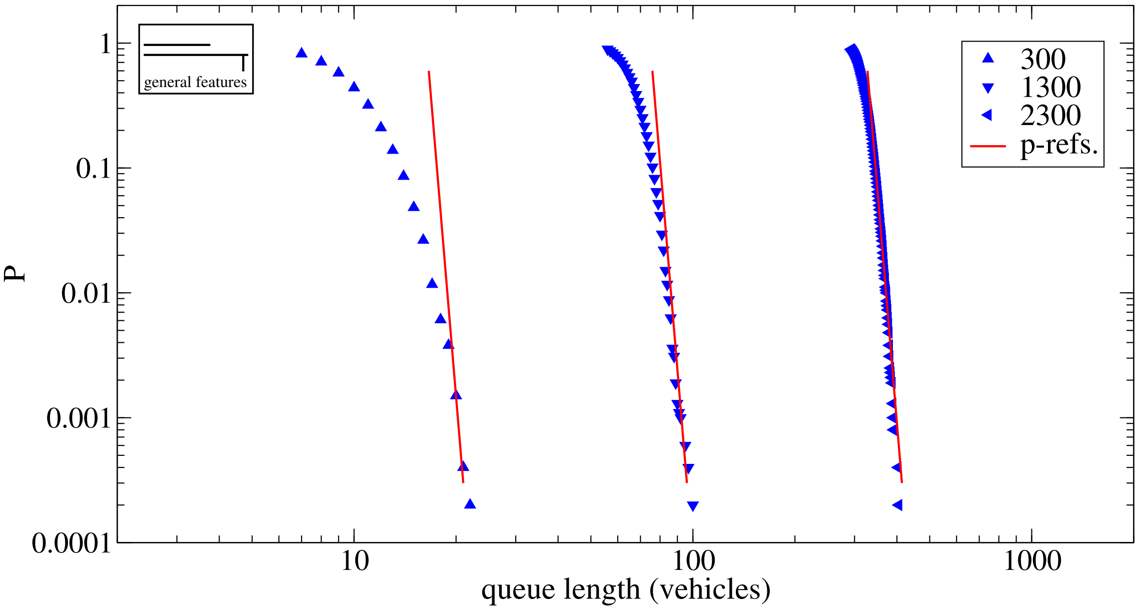


Figure B.81: Power law references for the NaSch model (VMAX=5) runs.

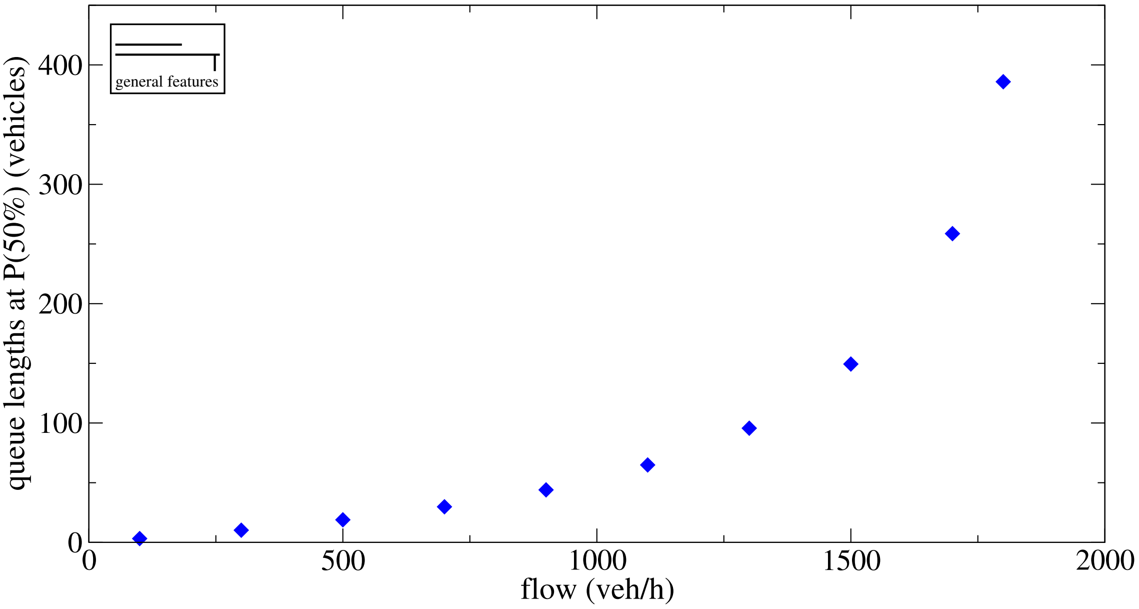


Figure B.82: Queue lengths at P(50%) for the VDR (VMAX=3) model runs.

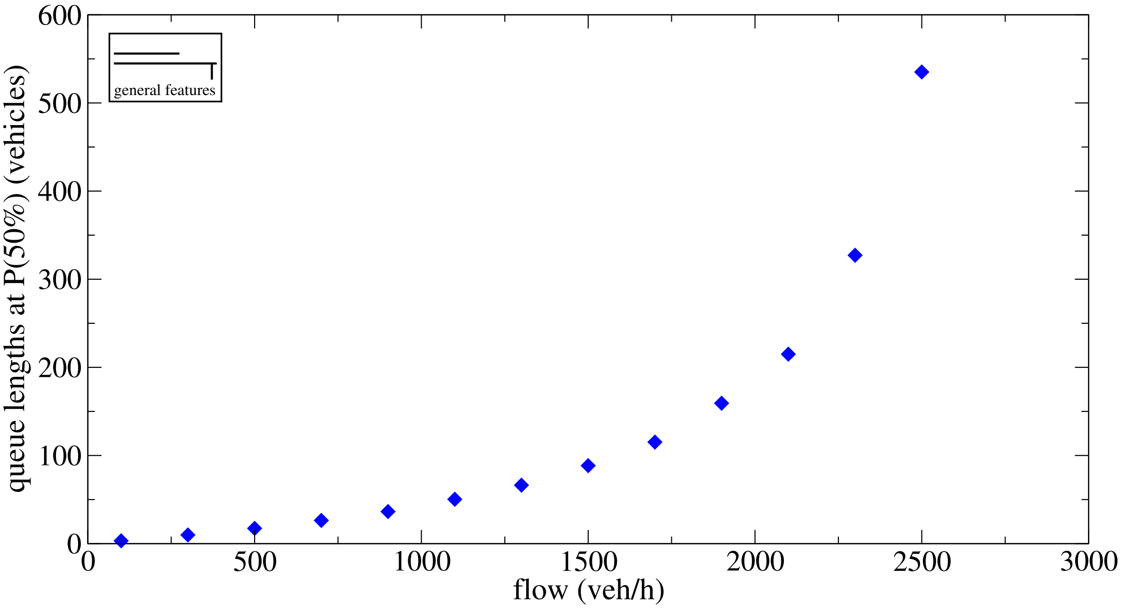


Figure B.83: Queue lengths at P(50%) for the NaSch (VMAX=5) model runs.

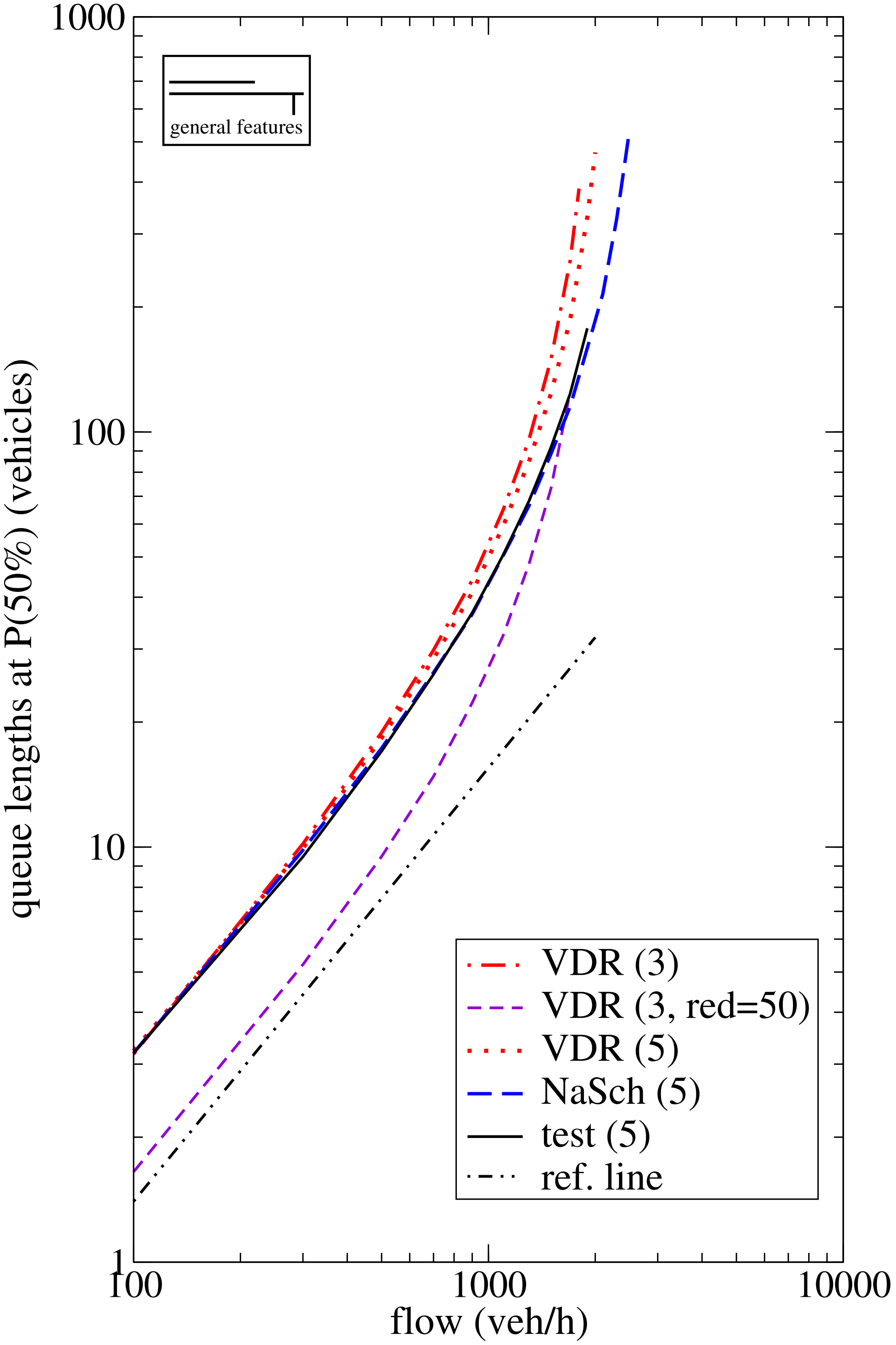


Figure B.84: Queue lengths at P(50%), VMAX-value in parentheses.

Appendix C

Program description

This appendix contains CA simulator program information. Fig. C.1 contains a unified (rough) flow chart of the C- and Tcl/Tk-parts of the CA simulator. The often used files associated with CA runs are depicted in Fig. C.2, and Fig. C.3 describes the most important internal array names in the program and their connection to input files. A list and a short description of the functions in the C-part of the program can be found in Tables C.1–C.2, and a similar list of the procedures in the Tcl/Tk-part is in Table C.3.

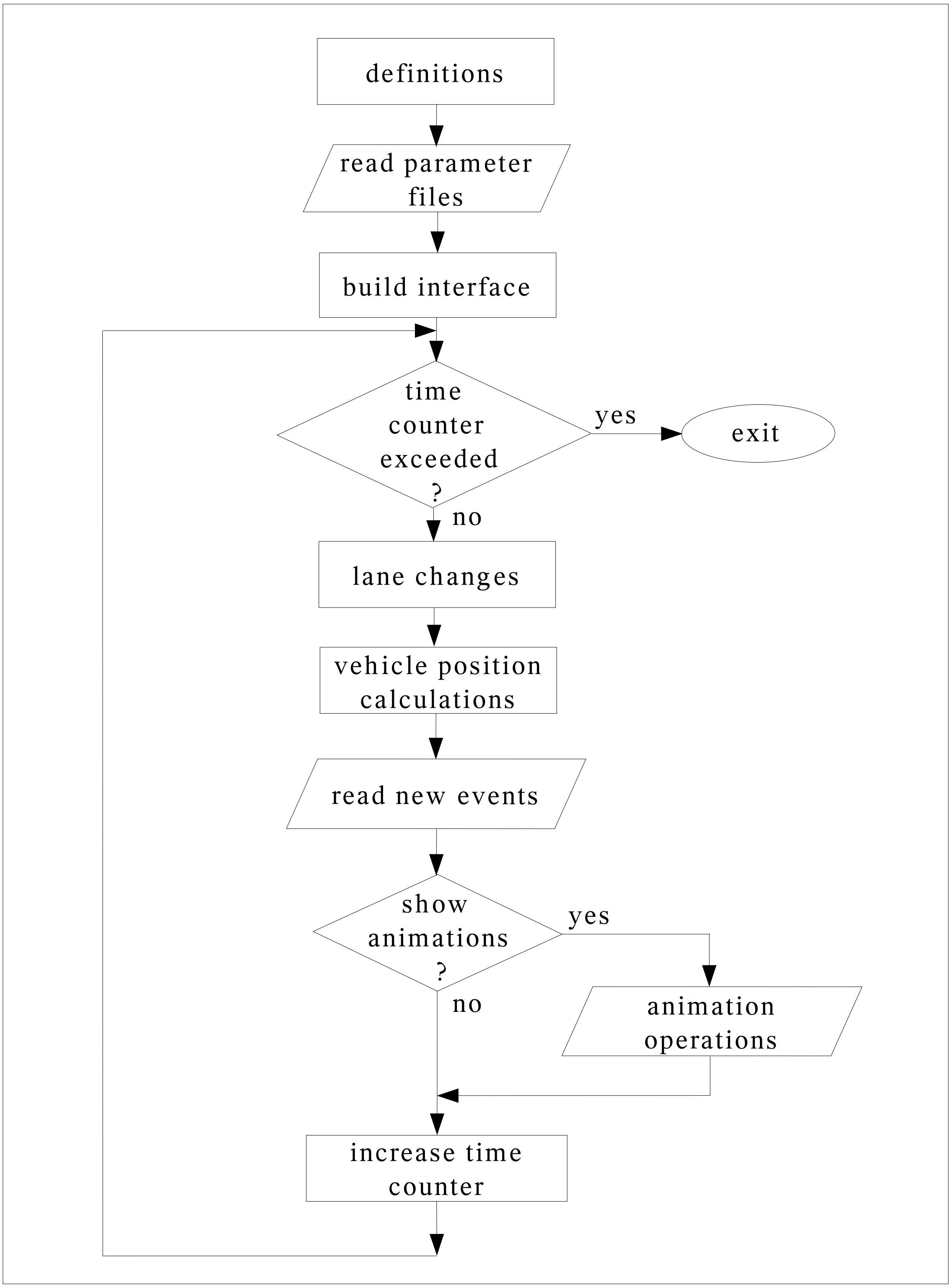


Figure C.1: Rough flow chart of the CA simulator.

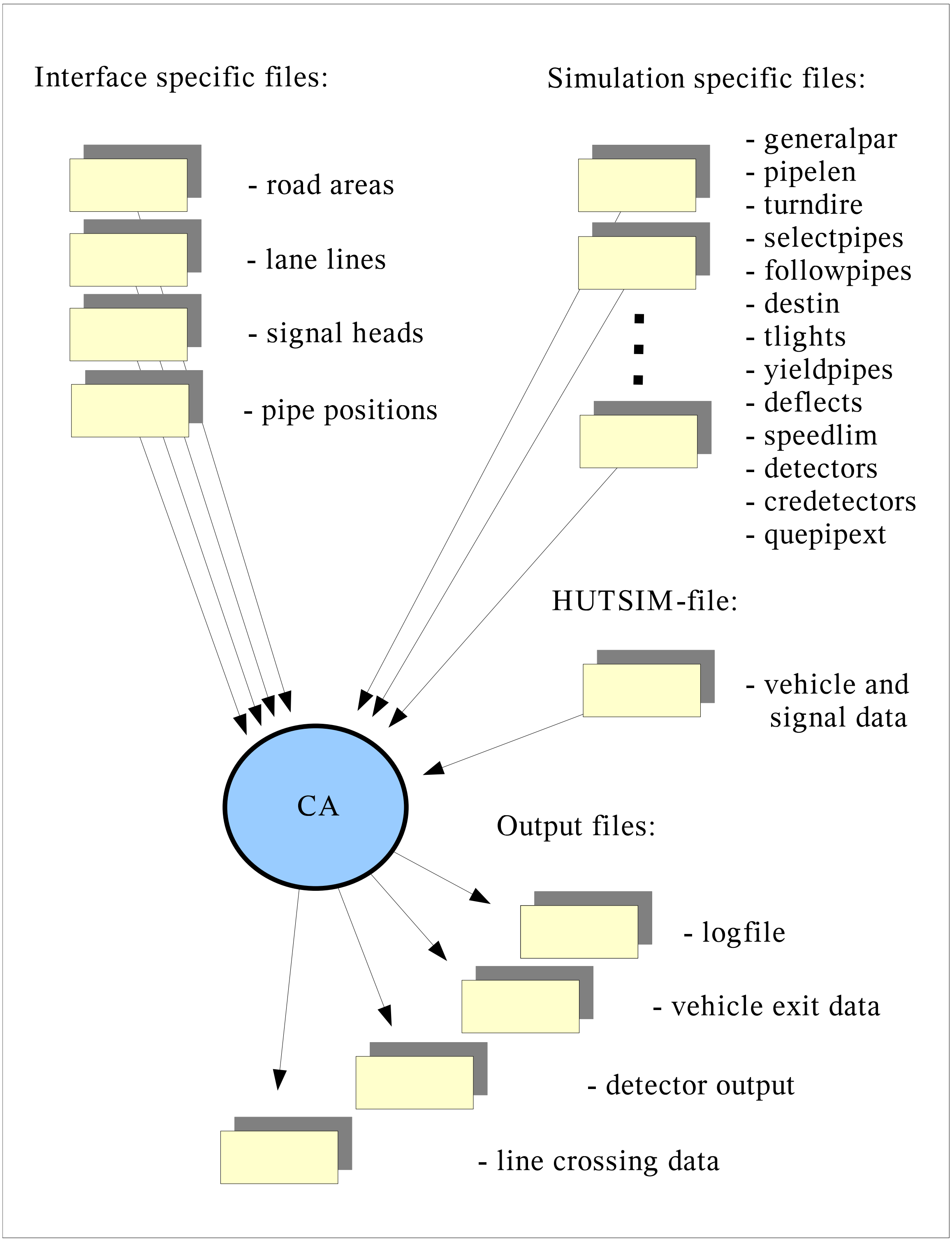


Figure C.2: Files used with the CA simulator.

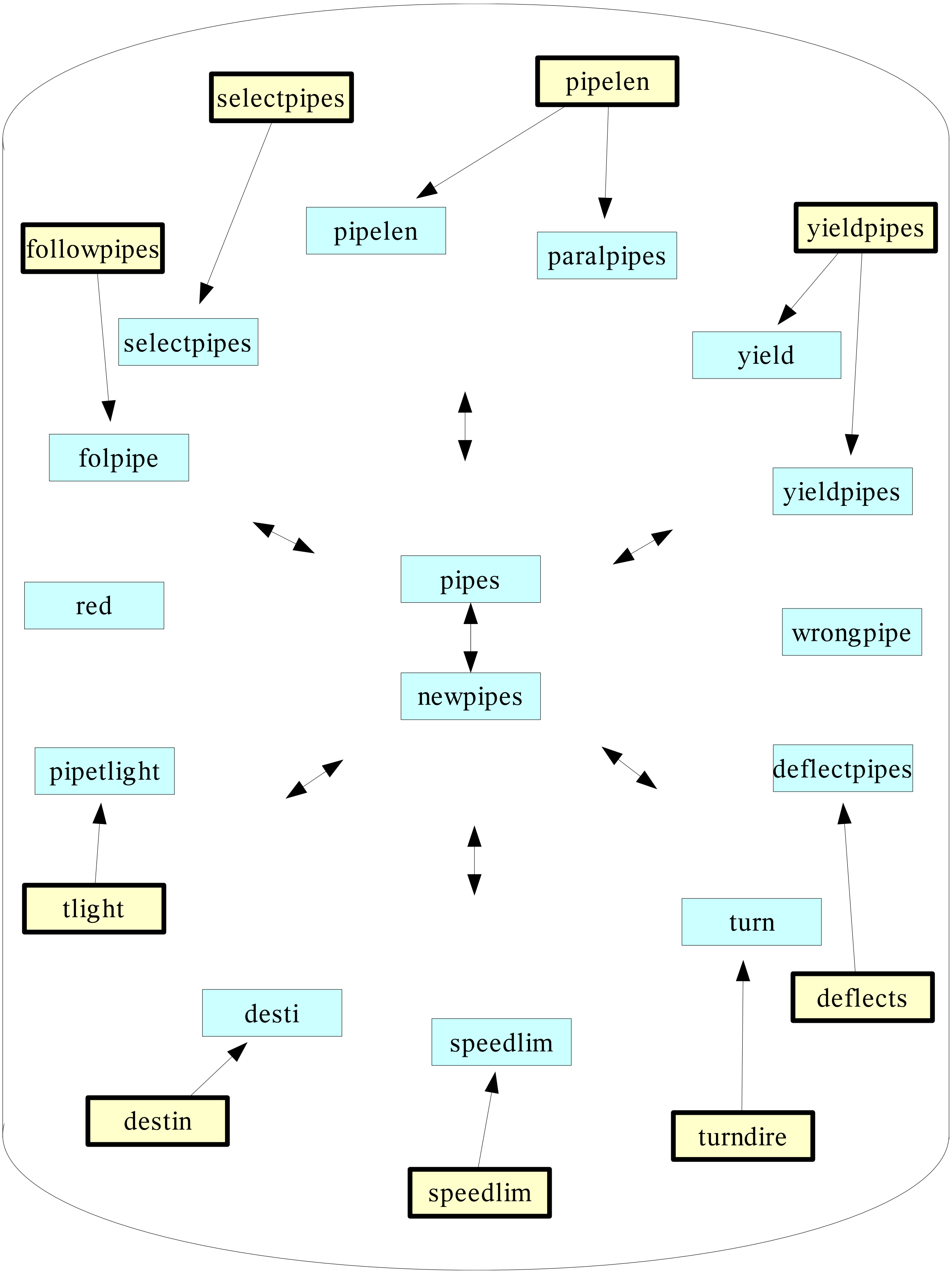


Figure C.3: Internal arrays in the CA simulator program and files feeding data into them at start-up.

function	description
changevehicles	change the places of vehicles in a deadlock situation
closeprog	close open files, write logfile and possible outputfile
colanechange	try changing lane for convenience to parallel pipe
createvehicle	create a vehicle at pipe's first cell
credetectorread	read *credetectors* file
deflectpiperead	read *deflects* file
destinationread	read *destin* file
detectorread	read *detectors* file
detfronts	check the distance of first vehicle in front of each detector
finddestination	determine a destination for a vehicle entering the pipe
findfirstcar	search for the first vehicle in pipe
findminusgap	find minus gap (empty cells) from given position
findnextinpipe	find next vehicle in this pipe
findnextpipe	choose a follower pipe for the present pipe
findplusgap	find plus gap (empty cells) from given position
findqueuemax	calculate queue length from start side of pipe in 'quemaxpipe'
folpiperead	read *followpipes* file
genparread	read general parameters from *generalpar*
getroadCmd	create new vehicles, send vehicle positions to Tcl and call the pipe-iterations function iterpipes
handlevents	handle events from *detsig* file (vehicles, signals)
handlexcesv	try to create vehicles at detectors written to [excesvehs]
initpipes	initialise pipes with zeros and [quels] with −1's
inwrongp	find whether vehicle is in [wrongpipe]
iscredetp	find whether pipe is a credet (generation) pipe
isdeflectpair	check if the pipe belongs to a deflect pair
ispeedlpipe	check if the pipe has speed limit; return the limit
ispipeselect	check if the pipe is one with selecting new turn rate %
ispipeyield	check if the pipe one that must yield other pipes
iterpipes	iterate the network for one time step
lanechange	try changing lane to parallel pipe; for vehicles in [wrongpipe]
leavepipeok	check whether leaving the pipe is possible — if it is, handle the leave situation in this function
main	main program
mustgoparal	check whether a vehicle must take the parallel pipe, because it cannot reach destination from this or some follower pipe
nextpipeok	check if next pipe is OK for entering
npfindfirstcar	search for the first vehicle in pipe for [newpipes]
occucalc	calculate occupation in pipe (vehicles/cells)
oneperoo	calculate 1/density of vehicles ahead of a specified vehicle
openrunfiles	open *detsig* and *cadetsig*/*vehleave* files
paralpmake	build up all necessary parallel pipe pairs from [lenpipes]
parread	read run-specific parameters from input stream
pipelengthsCmd	convey pipe number and pipe lengths to Tcl-part
pipesread	read *pipelen* file
pipetlightread	read *tlights* file

Table C.1: Functions in the CA simulator C-program, part I.

function	description
quelengths	calculate vehicle queue lengths at signal heads
queleninpipe	calculate queue length in one pipe
quepipextread	read *queuepipext* file
removewrong	remove a needless element from [wrongpipe]
removexcesv	remove a needless element from [excesvehs]
searchfile	search for the index of filename by internal name
selectpiperead	read *selectpipes* file
signalupdate	update the signal array [red], change signal value
sihe	calculate occupation parameter of vehicles ahead
simfiles	read internal and external names from *cafiles.dat*
speedlpiperead	read *speedlim* file
stoptoyield	check whether in this pipe at this position it is necessary to stay put for yielding
sumgap	calculate reference gap for speed reduction
Tcl_AppInit	Tcl/Tk-application initialisation
towrong	check if vehicle in wrong pipe and if so, write to [wrongpipe]
turndireread	read *turndire* file
updatedets	update vehicle counts in detector counters
updatewrongp	if vehicle in [wrongpipe], update its position information
yieldread	read *yieldpipes* file and make array [yieldpipes]

Table C.2: Functions in the CA simulator C-program, part II.

procedure	description
ChangeLight	change proper signal colour to signal head
CrossPoly	draw intersection polygon
CrossRect	draw intersection rectangle and accompanying signal stop line markings
CrossSquare	draw intersection square and accompanying signal stop line markings
DelVehs	delete vehicles from screen
DetRect	draw a detector rectangle
FloatBrp	change %-type ibrprob to floating brprob
FloatSpp	change %-type ispprob to floating spprob
LaneLine	draw a lane line
Main	main program
PipeTable	create the arrays for vehicle positions in pipes
RoadPoly	draw a road polygon
RoadRect	draw a road rectangle
TrafficLight	draw signal head according to north-west corner coordinate
WritePipeL	write pipe number (npipes) and pipe lengths to an array
WritePos	draw vehicles according to their pipe positions

Table C.3: Procedures in the CA simulator Tcl/Tk-program.