UNIVERSITY OF TWENTE

Development of a new pro-active vehicle actuated signal control strategy - Evaluation on an urban intersection in Athens

Master Thesis

Antonios Chaziris
s1000667
11 April 2011

Table of Contents

List of Tables

List of Figures

List of Pictures

1. Introduction

1.1 Background

Traffic congestion is a problem that has appeared during the second half of the 20th century, as car ownership has risen dramatically. Along with it, the need for efficient traffic signal control became apparent. Efficient control of traffic with the use of traffic signals, especially in large cities, results in the reduction of lost time, increased safety and potential environmental benefits. Although the use of signal control alone, is not able to entirely solve the problem of traffic congestion, it can significantly improve the movement of vehicles and goods through road networks yielding significant benefits in economic and social terms.

The use of traditional fixed-time signal control is efficient in cases where demand is relatively constant and results in an straightforward operation which is relatively easy to maintain. Several methods exist for optimizing the operation of fixed-time traffic signals, like the well known delay-minimization method proposed by Webster (1958). Moreover, commercial systems like TRANSYT-7F (Wallace et. al., 1998) offer a well established solution for assisting in the optimization and management of fixed-time traffic control.

However, in real-life traffic operations and especially within large cities, demand is rarely constant throughout the day and significant fluctuations can occur even within short time due to incidents, even within periods when demand is usually constant. In such cases, the operation of fixed-time traffic control can be sub-optimal and the need for **vehicle actuated (VA)** (elsewhere referred to as *traffic responsive* or *adaptive*) control arises.

VA control exploits traffic measurements in real time in order to adjust and optimize signal timings according to the prevailing (or future) traffic conditions. The advantage of vehicle actuated control is that it quickly responds to the prevailing traffic conditions therefore it is able to minimise several performance measures over the controlled network. The disadvantage is that usually, strategies are more complicated and include sophisticated logics which might be difficult to implement in practice. Moreover, this type of control is more prone to hardware failures.

A large number of different VA strategies exist which make use of several different methodologies ranging from simple modifications to fixed-time control to very sophisticated optimising strategies which make use of complex traffic prediction models. A detailed review of such strategies is provided in the literature review chapter. The benefits yielded by those strategies also vary significantly, therefore research and discussion around VA signal control approaches is vivid.

1.2 Scope

The motivation for this project was the trend towards vehicle actuated control in many parts of the world and the development of a large number of different approaches towards it. These trends and developments prove the high interest in this research area as well as the potential for improvements arising from this type of control. Athens is a large city with severe congestion (and subsequently environmental) problems where vehicle actuated control has not been widely implemented and tested, apart from few small scale heuristic applications. Moreover, the typical traffic characteristics of a southern European country makes the development and testing of vehicle actuated control more interesting and challenging.

The scope of this Master Thesis is to develop a new vehicle actuated signal control strategy that can operate efficiently in an urban network with the particularities of a city like Athens, with the resources available in the Athens Traffic Management Centre (ATMC); and to contribute to the current state-of-the-art by identifying, studying and enhancing promising methods for VA control, found in the literature.

To implement a vehicle actuated signal control strategy (either for an isolated intersection of for a network) there is a need to tailor it to the needs and particularities of the area of interest. Even a very efficient signal control strategy will be successful only if it can be fully applicable to the selected field. Restrictions imposed by the available hardware (i.e. type of controllers and detectors), particularities in the traffic profile of the area of interest, capability to 'translate' the strategy into the language of the controller etc, could pose significant problems on the implementation of a VA signal control strategy. For this project, these concerns apply only to a certain extent as the target is to implement the signal control strategy at a micro-simulation environment. However, the strategy that will be developed aims at being potentially applicable to a real environment therefore the above considerations are taken into account.

Finally, this thesis, besides implementing a new vehicle actuated signal control strategy, aims to develop a methodology that will also be applicable to other *similar* networks. The development of the strategy does not aim to be an area specific study but a work with wider applicability.

1.3 Research questions - approach

In this section, the research questions of this thesis as well as the approach that was adopted to address each question are discussed. The research questions are divided into three parts: the first part concerns questions related to the definition of the strategy, the second part concerns questions about building the strategy and the third part is concerned with questions related to implementing and testing the strategy.

1.3.1 Strategy definition

The first part of the research questions aimed at outlining the possible objectives and methods that could be adopted for the development of the strategy. The first research question was:

1. *What are the candidate objectives for the new vehicle actuated signal control strategy?*

The selection of the candidate objectives was obviously affected by the particularities of the study area. Therefore before selecting the candidate objectives, a number of considerations related to the operation of the network should be taken into account. A sub question of the above research question was:

2. *How will the particularities of the study area guide us in the definition of the strategy objectives?*

An extensive literature review (chapter 2) was also important to identify different objectives used in existing VA approaches, which would assist in the definition of the most appropriate ones for this case. After having defined a number of candidate objectives (chapters 2 and 3), the methods that could be exploited to achieve these, were explored. A set of research questions related to the exploration of these methods followed:

3. *What can be measured from the available detectors? What are the advantages and disadvantages of using each type of measurement for the signal timing optimization?*
4. *Should the strategy aim at pro-active or reactive signal control? (from another point of view: the optimization procedure should be on a rolling horizon or once in each cycle?)*
5. *What models can be employed for traffic predictions? (in case of pro-active control)*

Question 4 was a main research question which took into account the advantages and disadvantages of each approach and defined many subsequent parts of the strategy. Apart from the nature of the signal control method, research questions related to the optimisation process were defined:

6. *Which measures of performance can be used for the signal optimization process? Why these?*
7. *What convergence algorithms can be exploited for the signal optimization process?*

Concerning question 6 the selection of performance measures needed to be made by considering several factors, instead of simply selecting the minimization of delays as the most well-known and widely-used performance measure. The answer to questions 3 - 7 was provided through the literature review (chapter 2).

1.3.2 Strategy building

The definition of the objectives and of an appropriate toolbox of methods allowed to start developing the strategy. The set of research questions concerning the building of the strategy were:

8. *How can the selected methods be developed with the existing software?*
9. *What are the limitations that restrict us from using some tools? What compromises should be made?*
10. *How can the selected methods be enhanced and be tailored to the needs of this project?*
11. *What is the added value of the developed strategy compared to previous VA approaches?*

Research question 8 is related to the actual development of the VA strategy, in other words, the programming part. Providing answers to all the above research questions actually included the main workload of this master thesis (chapter 4).

1.3.3 Implementation/testing

After the development of the strategy and the set up of the simulation environment, traffic operations under the new VA control were tested. On that purpose, firstly, a set of appropriate efficiency measures had to be selected:

12. *Which set of efficiency measures will be employed to test the strategy?*

Moreover, the performance of the strategy was compared to the performance under other operational situations and under different demands, so as to quantify the reliability of the strategy:

13. *How does the strategy perform in comparison to the existing fixed time signal plans?*

Finally, it was important to investigate the performance of the strategy under extreme scenarios concerning incidents or hardware failures. In other words:

14. *What is the robustness of the developed strategy?*

A final research question related to testing the strategy concerned its appropriateness for road networks with different characteristics. The strategy objectives as well as the measures of performance were affected by the study area particularities, therefore, its effectiveness on a different network would be under discussion. The above can be summarized to the following research question:

15. *How might the applicability of the developed strategy be hindered on networks with significantly different characteristics than the one studied?*

Research questions 12-14 were eventually answered through the evaluation study presented in chapter 5 and research question 15 through the discussion presented in chapter 6.

1.3.4 Further questions

A potential limitation arising from the adoption of vehicle actuated signal control strategies (as well as any other traffic management scheme aiming to improve traffic flow) concerns the rise of latent demand. In this respect, the following research question could have been addressed:

- *What will be the impact in traffic demand from the implementation of the strategy in the selected arterial?*

This question was not addressed because to answer this question adequately, fluctuations in demand at a macroscopic level, including the network outside the arterial (impact on OD matrices on a larger area, after the application of the new strategy) should be studied. However, this requires an extensive work which is not possible to be carried out with the time available for this study. However the potential of the rise of latent demand, might be considered as an interesting extension of this research in the future.

1.4 Available resources

In this section, the type of controllers for which the strategy is operational are described and the software that was exploited for the development of the strategy is presented.

1.4.1 Hardware

The most common type of controller used in the city of Athens is the SIEMENS C800. The controller's microcomputer is able to control up to 48 signal groups. The maximum allowable signal cycle is 300 seconds. The standard functions of signal control are realized independently in the C800 controllers with the aid of the data and parameters entered by the user (signal plans, intergreen times etc.). Furthermore, the C800 controllers are capable of integrating vehicle actuated signal control strategies. For that purpose, various control processes are available for use from the controller. The most suitable control process for the needs of the current project (full traffic actuation, several intersections, control flexibility) is the PDM control process. In this process, signal groups are controlled at the stage transitions. Within one stage, the status of the signal groups remain the same.

The Athens Traffic Management Center uses two types of detectors for traffic measurements, namely inductive loop detectors and video detectors. Single inductive loop detectors will be exploited in this project as they are available in the selected study area (see following chapter). Each of the single loop detectors is connected to an LD4 module which has high sensitivity and quick response time. LD4 switches on the detector when a vehicle is present and switches it off when there is no vehicle. It has several sensitivity levels to account for different vehicle categories and hence it needs appropriate calibration. Furthermore, the status of LD4 is checked every 10 ms and the 10ms intervals are added for the time the detector is occupied (to calculate occupancy). These recordings also lead to the estimation of several other traffic data such as traffic flows and gaps.

1.4.2 Software

To develop a vehicle actuated signal control strategy and to implement it to a simulation environment three packages need to be used, namely: P2, Control and Traffic Language. Apart from the software mentioned before, the VISSIM micro simulation environment developed by PTV will be used. In this section, these packages are briefly presented.

P2

SITRAFFIC P2 is an object-oriented tool for planning traffic signal controlled networks and intersections in fixed-time and traffic-actuated control systems (SIEMENS, 2005). It covers the development of signal programs (taking any evaluation of signal plans into account), time-distance diagrams and basic data for vehicle actuated control.

The planning of intersections in P2 takes place in a number of consecutive steps. First, the general intersection data are defined (concerning regulations, types of controllers, control methods etc). P2 offers a number of default files concerning the type of standards that will be used for each project. At the next step, the topography of the intersection is defined. On this step, a layout plan can be used as a background to help in the design. Then, the appropriate signal groups, lanes and crossings should be created on the basis of the designed topography. After this step, the conflict matrices, the respective intergreen times and the offsets can be calculated. The basic concept concerning the estimation of the intergreen times in Athens is similar to the one described in the German RiLSA 1992 standards.

After the previous steps, the traffic data can be inserted to serve as a basis for the calculation of the appropriate signal plans. P2 is capable of estimating green times for fixed-time signal plans, based on a simple method which aims at balancing the degree of saturation in each intersection approach. In order to develop signal plans, the appropriate control method should be selected (PDM in our case). The stages, the possible stage sequences, details concerning the stage transitions and the possible green extensions can then be defined. Finally, after the signal programs are developed, the appropriate switch-on and switch-off points are defined.

The data created in P2 can be exported for use to other programs. In case of vehicle actuated control, the output data from P2 are used by other packages (Control) and are used as a basis for the data that will be used as input in the C800 controllers.

Control

The SITRAFFIC Control data input program offers a convenient interface to communicate all required data of the signal programs to the C800 controllers (SIEMENS, 2003). The program allows the input data of the C800 to be created/imported, modified and loaded to the controller. All the data of an intersection is combined in one project. As early as the project configuration stage, the project administration function checks every component (basic data inputs, user program, simulation, etc.) for dependence relationships and compatibility.

The heart of Control is the Project Management component. The objects for all the sub-areas can be accessed from this component. When a new project is created, the type of controller should first be defined. A number of basic data should also be inserted (i.e. data transmission methods, activation/deactivation processes, signal groups, cycle times, etc). Color combinations and signal group descriptions are defined in the area concerning signal definitions. Minimum greens/reds, intergreen times, offsets, and switch-on/off patterns are defined in the area concerning signalization. Most of these data are automatically retrieved when an existing project from P2 is imported. Some additional data that need to be defined by the traffic engineer concern information that was not defined in P2 and concerning traffic actuation, i.e. the assignment of specific functions to detectors (type of traffic measurements, critical values etc.), the assignment of detectors to signal groups, the state of the various signal groups in the various stages, additional information on stage transitions etc.

Traffic Language

To integrate complicated processes and algorithms into the signal control strategy, the use of Traffic Language (TL) is required (SIEMENS, 2005). This tool serves for solving both simple and more demanding signal control problems. TL includes a flow-chart editor which assists the user in the formulation of logical conditions in a graphical manner. Within the program module, traffic function libraries offer solutions for a number of traffic sub-problems.

A debugger facilitates troubleshooting by evaluating a program and reporting inconsistencies and problems in the code. With the help of a compiler, the vehicle actuated control logic can be exported for use in VISSIM.

VISSIM

The micro simulation software VISSIM (PTV, 2008) will be used in this study as a platform for testing the applicability, efficiency and robustness of the developed signal control strategy.

The traffic flow model in VISSIM is based on a psycho-physical car following model for longitudinal vehicle movement and a rule-based algorithm for lateral movements. The model is based on an extension of Wiedemann's 1974 car following model (Wiedemann, 1974), which is referred in VISSIM as "Wiedemann 1999". VISSIM is a microscopic, time-step and behaviour based simulation model developed to model urban traffic and public transit operations. The program can analyze traffic and transit operations under constraints such as lane configuration, traffic composition, traffic signals, transit stops, etc., thus making it a useful tool for the evaluation of various alternatives based on transportation engineering and planning measures of effectiveness. VISSIM can record a large number of parameters for each vehicle moving in the network and provide disaggregate results per vehicle and time-step. Moreover, it can provide aggregate measures for the network performance and also evaluate specific parts of the network (i.e. intersections). Moreover, in order to obtain results which are not dependent on the arrival process (seed) of one specific demand, several simulations for each demand profile can be made using different seeds by exploiting the VISSIM feature called "Multirun simulation". In this way, all data needed for signal control strategy evaluation can be obtained.

To design a realistic simulation, a lot of data have to be set manually in the software. Some of these are readily available (i.e. traffic demand for different days and hours), some can be obtained by drawings (i.e. intersection layouts) and some need to be estimated by the traffic engineer according to the selected study area (i.e. driving behavior, lane closures etc.).

The data and the traffic control strategy created and processed with P2, Control and Traffic Language can be transferred to VISSIM for implementation. The equivalent to the C800 controller can be selected in VISSIM (called SIEMENS VA). From P2 an export of a file is possible in order to transfer the intersection layout and fixed time signal plans directly to the VISSIM environment. However, as the design environment of the two programs differs, some elements (such as connectors, routes, conflict areas etc) have to be redefined or corrected in VISSIM. After the correct topography is created, the signal control strategy should be imported by selecting several program and configuration files and libraries exported from P2, Control and TL. Once the files are imported, the user can select the appropriate output files and windows in order to evaluate the operation of the signal control strategy.

1.5 Report Structure

The present report comprises of seven chapters in total. The four chapters that follow the introduction (2-5) are similarly structured to the four basic deliverables of this project (Literature review, Study area design, Strategy development, Implementation and testing) while the final two chapters (Discussion, Further research) are conclusive for this report.

More specifically, the second chapter concerns the literature review. The purpose of this chapter is to provide a good understanding of the existing methods and techniques used in traditional and contemporary VA control strategies in order for the selection of the methods for this Thesis to be properly supported.

The third chapter concerns the design of the study area using the available software. A robust picture of the study area and its particularities is formulated, the traffic data are collected, the demand scenarios to test the effectiveness of the strategy are developed and the design of the area in SITRAFFIC P2 (SIEMENS, 2005) and VISSIM (PTV, 2008) is presented. The simulated network resembles traffic operations in the area in a realistic way and operation under the existing fixed time plans and the new VA control strategy can be simulated.

The fourth chapter presents the development of the VA strategy. The chapter describes the concept in detail, describes the rationale of the new idea, analyses all technical issues concerning its development and discusses the programming effort. The programming code as well as an extensive manual that helps reading it and understanding it are provided in Annexes 3 and 4.

The fifth chapter includes the results from the testing of the VA strategy in VISSIM. This chapter includes comparisons of the results yielded by the VA strategy with the fixed time signal plans, performance under different demands and the robustness assessment of the road network under the new strategy.

The sixth chapter presents discussion around the new VA strategy, its usefulness, its applicability and its added value. This chapter draws conclusions from the whole development process as well as from the evaluation.

Finally, the seventh chapter presents the proposals for further research, which occurred during the development of this strategy. Several issues are identified that could be researched further, either to make the strategy more efficient or to further assess its success.

2. Literature review

This chapter presents the literature review concerning existing methods and techniques used in vehicle actuated signal control, as well as other aspects (such as hardware capabilities) related to the development of the strategy of this Thesis. Moreover, the aim is to discuss the applicability of the existing methods on the strategy to be developed and support the decisions taken for the design of the strategy.

More specifically, the chapter provides answers to the research questions 1-7:

1. *What are the candidate objectives for this vehicle actuated signal control strategy?*
2. *How will the particularities of the study area guide us in the definition of the strategy objectives? (answered also through chapters 3 and 4)*
3. *What can be measured from the available detectors? What are the advantages and disadvantages of using each type of measurement for the signal timing optimisation?*
4. *Should the strategy aim at pro-active or reactive signal control? (from another point of view: the optimisation procedure should be on a rolling horizon or once in each cycle?)*
5. *What methods can be employed for traffic predictions?*
6. *Which measures of performance can be used for the signal optimisation process? Why these?*
7. *What convergence algorithms can be exploited for the signal optimisation process?*

The structure of this chapter is as follows:

The following section provides a background concerning the particularities of the strategy developed in this Thesis. The next section presents a review of signal control objectives and illustrates the relation of the objectives with wider transportation policies. Moreover, it presents the measures of effectiveness used for the evaluation of the strategy. The fourth section presents the hardware available for traffic detection and discusses its capabilities as well as the limitations arising from its use. The fifth section discusses methods for the estimation of the traffic state, as this is the most essential input required for the signal optimisation. The sixth section presents a wide range of signal optimisation techniques, used in the most well-known systems and discusses their applicability. Finally, the seventh section is concerned with other aspects related to vehicle actuated control that are not discussed in the previous sections.

2.1 Background

This section provides information on the particularities of this study that played an important role in the decisions taken for the formulation of the strategy. These particularities should be kept in mind when reading the following chapters.

The signal control strategy formulated in this Thesis is intended to be applicable not only on isolated intersections but also to arterial networks. A very wide variety of vehicle actuated strategies developed during the last years, although very efficient, are difficult to be applied to more than one intersection either due to their complexity (in terms of required computational power) or due to their critical assumptions concerning the topology of the network, the detectors etc. Strategies like RHODES (Mirchandani and Head, 2001), OPAC (Gartner, 1983) and PRODYN (Farges et. al., 1983) aim to provide a solution to the optimal signal control problem by estimating signal timings that minimise a given performance measure, for traffic estimates referring to a future time horizon. In those cases, the formulation of the problem requires knowledge of vehicle arrivals on the downstream intersections, based on upstream detector data, for the near future. However, to obtain a reliable traffic forecast, the signal decisions of the upstream intersections are also required; while these signal decisions are based on the traffic forecasts. This results in a loop which requires employment of recursive algorithmic techniques such as dynamic programming, to be solved. Furthermore, the problem cannot be solved locally (for each intersection) but should be solved globally (for all intersections) and the complexity of the algorithm rises exponentially with the number of intersections. For that reason, such strategies have either remained at a theoretical level or have only been evaluated only in a simulation environment, mostly at a single intersection.

Other strategies, like GASCAP (Owen and Stallard, 1999) although conceptually applicable to more than one intersection, employ traffic estimation methods (in the specific case queue estimation) that requires upstream detectors at a large distance from the intersection, without significant changes in the amount of traffic between the detectors and the intersection. Such requirements make those strategies difficult to be applied in a real world arterial as the necessary assumptions are very difficult to be satisfied. Other strategies, like the one proposed by Liu et. al., (2002), present viable and efficient strategies which; however, use very specific applications especially developed for that purpose, such as vehicle re-identification from the detectors using waveforms (which serve as a unique id of each vehicle) to estimate individual vehicle delay, therefore are severely restricted by the capabilities of the available hardware.

One of the main purposes of this strategy is to be efficient but also applicable in an urban arterial with complicated traffic patterns (like in the Athens road environment), relatively closely spaced intersections and limited hardware availability (only inductive loop detectors). This challenge restricts the use of some methods as will be discussed in the following chapters.

A second aspect that had to be considered for this strategy was whether the strategy would be centralised or not. Centralised strategies like SCOOT (Hunt et. al., 1981), SCATS (Lowrie,

1982) and UTOPIA (Mauro and Di Taranto, 1989), compute the whole control logic at a central computer and then distribute it to local traffic controllers. On the other hand, in a decentralised system, each intersection controller contains the whole control logic and decides what its signal state should be. There are, of course, hybrids of the decentralised/centralized architectures because, for any practical system, some of the intelligence for the signal system must reside at a central computer. However, decentralised vehicle actuated control potentially can serve an unlimited number of intersections because the control logic is distributed at each intersection. Examples of such systems are DYPIC (Robertson and Bretherton, 1974), OPAC, PRODYN, MOVA (Vincent and Young, 1986) and lately GASCAP (Owen and Stallard, 1999).

In this project, the choice between a centralised and a decentralised system was dictated by the capabilities of the available hardware. In fact, vehicle actuation applications, from the simplest to the most sophisticated, have to be carried out at the local controller as no superior level (central, or strategic computer) exists. Athens is a city which is still mostly controlled by semi-actuated strategies (automatic selection between a number of pre-specified fixed time plans) therefore no architecture that can serve a centralized, fully adaptive strategy has been developed. On that purpose, this strategy will be developed to fully operate locally, in each traffic controller.

2.2 Signal control objectives / measures of effectiveness

This section discusses the results of the literature review on signal control objectives, performance measures and measures of effectiveness. It should be mentioned that the signal control objectives and the performance measures are related but, for the same objective (i.e. minimisation of the environmental impact) different performance measures of can be used (i.e. minimisation of accelerations or number of stops or fuel consumption, etc). The measures of effectiveness though are the criteria by which the success of the strategy implementation will be evaluated. These measures can be different from the performance measures used for the optimisation. For example, the performance measure can be the delay over the intersection but the strategy can be evaluated also by the number of stops and the queues it results to.

The objectives as well as the strategies used to pursue them, presented in this chapter, include traditional as well as more modern approaches towards signal control. The possibility to select each objective for the strategy of this Thesis is discussed and leads to the selection of objectives presented in the final section. Finally, the last section, apart from the selection of objectives aims to provide an answer to the question: *when is the strategy successful*, or in other words: *when can we say that the objectives were fulfilled*. This question relates to the evaluation of the strategy presented in chapter 5.

2.2.1 Objectives

The signal control objectives presented in this section are divided into two families. The first family includes *efficiency* objectives such as delays, queues and stops. These objectives are directly related with the operation of the network and aim to optimise a performance measure (which can consist of one or a weighted combination of indicators) related to its effectiveness. The second family of objectives, includes other objectives, not directly related to traffic efficiency which; however, are of high interest for other reasons. Such objectives include the environmental impact of traffic or the accessibility for vulnerable road users. The usefulness and applicability of each objective for the present Thesis is also discussed.

Efficiency related objectives

The most common objective, pursued by the majority of the signal control strategies, is the minimisation of vehicle delays. The minimisation of delay, as an objective, can be interpreted in many ways, depending on the examined area (total network delay, total intersection delay, link delay), on the type of delay (due to the presence of other vehicles, due to traffic signals, due to existing queues, etc) and on the measure of delay (per vehicle or rate of delay).

Most well-known vehicle actuated signal control strategies (SCOOT, SCATS, OPAC, PRODYN, RHODES, CRONOS), use delay as a performance measure to minimise within their optimisation procedure. However, delay can be quantified by using several methodologies. Traditional delay formulas such as the one proposed by Webster (1958) or by Kimber and Hollis (1979) estimate the delay due to the presence of traffic signals and not due to other reasons such as the presence of other moving vehicles. However, a vehicle actuated strategy might use these formulations to estimate the optimal traffic signals based on "delay minimisation". Therefore, it should be kept in mind that the term *delay minimisation* is used by several strategies; however, not always in exactly the same way.

An example of a very simple delay minimisation method is provided by the CRONOS system (Boillot et. al., 2006). The system aims to minimise the total delay over the control zone (single or multiple intersections), for a given time horizon. The performance measure is a simple aggregation of the stopped vehicles (vehicles in queue, plus vehicles stopped within intersections) in each time step, multiplied by the duration of the time step, and summed up for the time horizon.

The delay minimisation objective can also include several weighted terms of delay. The weights can be assigned to distinguish between different links (i.e. the delay in the main direction can be considered as more important than the delay in a secondary road), different vehicle types (i.e. delay of transit vehicles more important than delay of cars) and so on. The delay is a performance measure considered closely related to other performance measures like vehicle emissions (Bretherton et. al., 2002, SCOOT advice leaflet). In that sense, delay is a universal performance measure that can be used as a proxy measure to satisfy various objectives.

Another widely used objective, which most likely will lead to different splits compared to the delay minimisation, is capacity maximisation. The adoption of this objective is suitable when the intersection is operating under oversaturated conditions. The objective was adopted in the SIGCAP stage-based strategy (Allsop, 1971, 1976), as an alteration of the SIGSET, which was minimising delays. The objective of capacity maximisation leads to an effort to utilize all the reserve capacity of each intersection approach within the respective stage and therefore will most likely lead to the extension of the green time until reaching its maximum value. Moreover, for the same reasons, this approach will always lead to the maximum allowable cycle time. Due to that fact, the objective of capacity maximisation will lead to significantly different signal timings than delay minimisation, these two objectives are contradicting.

An objective related to the capacity maximisation, from a more general point of view, aims at minimising the risk of oversaturation or the spillback of link queues at upstream intersections. As an example, the TUC strategy (Diakaki et. al., 2002), attempts to fulfill the above objective by balancing the link's relative occupancies.

Signal control strategies are not always designed to pursue a single objective but usually use a combination of objectives. In that sense they attempt to minimise performance indices consisting of a weighted combination of different performance measures. As an example, the system proposed by Vasudevan and Chang (2006), uses a weighted combination of vehicle queues, intersection control delays, and stop times. Another characteristic example is SCATS, which adopts the multiple objective of minimising stops when the traffic is low, delay when the traffic is heavy and travel time. SCATS attempts to fulfill these objectives by balancing the degrees of saturation of the intersection approaches.

Efficiency, as expressed from all previous objectives, can be regarded as the main target set by the traffic engineer; however, other objectives not directly related to efficiency can be of equal or greater importance. These are discussed in the following section.

Other objectives

A limitation for many of the existing signal control strategies is that there is no special provision for transit vehicles, which are effectively treated as normal passenger cars. In networks where transit traffic shares the physical space with passenger cars this limitation is even more pronounced. However, in mixed traffic operation, transit vehicles may hold up other traffic while loading passengers even if the signal is green. When this happens, especially near a signalized intersection, this traffic-transit interaction should be described properly as an input into the signal plan generation procedure. One should also give appropriate priority to transit vehicles, so that the total cost and delay to cars and transit passengers can be minimised (Yagar and Han, 1994).

The approach of Yagar and Han, as well as more recent and sophisticated approaches, like SPPORT (Dion and Hellinga, 2002) makes signal-switching decisions following heuristic, rule-based signal optimisation procedures, as the consideration of transit vehicles imposed an additional barrier to the adoption of more analytical models. Moreover, there were concerns that exhaustive optimisation procedures such as dynamic or linear programming may be too computationally demanding for real-time signal control applications including transit vehicles in networks with highly variable demands. However, even with a rule based approach, the special consideration of transit vehicles in a vehicle actuated signal control strategy became possible.

In the latest version of SCOOT, a differential priority objective is included where the degree of priority is defined according to how late a transit vehicle is according to its schedule as well as on the link that the transit vehicle is moving (main or secondary road). Of course, to include such an objective to the signal control strategy, an automatic vehicle location on every transit vehicle is required. Moreover, this objective is meaningful only in networks with significant passenger traffic using transit vehicles and where transit vehicles are more likely to experience significant delays due to congestion.

The environmental impact is increasingly taken into account in the objectives of vehicle actuated signal control strategies. However, the variety of approaches in the subject results in different objectives.

Bretherton et. al., (2002), referring to the latest version of SCOOT, claim that the reduction of delay, which is the primary objective of SCOOT, leads to a reduction in vehicle emissions. However, from other studies it has been suggested that pursuing efficiency in signal control will not always lead to a respective decrease in traffic emissions. Signal control strategies with an objective to mitigate traffic congestion and with an objective to reduce traffic emissions are usually different, and sometimes conflictive (Chen and Yu, 2007). On that purpose, signal control strategies including clearly environmental objectives have been developed.

To unambiguously include the emission reduction in its objectives, SCOOT uses an emission estimation model. The emissions model works by first calculating an emissions rate per vehicle and unit distance from the average speed on the link. An emissions rate is estimated for each vehicle class. The emissions are then multiplied by the length of the link and the flow rate of the relevant vehicle class. Finally, the emissions for each vehicle class are summed to provide a mass per unit time on the link. Apart from the availability of real time traffic flows, speeds and proportions of vehicle classes, this methodology implies the existence of an appropriate model which estimates the emissions per vehicle from the above data. However, such a model should be developed or calibrated to a specific study area otherwise differences in vehicle fleet technologies, vehicle ages or driving styles might result in significantly different emission estimates compared to reality.

Due to the absence of a model that can effectively be used online for the estimation of emissions from traffic, it is difficult to include the minimisation of the environmental impact of the strategy in this Thesis as a primary objective. However, it is possible to use a number of proxy measures directly related to emissions to evaluate the relative performance of different signal control strategies. On that purpose, these proxy measures can be used as measures of effectiveness of the strategy to allow comparisons with the existing fixed time plans. This subject is further discussed in the following sections.

Most vehicle actuated strategies aiming at the minimisation of delay refer to the delay of passenger cars. A number of approaches aiming at the prioritization of public transport may also provide special weights to the delay of transit vehicles. Traditionally, delay to pedestrians has been far less important for traffic signal control optimisation, but there is nowadays a shift of emphasis to vulnerable road users like pedestrians and cyclists (Carsten et al., 1998). The consideration only of vehicular traffic on signal control optimisation might well lead to longer cycle times, especially during congested periods. However, longer cycles mean increased waiting times for pedestrians therefore can increase non-compliance and reduced safety.

Video and microwave technology have recently enabled signal control strategies to be not only "vehicle-actuated" but also "pedestrian-actuated". This was the basis for the study conducted by Carsten et al. (1998) in the vulnerable road user traffic observation and optimisation (VRU-TOO) project. Microwave detectors were employed to register the approach of pedestrians at three different test sites. Results showed that the pedestrian green phase was prolonged if there was a high pedestrian demand or if there were still pedestrians on the road at the end of the green phase. Moreover, pedestrian actuated control lead to improved safety (measured in terms of fewer conflicts between cars and pedestrians, reduced crossing of pedestrians during red and less crossing nearby the junction) and significantly reduced delays for pedestrians. The vehicle delays increased but not significantly compared to the significantly reduced pedestrian delays. In that sense, multi-objective strategies including a combination of passenger car and vehicle delays might result in improved equity between the various road users.

The importance of pedestrian delay strongly depends on the location of the network as well as on the magnitude of pedestrian volumes. At busy central areas, commercial sites etc, pedestrian delays are far more important compared to main arterials outside the city centre with less frequent crossing attempts. Finally, to efficiently include pedestrians in the control loop, appropriate pedestrian identification systems like the one mentioned above should exist.

The range of objectives covered in the above sections reveal the main tendencies towards signal control derived from the most well-known vehicle actuated strategies. The following section summarises the findings and provides some discussion around the information presented.

2.2.2 Discussion

It is believed that the inclusion of an objective, the fulfillment of which is easily perceived by users is important because it can improve the user acceptance for the implemented strategy. Delay is a measure which cannot be easily perceived by users. The minimisation of delay brings significant benefits to the system efficiency but the benefit for the individual road user is difficult to be perceived. However, other objectives related to efficiency, like the minimisation of queues, stops or waiting times are more easily perceived. For that purpose, avoiding queues to spill back to critical locations can be considered as an important objective. Moreover, queues are directly perceived by users and their minimisation is a straight forward objective that can improve user acceptance. For these reasons, the efficient control of queue lengths is considered as an important objective that should be pursued by the strategy.

For the examined network, pedestrian delays are important because commercial land uses are adjacent to the road and relatively frequent crossings are taking place. Pedestrian detection can only be realized by using pedestrian buttons on traffic signals. The number of pedestrians waiting or approaching, the crossing time and other variables required for a more sophisticated pedestrian-actuated strategy do not exist. As a result, pedestrian stages will be included in the strategy but will not be included in the optimisation procedure. Pedestrian signal groups will run concurrently with other non-conflicting signal groups, ensuring that pedestrians receive sufficient green within a cycle, without however further optimisation.

The prioritization of public transport might be of high importance for a local traffic agency and might need to be achieved at expense of the rest of the traffic. As mentioned also in chapter 3, in the arterial studied in this Thesis, traffic consists only of passenger cars, while bus lines are only using a small part of the network. Moreover, traffic disruption by light rail or other public transport means does not exist. Finally, Automatic Vehicle Location for transit vehicles is not available in Athens. This poses a significant obstacle in the realization of a strategy including transit priority, as sometimes the transit vehicle needs to be detected at a distance further upstream from the available detector location. For that reason, the intersection for the strategy evaluation was selected so as to have minimal transit traffic.

2.2.3 Measures of effectiveness

This section discusses the selection of measures for the evaluation of the network operation under the new strategy. These measures should provide a good picture of the network operation and of the strategy efficiency by different viewpoints.

The first and most straight forward criterion, is whether the objectives of the strategy are *sufficiently fulfilled*. It is crucial to determine whether the operation of the intersection under the new strategy is more efficient in terms of the selected performance measures compared to the old fixed-time plans. Therefore, the performance measures related to the strategy objectives will be quantified and compared between the new and the existing strategy.

Apart from the measures related to the objectives, there is a need to further assess the strategy in terms of effectiveness measures related to the specific study area. Although the strategy is developed to be applicable to any network, its effectiveness in each case has to be evaluated independently therefore this set of measures is related to the location of the intersection and the most imminent problems associated with traffic in this area. As discussed in the following chapter, the study area is located on the outskirts of the Athens city centre, in an environment suffering from air-quality degradation which is significantly affected by road traffic, the built environment and the lack of green areas. It has been continuously suggested (Chen and Yu, 2007, Zhang et. al., 2009) that signal control strategies significantly affect traffic emissions and that a strategy aiming at increasing traffic efficiency will not necessarily lead to a respective reduction in emissions. On that purpose, evaluating the new strategy in terms of emissions is of high importance.

A final measure to evaluate the effectiveness of the strategy will be its **robustness**. Network robustness, regarding to Li (2008), is understood “as the analysis of the performance of the road network under the situations with considerable changes in its supply or/and demand compared with its normal or desired performance”. Therefore, robustness is critical for the operation of signal controlled networks under non-recurrent congestion caused i.e. by incidents. On that purpose, it is critical to study the degree of degradation of the network under these circumstances.

VISSIM evaluation files, include several indices which can cover the aforementioned performance measures. Such measures are the average speed in the network, the throughput, the travel times, the overall network delays, queues, stops, emissions and more. The selection of measures, the description of the evaluation testing and the results are presented in chapter 5.

2.3 Detector capabilities

The capabilities of the available vehicle detection hardware can be a critical bottleneck for the real-world implementation a vehicle actuated signal control strategy. A very integrated and sophisticated strategy that requires data that are not possible or very difficult to be collected, might not be able to be implemented. Detector capabilities can differ significantly, depending on the type of the detector (inductive or video) and its technology. For this study, the detectors used in the Athens urban road network are the only source of on-line traffic data. The capabilities as well as the strengths and weaknesses of the specific detectors are discussed in this section, aiming to provide an answer to the research question 3:

3. *What can be measured from the available detectors? What are the advantages and disadvantages of using each type of measurement for the signal timing optimisation?*

In general, two types of detectors exist in Athens. Inductive loop detectors, placed under the street surface and video detectors placed on poles above the street. This study will make use only of loop detectors as these are the only type of detectors available in the selected arterial. Therefore, this chapter focuses on the capabilities of the loop detectors.

The inductive loop detectors can be further categorized into single and double ones depending on the number of successive loops used. Single inductive loop detectors can primarily measure **traffic flow**, **time occupancy** and **gap**. Traffic flow is calculated as the sum of the times that the loop became occupied within a given interval, reduced to hourly rates. Occupancy is defined as the percentage of time for which the detector remained occupied. When using single loop detectors, speed can be estimated by a formula using the detector length, the average vehicle length, and the flow and occupancy for a given time interval. However, due to the usage of an average vehicle length, the speed measurements obtained from this formula are not always accurate. When using double loop detectors the vehicle speed can be estimated more accurately by exploiting the gap between the actuation of the two loops.

The typical loop dimensions used in Athens, are: length equal to 1,5 and width equal to 2m. The same dimensions were used for the detectors in the VISSIM simulation. The loop dimensions can affect the detection accuracy as a very wide loop might interfere with vehicles passing on adjacent lanes. For that reason, the loop width should be maintained substantially lower than the lane width.

An observation relevant to the formulation of the vehicle actuated strategy concerns the utilization of traffic detectors which affects the quality of measurements. At an urban arterial, where vehicles follow several different routes and change lanes frequently, detectors are not always capable to count all vehicles, as some vehicles might be passing between two detectors. Moreover, large vehicles can be counted by two detectors simultaneously, when changing lanes. Experience in the Athens TMC, has proven that the measurement of traffic flows can be problematic for the above reasons. A possible solution could be to use occupancy instead of traffic flow to determine the state of traffic; however this might be an obstacle for using specific formulas (i.e. for the estimation of delays) which require traffic flow as an input. Another solution is to carefully select the locations of the detectors and place them in sections where those anomalies rarely occur.

A final remark, concerns the presence of parked vehicles on the rightmost lane at specific parts of the arterial. This subject will be discussed in detail in the following chapter and the appropriate modifications to the simulation will be described to achieve resemblance with reality.

2.4 Traffic estimates

2.4.1 General

The availability of reliable traffic estimates is vital for every signal control strategy as it provides the magnitude of traffic for which the signal timings will be optimised. In traditional fixed time control these estimates are retrieved from historical data and refer to a number of *representative* periods which usually resemble the most characteristic traffic profiles within a typical day. In vehicle actuated control, traffic estimates need to be retrieved in real time and be updated frequently, enabling thus the adaptability component to the implemented strategy.

Vehicle actuated signal control strategies are optimizing signal timings to respond to the actual demand in the network. This means that, in order for the strategy to make the appropriate decisions, an estimation of the demand for the near future is required. Many strategies use traffic measurements from a past time period and use them as estimates of the current traffic in the network. This period can range from the last 15 minutes, for strategies that select the best fixed time plan from a number of pre-specified options, to the last 60-90 seconds for *optimising* strategies like SCOOT, SCATS, TUC, etc. These strategies are referred to, as *reactive*, in the sense that they react to the current flow in the network. An alternative approach is to project the vehicle arrivals from a specific area upstream, to a specific point downstream, using short-term traffic counts. In this way, a more accurate prediction of vehicle arrivals is made, usually by using traffic models and allows for the optimisation according to a more accurate flow prediction. The latter strategies are known as *pro-active* or *anticipatory*.

Eventually, this chapter will address the following research questions:

4. *Should the strategy aim at pro-active or reactive signal control?*
5. *What methods can be employed for traffic predictions?*

This section first discusses a number of traffic estimation methods used in existing vehicle actuated strategies. Furthermore, it discusses a number of issues related to reactive and pro-active strategies, to clarify a number of advantages and disadvantages associated with each approach. Finally it presents a discussion which leads to useful conclusions for the definition of the methods that will be employed in this Thesis.

2.4.2 Existing methods

A large number of methods exist for obtaining traffic data for the signal timing optimisation process. The complexity of the methods clearly depends on the traffic measures and the accuracy required from the signal optimisation method.

Strategies like SCOOT, TUC, SCATS and GASCAP use the measurements from upstream detectors of a link to optimise the signal timings for the next intersection. These strategies can use different estimates, for example, SCATS, which makes use of detectors at the stop line, collects the gaps between vehicles and the unused green time in each cycle, while SCOOT , which utilises upstream detectors, uses the traffic flow on the measurement location during the cycle. However, they have in common the fact that they use detector readings for a given time period to optimise signal timings for the next period. In that sense, depending on the signal optimisation logic, the traffic engineer can select the required detector readings and the collection method to obtain the most suitable estimates.

Contrary to the use of traffic measurements from past time periods, a number of methods make projections of traffic from an area upstream to a stop line downstream. These methods may require the exploitation of detectors many intersections upstream from the stop line of interest. A representative method of this kind is the PREDICT algorithm which is used in the RHODES system.

The PREDICT algorithm (Head, 1995) uses the actuations of upstream detectors together with the upstream signal decisions and the traffic state in the upstream intersections to determine the vehicle arrivals at the downstream intersection. The use of detector actuations from upstream detectors allows for a long time horizon of predictions to be adopted but also increases the complexity of the algorithm. This approach enables the consideration of the upstream signal decisions and the respective delays to be considered for the signal optimisation downstream.

To predict vehicle arrivals at the downstream location, detector actuations from all approaches of the upstream intersection need to be considered. Furthermore, for each vehicle actuation a large number of elements needs to be estimated. More specifically, the turning percentages (which might change through time) need to be estimated for each approach of the upstream intersection. Each vehicle that actuates an upstream detector is projected downstream based on a predicted travel time. This travel time is estimated by taking into account: the free flow travel time until the upstream intersection, the respective delay due to other vehicles, the delay due to a possible queue in the upstream traffic signal, the delay due to the signal itself, and the delay due to other vehicles and possible queues after passing the upstream intersection until it reaches the downstream location.

Obviously, each vehicle that actuates a detector should be monitored and its travel time should be updated at every time step until it reaches the downstream location. Moreover, a queue growth and decay estimation should be made continuously. If the upstream intersection also operates under vehicle actuated control, its signal decisions might change over time therefore the delay due to the signal, and the growth and decay of queues will also be subject to further changes. In this way, at each time step, the algorithm returns a traffic arrival prediction for a future time horizon at the downstream location. This method includes monitoring of every vehicle that actuates a detector until it reaches the downstream location, at every time step. It is clear that the complexity for the development and the calibration of such algorithms is quite a large task, however a successful algorithm can provide very useful predictions.

An alternative approach which lacks mathematical elegance but has proved to be quite promising in terms of the accuracy of its traffic predictions is by using *artificial neural networks*. All neural networks share some basic common features. They are composed of a number of very simple processing elements, known as neurons. These elements receive data from a number of sources and calculate an output which depends on some way on the values of the inputs, using an internal "transfer function". The neurons are joined together by weighted connections; data flows along these connections and is scaled during transmission according to the values of the weights. The output of a particular neuron may therefore contribute to the input received by another. Naturally such a system is of little use unless it communicates with the outside world and so some connections receive data from an external source, whilst others pass data back out. The neural network's functionality is very much dependent on the values of the connection weights, which can be updated over time, enabling the neural network to adapt (Dougherty, 1995).

There are various neural architectures, but the most commonly used in traffic forecasting is the multilayer feed-forward network trained by back propagation (Vasudevan and Chang, 2006). A common formulation of this type of neural networks consists of one input layer, one hidden layer, and one output layer. The input layer receives information on real time, i.e. from the detectors, whereas the output layer sends the predicted time-dependent traffic measure back to the control system. In a multilayer, feed-forward neural network, the feed-forward process will determine only the output of each processing element, based on the current input pattern and the weight connections (Chang and Su, 1995). Finally, a disadvantage of using neural networks for traffic predictions is that they need a large number of observed data for their "learning" process. A neural network, will not assign the "correct" weight factors from the beginning but will only "learn" to output correct traffic predictions after a period of "training". This period of training includes continuously modifying the weights of the connections by comparing the network outputs (predictions) with observed values.

A different approach for obtaining the required traffic estimates is by using on-line or real-time simulation. This means that a microscopic simulation system is receiving real-time detector data where the objective is to model the prevailing traffic situation as realistically as possible. The on-line simulation can be used for monitoring the overall status of the traffic system despite incomplete detector input. An example of such a simulation system is HUTSIM, which was developed for operation along with the HUTSIG signal optimisation algorithm (Kosonen, 2003).

Finally a number of *hybrid* approaches for traffic predictions exist. An example is the method used in OPAC. The length of the optimisation stage is divided into two parts: the “head” (length r) and the “tail” (k - r, where k is the stage length). The flow data for the head are obtained from the upstream detectors of the same link, and the flow data for the tail are estimated from a model. An optimal policy is then calculated for the entire optimisation stage but implemented only for the head section. The projection period is then shifted (rolled) up to r units ahead, new flow data are obtained for the optimisation stage, and the process is repeated. This approach is very promising and could potentially be altered, to be combined with simpler estimates for the “tail” period (such as using data from past time-periods).

2.4.3 Reactive vs Pro-active

As mentioned above, reactive strategies use traffic counts for a past time period to optimise signal timings for a future time period. The length of the time period can vary according to the scope of the application. For example, in strategies like TUC, counts of the current number of vehicles present in a link are used to optimise traffic signals, for an infinite future time horizon. The optimisation procedure is repeated at time intervals equal to the cycle (therefore of the magnitude of 60-120 sec). The infinite time horizon ensures that the control does not make myopic decisions and the repetition of the optimisation ensures that the decisions are always in accordance with current demand in the network. However, the fact that the optimisation cannot be made in intervals smaller than the cycle has certain disadvantages which are discussed later on.

Pro-active strategies generally make use of projections of vehicles. The projections are made from detector actuations upstream, on a future time instant at a location of interest downstream. The downstream location is usually the stop line of an intersection while the upstream area can vary depending on the application. On strategies designed to operate on isolated intersections like GASCAP, it is usually assumed that predictions of vehicle arrivals at the stop line can be based on detector actuations upstream, on the same link. However, this assumes that the upstream detectors are quite far from the stop line and that no significant generation or termination of vehicles takes place in between. On strategies intended to operate on arterials with more closely spaced intersections these assumptions will obviously not hold. In such cases, the projection of vehicle arrivals to a specific stop line might need to be made from an area including detectors upstream from the previous intersections. This will eventually include the upstream intersection signal decisions on the prediction process, like in the case of PREDICT algorithm. This has certain implications that are discussed below.

The upstream signal decisions will eventually affect the time that each vehicle will arrive at the downstream stop line. More specifically, they will affect the queue length on the upstream intersection, the vehicle waiting time to receive green and the vehicle outputs continuing to the downstream intersection. Therefore when using an arrival flow profile to optimise traffic signals, the new optimised plan will alter the initial traffic predictions at the downstream locations. In this way the problem needs to be solved dynamically, resulting thus at more complicated and computational costly algorithms. Furthermore, the development of such algorithms is not an easy process. Apart from building a good traffic model, the algorithm should be extensively tested and calibrated to produce reliable estimates which eventually agree with the measured values. On the other hand, an attempt to develop a vehicle projection method which is not very sophisticated might not provide better results than a simpler and robust use of traffic data from past time periods.

Concerning reactive strategies, a disadvantage is that they make decisions concerning the next cycle and implement these decisions without re-evaluating them during the implementation period. The decision interval equal to the cycle means that if the initial decision proves to be sub-optimal (i.e. due to imperfectness of the flow prediction), the strategy cannot react before the next cycle. Secondly, if predictions concerning the vehicle arrivals are made, these will be based on the sum of arrivals (detector actuations) in the past, therefore an average flow is obtained. However, despite the fact that this might be a quite good approximation, it does not provide any information on the arrival procedure (seed) during the optimisation period. Finally, using traffic data from previous time periods to optimise for the next time period means that the reaction of the strategy in sudden flow changes might not be quick enough. This disadvantage can be mitigated though, with more frequent updates of the traffic estimate (based i.e. on a rolling horizon with a small time step), so as to adopt to the changing traffic conditions more quickly.

Despite the shortcomings mentioned above, reactive approaches are simpler in their development and need less input data compared to pro-active. Furthermore, they are less sensitive to errors and to hardware failures (as less detectors are required). However, they will not provide the traffic predictions with the same resolution of more integrated models.

2.4.4 Discussion

Two issues are important when making traffic predictions: The first is the length of the future horizon for which the predictions will be made. The length of the prediction horizon defines the ability of the strategy to optimise signal decisions. If the horizon is short (i.e. a few seconds) then the strategy can only make decisions concerning the extension of the pre-mature termination of a stage and cannot plan stage sequences rationally for a longer period. However, if the horizon is long, the predictions unavoidably become less reliable.

The second important issue is the number of prediction points over the horizon (or the resolution of the prediction). To make the importance of the resolution clearer, consider the signal timing optimisation problem given two possible predictions of traffic flow for a future time horizon, as depicted in Figure 1. Each arrival pattern represents the number of vehicles to arrive at the next intersection for a given time interval. Both arrival patterns are identical until time t_0 when the signal control algorithm has to optimise signal timings for the next period. In the first case, the demand occurs immediately following t_0 whereas in the latter case the demand is zero immediately after t_0 and rises after a some time. In both cases, the total demand over the horizon is equal, however the optimal signal timings between the two cases might well be different.

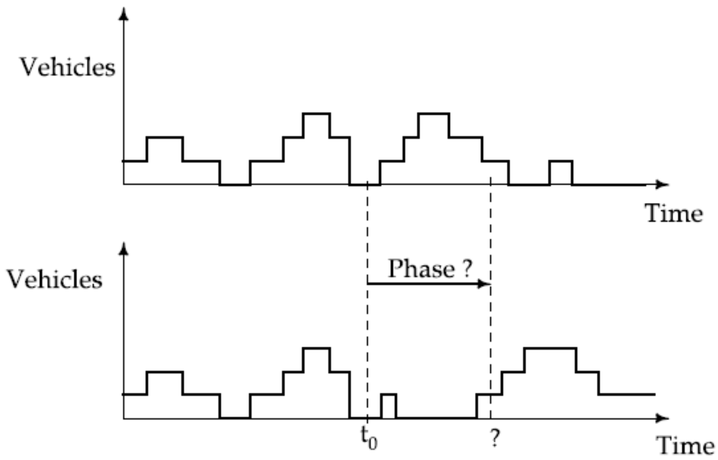


Figure 1 Graphical representation of two possible future vehicle arrivals (Source: Mirchandani and Head, 2001)

Obviously the arrival profile for a specific location is dependent on the signal decision of the upstream intersection. In sufficiently dense traffic conditions in the upstream intersection, the departures (arrivals towards the location of interest) will be equal to the saturation flow when the signal is green and zero otherwise. A simplifying approach would be to consider the average of the arrival flow for the whole time horizon, instead of taking into account the flow oscillations occurring from the upstream signal decisions. Such a model describes a continuous (uninterrupted) average outflow from each network link (as long as there is sufficient demand upstream and sufficient space downstream). The consequences of this simplification is the price to pay for avoiding the explicit modeling of intra-cycle red-green switches which would render the resulting optimisation problem discrete (combinatorial) and would lead to exponential increase of computational complexity for its exact solution as in several previous works (Aboudolas et. al., 2009).

The decision whether the strategy should result to reactive or pro-active control depends on the method employed for traffic predictions. Pro-active control requires vehicle projections from areas upstream to specified downstream locations. This can be achieved by various methods such as those described previously. However, integrated prediction algorithms like PREDICT are not publicly available for implementation while their development from scratch and more importantly their calibration until they yield reliable estimates is a significant research effort. Such a work cannot be carried out within this master thesis, therefore there is a need to adopt a simpler method for traffic predictions. Concerning neural networks and on-line simulation models, although they are promising venues for future research, it is considered that the lack of sufficient theoretical knowledge is a major obstacle for their development within the available time.

However, reliable traffic estimates can be obtained without necessarily adopting the above methods. It should be born in mind that, eventually, the traffic estimates needed depend on the signal optimisation algorithm. An efficient hybrid approach, developed in this thesis is described in chapter 4, where data from upstream detectors on the same link are used for making accurate arrival projections for a short horizon and estimate more crude arrival values for a long horizon. This method, implemented on a rolling horizon, proved to be promising for providing good traffic estimates to the optimisation modules.

2.5 Signal optimisation methods

2.5.1 General

This section presents a number of methods employed by various signal control strategies for the estimation of the optimal signal timings. These methods/algorithms are the core of each strategy therefore, their properties eventually define the effectiveness of each strategy and its applicability to a specific network. A discussion of the potential of using each algorithm for the strategy of this Thesis is also attempted.

The role of the optimisation algorithm as a part of the vehicle actuated signal control strategy is very specific. It estimates (or evaluates alternative) signal timings, aiming to minimise a performance index, for **given** traffic flow, arrival profile, time horizon, upstream signal decision and initial intersection conditions. In controlled networks with more than one intersection, it is evident that the objective of finding the optimal signal plans for a given intersection is dependent on the signal plans of the upstream intersections.

The formulation of the real-time optimised signal control problem is significantly different between various strategies, basically depending on their nature (reactive – pro-active). The following sections present various approaches towards estimating the optimal signal timings, based, each time, on their own problem formulation.

2.5.2 Heuristic (rule-based) techniques

Heuristic (or rule-based) techniques are commonly used in vehicle actuated signal control (Yagar and Han 1994, Dion and Hellinga 2002) mostly because of their simplicity but also because of the ability they offer to include influences in traffic operations that are difficult to be captured by analytical models (i.e. the effects of transit vehicles). Heuristic techniques in signal control are based on the fact that, in general, traffic is controlled by rules. A heuristic approach can provide satisfying control related to how the system should behave, although it does not search for, or guarantee, an optimal solution (in the sense of a global optimum).

Heuristic optimisation techniques basically generate a set of candidate control strategies to serve the expected demand. These strategies correspond to a set of lists of priority rules for switching the traffic signal. Each list contains a set of candidate strategic events for switching signal phases. These lists might be developed by experts in traffic control. Key events in the list might include, for example, the arrival of a transit vehicle, the dissipation of a queue or even the queue reaching a critical length. The ordering of events in a list reflects relative priority and is certainly dependent on the strategy objectives. Furthermore, the greater the capability of the processing computer, the greater the number and size of candidate lists that can be considered. This limitation requires the engineer to carefully select and order the important events, minimising effective redundancy. After the generation of the candidate signal plans, usually, an additional plan-selection technique should be employed to select the optimal signal plan based on some performance index.

Heuristic optimisation techniques can efficiently be used by adopting the discrete time, rolling horizon process. The optimisation procedure basically accepts that signal switches occur after specific discrete events such as after a queue has reached a critical length, after a queue has just finished dissipating, or after a transit vehicle has been detected. At the same time, all events that have no importance for the signal operation should be ignored so as to reduce the number of potential switching combinations that need to be considered to find an optimal solution. A representative example of a list of traffic events that are considered for the generation of the candidate signal plans is provided in the SPPORT model of Dion and Hellinga (2002):

1. If a stop line queue of n vehicle exists and is not being served, start serving the queue as soon as possible.
2. Switch the signal display to green if the stop line queue exceeds a user-defined length.
3. Maintain the current green signal indication on an approach on which the reach of the upstream queue exceeds a user-defined location.
4. Switch the signal to red if a queue on one of the approach's user-defined major exit links threatens to spill back across the intersection.
5. If a platoon of n vehicles or more is approaching the intersection, switch the signal display to green at a time that will allow the platoon to cross the intersection without being affected by vehicles stopped at the stop line.
6. If a queue of vehicles is being served, continue serving the queue.
7. If a platoon of vehicles is being served, continue serving the platoon.
8. If a transit vehicle is approaching its transit stop, switch the signal display to green at a time that will allow the vehicle to proceed uninterrupted up to its loading point.
9. If a transit vehicle is approaching the stop line, switch the signal display to green at a time that will allow the vehicle to cross the intersection without having to stop.

Of course the aforementioned traffic events will not be assigned with the same weights, as these will be primarily dependent on the strategy objectives. Using these weights and the demand estimate for the next planning horizon, the signal optimisation algorithm will generate requests calling for either a green or a red signal indication on specific approaches at specific times. After all the requests have been generated, the model will generate the respective signal switching decisions.

The use of the list of events presented above will allow the strategy to determine the relative importance of the various traffic events; however, it might not be clear how to determine beforehand which event should have the highest priority. To solve the above problem, the user is permitted to provide many candidate prioritized list of events for consideration by the model. When more than one list is provided, the signal optimisation algorithm generates a candidate timing plan for each list and then selects for implementation the one yielding the best performance measures. The best switching plan is selected on the basis of a performance function that can include several performance measures such as delays, stops, queues etc.

2.5.3 Fuzzy logic in signal control

Heuristic rules are also used by fundamentally different signal control strategies; namely by strategies based on fuzzy logic. Fuzzy logic is a form of multi-valued logic derived from fuzzy set theory to deal with reasoning that is approximate rather than precise. In contrast with "crisp logic", where binary sets have binary logic, fuzzy logic variables may have a truth value that ranges between 0 and 1 and is not constrained to the two truth values of classic propositional logic. The term *degree of truth* is introduced to account for the quantification of this concept.

Fuzzy control has proven to be successful in problems for which an exact mathematical modeling is hard or impossible but an experienced human operator can control the process sufficiently good. As far as its use in signal control is concerned, fuzzy logic offers a tool for estimating signal timings in a way an experienced expert would do (using rules and perception), rather than optimising them through mathematical techniques. An advantage of fuzzy logic control is that it allows linguistic and inexact traffic data to be manipulated. The *fuzziness* of signal control can be divided into three levels: input, control, and output level. In the input level, only a partial picture of the prevailing traffic situation through measurements exists. In the control level, many possibilities exist; however, it is not certain which is the best, because the cause-consequence relationship of signal control cannot be explained. In the output level, the crisp control decisions are obtained (for example splits).

Fuzzy control can be applied in complex problems with multi-objective decisions, such as the signal control problem, where several traffic movements compete for the same time and space and different priorities and objectives exist. The design of a fuzzy controller for this system requires the expert knowledge and experience of traffic control in formulating a linguistic protocol, which generates the control input, to be applied to the traffic control system (Niittymaki and Pursula, 2000)

The rules and conditions in signal control are typically vague and it is difficult to say when a queue is, for example, “long”. In fuzzy logic, *membership functions* can be used to define terms like “short-queue” and “big delay”. A membership function is basically a graph that defines how each point in the input space is mapped to a membership value between 0 and 1. This value eventually defined the *degree of truth* of the statement. Vague arguments can then be used in the rule base using expressions such as: “if queue is long then extend green” (Kosonen, 2003).

A number of applications of fuzzy logic in signal control have been developed, mentioning but a few Anderson et. al., (1998) Niittymaki and Kikuchi (1998), Kosonen, (2003) and

Schmöker et. al., (2008). However, most applications found in the literature refer to the control of isolated intersections or to simple vehicle actuated applications (i.e. only green extensions) and not full optimisation or arterial-wide applications. The main advantages claimed for expressing control laws in this way are that it enables to capture the knowledge of how the system should work in linguistic terms and the laws are naturally broken down into individual if-then statements that lend themselves to parallel processing. However, fuzzy logic basically has the same main disadvantage as the heuristic approaches, namely the rule-base should be as complete as possible, while missing an important rule or over-valuing another has a significant impact on the signal operation.

2.5.4 Dynamic programming

Dynamic Programming (DP) has been employed by various contemporary signal control strategies as an efficient method to estimate the optimal signal timings given an initial state, a set of traffic predictions and a planning horizon. Usually, the strategies using DP optimisation algorithms also adopt the rolling horizon approach for signal control so as to optimise for a future horizon comprising of several time steps, but only implement the decisions for the first step and then re-optimise for the next horizon.

The strategies that use DP formulations to optimise signal timings adopt a specific problem formulation, which is well described by Shelby (2004) as follows:

The signal optimisation problem is formulated as a discrete time look-ahead search problem, which is solved each time step, t, to find the optimum control action, u_t^* , to minimise delay over an immediate, short-term planning horizon. The planning horizon, or optimisation horizon, is most often a finite time interval, such as $[t, t + H]$, where H is the number of time steps in the planning horizon. This formulation is as follows:

$$u_t^* = argmin_{u_t \in U(s_t)} f(s_t)$$

where:

t	is the time step
$U(s_t)$	is the set of possible control actions,
u_t^*	is the optimal control action,
s_t	is the signal state,
$f(s_t)$	is the state value function (cost-to-go), i.e. the minimal cost possible from state s_t to the end of the planning horizon $f(s_t) = \begin{cases} c(s_t) + minf(s_t, u)\,, & if\ t < t_H \\ c(s_t) \qquad\qquad\qquad , & if\ t = t_H \end{cases}$
$c(s_t)$	is the cost of being at state s_i (estimated i.e. by a delay formula).
$f(s_t, u)$	is the state-action value function (cost to move to the next state).

Therefore, they consider a time horizon for which the signal timings will be optimised, divide it into time steps of a fixed duration creating thus a decision tree which starts from the initial time and leads to the end of the optimisation horizon through the consecutive times steps and decision points. The signal states are allowed to change at each decision point. The problem is eventually to find the optimal path from the current state to the end of the horizon.

In general, there are two approaches to DP formulations, backward and forward. Backward DP, which is usually solved by a recursive fixing method (i.e. like the Dijkstra algorithm for solving the shortest path problem), requires complete prior knowledge of the DP network states for computation (i.e. the link costs). For the optimal signal control problem (optimal signal switches over a horizon), if backward recursion is used, it is not possible to know the boundary states in advance to start the recursion. However, if a forward DP formulation is adopted, the relevant DP network states can be generated as the algorithm proceeds, removing the computational burden of searching the entire DP network prior to algorithmic implementation. This is the reason most DP optimisation algorithms use forward recursion (Kim et. al. 2005).

To formulate the signal control problem of minimising a selected performance measure for an intersection during the control horizon, using DP, it is crucial to define an appropriate state for the problem. The state can be defined by a set of indicators (i.e. time, delay, current stage, elapsed green time etc) for a given decision point, at a specific time instant. State in dynamic programming is so fundamental that if it is properly defined, the DP formulation can be completed by providing a dynamic recursive functional equation based on the state defined. Different DP state definitions for a problem are possible, depending on the underlying decisions used to formulate the problem, ultimately resulting in different DP formulations. A typical formulation is, at each time step to decide which stage will be green until the next decision point (start of next time step). Each decision leads to a new state. The state should incorporate sufficient information about preceding decisions made to complete the decision tree arising from this point.

The most imminent example of a DP algorithm applied to signal control is COP (controlled optimisation of phases) (Sen and Head, 1997), the algorithm used in RHODES. RHODES, consists of a collection of several algorithms among which, is the arrival prediction algorithm PREDICT (see previous section) and the adaptive intersection control algorithm COP. COP is an approximate forward dynamic programming method that utilizes a partitioning of the state space into phase-wise stages rather than the typical formulation of partitioning the state space into stages over time. COP proceeds in the problem solution by identifying equivalent states, such that no redundant calculations are wasted on their equivalent sub-trees. Two states are considered equivalent if they have just switched into the same phase at the same time step and have switched phases the same number of times since the initial state. Whichever of the two equivalent states has experienced the least cost (i.e. delay) to arrive at the current time step is stored at the current stage, and the other state is discarded.

PRODYN (Farges et. al., 1983) is another forward dynamic programming technique for solving the problem formulation presented in the beginning of this chapter. Like COP, PRODYN also uses a state equivalence criterion to prune candidate solutions from the decision tree, thus achieving algorithmic efficiency. The state equivalence criterion of PRODYN suggests that two states are considered equivalent if they are at the same time step in the tree, have just switched to the same phase and have approximately equal queue lengths on all lanes. Stages of the dynamic program are partitioned by the time step. Starting with the initial state as the initial stage, all reachable nodes in the subsequent time step (i.e., the next stage) are realized. The state is augmented to incorporate the vector of control actions it has taken since the initial state. Then data storage affiliated with the previous stage is freed, and from the current stage, all possible successor states appearing in the next stage are evaluated. If any two states in the same stage are deemed equivalent, the state reached with the least delay is stored while the other is discarded.

The DP formulations for solving the real time optimised signal control problem are very efficient; however, a significant disadvantage is that they are not able to cope with several intersections (especially as the number of stages of each intersection increases). This is a fact due to the computational effort needed to solve the optimisation problem within each time step.

2.5.5 Explicit/Implicit enumeration techniques

Two optimisation methods that yield an optimal solution to the signal control problem are the explicit and implicit enumeration technique. As their name suggests, their target is to evaluate the complete set of possible signal decisions for a time horizon and select the optimal. To use these techniques, the problem should be formulated in a similar way as for the DP algorithms, i.e. adopting a time horizon divided into discrete time steps and allowing signal switches at the end of each step creating thus a decision tree expanded until the end of the time horizon.

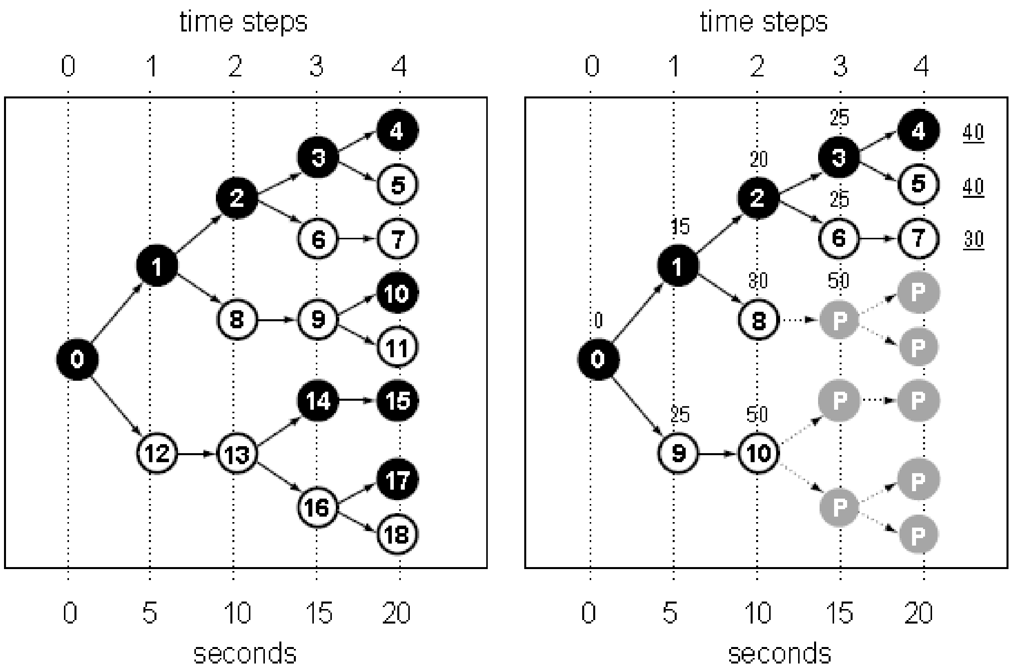


Figure 2. Graphical representation of the explicit (left) and implicit (right) enumeration techniques (source: Shelby, 2004)

The most straightforward approach to solving the signal optimisation problem is to explicitly enumerate (evaluate) all feasible control paths in the solution space (decision tree) and to simply select the lowest-cost solution. This technique is also referred to as exhaustive search method. The method implies that all possible combinations of signal switches until reaching the end of the control horizon will be evaluated in terms of the selected performance measure (i.e. delay) allowing thus the selection of the optimal (least cost) switching sequence. Figure 2 (left) illustrates a tree representation of all feasible control actions (links) and the resulting states (nodes) assuming a control horizon of 4 time steps and a two-phase intersection (phase 1 is black and phase 2 is white). The same Figure also enumerates the order in which states would have to be evaluated in explicit enumeration. Unfortunately, this is a slow algorithm, which is difficult to be practically implemented in long time horizons and more than two stages. On that purpose, constraints have been proposed to reduce the number of possible solutions, without a compromise in optimality, as is the case in the implicit enumeration technique.

Results equivalent to that of the explicit enumeration can be achieved by implicitly enumerating all solutions. The implicit enumeration method differs from the explicit enumeration by including a lower-bound variable, LB and an upper-bound variable UB. The variable LB accumulates the cost (i.e., delay) from the initial search state to the current state such that $LB_j = LB_i + c(s_j)$. The upper-bound variable, UB, serves to store the maximum cost occurring from the full horizon signal plans evaluated so far. Therefore, the upper bound is defined as follows:

$$UB = \begin{cases} LB_i, & if\ t_i = t_H\ and\ LB_i < UB \\ UB, & otherwise \end{cases}$$

Before the search commences from the current state, s_i, the lower and upper bounds are initialized such that $LB_t = 0$ and UB equal to a very large value (or infinity). Whenever a state is evaluated such that $LB_i \geq UB$, the search in the specific branch is terminated and the state value function $f(s_i)$ is set to infinity (or an otherwise large number greater than the upper bound) in order to prevent the controller from choosing the action associated with the pruned state as the optimal control action. When the algorithm commences, the lower bounds are successively updated at every subsequent state. The upper bound is lowered whenever a lower-cost, full-horizon plan is achieved. When a state is reached when $LB_i \geq UB$ the recursive node expansion from these states is preempted. Sub-trees emanating from these states can only experience non-decreasing delay and thus need not be enumerated. An indicative illustrative search resulting by an implicit enumeration technique is shown in Figure 2 (right).

2.5.6 Store and Forward methods

Store and forward modeling, which has originally been introduced by Gazis and Potts (1963), has been exploited for formulating the optimal signal control problem in a number of strategies. The most well-known application of store-and-forward modeling has been proposed by Diakaki et. al., (2002) in the TUC strategy.

The main idea when using store-and-forward models for signal control is to introduce a model simplification that enables the mathematical description of the traffic flow process without using discrete variables. This simplification has to do with using the average outflow from a link during the whole cycle, instead of the actual outflow when the signal is green and zero flow when the signal is red. This assumption provides the advantage of avoiding the use of discrete variables; however it also results in an inability of the strategy to react in time intervals smaller than the cycle, it prevents the model to take into account the oscillations of queues in the links due to the upstream signal switches during the cycle and the effect of offsets for consecutive intersections cannot be described by the model.

Nevertheless, the aforementioned assumption allows for the development of an efficient and relatively simple control algorithm for the determination of optimal splits. The use of the simplifying assumption for continuous traffic flow leads to the following linear state-space model for road networks (Papageorgiou et. al., 2003):

$$\boldsymbol{x}(k+1) = \boldsymbol{x}(k) + \boldsymbol{B}\Delta\boldsymbol{g}(k) + \boldsymbol{D}\Delta\boldsymbol{d}(k)$$

where k is the time-step, the state $\boldsymbol{x}$ is the vector of the numbers of vehicles x_i in network links i; **B** and **D** are constant matrices reflecting the network characteristics; $\Delta\boldsymbol{g} = \boldsymbol{g}(k) - \boldsymbol{g}^N$ and $\Delta\boldsymbol{d} = \boldsymbol{d}(k) - \boldsymbol{d}^{\boldsymbol{N}}$; $\boldsymbol{g}$ (the control input) is the vector of green times g_i for each stage i in all intersections of the network, while $\boldsymbol{g}^N$ comprises some corresponding constant nominal green times g_i^N; $\boldsymbol{d}$ (the disturbance vector) and $\boldsymbol{d}^N$ comprise the demand flows and the constant nominal flows respectively. Suitable bounds for minimum green times and maximum storage capacity of links must also be considered.

The objective of this strategy is not to minimise a traffic measure (like delays) but to minimise the risk of oversaturation, i.e. avoid queues spilling back to previous intersections. To minimise the risk of oversaturation, one may attempt to minimise and balance the links' relative occupancies $x_i/x_{i,max}$ where $x_{i,max}$ (in veh) is the storage capacity of link i. This leads to the definition of a quadratic criterion and an optimisation problem which can be solved quickly even for large networks. Under specific assumptions for the optimisation horizon, one can obtain the following Linear-Quadratic Regulator problem (Diakaki et. al., 2002):

$$\boldsymbol{g}(k) = \boldsymbol{g}^N - \boldsymbol{L}\boldsymbol{x}(k)$$

where **L** is a control matrix which depends upon the previously mentioned optimisation problem parameters and can be calculated offline. In this way, optimal splits for the control zone can be calculated in real time, and be implemented at the start of each new cycle.

The TUC strategy optimises green splits for each intersection, assuming fixed cycle time and offsets, which are estimated by other independent algorithms. Moreover, it assumes constant and known turning proportions for each intersection approach and saturation flows for each outgoing link.

To apply the linear-quadratic control law, availability of measurements for all state variables is required in real-time. However, the numbers of vehicles x_z are usually not directly measurable, unless video detection systems are utilised. For this reason, local occupancy measurements o_z; collected in real time by traditional detector loops, may be transformed into (approximate) numbers of vehicles via suitable nonlinear functions (Diakaki, 1999).

2.5.7 Discussion

Significant differences between the optimisation algorithms described in this chapter can be observed. In a way, this reveals the diversity of existing approaches for solving the optimal signal control problem. The development of some algorithms is more difficult compared to others; however, it cannot be claimed that some have clearly superior performance. All algorithms can be used efficiently and can yield significant benefits (whatsoever the efficiency measures are), depending on the control objectives and the level of integration of the algorithm (i.e. simple green extension vs full optimisation of cycles, splits, offsets etc). In conclusion, there is no “suitable” algorithm; all optimisation methods can prove suitable, if they are tailored successfully to the specific problem. The most important factors that seem to affect this decision are the extent of the control area, the strategy objectives, the computational effort needed and the methods employed for traffic state prediction.

The general *look-forward* framework employed by a large number of strategies makes use of a time step and a planning horizon. Signal decisions are planned for the next time horizon comprising of a number of discrete time steps; however, only the decision for the first time step is implemented and the future plan is re-evaluated moving on to the next horizon. The concept of this approach is illustrated in Figure 3:

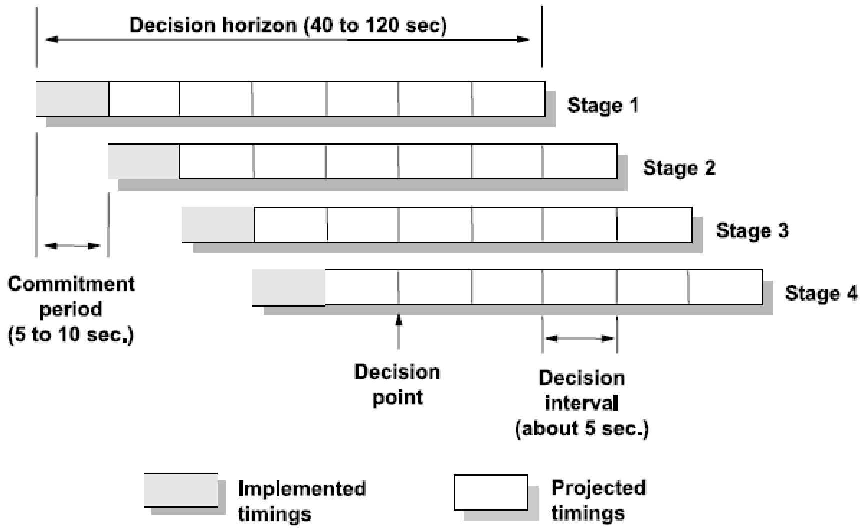


Figure 3. The rolling horizon concept (source: Dion and Hellinga, 2002)

This approach can prove very efficient for various reasons:

- It updates its decisions frequently therefore adapts easier and faster to the prevailing traffic conditions,
- It avoids making myopic decisions as it optimises for the whole horizon and not only for the near future,
- It allows several optimisation techniques to be employed

A drawback in this approach is that, due to the computational complexity of some of the algorithms presented (especially as the number of intersections, the number of stages in each intersection and the length of the horizon increase) the time step of the horizon cannot be small (i.e. 1 second) as the decision tree becomes too complex resulting in a vast number of evaluations that cannot be performed in real-time. A typical value for the time step is 5 seconds; however, a resolution of 5 seconds might sometimes not be small enough to accurately capture the evolution of traffic, i.e. the exact time of the dissipation of a queue or the arrival of a platoon. Despite that fact, it is considered that this approach can prove superior to traditional “one-off” optimising strategies, which make restricting decisions for long time horizons (i.e. a cycle period).

2.6 Related aspects

This chapter refers to a number of aspects that were not reviewed extensively as they were not directly related to the research questions, but are of interest for the development of the vehicle actuated strategy.

2.6.1 Stage sequence flexibility

An important characteristic of many vehicle actuated strategies which adopt the rolling horizon approach is that they allow for a non-fixed stage sequence to be adopted. Moreover, strategies like RHODES or CRONOS, also allow for non fixed stages. In this way, all signal groups that can run concurrently are allowed to form a stage and consequently, the intersection is described only as a set of safety constraints on the traffic signal groups. This approach allows for much flexibility in real-time control as it can adopt better to the prevailing traffic conditions. However, in practice many traffic agencies prefer to maintain a fixed number of stages and a fixed stage sequence. The main reasons to do this are:

1. Firstly, for safety purposes. Drivers, and especially those who are familiar with the area (i.e. on routes from home to work) become familiar with the stages and the stage sequence therefore “expect” the right of way at specific moments. Moreover, in Athens, motorcyclists moving between lanes, overtake stopped vehicles at intersections and stop after the signal head without having a view on it. This behavior occurs more frequently when drivers are aware of the stage sequence and know exactly when they will receive the right of way.

2. Secondly, a fixed stage sequence can allow for significant computational effort to be avoided thus the algorithm to run faster. This is because each time a decision between the existing or the next stage has to be evaluated, instead of the existing and all other possible stages. If the number of stages is n and the allowed signal switches occurring within the horizon are s, the *maximum* number of signal plans that need to be evaluated by an algorithm are n^s. Therefore, for $n = 2$ the number of candidate solutions is significantly lower.

3. The above reduction in the number of possible signal plans, when a fixed stage sequence is adopted allows for a longer time horizon to be evaluated therefore, the strategy takes less myopic decisions.

In the strategy that will be developed in this Thesis, the stage sequence will be kept fixed, to satisfy the requirement of applicability in real-life situations. This subject is discussed in further detail in chapter 4.

2.6.2 Signal coordination

An issue that needs to be taken into account when designing signal control strategies intended to operate on arterials is signal coordination. Coordination is critical for traffic operations and for the eventual success of every strategy operating in successive intersections.

Signal coordination schemes have been well documented in the literature, especially for traditional fixed-time control. Applications like PASSER II (Chang et. al., 1988), MAXBAND (Little et. al., 1981), MULTIBAND (Gartner et. al., 1991) and TRANSYT-7F (Wallace et. al., 1998) allow for the development of appropriate offsets, estimated offline. These applications have also been extended to operate within on-line optimisation schemes to mention but a few REALBAND, (Dell'Olmo and Mirchandani, 1995) which is the extension of MULTIBAND for online operations and is used in RHODES, or a modified version of MULTIBAND introduced by Vasudevan and Chang, (2006) for their arterial signal control model.

Integration of the appropriate offsets within a vehicle actuated signal control strategy can be made at a higher level from intersection control. In this way, the appropriate offsets are estimated by a module that operates independently from the split optimisation module, and its results are considered as a set of constraints imposed to the intersection optimisation. In this way, the two modules do not interfere but cooperate to produce signal decisions that optimise traffic operations both locally and globally.

3. Study area design

This chapter is concerned with a detailed presentation of the study area and its characteristics, which are used to formulate the strategy evaluation platform. The related work package (WP2) was divided into three Tasks, which are summarized in the following Table.

Table 1. Work Package 2 objectives

WP2 Study area design	
Task 1 Traffic data / Study area	Exploration of available traffic data (online and manual traffic counts) Formulation of demand scenarios Analysis of the study area particularities.
Task 2 SITRAFFIC P2	Design of the network and existing fixed time signal plans in P2.
Task 3 VISSIM	Transfer of data from P2 to VISSIM. Development of the simulation in VISSIM. Test simulations and refinements until the achievement of realistic driving behavior (resembling behavior in the study area).

Before presenting the outcomes of each task, general information concerning the selection of the study area and its particularities are presented. Subsequently, the exploration of the traffic data is illustrated and the traffic demand scenarios which are adopted to test the efficiency of the strategy are formulated. The study area analysis provides insight on the research question 2:

2. *How will the particularities of the study area guide us in the definition of the strategy objectives?*

In the next section, the use of SITRAFFIC P2 for this project is presented. The layout of the intersection used for the evaluation and its existing fixed-time signal plans are implemented in the software.

Next, the development of the VISSIM simulation is presented. Based on the previous sections, the traffic data and the P2 outcomes are exploited to form the basis of the design of the network in VISSIM. Several additional parameters are discussed concerning the achievement of a realistic simulation which resembles the area characteristics in a realistic way.

3.1 Study area selection

Firstly, the criteria that were taken into account when selecting the study area are listed. The demand in the selected area needed to have varying profiles and pose significant challenges for the traffic engineer. More specifically, demand should change between days and hours to allow testing the reliability of the strategy. There should be good potential for improvement of traffic operations from the existing situation. Furthermore, the impact of the traffic conditions in the study area on other parts of the network (adjacent main arterials, central areas) should preferably be moderate to small. In this respect, it would be possible to focus on the optimization of the traffic conditions on the selected intersection. The intersection selected for the evaluation of the strategy should preferably have a complex layout with many traffic movements and subsequently several signal groups, so as to pose a greater challenge for the developed strategy. Finally, there should be measurement positions (detectors) along the arterial to allow for the exploitation of sufficient traffic measurements. Especially for the traffic flow on the major road, availability of reliable traffic measurements is important to study the traffic conditions in more detail (per hour, on special days, on incidents etc). If online measurements were not available for all traffic movements, at least manual traffic counts needed to be available.

An area found to satisfy all the above criteria was the main arterial called "Chamosternas", located near the centre of Athens. One of the busiest intersections within this arterial is the intersection of Chamosternas with Pireos, another major arterial expanding from the city centre to the port of Piraeus. This intersection was selected for the evaluation of the new strategy as it satisfied all the aforementioned selection criteria. The intersection layout is presented in Annex 1.

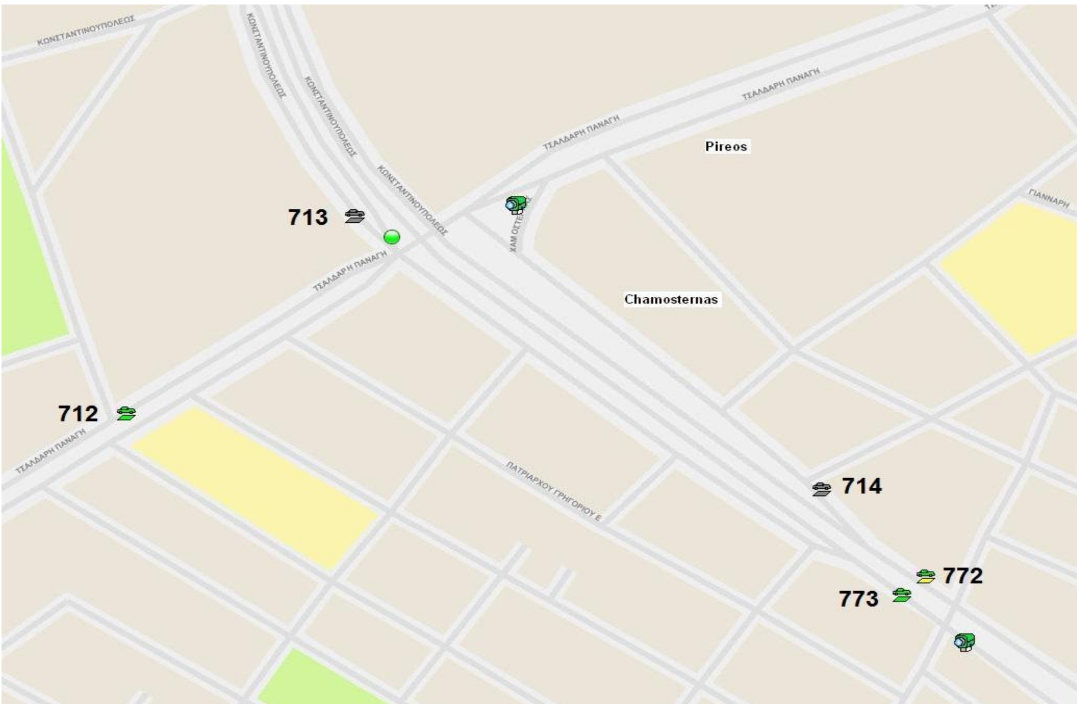
The land uses adjacent to the two arterials are exclusively commercial (stores, car selling companies, gas stations) while the land uses at the inner areas are exclusively residential. On-street parking in the area exists (as a result of non-existing parking areas close by) and as, in some parts, the right lane is under-utilized or not utilized at all due to parked vehicles. Subsequently, in some measurement locations, the rightmost detector might not be able to record any vehicles during some parts of the day. In those cases, the detector locations had to be changed for the application of the VA strategy, to correspond to locations where this problem did not occur. As will be illustrated later, good detector measurements are essential for the correct operation of the new strategy.

3.2 Traffic data

3.2.1 General

The intersection under evaluation includes 4 approaches and 16 signal groups (9 for vehicles and 7 for pedestrians). The south east approach includes two branches (per direction), the one leading below, into a tunnel and the other leading onto the intersection area. Traffic leading (or leaving) from the tunnel does not affect the implementation of the strategy as vehicles heading in and out of the tunnel do not come across any traffic signals and merge with the rest of the vehicles further downstream.

The following sections describe the data sources used in this project, the rationale behind the data selection, the methodology for the extraction of an appropriate data set using online data and manual traffic counts and finally, the demand scenarios used for the simulations in this project. Reference to the measurement locations around the intersection is made by using the numbering shown in Figure 1. The detectors currently on-line are shown as small vehicles with green color. Vehicles of grey color also indicate existing detectors which; however, were not functioning during the examined period.



Picture 1. Measurement locations (cars) around the intersection

Two sources of traffic data are exploited in this project: the first concerns **online data** from the available detectors and the second concerns data from **manual traffic counts**.

3.2.2 Finding an appropriate data set

Traffic flows do not only vary within the day, but also between days, seasons and years. However, for this project there was a need to select a specific period for which data will be derived to formulate demand scenarios to test the effectiveness of the existing and the new signal control strategies. The selected period was such that minimal special events, holidays etc exist within it and that the average daily flows are, statistically, as close to the Annual Average Daily Traffic (AADT) as possible.

Although online data from measurement locations are available for several years and can be retrieved at a disaggregate level (down to 1-min intervals), the manual traffic counts were not always available for very recent dates. The most recent manual traffic counts for the examined intersection were available for February 2005. It was eventually decided to use online data from the detectors for months February and March, where traffic flow patterns would be analogous to those recorded from the manual traffic counts.

After selecting the period, the discrepancies on traffic flows between weekdays were examined. It was concluded that for the examined network, Mondays tended to have lower flows while Fridays tended to have higher flows. The values from Tuesdays to Thursdays were closer to a weekly average. For that reason it was decided to obtain the average flow values for Tuesday, Wednesday and Thursday of each week. The data were obtained for 1-hour intervals, for each measurement location and are shown in the Table 2.

Table 2. Average flows (veh/h) per measurement location (for Tuesdays, Wednesdays and Thursdays, for February-March, 2008)

Time	712	773	772
0:00	798	761	784
1:00	761	601	643
2:00	513	498	538
3:00	477	480	524
4:00	456	484	512
5:00	548	561	623
6:00	883	836	825
7:00	1144	1043	1139
8:00	1547	1131	1376
9:00	1734	1155	1572
10:00	1698	1181	1543
11:00	1633	1276	1539
12:00	1580	1389	1512
13:00	1662	1463	1489
14:00	1672	1499	1457
15:00	1664	1503	1470
16:00	1685	1531	1482
17:00	1713	1551	1441
18:00	1645	1569	1409
19:00	1577	1490	1332
20:00	1487	1431	1185
21:00	1238	1362	1013
22:00	1096	1071	968

3.2.3 Building demand scenarios

To test the reliability of the vehicle actuated signal control strategy, its efficiency under varying demand conditions should be tested. On that purpose, different sets of traffic data will be used for the simulations. The fixed time signal plans of the Athens Traffic Management Centre cover four demand scenarios:

1. The morning peak,
2. The evening peak,
3. A scenario with balanced flows,
4. A night-time scenario with low traffic volumes.

These signal plans substitute each other within the day depending on the prevailing traffic conditions, measured from street detectors. In order to allow comparisons between the operation under the vehicle actuated strategy and the existing fixed time plans, the same four demand scenarios will be adopted for testing the vehicle actuated strategy. After retrieving the traffic data from the measurement locations, it was possible to divide the day in different periods corresponding to the aforementioned demand scenarios.

More specifically, by examining the traffic flows of Table 2, it can be observed that:

- From **01:00-06:00** traffic flows are significantly reduced, therefore this period is considered to correspond to the fourth (night-time) scenario.
- From **06:00-11:00**, flows are steadily increasing. This period corresponds to the first (morning) demand scenario.
- From **11:00-16:00**, traffic flows variations differ per direction, but overall, traffic is stabilised around the peak values. This period corresponds to the third demand scenario (balanced flows).
- From **16:00-19:00**, depending on the link, flows further increase or start decreasing. This (second) scenario corresponds to the evening peak period.
- From **19:00-01:00**, differences in traffic flows are again smaller, therefore it can be considered that this period corresponds to a second balanced flow scenario.

To obtain reliable results from the simulation, it was considered that a representative **2-hour period** for each scenario would be needed, plus **two 15-minute periods before and after**. The first 15-minute period served as warm-up for the network while the second served to assess the impact of the strategy after the target period (what are the resulting queues, delays etc, created by the strategy during the peak period). Therefore, each demand scenario covered a period of 2,5 hours in total.

For roads where detectors were not available, the measurements were retrieved from manual traffic counts. Data from the manual traffic counts were available only for the day period, from 07:00 to 16:00; however, a 24-hour dataset was needed to define all four demand scenarios. In order to obtain data for 24-hours, online data from adjacent detectors were exploited. More specifically, the derivation of the 24-hour dataset was based on the assumption that the percentage of the total daily traffic flow, for each hour of the day is constant (i.e. from 08:00 to 09:00, 5% of the daily traffic flow passes). For selected measurement locations, this percentage for each hour of the day was calculated. By using the same percentages for the secondary roads, the 9-hour data set (07:00-16:00) was expanded, pro rata, to a 24-hour data set.

3.2.4 Exporting the appropriate data to VISSIM

Following the previous assumptions and calculations, the measurements from the detectors and the manual traffic counts were exploited to define the inflows for the links in the boundaries of the VISSIM simulation, for each demand scenario.

Traffic data for the south western approach are based on measurement location 712, while for the south eastern approach, data from measurement location 772 were used (together with manual traffic counts). For the north western and north eastern approaches, manual traffic counts were exploited. Measurement location 773 was useful to perform consistency checks for the flows obtained for the rest of the approaches.

Finally, the turning proportions for each intersection arm were estimated from the manual traffic counts, assuming that these proportions did not change significantly over the last few years. It was also presumed that the turning proportions would not be constant within the day, therefore an examination of how turning proportions change within the day was attempted. The turning proportions are estimated per demand scenario, and are presented later on, in the description of the simulation environment.

Based on the data from Table 2, and the manual traffic counts for secondary roads, the inflows per VISSIM link, of the four, demand scenarios are formulated as follows:

Table 3. Link flows (hourly) per demand scenario

	Scenario 1 (morning peak)			Scenario 2 (balanced flows)			Scenario 3 (evening peak)			Scenario 4 (night-time)		
VISSIM Link	0-15 min	15-135 min	135-150 min	0-15 min	15-135 min	135-150 min	0-15 min	15-135 min	135-150 min	0-15 min	15-135 min	135-150 min
53	960	1730	1600	1600	1650	1680	1680	1700	1600	760	470	960
12	770	1330	1450	1450	1500	1600	1600	1440	1320	610	370	770
9	250	400	410	410	350	320	440	450	410	200	100	250
6	500	850	900	900	950	1000	1000	1100	900	450	350	500
104	1300	1560	1500	1500	1500	1500	1500	1430	1300	650	540	620

3.2.5 Additional observations

The literature review presented in the next chapter was the main driver for the selection of the strategy objectives. However, this section presents a number of observations related to the road network that provided additional guidance on the selection of appropriate objectives for the vehicle actuated strategy.

Avoiding the formation of large queues, needed to be taken into account, for specific locations. The space available for left turning movements is not always sufficient to accommodate the formed queues therefore these spill back and reduce the effective number of lanes for other traffic movements. For that purpose, efficient queue control was deemed important for the success of the VA control.

Another observation relevant to the formulation of the vehicle actuated strategy concerned the utilization of traffic detectors and the quality of measurements. At an urban arterial, where vehicles follow several different routes and change lanes frequently, detectors are not always capable to count all vehicles, as some vehicles might be passing between two detectors. Moreover, large vehicles can be counted by two detectors simultaneously, when travelling between lanes. Experience in the Athens TMC, has proven that the exploitation of traffic flows for the determination of green splits in a vehicle actuated strategy can be problematic for the above reasons. Careful consideration of the detector placement is considered critical so as to ensure that these effects are minimised in the locations where the detectors will be placed.

3.3 Design in SITRAFFIC

3.3.1 General

SITRAFFIC P2 (SIEMENS 2005) is an object-oriented tool for planning traffic signal controlled networks and intersections in fixed-time and traffic-actuated control systems (SIEMENS, 2005). It covers the development of signal programs (taking any evaluation of signal plans into account), time-distance diagrams and basic data for vehicle actuated control. This section discusses the use of SITRAFFIC P2 for the design of the network layout and the existing fixed time signal plans.

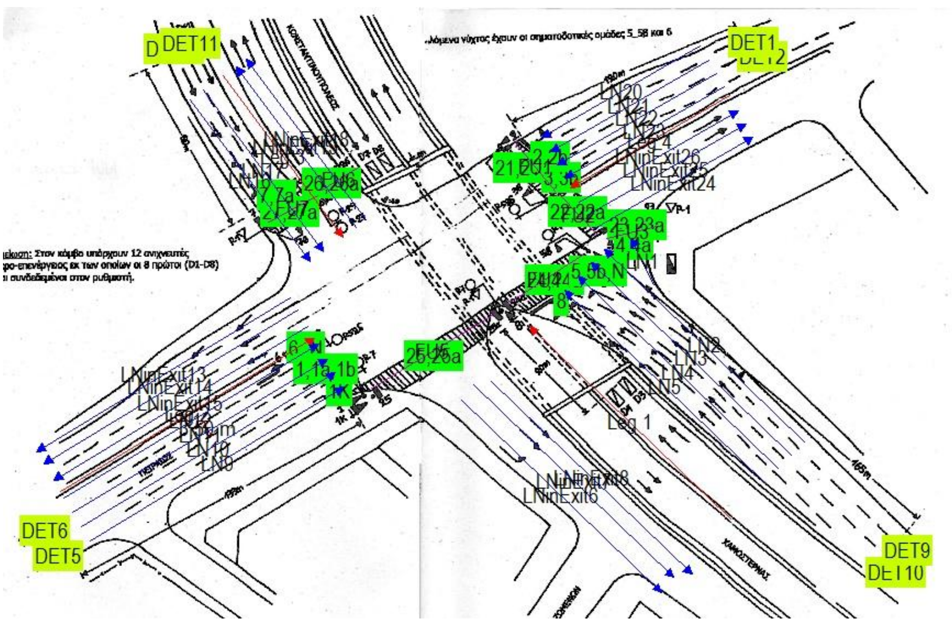
SITRAFFIC Control (SIEMENS 2003), is the software that allows the signal program information to be communicated to the controller. For the scope of this project, where the strategy is only going to be implemented only in VISSIM, communication of the signal programs to real controllers is not required. However, specific Control output files are required from VISSIM when the SIEMENS VA controller is selected. Moreover, Control will be used to export information concerning signal programs and detectors to Traffic Language.

3.3.2 Design process

The design process for the intersection where the strategy was evaluated is described in the following paragraphs. P2 provides a user interface that allows the engineer to design the intersection and the signal plans step by step. P2 can either estimate parameters (i.e. intergreen times based on the intersection layout and conflict points) or accept them manually from the user (if a signal plan is already made).

Topography

The design of an intersection in P2 starts with the definition of the intersection topography. A drawing of the intersection can be used (and scaled) as a background to assist in the design process. The layout plan visualizes the intersection arms, traffic lanes, signal groups, pedestrian crossings, detectors and conflict points, as shown in Figure 2.



Picture 2. Intersection layout in P2

The intersection arms are shown in the layout plan as red arrows in the middle of each approach. The possible turning movements from arm to arm also have to be defined. The traffic lanes are shown as blue arrows in the layout plan. Traffic lanes are distinguished between entry and exit lanes. Moreover, each lane is assigned to a specific intersection arm and signal group and has a user specified width. Signal groups are shown in the green rectangles. For each signal group, the type (car, pedestrian, public transit etc), the signal sequences (i.e. green-amber-red or amber flashing-red) and the minimum-maximum green times have to be defined. For detectors, the distance upstream from the stop line has to be defined. Finally, conflict points (yellow dots) are generated automatically, after the definition of reference lines (red lines in within the intersection area) which show the trajectories of vehicles in the intersection. The drawing of the intersection, is provided in greater resolution in Annex 1.

Signal group references

After having defined the intersection topography, the intergreen time calculations can be performed, to serve as a basis for the development of signal programs. On that purpose, a sequence of tables should be generated: Firstly at least one conflict matrix, showing conflicts between signal groups must be defined. More than one tables can be generated; however, one should be selected for each subsequent step. Secondly, intergreen time calculations can be performed by using a standardized methodology provided by the software. The methodology takes into account the entering and clearing speeds (which can be altered by the user) as well as the distances to each conflict point. The result of the calculation is an intergreen time matrix. If the user wishes, the intergreen time calculation part provided by the software can be skipped and intergreen times can be defined manually by the user. For this project, existing fixed-time signal plans were used therefore intergreen times were retrieved from the existing plans.

Traffic data

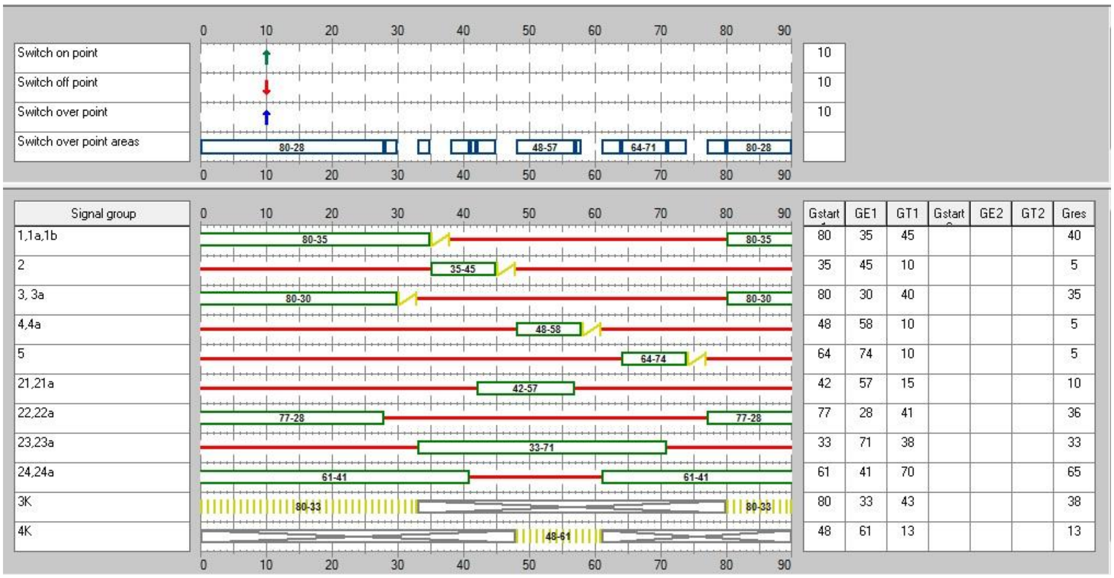
The insertion of traffic data can be performed, to allow for the development of new signal programs. However, the existing signal programs of the Athens Traffic Management Centre, for the four demand scenarios described in chapter 1 were used, therefore no need to insert traffic data was apparent.

Stage control

At the next step, signal stages have to be defined. Firstly, the signal groups have to be grouped in a number of stages. An intergreen time matrix should be assigned to the respective stage definition table. Secondly, the signal stage sequences can be defined. More than one stage sequences can be defined. Thirdly, the stage transitions should be defined, as not all signal groups start and end at the same second subsequently a transition from stage A to stage B might last for several seconds. For each stage transition, the exact ending second of each signal group in the ending stage and the exact starting second of each signal group in the starting stage should be defined.

Signal programs

As a final step, the signal programs are created. By using a specific intergreen time matrix and a specific stage sequence and stage transition plan, the starting and ending times of each signal group can be defined subject to constraints (stages, intergreen times). An example of a signal program is shown in Figure 3.



Picture 3. Example of signal plan creation in P2

Possible violations of intergreen times are automatically detected by the software and respective warnings are issued. Apart from the signal group starting and ending times, the switch on, switch off and switch over points for the signal program (points where the signal program can start, stop or change) can be defined.

VISSIM export

P2 offers a range of possibilities concerning the information to export to VISSIM. The user can export the whole intersection data (topography, signal group references, traffic data, stage control, signal programs), only the signal program, only the stream loads, or only the intergreen times (for supply to Traffic Language). In the present application, all intersection data were exported; however, significant additional work was required in VISSIM because the data initially exported need to be modified and calibrated to resemble the real network operation.

The development of the simulation, based on the data exported from P2 is described in the following chapter.

3.4 Simulation development

This section presents the process for the design of the network in VISSIM, from the export of the layout plans from P2, to a number of considerations for the design process and the assumptions made to achieve a realistic simulation.

3.4.1 Layout and signal plans

Despite the fact that the intersection layouts were automatically exported from P2, it was considered that the exported layouts were not satisfying. More specifically, each traffic lane is exported from P2 as a different link and not as a lane of the same link. In this way, the vehicles approaching an intersection would not be able to change lanes near the intersection and the vehicle behavior close to intersections will be quite restricted. Moreover, an increased number of connectors would be required, creating thus an increased number of conflict areas than those needed. To allow vehicles on different lanes to interact with each other (as is the case in reality), it was decided not to keep the exported layouts and redesign the intersections in VISSIM from the beginning. However, the signal plans as well as the signal heads were exported from P2 successfully. In order for the network to be tested before the development of the vehicle actuated strategy, the existing fixed time signal plan for the morning peak period is used.

3.4.2 Turning movements

Vehicles turning at intersections behave differently in each turning movement, according to the intersection layout. Usually, when turning flows are significant and the destination link has more lanes than the origin link, vehicles can turn from a single lane of the origin link to several lanes of the destination link. On that purpose, to simulate such turning movements in a realistic way, vehicles turning from one intersection arm to another should be allowed to enter in more than one lanes of the destination link; i.e. each origin lane has more than one connector to the destination link, on different lanes.

Despite the fact that the above consideration achieves a more realistic simulation, its complexity increases as: 1. More conflict areas are created and, 2. Additional routing decisions with appropriate proportions need to be set for each turning movement to distribute vehicles in the destination lanes.

3.4.3 Measurement locations

The detectors existing around the intersection were not sufficient for the operation of the new VA strategy. On that purpose, additional detectors had to be placed in VISSIM. More specifically, upstream and stop-line detectors are required in each intersection approach for the strategy to be operational. A new detector numbering was therefore introduced to distinguish between the detectors of the intersection.

Apart from adding new detectors, some existing detectors were also discarded (i.e. upstream from the south eastern approach) as they were not appropriately placed for the new VA strategy. The other existing detectors were placed further upstream (in locations where parked vehicles would not exist – see below) but also at a sufficient distance to enable the creation of a sufficient short-term horizon (for details see strategy description in chapter 4).

An on-site inspection as well as inspections from existing traffic control cameras were made to specify locations where parked vehicles exist. The inspections showed that most of the existing detectors were utilized correctly; apart from a few cases (detectors placed on the right lane) which were not utilized due to parked vehicles. This problem was further addressed in the way described in the following section.

3.4.4 Parked vehicles

The presence of parked vehicles at the right side of the road, in parts of the network poses an additional challenge as the available infrastructure is not fully utilized. This disadvantage was taken into account in the simulation. On that purpose, the lanes which are not utilized due to parked vehicles were closed for traffic (but not deleted) in the simulation. The lane closures are different in the various demand scenarios (i.e. a lane might be closed in the morning when stores are open but might be open during night-time).

In VISSIM, a lane closure works like a traffic rule: it means that a vehicle is not *permitted* to enter a lane, but it does not mean that a vehicle *is not able* to enter a lane. This means that a vehicle, if it has no other choice (i.e. the connector it moves on leads to a closed lane) it will enter the closed lane and try to leave it as soon as possible. In reality, the lanes which are partly closed by parked vehicles can be used by moving vehicles; however, their effective width is small therefore a vehicle moving along them is most likely to try to change lane as soon as possible. Therefore, lane closures as handled by VISSIM can be exploited to resemble the behavior observed in reality.

Finally, vehicles entering closed lanes are moving slower due to the reduced effective width of the lane. For that reason, *reduced speed areas* (see VISSIM help file) were defined for the lanes that are partly closed by parked vehicles.

3.4.5 Routes

In VISSIM, a vehicle, if not specified otherwise, will travel on a link and follow the first connector it will encounter. In this way, vehicle inputs, are not sufficient to simulate traffic operations on the network and routes need to be assigned to make sure that vehicles will be apportioned at intersections, according to the actual turning percentages. Moreover, routes are required for the correct functioning of merging and weaving lanes, as discussed below.

The turning proportions at some intersections change according to the time of day, depending on the demand profile (i.e. entry to the city centre, exit from the city centre). For that reason, different turning proportions were estimated from the manual traffic counts, per demand scenario. The turning proportions for each demand scenario and for each route are presented in Table 4.

Table 4. Turning proportions (%) at intersections per demand scenario

Routing decision	Route	Scenario 1	Scenario 2	Scenario 3	Scenario 4
7	1	56	constant		
	2	24			
	3,5	20			
8	1	22	constant		
	2,4	78			
9	1	48	52	54	49
	4	3	4	6	2
	3	49	44	40	49
24	1	38	37	34	38
	2	62	63	66	62
25	1	37	34	34	40
	2	21	19	19	20
	3	42	47	47	40

3.4.6 Conflict areas

To control conflicts between vehicles at intersections, VISSIM employs two different tools: “priority rules” and “conflict areas”. The latter is considered to be superior in terms of resulting behavior (PTV, 2008) and was used in this simulation. Conflict areas are defined as the common part of the intersection used by two conflicting movements. Usually, one movement is assigned with priority while the other has to yield.

At a signal controlled intersection, many movements, although conflicting, are protected by appropriate stage sequences and intergreen times. However, in some cases, conflicting movements are allowed to run simultaneously (i.e. in cases of low traffic volumes). In the network of this project, conflict areas for all possible conflicting movements at intersections have been defined, irrespectively whether the movements belong to separate stages or not.

The behavior of vehicles belonging to conflicting movements can be modified for each conflict area. More specifically, VISSIM allows the modification of a number of parameters that refer to the conflict area. These parameters are the “link visibility”, the “front gap”, the “rear gap” and the “safety distance factor”. The link visibility is the maximum distance, where an approaching vehicle can see vehicles on the other link. The Front Gap is used only for crossing conflicts and is the minimum gap in seconds between the rear end of a vehicle on the main road and the front end of a vehicle on the minor road, i.e. the time that a yielding vehicle waits before entering the conflict area after the vehicle with right of way has left it. Respectively, the Rear Gap is the minimum gap in seconds between the rear end of a vehicle on the minor road and the front end of a vehicle on the main road, i.e. the time that a yielding vehicle must provide after it has left the conflict area before a vehicle with right of way enters the conflict area. Finally, the Safety distance factor (used only for merging conflicts) is multiplied with the normal desired safety distance of a vehicle on the main road to determine the minimum headway that a vehicle from the minor road must provide at the moment when it is completely inside the merging conflict area (PTV, 2008).

These parameters were not modified for all conflict areas but only for those which are more likely to become active (i.e. conflicts due to the presence of traffic signals might not be avoided). The values for the parameters of the conflict areas were estimated by inspection from the traffic control cameras, for each intersection and conflict area individually. More specifically, manual counts of the front and rear gap were performed by using a stopwatch. The link visibility and the safety distance factor were not modified.

3.4.7 Driving behavior

The most essential modules of the VISSIM micro simulation environment are its car following and lane change models. Both the car following and lane change models in VISSIM use an extensive range of parameters. Some of these may be adapted to achieve a more realistic simulation, based on the particularities of the simulated network.

The traffic flow model in VISSIM is a discrete, stochastic, time step based, microscopic model with driver-vehicle-units as single entities. The model contains a psycho-physical car following model for longitudinal vehicle movement and a rule-based algorithm for lateral movements. (PTV, 2008). The model is based on the continued work of Wiedemann (Wiedemann, 1974, 1991). The basic idea of the Wiedemann model is the assumption that a driver can be in one of four driving modes, namely Free driving, Approaching, Following and Braking. For more information, refer to the VISSIM help file (PTV, 2008). The driver switches from one driving mode to another as soon as he reaches a certain threshold that can be expressed as a combination of speed difference and distance. For example, a small speed difference can only be realized in small distances, whereas large speed differences force approaching drivers to react much earlier.

Two versions of the Wiedemann model are available, called Wiedemann 74 and Wiedemann 99. The second version includes an increased number of parameters and is considered more suitable for the simulation of motorway links. For the scope of this project, the Wiedemann 74 model will be employed, which is suggested for the simulation of urban traffic (PTV, 2008). The parameters involved in this model are the "average standstill distance" (ax), which defines the average desired distance between stopped cars (and has a fixed variation of ± 1m), the additive part of desired safety distance (bx_add) and the multiplicative part of desired safety distance (bx_mult), which affect the computation of the safety distance. The distance d between two vehicles is computed using this formula:

$$d = ax + bx$$

where ax is the standstill distance,
$bx = (bx\_add + bx\_mult * z) * \sqrt{v}$,
v is the vehicle speed [m/s],
z is a value of range [0,1] which is normal distributed around 0.5 with a standard deviation of 0.15.

For this simulation, the average standstill distance was reduced to 1,5m from the default value of 2 (meaning that it ranges from 0,5-2,5m due to the deviation of ± 1m). The additive and multiplicative safety distances were not altered.

Concerning the lane change model, VISSIM distinguishes between two kinds of lane changes: Necessary lane change (in order to reach the next connector of a route) and Free lane change (because of more room / higher speed). In case of a necessary lane change, the driving behavior parameters contain the maximum acceptable deceleration for the vehicle and the trailing vehicle on the new lane, depending on the distance to the emergency stop position of the next connector of the route. In case of a free lane change, VISSIM checks for the desired safety distance of the trailing vehicle on the new lane. This safety distance depends on its speed and the speed of the vehicle that wants to change to that lane. There is currently no way for the user to change the "aggressiveness" for these lane changes. However, changing the parameters for the desired safety distance (which are used for the vehicle following behavior) will effect the free lane changes as well.

The default lane change behavior was modified for specific links in the network, as it was considered that, vehicles will attempt more aggressive lane changes in order to be able to proceed to their desired route unhindered. On that purpose, the maximum and desired deceleration values were increased for specific links in the network, where vehicles will attempt more “risky” lane changes.

Finally, the VISSIM parameter called “waiting time before diffusion” was also increased from 60 to 100sec, as otherwise VISSIM was removing vehicles from the network due to “excessive” waiting time, while in reality waiting times more than 60 seconds can occur.

3.4.8 Traffic compositions - Speeds

In VISSIM, different traffic compositions can be defined for each vehicle input location. The desired speed distributions are then defined per traffic composition. For this simulation, desired speed distributions as well as traffic compositions should also be defined per demand scenario. More specifically, two desired speed distributions are defined, one for daytime and one for night-time. Moreover, two traffic compositions are defined, as described below.

The majority of traffic in the intersection consists of passenger cars and motorcycles. An inherent disadvantage of the simulation with VISSIM is that motorcycles cannot be simulated; however, in the urban environment of Athens their percentage is significant while their behaviour in traffic differs substantially compared to passenger cars. Concerning buses, only one bus line uses the arterial, between the first and the second intersection, with a frequency of 2 buses/hour. Moreover, the arterial is very rarely used by HGVs. In conclusion, for the simulation, buses are going to be ignored as their percentage is negligible, while a very small amount of HGV’s will be considered only for day time. The percentage of cyclists is also negligible therefore it will also not be considered.

The speed distributions for daytime were defined by observing the speed data from the available detectors. It is observed that in the areas where parked vehicles and pedestrians are present, commercial land uses exist and the capacity of the road is not fully utilised, speeds range from 40-50 km/h. For night-time, speeds are usually higher; However, due to truncation of the speed data from the detectors it was not possible to have a respective indication. It can reasonably be assumed that speeds up to 90km/h would be observed, while a lower bound for the desired speeds can be reasonably set to 50km/h.

As a result, the speed distribution for daytime is an S-shaped cumulative distribution with a minimum of 40 and a maximum of 70km/h while the overall speed distribution for night-time is an S-shaped cumulative distribution with a minimum of 50 and a maximum of 90km/h. For HGV's, the minimum desired speed is set to 30 and the maximum is set to 60km/h.

3.4.9 Pedestrian crossings

Simulation of pedestrians can be performed in VISSIM by an additional module, not available in the standard version. This module is not available in the version of VISSIM used in this project. However, the signal groups concerning pedestrians are included in the simulation together with the respective signal heads. These are placed on separate links placed vertically to the main links, resembling pedestrian crossings.

The splits for pedestrian signal groups in the vehicle actuated strategy will be constant and will not be considered from the signal optimisation module due to the lack of real-time pedestrian flows and due to lack of real-time calls for pedestrian phases.

4. Strategy development

The proposed approach is based on contemporary methods towards vehicle actuated signal control. The traffic signal control problem is formulated as a look-ahead search. The strategy is inspired by a number of existing systems using look-ahead methods (for more information see chapter 2) and attempts to introduce methods that have not been widely exploited but are promising for yielding good results (i.e. the signal optimization technique). Moreover, in this strategy some techniques are specifically developed to operate with the other strategy modules (i.e. the traffic state estimation).

A main advantage of the proposed strategy is its high degree of flexibility, in respect to the achieved accuracy and speed of the algorithm. The length of the time horizon and the resolution of the optimum search (number and length of time steps) can easily be adjusted to meet the computational capabilities of the available hardware. Moreover, depending on the complexity of the intersection that needs to be controlled (i.e. number of signal groups and stages), the efficiency of the strategy can be adjusted depending on the real-time requirements of the control problem. For example, for a complicated intersection with 5 stages, the stage sequence can be fixed, while in a simpler 3-stage intersection the stage sequence can be variable.

This chapter analyses the architecture of the new strategy and provides a technical description of its distinct modules. Moreover, the programming code developed in Traffic Language is attached as Annex 4, together with a detailed description of each module and each step within each module (Annex 3) providing all technical details, to assist the reader to understand the code.

More specifically, this chapter consists of the following:

In section 4.1, a description of the strategy framework is provided. The rationale behind the selection of the methods and a description of each part is given along with the originalities of the proposed approach. The interconnection between the different modules of the strategy is presented and some advantages over previous systems are illustrated.

In section 4.2, additional technical details concerning the strategy modules are provided. The previous section was concerned with a more general description of the strategy therefore did not provide sufficient details on the way the modules operate; thus, this is the aim of section 4.2. This section also discusses some strengths and weaknesses of the strategy arising from the way the programming code was developed.

In section 4.3, the effort for developing the algorithm in SITRAFFIC Traffic Language (TL) is described. Some obstacles and the ways they were overcome are discussed. The strategy framework is visualised through a flowchart and finally, lessons learned from the development effort are discussed.

4.1 General framework

4.1.1 Adoption of the rolling horizon approach

The strategy employs a control logic which is based on a **rolling horizon**. The rolling horizon approach for signal control problems is described in chapter 2. Based on traffic estimates for the current and future state of the system, an optimal signal plan (or policy) is sought for the duration of the rolling horizon. Only a first part of the plan is implemented before the horizon is rolled forward and the optimal signal plan again sought, with the benefit of the latest detector data. The rolling horizon must therefore be long enough to allow the first part of the optimal plan to be determined uniquely. In order not to favour signal plans that are optimal over the rolling horizon but yield high costs thereafter, the state of the system at the end of the rolling horizon is assigned a value through a terminal cost function (Bell, 1992). The duration of the horizon can vary, from very short (i.e. 10-20 seconds) to quite long (i.e. 180 seconds) time. The importance and the implications of the horizon duration are discussed in the following paragraphs.

The most important advantage of this approach compared to traditional strategies (which optimise signal timings for the next cycle, i.e. SCOOT, SCATS) is that the strategy updates its traffic estimates and switching decisions on a more frequent time basis and therefore adapts easier and faster to the prevailing traffic conditions. At the same time, it avoids making myopic decisions as it optimises for the whole horizon and not only for the near future. Moreover, the possibility to adjust the horizon duration, the length of the time step and the admissible switching points allows for **maximum flexibility** in the strategy accuracy and adaptability, depending on the available computational power, on the accuracy required and on the selected objectives.

The formulation of the signal control problem as a look-ahead search, as illustrated by Shelby (2004), is a good example of how the problem is also tackled by this strategy.

4.1.2 Parameterisation

The selection of the **horizon duration** is interdependent with other factors; namely, the **length of the time step**, the **length of the minimum admissible switching interval** and the **computational effort** needed to solve the problem. All these factors should be defined simultaneously; however, a number of considerations separately for each, should be taken into account.

Horizon duration

The duration of the horizon should be long enough to allow envisioning the traffic situation over a sufficiently long period. A short duration of the horizon would produce signal timings that optimise the measure of operational performance for the near future but might yield high terminal costs afterwards. Moreover, a short horizon is not able to take into account the evolution of traffic within a whole cycle therefore cannot lead to fully informed decisions. A longer horizon, due to the re-optimisation every time step, reduces the effect of the period after the end of the planning horizon. However, the terminal cost of the signal decisions at the end of the horizon will be taken into account by adding a term including the weighted effect of the terminal queues.

If a short horizon is selected, the decisions taken may not be able to affect all intersection queues, as one queue might be served in a stage which can become active only after the end of the horizon. For example, if a short horizon of 20 seconds is selected, a queue served by the stage that has just passed might not be able to be taken into account in the optimisation as the minimum time for this stage to become active again is more than 20 seconds thus it cannot be reached within the next horizon. In this way, the optimisation will only take into account the evolution of the performance measure based on the rest of the queues, while the evolution of the specific queue will be irrelevant. To avoid this from happening, the horizon should be as long as a whole cycle (and ideally equal to the maximum admissible cycle). However, sometimes this might not be feasible due to the computational effort needed for the optimisation, when the maximum admissible cycle becomes too long.

This problem becomes less pronounced if a non-fixed stage sequence is adopted, because the algorithm has the flexibility to serve the most important queues first. However, this comes at the cost of increasing the complexity and thus the computational burden of the algorithm. The latter is further discussed later on.

Time step – switching step

A key point for the strategy is to define the time instant where the signal state is allowed to change as well as the time instant where the new optimisation procedure starts. In general, the horizon H is divided into n discrete time steps of duration t. The **time step is the interval between consecutive signal optimisations**. However, the **signal states can be allowed to switch also within a time interval** (i.e. every t/2, or even every second). This allows for a more complex decision tree to be developed, where signal switches are allowed more frequently, but the signal optimisation still takes place every t seconds. In this way the signal control decisions can be better adopted to the traffic dynamics. On the other hand the switching step can also be longer than the time step (meaning that each time the optimisation runs, the previous decision has not been completed and is re-evaluated). Finally, it should be mentioned that a short switching interval will allow better adaptation of the algorithm to the traffic conditions but will also result at a more computational costly problem. The relation between the time step and the switching step, as well as the implications of their lengths are better illustrated in the following paragraphs.

The length of the time step eventually defines the frequency of the optimisation procedure, therefore it should be **superior** to the maximum time required to solve the optimisation problem. An important difference exists between implementing a signal control strategy in VISSIM and in reality. In contrast to reality, in VISSIM, the simulation time pauses while an algorithm is executed. This means that, speaking in simulation time, the algorithms are executed at an instant. This has the crucial implication that the time needed in reality for the algorithm to run does not need to be taken into account within the algorithm itself. In reality, the optimisation should be performed in less time than the duration of the time step, therefore the time step should be chosen carefully to fulfill this criterion under any traffic conditions (where the algorithm might take more time to run). In VISSIM, due to the time pause, the time step can be chosen to be as small as possible to maximise the adaptability of the strategy.

In the current project, all the above are taken into account, together with the computational capabilities of the available hardware (a Pentium Core 2 Duo 2,2GHz, 3GB RAM), to define the optimal set of the aforementioned parameters. A time horizon of **80 seconds** was selected for the final version of the strategy, together with a **time step of 1 second** and a **switching step of 5 seconds**. In a real implementation, where the time needed for the optimisation would be taken into account, the time step might have been selected to be longer than the switching step (or equal). However, in this case, where the implementation is made in VISSIM, the lowest possible time step is selected to optimise the strategy performance. The selection of the length of the switching step is made together with the selection of the horizon duration. Long horizons will need longer time steps, for the optimisation to remain feasible, while very long time steps will result in decreased accuracy for the algorithm. Therefore the best compromise has to be sought, by a trial-and-error process.

The selection of the switching step duration, results in dividing the horizon in a **maximum of 16 steps.** This set of parameters means that every second, the strategy defines the signal states for the next 80 seconds, divided in 5 second intervals. However, only the first second of the first switching step is implemented and then the horizon is rolled forward and the optimisation takes place again.

In fact, several combinations of the above parameters were tested before choosing this final set. Strategies with time horizons ranging from 40-90 seconds, and switching steps of 4 or 5 seconds were tested. An overview of the strategies that were developed, using different parameter sets, is provided in table 5. Although a strategy with an horizon longer than 90 seconds would be desirable (as the maximum admissible cycle is 160 seconds), its implementation (with the existing hardware) wouldn't be possible with a switching step shorter than 6 seconds as the optimisation problem would be too complex to be solved within a reasonable time frame. It would be more desirable to adopt a shorter switching step to allow for more flexibility in the strategy decisions. On the other hand, adopting a step as short as 3 seconds and keeping the implementation feasible would result in a horizon of maximum 50-55 seconds.

The above outcomes can be more easily understood if the complexity of the decision tree, resulting from each parameter combination, is considered. The aspect of the decision tree is discussed in the following section (optimisation); however, to provide a basic understanding of the complexity, it is mentioned that for each extra time step introduced, the complexity of the algorithm rises by a power of 2 (from 2^{x} to 2^{x+1}).

Table 5. Versions of the strategy tested through the development process

Version	Horizon	Step length	Steps (max.)
1	120	4	30
2	80	4	20
3	40	4	10
4	75	5	15
5	80	5	16
6	90	5	18

The first version of the strategy was being developed with a 120 second horizon and a 4-second time step. However, due to an inability of the programming software (TL) to cope with a 30-times-repeated loop (which resulted in the program crashing) this attempt was soon abandoned. Later on, it would be realised that such a version would result in an intractable algorithm which anyway would have to be simplified. This would be realised with the second version of the strategy, which adopted a 80-second horizon, divided in 20 steps of 4 seconds. This version was able to run but each optimisation required several minutes therefore was considered practically intractable. The next version, including a 40-second horizon and a time step of 4 seconds (maximum 10 steps), was developed after the first two failed attempts, to check whether a strategy with a significantly shorter horizon would run. In the following attempts, it was reasoned that a slightly longer switching step (5 seconds) would provide the opportunity to adopt a longer horizon, without compromising a lot on the algorithmic accuracy. As a result, three strategies with a 75, 80 and 90 second horizons were developed. The first two were running quickly (in less than 1 and 1,5 second respectively) while the third required a few seconds for each run. For that reason, the version with the 80 second horizon and 5 second switching step was selected for the implementation.

The selected combination of parameters resulted in an optimisation problem which is being solved in approximately 1 second (thus is applicable to a real controller) while preserves a horizon and a time step which are not too short. The horizon duration is long enough to allow the observation of the evolution of queues and waiting times over the intersection for a sufficient time period (a sequence of all stages might well run within this period if traffic conditions are not too heavy). Finally, the combination of parameters means that, up to 16 signal switches are *theoretically* possible within the horizon (however, the *actual* switches are usually less, due to intergreen times greater than 5 seconds). The modification of the selected parameters can be performed without too much effort. Therefore, the modification of the strategy in order to prepare a more powerful version that can be used on a stronger computer remains possible.

4.1.3 Optimisation

The initialisation of the optimisation will be made from the stage that is currently running (or from a pre-defined switch-on picture if the signal program just starts). If a stage is currently running, the signal state needs to be continuously monitored so as to know how much time has already been allocated to the specific stage. This is an important requirement in order to terminate a stage when it reaches its maximum green, even if the optimisation algorithm suggests otherwise.

The **measure of effectiveness** (total waiting time in vehicle-seconds) was selected so as to **serve the following purposes**: As will be explained later, under certain demand conditions, the strategy is likely to lead to the maximum green times. Moreover, the total waiting time is a performance measure which can be more easily perceived by the road user (requirement discussed in chapter 2). Thirdly, as will be presented later, the formulation of the strategy adopting this performance measure takes into account critical queue lengths, which was also a primary requirement (see chapter 2).

The formulation of the signal control problem as look-ahead search has certain implications on the way the optimum has to be sought. More specifically, the definition of an admissible switching interval means that during the optimisation, the cost will be quantified and accumulated at all admissible switching points in the path so as to decide whether to terminate or continue serving the current stage.

At this point a critical decision has to be made by the traffic engineer and this is whether the stage sequence will be fixed, or allowed to vary depending on the demand. The advantages and disadvantages of each approach are discussed in section 2.6.1 of chapter 2. In summary, although stage sequence flexibility has the potential to result to more adaptive control as well as to make shorter-horizon strategies more effective, a fixed stage sequence will be maintained to reduce the computational burden of the algorithm (this is well illustrated in the following figure). Moreover, in Greek practice, it is preferred to keep the stage sequence fixed for safety purposes.

The strategy though has much **potential for improvement** (in efficiency terms), if implemented in a stronger computer. This can result from its ability to have a fixed or non fixed stage sequence by simply modifying a few commands in TL. However, for the reasons mentioned before this limited version could be implemented and tested.

By keeping the stage sequence fixed, the decision to switch from a signal state to the next results in a binary decision tree, starting at time t_i and expanding until the end of the time horizon $t_i + H$. The optimisation problem then, is to find the optimal sequence of switching decisions (path) from the beginning to the end of the horizon. A graphical representation of two possible binary trees is provided in Figures 4 and 5. In both cases the optimisation is made every time step t. However, in the first case, the time step is longer than the admissible switching interval while in the second case the opposite applies. In the first case, up to two switches can be made within one time step while in the second case, only the first second of the first decision is actually applied before the optimisation takes place again. The first case would be applied to a real controller with less computational capabilities and where the time needed for the optimisation should be taken into account, while in VISSIM (where, in simulation time, the optimisation is made instantly), the second case can be applied. Note that, irrespectively of the length of the switching interval, the time between two optimisations can vary due to the variable length of the stage transitions, because for practical reasons, there is no need to perform an optimisation while a transition runs, as this part of the signal program cannot be altered anyway. Therefore, when a transition starts, the length of the time step becomes equal to the length of the stage transition.

The first figure shows the decision tree formulated by a time step equal to two switching steps. The first two switching decisions are actually implemented before the next optimisation. In this example, all stage transitions are considered as having a duration a little longer than the switching step.

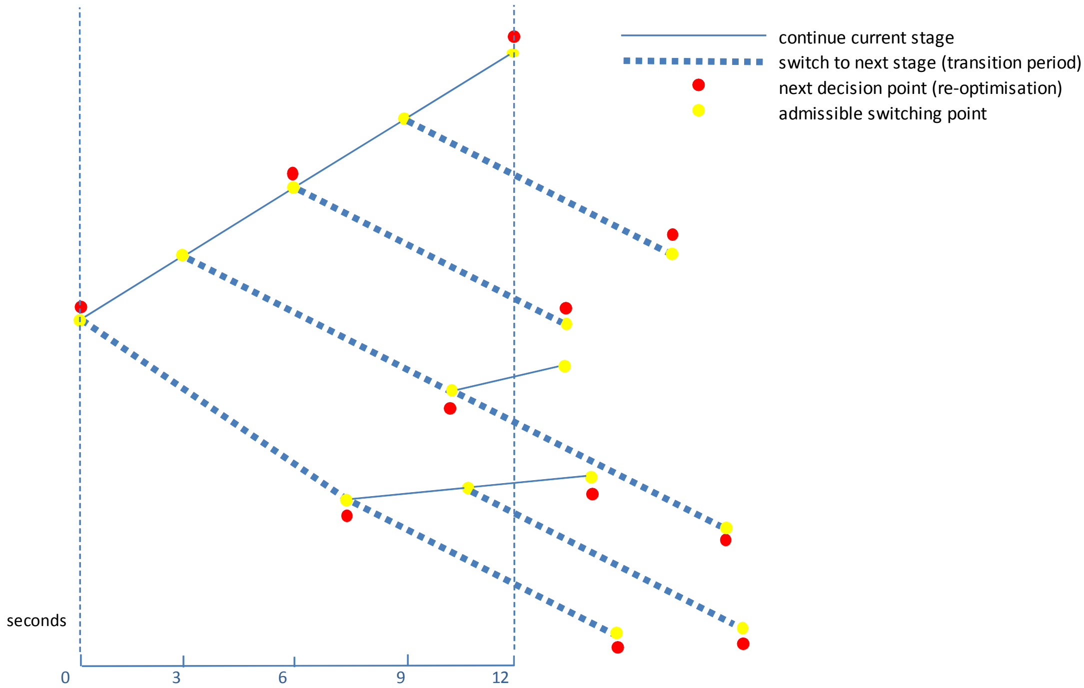


Figure 4. Sample 1 decision tree

The dashed vertical lines at second 0 and 12 show the beginning and the end of the horizon. It can be observed that all paths extending after the end of the horizon are truncated at the 12th second. The need and the implications of this are discussed in chapter 3.

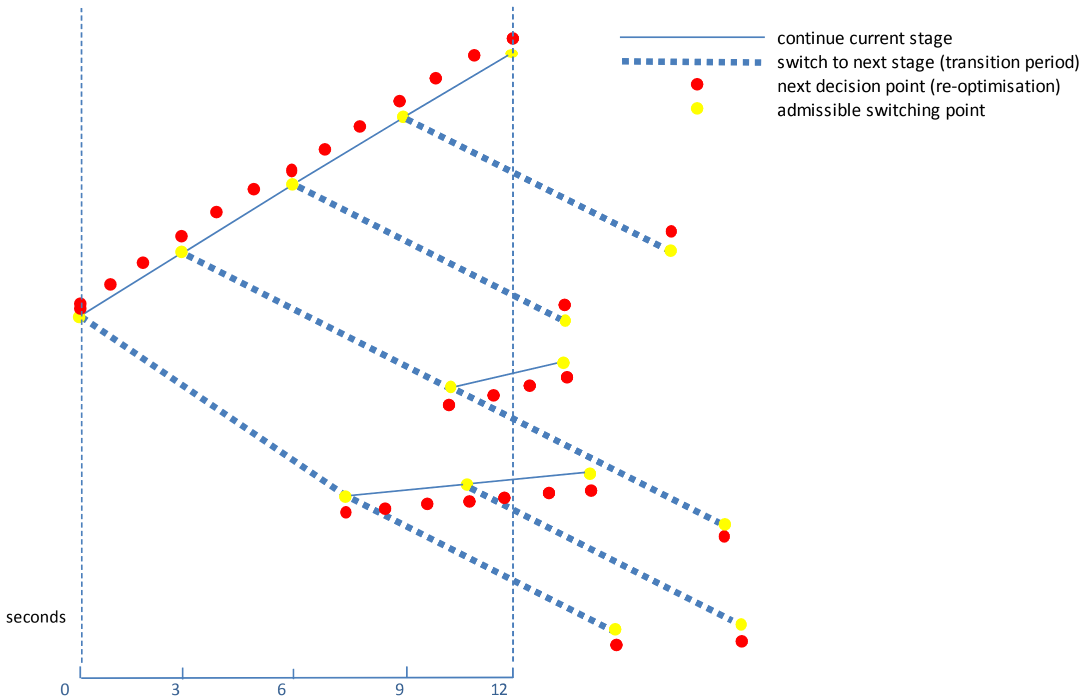


Figure 5 Sample 2 decision tree

The second figure shows the decision tree formulated by a time step of one second, smaller than the switching step (3 seconds). It can be observed that even the first step of the path is not completely implemented before the next optimisation. Note that the figure shows the tree expanding only from the first red dot. Actually, the same tree would expand from each subsequent red dot, replacing the previous one.

The examples refer to a decision tree, for a small horizon of 12 seconds. The admissible switching points within this horizon are indicated by the yellow dots. In the first red dot at the beginning of the tree, the optimisation problem is to find the optimal path (minimum *cumulative* cost per time step) that leads to the end of the horizon (in this case 12 seconds later). In both examples, it can be observed, that from the first red dot, three possibilities exist (either continue stage for the whole time step, or continue stage for the half time step and then switch following the transition period, or switch immediately). The path selection is made for all subsequent switching points (yellow dots) until reaching the end of the horizon. In this way a path is defined and then it is evaluated in its full length. The algorithm examines all paths of the decision tree that do not violate the pre-specified constraints and selects the one that yields the optimal performance measure. After the path selection is made, the optimal path is implemented until the next red dot and the optimisation is re-initiated for the next 12 seconds.

The interval between two subsequent optimisations is the time period between two red dots. It can be observed that, this duration can also be different from the time step, if the stage transition and the minimum green have variable durations. The interval between two optimisations is equal to the time step only if the optimal path includes continuing the current stage.

The optimal switching sequence is found based on the minimisation of a cost measure for the whole time horizon as well as a terminal cost at the end of each path. The structure of the decision tree, requires a cost measure that can be quantified **at each step** and **accumulated** while proceeding to the end of the horizon. For that purpose, traditional cost estimation methods, such as using the Webster (Webster, 1958) or the Kimber and Hollis (Kimber and Hollis, 1979) delay estimation formulas cannot be exploited, due to the absence of a pre-defined cycle. Therefore, a new measure suitable for use with the current optimisation method was developed.

The chosen measure of effectiveness to be minimised is the **total waiting time** (in vehicle-seconds) for all vehicles over the intersection, accumulated per time step over the control horizon. To define this term there is a need to specify at which time instant a vehicle is considered to start “waiting”. A vehicle starts waiting when it joins the back of a queue and stops waiting when it crosses the intersection stop-line and stops occupying the stop-line detector (therefore a vehicle moving within the queue still “waits”). Vehicles are projected to the back of the queues at a time t which is estimated using an estimated speed and the current distance from the detector to the back of the queue (the estimation of these measures is described later).

At each admissible switching step, the waiting time will also be accumulated for all traffic movements (streams) of the intersection to provide a universal measure. The cost over the intersection for each time step will be estimated as follows:

$$pc_i = LB_{i-1} + c_i \qquad (1)$$

where:

$$c_i = \sum_{j=1}^{n} (Q_j * ts + (a_j - d_j) * \frac{ts}{2}) \qquad (2)$$

and:

pc_i,	(*path cost*) is the total waiting time (from the beginning of the horizon) over the intersection at time i (i is any admissible switching step within the horizon)
c_i	is the *cost* incurred by waiting vehicles over the intersection within one switching step
j,	is the signal group (n in total)
LB,	is a variable that accumulates the waiting time from previous time steps, to take into account vehicles that were already waiting before the current step (LB stands for: lower bound – see branch and bound method)
Q,	is the existing queue at the start of the step
ts,	is the length of the admissible switching step
a and d,	are the arrivals and departures for this specific signal group during the step

The first part of the summation in equation 2, $(Q * ts)$ accounts for the waiting time incurred by the existing queues during the latest step. For that purpose, the length of the existing queues for all signal groups is multiplied by the duration of the step. The second part of the formula, $((a_j - d_j) * \frac{ts}{2})$ takes into account the waiting time of the vehicles that arrived or departed during the latest step. The waiting time for the arrivals and departures is estimated by considering the average waiting time per vehicle (the half of the time step). This assumes that the arrivals and departures were uniform within the time step. While this is generally not true, due to the fact that the step is a few seconds and that the waiting time of vehicles is accumulated over the next time steps, the error resulting from this assumption will be very small.

The departures will be equal to zero when the signal group is red and equal to the departure rate when the signal group is green (as long as vehicles are still present in the queue). The way the arrivals will be estimated is described in the following paragraphs by presenting the traffic estimation module developed specifically for this project.

The above formulae consist of several parts. The target is to obtain a measure of the total waiting time at the end of the planning horizon (LB_H), which also takes into account the greater weight of waiting for a long time compared to the less weight of waiting for a short time. Equation 1 includes the lower bound variable which serves for that purpose and is the cost term that is inherited from a previous switching step to the following. Therefore, the LB variable will be updated at each step as follows:

$$LB_i = LB_{i-1} + pc_i \qquad (3)$$

Therefore, to estimate the new cost pc_{i+1}, the new LB_i will be taken into account in equation 1. The Lower Bound at the end of the horizon will be the eventual performance measure. The LB is an exponentially rising function.

From equations 1, 2 and 3 it can be derived that the Lower Bound of a path consisting of n switching steps, at the nth step can be calculated as follows:

$$LB_n = 2^n * LB_0 + 2^{n-1} * c_1 + \cdots + 2^1 * c_{n-1} + c_n \qquad (4)$$

As mentioned before, the optimal path from the decision tree will also be evaluated in terms of the cost it yields at the end of the optimisation horizon. For that purpose, the following terminal cost formula is adopted:

$$TC = \sum_{j=1}^{n} \left[\frac{Q_{term,j}^2}{2q_j} \left[U_j + r_g\left(1 - U_j\right)\right] + Q_{term,j} * R_{min,j} * U_j \right] \qquad (5)$$

where:

TC	is the terminal cost of the path
j,	is the signal group (n in total)
$Q_{term,j}$	is the terminal queue of signal group j
q_j	is the saturation flow of signal group j
U_j	accounts for the signal display of signal group j at the end of the horizon (0: green, 1: red)
r_g	is a user specified parameter that defines the relative importance of stopped vehicles on a traffic movement with a green signal indication in relation to stopped vehicles on an approach with a red signal indication.
$R_{min,j}$	is the minimum red time remaining for signal group j (if it is red). This is estimated as the minimum red for the signal group within a cycle, minus the red duration that has already elapsed.

The terminal cost function used in this study is similar to the expression used in the SPPORT model (Dion and Hellinga, 2002). The function is weighing the queues in each traffic movement at the end of the decision horizon by taking into account the respective saturation flows and the signal indication in the respective signal group. This way, not only the absolute queue is considered but also how quickly and how easy this will be dissipated.

4.1.4 Traffic estimation

To calculate the cost by equations 1-5, the queue for each traffic movement needs to be estimated and the arrivals on the back of each queue at each switching step during the time horizon need to be predicted. For that purpose, a traffic estimation module was developed to satisfy the above requirements. The traffic estimation module is described in detail in the following chapter. In general, it divides the planning horizon in two periods, namely a **short-term** and a **long-term**. The arrivals for the short-term period are estimated by using data from the upstream detectors on the same link with the methodology described later. Therefore, there is no need to use data from detectors upstream from the previous intersections, as this would result in a significantly more complex problem (for more information, see the PREDICT algorithm, presented in chapter 2).

Two main requirements for the traffic estimation method were speed and simplicity (with the minimum impact on accuracy). The module needs to run every second, thus one of the existing approaches could lack on speed. Simplicity, with the minimum impact on the prediction accuracy is ensured by adopting the short-term and long-term prediction horizons.

The arrivals for the long-term part of the horizon, are based on an average of the arrivals during a pre-defined period. An average of the arrivals during a past period is a robust measure which however cannot provide accurate details on the arrival process (i.e. capture specific vehicle arrivals at certain moments) but can provide a good uniform arrival rate. In any case, the signal decisions for the long-term period might not be applied eventually as they will be subject to re-optimisation in the next time step, using the more accurate short-term predictions.

Even though the long-term horizon is based on an average number of arrivals, the respective estimates are updated quite frequently (every minute). Therefore, the consequences for the adaptability of the strategy in sudden incidents will not be severe. In any case, the short term horizon is capable of capturing the sudden changes in the traffic flows and will allow the strategy to partly react to an incident even before the first minute passes.

4.1.5 Optimum search technique

Previous strategies have also used the rolling horizon concept but using different optimisation techniques such as forward dynamic programming or explicit enumeration (for a more detailed description see chapter 2). This strategy will employ an efficient method which saves significant computational effort when seeking for the optimal solution therefore can allow for the adoption of a short admissible switching step and optimisation step which eventually results in a strategy that adopts better to the traffic conditions. The method used to find the optimal path is the implicit enumeration of all possible paths (otherwise called Branch and Bound technique). The Branch and Bound technique as well as its advantages are described in chapter 2. Its adoption for the specific strategy is further discussed in the following section (4.2) and in Annexes 3 and 4.

4.1.6 Signal coordination

Existing approaches

The first rolling horizon strategies, such as DYPIC (Robertson and Bretherton, 1974) and OPAC (Gartner, 1983) have been developed mainly to operate at a single intersection. More recent approaches in decentralised signal control such as UTOPIA (Mauro and Di Taranto, 1989) and later RHODES (Mirchandani and Head, 2001) first addressed the network control problem. In particular, UTOPIA or RHODES have used a superior decision layer that first predicts the movement of platoons along the network and performs a first optimisation of signal timings based on the macroscopic predictions. The results from this layer are eventually distributed to the intersections and act as constraints to the intersection level optimisation modules. RHODES has also adopted the rolling horizon approach for the upper (macroscopic) optimisation layer but with a significantly longer time horizon compared to the one used for the intersection (microscopic) optimisation.

Current framework

In this thesis, the approach taken by UTOPIA or RHODES is **not** adopted. The reason was that coordination should not be viewed as a prerequisite but as a rational choice taken by the intersection controllers, when needed (when it is more optimal for the network). In any case, the fully decentralised nature of the strategy, combined with the rolling horizon approach, does not allow for constraints to be imposed by a superior layer on the optimisation procedure.

Nevertheless, this strategy suggests a framework that can achieve coordination by adopting a simpler approach. This is by exploiting the properties of the traffic estimation and the optimisation module. More specifically, it was concluded that the properties of a pro-active queue evolution module such as the one that is developed for this strategy can be exploited for the establishment of a flexible coordination scheme without using predefined rules or imposing other constraints to the optimisation.

The basic assumption of the approach is, that a sufficiently good projection of queue data in the future horizon helps the intersection controller to make the best decisions, without having to rely on external rules. An advantage of this approach is that it might happen that an intersection that is regarded to be less critical, to have a sudden big flow on another direction than the critical. In this case, perhaps the non-critical intersection should not obey to a fixed coordination rule for the critical direction and let those (more) vehicles wait. It was decided that a downstream intersection can make its own decision: the platoon from an upstream intersection is let to drive on and the subsequent intersection will be able to realise it in time (see the TE module in next chapter) and make the optimal decision based on its optimisation procedure. If the cost imposed by stopping the platoon is larger than the cost of the other choices, the intersection will establish the required coordination.

It should be noted that, the further the distance of the upstream detectors from the stop-line is, the longer the short-time prediction horizon can be. This means that the information on the expected platoons will be provided for a longer future period therefore the intersection controller will make a more informed decision. For that purpose, the detectors should be placed as far upstream on each intersection approach as possible (without compromising the accuracy of the projected data).

Remarks on study design

The signal control strategy developed in this thesis is evaluated at a single intersection (see chapter 5). However, the developed framework intends to be applicable to an arterial or a network of intersections, by taking into account signal coordination as discussed above. Despite the fact that the thesis initial planning included a study area for the evaluation comprising of four consecutive intersections, time and software limitations did not allow for the tailoring of the algorithm to more than one intersection.

The application of the strategy to more than one intersection eventually reveals whether the theoretical concept of coordination presented above is efficient and how it performs compared to the fixed time coordination schemes. Moreover, it would reveal whether the assumption that coordination should not be considered as a prerequisite but as a rational decision (taken when needed), brings improvements to the network operation. Finally it would provide insight on how the traffic estimation would react to a non-random arrival process (as vehicles would be bundled into platoons) and how efficiently this would be depicted in the queue estimates. However, a definite answer on those subjects can only be provided by further research.

An important difference of the proposed approach compared to existing VA approaches with coordination schemes, is that no negotiation between intersections takes place. The traffic estimation (see also following section 4.2.1) is made by not taking into account additional information from upstream controllers concerning arrivals, queue dissipations etc. This isolation of the local optimisation comes with some advantages and disadvantages. On the one hand, information from upstream intersections enhances the queue evolution prediction while on the other it rises the complexity of the prediction problem and also establishes a relationship between local optimisations which makes the system more vulnerable to hardware failures (a local controller failure affects the neighbouring controllers).

4.1.7 Additional practical issues

A number of additional practical issues which concern the application of the above methods is mentioned in this section.

The rolling horizon approach aims at updating the signal timings every second. Despite the fact that the strategy is acyclic and has no fixed stage durations, the SIEMENS C800 controller needs a fixed time plan as a basis to act on. Therefore a plan including only the stage transitions is provided to the controller and the stage durations are then provided by the strategy as extensions to this fixed time plan. A maximum duration for each stage is also provided in order to determine a maximum admissible cycle.

The strategy outputs are in the form of vectors (one for each stage), the dimensions of which are equal to the seconds of the horizon and provide the state of the stage in each future second (0: off or 1: on). The summation of all the elements of each vector provide the duration of the respective stage during the next horizon. An exception is the stage currently running, for which, only the first 40 vector elements are summed. This is done, because in the occasion that a stage occurs twice in the same horizon, only its first duration is of interest for the current optimisation (note that the number 40 is selected carefully as it serves two purposes: firstly the same stage cannot occur twice within 40 seconds, and secondly the maximum green time for all stages is never greater than 40 seconds). Every second, TL produces a new set of output vectors and provides them to the controller, which implements the first second, until receiving the next ones (next second).

In this strategy it was reasoned that, the minimum green durations for all signal groups can preferably be taken into account when building the basic signal program consisting of the stage transitions. Therefore, the stage transitions might be slightly longer than the ones used in a fixed time program; however, in this way, when a stage starts, the algorithm does not need to stop and wait until the minimum green has elapsed as the minimum greens are ensured even by the transition alone. The algorithm can then choose to terminate or extend the current stage instantly when it starts.

4.2 Strategy modules

4.2.1 Traffic estimation (TE) module

General

The aim of the traffic estimation module is to provide data concerning the future arrivals in each queue of the intersection so as to enable the estimation of the queue evolution over the time horizon. The data have to be in a form which is usable by the optimisation module. The optimisation module needs to quantify the cumulative cost incurred by the decision at the end of each switching step, until reaching the end of the horizon, so as to obtain the optimal path from the decision tree using the branch and bound method. As mentioned in the previous chapter, the performance measure to express the cost is the total waiting time **per time step**, accumulated per step until the end of the horizon. To achieve this, it needs reliable input concerning the arrivals. The way these data are going to be obtained is described in this section.

The traffic estimation (TE) module was developed inspired by more complicated methods used in sophisticated strategies like RHODES. However, it provides successful estimates concerning the arrivals and the queue evolution at an intersection by using **less input and in a simpler way**. The compromise for this is that it can only provide very accurate estimates for a part of the horizon, while for the rest of the horizon the optimisation has to rely on average estimates.

Module description

The TE module divides the optimisation horizon in two periods, namely a short and a long term. The short-term horizon has a duration ranging from 1-60 seconds depending on the location of the detector upstream from the stop line on the respective link and on the current queue lengths (low values correspond to the case where the queues are approaching the upstream detectors). The long term period starts from the end of the short-term period and ends at the end of the optimisation horizon. Traffic estimates for these two periods are different as the first is based on a projection of future vehicle arrivals on the back of the queue while the second is based on an average flow estimate during the last minute. The difference in the use and eventually in the importance of the short and long term period was explained in chapter 2.

The length of the short term horizon also depends on the speed of the approaching vehicles and the traffic intensity. In low speeds cars need more time to arrive in the back of the queue therefore the length of the short term horizon increases. Moreover, if there is a time gap between two successive actuations of the upstream detector, this is also part of the short term horizon as it is certain (until the next actuation) that no vehicle is present between the previous car and the detectors. However, the maximum length of the short term horizon was set to 60 seconds (this of course depends on the distance between the upstream detectors and the stop-line). It can be understood that the length of the short term horizon can change and that this flexibility allows the TE module to be adoptive to the current traffic conditions.

The cost estimation method essentially requires to predict the evolution of all queues in each intersection approach. Each approach might consist of multiple distinct queues (i.e. vehicles waiting to go ahead and vehicles waiting to turn left on a separate lane). Moreover, vehicles waiting at the same physical queue might want to follow different directions. For that purpose, the evolution of the queue per signal group and not per approach or lane, has to be simulated separately. To predict the queue evolution, it can be observed from the cost estimation formulas (1, 2), that the term that needs to be predicted at each time step along the horizon are the vehicle arrivals on the back of each queue.

The aim of the TE module is to provide, each second, a matrix with m rows (where m is the length of the short term horizon, in seconds) and n columns (where n is the number of distinct queues of the intersection approach – n is equal to the number of signal groups of this approach). In this way, the first row of the matrix provides the arrivals in all queues in the next second, the second row in the following second and so on. This matrix is augmented by another $k \times n$ matrix, where k will be the remaining seconds from the end of the short-term period till the end of the optimisation horizon ($k = H - m$). The augmentation of the matrix will consist of the long-term period traffic estimates. The matrix presenting the arrivals during the **short-term** horizon has the form presented in equation 6.

$$C_{mn} = \begin{bmatrix} 1 & 2 & \dots & c_{1j} & \dots & 1 & 0 \\ 1 & 1 & & & & 1 & 2 \\ \vdots & \vdots & & & & \vdots & \vdots \\ c_{i1} & & \dots & c_{ij} & \dots & & c_{in} \\ \vdots & \vdots & & & & \vdots & \vdots \\ 2 & 0 & & & & 0 & 0 \\ 1 & 0 & \dots & c_{mj} & \dots & 1 & 1 \end{bmatrix} \qquad (6)$$

where, c_{ij} are the arrivals to the back of queue j, in the ith second (from the time that the estimation was made).

Due to the fact that TL cannot incorporate n-dimensional matrices but just single dimensional vectors, the above matrix had to be separated in distinct vectors per signal group. Therefore, instead of one matrix, **n vectors, with m rows each** are estimated by the TE module. However, due to the fact that the strategy is described here on a theoretical level, as well as for simplicity, the term “arrival matrix” will be used throughout this section.

An important requirement for the arrival matrix is to be readily available when required from the signal optimisation module and to include the latest, up-to-date projections. Therefore, speed is an important requirement as an arrival matrix that corresponds to the projections of a few seconds before is already outdated. More complicated methods, which take into account upstream intersection data as well as upstream signal decisions in order to increase accuracy, result in a simultaneous optimisation-traffic estimation and thus can be more time consuming, in contrast to this method which is fast.

The philosophy behind the matrix formulation is that whenever a new vehicle is detected from the upstream detectors, its **future footprint** on the intersection queues is projected. The term footprint, instead of arrival is preferred, as the concept of the partial vehicle will be introduced.

When a detector actuation occurs, the module will create a "partial" vehicle for each signal group of the intersection approach. Each vehicle created will be assigned a value from 0-1, based on the respective turning percentage for that signal group. For example, assume only the left, through, and right movements are supported on a particular approach. Let the left turning percentage be 25, the through movement percentage be 65, and the right turn percentage be 10. The module will create three "partial" vehicles: one for left with weighting of 0.25, one for through with weighting of 0.65, and one for right with weighting of 0.10. If multiple lanes are available for a specific movement, it will be considered that the queue in each lane will be equal (as vehicles will always select the shortest queue to wait).

A concept used in this strategy is that the **queues are simulated per signal group** and not per approach or lane as in other strategies. This was done because in some cases, more than one signal groups might use the same lane, more lanes might serve a signal group, or even more than one traffic movements might be served in the same signal group. However, if queues per signal group are simulated (using the concept of the partial vehicle), the above problems can be handled. In this way, in some of the above cases, an observed queue at a specific lane, will not always correspond to a specific signal group queue. In any case though, if all signal group queues are summed, the total queue of the intersection approach will be obtained and will correspond to the actual (observed) queue.

The traffic estimation module will be called each second. When the module is called, it will firstly check **whether there were actuations** in the upstream detectors during the last second and if yes how many. This figure will then be split into the various signal groups to create partial vehicles as described before. The turning percentages for each intersection approach are dynamic and might change within the day; however, it is considered that changes do not occur rapidly, therefore an updated estimate at fixed intervals can capture the changes with sufficient accuracy. The module that estimates and updates the turning percentages is described later on.

Since the number of vehicles assigned to each queue is known, the time to arrive at the back of the respective queue has to be estimated. At pre-specified intervals, the average speed of the vehicles that passed the upstream detectors will be calculated. This speed will be assigned to each partial vehicle created. By taking into account each current queue (see below for the estimation of the queue at each second) and the distance between the detector and the stop-line, the distance to travel until the back of the queue can be estimated. By using the vehicle speed and the above distance, the travel time to the back of the queue will be estimated for each vehicle.

There are two ways the speed can be estimated by using inductive loop detectors. Either to use a single detector using the occupancy and assuming a fixed vehicle length (see chapter 2) or by using two successive loop detectors and measure the time difference between their actuations. The second method obviously provides more accurate results as it does not assume a fixed vehicle length however it requires more detectors to be installed thus increases the cost of implementation. For this application the first method was chosen.

The next step in this module will be to **update the arrival matrix** from the previous second. At a first stage, the previously existing arrival projections will be moved on to the next second (therefore from the ith to the (i-1)th row in the arrival matrix. At a second stage, the partial vehicles generated during the last second, will be assigned to the correct place (c_{ij}) in the arrival matrix, based on the travel times calculated above.

The final step will be to **augment the matrix** with the additional $k \times n$ elements estimated by the rolling average of the arrivals to the back of each queue during the previous time horizon. After this step, the arrival matrix can be exported so as to project the evolution of the queues in all intersection approaches and quantify the cost along each path of the tree, as described later in the implementation of the branch and bound technique.

Consistency checks

As mentioned before, an **initial estimate of the standing queue**, each time the optimisation module is called, is needed. The initial queue is based on the estimate of the queue length in the previous second, by also including two consistency checks, to ensure that it corresponds to reality. More specifically, at time t, the second row of the *queue* matrix (the matrix which is identically structured as the *arrival* matrix of equation 6 but describes the evolution of queues instead of arrivals) is the estimation of the queue for the (t+1)th second. This is the initial queue that will be used from the TE module when it will be called again. However, due to the fact that this queue is based on arrival predictions of the previous second and due to the fact that possible errors in the prediction will act cumulatively in time, appropriate consistency checks need to be introduced.

The **first consistency check** is based on whether the stop-line detector is actuated or not. If, all the stop-line detectors of a specific signal group are not actuated for a specific period (set to 3 seconds) but an initial queue is estimated by the module, this queue will be set to zero. In this way, errors occurring from queues which dissipated during the green time but the respective queue estimate provided by the TE module did not reach zero, will be identified and corrected. Respectively, if any of the stop-line detectors referring to a specific signal group is actuated but the estimated queue is zero, the queue is augmented by a fixed amount (1 vehicle). In this way, if the TE module estimated that a queue will clear during the green time but actually a small part of the queue still remained, the module will correct the queue estimate. This first consistency check is introduced to provide a correction to small queue lengths.

The **second consistency check** is introduced to provide a correction to large queue lengths. More specifically, the number of vehicles present between the upstream and stop-line detectors can be easily monitored at each time instant, by simply subtracting the vehicles that passed from the stop line detectors and adding the vehicles that actuated the upstream detectors. Therefore, if the sum of the vehicles in all predicted queues is larger than the total number of vehicles present within the link at a certain moment (as provided by the detectors), the queue lengths will be reduced *pro-rata* to correspond to this total number of vehicles. However, if the predicted queue approaches its maximum length (the distance between the stop-line and the upstream detector) this consistency check will stop as the queue in fact can become larger (the queue can spill back from the upstream detector).

In general, the two consistency checks act as measures to avoid the small errors in the queue evolution prediction to accumulate and result is large deviations from reality.

Oversaturation

Some vehicle actuated strategies (i.e. GASCAP) although operating efficiently in light or moderate traffic conditions, cannot cope with heavy traffic and thus switch to a fixed time mode as long as traffic remains congested. This; however, requires a reliable way to define the traffic conditions and the time that the strategy needs to switch in each mode. An advantage of this strategy is that it copes with the effect of oversaturation, not by switching to fixed time mode, but simply by modifying its traffic estimates in a way such that the optimisation module will provide signal timings resulting in a capacity strategy. Of course, in oversaturated conditions, the strategy is expected to operate more or less equivalently to a fixed time plan; however, it does not need to define any threshold of light-moderate or heavy traffic conditions, or switch on and off as it keeps optimising signal timings in the same way.

Concerning the TE module, the spill-back of queues upstream from the detectors is addressed as follows. If the queue is approaching the upstream detectors, then the vehicles passing from the detectors will arrive on the back of the queues in a short time, therefore the projections to the back of the queues can be made only for a very short time horizon. After this short-term horizon, the optimisation has to be based on the estimates for the long-term horizon. Moreover, if the queue reaches the detectors, they will become constantly occupied therefore no traffic flows can be measured (and no predictions can be made, even for the long term horizon). Therefore, it is critical to avoid the queues growing larger than a maximum length as this will degrade the performance of the strategy.

If a queue is approaching the upstream detectors, the queue evolution is not monitored anymore by the TE module in the way described before. In this case (at the second where the module identifies that the queue has become larger than its critical value) it uses a uniform arrival rate equal to the saturation flow of this approach. In this way, the projected queues will grow at a maximum rate therefore the optimisation module will attempt to dissipate them quickly. This is a gating feature that assigns a greater weight to signal groups with queues that are approaching the respective upstream detectors and thus avoids the risk of oversaturation. In the occasion that all intersection approaches are becoming congested, it is unavoidable that all queues will become larger than their critical lengths therefore the respective traffic estimates will be set to maximum. As mentioned before, in this case, the optimisation procedure will result in providing the maximum green durations and operate in a maximum cycle. Finally, when the queues finally drop sufficiently below their critical lengths, the normal procedures of the TE module will become active again.

Advantages

Some advantages of the TE module over other traffic estimation methods are illustrated. Using the above method to obtain traffic estimates for the signal optimisation is superior to obtaining a simple rolling average of the arrivals in the past, as in many existing strategies, due to the inclusion of a short-term horizon for which accurate predictions about vehicle arrivals at each queue are retrieved. Moreover, this method does not employ detectors upstream from previous intersections, as this would render the prediction much more complicated. The upstream signal optimisation would have to be taken into account into the prediction of the vehicle's arrival time at the back of the queue. This would finally result in a simultaneous optimisation problem for all intersections which would be too computational costly. Furthermore, the algorithm employs only two measurement locations per link, consisting of inductive loop detectors, without the need for video sensing for the automatic detection of queue lengths as required from other algorithms, therefore it minimises the hardware needs. Finally, it is an algorithm that can run fast, so as to obtain an updated estimate very frequently. This means that it enables shorter time steps to be adopted which result in more adaptive strategies.

Further considerations

In conclusion, some considerations are mentioned concerning the needs of traffic estimation. The discussion around traffic estimation and the needs for detailed and accurate arrival information strongly depends on the particularities of each VA application. When traffic does not show sudden fluctuations and the arrival profile is relatively smooth, the requirements for fast and accurate traffic estimation can be relaxed. In such cases, rolling averages over the last 60 seconds (or even longer periods) can deliver sufficiently accurate information to achieve efficient control. On the other hand, computational costly applications including dynamic predictions can prove redundant. Therefore, on a case specific application, it is important to identify the characteristics of the network demand so as to develop targeted applications.

On the other hand, in cases where traffic flows may change significantly over short periods of time or in cases where the response to unexpected incidents (which cause significant sudden variations on traffic flows) is crucial the need for quickly responsive approaches with a high resolution is apparent. Depending on the topology of the examined network, such an approach would be very likely to require the exploitation of data from upstream intersections for the signal optimisation downstream. The prediction horizon should be long enough to allow for envisioning the trip of a vehicle well upstream from the previous intersections until the stop line of the optimised intersection. On that purpose, several delay components would have to be taken into account (section 2.4.2 for PREDICT algorithm). The consequences of this choice on the speed and complexity of the algorithm were discussed in more detail in section 2.4.4.

For this study, the traffic estimation module was developed as a hybrid approach between modules used for reactive and pro-active strategies. The choice was to avoid the complexity imposed by including upstream intersection data without however omitting the pro-active component. The pro-active component is established by exploiting detector data on the approaches of the optimised intersection. Due to the fact that these data can only be used for predictions over a limited horizon (depending on the stop line-detector distance), the predictions for the rest of the horizon (long-term) are based on simpler rolling averages. In this way, the TE module avoids the complications when using detector data from previous intersections but pays a cost on the accuracy of the long-term horizon estimates. However, the algorithm is still able to quickly respond to sudden flow fluctuations. At a first stage it responds partly (through the short-term arrival estimates) and at a second stage (which can be delayed up to maximum 60 seconds) it fully responds by modifying also the long term arrival estimates.

4.2.2 Signal monitoring (SM) module

This module monitors the state of each signal group at each second. This module is included in the algorithm to provide information on which signal groups were green/red in the last second to the TE module, that is used in the estimation of the departures and the initial queues.

Among the input required by the TE module to estimate the existing queue for the following second are the departures for each signal group, therefore it needs to know the state of each signal group in the following second. The logic behind the SM module is that although it monitors the state of each signal group in the current second, this state will effectively be maintained also in the next second. For example, when a signal group is green, even if the next second will be yellow, it can reasonably be assumed that this first second of yellow is used by vehicles. On the other hand, if a signal group is red, even if the next second is green, it can be assumed that it corresponds to the lost startup time, therefore the effective green will start in the next second. Therefore, the state of each signal group at the current second, can be used in any case as a reliable estimate of the effective state of the signal group in the following second.

4.2.3 Turning percentage (TP) module

The Turning Percentage (TP) module provides an updated estimate of the turning percentages of each intersection approach at pre-defined time intervals to the TE module. The turning percentages are dynamic and can change during the day. Knowledge of the current turning percentages is important for the TE module in order to create partial vehicles for each detector actuation and assign them to the correct queues.

Although the turning percentages change within the day, their changes are not as rapid (with a possible exception of unexpected incidents) as for example changes in traffic flow. The evolution of the turning percentages under normal conditions can be described with a sufficient accuracy if an updated estimate is obtained every 15 minutes. This is a rational assumption that can be drawn by observing the real traffic data obtained by the loop detectors for the examined network. A reliable turning percentage estimate could be obtained in even longer intervals; however, the 15 minute interval is maintained to increase the robustness of the algorithm to unexpected incidents that may affect demand.

Ideally, the turning percentages can be obtained by observing the vehicle routes through the examined network. With a sufficient number of route observations, representative turning percentages can be estimated. However, this approach might be problematic as the only means to detect vehicles in the examined network are inductive loops therefore it is difficult to re-identify the same vehicle in successive detectors. However, turning percentage estimates can also be obtained by observing the number of vehicles turning from a specific stop line, by using the stop-line detector actuations. A disadvantage of this approach is that it is not possible to estimate turning percentages if a stop line serves more than one turn. In the intersection used for the evaluation of this strategy 1 such stop line exists. In this stop line, the respective detectors were placed a bit downstream from the stop line on the trajectory of each turn. These detectors will be used for the estimation of the turning percentages, instead of using a single stop line detector.

A problem that might arise in this case is that dividing the number of vehicles that passed from a detector by the sum of the vehicles that passed the respective cross-section do **not necessarily** provide an accurate estimate of the **demand** for the specific traffic movement. This is because the green time for a specific stream might not be sufficient for all vehicles in the respective queue to clear the intersection, therefore, its demand can be higher than the one actually measured (case of growing queues). If this is the case for one traffic movement while the rest of the traffic movements have lower demand (therefore their queues are cleared each time the stage runs) the turning percentage estimate might be biased (show a lower percentage for the movement that is not cleared than the actual). To mitigate this, a longer period of traffic counts can be adopted. In a longer period, the possibility to capture the growth and decay of a queue as well as a period where the green is sufficient for a traffic movement and a period where it is not, is higher, therefore the estimates will be less biased. For that reason, the turning percentages will be updated in 15 minute periods, so as to include several stage sequences within the examined period.

A practical issue concerning the implementation of the algorithm in VISSIM should be mentioned: In VISSIM, the turning percentages cannot be set to be stochastic or follow a certain probability distribution. They can only be pre-defined for several fixed time periods. Therefore, for the VISSIM implementation of the strategy, the adoption of the TP module would not be needed (as the turning percentages could be fixed into the algorithm). Despite that fact, the module was included to test its effectiveness, as the feasibility of the strategy to a real world situation is one of the objectives of this project. In this way, the turning percentages estimated by the TP module were compared to the fixed turning percentages of VISSIM and the credibility of the produced results was assured.

4.2.4 Branch and bound (BB) module

Module structure

In order for the B&B module to run, there is a need to obtain the required data from the previous modules. More specifically, it will request the traffic estimates from the TE module, the turning percentages from the TP module as well as the signal group states from the signal monitoring module. All these modules run independently and some of them with a different frequency. Therefore, the algorithm will simply request the latest output of each module each time the optimisation is initiated.

The Branch & Bound technique will be the core of the optimisation module (BB). This module will be called every second to optimise the durations of the stages for the horizon starting from the next time step. If a stage transition is running, a specific variable indicating the current stage will take the value -1, which will in turn result in skipping the optimisation until the next time step, where the above interrogation will take part again.

The module is built so as to allow for the expansion of the full decision tree (see chapter 2), if this is needed. In most cases, many parts of the decision tree will be pruned due to maximum green violations as well as due to the Lower Bound becoming greater than the Upper Bound (for more information on how the Lower/Upper Bounds are filled, see description of Branch and Bound technique). However, no path can be excluded before starting being evaluated; the module will start to examine every path of the decision tree, until this becomes unfeasible or worse than the current optimal. Every path that reaches the end of the horizon will be compared to the current optimal and will either replace it or be discarded.

To achieve all the above, "for" loops are employed (see Annexes 3 and 4) so as to build the decision tree. For each "for" loop, the counter starts from 0 and ends at the number of possible stages that can follow. In this strategy, where the stage sequence is fixed, only one stage can follow the previous, therefore, the counter of the "for" loop takes only the values 0 and 1. The number of loops is equal to the maximum number of switching steps needed to reach the end of the horizon. This number is maximum because the stage transitions, will have durations greater than the switching step but will be evaluated within one step as well. Therefore, only the path which continues serving the current stage and extends it until the end of the horizon will include the maximum number of time steps, but of course this path will always be unfeasible because it will always violate the maximum green for this stage (in practice no maximum green is 80 seconds long). Therefore, although the maximum number of time steps is included in the decision tree for completeness, this number of steps will never be used in practice.

Under the loops of the decision tree all calculations concerning the cost estimation and accumulation, the definition of the respective stage durations and the consistency checks needed, take place. The cost calculations for the first time step are included within the BB module while the cost calculations for all subsequent time steps are written within two subroutines that are called respectively (see Annexes 3 and 4). At the end of each "for" loop, the consistency checks, which indicate whether the evaluation of this path should be continued or pruned are performed.

This way, the paths that reach the end of the horizon are evaluated and the terminal cost (TC) of the current path is added to their final cost (LB). If the examined path is the new optimal, the respective stage durations are calculated and stored to specified variables that are going to be passed to the interface module to be transferred to VISSIM.

The implementation of the Branch and Bound method for traffic signal optimisation has certain difficulties that were identified during the development of the code and can be useful to future researchers who wish to adopt this method. Those lessons learned are discussed below.

The fact that the stage transitions can (and most probably will) have durations different than the time step, has two main implications: **Firstly**, if the transition durations are not integer multiples of the time step, many paths will not end exactly at the last second of the horizon but some seconds after. **Secondly**, the number of used steps to reach the end of the horizon will differ between paths as a transition is examined within one step, but can last for more seconds than the switching step. Both implications can have a serious impact on the implementation of the method.

Concerning the first, a way has to be found to accumulate the cost only until the end of the horizon and not in the seconds that exceed the horizon. Otherwise, paths which end as close to the horizon as possible will be favored while paths that end many seconds after will be penalised. However, if these paths are examined only until the end of the horizon a path that exceeds the horizon more, will be evaluated fairly.

At first, it was considered that the longer the horizon, the smaller the effect of this implication would be. However, this did not turn out to be the true. If the horizon ends at the 80th second, then paths ending at this second will most probably yield lower costs than paths ending at the 90th (such a path can arise from the last step starting from second 75 and including a transition lasting 15 seconds). This is because the cost rises exponentially with the number of steps, therefore even if a path yields much less cost until the 80th second, the fact that it includes two more steps will result in a higher LB.

To achieve the cost accumulation until the end of the horizon only, many approaches can be adopted. Firstly, the cost estimation method can be diversified so as to calculate the cost only at the 80-th second and not at the end of every step. However, this method can only be used when explicit enumeration is adopted and not when Branch and Bound is employed because for the latter, accumulation per time step is required to prune paths that become sub-optimal. Secondly, a method can be developed to prune the cost of the last switching step, if this exceeds the 80-th second. For this, appropriate scaling factors can be used. However, this solution was not further examined as it was not possible to invent an appropriate scaling factor with the cost estimation formulas used. If this solution is to be adopted a different cost estimation method should also be adopted. The solution that was finally selected, was able to provide a solution also for the second implication mentioned above. This solution is described after the second implication is discussed.

Concerning the second implication, due to the fact that the cost should be accumulated at each time step and that the LB has an exponential function that is also dependent from the number of steps, and that the arrivals within a switching step are considered uniform, all paths that are evaluated at each optimisation should have equal number of steps. Moreover, all steps (within a path as well as between different paths) should have the same duration because otherwise the costs estimated from equations 1-5 are not comparable. All paths will have the same duration (equal to the horizon) therefore a path with 10 second switching steps will use half of the time steps than a path with 5 second switching steps. In this way the LB variable which is dependent on the number of steps cannot be compared in the two cases.

This implies that also the cost for the stage transitions cannot be estimated for the whole transition at once but should be estimated by breaking the transition into a number intervals equivalent to the switching step. However, an additional problem arises if the transition cannot be divided into an integer number of steps (i.e. when a transition lasts 11 seconds and the switching step is 5 seconds). For that reason, the stage transitions have to be modified to have a duration which is an integer multiplier of the switching step. This can usually be made easily by adding or subtracting (if possible) a couple of seconds from the existing stage transitions. This is easier to be achieved when small switching steps are adopted (i.e. 2-3 seconds). When the transitions are integer multiples of the switching step, they can be divided into sub-steps therefore, the total number of (sub) steps between all paths will be equal. Moreover, each path will definitely end at the exact ending second of the horizon and not afterwards therefore the first implication mentioned is also dealt with.

This solution does not have any serious implication in the effectiveness of the strategy and does not require adopting a different cost accumulation method than the one described above.

Possible enhancement

In this thesis, as well as in other works (Shelby, 2004) it proves that the B&B technique can sometimes be slow, especially as the time horizon grows and the time steps shrink. Therefore, as was illustrated in chapter 2, compromises have to be made by sacrificing part of the algorithmic accuracy for speed.

For that purpose, ways have been explored to further reduce the computational burden of the B&B technique. More specifically, it was observed, that although the strategy uses a horizon which is divided in a maximum number of time steps, not all steps might need to be used in the resulting optimal path. As mentioned also in the previous paragraphs, most paths will reach the end of the horizon before using all steps due to the stage transitions having durations larger than the time step. However, another observation can be made: For **certain sets of maximum greens** for each stage, the full number of time steps of the horizon might **never** need to be used. For example, with the set of maximum greens used in this thesis and with the current time step (5 seconds) the maximum number of steps that can be used in a path are 14, although the horizon is 80 seconds. This occurs as follows:

Starting from an arbitrary stage and assuming that it has just started, if we move towards the end of the horizon step by step, reaching each stage's maximum green and then adding one step for the next transition, at the end we will end up with the maximum number of steps that can be used starting from this stage. If this procedure is repeated starting from all stages the overall maximum number of steps can be retrieved. This number resembles the maximum number of "for" loops that will ever be used by the algorithm. Due to the fact that no stage has a very long maximum green and that all stage transitions have a duration longer than the step, this maximum number is less than 16. This maximum number is valid **only** for this combination of **horizon, step length, transitions' length and set of maximum greens**.

By omitting the excessive "for" loops, one can gain significantly in speed (as an unnecessary part of the decision tree is automatically pruned) without any compromise in accuracy. The horizon will still have the same length and the resolution of the algorithm will remain the same, but the part of the tree which would always be unfeasible, is removed.

However, as said this modification will hold only for the specific set of parameters. Thus, it cannot be generalised, but can only be implemented after the algorithm with all the steps is developed and after the set of maximum greens has been defined. Moreover, if any of the aforementioned parameters changes, the maximum number of steps has to be re-defined.

This modification was implemented in the version of the algorithm used for the evaluation of the strategy, to speed up the simulations. The resulting maximum number of "for" loops required, using the given maximum greens and time step, is 14. So, 14 loops instead of 16 were maintained in that version. However, the version presented in this document is the one with all 16 time steps, because it intends to be general and be valid for any set of maximum greens or step length. The modification is easy to be performed and simply needs to omit the excessive "for" loops and step evaluations from the respective module.

4.3 Development process

This chapter provides an overview of the strategy development process. It briefly describes some of the major obstacles faced during the development and how they were overcome. It also provides some lessons learned through the development process. The framework of the final version of the strategy and the interrelations between the different modules are presented in Figure 6. This figure shows only the basic functions of the algorithm and does not show a number of sub-routines called within these functions. For a detailed description of all functions and subroutines see Annexes 3 and 4.

In summary, the implementation of the strategy in VISSIM using the SITRAFFIC software included the following steps:

1. Development of the code in Traffic Language. The result after the compilation is an executable file which is used as an input to VISSIM.

2. Development of the signal plans, signal control definitions and intersection infrastructure related data in P2.
3. Development of the controller input in Control. The input to the controller included the fixed time program to be used as a basis (from P2), several other signal control definitions, TL parameter definitions, TL outputs to be used by the controller and more. Control finally produces two files for VISSIM and one file (parameter definitions) for TL.
4. The VISSIM file. All the above files should be properly inserted to VISSIM. The signal controller that has to be selected is the "SIEMENS VA" controller.

This chapter will discuss the first out of the four parts mentioned above. The first version of the strategy was ready around the end of August 2010 and the time required for its development was about 1,5 months. However, as expected in such tasks, the most challenging part was to correct errors in the code and inappropriate handlings in the use of the above software. The absence of comprehensive manuals for the use of the SITRAFFIC software was an additional difficulty in the development process. Since the first version, another 5 months and around 30 updated versions of the strategy were required to achieve a fully operational version.

Some of the major revisions that took place during the development process were selected and are briefly discussed below. The first was the re-definition of the number of time steps from the initial 30 to 16 and of the switching step from 3 to 5 seconds. The initial revision was made due to the fact that TL could not cope with 30 successive "for" loops as it resulted in the program crashing, even if no commands were included under the loop. At later stages it would be realised that a binary tree with 30 steps would also result in a too computational costly problem for the capabilities of the available hardware so the size of the tree would have to be reduced anyway. The exploration of the computational limits of the existing hardware so as to conclude to the optimal combination of number of steps and length of step but also to a combination that can result in sufficiently efficient optimisation was a trial-and-error process that is related to the hardware and does not imply any limitations to the theoretical concept of the strategy.

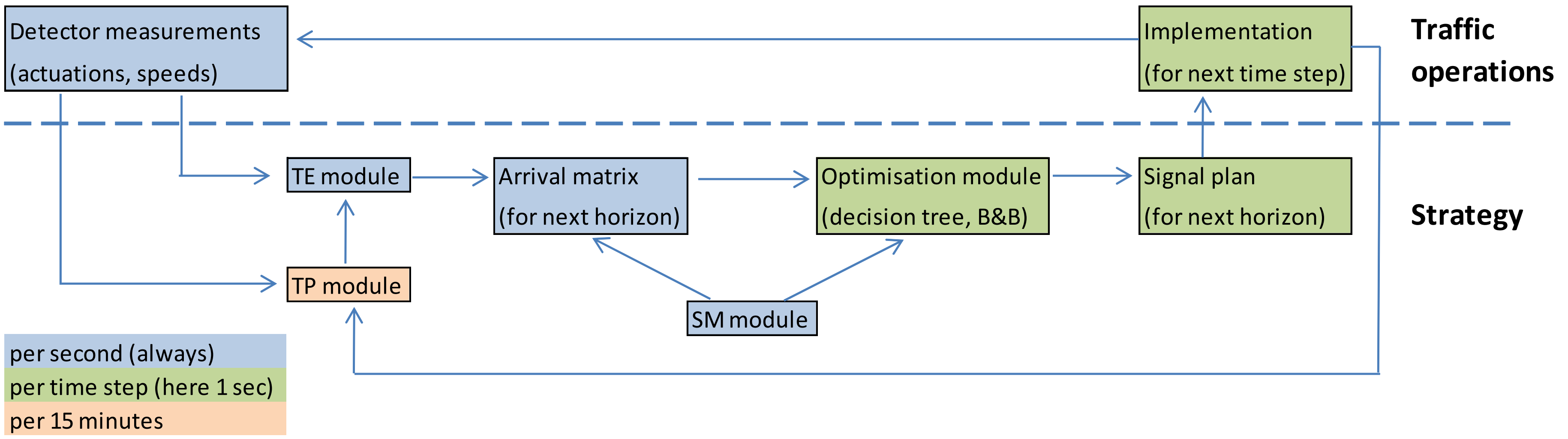


Figure 6. Strategy framework

A second major revision resulted from the fact that TL cannot include variables with more than 1 dimension but only single dimensioned vectors. The concept of the strategy was built so as to include several matrices (i.e. the output of the TE module) therefore a major part of the code had to be revised respectively. That required the inclusion of a number of additional subroutines that have to be called every time a matrix has to be used (either updated with new data or for the selection of data from it). Being unable to use matrices as well as counters (used in "for" or "while" loops) that count from the start till the end of the dimensions of the matrix (as is the case i.e. in Matlab) generally makes the code less flexible and requires variables with more dimensions to be used.

Moreover, TL has certain computational limitations that need to be addressed, i.e. it does not keep decimals. It simply discards them and keeps the integer part of a number (without rounding it). For that reason, in any calculation where a non-integer might occur, the respective figures were multiplied (by 10, or 100) to ensure that enough information would be preserved. This resulted in higher figures than those that would be obtained if decimals were kept. However, TL has an upper limit in the numbers that can be stored (around $4{,}2*10^9$). This was an obstacle in the implementation of the BB method as the cost rises exponentially and when using a high number of steps it is difficult to keep the LB rising below that threshold. This is more difficult due to the fact that many figures had to be multiplied by 10 or 100. For that purpose, many figures are continuously re-scaled along the algorithm (sometimes multiplied and other times divided).

After the development of the first version of TL that could be compiled without critical errors it was possible to start attempts for the implementation of the strategy in VISSIM. At this point, the algorithm has to be tested for its efficiency. However, before that, latent mistakes in the compilation have to be identified and corrected.

From the development process, three distinct categories of such mistakes were identified that result in different types of problems when trying to implement the strategy. These categories are:

1. Mistakes in the compilation (i.e. misuse of certain commands, mistakes in the variable definitions like the required number of dimensions, mistakes in the parameter definitions etc). Such mistakes arise due to lack of experience from working with software and are usually identified and eliminated with time. This category also includes mistakes in the handling of functions (i.e. assuming that the units of flow measured by the detectors are in veh/h while they are veh/sec). In such cases, a comprehensive manual for the existing software can significantly assist in minimising the delay incurred by such mistakes.

2. Mistakes in the logical arguments (i.e. an $x>=0$ condition is needed instead of a $x>0$, otherwise a division by zero will occur and terminate the code). Such mistakes arise quite commonly in programming. The best way to eliminate them is by observing the results and following a backward process, by identifying possible sources, examining each and eliminating them, until finding the actual cause and correcting it.

3. Mistakes in the strategy itself: This category refers to flaws in the core logic, i.e. how good estimates the traffic model provides or how efficient the optimisation can be. Obviously this category is the most important as a mistake of this type has the most serious implications to the strategy.

The log files resulting from the TL runs were the most important mean to assist to identify and correct mistakes. However, one major disadvantage existed. When the code included too many calculations (as was in the case of this strategy where the optimisation module had to evaluate tens of thousands of paths) the resulting log file was too big (of the magnitude of hundreds of Gigabytes for a 20-step optimisation) and thus could not be opened from TL neither from other text or C++ readers. This made the effort of identifying some mistakes much more difficult as a lighter version of the algorithm with fewer steps (smaller horizon) had to be prepared. All mistakes in the strategy had to be identified through this file.

However, some mistakes were only arising in the long horizon strategy and therefore were too difficult to identify and correct. As an example: a problem that can arise in the implementation of the Branch and Bound method (mentioned also in the previous sections) is the inequality of the path lengths, if the stage transitions do not have durations that are integer multiples of the time step. This problem can remain unnoticed when a 2-step strategy with 10 second time steps (20 second horizon) is implemented, because out of the 4 possible paths that can be chosen, not many possibilities for errors exist. Especially, when a simple version of the simulation is used (i.e. flows only from one direction) such mistakes in the optimisation can hardly be observed. Therefore, the impact of this mistake becomes important only when the algorithm becomes more complicated.

In the case where the log file could not be exploited, another method was employed. Any variable used in TL can be transferred and be displayed in the simulation environment during the run. This means that the code developer can observe the values assigned to any variable during the simulation and thus can identify mistakes. However, in this way, it is only possible to observe that a variable has a mistaken value and not examine the calculations themselves. Despite that fact, this proved to be a useful way to identify mistakes when the log file was unusable.

In conclusion, the most significant efforts for the development and fine-tuning of the strategy were required for the TE and the BB modules. The TE module had to be revised in several parts (speed calculation, queue evolution, arrival matrix augmentation, queue consistency checks, vehicle detection) until it provided a sufficiently good output of future arrivals and queue evolution. Concerning the BB module, the time required for the optimisation was a key parameter. Too large decision trees (resulting both from the excessive nr of steps and small step lengths) could not be handled therefore the right balance of algorithm efficiency and implementation feasibility had to be found.

In the following chapter the effectiveness of the strategy under different scenarios is assessed, so as to provide robust measures for its comparison with the existing fixed time plans as well as additional evidence that support the claims made in this report for its contribution to the current state-of-the-art.

5. Implementation and testing

This chapter deals with the presentation of the results from the implementation of the new VA signal control strategy in VISSIM. The theoretical concept described in chapter 4 is tested in practice and its performance is assessed. The results under different demand scenarios are analysed and discussed and conclusions concerning the effectiveness and plausibility of the strategy in general are drawn.

More specifically, this chapter is divided into the following sections:

The first section describes the selection of performance measures for the evaluation of the strategy and presents the method that is followed to present and organise the results. Moreover, it explains the selection of the specific performance measures and finally presents the way that the network robustness is assessed.

The second section discusses validation issues concerning the results presented in this chapter. Validation of results relates to the selection of the traffic flow sets described in chapter 3 and is important for the interpretation and usability of the results.

The third section is the core of this chapter and presents the results through a series of tables, which include aggregate data retrieved from the VISSIM evaluation files. A presentation and discussion around the results is carried out and an attempt to link the observed values to algorithmic actions and calculations is attempted. The results are presented in various degrees of disaggregation so as to obtain a clearer picture of the strategy performance. Finally it presents the evaluation of the network robustness under the new strategy and distinguishes and examines a number of cases which show high interest for the network operation under this control method.

5.1 Evaluation method

In this section the methodology used for the strategy evaluation is described. More specifically, the chapter refers to the adoption of a number of measures of effectiveness to assess the strategy performance compared to the existing fixed-time plans and the strategy robustness testing.

As discussed in chapter 2, the measures of effectiveness are the criteria by which the success of the strategy implementation is evaluated. These can be different from the performance measures used for the signal timings optimisation. For example, the optimised measure of a signal control strategy can be the vehicle delay over an intersection but the strategy can be evaluated by the average number of stops per vehicle and the average/maximum queues it results to. A more extensive discussion on the strategy objectives has been presented in that chapter while here, the aim is to define a set of performance measures so as to quantify and evaluate the performance of the strategy.

In general, measures of effectiveness can be distinguished in qualitative and quantitative as well as in efficiency-related and more general measures. A qualitative, general measure is for example the road user acceptance. Although such a measure can be important for the strategy success, it cannot be adopted in this project as its implementation is only made in a simulation environment. Quantitative, efficiency-related measures can be extensively used here by exploiting VISSIM capabilities. The categories that can be exploited include a vehicle record which provides data (such as speed, delay, stops, travel time) for every single vehicle travelling in the network, a node record, which concerns areas around the intersection defined by the user as nodes (and also includes delays, stops, emissions etc), link specific data, user specified point data (cross-sections) and network performance measures which provide results for the network as a whole.

Due to the fact that, in this exercise, one intersection is evaluated, the results were estimated:

1. Per intersection approach and
2. For the intersection as a whole

The second category serves for obtaining an overall picture of the improvement on the intersection performance under the new VA strategy while the first category serves for drawing a more analytical picture of the results per intersection approach so as to determine whether the strategy results in balanced control between all traffic movements and whether the improvements yielded by the strategy in the whole intersection can be attributed more, to the improvements yielded in specific approaches. Moreover, it can be recognised whether part of the improvements is retrieved due to the mitigation of specific weaknesses of the fixed-time plans, by the application of the VA strategy.

As the network mostly serves passenger car traffic, without significant PT existence (see chapter 3), the measures are not distinguished by vehicle type.

The measures used for the strategy evaluation are:
- The average delay per vehicle
- The average number of stops per vehicle
- The average vehicle speed
- The average stopped delay per vehicle
- The average and maximum queue lengths
- The total fuel consumption

The average delay, stops and stopped delay can be obtained from the VISSIM evaluation files, both for the intersection as well as per intersection approach. The queue length is measured by using a queue counter per approach that obtains an average queue estimate at intervals of 5 minutes. Finally, the fuel consumption is obtained by using the node evaluation feature. This can provide an estimate of CO, NOx, VOC and fuel consumption for the selected nodes. However, due to the fact that the emission model used is in fact a black box, and is not validated for the specific study area, the absolute values obtained cannot be trusted. However, the results can serve for comparison between the fixed time and the VA strategies. After running the simulations, only the fuel consumption was selected for presentation, because all emission measures were reduced by exactly the same percentage by the application of the VA strategy, thus presenting more emission estimates would not contribute to any additional knowledge.

The required number of simulations, to be ran for each scenario, was determined by the statistical significance of the obtained results. It was concluded that by running 10 simulations for each scenario and strategy, most of the differences between the fixed-time and VA results, were statistically significant at a 99% level of significance, while the resulting confidence intervals for the means were relatively low.

The robustness of the network under the application of the new strategy is also evaluated. However, road network robustness, being a relatively new concept, is not easy to test using a number of simulations. As discussed later in section 5.3.3, the lack of a well established methodology and the large number of scenarios that would be needed to retrieve a good overall picture, call for a different approach. A qualitative method is adopted here, in which, the impact of different scenarios on the algorithmic calculations is analysed and the resulting consequences are identified. In the end of that section, a practical example illustrates how a more analytical exercise is designed and performed.

For the following sections, a numbering of the intersection approaches is used (1-4), which serves for reference in the tables and the text. The numbering of the approaches is the following:
SE approach: 1, NW approach: 2, NE approach: 3, SW approach: 4.
The numbering is also illustrated in the intersection layout in Annex 1.

5.2 Validation of results

The simulations presented in this chapter are based on a standard set of traffic flows (presented in chapter 3). To derive this set of traffic flows, traffic data from existing measurement locations as well as manual counts were exploited. Different demand scenarios were identified and traffic data for a pre-defined period were selected and aggregated. This process resulted in a set of traffic flows per demand scenario, which was used to load the network during the evaluation of the strategy.

However, when using a standard set of traffic flows, the stochastic component in the process of loading a road network is overlooked. In a road network, for a given OD matrix (demand), depending on the traffic assignment, the resulting traffic flows per link can be different. On that purpose, demand should ideally be defined in terms of an OD matrix and not as given flows on specific links.

For this study, where the network consists only of a single intersection, differences in the traffic assignment can be translated as different turning percentages over the intersection. In this study, turning percentages per demand scenario are assumed to be fixed, based on the derived set of traffic data. This raises an issue concerning the validity of the results presented in the following sections.

The variations in the turning percentages (under a given traffic demand) would ideally be simulated if a probability function for each turning percentage could be defined. In this way, the turning percentages would be able to vary around a constant value (mean) which would be defined based on the provided dataset. Unfortunately this is not able when using VISSIM as routing proportions can only be deterministic.

Moreover, if a network of intersections was simulated, a more meaningful approach would be to provide an OD matrix as an input and retrieve different flow sets (based on different traffic assignments), for each level of demand. However, this would introduce an additional series of simulations for each demand scenario and control method, significantly increasing the total required number of simulations for the strategy evaluation.

In this study, it is assumed that the traffic assignment on the network is made deterministically, resulting in a standard set of flows and turning percentages per demand scenario. The reason is that the focus in this thesis is given more on the development of a new strategy and on the originalities of the proposed signal control approach. The evaluation results for the strategy are to be interpreted keeping in mind that a further validation of the results would provide additional confidence on the yielded improvements.

5.3 Testing

5.3.1 Evaluation design

Sections 5.2.2.1 – 5.2.2.4 of this chapter present the results from the simulations of four demand scenarios tested under the vehicle actuated and the fixed time strategies. This chapter answers research questions 13 and 14 presented in the introduction:

13. *How does the strategy perform in comparison to the existing fixed time signal plans?*
14. *What is the robustness of the developed strategy?*

The definition of the four demand scenarios and their characteristics, as well as the definitions of the four fixed time signal plans used currently in the examined network, are provided in chapter 3. The fixed time signal plans are also presented in Annex 2.

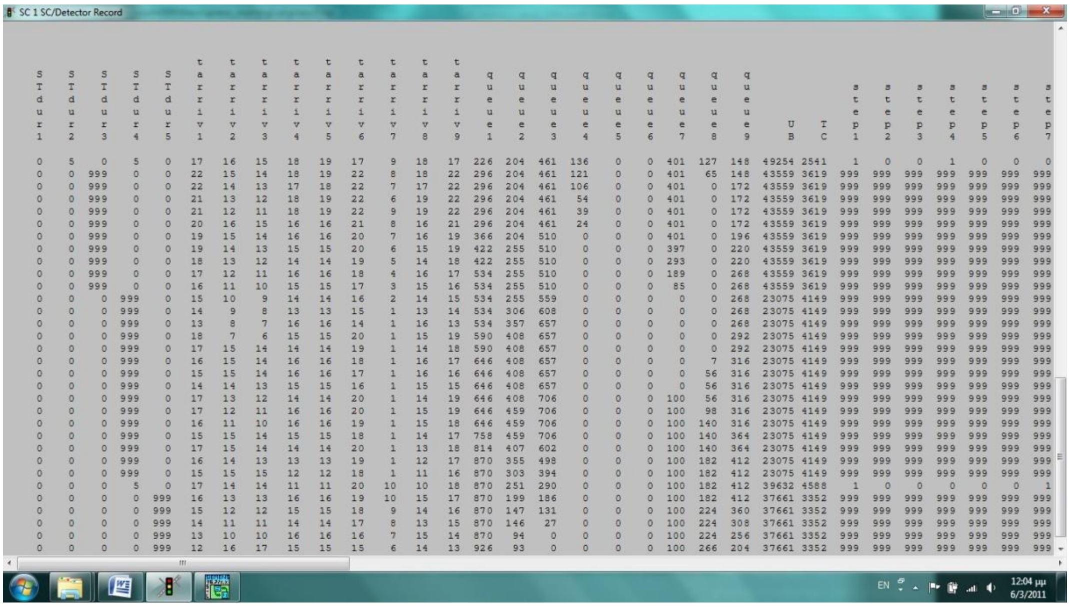
The simulations will include a warm-up, a main period and a transition period. The first will serve for loading the network in the appropriate state just before the examined scenario starts, while the latter will serve to include a transition phase to the next demand scenario. The four demand scenarios are sequential, when the one ends the next starts. All together, the four demand scenarios cover the demand profiles observed during a whole typical day.

The traffic flow within the main period of each demand scenario remains constant. Each simulation lasts for 2,5 hours (in simulation time), where the first and last 15 minutes correspond to the warm-up and transition periods and the 2 hours in-between correspond to the main period. Each simulation was repeated ten times, using different seeds, to ensure the credibility of the results.

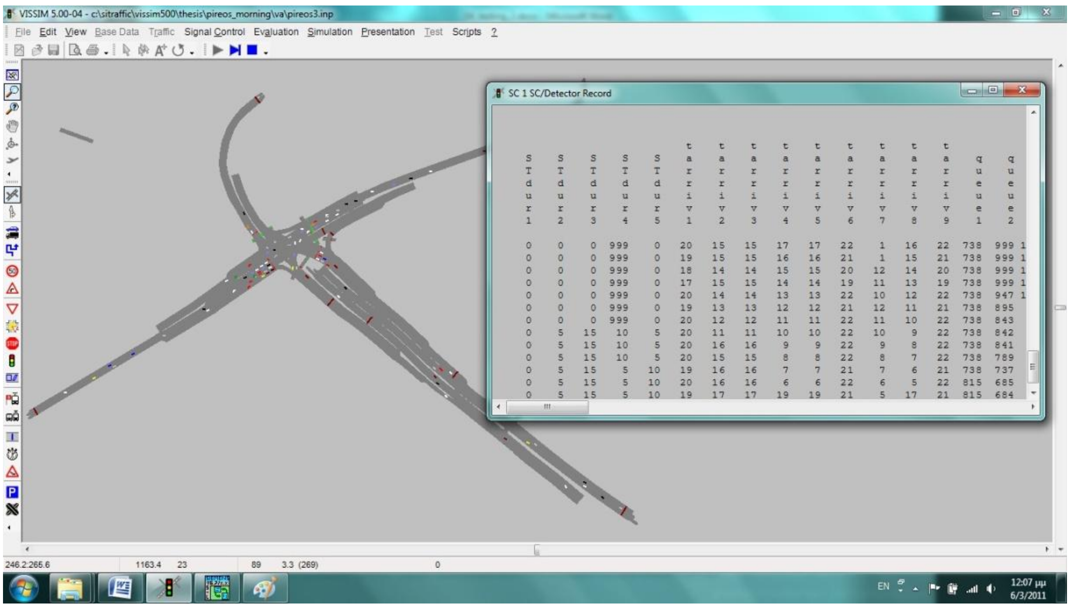
The arrival process of the vehicles in the network changes from day to day, although the average traffic flow can be approximately same. In order to obtain set of performance measures which are not dependent on the arrival process of one specific simulation, several simulations, for the examined peak hour are made. Each simulation uses a different seed (arrival process), although the hourly traffic flow remains the same. In order to run several simulations with different seeds, the VISSIM feature called “multirun simulation” is used. However, the set of random seeds that are used to evaluate each scenario is the same when comparing the performance of two different strategies. On that purpose, random seed numbers 1-10 were used in all multirun simulations performed.

In conclusion, 80 simulations were ran, occurring as follows: 4 demand scenarios multiplied by 2 (one for fixed-time and one for VA control), multiplied by 10 times that each simulation was ran. The results from each multirun were averaged to obtain a representative set of results per scenario and control method.

Apart from the evaluation files obtained at the end of a simulation, the evaluation is also made qualitatively during each simulation run, by observing the plausibility of the resulting signal timings and the variables used for the optimisation. A wide set of variables, is transferred from TL to VISSIM and is displayed in real time, each time the algorithm is executed (each simulation second). In this way, the optimisation process can be followed by observing the values of key variables, (i.e queue lengths), which can directly be compared with the observed ones. In this way, if an estimate significantly disagrees with the observations, a malfunction in the algorithm can directly be identified.



Picture 4. SC-Det record in full screen



Picture 5. Key variable evolution displayed in the SC-Det record during the simulation run

The VISSIM evaluation file that is used to display the transferred variables is the “SC-Det record” (signal controller-detector record), which can be opened through “Evaluation → Windows → SC-Det record”. Two screenshots of this window are shown in pictures 4 and 5.

5.3.2 Results

5.3.2.1 Scenario 1 - morning

Table 6 shows the results from the multirun simulations performed for the morning scenario, under the fixed-time and VA strategies. This table presents the values for the whole intersection, to provide an overall picture of the improvement yielded by the application of the new strategy. The results are further analysed per approach in the following tables.

The first ten columns show the results obtained from each simulation run while the last four columns show the average values as well as the variances, standard deviations and confidence intervals for the mean, on a 95% level of significance. All subsequent tables use the same level of significance for the mean values. Finally, the difference in the network performance between the fixed-time and VA strategies is quantified in the lower part of the table. All differences presented there are assessed using a 99% level of significance.

From table 6, it can be observed that the confidence intervals for the average values of all performance measures are small which means that irrespectively from the variations in the arrival process, the new VA strategy is able to significantly improve the traffic operations along the intersection.

This table shows the general trend when using the new strategy. It can be observed that all performance measures are improved when replacing the existing fixed-time plans with the new strategy. The average delay per vehicle is reduced 28% while the stopped delay is reduced as much as 30%. The average number of stops is reduced around 20% while the average vehicle speed is increased about 18%. Despite the overall improvement; however, there is a need to examine more carefully the source for each of those reductions. In the following paragraphs, an attempt is made to identify those parts of the intersection where the strategy yields more improvements and those where the differences are smaller. Moreover, it will be assessed whether the improvements are gained due to specific flaws of the fixed time plan or due to a general, well pronounced superiority of the new strategy.

Table 6. Average delay, nr of stops, speed and stopped delay per vehicle, for the intersection (Scenario 1)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	45,87	50,88	33,26	33,20	43,30	45,08	35,11	46,00	51,92	38,58	**42,32**	**47,95**	**6,92**	**4,95**
Average number of stops per vehicle	0,92	1,03	0,69	0,69	0,88	0,91	0,72	0,92	1,07	0,77	**0,86**	**0,02**	**0,14**	**0,10**
Average speed [km/h]	21,24	19,99	25,17	25,20	21,96	21,45	24,50	21,20	19,75	23,36	**22,38**	**4,17**	**2,04**	**1,46**
Average stopped delay per vehicle [s]	30,20	32,81	22,59	22,49	28,86	29,75	23,84	30,15	31,56	26,44	**27,87**	**14,26**	**3,78**	**2,70**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	31,39	31,39	29,10	29,06	29,95	31,00	29,87	30,06	30,78	30,35	**30,29**	**0,71**	**0,84**	**0,60**
Average number of stops per vehicle	0,70	0,70	0,67	0,66	0,68	0,70	0,68	0,68	0,69	0,68	**0,68**	**0,00**	**0,01**	**0,01**
Average speed [km/h]	25,89	25,89	26,81	26,83	26,46	26,01	26,48	26,42	26,11	26,32	**26,32**	**0,12**	**0,34**	**0,25**
Average stopped delay per vehicle [s]	20,18	20,18	18,59	18,70	19,27	20,14	19,18	19,34	20,01	19,50	**19,51**	**0,36**	**0,60**	**0,43**
Difference											**diff.**			
Average delay time per vehicle [s]											**-28,4%**			
Average number of stops per vehicle											**-20,4%**			
Average speed [km/h]											**17,6%**			
Average stopped delay per vehicle [s]											**-30,0%**			

In Table 7, the average and maximum queues, per intersection approach under the two control strategies are presented. In these tables, the values obtained from each distinct simulation run are not presented; only the average values for all 10 simulation runs are shown. However, the confidence intervals (on a 95% level of significance) for those means are included.

To obtain the queue estimates, a queue counter was placed before each stop line, to provide the average and maximum queue for that approach, at 5 minute intervals. The average value for the whole simulation time was obtained from those intervals.

A note must be made, that the maximum queue values, can be biased, if the queue is reaching the upstream end of the approach (thus expands outside the VISSIM network). If this happens for large parts of the simulation time, then the respective queue estimates can be under-estimated because the actual queue could in fact become greater if more space was available. For that reason, the *maximum queue* estimates that are close to the approach length should be interpreted with caution. This is the case for **approaches 3 and 4** where the maximum queue values (under **fixed-time** control) are approximately equal to the approach length. This practically, means that the reductions yielded by the application of the VA control can be even higher than those recorded, as the recorded queues under fixed time control might have been longer if the approach length was longer.

In general it can be observed that the average as well as the maximum queues are significantly reduced under VA control (at least 99% level of significance for the observed differences). While the reductions of the average queue, for approaches 1 and 2 range from 22-27%, the respective reductions for approaches 3 and 4 range from 45-70%. This reveals that a more significant part of the attained improvements presented in table 6, is achieved due to the higher queue reductions in approaches 3 and 4. This can also reveal a weakness of the fixed time program to serve traffic flows in approaches 3 and 4, which result in long queues during the morning peak hours. The VA strategy due to its dynamically changing green distribution and variable cycle length, manages to avoid those long queues from being formed and thus brings more significant improvements to those intersection approaches.

Even if it is claimed that a better suited fixed time program can reduce the critical queues in approaches 3 and 4, the queue reductions in approaches 1 and 2 are still significant evidence to support the clear superiority of the VA strategy. However, it is difficult to support that a better suited fixed time program could handle the queues of approaches 3 and 4 due to the following reason: It can be observed that the confidence intervals for the average queues of approaches 3 and 4 under fixed-time control are quite high, meaning that in some simulation runs the respective queues did not actually become very large.

Table 7. Average and maximum queue lengths (m) per intersection approach (Scenario 1)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	19,49	2,32	1,52	1,09	20,00	1,96	1,40	1,00	69,09	789,35	28,10	20,10	95,94	2431,84	49,31	35,28
Max. queue length [m]	76,60	269,38	16,41	11,74	84,10	144,54	12,02	8,60	266,50	3203,17	56,60	40,49	239,30	3473,79	58,94	42,16
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	14,25	0,37	0,61	0,43	15,45	0,58	0,76	0,54	37,77	16,09	4,01	2,87	28,26	9,05	3,01	2,15
Max. queue length [m]	60,20	41,96	6,48	4,63	70,70	86,23	9,29	6,64	190,50	1457,61	38,18	27,31	130,40	2799,16	52,91	37,85
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average queue length	-26,9%				-22,7%				-45,3%				-70,5%			
Maximum queue length	-21,4%				-15,9%				-28,5%				-45,5%			

This means that the formation of those queues is more dependent on the arrival process (seed) rather than on the average flow. In such a situation, only a dynamic control strategy which changes its decisions on a frequent time basis can efficiently control the queue lengths while even the best suited fixed time program cannot.

Table 8 presents the results concerning average vehicle delays, stopped delays and number of stops. The results are again presented per intersection approach so as to distinguish which parts of the network were improved more than others. It also reveals whether any parts of the fixed-time signal program were the cause of problems, that were eventually improved by the application of the new strategy. Again, the table shows the mean values for 10 simulation runs along with their confidence intervals.

The presented results show greater interest if they are read in conjunction with the results presented in table 7. It can be observed that the reductions in vehicle delays and stops on approaches 1 and 2 are analogous to the respective reductions yielded in the queue lengths. On the contrary, for approach 3, despite the fact that the average queue reduction was 45%, the delays are reduced around 25%. An observation that can be made here is that the queue forming on approach 3 under fixed-time control is mostly due to the insufficient green for vehicles turning left while vehicles moving ahead or turning right form significantly shorter queues as they are not much disturbed by the queue formed from the left turning vehicles.

On approach 4, a more significant reduction of the average vehicle delays and stops, is observed, which is closer to the respective queue length reductions. In this approach, in the fixed-time program, it can be observed that due to the insufficient green provided to the left turn, the queue not only expands further upstream but also disturbs vehicles moving forward and right. Therefore, avoidance of the queue forming in signal group 6 (left turn) results in a more concordant reduction in vehicle delays for the whole approach.

As was observed for the queues, also here, the confidence intervals for the performance measures under fixed time control, for approaches 3 and 4 are relatively high. This indicates that depending on the arrival process, the delays and stops can vary significantly when the fixed time program is implemented. On the contrary, under VA control the resulting performance measures are steadily lower, with significantly narrower confidence intervals.

Another observation that can be made is that the VA strategy results in narrower ranges for all performance measures. For example, in the fixed time program, the average delay per vehicle ranges from 33 (approach 1) to 52 (approach 4), while when VA is applied, the average delay time ranges from 25 (approach 1 and 4) to 37 (approach 3). The new strategy optimises a universal measure for the intersection (total waiting time) in every second and thus results in a more uniform distribution of the link costs. However, this is simply a choice that is left on the traffic engineer, and can be modified if needed. The strategy algorithm (and more specifically the cost estimation part in the optimisation module) can easily be modified so as to attribute different weights to different approaches, if for a special reason those needed to be favoured.

Table 8. Average delay, stopped delay and number of stops per vehicle, per intersection approach (Scenario 1)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	33,79	10,45	3,23	2,31	41,81	4,77	2,18	1,56	48,73	134,11	11,58	8,28	52,31	448,78	21,18	15,15
Average stopped delay [s]	25,80	7,72	2,78	1,99	33,30	3,57	1,89	1,35	34,49	69,46	8,33	5,96	32,69	115,66	10,75	7,69
Average number of stops	0,75	0,002	0,04	0,03	0,87	0,001	0,03	0,02	1,02	0,04	0,21	0,15	1,13	0,18	0,43	0,31
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	25,31	1,57	1,25	0,90	31,85	2,05	1,43	1,02	37,35	4,82	2,19	1,57	25,35	1,91	1,38	0,99
Average stopped delay [s]	18,25	1,18	1,08	0,78	24,33	1,77	1,33	0,95	25,88	2,83	1,68	1,20	16,12	1,12	1,06	0,76
Average number of stops	0,64	0,0004	0,02	0,01	0,76	0,0004	0,02	0,01	0,87	0,001	0,04	0,03	0,66	0,001	0,02	0,02
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average delay time [s]	-25,1%				-23,8%				-23,4%				-51,5%			
Average stopped delay [s]	-29,3%				-26,9%				-25,0%				-50,7%			
Average number of stops	-14,9%				-12,7%				-15,0%				-41,6%			

Table 9 presents the results concerning the fuel consumption over the intersection under the two control cases. The results were obtained by using the "node evaluation" feature of VISSIM. However, as mentioned in the previous section, the results (in terms of absolute values) cannot be considered reliable as the emission model used is not validated for the area under study and thus serves only for comparisons between control methods. Moreover, the reductions of the various measures (CO, NOx, VOC, fuel consumption) among control strategies were found to be same, therefore only the latter measure (fuel consumption) was maintained.

It can be observed that the average values obtained by the 10 simulation runs result in means that are quite steady (small confidence intervals) and the reduction under the VA control was found to be significant at a 99% level of significance.

It has been previously suggested (Chen and Yu, 2007, Zhang et. al., 2009) that signal control strategies significantly affect traffic emissions and that a strategy aiming at increasing traffic efficiency will not necessarily lead to a respective reduction in emissions. Efficiency-related objectives are difficult to correspond univocally to environmental objectives. In this project, due to a lack of a credible environmental model to determine emissions for the study area, no environmental measure was taken into account on the optimisation process. On that purpose, evaluating the new strategy in terms of emissions is of high importance.

A nearly 14% difference in the fuel consumption over the intersection (table 9) shows that the new VA strategy can significantly reduce the environmental impact of traffic compared to traditional fixed-time control.

Although it is difficult to attribute the reduction of fuel consumption to a certain efficiency-related improvement, it can be reasoned that, since the vehicle fleet in the examined network consists mostly of passenger cars, with a typical powertrain (no hybrid, or electric, or hydrogen-powered etc vehicles), the significant reduction of the number of stops contributes significantly to the reduction of the fuel consumption, as the most fuel-consuming action of the vehicle movement takes place during accelerations and decelerations.

Table 9. Fuel consumption over the intersection under fixed-time and VA strategies (Scenario 1)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	195,03	203,27	172,57	176,35	184,36	193,83	172,99	204,60	225,47	173,85	**190,23**	**306,54**	**17,51**	**12,52**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	163,42	165,17	161,27	163,14	162,30	162,95	161,65	164,42	167,08	162,66	**163,41**	**3,04**	**1,74**	**1,25**
Difference											**-14,1%**			

5.3.2.2 Scenario 2 – mid-day

This section presents the results from the application of the new strategy under the second demand scenario, which concerns traffic operations between the morning and evening periods. In this scenario, traffic flows appear to be higher in all directions and thus it does not correspond to an "entrance" or "exit" from the city centre. The results appear to be different from those presented for the first scenario, unveiling new evidence about the behaviour of the new strategy.

From table 10, it can be observed that a significant reduction in all performance measures is achieved when applying the new strategy in the second demand scenario. The reductions are analogous to those observed for the morning scenario. Moreover, the resulting performance measures vary less under VA control compared to fixed-time control, between simulation runs, showing that the new strategy is better adapting to variations in the arrival process and keeps producing equivalent results under different seeds.

As mentioned above, in the second demand scenario, the intersection operates under more intense traffic conditions, therefore, the performance measures for each control method are increased, compared to the first scenario. However, it is observed that the inter-scenario increase in the performance measures is lower for VA control compared to fixed time control. More specifically, in fixed time control, the average vehicle delay increased from the first to the second scenario by 16,3%, while in VA control the respective increase was only 12,5%. Concerning the number of stops, the difference was even higher, as the increase between scenarios in fixed-time control was 16,3% while in VA control the increase was only 5,8%. This implies that the sensitivity of the new strategy in flow variations is lower than for fixed-time control.

Table 10 provides the trends concerning the intersection as a whole; however, these trends vary significantly when observing the disaggregate results, per intersection approach. This table only provides an indication for the overall performance of the strategy; while, to study its impact more closely, there is a need to examine the results presented in tables 11 and 12.

Table 11 shows that despite the overall improvement in the intersection performance measures, the results per approach do not always follow the same trend. More specifically, while both the average and maximum queues are significantly reduced in approaches 1 and 4, the picture concerning approaches 2 and 3 is not so clear. For approach 2, a slight increase in the maximum queue can be observed which; however, is not statistically significant. The respective average queue for this approach did not change at all. For approach 3, a non-statistically significant increase in the maximum queue appears, along with a statistically significant increase in the average queue.

The confidence intervals for all the means concerning average queues are relatively narrow (except for approach 4 under fixed-time control). The confidence interval for the maximum queue of approach 4 is narrow due to the fact that this queue was frequently reaching the end of the upstream link, therefore longer queue lengths could not be recorded.

Table 10. Average delay, nr of stops, speed and stopped delay per vehicle, for the intersection (Scenario 2)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	47,40	56,78	34,91	41,66	54,93	50,84	48,53	57,88	57,62	41,62	**49,22**	**62,38**	**7,90**	**5,65**
Average number of stops per vehicle	0,97	1,17	0,71	0,86	1,13	1,04	0,98	1,17	1,16	0,83	**1,00**	**0,03**	**0,16**	**0,12**
Average speed [km/h]	20,85	18,73	24,58	22,41	19,10	19,99	20,57	18,50	18,54	22,44	**20,57**	**4,14**	**2,04**	**1,46**
Average stopped delay per vehicle [s]	30,44	34,97	23,54	27,35	33,89	31,96	31,24	36,32	36,20	28,26	**31,42**	**17,27**	**4,16**	**2,97**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	35,36	35,53	31,91	33,69	33,85	33,60	32,66	35,57	34,04	34,73	**34,09**	**1,51**	**1,23**	**0,88**
Average number of stops per vehicle	0,74	0,75	0,69	0,71	0,71	0,71	0,70	0,74	0,71	0,73	**0,72**	**0,0004**	**0,02**	**0,01**
Average speed [km/h]	24,43	24,36	25,68	25,01	24,97	25,02	25,40	24,36	24,88	24,66	**24,88**	**0,20**	**0,44**	**0,32**
Average stopped delay per vehicle [s]	23,40	23,47	20,80	22,23	22,30	22,14	21,43	23,62	22,63	22,83	**22,48**	**0,82**	**0,91**	**0,65**
Difference											**diff.**			
Average delay time per vehicle [s]											**-30,7%**			
Average number of stops per vehicle											**-28,2%**			
Average speed [km/h]											**20,9%**			
Average stopped delay per vehicle [s]											**-28,4%**			

Table 11. Average and maximum queue lengths (m) per intersection approach (Scenario 2)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	31,41	80,34	8,96	6,41	15,81	0,67	0,82	0,58	47,65	51,90	7,20	5,15	161,20	2999,92	54,77	39,18
Max. queue length [m]	147,50	6252,50	79,07	56,57	67,60	70,93	8,42	6,02	219,60	1112,04	33,35	23,86	284,80	11,29	3,36	2,40
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	16,07	0,36	0,60	0,43	15,807	0,88	0,94	0,67	57,32	60,14	7,76	5,55	36,99	22,95	4,79	3,43
Max. queue length [m]	63,50	111,61	10,56	7,56	70,50	64,28	8,02	5,74	241,00	1346,67	36,70	26,25	147,60	703,82	26,53	18,98
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average queue length	-48,8%				-0,02%	*			20,3%				-77,1%			
Maximum queue length	-56,9%				4,3%	*			9,7%	*			-48,2%			

** result not statistically significant*

In fixed time control, average queues for approaches 1-3 appear to be significantly shorter than for approach 4. Therefore, an apparent imbalance in the link costs under fixed-time control can be observed. The new strategy, due to the use of a universal cost measure that increases the cost of a vehicle waiting on a queue exponentially with time, does not allow for such imbalances to occur. Obviously, a queue which is not dissipated after one cycle, imposes a higher cost than a queue that has just formed, due to the exponential rise of the cost from the vehicles waiting there. Therefore, the strategy results in average queue estimates that show a smaller range.

The VA strategy eventually results in increasing the average queue on approach 3 by 20%, compared to fixed-time control, so as to achieve an overall reduction in the performance measure and to manage to produce more balanced queues along the intersection. Indeed, the range of the average queues between approaches under VA control is much narrower than for fixed-time control.

It can be assumed that, if the horizon of the strategy was longer, the results might have been even more balanced, because the strategy would have been able to foresee more choices (paths) for the next horizon, in every time step. In the case that was applied, the resulting cycles (the strategy is rolling-horizon and acyclic, but in the resulting signal timings, a cycle can be identified) might have been greater than 80 seconds, which means that, just after having served a specific stage, it was difficult for the strategy to foresee into the future, into serving that stage again. For that reason, the only paths that could extend from that time instant to the same stage again, would have to include small stage durations for the in-between stages, and those paths would most probably be sub-optimal. However, with a greater horizon, paths that would include sufficient durations for the next stages, as well as for the stage that has just been terminated (for the next time it would be called) might have been more optimal. In this case, the decisions for extending or terminating the current stage would be more informed.

Moreover, in the specific application of the strategy, the stages that serve approach 3, are stage 4 and stage 5, while the stage that serves approach 4 is stage 1. This means that (due to the fixed stage sequence) approach 4 is always served after approach 3. Approach 4 has the most critical movement in this scenario (left turn), as revealed also from the fixed-time control application, therefore, the VA strategy would tend to provide a great extension to the respective stage. However, due to the limited horizon, a long extension in stage 1 would result in difficulty to "see" into the future into serving approach 3 again. This might have been a contributory factor for approach 3 having longer average queues than the other approaches: the strategy might have been slightly **"myopic"** for a few seconds, every time after approach 3 was served. A non-fixed stage sequence (which is easily realisable with the developed strategy) could contribute to mitigate this effect (although it would also impose the need to adapt an even shorter horizon, due to the increased computational effort needed – for more information see chapter 4).

Table 12. Average delay, stopped delay and number of stops per vehicle, per intersection approach (Scenario 2)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	54,24	176,27	13,28	9,50	35,40	3,20	1,79	1,28	36,82	7,85	2,80	2,00	82,73	682,32	26,12	18,69
Average stopped delay [s]	42,89	114,30	10,69	7,65	27,85	2,57	1,60	1,15	24,96	4,44	2,11	1,51	49,90	169,10	13,00	9,30
Average number of stops	1,02	0,036	0,19	0,14	0,78	0,001	0,02	0,02	0,83	0,002	0,05	0,03	1,76	0,30	0,55	0,39
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	28,32	1,16	1,07	0,77	34,98	2,88	1,70	1,21	45,69	15,03	3,88	2,77	29,27	5,00	2,24	1,60
Average stopped delay [s]	20,98	0,88	0,94	0,67	27,50	2,24	1,50	1,07	31,76	9,15	3,03	2,16	19,84	3,19	1,79	1,28
Average number of stops	0,66	0,0003	0,02	0,01	0,78	0,0004	0,02	0,01	0,98	0,003	0,05	0,04	0,70	0,001	0,04	0,03
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average delay time [s]	-47,8%				-1,2%	*			24,1%				-64,6%			
Average stopped delay [s]	-51,1%				-1,3%	*			27,2%				-60,2%			
Average number of stops	-34,9%				0,0%	*			17,6%				-60,4%			

** result not statistically significant*

Table 12 shows the results concerning vehicle delays and stops per intersection approach, under the second demand scenario. At a first glance, the conclusions drawn by reading the respective results for the queues can be confirmed by the figures presented in this table.

It can be observed that the increased delays and stops on approaches 1 and 4 under fixed-time control are significantly reduced, resulting; however, in an increase in vehicle delays and stops on approach 3. Marginal differences in the delays are observed on approach 2 which are not statistically significant. The resulting delays under the VA control, once again, have a narrower range between approaches, resulting in more balanced cost distribution along the intersection.

More specifically, the average vehicle delay on approach 3 is increased by 9 seconds, while vehicle delay on approaches 1 and 4 is reduced by 26 and 53 seconds respectively. The magnitude of the differences is the same as far as the average stopped delay is concerned. The increase in the delays and stops on approach 3, comes as a result of the increase (compared to the fixed-time plan) in the average green durations of the signal groups on approaches 1 and 4.

Concerning the inter-scenario differences for the VA strategy, the increase on the delays and stops from scenario 1 to scenario 2, ranges from 10-15% for approaches 1,2 and 4 and is around 20% for approach 3. Traffic flows from the first to the second scenario, slightly decrease for approaches 1,2 and 4 and increase for approach 3. Despite that fact, the VA strategy does not result only in an increased delay on approach 3 and decreased delay on approaches 1,2 and 4 but allocates green time in a way such that all approaches will receive a part of the additional burden imposed by the flow increase on approach 3.

Table 4 presents the results concerning fuel consumption over the intersection for the second demand scenario. Fuel consumption is reduced by 22% when the VA strategy is applied, compared to the fixed-time plan. The mean resulting from VA control has a smaller confidence interval, showing thus less sensitivity on the arrival process.

Finally, it is interesting to note that although fuel consumption under fixed-time control, rises by 13,7% from the first to the second scenario, the respective increase under VA control is only 2,5%. This reveals that the flow changes in the network affect fuel consumption less, when VA control is applied.

Table 13. Fuel consumption over the intersection under fixed-time and VA strategies (Scenario 2)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	207,55	236,54	177,92	200,99	233,44	220,32	214,16	237,38	241,78	191,27	**216,14**	**469,78**	**21,67**	**15,50**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	166,88	169,44	164,25	168,67	167,17	165,62	166,27	171,75	171,18	166,5	**167,77**	**5,90**	**2,43**	**1,74**
Difference											**-22,4%**			

5.3.2.3 Scenario 3 – evening

The results from the third demand scenario concerning the evening peak period are presented in aggregate form on table 14. This scenario has a different combination of flows, with the overall flow in the network maintained at high levels.

In this scenario, the reduction in all performance measures from the application of VA control appears to be lower, compared to the reduction observed in the previous scenarios. The average vehicle delay is reduced almost by 14%, while the average number of stops is decreased by 12%, the average speed is increased by some 7% and the average stopped delay is dropped by almost 13%. A closer look at each intersection approach reveals that contradicting trends can be identified. Apart from the benefits brought to the intersection performance, table 14 shows that the variations in the performance measures between simulation runs are smaller when VA control is applied, proving once more, the small sensitivity of the strategy to variations in the arrival process.

For approaches 1,2 and 3 and for fixed-time control, the average queue lengths appear to be lower compared to the previous scenarios, meaning that the green allocation in this program is such that it prevents long queues from being formed. However, the fixed-time plan, does not seem to be well suited as far as approach 4 is concerned, as the average queue length appears to be significantly higher. This effect, is of course mitigated by the application of the VA strategy.

What needs to be highlighted for the VA strategy, is the effect of slightly myopic control that was also observed in the second scenario. It appears that, the queue in approach 3 is quite longer than for the rest of the approaches. This could be explained if a significantly higher flow was present in this approach which could not be served properly. However, this is not the case, therefore another reason is more likely to contribute. The fact that approach 4 has the highest flows in the third demand scenario, and that, the stages that serve the traffic movements of approach 4 are called just after the stages that serve approach 3 is likely to cause this effect.

When the stages that serve approach 3 are running, each second, the strategy “sees” 80 seconds forward therefore, recognises the quickly forming queue on approach 4 and tends to pass to the respective stages quickly. However, when this is done, the maximum green is likely to be attributed to these stages, therefore, it is difficult to reach the time instant where approach 3 would be served again, with the 80 second horizon. Therefore, for some time, the strategy “ingores” the forming queue on approach 3. Until it is able to take those queues into account in the optimisation, some time is already wasted, therefore, the resulting queue appears to be a bit unbalanced with the rest. The possible solution, which was also mentioned before, is the adoption of a longer optimisation horizon or a non-fixed stage sequence.

Table 14. Average delay, nr of stops, speed and stopped delay per vehicle, for the intersection (Scenario 3)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	39,82	49,74	33,92	35,86	38,74	41,80	38,53	53,77	55,00	34,65	**42,18**	**61,26**	**7,83**	**5,60**
Average number of stops per vehicle	0,80	1,00	0,69	0,72	0,77	0,82	0,75	1,06	1,10	0,70	**0,84**	**0,02**	**0,15**	**0,11**
Average speed [km/h]	22,80	20,12	24,78	24,11	23,17	22,22	23,22	19,21	18,95	24,55	**22,31**	**4,65**	**2,16**	**1,54**
Average stopped delay per vehicle [s]	26,59	31,99	23,37	24,59	26,49	28,13	26,93	35,06	34,97	23,89	**28,20**	**18,80**	**4,34**	**3,10**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	37,37	38,88	34,45	34,94	37,73	36,09	36,18	36,19	36,38	35,29	**36,35**	**1,80**	**1,34**	**0,96**
Average number of stops per vehicle	0,75	0,78	0,71	0,72	0,77	0,73	0,74	0,73	0,74	0,72	**0,74**	**0,0005**	**0,02**	**0,02**
Average speed [km/h]	23,60	23,12	24,58	24,43	23,48	24,01	24,00	24,00	23,91	24,32	**23,94**	**0,20**	**0,45**	**0,32**
Average stopped delay per vehicle [s]	25,42	26,38	23,17	23,43	25,53	24,38	24,42	24,53	24,67	23,80	**24,57**	**0,99**	**0,99**	**0,71**
Difference											**diff.**			
Average delay time per vehicle [s]											**-13,8%**			
Average number of stops per vehicle											**-12,3%**			
Average speed [km/h]											**7,3%**			
Average stopped delay per vehicle [s]											**-12,9%**			

Table 15. Average and maximum queue lengths (m) per intersection approach (Scenario 3)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	30,97	90,16	9,50	6,79	36,29	36,76	6,06	4,34	31,79	5,44	2,33	1,67	105,82	3042,73	55,16	39,46
Max. queue length [m]	152,70	6724,01	82,00	58,66	116,60	256,27	16,01	11,45	156,40	361,82	19,02	13,61	248,80	2637,51	51,36	36,74
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	16,44	0,60	0,78	0,55	20,32	2,18	1,48	1,06	59,81	35,92	5,99	4,29	46,42	99,71	9,99	7,14
Max. queue length [m]	64,50	30,94	5,56	3,98	79,80	83,73	9,15	6,55	213,90	615,21	24,80	17,74	211,70	4648,90	68,18	48,78
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average queue length	-46,9%				-44,0%				88,1%				-56,1%			
Maximum queue length	-57,8%				-31,6%				36,8%				-14,9%			

Table 16. Average delay, stopped delay and number of stops per vehicle, per intersection approach (Scenario 3)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	54,17	215,89	14,69	10,51	66,27	97,92	9,90	7,08	30,37	1,72	1,31	0,94	55,61	537,99	23,19	16,59
Average stopped delay [s]	42,82	142,49	11,94	8,54	53,96	68,33	8,27	5,91	20,33	0,91	0,96	0,68	35,11	146,84	12,12	8,67
Average number of stops	1,02	0,043	0,21	0,15	1,21	0,02	0,14	0,10	0,73	0,0004	0,02	0,01	1,20	0,23	0,48	0,34
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	29,15	1,25	1,12	0,80	37,95	3,23	1,80	1,29	50,81	8,53	2,92	2,09	33,26	19,12	4,37	3,13
Average stopped delay [s]	21,83	1,08	1,04	0,74	29,82	2,57	1,60	1,15	36,27	4,77	2,18	1,56	22,69	10,05	3,17	2,27
Average number of stops	0,67	0,0002	0,01	0,01	0,82	0,0006	0,02	0,02	1,04	0,002	0,05	0,03	0,76	0,006	0,08	0,06
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average delay time [s]	-46,2%				-42,7%				67,3%				-40,2%			
Average stopped delay [s]	-49,0%				-44,7%				78,4%				-35,4%			
Average number of stops	-34,5%				-31,9%				43,4%				-37,0%			

However, the flexibility of the proposed strategy in choosing length of the time horizon and the fixed (or not) stage sequence, according to the situation under study and the available computational power, can be seen as an advantage of the proposed approach against other VA strategies.

The results concerning the average vehicle delay, number of stops and stopped delay, per approach, show that, despite the increased values of the performance measures on approach 3, the significant reductions in the rest of the approaches result to an overall improvement on the intersection performance. However, it can be observed that the overall 14% reduction on the average vehicle delay and the 13% reduction on the average stopped delay result from non-homogenous increases/decreases per approach. The average vehicle delay decreased for approaches 1,2 and 4 from 40-46% while the respective delay increased for approach 3 some 67%.

However, even after this significant increase, the absolute value of the delay on approach 3, remains lower than the resulting delays on approaches 1,2 and 4, under the fixed-time program. Moreover, despite the increase in the performance measures on approach 3 and the decrease on approaches 1,2 and 4, the range of all performance measures among the approaches remains narrower when the VA strategy is applied, compared to the fixed-time control.

Finally, by observing the results of table 17, concerning the fuel consumption over the intersection, a decrease of the magnitude of 16% shows that the VA strategy results in more environmental friendly control. Certainly, the results per approach would be expected to differ, following the trends shown in tables 10 and 11; however, in environmental terms, only the overall result for the intersection is significant.

Table 17. Fuel consumption over the intersection under fixed-time and VA strategies (Scenario 3)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	202,45	230,79	187,39	195,28	202,26	210,19	199,61	245,21	251,7	190,24	**211,51**	**525,69**	**22,93**	**16,40**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	177,22	186,34	175,44	176,09	182,47	178,97	175,32	176,32	179,77	174,25	**178,22**	**14,2231**	**3,77**	**2,70**
Difference											**-15,7%**			

5.3.2.4 Scenario 4 – night

The fourth demand scenario examines the operation of the intersection during night-time. This scenario is significantly different from the rest, as traffic flows are very low and no significant delays are observed either under fixed-time or VA control. No (or small) queues are formed, while the absolute values also for the rest of the performance measures are not high. Parked vehicles do not exist in the road-side thus the reduced speed areas of the previous scenarios are omitted here.

In this case, it is more interesting to examine measures that have a high impact to user acceptance rather than efficiency. For example, the number of stops can be considered as more important than the vehicle delay, as the latter will anyway be low enough, therefore a further reduction can be indifferent for the road user. On the contrary, the stops can be more important to the road user, as it is likely that he/she will consider that since there is no significant traffic over the intersection, he/she should travel without stopping.

In scenario 4, the VA strategy operates for a significant part of the time under a minimum cycle (without giving extensions to any stages) . This results to a 55 second cycle, consisting only of the stage transitions (which satisfy the minimum greens). Extensions are observed only in cases where more vehicles are approaching simultaneously on one approach, to allow them to pass the intersection without stopping. However, platoons are not common at night, therefore extensions are rarely given.

This means that, for a large proportion of the simulation time, the VA program operates similarly to a fixed-time control program, but with a smaller cycle. This also implies that the 90-cycle used in the fixed-time program is probably higher that needed for the night-time traffic conditions.

By observing table 18 it can be seen that the number of stops is marginally increased in the VA strategy (however, this is not statistically significant). On the contrary by observing the number of stops per approach it can be seen that, for all approaches, the average number of stops is decreased by at least a 15% (except for approach 3). This results seems contradicting, however, it is explained by the fact that the average nr. of stops per approach is estimated only for pre-specified travel time sections while the other measure is estimated for the whole intersection. The travel time sections stretch from each approach stop-line till the upstream end of the link. However, vehicles reaching the stop-line are considered to have reached the end of the travel time section, therefore are withdrawn from the calculation. Thus, the number of stops per approach, is an estimate referring to the vehicles stopping before the stop-line. This means that the estimate referring to the number of stops per approach, in fact does not include the first vehicles of a queue, but only counts from the second vehicle and on.

Table 18. Average delay, nr of stops, speed and stopped delay per vehicle, for the intersection (Scenario 4)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	average	var.	st.dev.	conf.
Average delay time per vehicle [s]	19,556	20,186	20,677	19,998	20,679	19,49	19,772	19,788	19,874	20,118	**20,01**	**0,17**	**0,41**	**0,30**
Average number of stops per vehicle	0,477	0,484	0,495	0,478	0,478	0,473	0,479	0,478	0,48	0,482	**0,48**	**0,00003**	**0,01**	**0,004**
Average speed [km/h]	36,075	35,657	35,323	35,78	35,322	36,165	35,99	35,956	35,856	35,768	**35,79**	**0,08**	**0,29**	**0,21**
Average stopped delay per vehicle [s]	14,477	14,961	15,277	14,837	15,488	14,414	14,588	14,687	14,683	14,878	**14,83**	**0,12**	**0,34**	**0,24**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Average delay time per vehicle [s]	15,868	15,89	15,777	15,516	15,641	15,761	15,219	15,772	15,51	15,909	**15,69**	**0,05**	**0,22**	**0,16**
Average number of stops per vehicle	0,478	0,492	0,494	0,487	0,481	0,487	0,474	0,482	0,49	0,49	**0,49**	**0,00004**	**0,01**	**0,005**
Average speed [km/h]	39,165	39,186	39,328	39,518	39,459	39,285	39,824	39,312	39,518	39,24	**39,38**	**0,04**	**0,20**	**0,14**
Average stopped delay per vehicle [s]	10,613	10,494	10,318	10,094	10,249	10,434	9,907	10,499	10,025	10,53	**10,32**	**0,06**	**0,24**	**0,17**
Difference											**diff.**			
Average delay time per vehicle [s]											**-21,6%**			
Average number of stops per vehicle											**1,1%**	*****		
Average speed [km/h]											**10,0%**			
Average stopped delay per vehicle [s]											**-30,4%**			

** result not statistically significant*

This means that, if all vehicles are considered, the number of stops does not change significantly between fixed-time and VA, but if the vehicles in queue (omitting the first vehicle in each row) are considered, then the number of stops is lower. This is reasonable, as the average queues are lower in the VA scenario. The impact of this effect is hardly observed in the previous scenarios, as the majority of stops occurs before the stop line, therefore, even if the first vehicles of the queue are omitted, no significant error occurs. However, in this scenario, most stops occur at the stop line, as no queues are formed therefore the impact of this effect is higher.

As can be observed in table 19, the average queues are almost halved for approaches 1,2 and 4 under VA control, a result probably arising from the smaller cycle. The maximum queues are reduced less; however, not all reductions are statistically significant. In any case, when the absolute values of the queues are small, it is more difficult to obtain reliable averages as the role of the arrival process to the queue formation is more important.

In the fourth demand scenario, due to the low traffic flows, the cycle time will most probably never become larger than 80 seconds (thus will always be smaller than the horizon). This means that even with a shorter time horizon, signal control can be equally efficient. The adoption of a smaller horizon (thus less time steps) simplifies the optimisation algorithm and can theoretically allow for the adoption of a non-fixed stage sequence, which could result in significantly lower performance measures.

As expected, the reduction in the fuel consumption under the fourth scenario is not as significant as for the previous cases; an average 7% reduction (which however, is statistically significant) under the VA strategy is recorded.

Table 19. Average and maximum queue lengths (m) per intersection approach (Scenario 4)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	4,40	0,07	0,27	0,19	2,96	0,06	0,24	0,17	8,04	0,05	0,22	0,15	6,76	0,19	0,43	0,31
Max. queue length [m]	35,40	32,04	5,66	4,05	36,90	53,43	7,31	5,23	45,30	46,23	6,80	4,86	49,30	27,34	5,23	3,74
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Aver. queue length [m]	2,17	0,04	0,21	0,15	1,62	0,03	0,17	0,12	6,58	0,03	0,18	0,13	3,81	0,05	0,22	0,16
Max. queue length [m]	27,30	43,57	6,60	4,72	32,30	26,01	5,10	3,65	41,90	16,32	4,04	2,89	35,80	7,07	2,66	1,90
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average queue length	-50,7%				-45,3%				-18,2%				-43,7%			
Maximum queue length	-22,9%				-12,5%	*			-7,5%	*			-27,4%			

** result not statistically significant*

Table 20. Average delay, stopped delay and number of stops per vehicle, per intersection approach (Scenario 4)

	Approach 1				Approach 2				Approach 3				Approach 4			
Fixed time	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	14,00	0,76	0,87	0,62	15,93	1,51	1,23	0,88	17,86	0,38	0,61	0,44	13,07	0,32	0,57	0,41
Average stopped delay [s]	9,29	0,63	0,79	0,57	11,38	1,12	1,06	0,76	12,34	0,30	0,55	0,39	7,93	0,27	0,52	0,37
Average number of stops	0,34	0,0002	0,02	0,01	0,40	0,001	0,03	0,02	0,46	0,0002	0,01	0,01	0,36	0,0001	0,01	0,01
VA	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**	**average**	**var.**	**st.dev.**	**conf.**
Average delay time [s]	8,40	0,38	0,61	0,44	9,76	1,02	1,01	0,72	14,99	0,33	0,57	0,41	9,33	0,26	0,51	0,36
Average stopped delay [s]	4,14	0,27	0,52	0,37	5,54	0,75	0,86	0,62	9,12	0,30	0,55	0,39	4,24	0,13	0,36	0,26
Average number of stops	0,24	0,0005	0,02	0,02	0,31	0,0011	0,03	0,02	0,45	0,0002	0,01	0,01	0,30	0,0003	0,02	0,01
Difference	**diff.**				**diff.**				**diff.**				**diff.**			
Average delay time [s]	**-40,0%**				**-38,7%**				**-16,1%**				**-28,6%**			
Average stopped delay [s]	**-55,4%**				**-51,3%**				**-26,1%**				**-46,5%**			
Average number of stops	**-28,9%**				**-21,8%**				**-2,4%**	*****			**-15,8%**			

** result not statistically significant*

Table 21. Fuel consumption over the intersection under fixed-time and VA strategies (Scenario 4)

Fixed time	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	52,56	52,36	53,97	52,53	53,32	52,06	52,01	52,41	52,59	52,74	**52,66**	**0,35**	**0,59**	**0,42**
VA	run1	run2	run3	run4	run5	run6	run7	run8	run9	run10	**average**	**var.**	**st.dev.**	**conf.**
Fuel Consumption [gal]	49,42	48,78	49,79	48,66	49,16	48,64	48,32	48,89	48,79	49,3	**48,98**	**0,1913**	**0,44**	**0,31**
Difference											**-7,0%**			

5.3.3 Robustness

Road network robustness is the insusceptibility of a road network to disturbing incidents, and can be understood as the opposite of network vulnerability. In other words, road network robustness is the ability of a road network to continue to operate correctly across a wide range of operational conditions (Li, 2008). It can be easily understood that for the case of a signalised intersection or arterial, the network robustness strongly depends on the signal control strategy applied. The aim of this section is to study the robustness of the examined intersection under the new VA strategy.

The operation status of the road network is usually evaluated with some indicators of its network-level performance, such as average speed and network throughput (Li, 2008). Immers et al (2004) have also defined robustness and they subdivide it into the following four corrective measures that enhance the robustness of the transportation system: redundancy, interdependency, flexibility and resilience. The redundancy of a system may be improved by introducing a certain amount of redundancy or spare capacity into the system. With interdependency the following is meant: congestion at a centrally located link or node may cause a series of cascading failures disrupting traffic on large parts of the network. The robustness of a network increases if the interdependencies decrease. Resilience is the capability of the transport system to repeatedly recover, preferably within a short time period, from a temporary overload. Finally, flexibility is the extent in which the system is able to carry out more and other functions than it was originally designed for. These four aspects together give a more comprehensive understanding of robustness.

In the case of an unexpected disruption, it is important for a road network first to maintain its function as much as possible after the disruption; and to recover its function as quick as possible from the partial or complete failure. In this section, the network robustness when operating under the new VA strategy is studied. The aim is not to estimate robustness using a series of quantitative indicators but to identify and discuss the features of the strategy algorithm that affect robustness, analyse the impact of various incidents in the network operation under the new strategy and draw conclusions based on certain evidence.

The concept of network robustness is often confused with the concept of network reliability. However, reliability focuses on analyzing the probability of a road network performing its proposed service level adequately taking into account the uncertainties of the circumstances, while network robustness emphasizes particularly on the ability of a road network functioning properly facing unpredictable and exceptional incidents. An analysis of the network reliability under the new VA strategy was made in sections 5.2.2.1-5.2.2.4 where its behaviour under different demand ranges was tested.

Network robustness is a relatively new concept and thus it has not been extensively studied. Therefore, quantifying network robustness is not an easy task. Moreover, a wide variety of operational deficiencies might occur, having different impacts on the network operation.

What is done in this chapter for the assessment of the network robustness, is an examination, on a theoretical level, of a wide range of operational deficiencies that can occur, so as to identify the impact on the VA control, of the limitations that will occur. For example, the reduction of capacity for a specific part of the network is studied by assessing its impact on the algorithmic calculations that can cause strategy deficiencies. Apart from this theoretical assessment, a practical example is implemented in VISSIM to retrieve a set of results related to the network robustness.

Hardware failures

A first category of unexpected incidents that can cause serious disruptions in the normal network operation, is the deficiency of control hardware parts. The main hardware parts included in the application of the VA strategy is the intersection controller, the inductive loop detectors and the cable network that expands under the surface and serves for power supply, data transfer etc. Obviously, the effect of deficiencies on the latter cannot be studied in this project as no real-world implementation takes place.

The strategy developed in this thesis is intended to operate in a decentralised way, meaning that each intersection controller can make its own decision about how to serve the approaching vehicles (this subject as well as the establishment of coordination are discussed in chapter 4). This means that, in a network operating under the VA strategy, a controller deficiency does not mean total control loss, but just control loss over one intersection. In such a case, the inevitable queues and changes in the output flows from the intersection will obviously affect neighbouring intersections. Other VA strategies (like RHODES) impose superior rules that serve for establishing coordination, which act as constraints for the optimisation performed locally. This means that if one controller is lost, the super-imposed coordination scheme might not be suitable any more as the one intersection controller will not be able to optimise traffic operations locally. Therefore, in this case, a problem can occur if these rules continue to act as constraints for the rest of the intersection controllers. On the contrary, in the strategy developed here, each controller is free to make its own decision without being subject to superior constraints, therefore, in the event of a deficient controller, other controllers can choose how to cope with the resulting traffic.

Another hardware deficiency that will affect the network operation under the new strategy is the partial or complete failure of one or more loop detectors. In the case of a detector failure, the effect varies depending on whether the deficient detector is upstream on the link or at the stop line. In the first case, a series of events takes place: firstly, vehicle passing from this detector are not projected downstream to the end of the queues therefore the Traffic Estimation module will provide incomplete arrival information for the short term horizon (for more information on the TE module see chapter 4). The long term arrivals will also be affected as they are based on the average actuations over the last minute. If more detectors which are still operational are present, the long term estimates will not be zero but will just be reduced. As a result, the queue growth for the signal groups in this approach will be slower therefore the actual queues will be larger than the estimated ones. This will result the strategy allocating insufficient green time to the respective stages. Furthermore, the respective queues are expected to grow until the occupancy of the remaining upstream detectors becomes larger than the critical value. At this point, the strategy will enable its gating feature and stop using the false traffic estimates. If the detector deficiency persists, it can be expected that the strategy will pass from normal operation, to the gating feature and back, (most probably switching from mode to mode every few cycles). This type of control is obviously not optimal; however, it is not expected to lead to a gridlock. In this respect, it is considered that the strategy can cope with a detector failure; however, a negative effect to the network operation is inevitable.

In the case of a stop-line detector failure, the effect differs. Stop-line detectors serve several purposes. Firstly, to determine how many vehicles exited the approach, secondly, to perform consistency checks for the queue estimates and thirdly, to provide data for the estimation of the turning percentages. If a detector fails, the queue estimate is only slightly affected, in an indirect way. The algorithmic estimate does not use the stop-line detector actuations directly but uses a standard departure rate to reduce the queues during green. However, these actuations serve for the check of consistency for the algorithmic queue estimate. Therefore, one stop-line detector less, means that the consistency checks for the queue in the respective lane cannot take place. From the simulations, it can be observed that the consistency checks quite frequently provide corrections for the queue estimates, especially for small queues. If other stop-line detectors exist and are operational, the effect is not expected to be severe; however, if all stop-line detectors fail, the queue estimate is soon expected to become unreliable, with effects similar to those described for the upstream-detector-failure case.

A more serious impact from a stop-line detector failure is related to the estimation of the turning percentages. The cumulative actuations during a pre-defined period are used to estimate the turning percentages per traffic movement. If a detector fails, the respective sum will be zero, therefore the estimated turning percentage for some signal groups will be under-estimated while the rest will be over-estimated. This means that the queue growth for the signal groups will follow a similar trend. As a result, despite the fact that the total approach queue will be correct, the allocation of the queue per signal group will be affected. If the affected signal groups belong to the same stage, the impact will not be so severe, as the maximum from the estimated queues will be decisive for the green time allocated. However, this will be longer than the actual, therefore excessive green will be provided to that stage. If the affected signal groups belong to different stages, one stage will receive excessive green and the other will receive insufficient green. In conclusion, stop-line detector failures, at the worst case, mean inefficient green allocation for one or more stages but does not mean a total control loss of a serious interruption of traffic operations.

Capacity drops

A second category of incidents includes sudden and unexpected capacity drops. That can be caused i.e. by road accidents or vehicles breaking down and can result in partial or total closure of a link. This category can further be divided into: capacity drops within the intersection conflict areas (i.e. in the middle of the intersection) or in locations upstream of the stop-line of a link.

In general, when such an incident occurs, the effect in traffic operations depends on the prevailing traffic flows. If traffic is heavy, the sudden capacity drop will create a shockwave that will travel upstream. Vehicles will not be able to clear the intersection in time and (depending on the intersection capacity and demand) queues can continue rising even after the incident has ended. It is therefore up to the control strategy how efficiently to cope with such capacity drops.

A capacity drop that occurs in the middle of the intersection has a more serious impact on the correct operation of the VA strategy as it affects more than one traffic movements. The strategy estimates many parameters in real time (i.e. arrivals, turning percentages, etc) but also uses fixed values for some other (i.e. departure rate from a stop line). This means that, if for a reason, the value for one of these parameters changes, the strategy will optimise using inaccurate data. The practical example presented later on quantifies this effect through an experiment in VISSIM, including a sudden capacity drop in the middle of the intersection.

Demand

The previous two categories of disruptions had a direct effect on the network supply (capacity) as they affected the signal program. However, as a result of an incident at a location upstream from the examined intersection, changes in the traffic demand can also occur. For example, drivers may massively re-route after an incident occurring before the upstream intersection, especially if real-time traffic information is available through VMS, handheld devices etc. This category can include sudden big changes in the network demand. The flow on a specific link can become very low while the flows on other links might increase.

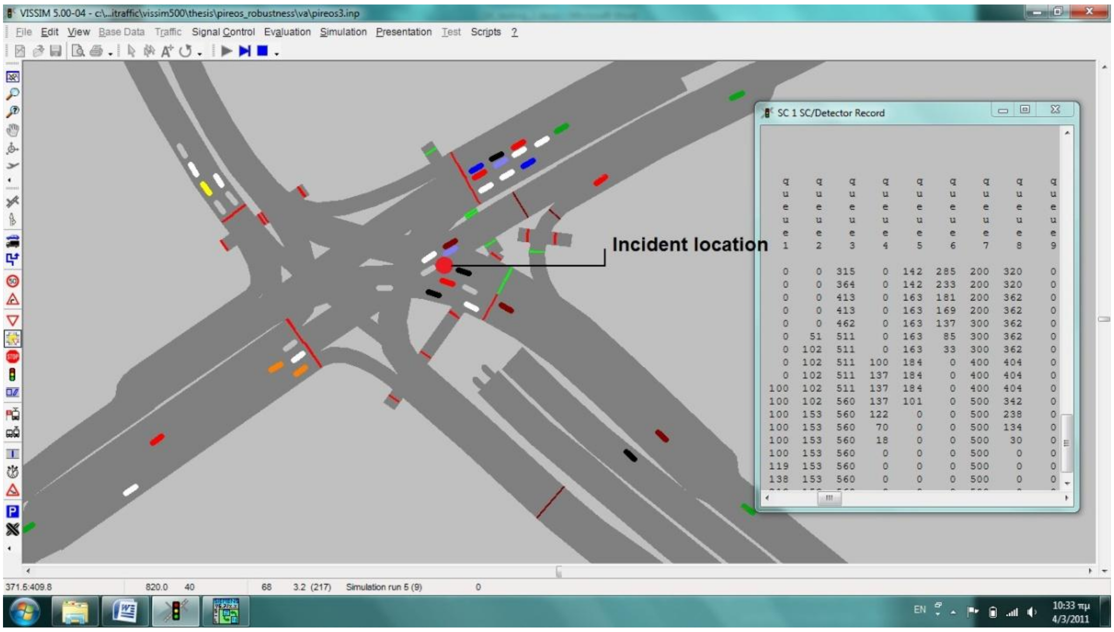
It can be considered, that the network will be robust under the new VA strategy when such demand changes occur, because the strategy, due to its rolling horizon nature, is able to change signal timings very fast. The strategy re-evaluates its decisions per second, using traffic data that are also updated per second, therefore can respond instantly to sudden flow changes, much quicker than other VA strategies, especially reactive-ones, that change their decisions per cycle, or those using smoothed average estimates obtained less frequently.

However, a problem that can occur in reality in this case, is that, when the traffic flow on a specific link is dropped to zero for a large amount of time, the detectors on the respective approach can become idle. As an example, the LD4 detectors used in the Athens road network, become idle (go offline) if they remain unoccupied or become continuously occupied for 15 consecutive cycles. This is a fail-safe measure, to identify detectors that are malfunctioning. If this happens, the consequences will be similar to those discussed in hardware malfunction cases.

Practical example

In the following paragraphs, a practical example of robustness testing is presented. However it is reminded that this example concerns only a special case, when several different incidents with different effects could happen. Thus, the results presented for the robustness are indicative.

It is assumed that an accident happens in the middle of the intersection, between two vehicles, one coming from signal group 1 (straight movement from approach 4 – down left) and one from signal group 5 (straight movement from approach 1 – down right). As a result, vehicles cross this point with difficulty and with a very low speed. This is realised in VISSIM by introducing reduced speed areas in two link connectors, the one corresponding to the straight movement from approach 4 and the one corresponding to the straight movement from approach 1. It is assumed that the incident will last for 10 minutes (until the cars are removed from the intersection) from the 600th till the 1200th simulation second. The strategy reaction, the implications for the signal control and the effects during the incident and during the recovery period are examined.



Picture 6. Incident between signal groups 1 and 5 and resulting conflict

As a result of the incident, the vehicles cannot cross the shared space of the intersection quickly enough, therefore vehicles are being blocked within the conflict areas after the end of their green duration. Those vehicles block other vehicles that have just started crossing, therefore the intersection capacity drops and queues are formed.

The strategy algorithm is built in a way assuming that during the green time, the queues are dissipated at a standard rate; however, in this case this assumption does not hold. Therefore, the algorithm drops the queue estimates faster that they dissipate in reality. As a result, at the end of a stage, the algorithm might estimate that a queue has dissipated while in fact it has not. This causes the real queues to be constantly rising during the incident without the strategy being aware of it. As a further consequence, the projected arrival times and the eventual estimated arrivals are becoming unreliable. This further magnifies the error of the queue estimate, resulting on a self-magnifying error mechanism. This process continues until the queues are approaching the upstream detectors. At this point the strategy reacts well, as it stops estimating arrivals based on the standard procedure and starts providing the maximum green times to the approaches (for more information on when and how this process is initiated and ended, see chapter 4).

After the end of the incident, depending on how long the actual queues are, the strategy needs some time to restore queues and flows to the normal levels. If the occupancy of the upstream detectors on some approaches has become larger than the critical thresholds, the strategy will try to provide longer green durations to the respective stages. It appears that, after the incident has ended, the strategy drops the formed queues quickly, while a fixed-time program would not be able to cope with the excessive queues until the demand had dropped significantly. The following figures (7-12) present quantitative results concerning the queue evolution before, during and after the incident, showing both the growth and the decay phases due to the incident.

The results were obtained from 10 simulations using different seeds, the same as those used in the simulations presented in section 5.2.2.1. The average (from the 10 simulations) queue evolution, in 100 second time steps is recorded (for 3000 seconds in total) and is compared with the respective queue evolution when no incident takes place.

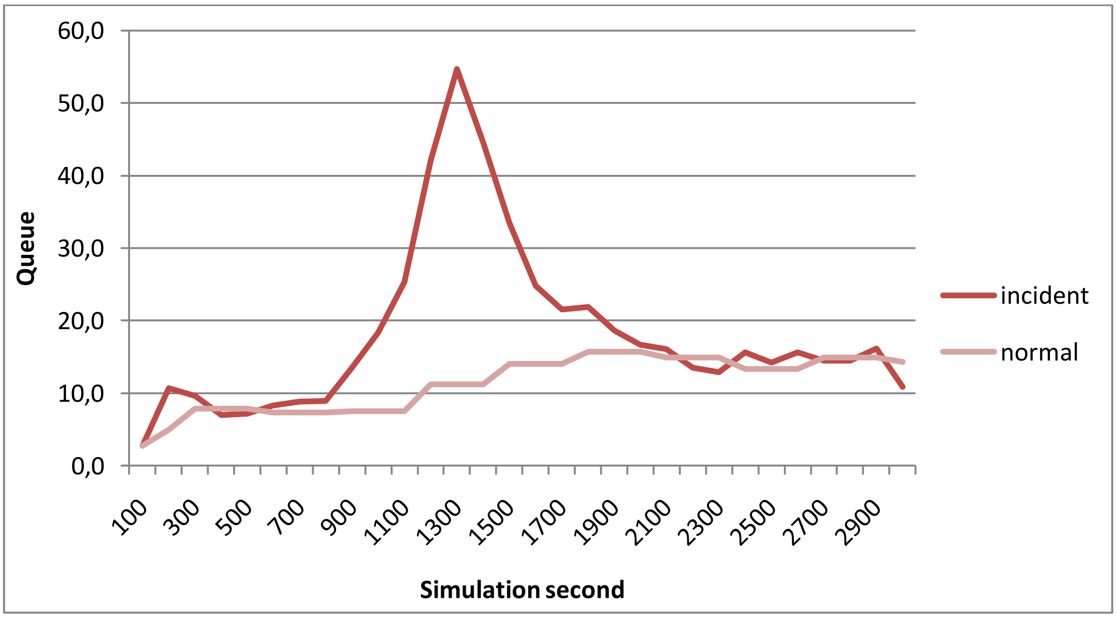


Figure 7. Queue evolution under incident and under normal flow (approach 1)

The dark red line in Figure 7 shows the queue evolution in the case of the incident. It can be observed that just after the incident, the queue starts rising steeply due to vehicles slowing down and preventing following vehicles to clear the intersection in time. The effect to this approach is direct as the actual queues rise above the algorithmic predictions. When traffic in this approach becomes heavy (occupancy exceeds the critical threshold) the strategy allocates extra time to the respective stages, in an attempt to minimise its performance measure. At the point where the queue becomes maximum, the incident has already ended and the queue returns to the normal values after approximately 10 minutes.

Figure 8. Queue evolution under incident and under normal flow (approach 2)

The picture for the second approach is different, as the related traffic movements are not in conflict with the movements crossing the incident spot. The queue rises above the respective queue under no incident, as the green time that is allocated to the approach is reduced, due to the increased green time allocated to the approaches with conflict. The queue rises only slightly above the normal case. The oscillation of the queue observed in the section where the queue is high, shows that the occupancy in the respective approach (which is obtained as an average of 60 seconds) oscillates around the critical value, thus, the respective stage is served once with increased green time and once with a shorter duration.

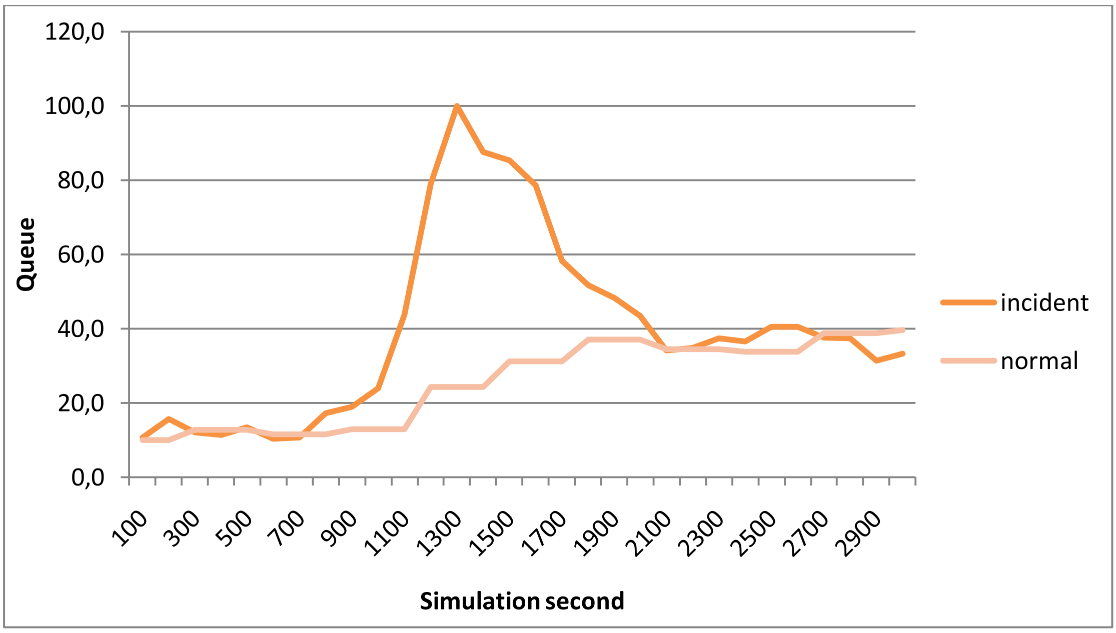


Figure 9. Queue evolution under incident and under normal flow (approach 3)

Figure 4 concerns the third approach, which also does not include any traffic movement in conflict with the incident spot. However, in this case, the queue rises rapidly as this approach is very loaded in the morning scenario. Moreover, in contrast with approach 2 (Figure 8), this approach has more space available (in the VISSIM network), and the upstream detectors are at a longer distance from the stop line. This means that additional time is required until the strategy realises the increase in the queue and reacts. Therefore, despite the fact that the approach does not include any traffic movements conflicting with the incident, the evolution of the queue is more similar to the one of approaches 1 and 4.

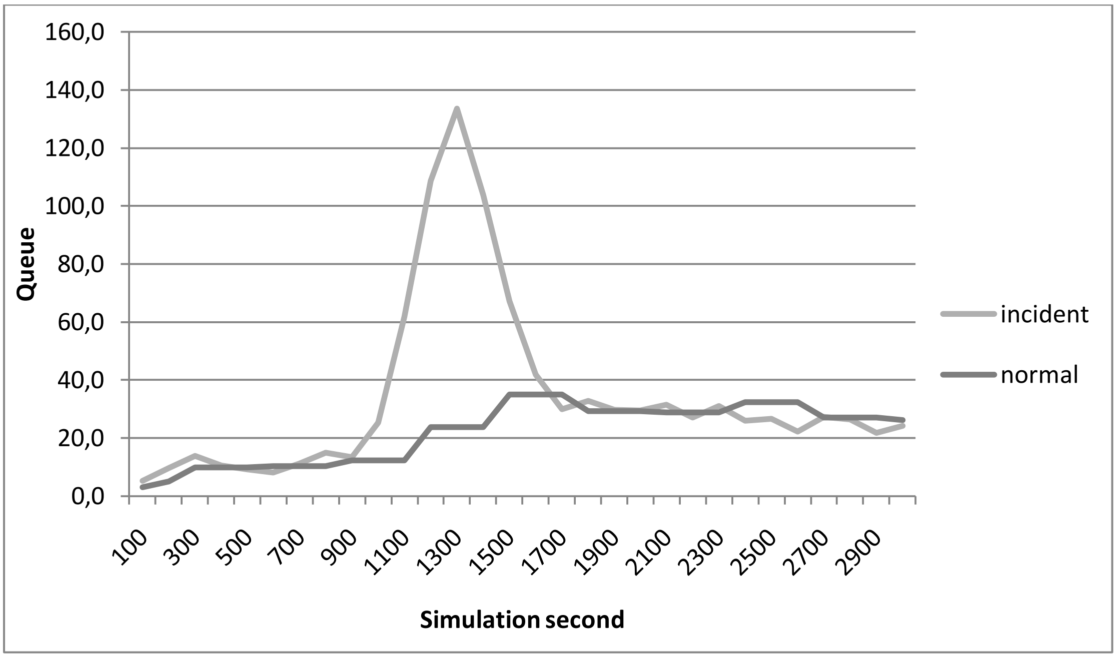


Figure 10. Queue evolution under incident and under normal flow (approach 4)

The queue evolution for approach 4 is similar to the evolution for approach 1, as the traffic movement crossing the intersection straight is one of the two conflicting movements. As for all previous figures, it can be observed that despite the fact that the incident happens at the 600th second, only a small rise happens until the 900th second while the steep rise in the queue starts after the 900th second. This happens because the same simulation file as for the normal morning scenario is used, in which, the first 900 seconds have lower flows than the next 7200 seconds. Therefore, it can be observed that, in low traffic, the impact of the incident would be much smaller. Only when traffic flows become high, the strategy starts misestimating the queue, until it becomes large.

The main conclusion drawn from this example, is that the strategy does not react to the queue growth instantly from the start of the incident, but only after it has become large enough to initiate the gating feature. However, since the queues have grown and after the incident has ended, the strategy resolves congestion relatively quickly. The slow reaction can be a disadvantage which could be mitigated if the departure rate was not constant but was dynamically estimated by the algorithm at small time intervals. Such a feature was planned to be included but was abandoned due to lack of time. This; however, is one interesting path concerning possible further research on the strategy.

In any case, the main purpose of this example is to illustrate an experiment that can be exploited to evaluate the strategy robustness while the evidence from this experiment are only indicative. An extensive quantification of the network robustness would require an thorough analysis including several different experiments and preferably on a network including more than one intersections so as to quantify the effect of an incident to other parts of the network.

6. Discussion

6.1 Efficiency and applicability

The results presented in chapter 5 show that the strategy improves the performance measures in a wide range of demands, and results in more balanced and stable control compared to the fixed time plans. However, this is common between most existing VA strategies due to the sophisticated control methods employed, in contrast to the rigid control offered by the fixed time plans. Therefore, even more important than the efficiency improvement, are two aspects: the applicability of the strategy in various environments and the flexibility of the control method employed by the strategy.

The examined intersection includes several traffic movements and 5 stages, therefore can be considered relatively complicated. In simpler intersection layouts with less traffic movements or less stages, the control method can be applied even more efficiently as the complexity of the calculations will be reduced, therefore the horizon and number of time steps can be increased. The impact of a longer horizon on the efficiency of the signal control is positive and was discussed in chapter 5.

Moreover, in simpler environments it is also possible to adopt a non-fixed stage sequence which can result in more efficient green allocation during a cycle. As was discussed in chapter 2, a non-fixed stage sequence is not preferred in Greek practice, mostly due to safety reasons (drivers familiar with the network often "expect" a stage sequence and can be confused, with a negative impact on safety). However, in other countries such as The Netherlands this is implemented in practice. Therefore, a further efficiency improvement can be expected in this case.

Concerning the applicability of the strategy to other networks, the strategy has a number of requirements which need to be satisfied, in order for the control to be as efficient as presented in this chapter. Firstly, the further the upstream detectors are placed from the stop-line, the better for the obtained arrival estimates, as the length of the short-term horizon is increased. This of course means that no significant trips are generated or terminated between the upstream detectors and the stop line, as well as no significant proportion of the vehicles turn to roads existing in-between (or at least the number of vehicles exiting or stopping to be approximately equal to the number of vehicles entering or starting). This means that the strategy application can be less efficient (but still possible) to dense road networks with several secondary streets intersecting the main roads, as the detectors will have to be placed closer to the stop-lines therefore the short term horizon for the arrival estimates will be shorter and the optimisation will have to rely more on the long term predictions.

Furthermore, the strategy needs detector readings to be accurate. This of course means that vehicles are not missed (i.e. passing between the detectors). In Greece, in busy urban networks, it is possible that vehicles moving on the right lane will have to move more to the left as half of the right lane can be closed from parked vehicles. This means that all vehicles in the other lanes will also have to move slightly left from their normal trajectories. At certain cross sections it can happen that vehicles are even moving between two lanes, therefore if detectors are not placed properly, there is a chance of missing vehicles passing from the measurement locations as they can be passing between the detectors. The problem can get even more complicated as in some locations, parked vehicles exist within daytime but not during the evening/night. For that reason, the detector locations should be carefully selected so as to ensure that no (or few) such changes in trajectories occur at any time during the day.

As far as the network robustness under the new strategy is concerned, it can be considered satisfying; however, the strategy has a weakness in reacting fast under certain incidents (i.e. accidents) and will only react when the effect of the incident is already quite apparent (significant queues have formed). In any case, the network operation under this control method will be more robust in the case of an incident, compared to using fixed-time control (with the exception of hardware failures which can be more frequent in VA control, as more hardware is used).

Finally, it should be highlighted that the developed strategy optimises signal timings based on the minimisation of the total waiting time, which solely has to do with traffic efficiency. No objectives concerning equity between road users were taken into account assuming that the selected performance measure would not result in signal timings which would face low user acceptance. A closer look in this subject is useful:

A strategy which optimises based solely on efficiency could result in cases where heavy traffic in the main direction would require long green times in expense of secondary roads which would have to wait longer. In extreme cases, the queue on a secondary road would be such that its imposed cost would always be significantly lower than the cost of the main direction and as a result vehicles would have to wait even 3-4 cycles to clear the intersection. As seen by efficiency terms, this might still be optimal; however, in terms of user acceptance this would clearly render a problem.

Although in the approach presented in this thesis the optimisation has no heuristic rules to prevent this from happening, there are reasons to believe that the structure of the cost function and the constraints in the signal timing process would not allow for such extremes to arise. Firstly, the fixed stage sequence and the maximum green durations ensure that – at least – once in each cycle all stages will be served sequentially. A minimum green time will also ensure that a certain number of vehicles will be served. But apart from these constraints, the cost function is built in a way that cost rises exponentially with time. In this way, a few vehicles waiting for a long time will impose a larger cost than several vehicles which have been waiting for less time. This effect will be more pronounced if a longer horizon is adopted due to the longer monitoring of waiting times.

6.2 Added value

This section presents the advantages of the new strategy over existing VA approaches in a comprehensive and concise way. It explains which insufficiencies, existing on other strategies, are addressed by this approach and in which way.

A basic advantage of this strategy is its **pro-active nature**. Traditional vehicle actuated signal control strategies are optimizing signal timings to respond to the actual demand in the network. Many well-established strategies use traffic measurements from a past time period (i.e. the last 60-90 seconds for strategies like SCOOT, SCATS, TUC) and use them as estimates of the current traffic in the network. These strategies are referred to, as *reactive*, in the sense that they react to the current flow in the network. These strategies react more slowly to quick traffic changes therefore a pro-active approach like the one proposed in this study has a clear advantage on reaction speed.

A second advantage of the proposed approach is that efficient control can be delivered with **less computational effort** compared to other sophisticated approaches and that it is theoretically applicable to **any number of intersections**, as each controller acts independently. The decentralised nature of the strategy means that no universal optimisation takes place therefore the complexity does not rise exponentially with the number of intersections. A number of sophisticated vehicle actuated strategies are difficult to be applied to more than one intersection either due to their complexity (in terms of required computational power) or due to their critical assumptions concerning the topology of the network (i.e. RHODES, OPAC, PRODYN – for more information, see chapter 2). Their optimisation techniques require employment of recursive algorithmic techniques such as dynamic programming and the problem cannot be solved locally (for each intersection) but should be solved globally (for all intersections), thus the complexity of the algorithm rises exponentially with the number of intersections thus severely limiting their applicability.

Furthermore, the strategy is realisable using very **basic** and relatively **inexpensive** hardware (loop detectors). Other VA strategies, like the one proposed by Liu et. al., (2002), propose viable and efficient strategies which; however, use very specific applications especially developed for that purpose, such as vehicle re-identification from the detectors using waveforms (which serve as a unique id of each vehicle) to estimate individual vehicle delay, therefore are severely restricted by the capabilities of the available hardware.

Another originality of this strategy is that it modifies an already promising optimisation technique (Branch and Bound) so as to further enhance the results it yields. In the literature, rolling horizon strategies refer only to the use of a “time step”. They do not distinguish between the time step and switching step as done in this study. In this strategy the distinction is made and as a result control is made more flexible and efficient. However, this distinction itself is not original and has already been applied in work performed by private firms (i.e. PEEK Traffic). What is original, is that it adopts a time step (interval between optimisations) shorter than the switching step (building block of the horizon). In general, as the switching step shrinks, the optimisation problem gets more complex and the control becomes more efficient. A time step longer than the switching step allows more time to be available for the optimisation therefore allows for shorter switching steps to be adopted. This is the more widely applied rational choice. However, in this work, this subject was viewed from a different perspective: a longer switching step results in a fastly-solved problem, therefore allows for a very short time step to be adopted. In this way, if the time step is very small (i.e. the optimisation is performed every 1 second) the negative effect of the longer time step (i.e. 5 seconds) is partly diminished, because the decisions are re-evaluated much more frequently. The results presented in chapter 5 prove that such an approach is viable and provides efficient and stable control.

A further enhancement in the application of the Branch and Bound technique was made by pre-identifying the part of the decision tree that will always be unfeasible under the current set of maximum greens, transition durations, horizon and step length, and omit it from the algorithm. This is described in detail in chapter 4. Although this enhancement can only be made case-specifically and with caution, it significantly speeds up the optimisation process.

In literature, vehicle actuated strategies appear to be using either quite complicated or relatively simple methods for retrieving traffic estimates. From the simple averages (rolling or not – smoothed or not) of reactive strategies, to the sophisticated traffic projection algorithms like PREDICT, there seems to be a gap, filled in by the approach taken in this study. The TE module remains simple in its concept and implementation; however, it provides clearly superior traffic estimates compared to those used by most reactive strategies. Its aim is to allow the envisioning of the queue evolution during the next horizon by using arrival estimates and intersection geometry characteristics. The division into a long-term and short-term horizon as well as the inclusion of three different consistency checks, which serve for correcting mistakes in the prediction, results in a fast queue evolution model which is simpler, easier to implement and more comprehensive than more sophisticated algorithms, yet it does not compromise a lot in accuracy.

Concerning signal coordination along arterials, other VA strategies (like RHODES or UTOPIA) impose rules specified at a greater control level, which serve for that purpose. These act as constraints for the optimisation performed locally. As a result, in the event of a controller hardware failure, the super-imposed coordination scheme might not be suitable any more as at least one intersection will not be able to optimise correctly. In this thesis, the approach taken by UTOPIA or RHODES is not adopted because coordination was not viewed as a prerequisite but as a rational choice taken by the intersection controllers, when needed (when it is more optimal for the network). The basic assumption of the present approach was that a sufficiently good projection of queue data in the future horizon helps the intersection controller to make the best decisions, without having to rely on external rules (see chapter 4 for more details). As a result, the approach taken in this study, makes the strategy more robust in the event of a deficient local controller.

7. Further research

In conclusion, some proposals for further research are outlined. During the development and testing of the signal control strategy, a number of aspects were identified that could have been researched further but would result in excessive delay for the time frame of this thesis. However, they are considered as interesting paths for further development of the strategy and research in general.

Firstly, the strategy performance can be tested further under different time horizons and the results can be compared so as to determine the sensitivity of the control method to the length of the horizon and the time step. This can result in a more informed decision on the selection of these parameters, depending on the type of application, traffic intensity, complexity of the intersection and computational limits. Different versions of the strategy were prepared during the development process; however, not enough time to carry out this type of analysis was available, as the versions and combinations of parameters that can be tested are numerous.

Secondly, the application of the Branch and Bound method, although enhanced in different ways (see chapters 4 and 6), has a potential to return even better results. More specifically, the “Bound” part can be further improved if parts of the decision tree which are very likely to yield high costs during the accumulation process, are identified at an early stage and be pruned, even before the LB becomes greater than the UB. This can be realised by adopting a probabilistic method which will provide evidence that a certain path is very likely to yield an Upper Bound higher than the current, although currently its Lower Bound is low. In this way, significant computational effort can be saved, therefore enable the adoption of smaller time steps and/or longer horizon and/or non-fixed stage sequences.

Apart from the basic parameters of the optimisation method, several other parameters are involved in certain parts of the algorithm, the values of which were determined roughly. For example, the use of standard departure rates during the green time is a good approximation for the scope of this project; however, can differ at certain cases, i.e. in the case of an incident. Therefore, it would be interesting to further develop the algorithm so as to estimate more of those parameters repeatedly and not use fixed values. In this way, the robustness of the strategy can be significantly improved.

Moreover, in this thesis, the implementation and testing of the strategy was made at a single intersection. The application on an arterial would allow to draw further conclusions concerning the established coordination. This aspect was discussed in chapter 4, at a theoretical level; however, testing the strategy in VISSIM would provide additional evidence to support the theoretical concept or assist to improve it.

Finally, the testing of robustness can be performed more extensively by executing a large number of experiments using different scenarios and a well established methodology. This work can be extensive enough to form an independent study with interesting results.

References

Aboudolas K., Papageorgiou M., Kosmatopoulos E., 2009. Store-and-forward based methods for the signal control problem in large-scale congested urban road networks. Transportation Research Part C, vol. 17, pp. 163–174.

Advice Leaflet 1: The "SCOOT" Urban Traffic Control System, http://www.scoot-utc.com/documents/1_SCOOT-UTC.pdf

Allsop R., 1971. SIGSET: A computer program for calculating traffic signal settings. Traffic Engineering and Control, vol. 13, issue 2.

Allsop R., 1976. SIGCAP: A computer program for assessing the traffic capacity of signal-controlled road junctions. Traffic Engineering & Control, vol. 17, pp. 338–341.

Anderson, J., Sayers, T., Bell, M.G.H., 1998. Optimisation of a Fuzzy Logic Traffic Signal Controlled by a Multi objective Genetic Algorithm. Road Transport Information and Control. IEE, pp. 186–190.

André M., Hammarström U., 2000. Driving speeds in Europe for pollutant emissions estimation. Transportation Research Part D, vol. 5, pp. 321-335.

Bang, K. L. (1976). Optimal control of isolated traffic signals. Traffic Engineering & Control, 17(7), 288-292.

Bell M.G.H., 1992. Future directions in traffic signal control, Transportation Research Part A, vol. 26A, issue 4, pp. 303-313.

Boillot F., Midenet S., Pierrele J-C., 2006. The real-time urban traffic control system CRONOS: Algorithm and experiments Transportation Research Part C, vol. 14, pp. 18–38.

Bretherton D., Bowen G., Wood K., 2002. Effective Urban Traffic Management and control – SCOOT version 4.4. European Transport Conference (ETC) Proceedings.

Carsten, O.M.J., Sherborne, D.J., Rothengatter, J.A., 1998. Intelligent traffic signals for pedestrians: evaluation of trials in three countries. Transport Research Part C, vol. 6, issue 4, pp. 213–229.

Chang, E., Lei, J.C., Messer, C.J., 1988. Arterial Signal Timing Optimisation Using PASSER-II-87 Microcomputer User's Guide. Research Report 467-1. Texas Transportation Institute, Texas A&M University, College Station

Chang G.L., Su C.C., 1995. Predicting Intersection queue with neural network models. Transportation Research Part C, vol. 3, issue 3, pp. 175-191.

Chen K., Yu L., 2007. Microscopic Traffic-Emission Simulation and Case Study for Evaluation of Traffic Control Strategies. Journal of Transportation Systems Engineering and Information Technology, vol 7, Issue 1, pp. 93-100.

Dell'Olmo P., Mirchandani P.B., 1995. REALBAND: An approach for real-time coordination of traffic flows on networks. Transportation Research Record, vol. 1494, pp. 106-116.

Diakaki C., Papageorgiou M., Aboudolas K., 2002. A multivariable regulator approach to traffic-responsive network-wide signal control. Control Eng. Practice, vol. 10, pp. 183–195.

Diakaki C., 1999. Integrated Control of Traffic Flow in Corridor Networks. Ph.D. Thesis, Technical University of Crete, Department of Production Engineering and Management, Chania, Greece.

Dion F., Hellinga B., 2002. A rule-based real-time traffic responsive signal control system with transit priority: application to an isolated intersection. Transportation Research Part B, vol. 36, pp. 325–343.

Donati, F., Mauro, V., Roncolini, G., Vallauri, M., 1984. A hierarchical decentralized traffic light control system. The First Realization: Progetto Torino. In: 9th World Congress of the International Federation of Automatic Control, Budapest, vol II 11G/A-1.

Dougherty M., 1995. A review of neural networks applied to transport. Transportation Research Part C, vol. 3, issue 4, pp. 247-260.

Farges J-L., Henry J-J., Tufal J., "The PRODYN real-time traffic algorithm," in *Proc. 4th IFAC Symp. Transportation Systems*, 1983, pp. 307–312.

Gartner, N.H., 1982. Development and testing of a demand responsive strategy for traffic signal control. In: Proceedings of American Control Conference, pp. 578–583.

Gartner, N. H. OPAC: A Demand-Responsive Strategy for Traffic Signal Control. In *Transportation Research Record 906,* TRB, National Research Council, Washington, D.C., 1983, pp. 75–81.

Gartner N.H., Assmann S.F., Lasaga F., Hou D., 1991. A multi-band approach to arterial traffic signal optimisation. Transportation Research Part B, vol. 25B, issue 1, pp. 55-74.

Gazis D.C., Potts R.B., 1963. The oversaturated intersection. Procceedings of the 2nd Int. Symp. Traffic Theory, pp. 221–237.

Head K.L., 1995. An event-based short-term traffic flow prediction model. Transportation Research Record, vol. 1510, pp. 45-52.

Henry, J.J., Farges, F., Tuffal, J., 1983. The PRODYN real time traffic algorithm. In: 4th IFAC, IFIP, IFORS Conference on Control in Transportation Systems, Baden, pp. 307–312.

Hunt, P.B., Robertson, D.I., Bretherton, R.D., Winton, R.D., 1981. SCOOT, A Traffic Responsive Method of Co-Ordinating Signals. TRRL Laboratory Report 1014.
Kim O.C., Park Y., Baek J.G., 2005. Optimal Signal Control Using Adaptive Dynamic Programming. Computational Science and Its Applications – ICCSA 2005, Springer Berlin / Heidelberg, pp. 148-160.

Immers, L.H., J.E. Stada , I. Yperman (2004), *"Robustness and Resilience of Transportation networks; problem survey and examples",* Paper presented at NECTAR Cluster Meeting on Reliability of Networks, Amsterdam, March 19-20, 2004

Kimber, R. M and Hollis E. M., 1979. Traffic queues and delays at road junctions. Transport and Road Research Laboratory Report LR909, Crowthome.

Kosonen I., 2003. Multi-agent fuzzy signal control based on real-time simulation. Transportation Research Part C, vol. 11, pp. 389–403.

Little J.D.C., Kelson M.D., Gartner N.H. MAXBAND, A Versatile Program for Setting Signals on Arteries and Triangular Networks. WP# 1185-81, January 1981.

Li M., 2008. Robustness Analysis for Road Networks - A framework with combined DTA models. PhD thesis, Delft University of Technology, The Netherlands.

Liu H.X., Oh J.S., Recker W., 2002. Adaptive Signal Control System with Online Performance Measure for a Single Intersection. Transportation Research Record, vol. 1811, pp. 131-138.

Lowrie, P.R., 1982. The Sidney co-ordinated adaptive traffic system: principles, methodology, and algorithms. In: Proceedings of the IEE Conference on Road Traffic Signalling, London, vol. 207, pp. 67–70.

Mauro, V., Di Taranto, C., 1989. UTOPIA. In: CCCT'89 – AFCET Proceedings, Paris.

Miller, A.J., 1963. A computer control system for traffic network. Proc., 2nd Int. Symp. on Theory of Road Traffic Flow, London, pp. 201-220.

Mirchandani P., Head L., 2001. A real time traffic signal control system: architecture, algorithms and analysis. Transportation Research Part C, vol. 9, pp. 415-432.

Niittymaki, J., Kikuchi, S., 1998. Application of Fuzzy Logic to the Control of a Pedestrian Crossing Signal. In: Transportation Research Record: Journal of the Transportation Research Board, No. 1651, TRB, National Research Council, Washington, DC, 1998, pp. 30–38.

Niittymaki J., Pursula M., 2000. Signal control using fuzzy logic. Fuzzy Sets and Systems, vol. 116, pp. 11-22.

Owen L., Stallard C.M., 1999. Rule-Based Approach to Real-Time Distributed Adaptive Signal Control, Transportation Research Record, vol. 1683, pp. 95-101.

Papageorgiou M., Diakaki C., Dinopoulou V., Kotsialos A., Wang Y., 2003. Review of Road Traffic Control Strategies. Proceedings of the IEEE, vol. 91, nr. 12.

PTV Vision, 2008. VISSIM 5.10 User Manual.

Robertson, D. I., R. D. Bretherton. Optimal Control of an Intersection for Any Known Sequences of Vehicle Arrivals. Proc., 2nd Symposium on Traffic Control and Transport Systems, International Federation of Automatic Control (IFAC), International Federation for Information Processing, International Federation of Operation Research Societies, Monte Carlo, 1974.

Schmöker J-D, Ahuja S., Bell M.G.H., 2008. Multi-objective signal control of urban junctions – Framework and a London case study. Transportation Research Part C, vol. 16, pp. 454–470.

Sen, S., and K. L. Head. Controlled Optimisation of Phases at an Intersection. Transportation Science, Vol. 3, 1997, pp. 5–17.

Shelby S., 2004. Single-Intersection Evaluation of Real-Time Adaptive Traffic Signal Control Algorithms. Transportation Research Record, vol. 1867, pp. 183–192.

SIEMENS, 2003. SITRAFFIC Control V 4.0 Operating Manual.

SIEMENS, 2005. SITRAFFIC Language V 3.1 Bedienungsanleitung.

SIEMENS, 2005. SITRAFFIC P2 V 2.0 Traffic Engineering Workstation, User Manual.

Vincent R.A., Young C.P., “Self-optimizing traffic signal control using microprocessor: The TRRL “MOVA” strategy for isolated intersections,” Traffic Eng. Control, vol. 27, pp. 385–387, 1986.

Vasudevan M., Chang G.-L., 2006. Design and Development of Integrated Arterial Signal Control Model. Transportation Research Record, vol. 1978, pp. 76–86.

Wallace, C.E., Courage, K.G., Hadi, M.A., Gan, A.G. 1998, TRANSYT-7F User's Guide. University of Florida, Gainesville

Webster F.V., 1958. Traffic signal settings. Road Research Laboratory Technical Paper No 39.

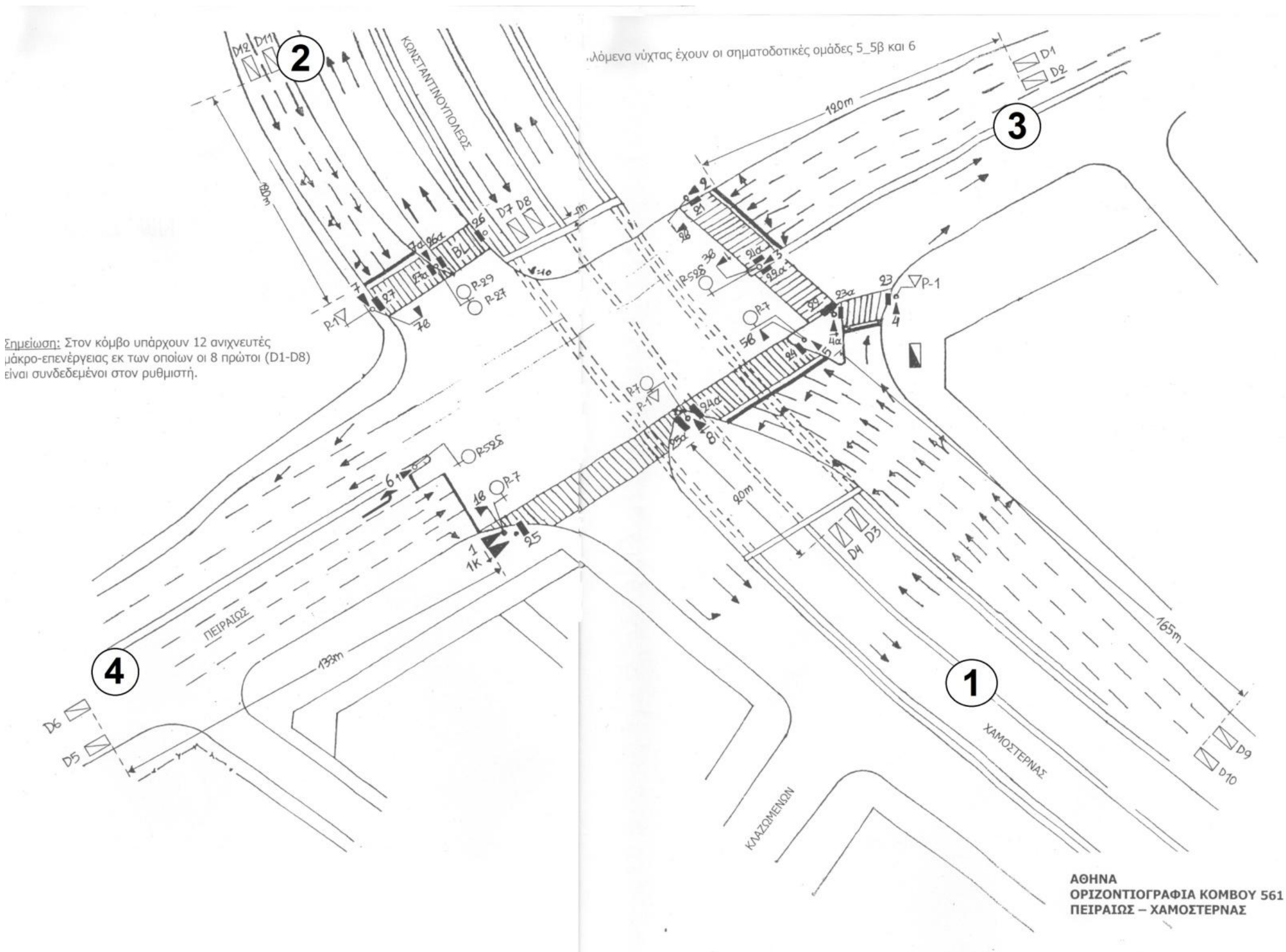
Wiedemann, R., 1991. Modeling of RTI-Elements on multi-lane roads. Advanced Telematics in Road Transport, edited by the Commission of the European Community, DG XIII, Brussels.

Wiedemann R., 1974. Simulation des Strassenverkehrsflusses (in German), University of Karlsruhe.

Yagar S., Han B., 1994. A procedure for real-time signal control that considers transit interference and priority. Transportation Research Part B, vol. 28, issue 4, pp. 315-331.

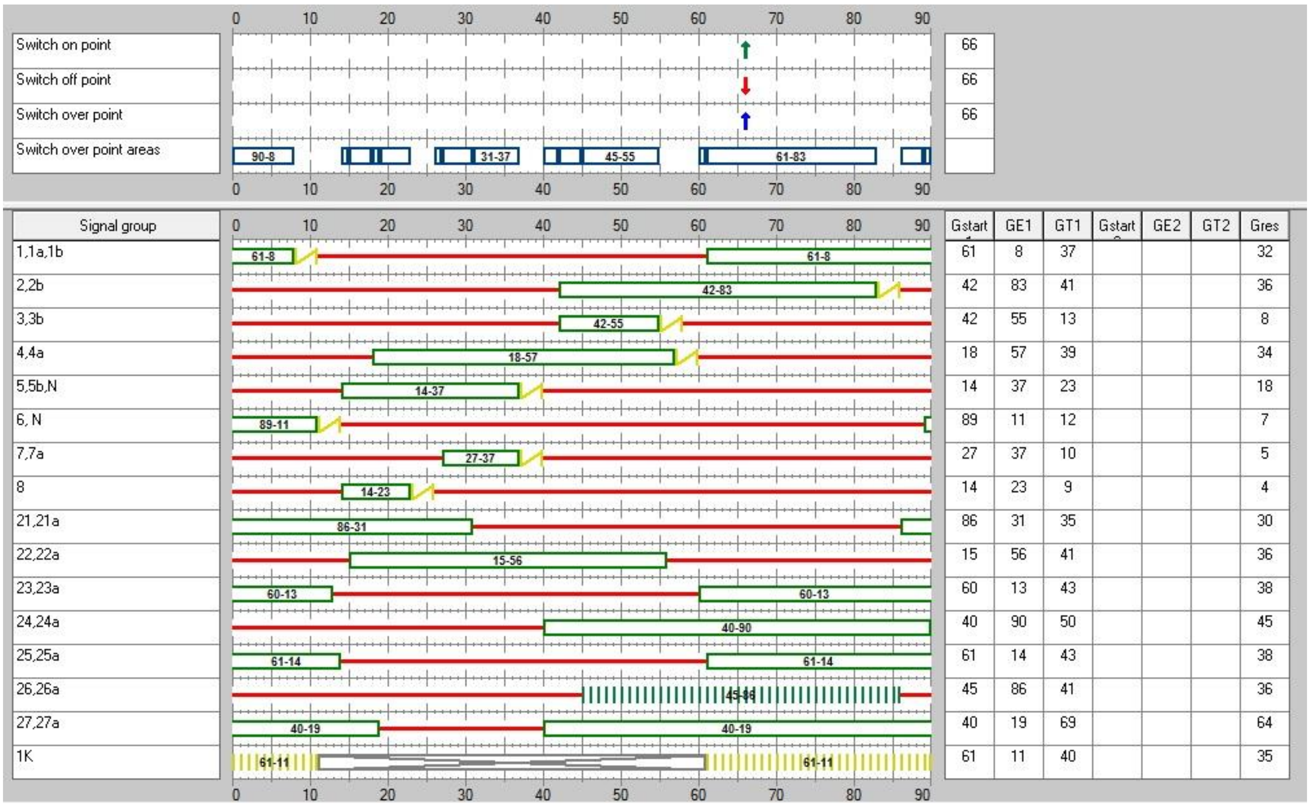
Zhang Y., Chen X., Zhang X., Song G., Hao Y., Yu L., 2009. Assessing Effect of Traffic Signal Control Strategies on Vehicle Emissions. Journal of Transportation Systems Engineering and Information Technology, vol. 9, issue 1, pp. 150-155.

Annex 1 – Intersection layout



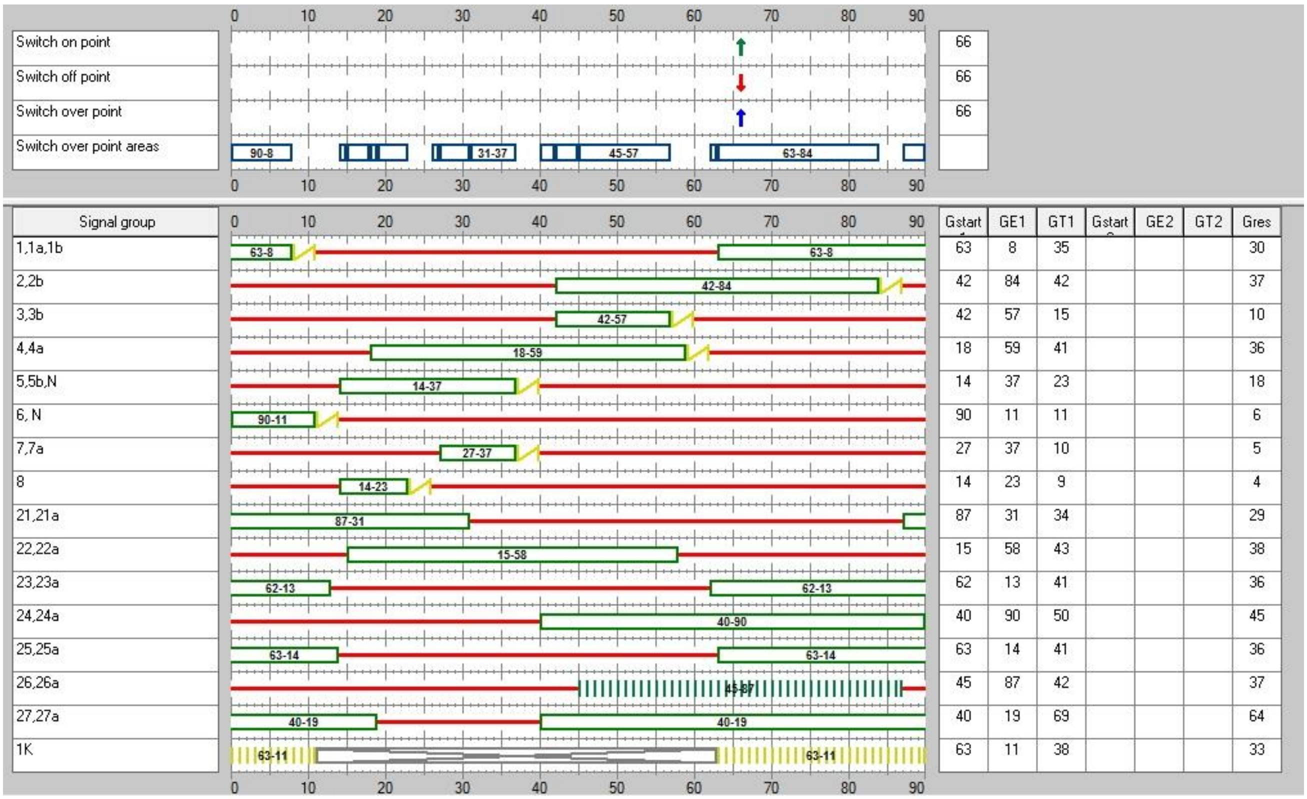
Annex 2 – Fixed time signal plans

Signal Program 1 (scenario 1)



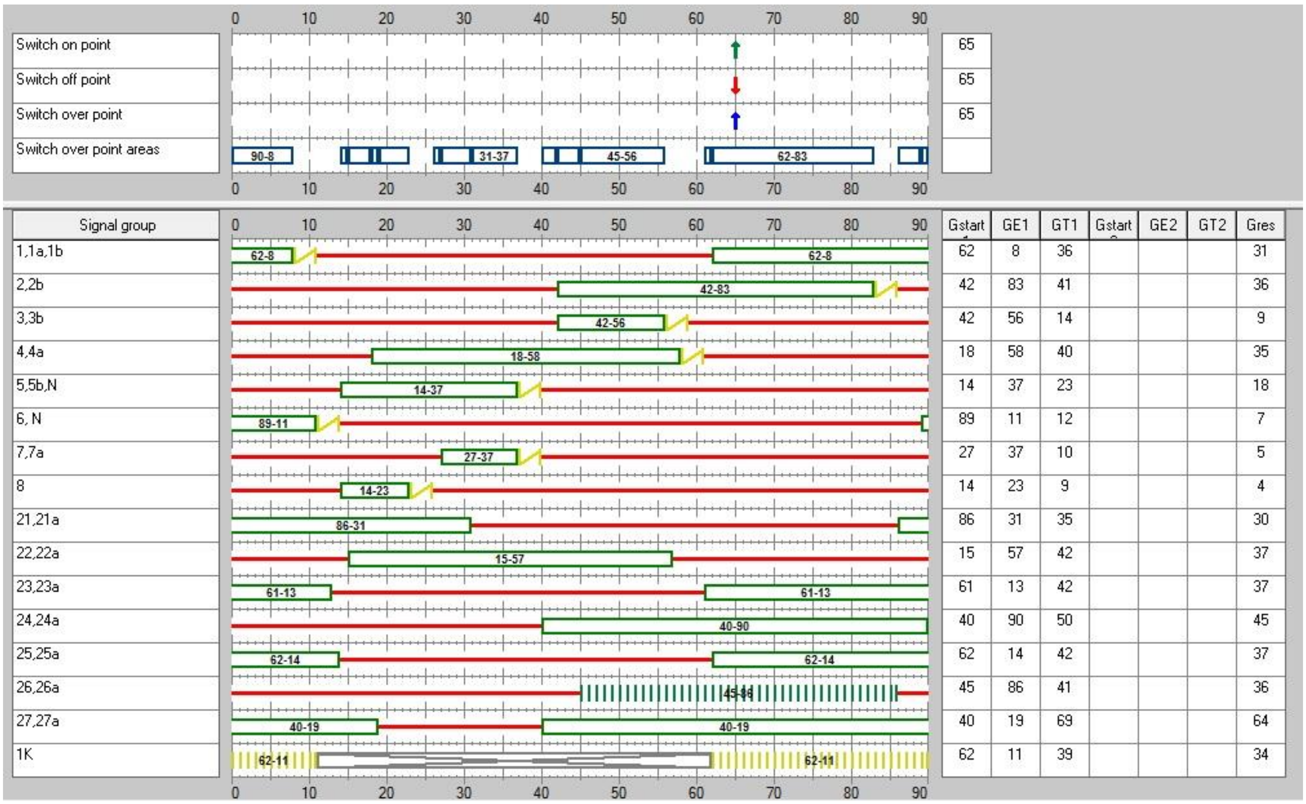
Signal group	Gstart	GE1	GT1	Gstart	GE2	GT2	Gres
1,1a,1b	61	8	37				32
2,2b	42	83	41				36
3,3b	42	55	13				8
4,4a	18	57	39				34
5,5b,N	14	37	23				18
6, N	89	11	12				7
7,7a	27	37	10				5
8	14	23	9				4
21,21a	86	31	35				30
22,22a	15	56	41				36
23,23a	60	13	43				38
24,24a	40	90	50				45
25,25a	61	14	43				38
26,26a	45	86	41				36
27,27a	40	19	69				64
1K	61	11	40				35

Signal Program 2 (scenario 2)



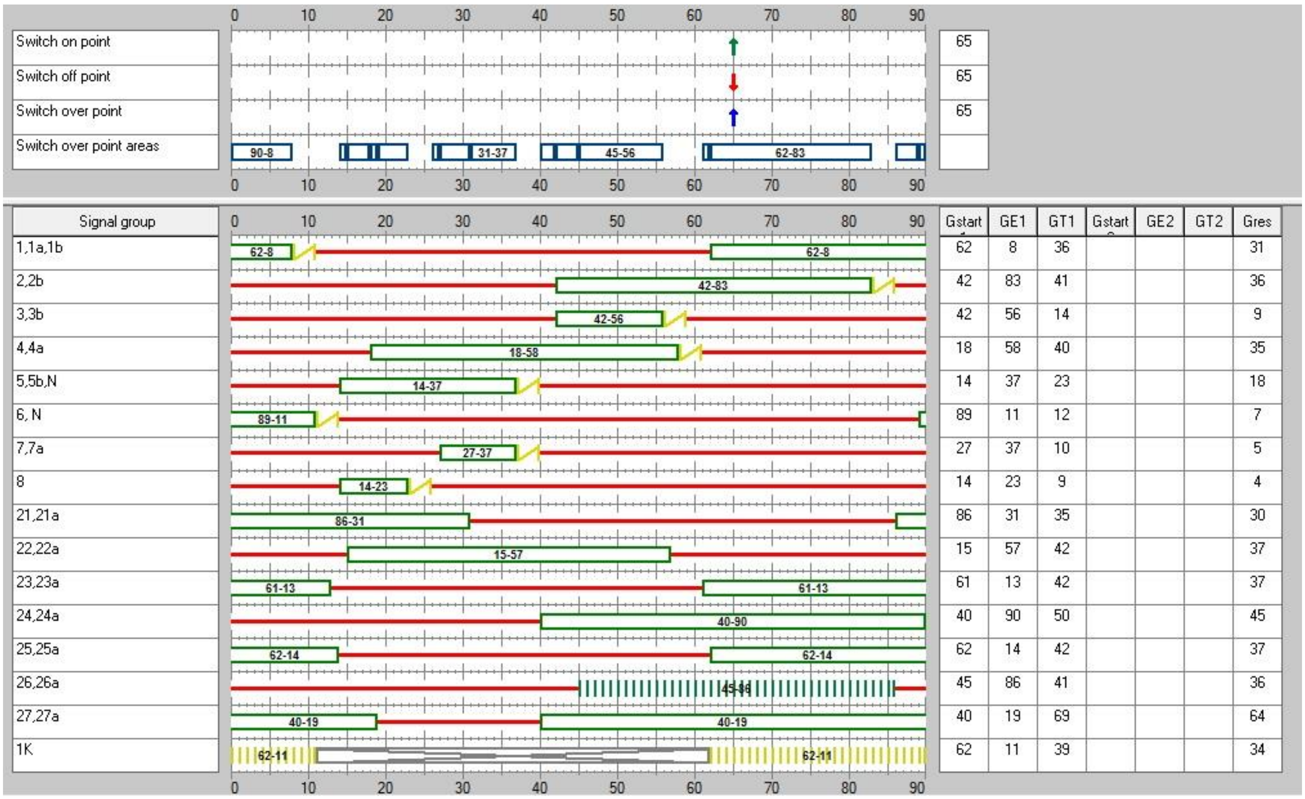
Signal group	Gstart	GE1	GT1	Gstart	GE2	GT2	Gres
1,1a,1b	63	8	35				30
2,2b	42	84	42				37
3,3b	42	57	15				10
4,4a	18	59	41				36
5,5b,N	14	37	23				18
6, N	90	11	11				6
7,7a	27	37	10				5
8	14	23	9				4
21,21a	87	31	34				29
22,22a	15	58	43				38
23,23a	62	13	41				36
24,24a	40	90	50				45
25,25a	63	14	41				36
26,26a	45	87	42				37
27,27a	40	19	69				64
1K	63	11	38				33

Signal Program 3 (scenario 3)



Signal group	Gstart	GE1	GT1	Gstart	GE2	GT2	Gres
1,1a,1b	62	8	36				31
2,2b	42	83	41				36
3,3b	42	56	14				9
4,4a	18	58	40				35
5,5b,N	14	37	23				18
6, N	89	11	12				7
7,7a	27	37	10				5
8	14	23	9				4
21,21a	86	31	35				30
22,22a	15	57	42				37
23,23a	61	13	42				37
24,24a	40	90	50				45
25,25a	62	14	42				37
26,26a	45	86	41				36
27,27a	40	19	69				64
1K	62	11	39				34

Signal Program 4 (scenario 4)



Signal group	Gstart	GE1	GT1	Gstart	GE2	GT2	Gres
1,1a,1b	62	8	36				31
2,2b	42	83	41				36
3,3b	42	56	14				9
4,4a	18	58	40				35
5,5b,N	14	37	23				18
6, N	89	11	12				7
7,7a	27	37	10				5
8	14	23	9				4
21,21a	86	31	35				30
22,22a	15	57	42				37
23,23a	61	13	42				37
24,24a	40	90	50				45
25,25a	62	14	42				37
26,26a	45	86	41				36
27,27a	40	19	69				64
1K	62	11	39				34

Annex 3 – TL code manual

This Annex aims at providing assistance to the reader for understanding and following the programming code step by step. This assistance is provided in the form of comments included throughout the code. Whenever a footnote: "*Comment(number)*" is included in the code, it corresponds to an explanation given in this Annex.

Here, the purpose of each module is described and it is put in the context of the strategy. In this way the reader can understand and link what is written in the code to the strategy described before.

The programming code is provided as Annex 4. However, in that Annex, the modules are sorted in an alphabetical order, while it is preferable to follow a different order when reading the code. The proposed order is the following:

1.1 arrivalselection
1.2 arrivalupdate, arrivalupdate1, arrivalupdate2
1.3 STstateselection, STstateupdate

2. Signal Monitoring (SM) module

3.0 Traffic Estimation (TE) module
3.1 queuecheck module (subroutine called within TE)

4. Turning Percentages (TP) module

5.0 Branch and Bound (BB) module
5.1 bbcontinue module (subroutine called within BB)
5.2 bbswitch module (subroutine called within BB)
5.3 pathindicator module (subroutine called within BB)

6.1 Initial values module

7.1 Interface SL module

8.1 Anwenderfunktion, AnwenderInit, Hauptfunktion

1 Commonly used subroutines

The subroutines that are going to be presented are included in the code due to the inability of TL to include variables with more than 1 dimension. To solve this problem, these variables are separated into n vectors where n is the number of columns of the desired matrix and correspond to one vector each. Each vector is a different variable and all together are included in a subroutine which forms an artificial "matrix". In this way an element of the matrix can more easily be selected or updated.

1.1 arrivalselection

This subroutine is used to select the estimated arrivals of a signal group, at a future time instant. The "isg" indicator, indicates the signal groups (one column/vector corresponds to each signal group) and "t2" indicates the rows of the matrix (time in seconds). Thus, when the arrivals for SG6, for the 8th second are required, the subroutine will be called, with isg=6 and t2=8. The output will be stored to the variable value2.

1.2 arrivalupdate, arrivalupdate1, arrivalupdate2

The first subroutine is called to perform an update to the arrival estimates of a signal group. For example, it can be called when a vehicle has just passed the upstream detectors and should be projected to the future arrivals. It functions in the same way as "arrivalselection" (isg: signal group indicator/columns, t: time indicator/rows). The arrivals of a signal group at a certain time are assigned the "value" variable (which is input to the subroutine).

The next two subroutines, are built in exactly the same way; however, the update of the "arrivals" value is made slightly differently. The second subroutine serves for updating the arrival matrix from the previous to the current second when the TE module is called again. It actually brings all elements of the matrix one row upwards which means that the previous seconds' arrival projections are brought to the current second. The third subroutine is used when new arrivals are estimated by the TE module and should be placed to the appropriate signal group (isg) and future time (t). These obviously have to be added to the existing arrival estimate of the current second.

1.3 STstateselection, STstateupdate

These subroutines were developed as a result of the way the BB module is built. The final output of the BB module that is passed to the controller is the duration of each stage. The stage durations are calculated from the summation of all the elements of one vector, denoting the stage and its state in each second along the horizon (0: stage not running, 1: stage running). An example of such an output is the following: the "STstate1" variable (state of stage 1 in each second) has its first 30 elements equal to zero (stage not running during the first 30 seconds), then the following 25 elements equal to 1 (stage is running from the 31st to the 55th second) and the rest of the elements equal to zero (stage not running for the rest of the horizon). If all elements are summed, for the next horizon, stage 1 will receive 25 seconds of green.

The first subroutine (STstateselection) is used in the summation of the stage states along the horizon. The variable value4 selects the state of the stage "ist3" at second "t4". The second subroutine is used for the assignment of the state of a stage, during the optimisation steps. It assigns the value of the variable "value4"to the stage "ist2" at time "t3".

2. Signal Monitoring (SM) module

As described in chapter 4, this module provides the state of each signal group in the following second. This is used as input from the TE module. The module is called every second.

Comment 2.1

The examined intersection has 9 vehicular signal groups in total. A “for” loop is used to interrogate the state of the first 8 signal groups. The command SG_ZUSTAND_AL serves for that purpose. If the signal group is green or yellow, the “sstate” variable takes the value 1, otherwise, it takes the value 0. In the VISSIM simulation, cars use all the yellow indication and for that reason it is considered as effectively green.

Comment2.2

One of the vehicle signal groups is a special case and is treated separately from the previous loop. In the signal program defined in Control, all signal groups (including pedestrians) are defined. However, in TL, pedestrian signal groups are not included in the optimisation thus, the number of signal groups is smaller. This signal group, although it refers to vehicles, has the number 16 in Control and VISSIM, while it has the number 9 in TL (as pedestrian signal groups don’t exist). In this module, the state of the signal groups is asked from the controller, thus number 16 should be used. In the rest of the TL code, this signal group is denoted as 9, because in this way continuous counters to run through all signal groups can be used.

3.0 Traffic Estimation (TE) module

The goal of the TE module is to provide an arrival matrix, with columns equal to the signal groups and rows equal to the seconds of the next horizon. Its update with new data is achieved by the “arrivalupdate” subroutines while the selection of data from it is performed by the “arrivalselection” subroutine. The arrival matrix is the basic input for the optimisation.

Comment 3.0.1

This loop updates the arrival estimates from the previous second. Each time the TE is called, all arrival estimates from the previous second should be moved one row upwards as one second has passed. “tarriv[isg]” is the ending second of the short-term horizon for each signal group. “longstart[isg]” is the starting second of the long-term horizon for each signal group. The short-term horizon can have a maximum length of 60 seconds, therefore if the estimated “tarriv” is above 60, the update is done till the 60th second. If the short term horizon has dropped down to zero (when queues reached the upstream detectors) the arrivals for the next second are set to 0.

Comment 3.0.2

The following loop (for “approach”=1 till 4) is the main loop in this module. It performs all the basic calculations for each approach of the intersection. Within this loop, the short-term and long-term arrivals for each signal group of each approach are estimated.

Comment 3.0.3

“kdet” denotes the upstream detectors of the approach. It runs from 0 to 1 because in the examined intersection each approach has maximum 2 upstream detectors. Under this loop, data from the upstream detectors are collected.

Comment 3.0.4

The variable vehdet is the number of vehicles that passed the detector during the previous second. This is retrieved by reading a continuous counter of vehicles that runs for all detectors and subtracting the value it had in the previous second (vehdetprev). However, this counter is reset in the TP module. In this case, vehdet can become <0. This branch, checks whether this is the case and corrects it.

Comment 3.0.5

"vehapp" is the summation of "vehdet" for all upstream detectors and provides the number of vehicles that passed from all detectors of the measurement location in the last second.

Comment 3.0.6

Every 60 seconds, the sum of vehicles that passed from the measurement location should be estimated (vehapp80). This flow will be used for the estimation of the average speed and the long term arrivals. The counter "take" is augmented by 1 each second (the augmentation is done later in this module). When it reaches 60, "vehapp80" is estimated and "take" is set to 1. The calculations are similar to the ones made for the "vehapp" variable.

Comment 3.0.7

Every 60 seconds the average occupancy for the measurement location should be estimated. This average occupancy will be used for the estimation of the speed. The command DET_LESEN_AL(…, DBS) provides the occupancy of a detector during the last second. Therefore, the occupancies of all upstream detectors of the approach, for the last 60 seconds are accumulated (occapptemp). Then, then the counter "take2" becomes 60, the average occupancy (occapp) is calculated.

Comment 3.0.8

These commands provide the time gap from the last actuation of any detector on this measurement location. This gap is required for the determination of the short-term horizon but also for the queue consistency checks (see later on). The command DET_LESEN_AL(…, DLS) provides this gap. The minimum gap from all detectors of the measurement location is eventually stored.

Comment 3.0.9

The following loop concerns the stop-line detectors for this approach. It calculates the number of vehicles that left this link during the last second. The calculations are similar to the ones described in comments 3.0.3, 3.0.4, 3.0.5, with one difference: in order to count a vehicle, it is not enough to check whether the detector was actuated, but also to check that the detector became non-occupied afterwards (otherwise, it means that a vehicle just stopped on the detector just before the stop-line and waits for the green indication, therefore it is still in the queue. This is checked with the DET_LESEN_AL(…, DAU) function (>0: the detector is occupied, 0: the detector is not occupied).

Comment 3.0.10

The variable "actualqueue" is used to estimate the number of vehicles present between the upstream and stop-line detectors. This variable is used for the second consistency check concerning the queue lengths (see chapter 3.1). The actual queue is estimated by adding the vehicles that actuated the upstream detectors and subtracting the vehicles that actuated left the stop-line detectors.

Comment 3.0.11

The purpose of this loop is to find the signal groups that correspond to the current approach and store them to the "appsg" variable. The signal groups that correspond to each approach are stored in the TL parameter "AppSG". The reason for including this variable and not using directly the parameter AppSG is that only the signal groups of the examined approach are selected and stored therefore they are more usable for the purposes described later.

Comment 3.0.12

This conditional branch examines whether the number of vehicles present between the upstream and stop-line detectors have reached a critical value. Above this value, it is considered that the queue is approaching the upstream detectors therefore it is urgent to act before the queue reaches the detectors (for more information see chapter 3.1).

If the condition is true, the arrivals for the signal groups of this approach are manually set to 0,6 vehicles/second (above the maximum flow rate) so as to "force" the optimisation module to provide green soon. **Please note** that 60 instead of 0,6 is used, because all arrival, departure and queue values are multiplied by 100. This is done because TL cannot incorporate decimals therefore truncates non integer values. This applies to the rest of the code.

If the condition is false (normal case) the following commands will describe all calculations required to update the short-term and long-term arrivals.

Comment 3.0.13

The following commands examine the case where the previous condition is false. The conditional branch included here checks whether there were actuations in the previous second, and whether the average flow and occupancy for the last 60 seconds is >0. If yes, the appropriate partial vehicles are generated (see chapter 3.1) and assigned at the correct time in the arrival matrix. Firstly, the speed is estimated (vel). Secondly, the partial vehicles are estimated (partialveh) as the product of the number of actuations (vehapp) by the respective turning percentage (TPSG). Thirdly, the length of the link which is not occupied by a queue is found (clear) by subtracting the length of the existing queue (existqu*45) from the stop-line – detector distance. Finally, arrival time to the back of the queue is estimated assuming a constant speed (space/time) and if this is less than 60 (maximum value) and also not zero, the partial vehicles are assigned to the respective place in the arrival matrix.

In the occasion that no actuations occurred in the last second, the arrival times for the previous actuations are reduced by 1 second.

Comment 3.0.14

The following lines provide the calculations for the update of the long term arrivals. Firstly, the start of the long term horizon is calculated. The long term horizon starts at the end of the short term horizon (arrival time of last vehicle that passed the detectors plus time gap from this actuation). If the estimated end of the short term horizon is above the 60th second, then it is set to the 60th second and the long term arrivals start there. The arrivals/second are uniform and are equal to average flow of the last 60 seconds.

Comment 3.0.15

This conditional branch checks whether the "take2" counter used for the estimation of the average occupancy has reached the 60th second and if yes it sets it back to 0.

Comment 3.0.16

This loop provides an estimate for the existing standing queue of each signal group for the next second. This queue is used as an initial estimate from the BB module. The queue estimate from the previous second is augmented with the arrival projections for the next second and is also reduced by the departures of the next second, if the respective signal group is green (again, note that all values are multiplied by 100). At this point, the “queuecheck” subroutine is called, which is the first consistency check for the queue (see chapter 3.1) and is described later (see comment 3.1.1 below). After this point, the main loop of the module examining all the approaches of the intersection ends.

Comment 3.0.17

This loop is called after the existing queues for all signal groups have been estimated. It is the second consistency check (see chapter 3.1) and checks whether a queue estimate has become unreasonably large. This consistency check is performed only if the current queues are not close to the upstream detectors (first conditional branch used). In this case, for the signal groups of each approach (the appsg variable is used again) the queues estimated are summed (appsg<>0 while loop). This sum is then compared to the number of vehicles present between the upstream and stopline detectors (actualqueue variable) which may well include vehicles not standing). If the first is larger than the second, it is reduced pro-rata by the use of the “factor” variable.

3.1 “queuecheck” subroutine

Comment 3.1.1

Note 1: Detector 88 is not assigned to any signal group, it is an “imaginary” detector. It has to be used in certain cases to fill-in extra cells existing in the TL parameters (as cells cannot be left blank). It is a detector that always returns zero values.

Note 2: Detectors 23 and 17 are a special case of stop-line detectors as they are placed downstream from their stop line. For the need to place them downstream and for more information see chapter 3. The consistency check cannot take place for these two detectors as well as for detector 88.

This subroutine is the first consistency check for the queue estimate. It provides corrections for small queue lengths (see chapter 3.1). This subroutine has two parts. The first part includes the commands under the first conditional branch (existqu>0) and the second part the commands under the “for” loop: “jdet = 0 … 2”.

The first part examines whether all the stop-line detectors of a signal group have a gap larger than 3 seconds. If yes, then if the queue estimate is >0, it is reduced to zero (as no vehicles are waiting). This check, helps to nullify small queues that were actually dissipated in the green time but the algorithm estimates that a small queue still exists. As mentioned before, the check cannot take place for detectors 88, 17 and 23 therefore, it has to be checked if any of the signal group’s detectors is one of those mentioned above. If a detector of this signal group is one of those mentioned, then the consistency check should take place for all other detectors **apart** from that one. This generates a number of cases that need to be examined (8 in total) so as to conclude which detectors will be checked. In any case, the condition to be checked is whether the DET_LESEN_AL(…, DLS) function has a value >8.

The second part, examines whether the estimated queue is zero but in the same time the stop-line detector is actuated. In this case, the queue is augmented by 1 vehicle (100). This helps correct errors arising from over-estimation of the queue dissipation rate. Once again, this check can be made only if the stop-line detectors of the signal group do not match the ones mentioned above.

4. Turning Percentages (TP) module

The TP module provides an estimate of the latest turning percentages per signal group. For reason explained in chapter 3.3, the module cannot be called frequently. An arbitrary frequency of 15 minutes is chosen, by which the module seems to be producing sufficiently accurate estimates. The initial values of the turning percentages are set in the “initialvalues” subroutine which is called the first time the algorithm starts. These values are set equal to the VISSIM turning percentages.

Comment 4.1

The turning percentages are estimated per approach. All intersection approaches are examined sequentially (counter iapp: from 0 to 3).

Comment 4.2

In the first part of this module, all the detectors of the approach are interrogated to obtain their cumulative actuations during the last 15 minutes, using the “detcount” variable. The counter used is jdet. However, the parameter AppDet in which the detectors of each approach are stored, has a fixed size of 5 columns (the maximum number of detectors that can exist in an approach). If less detectors exist, the extra cells are filled with detector 88 (imaginary detector). Obviously, for this detector, no counts are obtained. When the vehicle counts for a detector are obtained, its counter is reset to zero (function DET_LADEN(DET, DZS, 0)). The counts for each detector are then assigned to a separate variable (counts1-6). Moreover, all detector counts for the approach are summed so as to obtain the vehicle counts for the whole approach (stoplinesum).

Comment 4.3

In the following part of the module, if the approach total count was > 0, the turning percentages per stop-line detector (TPdet) are calculated, by dividing the detector’s vehicle count with the approach total count.

Comment 4.4

In the third part of this module, the turning percentages by signal group are estimated from the turning percentages by detector. To do this, firstly, the signal groups per approach as well as the detectors per signal group should be retrieved. The first is done by the first loop (k=1…4). Each approach has maximum four different signal groups, stored in the AppSG parameter. If it has less, the extra cells of the parameter are filled with signal group 0 (imaginary). Counting from 1 to 4, all signal groups of the approach are selected. If the current signal group is not zero, then all the detectors of this signal group are sequentially selected. (j=1…3). This detector is stored in the “currentdet” variable. Then, by examining all detectors of the approach, it is identified when a detector of the approach belongs to the currently examined signal group. In this case, the turning percentage of the detector is accumulated to the turning percentage of that signal group.

5.0 Branch and Bound (BB) module

The output provided by the previous modules is exploited by the optimisation module to obtain the optimal path for the next horizon. The output of the BB module is a set of 5 stage durations for the next horizon which are implemented for the next time step (1 second).

Comment 5.0.1

In order to be applicable, the optimal path should be starting from the stage that is currently running. Therefore, at the beginning of the BB module, all stages are interrogated and the one that is running is found using the PHA_INFO function. If a stage is not running, then a transition to a stage should be running. If this is true (the function has the value 16), the variable denoting the current stage (curST), takes the value -1.

Comment 5.0.2

If a transition is running then no optimisation needs to be made for this step (for more information see chapter 2.3). In this case, the rest of the module is skipped. Otherwise, the optimisation is started. Initially, the Upper Bound variable (UB) takes the value 4,294,967,295 which is the maximum number that can be stored in TL (if possible it should have been set to infinity).

Comment 5.0.3

The binary decision tree described in chapter 2.3 is created by adopting a 16-ple "for" loop. Sixteen variables are used (step1-step16) which can take the values 0 or 1. Value 0 means that the stage is terminated and the next transition starts while value 1 means that the current stage will be running for the next 5 seconds. In this way 65536 (2^{16}) paths are created. Under these loops, the cost calculations take place.

Comment 5.0.4

At the start of the evaluation of each path, the current stage has to be re-defined, as the "curST" variable will be altered along the cost calculation. Therefore, when a new path is defined, the "curST" variable denotes again the stage that is currently running. The actions performed are similar to those described in comment 5.0.1. Additionally, The elapsed duration of the current stage is obtained. The SG_ZUSTAND_AL variable returns the time from the start of green for a signal group. The green duration of this signal group belonging to the previous transition is subtracted from this, to obtain the net duration of the signal group green (since the start of the stage). This is stored to the "elapsdur" variable and is used later for the pruning of paths that exceed the maximum green.

Comment 5.0.5

The following commands make an initialisation of some variables that will be used in the cost calculation. "LB", "pathcost" and "cumcost", are the cost accumulation variables used in the branch and bound method and are set to zero at the beginning of the path. Moreover, the state of all stages, for the seconds of the following horizon, is set to zero (for the STstate variable and its use, see comment 1.3 about the STstateupdate subroutine). Finally, the initial queues of all signal groups at the beginning of the horizon are obtained from the TE module.

Comment 5.0.6

The cost accumulation for the path under evaluation starts here. According to the values of the variables step1, step2, …, step 16, the stage states along the horizon are defined. According to these states, the evolution of the queues along the intersection can be estimated (using also the input from the TE module). The cost calculation is made per step. For the first step (whether it is 0 or 1) the calculations are written in the BB module. On the contrary, the cost calculations for steps 2-16 are written in two subroutines named "bbcontinue" (called when the stage is extended) and "bbswitch" (called when the next

stage is called). This is done because the cost calculations for the first step are somewhat different compared to the calculations for the next steps.
The following commands present the cost calculations for the first step. The case where the stage is extended (step1 = 1) is examined first.

Comment 5.0.7

In the occasion that the path extends the currently running stage, the starting second is second 1 and the ending second is 5. Variables “tstart” and “tend” are used in each step to denote the starting and ending time of the step. If a stage is extended, the duration of the step will always be 5, while if a stage is terminated the duration of the step will be equal to the duration of the transition. Here, a counting loop is used, (counting from tstart till tend) to update the states of all stages during these 5 seconds. For this the “STstateupdate” subroutine (see comment 1.3) is called. Obviously, the current stage state along the 5 seconds will take the value 1 while the rest of the stage states will take the value 0.

Comment 5.0.8

The variable “cumcost” (cumulative cost) is firstly set to zero. Then, the main loop of the cost accumulation starts. This is used to examine all signal groups sequentially (isg = 1…9). Therefore, the calculations that follow are made per signal group.

Comment 5.0.9

The first action is to accumulate the arrivals during this step for this signal group. The input is requested from the TE module using the “arrivalselection” subroutine (comment 1.2) and an appropriate “while” loop. The arrivals are accumulated to the “AR” variable.

Comment 5.0.10

The following commands aim at the estimation of the departures for this step. Firstly, it is found whether the examined signal group belongs to the current stage (by questioning whether this stage’s signal groups, which are stored in the STSG parameter, match the current “isg”). If this is true, then the “SGstate” variable takes the value 1, otherwise it remains 0.

Comment 5.0.11

The departures for this signal group for the duration of the step are estimated. The departures are non-zero only when the signal group is green (SGstate=1) and are assumed to be equal to the product of: an approximate departure rate of 0,52veh/sec (52, because it is multiplied by 100), by the step duration (5 seconds) by the respective number of lanes. The departure rate was set to this value after performing a series of counts in VISSIM. It is the average of all the departure rate estimates that were obtained. Then the estimated number is checked for consistency: If the estimated departures are greater than the arrivals and the existing queue, then they are set equal to the sum of the arrivals and the existing queue, meaning that no more vehicles than the ones existing plus those arriving can depart.

Comment 5.0.12

The following formula estimates the cost incurred by the queue of this signal group during the current step. It is an application of equation 2, presented in chapter 2.3. Note that the difference ((tend-tstart)/2) is multiplied by 10, otherwise the resulting decimals would be discarded by TL. In the end, the cost is divided by 100 because the arrivals/departures were already multiplied by 10 and the time duration was also multiplied by 10.

A consistency check takes place so as to ensure that the current cost will not be negative. Finally, the cost for this signal group is added to the total cost of the step ("cumcost"). This variable will be the sum of the costs for the 9 signal groups.

Comment 5.0.13

The module has to estimate the queue that would result from the decision dictated by this step. In the current case, the decision would be to extend the stage for the next 5 seconds. To continue the cost estimation from this point on, there is a need to know the resulting queue after these first 5 seconds. This queue is estimated by adding the arrivals and subtracting the departures from the previous queue estimate. If this sum is less than zero (meaning that the estimated departures would be more than the arrivals and the existing queue) the queue is set to zero.

Comment 5.0.14

Finally, the estimated cost is accumulated as described in the equations presented in chapter 2.3, to obtain the current Lower Bound. Moreover, the elapsed duration for the running stage is augmented by the 5 seconds of this step.

Comment 5.0.15

If the variable step1 of the selected path had the value 0, then the cost accumulation would start here and not at the point where comment 5.0.6 referred. Value 0 means that the current stage would be terminated. The calculations have many similarities with the ones described for step1=1. However, due to the fact that a new stage starts as well as that the stage transitions have different durations, some differences have to be mentioned.

Comment 5.0.16

The first commands describe some initialisation actions required. Firstly, if the current stage was the fifth (last), then the next stage would be the first, otherwise, the next stage would be the previous plus one. Secondly, the elapsed duration of the new stage is set to zero. Thirdly, the starting and ending times of this step are defined like before, with the difference that the duration is not 5 seconds, but equal to the duration of the respective transition. The update of the stage states is made similarly as before. However, all stage states are set to zero because no stage runs during the transition period.

Comment 5.0.17

The following commands divide the transition into appropriate intervals, as required by the cost estimation method (see chapter 3.4 for details). These commands are capable of dividing stage transitions into intervals, even if they are not exact multiples of the time step, however, in this version of the algorithm this feature is not needed. In this case, the following lines break the transition interval into parts of 5 seconds. The number of parts is stored in the "parts" variable, while the ending time of each part is provided by the variable "tparts".

Comment 5.0.18

The main difference in the cost estimation during a transition, is that the estimation is done per part and not for the whole transition at once. For that reason, the "cumcost" variable used before, is replaced by a multi-dimensioned "cumcost2" variable, with dimensions equal to the number of parts. Thus, the same loop for all 9 signal groups is adopted, however, the loop is repeated for all parts.

Comment 5.0.19

The following commands are similar to the ones described in comments 5.0.7 – 5.0.10, with the difference that instead of the time indicator "tend", "tparts" is used, as the end of the part and not the end of the step is sought. Moreover, it is determined whether the examined signal group belongs to the previous, to the next or none of these stages. This is required for the commands that follow.

Comment 5.0.20

In the case of a transition, the estimation of the departures is more complicated, because the state of each signal group can change within the step (and the part of the step). For that purpose, there is a need to examine how many seconds of green (if any) each signal group has, within the examined part. This is done using the following conditional branches. Firstly, it is asked, whether the signal group has any green duration within the examined transition (the seconds of green for each signal group in each transition are stored in the TransSG parameter). If no, the departures are zero. If yes, it is asked whether the signal group receives this green after the end of the previous stage or before the start of the next stage. Moreover, it is asked whether the ending second of the current part is before or after the end of this green time. By using these three conditional branches sequentially, all cases are covered. By finding the correct case, it is possible to calculate exactly how many seconds of green exist (if any) for this signal group in the current part. This duration is then used for the estimation of the departures from the start of the transition till the end of the current part. However, if this part is not the first, the departures estimated in the previous part should also be subtracted, to obtain the departures only in the last part.

Comment 5.0.21

The estimation of the current cost for this part of the step follows. A small alteration in the calculation exists if the examined part is the first or not, concerning the starting time of the part. Moreover, the usual consistency check for ensuring the non-negativity of the cost takes place.

Comment 5.0.22

The following commands apply equation 4 (described in section 2.3). Firstly, the appropriate powers of 2 (depending on the number of parts) are obtained (stored in the power2_int variable) for the multiplication of the LB and the c_i variables. These powers are stored in the "multi" variable. Secondly, the "multi" variables, together with the current LB, and the cost of each part estimated before, are used to update the lower bound, to correspond to the end of the transition period.

Comment 5.0.23

In the end of this step, the indicator of the current stage is updated and the elapsed duration of this stage is set to zero, as it just starts.

Comment 5.0.24

From this point on, the BB module continues the cost calculation for the selected path, by following the same procedures. The cost estimation and accumulation is made from the "bbcontinue" and "bbswitch" subroutines, depending on whether the current stage is extended or terminated. These subroutines have similar structure to the one described before, however their small differences are discussed later on.

An additional point is that after each step, a number of checks whether this path should be continued or discarded are made. These checks are featured by the Branch and Bound method, or simply from common sense. Firstly, it is examined whether the path has reached the end of the horizon. A path can reach the 80^{th} second before using all 16 steps (see chapter 2.3 for details). If this is true, the algorithm should move on to its final evaluation. In any case, the last step used by the path is stored in the "laststep" variable. Secondly, it is examined whether the LB variable has become larger than the UB, thus the path has to be pruned (as dictated by the Branch and Bound method). Thirdly, it is examined whether the stage that is running has exceeded its maximum admissible duration. If none of the above is true, the cost accumulation for the next step takes place. If any is true, the algorithm moves at the end of the last step.

Comment 5.0.25

If any of the conditions mentioned in comment 5.0.24 is true or if all steps have been used, the algorithm reaches this point. Here, a path has either been pruned and discarded or needs to be finally evaluated. To distinguish these two cases, the first two conditional branches are used. Firstly, the paths that have reached the end of the horizon are filtered. Secondly, from these paths, the ones that still have a LB smaller than the UB and elapsed duration smaller than the maximum green are kept. The purpose of the second conditional branch is that these two conditions might have been violated within the last step.

Eventually, a for path that needs to be evaluated, the terminal cost for each signal group is estimated and accumulated, using equation 5 (see chapter 2.3).

Comment 5.0.26

The final product, is compared to the current UB. If it is smaller, then it replaces the current UB and the path becomes the current optimal. The path TC, the number of the last step and the path steps, are stored, in order to be displayed in the simulation (see "interfaceSL" module described later). Finally, the stage durations resulting from this path are calculated by using the "STstateselection" subroutine, to be transferred to the controller. Note that, the accumulation of the stage states for the current stage is done only for the first 40 seconds and not for the whole horizon. This is done because otherwise, the duration given to this stage after a whole cycle could be added and this could result in giving larger extensions than feasible.

5.1 "bbcontinue" subroutine

The calculations are the same as those described in comments 5.0.6 – 5.0.14, with the difference that the starting time used is not 1, but "tend+1" (where tend the ending time of the previous step) and the ending time is not 5 but "tend+5". Moreover, the subroutine is built in a way to be able to cope with paths ending after the end of the horizon. This can result if the stage transitions have durations which are not integer multiples of the time step (for more information see chapter 3.4). This is not the case in this strategy, however, the feature was maintained in case the strategy needs to be implemented with different stage transitions.

5.2 "bbswitch" subroutine

The calculations are similar to the ones described in comments 5.0.15 – 5.0.23. However, the starting time used, is again equal to "tend+1" while the ending time is equal to the previous ending time plus the transition duration. The same feature as for the "bbcontinue" module exists here (the ability to truncate the cost estimation at the 80^{th} step, if the path exceeds the horizon). However, this feature does not become active using the existing stage transitions.

5.3 “pathindicator” subroutine

This subroutine is used to enable the display of the optimal path in the simulation environment. Each path that becomes optimal while the BB module runs is stored (and replaces the previous) using this module. At the end of the optimisation, the “interfaceSL” module, asks for the optimal path from this module. Its function is simple: all the steps used in the optimal path are stored in the corresponding “opt_step” variables, while the “opt_step” variables that are not used take the value 999. To check how many steps were used, the “laststep” variable (see BB module) is used.

6.1 “Initial values” module

This module simply sets some initial values in the program. Firstly, the initial turning percentages for the intersection are set, equal to the VISSIM tuning percentages. These will be updated from the TP module the first time it runs. Secondly, the number of lanes corresponding to each signal group are stored. Thirdly, the critical queues, above which the gating feature discussed in chapter 3.1 is enabled, are defined. Fourthly, the powers of 2, used in by equation 4 in the BB module are stored here (as TL does not have a power calculation feature).

7.1 “Interface SL” module

This module serves for transferring variables from TL in the simulation environment. It is not a module related to the optimisation but a communication interface between VISSIM and TL. All variables that need to be transferred from TL to VISSIM need to be stored in the “ANW_VAR_AL” variable. This variable is a vector with 128 elements, the 127 of which are available to the user. Thus, up to 127 variables can be transferred from TL to VISSIM.

Please note that the order of the elements presented here (which is the order they are sorted in the TL code) is not the same as the order these variables are sorted in VISSIM! Thus, element “i” does not correspond to the i^{th} column of the VISSIM output window. However, in VISSIM, each output variable is identified by its name.

The first nine elements are used for the stage durations.
Elements 10-18 are used for the arrival time of the last vehicle that actuated the upstream detector (per signal group).
Elements 19-27 are used to display the estimate of the existing queue in each signal group.
Element 28 is used to display the UB of the optimal path (divided by 10)
Element 29 is used to display the TC of the optimal path
Elements 30-45 are used to display the steps of the optimal path. Each element shows whether the respective step was 0 (transition) or 1 (extend stage).
Elements 46-63 are used to display some aggregate outputs from the TE module. Every two elements correspond to a signal group (i.e. 46 and 47 to the first, 48 and 49 to the second, etc). For each signal group, the first element displays the sum of the arrivals in the short term horizon and the second element displays the average arrivals (per second) for the long term horizon.

Element 64 is used for displaying the elapsed duration of the stage that is running.
Elements 65-68 are used to display the actuations of the upstream detectors of an approach in the last second.
Elements 69-72 are used to display the actuations of the upstream detectors of an approach during the last 60 seconds.
Elements 73-76 are used to display the average occupancy of the upstream detectors of an approach during the last 60 seconds.
Elements 77-80 are used to display the actual number of vehicles present between the upstream and stop-line detectors of an approach at any second.
Elements 81-84 are used to display the estimated speed (in km/h), assigned to each approach.
Elements 85-93, are used to display the clear distance (not occupied by queue) between the upstream detectors and the back of the queue, per signal group.

8.1 Anwenderfunktion, AnwenderInit, Hauptfunktion

Those three functions are built-in in TL and need to be called in every algorithm developed with this software. They are not related to the logic developed in this Thesis. In the first function, the modules that need to run and the order that they are called is defined. The second and the third functions are called to initialise the control process. In the second function, the call of the TE and SM modules are also defined.

Annex 4 – TL code

Annex 4 includes the programming code written in Siemens Traffic Language. The first 6 pages show the variables and the parameters used in the code. The strategy modules are shown in pages 7-45.

Project:strategy.tlp

File list:

c:\sitraf~1\control\thesis\pireos4.mod
c:\va\english\language\sybib\s-l60701\s_l60701.bib
c:\va\english\language\sybib\ker60701\ker60701.bib
s-l_103.qel

Description:

Parameter:

```
(Identifier)          (Category)
    (Groups)              (Parameter,           Type)

AppDet
     0: App1               0: DET1              WORT
     1: App2               1: DET2              WORT
     2: App3               2: DET3              WORT
     3: App4               3: DET4              WORT
                           4: DET5              WORT
                           5: DET6              WORT
                           6: DET7              WORT

Applanes
     0: App1               0: lanes             WORT
     1: App2
     2: App3
     3: App4

AppSG
     0: App1               0: SG1               WORT
     1: App2               1: SG2               WORT
     2: App3               2: SG3               WORT
     3: App4               3: SG4               WORT

Detdist
     0: App1               0: detdist           WORT
     1: App2
     2: App3
     3: App4

SGDet
     0: SG1                0: DET1              WORT
     1: SG2                1: DET2              WORT
     2: SG3                2: DET3              WORT
     3: SG4
     4: SG5
     5: SG6
     6: SG7
     7: SG8
     8: SG9

STExtension
     0: G1                 0: max               WORT
     1: G2
     2: G3
     3: G4
     4: G5

STSG
     0: Stage1             0: SG1               WORT
     1: Stage2             1: SG2               WORT
     2: Stage3             2: SG3               WORT
     3: Stage4
     4: Stage5

TransDur
     0: G1                 0: TRdur             WORT
     1: G2
     2: G3
     3: G4
     4: G5

TransSG
     0: G1                 0: SG1               WORT
     1: G2                 1: SG2               WORT
     2: G3                 2: SG3               WORT
     3: G4                 3: SG4               WORT
     4: G5                 4: SG5               WORT
                           5: SG6               WORT
                           6: SG7               WORT
                           7: SG8               WORT
```

8: SG9 WORT

Variables:

(Identifier)	(Type)	(Dimensions)	(Category)
(Description, optional)			
a	BYTE		
a2	VBYTE		
actualqueue	VWORT	[5]	
approach	BYTE		
appsg	WORT	[5]	
AR	WORT	[10]	
arrivals1	WORT	[91]	
arrivals2	WORT	[91]	
arrivals3	WORT	[91]	
arrivals4	WORT	[91]	
arrivals5	WORT	[91]	
arrivals6	WORT	[91]	
arrivals7	WORT	[91]	
arrivals8	WORT	[91]	
arrivals9	WORT	[91]	
b	BYTE		
clear	WORT	[10]	
counts2	WORT	[5]	
counts3	WORT	[5]	
counts4	WORT	[5]	
counts5	WORT	[5]	
counts6	WORT	[5]	
criticalqueue	WORT	[5]	
cumcost	VLWORT		
cumcost2	VLWORT	[10]	
cumcost3	VLWORT	[10]	
cumcost6	VLWORT	[10]	
cumcost7	VLWORT	[10]	
cumcost8	VLWORT	[10]	
curcost	VLWORT	[10]	
currentdet	WORT		
curST	VWORT		
DE	WORT	[10]	
DEprev	WORT	[10]	
DET	BYTE		
DET2	WORT		
detcount	WORT		
detocc	WORT	[89]	
detocctemp	WORT	[89]	
detocctemp2	WORT	[89]	
drcount	BYTE		
dur	BYTE		
elapsdur	WORT	[6]	
existAppqu	WORT	[5]	
existqu	WORT	[10]	
factor	WORT	[5]	
gap	WORT	[3]	
i	BYTE		
iapp	BYTE		
ii	BYTE		
ind	BYTE		
isg	WORT		
ist	WORT		
ist2	WORT		
ist3	WORT		
j	BYTE		
jdet	BYTE		
k	BYTE		
kdet	BYTE		
komma	GLKOMMA		
laststep	BYTE		
laststep2	BYTE		
LB	LWORT		
long	BYTE		
longstart	WORT	[10]	
longstart2	WORT	[10]	
maxminext	WORT		
mincyclegreen	WORT	[10]	

minext	WORT	[10]
mingap	WORT	[19]
mingreen	WORT	
mingreeni	WORT	
multi	WORT	[10]
n_decimal	WORT	
n_integer	WORT	
next1	BYTE	
next8	BYTE	
next900	BYTE	
nextST	WORT	
now	WORT	
occapp	WORT	[5]
occapptemp	WORT	[5]
opt_step1	WORT	
opt_step10	WORT	
opt_step11	WORT	
opt_step12	WORT	
opt_step13	WORT	
opt_step14	WORT	
opt_step15	WORT	
opt_step16	WORT	
opt_step2	WORT	
opt_step3	WORT	
opt_step4	WORT	
opt_step5	WORT	
opt_step6	WORT	
opt_step7	WORT	
opt_step8	WORT	
opt_step9	WORT	
opt_t	BYTE	
outapp	WORT	[5]
outdet	VWORT	
outdetprev	VWORT	[89]
partialveh	WORT	[10]
parts	WORT	
pathcost	LWORT	
power2_dec	WORT	[11]
power2_int	WORT	[10]
queue	WORT	[10]
run	WORT	
SGlanes	WORT	[10]
SGstate	BYTE	[10]
SGstate2	BYTE	[10]
short	BYTE	
skip	VBYTE	
sstate	BYTE	[10]
STduration	WORT	[6]
step1	BYTE	
step10	BYTE	
step11	BYTE	
step12	BYTE	
step13	BYTE	
step14	BYTE	
step15	BYTE	
step16	BYTE	
step17	BYTE	
step18	BYTE	
step19	BYTE	
step2	BYTE	
step20	BYTE	
step3	BYTE	
step4	BYTE	
step5	BYTE	
step6	BYTE	
step7	BYTE	
step8	BYTE	
step9	BYTE	
STmaxremain	WORT	
stoplinesum	WORT	
STstate1	WORT	[100]
STstate2	WORT	[100]
STstate3	WORT	[100]

```
STstate4             WORT            [100]
STstate5             WORT            [100]
t                    BYTE
t1                   BYTE
t2                   BYTE
t3                   BYTE
t4                   BYTE
take                 BYTE            [89]
take2                BYTE
tarriv               WORT            [10]
tarrivtemp           WORT            [10]
TC                   VLWORT
TC2                  VLWORT
tend                 WORT
tend2                WORT
tparts               WORT            [10]
TPdet1               WORT            [6]
TPdet2               WORT            [6]
TPdet3               WORT            [6]
TPdet4               WORT            [6]
TPdet5               WORT            [6]
TPSG                 WORT            [10]
TPSGtemp             WORT            [10]
tstart               WORT
UB                   LWORT
value                WORT
value2               WORT
value4               WORT
vehapp               WORT            [5]
vehapp80             WORT            [5]
vehdet               VWORT
vehdet80             VWORT           [89]
vehdet80temp         WORT            [89]
vehdetprev           VWORT           [89]
vehdetprev2          VWORT           [89]
vel                  WORT            [5]
VISSIM_mexwa_interval BYTE
www                  WORT
www2                 LWORT
www3                 VWORT
www4                 VLWORT
```

Function:_arrivalselection

Category:
Result: LEER
Input: --
Output: --

Description:

```
isg
  = 1
    value2:=arrivals1[t2]
  = 2
    value2:=arrivals2[t2]
  = 3
    value2:=arrivals3[t2]
  = 4
    value2:=arrivals4[t2]
  = 5
    value2:=arrivals5[t2]
  = 6
    value2:=arrivals6[t2]
  = 7
    value2:=arrivals7[t2]
  = 8
    value2:=arrivals8[t2]
  = 9
    value2:=arrivals9[t2]
  SONST
```

Function:_arrivalupdate

Category:
Result: LEER
Input: --
Output: --

Description:

```
isg
  = 1
    arrivals1[t]:=value
  = 2
    arrivals2[t]:=value
  = 3
    arrivals3[t]:=value
  = 4
    arrivals4[t]:=value
  = 5
    arrivals5[t]:=value
  = 6
    arrivals6[t]:=value
  = 7
    arrivals7[t]:=value
  = 8
    arrivals8[t]:=value
  = 9
    arrivals9[t]:=value
  SONST
```

Function:_arrivalupdate1

Category:
Result: LEER
Input: --
Output: --

Description:

```
isg
  = 1
    arrivals1[t-1]:=arrivals1[t]
    arrivals1[t]:=value
  = 2
    arrivals2[t-1]:=arrivals2[t]
    arrivals2[t]:=value
  = 3
    arrivals3[t-1]:=arrivals3[t]
    arrivals3[t]:=value
  = 4
    arrivals4[t-1]:=arrivals4[t]
    arrivals4[t]:=value
  = 5
    arrivals5[t-1]:=arrivals5[t]
    arrivals5[t]:=value
  = 6
    arrivals6[t-1]:=arrivals6[t]
    arrivals6[t]:=value
  = 7
    arrivals7[t-1]:=arrivals7[t]
    arrivals7[t]:=value
  = 8
    arrivals8[t-1]:=arrivals8[t]
    arrivals8[t]:=value
  = 9
    arrivals9[t-1]:=arrivals9[t]
    arrivals9[t]:=value
  SONST
```

Function:_arrivalupdate2

Category:
Result: LEER
Input: --
Output: --

Description:

```
isg
  = 1
    arrivals1[t]:=arrivals1[t]+value
  = 2
    arrivals2[t]:=arrivals2[t]+value
  = 3
    arrivals3[t]:=arrivals3[t]+value
  = 4
    arrivals4[t]:=arrivals4[t]+value
  = 5
    arrivals5[t]:=arrivals5[t]+value
  = 6
    arrivals6[t]:=arrivals6[t]+value
  = 7
    arrivals7[t]:=arrivals7[t]+value
  = 8
    arrivals8[t]:=arrivals8[t]+value
  = 9
    arrivals9[t]:=arrivals9[t]+value
  SONST
```

Function:_BB_BranchandBound

Category:
Result: LEER
Input: --
Output: --

Description:

```
Branch and Bound module
COMMENT 5.0.1
ist := 1 ... 5
  STduration[ist]:=0
  PHA_INFO(CHG_WORT_TO_BYTE(ist),0)=1
  Y
    curST:=CHG_WORT_TO_VWORT(ist)
  N
  PHA_INFO(0,CHG_WORT_TO_BYTE(ist))=16
  Y
    curST:=-1
    STduration[ist]:=999
  N
COMMENT 5.0.2
curST<>-1
Y
  UB:=4294967295
  COMMENT 5.0.3
  step1 := 0 ... 1
   step2 := 0 ... 1
    step3 := 0 ... 1
     step4 := 0 ... 1
      step5 := 0 ... 1
       step6 := 0 ... 1
        step7 := 0 ... 1
         step8 := 0 ... 1
          step9 := 0 ... 1
           step10 := 0 ... 1
            step11 := 0 ... 1
             step12 := 0 ... 1
              step13 := 0 ... 1
               step14 := 0 ... 1
                step15 := 0 ... 1
                 step16 := 0 ... 1
                  COMMENT 5.0.4
                  ist := 1 ... 5
                   PHA_INFO(CHG_WORT_TO_BYTE(ist),0)=1
                   Y
                    curST:=CHG_WORT_TO_VWORT(ist)
                    ist
                     = 1
                      elapsdur[curST]:=SG_ZUSTAND_AL(6,TFRB,-1)/10 - PARA TransSG.G5.SG6
                     = 2
                      elapsdur[curST]:=SG_ZUSTAND_AL(8,TFRB,-1)/10 - PARA TransSG.G1.SG8
                     = 3
                      elapsdur[curST]:=SG_ZUSTAND_AL(7,TFRB,-1)/10 - PARA TransSG.G2.SG7
                     = 4
                      elapsdur[curST]:=SG_ZUSTAND_AL(3,TFRB,-1)/10 - PARA TransSG.G3.SG3
                     = 5
                      elapsdur[curST]:=SG_ZUSTAND_AL(1,TFRB,-1)/10 - PARA TransSG.G4.SG1
```

```
SONST

N

COMMENT 5.0.5
LB:=0
pathcost:=0
cumcost:=0
t := 1 ... 80
  t3:=t
  ist := 1 ... 5
    ist2:=ist
    value4:=0
    _STstateupdate()
isg := 1 ... 9
  queue[isg]:=existqu[isg]
COMMENT 5.0.6
step1
  = 1
    COMMENT 5.0.7
    tstart:=1
    tend:=5
    t := CHG_WORT_TO_BYTE(tstart) ... CHG_WORT_TO_BYTE(tend)
      t3:=t
      ist := 1 ... 5
        ist2:=ist
        value4:=0
        _STstateupdate()
      ist2:=curST
      value4:=1
      _STstateupdate()
    COMMENT 5.0.8
    cumcost:=0
    isg := 1 ... 9
      t:=CHG_WORT_TO_BYTE(tstart)
      AR[isg]:=0
      DE[isg]:=0
      COMMENT 5.0.9
      t<tend+1
        t2:=t
        _arrivalselection()
        AR[isg]:=AR[isg]+value2
        t:=t+1
      COMMENT 5.0.10
      SGstate[isg]:=0
      i := 1 ... 3
        isg=PARA STSG[curST-1][i-1]
        Y SGstate[isg]:=1
        N
      COMMENT 5.0.11
      DE[isg]:=SGstate[isg]*5*52*SGlanes[isg]
      DE[isg]>AR[isg]+queue[isg]
      Y DE[isg]:=AR[isg]+queue[isg]
      N
      COMMENT 5.0.12
      curcost[isg]:=(queue[isg]*10*(tend-tstart+1)+(AR[isg]-DE[isg])*((10*(tend-tstart+1))/2))/100
```

```
curcost[isg]<0
Y  curcost[isg]:=0
N
cumcost:=cumcost+curcost[isg]
COMMENT 5.0.13
queue[isg]+AR[isg]-DE[isg]<0
Y  queue[isg]:=0
N  queue[isg]:=queue[isg]+AR[isg]-DE[isg]
COMMENT 5.0.14
pathcost:=LB+cumcost
LB:=LB+pathcost
elapsdur[curST]:=elapsdur[curST]+5
= 0
COMMENT 5.0.15
COMMENT 5.0.16
curST=5
Y  nextST:=1
N  nextST:=curST+1
elapsdur[nextST]:=0
tstart:=1
tend:=PARA TransDur[curST-1].TRdur
t := CHG_WORT_TO_BYTE(tstart) ... CHG_WORT_TO_BYTE(tend)
  t3:=t
  ist := 1 ... 5
    ist2:=ist
    value4:=0
    _STstateupdate()
COMMENT 5.0.17
tend-(tstart-1)>5
Y  parts:=(tend-(tstart-1))/5
   (tend-(tstart-1))-5*parts>0
   Y  parts:=(tend-(tstart-1))/5+1
      tparts[1]:=(tstart-1)+5
      i := 2 ... parts-1
        tparts[i]:=tparts[i-1]+5
      tparts[parts]:=tend
   N  tparts[1]:=(tstart-1)+5
      i := 2 ... parts
        tparts[i]:=tparts[i-1]+5
N  parts:=1
   tparts[1]:=tend
COMMENT 5.0.18
ii := 1 ... 9
  cumcost2[ii]:=0
ii := 1 ... parts
  isg := 1 ... 9
    COMMENT 5.0.19
    ii=1
    Y  t:=CHG_WORT_TO_BYTE(tstart)
```

```
t:=CHG_WORT_TO_BYTE(tparts[ii-1]+1)
N
AR[isg]:=0
DE[isg]:=0
t<tparts[ii]+1
  t2:=t
  _arrivalselection()
  AR[isg]:=AR[isg]+value2
  t:=t+1
SGstate[isg]:=0
SGstate2[isg]:=0
i := 1 ... 3
  isg=PARA STSG[nextST-1][i-1]
  Y SGstate[isg]:=1
  N isg=PARA STSG[curST-1][i-1]
    Y SGstate2[isg]:=1
    N
COMMENT 5.0.20
ii=1
Y DEprev[isg]:=0
N
PARA TransSG[curST-1][isg-1]>0
Y SGstate[isg]=1
  Y (tstart-1)+PARA TransDur[curST-1].TRdur-PARA TransSG[curST-1][isg-1]>tparts[ii]
    Y DE[isg]:=0
    N DE[isg]:=(tparts[ii]-((tstart-1)+PARA TransDur[curST-1].TRdur-PARA TransSG[curST-1][isg-1]))*52*SGlanes[isg]
  N SGstate2[isg]=1
    Y (tstart-1)+PARA TransSG[curST-1][isg-1]>tparts[ii]
      Y DE[isg]:=(tparts[ii]-tstart+1)*52*SGlanes[isg]
      N DE[isg]:=PARA TransSG[curST-1][isg-1]*52*SGlanes[isg]
    N
N DE[isg]:=0
DE[isg]:=DE[isg]-DEprev[isg]
DEprev[isg]:=DE[isg]
DE[isg]>AR[isg]+queue[isg]
Y DE[isg]:=AR[isg]+queue[isg]
N
COMMENT 5.0.21
ii=1
Y curcost[isg]:=(queue[isg]*10*(tparts[ii]-tstart+1)+(AR[isg]-DE[isg])*((10*(tparts[ii]-tstart+1))/2))/100
N curcost[isg]:=(queue[isg]*10*(tparts[ii]-tparts[ii-1])+(AR[isg]-DE[isg])*((10*(tparts[ii]-tparts[ii-1]))/2))/100
curcost[isg]<0
Y curcost[isg]:=0
N
cumcost2[ii]:=cumcost2[ii]+curcost[isg]
queue[isg]+AR[isg]-DE[isg]<0
Y queue[isg]:=0
```

```
queue[isg]:=queue[isg]+AR[isg]-DE[isg]
N
COMMENT 5.0.22
n_integer:=(tend-(tstart-1))/5
i:=1
parts-i>=0
  multi[i]:=power2_int[n_integer-i+1]
  i:=i+1
  multi[1]=2^n         i.e. 2^3
  multi[2]=2^(n-1)   i.e. 2^2
  ...
  multi[n]=2^(n-i)     i.e. 2^0
i:=1
ii:=1
LB:=multi[i]*LB
i+1<=parts
  LB:=LB+multi[i+1]*cumcost2[ii]
  i:=i+1
  ii:=ii+1
COMMENT 5.0.23
LB:=LB+cumcost2[ii]
curST:=CHG_WORT_TO_VWORT(nextST)
elapsdur[curST]:=0
SONST
COMMENT 5.0.24
laststep:=0
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
step2
  = 1
    _bb_continue()
  = 0
    _bb_switch()
  SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
step3
  = 1
    _bb_continue()
  = 0
    _bb_switch()
  SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
step4
  = 1
    _bb_continue()
  = 0
    _bb_switch()
```

```
SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
  step5
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
  step6
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
  step7
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y laststep:=7
  step8
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y laststep=0
  Y laststep:=8
  N
  step9
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
```

```
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y  laststep=0
   Y  laststep:=9
   N
   step10
      = 1
         _bb_continue()
      = 0
         _bb_switch()
      SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y  laststep=0
   Y  laststep:=10
   N
   step11
      = 1
         _bb_continue()
      = 0
         _bb_switch()
      SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y  laststep=0
   Y  laststep:=11
   N
   step12
      = 1
         _bb_continue()
      = 0
         _bb_switch()
      SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y  laststep=0
   Y  laststep:=12
   N
   step13
      = 1
         _bb_continue()
      = 0
         _bb_switch()
      SONST
N
```

```
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
  laststep=0
  Y
    laststep:=13
  N
  step14
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
  laststep=0
  Y
    laststep:=14
  N
  step15
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
tend>=80 ODER LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
Y
  laststep=0
  Y
    laststep:=15
  N
  laststep:=16
  step16
    = 1
      _bb_continue()
    = 0
      _bb_switch()
    SONST
N
  COMMENT 5.0.25
tend>=80
Y
  LB>=UB ODER elapsdur[curST]>PARA STExtension[curST-1].max
  Y
    TC:=0
    isg := 1 ... 9
        **
        Saturation flow for the TC assumed 0.5 veh/sec
        r(g) assumed 0,25
        The 10000 is: 2*0.5*10000 (the 10000 is to balance the queue being 100ple)
        Then last term is: queue*min red for the sg.
        So all the TC is finally expressed in veh-sec, just like the LB
      TC:=TC+(((queue[isg]*queue[isg])/(10000))*((100-SGstate[isg]*100)+25*SGstate[isg]))/10 + (queue[isg]*(1-SGstate[isg])*(55-mincyclegreen[isg]-SG_~
      ZUSTAND_AL(CHG_WORT_TO_BYTE(isg),TFRE,-1)/10))/10
      COMMENT 5.0.26
    TC+LB<UB
```

```
Y
UB:=TC+LB
TC2:=TC
laststep2:=laststep
_pathindicator()
ist := 1 ... 5
  STduration[ist]:=0
  PHA_INFO(CHG_WORT_TO_BYTE(ist),0)=1
  Y
    ist3:=ist
    t := 1 ... 40
      t4:=t
      _STstateselection()
      STduration[ist]:=STduration[ist]+value4

    ist3:=ist
    t := 1 ... 80
      t4:=t
      _STstateselection()
      STduration[ist]:=STduration[ist]+value4
  N
N
N
N
N
```

Function:_bb_continue

Category:
Result: LEER
Input: --
Output: --

Description:

```
The duration of the step is calculated:
tstart:=tend+1
tend:=tend+5
tend>80
Y  tend2:=80
N  tend2:=tend
The signal states of this step are defined:
t := CHG_WORT_TO_BYTE(tstart) ... CHG_WORT_TO_BYTE(tend2)
  t3:=t
  ist := 1 ... 5
    ist2:=ist
    value4:=0
    _STstateupdate()
  ist2:=curST
  value4:=1
  _STstateupdate()
The following lines calculate the cost of the switching step:
cumcost:=0
isg := 1 ... 9
  t:=CHG_WORT_TO_BYTE(tstart)
  AR[isg]:=0
  DE[isg]:=0
  t<tend2+1
    t2:=t
    _arrivalselection()
    AR[isg]:=AR[isg]+value2
    t:=t+1
  SGstate[isg]:=0
  i := 1 ... 3
    isg=PARA STSG[curST-1][i-1]
    Y  SGstate[isg]:=1
    N
  DE[isg]:=SGstate[isg]*(tend2-tstart+1)*52*SGlanes[isg]
  DE[isg]>AR[isg]+queue[isg]
  Y  DE[isg]:=AR[isg]+queue[isg]
  N
  curcost[isg]:=(queue[isg]*10*(tend2-tstart+1)+(AR[isg]-DE[isg])*((10*(tend2-tstart+1))/2))/100
  curcost[isg]<0
  Y  curcost[isg]:=0
  N
  cumcost:=cumcost+curcost[isg]
  queue[isg]+AR[isg]-DE[isg]<0
  Y  queue[isg]:=0
  N  queue[isg]:=queue[isg]+AR[isg]-DE[isg]
```

```
LB:=2*LB+cumcost
elapsdur[curST]:=elapsdur[curST]+(tend2-tstart+1)
```

Function:_bb_switch

Category:
Result: LEER
Input: --
Output: --

Description:

```
curST=5
  Y: nextST:=1
  N: nextST:=curST+1
elapsdur[nextST]:=0
tstart:=tend+1
tend:=tend+PARA TransDur[curST-1].TRdur
tend>80
  Y: tend2:=80
  N: tend2:=tend
t := CHG_WORT_TO_BYTE(tstart) ... CHG_WORT_TO_BYTE(tend)
  t3:=t
  ist := 1 ... 5
    ist2:=ist
    value4:=0
    _STstateupdate()
tend2<tend
  Y: t := CHG_WORT_TO_BYTE(tend2+1) ... CHG_WORT_TO_BYTE(tend)
       t3:=t
       ist := 1 ... 5
         ist2:=ist
         value4:=0
         _STstateupdate()
  N:
tend2-(tstart-1)>5
  Y: parts:=(tend2-(tstart-1))/5
     (tend2-(tstart-1))-5*parts>0
       Y: parts:=(tend2-(tstart-1))/5+1
          tparts[1]:=(tstart-1)+5
          i := 2 ... parts-1
            tparts[i]:=tparts[i-1]+5
          tparts[parts]:=tend2
       N: tparts[1]:=(tstart-1)+5
          i := 2 ... parts
            tparts[i]:=tparts[i-1]+5
  N: parts:=1
     tparts[1]:=tend2
ii := 1 ... 9
  cumcost2[ii]:=0
ii := 1 ... parts
  isq := 1 ... 9
```

```
ii=1
  Y: t:=CHG_WORT_TO_BYTE(tstart)
  N: t:=CHG_WORT_TO_BYTE(tparts[ii-1]+1)
AR[isg]:=0
DE[isg]:=0
t<tparts[ii]+1
  t2:=t
  _arrivalselection()
  AR[isg]:=AR[isg]+value2
  t:=t+1
SGstate[isg]:=0
SGstate2[isg]:=0
i := 1 ... 3
  isg=PARA STSG[nextST-1][i-1]
    Y: SGstate[isg]:=1
    N: isg=PARA STSG[curST-1][i-1]
         Y: SGstate2[isg]:=1
         N:
ii=1
  Y: DEprev[isg]:=0
  N:
PARA TransSG[curST-1][isg-1]>0
  Y: SGstate[isg]=1
       Y: (tstart-1)+PARA TransDur[curST-1].TRdur-PARA TransSG[curST-1][isg-1]>tparts[ii]
            Y: DE[isg]:=0
            N: DE[isg]:=(tparts[ii]-((tstart-1)+PARA TransDur[curST-1].TRdur-PARA TransSG[curST-1][isg-1]))*52*SGlanes[isg]
       N: SGstate2[isg]=1
            Y: (tstart-1)+PARA TransSG[curST-1][isg-1]>tparts[ii]
                 Y: DE[isg]:=(tparts[ii]-tstart+1)*52*SGlanes[isg]
                 N: DE[isg]:=PARA TransSG[curST-1][isg-1]*52*SGlanes[isg]
            N:
  N: DE[isg]:=0
DE[isg]:=DE[isg]-DEprev[isg]
DEprev[isg]:=DE[isg]
DE[isg]>AR[isg]+queue[isg]
  Y: DE[isg]:=AR[isg]+queue[isg]
  N:
ii=1
  Y: curcost[isg]:=(queue[isg]*10*(tparts[ii]-tstart+1)+(AR[isg]-DE[isg])*((10*(tparts[ii]-tstart+1))/2))/100
  N: curcost[isg]:=(queue[isg]*10*(tparts[ii]-tparts[ii-1])+(AR[isg]-DE[isg])*((10*(tparts[ii]-tparts[ii-1]))/2))/100
curcost[isg]<0
  Y: curcost[isg]:=0
  N:
cumcost2[ii]:=cumcost2[ii]+curcost[isg]
queue[isg]+AR[isg]-DE[isg]<0
```

```
Y  queue[isg]:=0

   queue[isg]:=queue[isg]+AR[isg]-DE[isg]
N

n_integer:=(tend2-(tstart-1))/5
i:=1
parts-i>=0
  multi[i]:=power2_int[n_integer-i+1]
  i:=i+1

  multi[1]=2^n         i.e. 2^3
  multi[2]=2^(n-1)   i.e. 2^2
  ...
  multi[n]=2^(n-i)     i.e. 2^0
i:=1
ii:=1
LB:=multi[i]*LB
i+1<=parts
  LB:=LB+multi[i+1]*cumcost2[ii]
  i:=i+1
  ii:=ii+1

LB:=LB+cumcost2[ii]
curST:=CHG_WORT_TO_VWORT(nextST)
elapsdur[curST]:=0
```

Function:_initialvalues

Category:
Result: LEER
Input: --
Output: --

Description:

```
TPSG[1]:=56
TPSG[2]:=51
TPSG[3]:=49
TPSG[4]:=37
TPSG[5]:=21
TPSG[6]:=20
TPSG[7]:=100
TPSG[8]:=42
TPSG[9]:=24
SGlanes[1]:=2
SGlanes[2]:=1
SGlanes[3]:=2
SGlanes[4]:=1
SGlanes[5]:=2
SGlanes[6]:=1
SGlanes[7]:=2
SGlanes[8]:=2
SGlanes[9]:=1
criticalqueue[1]:=42
criticalqueue[2]:=43
criticalqueue[3]:=80
criticalqueue[4]:=70
mincyclegreen[1]:=16
mincyclegreen[2]:=16
mincyclegreen[3]:=5
mincyclegreen[4]:=30
mincyclegreen[5]:=15
mincyclegreen[6]:=8
mincyclegreen[7]:=5
mincyclegreen[8]:=6
mincyclegreen[9]:=29
power2_int[0]:=1
power2_int[1]:=2
power2_int[2]:=4
power2_int[3]:=8
power2_int[4]:=16
power2_int[5]:=32
power2_int[6]:=64
power2_int[7]:=128
power2_int[8]:=256
power2_int[9]:=512
```

Function:_interfaceSL

Category:
Result: LEER
Input: --
Output: --

Description:

```
i := 1 ... 9
  i
    = 1
      ANW_VAR_AL[i-1]:=STduration[1]
    = 8
      ANW_VAR_AL[i-1]:=STduration[2]
    = 7
      ANW_VAR_AL[i-1]:=STduration[3]
    = 3
      ANW_VAR_AL[i-1]:=STduration[4]
    = 2
      ANW_VAR_AL[i-1]:=STduration[5]
    SONST
i := 10 ... 18
  ANW_VAR_AL[i-1]:=tarriv[i-9]
i := 19 ... 27
  ANW_VAR_AL[i-1]:=existqu[i-18]
ANW_VAR_AL[27]:=UB/10
ANW_VAR_AL[28]:=TC2
curST<>-1
Y
  ANW_VAR_AL[29]:=opt_step1
  ANW_VAR_AL[30]:=opt_step2
  ANW_VAR_AL[31]:=opt_step3
  ANW_VAR_AL[32]:=opt_step4
  ANW_VAR_AL[33]:=opt_step5
  ANW_VAR_AL[34]:=opt_step6
  ANW_VAR_AL[35]:=opt_step7
  ANW_VAR_AL[36]:=opt_step8
  ANW_VAR_AL[37]:=opt_step9
  ANW_VAR_AL[38]:=opt_step10
  ANW_VAR_AL[39]:=opt_step11
  ANW_VAR_AL[40]:=opt_step12
  ANW_VAR_AL[41]:=opt_step13
  ANW_VAR_AL[42]:=opt_step14
  ANW_VAR_AL[43]:=opt_step15
  ANW_VAR_AL[44]:=opt_step16
N
  i := 30 ... 45
    ANW_VAR_AL[i-1]:=999
isg := 1 ... 9
  isg
    = 1
      ANW_VAR_AL[45]:=0
      i := 1 ... tarriv[isg]
        ANW_VAR_AL[45]:=ANW_VAR_AL[45]+arrivals1[i]
```

```
ANW_VAR_AL[46]:=arrivals1[80]
= 2
ANW_VAR_AL[47]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[47]:=ANW_VAR_AL[47]+arrivals2[i]
ANW_VAR_AL[48]:=arrivals2[80]
= 3
ANW_VAR_AL[49]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[49]:=ANW_VAR_AL[49]+arrivals3[i]
ANW_VAR_AL[50]:=arrivals3[80]
= 4
ANW_VAR_AL[51]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[51]:=ANW_VAR_AL[51]+arrivals4[i]
ANW_VAR_AL[52]:=arrivals4[80]
= 5
ANW_VAR_AL[53]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[53]:=ANW_VAR_AL[53]+arrivals5[i]
ANW_VAR_AL[54]:=arrivals5[80]
= 6
ANW_VAR_AL[55]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[55]:=ANW_VAR_AL[55]+arrivals6[i]
ANW_VAR_AL[56]:=arrivals6[80]
= 7
ANW_VAR_AL[57]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[57]:=ANW_VAR_AL[57]+arrivals7[i]
ANW_VAR_AL[58]:=arrivals7[80]
= 8
ANW_VAR_AL[59]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[59]:=ANW_VAR_AL[59]+arrivals8[i]
ANW_VAR_AL[60]:=arrivals8[80]
= 9
ANW_VAR_AL[61]:=0
i := 1 ... tarriv[isg]
  ANW_VAR_AL[61]:=ANW_VAR_AL[61]+arrivals9[i]
ANW_VAR_AL[62]:=arrivals1[80]
SONST
ist := 1 ... 5
  PHA_INFO(CHG_WORT_TO_BYTE(ist),0)=1
 Y  curST:=CHG_WORT_TO_VWORT(ist)
    ist
```

```
= 1
  ANW_VAR_AL[63]:=SG_ZUSTAND_AL(6,TFRB,-1)/10 - PARA TransSG.G5.SG6
= 2
  ANW_VAR_AL[63]:=SG_ZUSTAND_AL(8,TFRB,-1)/10 - PARA TransSG.G1.SG8
= 3
  ANW_VAR_AL[63]:=SG_ZUSTAND_AL(7,TFRB,-1)/10 - PARA TransSG.G2.SG7
= 4
  ANW_VAR_AL[63]:=SG_ZUSTAND_AL(3,TFRB,-1)/10 - PARA TransSG.G3.SG3
= 5
  ANW_VAR_AL[63]:=SG_ZUSTAND_AL(1,TFRB,-1)/10 - PARA TransSG.G4.SG1
SONST
  ANW_VAR_AL[63]:=0
N
i := 65 ... 68
  ANW_VAR_AL[i-1]:=vehapp[i-64]
i := 69 ... 72
  ANW_VAR_AL[i-1]:=vehapp80[i-68]
i := 73 ... 76
  ANW_VAR_AL[i-1]:=occapp[i-72]
i := 77 ... 80
  ANW_VAR_AL[i-1]:=actualqueue[i-76]
i := 81 ... 84
  ANW_VAR_AL[i-1]:=vel[i-80]*36/100
i := 85 ... 93
  ANW_VAR_AL[i-1]:=clear[i-84]
```

Function:_pathindicator

Category:
Result: LEER
Input: --
Output: --

Description:

```
opt_step1:=step1
opt_step2:=step2
opt_step3:=step3
opt_step4:=step4
opt_step5:=step5
opt_step6:=step6
opt_step7:=step7
laststep2
  = 7
    opt_step8:=999
    opt_step9:=999
    opt_step10:=999
    opt_step11:=999
    opt_step12:=999
    opt_step13:=999
    opt_step14:=999
    opt_step15:=999
    opt_step16:=999

  = 8
    opt_step8:=step8
    opt_step9:=999
    opt_step10:=999
    opt_step11:=999
    opt_step12:=999
    opt_step13:=999
    opt_step14:=999
    opt_step15:=999
    opt_step16:=999

  = 9
    opt_step8:=step8
    opt_step9:=step9
    opt_step10:=999
    opt_step11:=999
    opt_step12:=999
    opt_step13:=999
    opt_step14:=999
    opt_step15:=999
    opt_step16:=999

  = 10
    opt_step8:=step8
    opt_step9:=step9
    opt_step10:=step10
    opt_step11:=999
    opt_step12:=999
    opt_step13:=999
    opt_step14:=999
    opt_step15:=999
    opt_step16:=999

  = 11
    opt_step8:=step8
    opt_step9:=step9
    opt_step10:=step10
```

```
opt_step11:=step11
opt_step12:=999
opt_step13:=999
opt_step14:=999
opt_step15:=999
opt_step16:=999

= 12
opt_step8:=step8
opt_step9:=step9
opt_step10:=step10
opt_step11:=step11
opt_step12:=step12
opt_step13:=999
opt_step14:=999
opt_step15:=999
opt_step16:=999

= 13
opt_step8:=step8
opt_step9:=step9
opt_step10:=step10
opt_step11:=step11
opt_step12:=step12
opt_step13:=step13
opt_step14:=999
opt_step15:=999
opt_step16:=999

= 14
opt_step8:=step8
opt_step9:=step9
opt_step10:=step10
opt_step11:=step11
opt_step12:=step12
opt_step13:=step13
opt_step14:=step14
opt_step15:=999
opt_step16:=999

= 15
opt_step8:=step8
opt_step9:=step9
opt_step10:=step10
opt_step11:=step11
opt_step12:=step12
opt_step13:=step13
opt_step14:=step14
opt_step15:=step15
opt_step16:=999

= 16
opt_step8:=step8
opt_step9:=step9
opt_step10:=step10
opt_step11:=step11
opt_step12:=step12
opt_step13:=step13
opt_step14:=step14
opt_step15:=step15
opt_step16:=step16

SONST
```

Function:_queuecheck1

Category:
Result: LEER
Input: --
Output: --

Description:

```
COMMENT 3.1.1
existqu[isg]>0
Y PARA SGDet[isg-1][0]<>88 UND PARA SGDet[isg-1][0]<>23 UND PARA SGDet[isg-1][0]<>17
  Y PARA SGDet[isg-1][1]<>88 UND PARA SGDet[isg-1][1]<>23 UND PARA SGDet[isg-1][1]<>17
    Y PARA SGDet[isg-1][2]<>88 UND PARA SGDet[isg-1][2]<>23 UND PARA SGDet[isg-1][2]<>17
      Y DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][0]),DLS)>30 UND
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][1]),DLS)>30 UND
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][2]),DLS)>30
        Y existqu[isg]:=0
        N
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][0]),DLS)>30 UND
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][1]),DLS)>30
        Y existqu[isg]:=0
        N
      N
      PARA SGDet[isg-1][2]<>88 UND PARA SGDet[isg-1][2]<>23 UND PARA SGDet[isg-1][2]<>17
      Y DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][0]),DLS)>30 UND
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][2]),DLS)>30
        Y existqu[isg]:=0
        N
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][0]),DLS)>30
        Y existqu[isg]:=0
        N
      N
    N
    PARA SGDet[isg-1][1]<>88 UND PARA SGDet[isg-1][1]<>23 UND PARA SGDet[isg-1][1]<>17
    Y PARA SGDet[isg-1][2]<>88 UND PARA SGDet[isg-1][2]<>23 UND PARA SGDet[isg-1][2]<>17
      Y DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][1]),DLS)>30 UND
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][2]),DLS)>30
        Y existqu[isg]:=0
        N
        DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][1]),DLS)>30
        Y existqu[isg]:=0
        N
      N
      PARA SGDet[isg-1][2]<>88 UND PARA SGDet[isg-1][2]<>23 UND PARA SGDet[isg-1][2]<>17
      Y DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][2]),DLS)>30
        Y existqu[isg]:=0
        N
      N
    N
  N
```

```
N
jdet := 0 ... 2
    The consistency check cannot be made for the detectors that are downstream from the stopline. Nor for the neutral detector.
  PARA SGDet[isg-1][jdet]<>88 UND
  PARA SGDet[isg-1][jdet]<>23 UND
  PARA SGDet[isg-1][jdet]<>17
 Y  DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA SGDet[isg-1][jdet]),DAU)>0 UND
    existqu[isg]=0 UND
    sstate[isg]=0
   Y  existqu[isg]:=100
   N
 N
```

Function:_SM_SignalMonitoring

Category:
Result: LEER
Input: --
Output: --

Description:

COMMENT 2.1

isg := 1 ... 8
- SG_ZUSTAND_AL(CHG_WORT_TO_BYTE(isg),FREI,-1)>0
  - Y: sstate[isg]:=1
  - N:
    - SG_ZUSTAND_AL(CHG_WORT_TO_BYTE(isg),SPERR,-1)>0
      - Y: sstate[isg]:=0
      - N:
        - SG_ZUSTAND_AL(CHG_WORT_TO_BYTE(isg),UGB_FR_SP,-1)>0
          - Y: sstate[isg]:=1
          - N:

COMMENT 2.2

SG_ZUSTAND_AL(16,SPERR,-1)>0
- Y: sstate[9]:=0
- N: sstate[9]:=1

Function:_STstateselection

Category:
Result: LEER
Input: --
Output: --

Description:

```
ist3
  = 1
    value4:=STstate1[t4]
  = 2
    value4:=STstate2[t4]
  = 3
    value4:=STstate3[t4]
  = 4
    value4:=STstate4[t4]
  = 5
    value4:=STstate5[t4]
  SONST
```

Function:_STstateupdate

Category:
Result: LEER
Input: --
Output: --

Description:

```
ist2
  = 1
    STstate1[t3]:=value4
  = 2
    STstate2[t3]:=value4
  = 3
    STstate3[t3]:=value4
  = 4
    STstate4[t3]:=value4
  = 5
    STstate5[t3]:=value4
  SONST
```

Function:_TE_TrafficEstimation

Category:
Result: LEER
Input: --
Output: --

Description:

Traffic estimation module

```
Traffic Estimation Module
COMMENT 3.0.1
isg := 1 ... 9
  tarriv[isg]>1
  Y longstart[isg]>60
    Y short := 2 ... 60
        t:=short
        value:=0
        _arrivalupdate1()

      short := 2 ... tarriv[isg]
        t:=short
        value:=0
        _arrivalupdate1()
    N

    t:=1
    value:=0
    _arrivalupdate()
  N

COMMENT 3.0.2
approach := 1 ... 4
  vehapp[approach]:=0
  outapp[approach]:=0
  COMMENT 3.0.3
  kdet := 0 ... 1
    vehdet:=0
    DET2:=PARA AppDet[approach-1][kdet]
    DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2),DZS)=0 UND vehdetprev[DET2]<>0
    Y vehdetprev2[DET2]:=vehdetprev[DET2]

    N
    # vehicles that passed the detector during the last second:
    vehdet:= CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS)-vehdetprev[PARA AppDet[approach-1][kdet]])
    COMMENT 3.0.4
    vehdet >= 0
    Y vehdetprev[PARA AppDet[approach-1][kdet]]:=CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS))

      vehdet:= 1
      vehdetprev[PARA AppDet[approach-1][kdet]]:= CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS))
    N
    COMMENT 3.0.5
    vehapp[approach]:=vehdet+vehapp[approach]
    COMMENT 3.0.6
    take[DET2]=60
    Y take[DET2]:=1
      kdet=0
      Y vehapp80[approach]:=0

      N
```

```
vehdet80[DET2]:=CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS)-vehdet80temp[DET2])
vehdet80[DET2] >= 0
Y   vehdet80temp[DET2]:=DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS)
N   vehdet80[DET2]:= CHG_WORT_TO_VWORT(vehdetprev2[DET2]-vehdet80temp[DET2]) + CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS))
    vehdet80temp[DET2]:= DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DZS)
vehapp80[approach]:=vehapp80[approach]+vehdet80[DET2]
take[DET2]:=take[DET2]+1
N
COMMENT 3.0.7
detocctemp[DET2]:=DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2), DBS)
detocctemp[DET2]>0
Y   occapptemp[approach]:=occapptemp[approach]+detocctemp[DET2]
N
take2=60
Y   kdet=1
    Y   occapp[approach]:=occapptemp[approach]/60
        occapptemp[approach]:=0
    N
N
COMMENT 3.0.8
DET2<>88
Y   gap[kdet+1]:=(DET_LESEN_AL(CHG_WORT_TO_BYTE(DET2),DLS))/10
    kdet=0
    Y   mingap[approach]:=gap[kdet+1]
    N   gap[kdet+1] < gap[kdet]
        Y   mingap[approach]:=gap[kdet+1]
        N
N
COMMENT 3.0.9
jdet := 2 ... 6
    outdet:=0
    outdet:= CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA AppDet[approach-1][jdet]), DZS)-outdetprev[PARA AppDet[approach-1][jdet]])
    No is for cases where the DZS has been reset to zero. Yes is for all other cases
    outdet >= 0
    Y   DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA AppDet[approach-1][jdet]), DAU)>0
        Y   outdet:=0
        N   outdetprev[PARA AppDet[approach-1][jdet]]:=CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA AppDet[approach-1][jdet]), DZS))
    N   DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA AppDet[approach-1][jdet]), DAU)>0
        Y   outdet:=0
        N   outdet:= CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA AppDet[approach-1][jdet]), DZS))
            outdetprev[PARA AppDet[approach-1][jdet]]:= CHG_WORT_TO_VWORT(DET_LESEN_AL(CHG_WORT_TO_BYTE(PARA AppDet[approach-1][jdet]), DZS))
    outapp[approach]:=outdet+outapp[approach]
COMMENT 3.0.10
actualqueue[approach]:=actualqueue[approach]+CHG_WORT_TO_VWORT(vehapp[approach])-CHG_WORT_TO_VWORT(outapp[approach])
actualqueue[approach]<0
Y   actualqueue[approach]:=0
N
```

```
The following loop finds the signal groups only for the current approach:
a:=1
COMMENT 3.0.11
isg := 0 ... 3
  appsg[a]:=PARA AppSG[approach-1][isg]
  a:=a+1

COMMENT 3.0.12
actualqueue[approach] > criticalqueue[approach]
Y
  As long as the approach remains congested, there is no need to predict their arrivals,
  as their queues will be set equal to max in the optimisation module.
  However, to keep producing an arrival matrix, I just use a constant 0,5veh/sec arrival rate for all their signal groups
  b:=1
  appsg[b]<>0
    t := 1 ... 90
      isg:=appsg[b]
      value:=60
      _arrivalupdate()

    b:=b+1


  COMMENT 3.0.13
  vehapp[approach] > 0 UND occapp[approach] > 0 UND vehapp80[approach]>0
  Y
    volume in veh, occupancy in %
    Assumptions: vehicle length= 4,5m, detector length= 1,5m, interval= 60 sec. vel in (m/s)*10 (because otherwise, in speeds less than 1m/s it will give 0 and the program will c~
    rash)
    80/100: reduction factor, to account for the speed reduction as the vehicle approaches the queue (determined after tests)
    vel[approach]:=(10*(10*vehapp80[approach])/occapp[approach])*80/100
    The partial vehicles are created and assigned to each SG
    b:=1
    appsg[b]<>0
      isg:=appsg[b]
      partialveh[isg]:=0
      partialveh[isg] := vehapp[approach] * TPSG[isg]
      PARA Detdist[approach-1].detdist-existqu[isg]*45/1000>0
      Y
        clear[isg]:=PARA Detdist[approach-1].detdist-existqu[isg]*45/1000

        clear[isg]:=0
      N
      vel[approach]<15
      Y
        vel[approach]:=15

      N
        clear is multiplied by 10 because speed was also multiplied by 10 before
      tarrivtemp[isg]:=(10*clear[isg])/vel[approach]
      tarrivtemp[isg]<=60
      UND
      tarrivtemp[isg]>0
      Y
        tarriv[isg]:=tarrivtemp[isg]
        t:=CHG_WORT_TO_BYTE(tarriv[isg])
        value:=partialveh[isg]
        _arrivalupdate2()

        tarriv[isg]:=tarrivtemp[isg]
      N
      b:=b+1


    If there were no actuations in the last second, just update the previous arrival times
    mingap[approach]<>0
    Y
      b:=1
      appsg[b]<>0
        isg:=appsg[b]
        tarriv[isg]>1
        Y
          tarriv[isg]:=tarriv[isg]-1
```

```
N
b:=b+1
N
N
b:=1
COMMENT 3.0.14
appsg[b]<>0
  isg:=appsg[b]
  The following lines provide the long-term horizon arrivals
  tarriv[isg]>0
  Y
    longstart[isg]:=tarriv[isg]+mingap[approach]+1
    longstart2[isg]:=mingap[approach]+1
  N
    longstart[isg]:=longstart2[isg]+1
  longstart[isg]>60
  Y
    long := 61 ... 90
      value:=(vehapp80[approach]*TPSG[isg])/60
      t:=long
      _arrivalupdate()
  N
    long := CHG_WORT_TO_BYTE(longstart[isg]) ... 90
      value:=(vehapp80[approach]*TPSG[isg])/60
      t:=long
      _arrivalupdate()
  b:=b+1
N
COMMENT 3.0.15
take2=60
Y
  take2:=0
N
take2:=take2+1
COMMENT 3.0.16
isg := 1 ... 9
  t2:=1
  _arrivalselection()
  (existqu[isg]+value2-sstate[isg]*52*SGlanes[isg]) < 0
  Y
    existqu[isg]:=0
  N
    existqu[isg]:=existqu[isg]+value2-sstate[isg]*52*SGlanes[isg]
  _queuecheck1()
COMMENT 3.0.17
approach := 1 ... 4
  existAppqu[approach]:=0
  actualqueue[approach]*4.5 < 0.75 * PARA Detdist[approach-1].detdist
  Y
    a:=1
    isg := 0 ... 3
      appsg[a]:=PARA AppSG[approach-1][isg]
      a:=a+1
    b:=1
    appsg[b]<>0
      isg:=appsg[b]
```

```
existAppqu[approach]:=existAppqu[approach]+existqu[isg]
b:=b+1
existAppqu[approach]>actualqueue[approach]*100
Y
factor[approach]:=100-((existAppqu[approach]-actualqueue[approach]*100)*100/existAppqu[approach])
b:=1
appsg[b]<>0
isg:=appsg[b]
existqu[isg]:=(factor[approach]*existqu[isg])/100
b:=b+1
N
N
```

Function:_TP_TurningPercentages

Category:
Result: LEER
Input: --
Output: --

Description:

Turning Percentages Module

```
COMMENT 4.1
iapp := 0 ... 3
  stoplinesum:=0
    COMMENT 4.2
  jdet := 2 ... 6
    detcount:=0
    DET:=CHG_WORT_TO_BYTE(PARA AppDet[iapp][jdet])
    detcount:=DET_LESEN_AL(DET,DZS)
    DET<>88
    Y  DET_LADEN(DET,DZS,0)
    N
    jdet
      = 2
        counts2[iapp+1]:=detcount
      = 3
        counts3[iapp+1]:=detcount
      = 4
        counts4[iapp+1]:=detcount
      = 5
        counts5[iapp+1]:=detcount
      = 6
        counts6[iapp+1]:=detcount
      SONST
    stoplinesum:=detcount+stoplinesum
    COMMENT 4.3
  j := 1 ... 5
    stoplinesum>0
    Y  j
        = 1
          TPdet1[iapp+1]:=(counts2[iapp+1]*100)/stoplinesum
        = 2
          TPdet2[iapp+1]:=(counts3[iapp+1]*100)/stoplinesum
        = 3
          TPdet3[iapp+1]:=(counts4[iapp+1]*100)/stoplinesum
        = 4
          TPdet4[iapp+1]:=(counts5[iapp+1]*100)/stoplinesum
        = 5
          TPdet5[iapp+1]:=(counts6[iapp+1]*100)/stoplinesum
        SONST
```

```
N
COMMENT 4.4
k := 1 ... 4
  isg:=PARA AppSG[iapp][k-1]
  isg>0
  Y
    TPSG[isg]:=0
    j := 1 ... 3
      currentdet := PARA SGDet[isg-1][j-1]
      jdet := 2 ... 6
        PARA AppDet[iapp][jdet] = currentdet
        Y
          jdet
            = 2
              TPSGtemp[isg]:= TPdet1[iapp+1]
              TPSG[isg]:=TPSG[isg]+TPSGtemp[isg]
            = 3
              TPSGtemp[isg]:= TPdet2[iapp+1]
              TPSG[isg]:=TPSG[isg]+TPSGtemp[isg]
            = 4
              TPSGtemp[isg]:= TPdet3[iapp+1]
              TPSG[isg]:=TPSG[isg]+TPSGtemp[isg]
            = 5
              TPSGtemp[isg]:= TPdet4[iapp+1]
              TPSG[isg]:=TPSG[isg]+TPSGtemp[isg]
            = 6
              TPSGtemp[isg]:= TPdet5[iapp+1]
              TPSG[isg]:=TPSG[isg]+TPSGtemp[isg]
            SONST
        N
  N
```

Function:Anwenderfunktion

Category:Haupt
Result: LEER
Input: --
Output: --

Description:

This function is called up by the control process. The function contains all TL-Functions which should run in the traffic controlled mode.

```
Functions called up here will work before the S-l logic runs.
Decisions in this functions can be set into user-variables.
The variables can be tested in the parameterized logic-cases.
__arrivaltest()
_BB_BranchandBound()
  __bbtest()
  run=900
  Y  _TP_TurningPercentages()
     run:=0
  N  run:=run+1
  __costtest()
_interfaceSL()
VAWLOGIKTEIL()
Functions called up here will work after the S-l logic has run.
Decisions in this functions can be set into user-variables.
The variables can be tested in the parameterized logic-cases
(next call up of the S-L logic).
```

Function:AnwenderInit

Category:Haupt
Result: LEER
Input: BYTE Initzeitpunkt
Output: --

Description:

function Initfunktion

This function has to be chanched as required.

```
wVALetzteSekunde
Y
    This variable specifies whether the user program
    is to be ended at the end of the signalisation
    interval. The user can use this information to
    reset influenced ports, for instance.
  FUNKTIONSENDE
N
  This function calls mustn't be removed
VAWFESTZEIT_EIN_AUS_PARA()
VAWFESTZEIT_EIN_AUS_DETSTOE()
  initialization
Initzeitpunkt
  = START_TL
      Initialization after power off,
      one call up after the parameters are complete supplied the first time
      or every second if only the TL-Parameter are not supplied
    _initialvalues()
  = PAS_VA
      start of the traffic controlled mode
    VISSIM_mexwa_interval := 8
  = PAS_SEK
      every second if traffic controlled mode
  SONST
_SM_SignalMonitoring()
_TE_TrafficEstimation()
```

Function:Hauptfunktion

Category:Haupt
Result: LEER
Input: --
Output: --

Description:

this function calls up the S-L main function

call up of the control process
VAWSTEUERUNG()
FUNKTIONSENDE