

UC Berkeley

Research Reports

Title

Development of an Integrated Microscopic Traffic Simulation and Signal Timing Optimization Tool

Permalink

<https://escholarship.org/uc/item/3r67f927>

Authors

Yin, Yafeng
Liu, Henry X.
Laval, Jorge A.
et al.

Publication Date

2007

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Development of an Integrated Microscopic Traffic Simulation and Signal Timing Optimization Tool

**Yafeng Yin, Henry X. Liu, Jorge A. Laval, Xiao-Yun Lu,
Meng Li, Joshua Pilachowski, Wei-Bin Zhang**

**California PATH Research Report
UCB-ITS-PRR-2007-2**

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation, and the United States Department of Transportation, Federal Highway Administration.

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California. This report does not constitute a standard, specification, or regulation.

Final Report for Task Order 5325

January 2007

ISSN 1055-1425

Development of an Integrated Microscopic Traffic Simulation and Signal Timing Optimization Tool

Yafeng Yin

Henry X. Liu

Jorge A. Laval

Xiao-Yun Lu

Meng Li

Joshua Pilachowski

Wei-Bin Zhang

ACKNOWLEDGEMENTS

This work was performed by the California PATH Program at the University of California at Berkeley, University of Florida and University of Minnesota in cooperation with the State of California Business, Transportation and Housing Agency, Department of Transportation (Caltrans). The contents of this report reflect the views of the authors, who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California.

The authors wish to thank Kai Leung, Jerry Kwong, Paul Chiu, James Lau, Lindy Cabugao and Sonja Sun of Caltrans for their continuing cooperation and support during the study. We also thank our colleagues, Prof. Alex Skabardonis and Dr. Kun Zhou of California PATH, for their help and discussion.

Author List

University of Florida

Yafeng Yin (primary author for Chapters 1, 5, 8 and 9)

University of Minnesota

Henry X. Liu (primary author for Chapters 3 and 4)

Georgia Institute of Technology

Jorge A. Laval (primary author for Chapters 2 and 6)

University of California, Berkeley

Xiao-Yun Lu (primary author for Chapter 7)

Meng Li

Joshua Pilachowski

Wei-Bin Zhang

ABSTRACT

A big segment of the traffic signal control systems in California and United States are closed-loop systems. Because wide-scale deployment of advanced adaptive control systems may be many years away due to the associated high costs, there is a significant need to improve the effectiveness of the state-of-the-practice closed-loop systems. To address the need, this project focuses on: 1) developing an integrated micro-simulation/signal optimization tool to enhance the capability of generating efficient signal timing plans, and 2) developing a systematic approach to make closed-loop systems be more robust and traffic responsive.

An integrated simulation/signal optimization tool can generate, evaluate and fine tune signal timing plans in a cohesive manner. This report presents an approach for integrating Paramics with Synchro and TRANSYT-7F. Two sets of Paramics plug-ins are developed to facilitate the two-way data conversion between Paramics and Synchro or TRANSYT-7F. The first set of plug-ins read Paramics network data and traffic volume data and generate Synchro or TRANSYT-7F data files while the second transfer optimized signal timing data back to Paramics. These tools may assist traffic engineers in developing efficient signal timing plans for arterial traffic operations. Step-by-step tutorials are also included in the report to teach how to use the new plug-ins.

The advancement and deployment of telecommunication and ITS technologies make traffic and signal status data more readily available. These high-resolution data provide opportunities to allow closed-loop control systems to operate more adaptively and robustly to the changes in traffic demands and patterns. This report presents a systematic approach to make use of traffic and signal data to further improve the control performance of closed-loop systems. The systematic approach includes three components: timing, monitoring, and fine-tuning. For timing, two innovative models are developed to generate robust optimal signal timings that are less sensitive to fluctuations of traffic flows at the same time minimizing the mean of the delays per vehicle across all possible realizations of uncertain traffic flows. Another procedure is proposed to optimally

determine time-of-day intervals for time-of-day controls based on a large set of archived traffic data. For monitoring, a prototype signal performance monitoring system is developed to report performance measures of signal control operations and help traffic operation staffs make the decision whether a retiming or fine-tuning effort is needed or not. Finally, for fine-tuning, an offset refiner is introduced to fine tune signal offsets to provide smoother progression in either one-way or two-way coordination. The offset refiner is easy to implement, and could be run periodically or together with the performance monitoring system. If the signal performance degrades, the refiner can be called to fine-tune the offsets for better progression.

Keywords: signal optimization, simulation, archived data, and closed-loop systems

TABLE OF CONTENTS

SECTION	PAGE
ACKNOWLEDGEMENTS	ii
ABSTRACT	iii
1 INTRODUCTION	1
1.1 Motivation	1
1.2 Proposed Research	3
1.3 Research Objective	5
1.4 Reference	6
2 REVIEW AND EVALUATION OF SIGNAL OPTIMIZATION SOFTWARE	9
2.1 Background	9
2.2 Literature Review	10
2.3 Evaluation Study	12
2.4 References	15
3 INTEGRATION OF PARAMICS WITH SYNCHRO AND VERIFICATION	16
3.1 Introduction	16
3.2 Paramics – Synchro/TRANSYT-7F Application Architecture	18
3.3 Implementation of an Integrated Paramics – Synchro Application	21
3.4 Step-By-Step Procedure	23
3.5 Step-By-Step Tutorial	25
3.6 Conclusion and Approach Limitations	33
3.7 References	34

4	INTEGRATION OF PARAMICS WITH TRANSYT-7F AND VERIFICATION	36
4.1	Introduction	36
4.2	Implementation of an Integrated Paramics – TRANSYT-7F Application	39
4.3	Step-By-Step Procedure	43
4.4	Step-By-Step Tutorial	45
4.5	Conclusion and Approach Limitations	53
4.6	References	55
5	ROBUST TIMING PLANS FOR ISOLATED SIGNALIZED INTERSECTIONS	57
5.1	Introduction	57
5.2	Scenario-Based Optimization	59
5.3	Min-Max Optimization	62
5.4	Numerical Examples	67
5.5	Conclusions	73
5.6	References	74
6	SIGNAL TIMING STRATEGIES FOR TIME-OF-DAY CONTROL	77
6.1	Motivation	77
6.2	Background	78
6.3	Proposed Method	83
6.4	Conclusions	85
6.5	References	85
7	DEVELOPMENT OF A PROTOTYPE SIGNAL OPERATIONS MONITORING SYSTEM	87
7.1	Introduction	87
7.2	Document Review	88
7.3	List of Parameters for Performance Measurement	92

7.4	Performance Parameter Calculation	93
7.5	Software Structure	97
7.6	References	99
8	AN OFFSET REFINER FOR COORDINATED ACTUATED SIGNAL CONTROL SYSTEMS	100
8.1	Introduction	100
8.2	Premise of the Offset Refiner	102
8.3	Maximization of Expected bandwidth	107
8.4	Minimization of Red-Meeting Probability	111
8.5	Concluding Remarks	117
8.6	References	118
9	CONCLUSIONS AND NEXT STEPS	120
9.1	Conclusions	120
9.2	Next Steps	121

1 INTRODUCTION

This report constitutes the final deliverable for California PATH Task Order 5325 - “Development of An Integrated Microscopic Traffic Simulation and Signal Timing Optimization Tool”. The project has investigated the following:

- Integration of Paramics with Synchro and TRANSYT-7F to generate, evaluate and fine-tune signal timing plans in a cohesive manner;
- Development of a systematic approach to improve the efficiency of the state-of-the-practice closed-loop control systems, making use of archived traffic and signal data.

1.1 Motivation

Many of the traffic signal control systems in California and United States are closed-loop systems with control logic distributed among three levels: the local controller, the on-street master, and the office computer. The typical closed-loop control system consists of 8-10 local controllers connected to a field master controller. The controller deployed in the field typically operates a time-of-day (TOD) schedule with 3-5 plans. Some agencies run as many as 7 plans in a given day. A small percentage of systems operate in traffic responsive plan selection mode [1-1].

Common practice for signal timing involves data collection during the peak hours of a “typical” day of the week and a cursory examination of traffic volumes during off-peak periods and on weekends. The information collected is normally analyzed by using an off-line optimization program, such as Synchro, TRANSYT-7F and PASSER II/III and the results are then translated into signal settings, and the changes at the signal controller are made accordingly.

The off-line signal optimization programs that have been used to help optimize traffic signal timing are usually macroscopic, representing traffic in terms of aggregate measures of vehicle movements at intersections and applying analytic formulae to determine

measures of effectiveness such as delay and queue length. However, some of these formulae are known to be unstable or inaccurate, particularly under congested conditions. Moreover, these programs are mostly deterministic and thus fail to capture the stochastic and varying nature of traffic. As a consequence, the resulting signal timing plans may not be efficient. It is desirable that the plans can be evaluated by a tool before being deployed in the field. Microscopic simulation technology provides the ability to simulate the detailed movements of individual vehicles in a traffic network according to behavioral models that attempt to mimic actual driver decisions and actions. A properly calibrated and validated simulation program can model a traffic network accurately. Although most of current microscopic traffic simulation software, such as Paramics, VISSIM and CORSIM have limited pre-programmed controller functionality, studies have been conducted to expand the functionality through development of a set of plug-in modules (e.g., [1-2]) or a hardware-in-the-loop setup (e.g., [1-3] and references therein). With incorporating field traffic controller functionality, micro-simulation can be used to evaluate signal timing plans, signal optimization programs and even signal control systems. The validity of use of micro-simulation for this purpose has been demonstrated by dozens of prior studies (e.g., [1-4], [1-5] and [1-6]). In fact, some of signal optimization programs have started providing some functionality to facilitate the use of micro-simulation. For example, Synchro can build input files for CORSIM such that the timing plan can be simulated with CORSIM for a more detailed analysis [1-7]. On the other hand, some of microscopic traffic simulators have also provided interfacing tools to work with signal optimization programs. For example, AIMSUN has available an interface with TRANSYT to allow users to convert AIMSUN network into TRANSYT one, and use a TRANSYT control plan into AIMSUN simulator [1-8]. Similarly, VISSIM also can interface with Synchro and TEAPAC [1-9].

While recent research has focused primarily on developing real-time adaptive signal control algorithms (e.g. [1-10], [1-11] and [1-12]), practicing traffic engineers are quick to note that the wide scale implementation of fully adaptive systems is many years away, due to the high costs associated with high-end controllers, increased system-wide detection and wide-bandwidth communications [1-13]. Therefore, there is a significant

need to improve the effectiveness of current closed-loop systems. Efforts have been made to address the need. For example, in 2001, FHWA initiated a program to assess, and then pursue a cost-effective solution for applying adaptive control systems (ACS) technology to current, state-of-the-practice closed-loop traffic signal control systems. The resulting ACS-Lite system is being tested by FHWA [1-1]. Over the past several years, traffic signal vendors have implemented many traffic responsive features in their closed-loop systems. However, none of the vendor developed closed-loop signal systems have undergone rigorous evaluation [1-3].

1.2 Proposed Research

In order to improve the efficiency of currently-deployed closed-loop systems, two research directions can be adopted among others. One is to enhance the capability of developing efficient signal timing plans and the other is to enable the state-of-the-practice closed-loop systems to operate in a more traffic-responsive manner.

An integrated micro-simulation and signal timing optimization tool that can efficiently predict traffic condition and optimize signal timing strategies in a cohesive manner will help traffic engineers develop efficient signal timing plans.

Paramics is a scalable high-performance microscopic traffic simulation package developed in Scotland, and has been widely used in Caltrans for years. A properly calibrated and validated Paramics simulation can model a traffic network accurately. This allows traffic engineers to study the traffic behavior and test control strategies before implementation. However, Paramics, like most simulation tools, is not designed for traffic operation and thus lacks signal timing optimization capability. If Paramics can interact with signal optimization programs, it will provide a powerful tool for traffic signal operation, because predicated traffic demand from Paramics can be used as input to the signal optimization program to obtain timing plans, and the timing plans can then be evaluated by Paramics to provide guidance for fine-tuning before they are deployed in

the field. With this tool, further analyses can be performed to study how to develop efficient timing plans, for example, by appropriately determining TOD intervals.

Actuated traffic controllers receive all of their information regarding the current state of the traffic system in the form of detector calls, typically from inductive loop detectors cut into the pavement surface. These calls, or actuations, simply indicate something is in a specific location, demanding service for a particular movement. The controller can not determine if that call is due to a single vehicle or a large platoon of vehicles. However, in some configurations where advance loops and/or four 6'×6' presence loops are placed, appropriate adjustments can be made available arrival and/or departure traffic counts and occupancies. Although Paramics still lacks the functionality to receive and utilize real-time loop data, considerable efforts can be made to facilitate the use of real-time traffic data in the simulation, which will further improve the capability of Paramics to replicate real-world traffic conditions. Moreover, the integrated micro-simulation and signal optimization tool can then optimize signal timing strategies in a near real-time (multi-cycle) manner.

The traditional signal timing process is quite time-consuming. It is rarely repeated unless changes in traffic conditions are so significant that the system begins performing poorly. Therefore, the process is not able to produce a timely solution. Though actuated signals can respond to traffic fluctuations, they cannot automatically respond to changes in traffic pattern or overall increase in traffic volume. It has been estimated that traffic experiences an additional 3%-5% delay per year as a consequence of not retiming signals as the conditions evolve over time [1-1]

The advancement and deployment of telecommunication and ITS technologies make real-time (true) traffic data more readily available. In addition to the loop data, second-by-second returns of signal status are available for all phases in California. These real-time data provide opportunities to allow closed-loop control systems to be more adaptive to traffic changes in both strategic and tactic levels. More specifically, through monitoring the signal control operations, it will be easier to find out the problem, if any, and produce

a timely and efficient signal timing solution based on a large set of archived traffic data. In this sense, traffic controls will become more adaptive in a strategic level. By using real-time traffic and signal status data, a plan refiner can be developed to adjust to some extent the cycle, splits and offsets of the active timing plan to allow traffic controllers to respond to changes in traffic demands and patterns, in other words, make traffic controls to be more adaptive in a tactic level.

1.3 Research Objectives

As the first attempt along the research directions sketched above, this project primarily focuses on the following two objectives:

- Objective 1: Integrating Paramics with two widely-used signal timing programs: Synchro and TRANSYT-7F.
- Objective 2: Developing a systematic approach to enable the state-of-the-practice closed-loop control systems to be more adaptive to the changes in traffic demands and patterns.

To fulfill the first objective of the project, Chapter 2 briefly reviews and evaluates two signal optimization software packages, TRANSYT-7F and Synchro. Chapters 3 and 4 introduce an approach for integrating Paramics with Synchro and TRANSYT-7F, which is geared towards signal timing optimization and performing a two-way conversion between Paramics and the signal optimization programs. Two sets of converters are developed as Paramics plug-ins, one for Synchro and the other for TRANSYT-7F. After converting the Paramics network into the signal optimization program's format, the user can use the powerful features of the program to design signal timing plans, which can then be transferred back to the Paramics model for evaluation and fine-tuning.

Chapters 5, 6, 7 and 8 compose a systematic approach for improving the efficiency of the state-of-the-art closed-loop signal control systems (Objective 2). The systemic approach

includes three components: timing, monitoring, and plan adjustment. The key of the approach is to make the best use of archived traffic and signal status data. Chapter 5 presents two approaches, scenario-based and min-max, to determine robust optimal signal timings for isolated signalized intersections. The robust plans would be less sensitive to fluctuations of traffic flows at the same time minimizing the mean of the delays per vehicle across all realizations of uncertain traffic flows. Chapter 6 makes use of a large set of archived traffic data to optimally determine TOD intervals for TOD controls. Chapter 7 develops a prototype signal performance monitoring system that enables traffic operation staffs to understand the historical and current performances of signal control operations and help them make the decision whether a retiming or fine-tuning effort is needed or not. Finally, Chapter 8 provides a proof of concept of an offset refiner, which fine tunes the signal offsets to provide smoother progression in either one-way or two-way coordination. The proposed offset refiner is easy to implement and can work readily with current closed-loop signal control systems to improve their system performance.

This is the first of nine chapters of the final report, and conclusions and a brief description of future work are provided in Chapter 9.

1.4 Reference

- 1-1 Luyanda, F. et al. ACS-Lite Algorithmic Architecture. Paper presented at the 82nd Transportation Research Board Annual Meeting, Washington, D.C., 2003.
Available at: http://www.nawgits.com/icdn/acslite_fldtest.html
- 1-2 Liu, H., Chu, L., and Recker, W. Paramics API Development Document for Actuated Signal, Signal Coordination and Ramp Control, California PATH Program Working Paper, UCB-ITS-PWP-2001-11, 2001.
- 1-3 Bullock, D. et al. Hardware-in-the-Loop Simulation. Transportation Research, Vol. 12C, 2004, pp.73-89.

- 1-4 Stallard C. and Owen, L. E. Evaluating Adaptive Signal Control Using CORSIM. Proceedings of the 1998 Winter Simulation Conference, pp. 1147-1153, 1998.
- 1-5 Park B. et al. Evaluating Reliability of TRANSYT-7F Optimization Schemes. Journal of Transportation Engineering, Vol. 127, No.4, pp.319-326, 2001.
- 1-6 Saiyed S. and Stewart J. An Assessment of Pre-timed, Actuated and Adaptive Signal Control Strategies for Unsaturated and Saturated Arterial Network. Paper presented at the 83rd Transportation Research Board Annual Meeting, Washington, D.C., 2004.
- 1-7 Trafficware. Synchro 6 User Guide. 2003.
- 1-8 TSS – Transportation Simulation Systems. AIMSUN Microscopic Traffic Simulator: A Tool for the Analysis and Assessment of ITS Systems. <http://www.tss-bcn.com>, 2002.
- 1-9 PTV America, Inc. VISSIM: Traffic & Transit Simulation. Available at <http://www.itc-world.com/vissim.html>.
- 1-10 Farradyne Systems, Inc. RT-TRACS Prototype Control Strategies (Task D Working Paper). FHWA Research Report, Contract No. DTFH61-C-00001, 1995.
- 1-11 Mirchandani P. and Head L. A Real-Time Traffic Signal Control System: Architecture, Algorithms and Analysis. Transportation Research, Vol.9C, 2001, pp. 415-432.
- 1-12 Haghani A. Development of Additional Prototype Strategies for Inclusion in RT-TRACS. University of Maryland, College Park, Maryland, FHWA Research Report, Contract No. DTFH61-94-C-00205, 1995.

- 1-13 Smith, B. et al. Data-Driven Methodology for Signal Timing Plan Development: A Computational Approach. *Computer-Aided Civil and Infrastructure Engineering*, 17, pp.387-395, 2002.
- 1-14 Park B. et al. Optimization of Time-of-Day Breakpoints for Better Traffic Signal Control. Paper presented at the 83rd Transportation Research Board Annual Meeting, Washington, D.C., 2004.

2 REVIEW AND EVALUATION OF SIGNAL OPTIMIZATION SOFTWARE

The objective of this chapter is to briefly review and evaluate two signal optimization software packages, TRANSYT-7F and SYNCHRO 6, which are integrated with PARAMICS, as documented in Chapters 3 and 4.

2.1 Background

2.1.1 TRANSYT-7F

TRANSYT-7F was originally developed in the United Kingdom by the Transport and Road Research Laboratory (TRRL); it is currently developed by McTrans. The main features of TRANSYT-7F are summarized below:

- mesoscopic-deterministic optimization model for arterials and networks
- traffic model includes platoon dispersion
- objective functions: minimize delays and/or stops; progression maximization.
- optimization process includes genetic algorithm, hill-climb, and multi-period optimization

Notice that the most recent release (version 10.2) introduces “direct CORSIM optimization”, where TRANSYT-7F applies supply timing plan candidates, and CORSIM evaluates them.

2.1.2 SYNCHRO

SYNCHRO is developed by Trafficware Inc., a company based in Berkeley, Calif. It has the best user interface of all signal-timing tools currently available. The key features of the program include:

- mesoscopic-deterministic optimization model for arterials and networks
- objective functions: a composite of delay, number of stops and the number of vehicles affected by the queue,

- optimization process: this process requires constant human intervention and consist of the following five steps:
 1. Set up initial timing plans for each intersection individually
 2. Partition into network into subsystems (optional)
 3. Optimize network cycle length
 4. Optimize offsets and phasing

2.1.3 The Percentile Delay Method

SYNCHRO uses the percentile delay method in order to account for the variability of traffic flow. It assumes that arrivals follow a Poisson distribution and uses the 10th, 30th, 50th, 70th and 90th percentile scenarios for the delay calculations. The delay output by the model is the average of these five scenarios weighted by the percentile flow rates. According to the manual, in most cases the percentile delay method would give similar results than Webster's formula. However, it is indicated that the method is more accurate on *actuated signals* in the presence of pedestrian phases or skipped phases.

2.2 Literature Review

The following publications are concerned in comparing several signal timing optimization software. The comparison identified that SYNCHRO consistently produced lower MOEs relative to TRANSYT.

2.2.1 Chaudhary et al, 2002

This publication [3-1] corresponds to a research report by the Texas Transportation Institute for the Texas Department of Transportation. It summarizes a 2-year project where they compared the software packages TRANSYT-7F, SYNCHRO 5 and PASSER II. The comparison was performed using actual data from several arterials located in Texas, while the MOEs were computed using the microscopic traffic simulator CORSIM. The main conclusions of this study are:

- SYNCHRO and PASSER outperform TRANSYT in all aspects.
- In particular, TRANSYT was found to provide poor progression band with, inconsistent cycle length selection and the greater delays.
- bandwidths produced by SYNCHRO and PASSER are similar for short arterials but SYNCHRO bandwidths deteriorate for large arterials.
- SYNCHRO produced lowest delays

2.2.2 Yang, 2001

This publication [3-2] compares the software packages TRANSYT-7F, SYNCHRO 3 and PASSER II-90. It uses real data from a single arterial with nine signalized intersections (Iowa Street located in Lawrence, Kan.) while the MOEs were obtained using CORSIM. From this paper it can be concluded that:

- PASSER provides the lowest delays and stops, followed by TRANSYT-7F and SYNCHRO 3
- PASSER provides the best timing plan for this particular arterials
- SYNCHRO is the only package able to coordinate a network of streets
- TRANSYT generates longer cycle length, which induces larger delays

2.2.3 Washburn and Larson, 2002

This paper [3-3] compares TRANSYT-7F, SYNCHRO 3 and the highway capacity manual software HCS for a single coordinated arterial in the University district in Seattle, Washington. Unlike the previous papers analyzed in this report the arterial is operated using actuated control.

The delay was the only MOE considered in this study, and it was obtained as reported by each software package.

The following conclusions can be drawn:

- HCS reported the lowest delays

- TRANSYT yield lower delays than SYNCHRO

Notice that these results are based on each program's estimation, and thus may not be comparable.

2.3 Evaluation Study

Paramics is one of the widely used microscopic traffic simulation models. One important feature of Paramics is that Paramics allows the user to customize many features of the underlying simulation model through a Functional Interface or Application Programming Interface (API). The proposed research will integrate Paramics with a signal optimization program. The candidates include TRANSYT-7F, SYNCHRO, and possibly others to be identified after consultation with the Caltrans traffic operation staffs. In any event, the candidates should be well accepted and widely used by practicing traffic engineers. A thorough evaluation study will be conducted to assess the timing performance and reliability of these candidates.

To facilitate the evaluation work, a testing environment with a preliminary level of integration through "middleware" was created. The "middleware" was developed to automate the process of exchanging data between Paramics and signal optimization programs. More specifically, the middleware takes the simulated traffic volumes from Paramics and generate input files for the signal optimization program. Meanwhile, it also feeds signal timing plans into Paramics. For each of the candidates, the middleware was coded respectively. The testing environment is illustrated by Figure 2.1.

The candidates ran with several sets of traffic and roadway conditions commonly encountered in street networks including both under-saturated and saturated flows. MOEs, such as delay, throughput and the number of stops, from all runs of simulations were statistically compared to assess the performance of each signal optimization program for different sets of conditions. The network used is a series of twelve intersections between 2nd Avenue and 28th Avenue on El Camino Real.

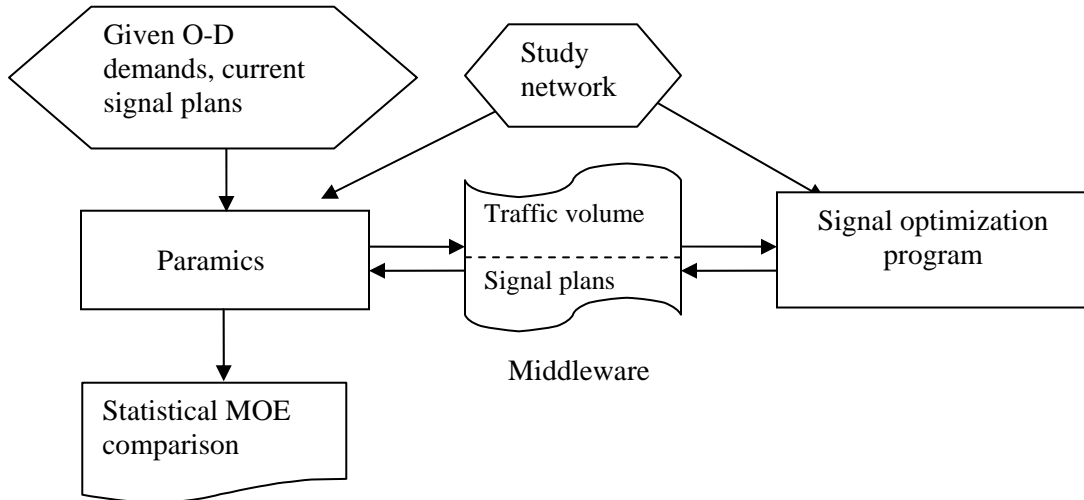


Figure 2.1 Evaluation Environment

2.3.1 TRANSYT-7F

The series of intersections were inputted into TRANSYT-7F. The optimization was then run in order to specify the best cycle length for the system as well as the individual offset for each intersection. The values obtained from this optimization are shown in Table 2.1.

Table 2.1 TRANSYT-7F Optimization Results

Cross Street	Cycle Length (sec)	Offset (sec)
2nd Avenue	115	50
3rd Avenue	115	50
4th Avenue	115	46
5th Avenue	115	52
9th Avenue	115	62
12th Avenue	115	115
Barneson Avenue	115	62
17th Avenue	115	85
20th Avenue	115	41
25th Avenue	115	112
27th Avenue	115	101
28th Avenue	115	91

These values were then input into a Paramics simulation of the series of intersections. The simulation was then run for an in-simulation time of 2 hours. The resulting data shows an average speed of 16.4 kph and an average travel time of 367.0 seconds.

2.3.2 SYNCHRO

The same network was created in a SYNCHRO model. The optimization found the optimal cycle length for the system as well as the optimal offsets for each intersection. The values obtained from this optimization are shown in Table 2.2.

Table 2.2 SYNCHRO optimization results.

Cross Street	Cycle Length (sec)	Offset (sec)
2nd Avenue	106	9
3rd Avenue	106	97
4th Avenue	106	87
5th Avenue	106	77
9th Avenue	106	70
12th Avenue	106	49
Barneson Avenue	106	35
17th Avenue	106	62
20th Avenue	106	44
25th Avenue	106	93
27th Avenue	106	3
28th Avenue	106	3

These values were then input into a Paramics simulation of the series of intersections. The simulation was then run for an in-simulation time of 2 hours. The resulting data showed an average speed of 21.1 kph and an average travel time of 291.4 seconds.

2.3.3 Results

Based on the above results, it is clear that SYNCHRO outperforms TRANSYT by as much as 26%. Reassuringly, this result coincides with the literature review, which identified that SYNCHRO consistently produced lower MOEs relative to TRANSYT. Additionally, we performed the same experiment described in the previous section but with a 20% demand increase. The results are summarized in Table 2.3, which shows the average vehicle delay for both demand loads.

Table 2.3 Average Vehicle Delay

Demand	SYNCHRO	TRANSYT	SYNCHRO improvement
100%	291 sec	367 sec	26 %
120%	375 sec	406 sec	8 %

As expected, in a more congested scenario the impact of a better timing diminishes, but is still significant.

2.4 References

- 2-1. N.A. Chaudhary, V.G. Kovvali, C. Chu, S.M. Alam. Software for Timing Signalized Arterials, Research Report 0-4020-1. Texas Transportation Institute, College Station, TX. August 2002.
- 2-2. X.K. Yang. Comparison among Computer Packages in Providing Timing Plans for Iowa Arterial in Lawrence, Kansas. Journal of Transportation Engineering, Vol. 127, No. 4, July/August 2001, pp. 311-318
- 2-3. Washburn, Scott S. and Larson, Nate. Signalized Intersection Delay Estimation: Case Study Comparisons of Transyt-7F, SYNCHRO, and HCS. Institute of Transportation Engineers Journal, March 2002.

3 INTEGRATION OF PARAMICS WITH SYNCHRO AND VERIFICATION

3.1 Introduction

Microscopic traffic simulation provides the highest level of detailed traffic dynamics representation when compared to other traffic analysis tools. Unlike other tools, individual vehicles are the basic modeling unit instead of entire traffic streams. However, most simulation tools lack signal timing optimization features. Interfacing microsimulation with signal timing optimization software will help integrate the strengths of microsimulation into the signal timing design processes.

Common practice for signal timing involves data collection during the peak hours of a “typical” day of the week and a cursory examination of traffic volumes during off-peak periods and on weekends. The information collected is normally analyzed by using an off-line optimization program such as Synchro and TRANSYT-7F, and the results are then translated into signal settings, and the changes at the signal controller are made accordingly.

Synchro and TRANSYT-7F are both off-line signal optimization programs that are used to help optimize traffic signal timing. They present traffic at a macroscopic level, use aggregate measures of vehicle movements at intersections and apply analytic formulae to determine measures of effectiveness (MOEs) such as delay and queue length. However, some of these formulae are known to be unstable or inaccurate, particularly under congested conditions. Moreover, these programs are mostly deterministic and thus fail to capture the stochastic and varying nature of traffic. As a consequence, the resulting signal timing plans may not be efficient. It is desirable that the plans can be evaluated by a tool before being deployed in the field.

Microsimulation can simulate the detailed movements of individual vehicles in a traffic network, attempting to mimic actual driver decisions and actions. Although most microsimulation software have limited pre-programmed controller functionality, studies

have been conducted to expand their functionality through the development of plug-in modules [3-1] or hardware-in-the-loop setups [3-2]. By incorporating field traffic controller functionality, microsimulation can be used to evaluate signal timing plans, signal optimization programs and even signal control systems. The validity of use of microsimulation for this purpose has been demonstrated by dozens of prior studies [3-3, 4, and 5]. In fact, some signal optimization programs have started providing some functionality to facilitate the use of microsimulation. For example, both Synchro and TRANSYT-7F can build input files for CORSIM such that timing plans can be simulated with CORSIM for a more detailed analysis [3-6 and 7]. On the other hand, some microscopic traffic simulators provide interfacing tools to work with signal optimization software. For example, AIMSUN can interface with TRANSYT-7F to allow users to convert AIMSUN networks into TRANSYT-7F networks, and use a TRANSYT-7F control plan in AIMSUN [3-8]. Similarly, VISSIM can interface with Synchro and TEAPAC [3-9].

Paramics is a scalable high-performance microscopic traffic simulation package developed by Quadstone in Scotland, is one of the leading microscopic traffic simulators worldwide. A properly calibrated and validated Paramics simulation can model a traffic network accurately. This allows traffic engineers to study the traffic behavior and test control strategies before implementation. However, Paramics, like most simulation tools, is not designed for signal timing optimization. If Paramics can interface with a signal timing optimization program, it will provide a powerful tool for traffic signal operations analysis. Simulated traffic demands from Paramics can be used as input to the signal optimization program to obtain timing plans, and the timing plans can then be evaluated by Paramics to provide guidance for fine-tuning before they are deployed in the field.

Previous efforts that aim at converting existing network data from a range of sources into Paramics exist. Quadstone, for example, developed the Paramics Converter [3-10], which can read network data from various sources including emme/2, Mapinfo, ESRI ArcGIS, Synchro, CORSIM, Cube/TP+/Viper, flat ASCII files, and CSV files. Researchers at the University of California - Santa Barbara developed a software tool,

nicknamed S2P (Shapefile to Paramics), to translate files from ESRI ArcGIS format to Paramics [3-11]. The S2P converter detailed geometric information from GIS shape files and builds a set of files for Paramics. However, both Paramics Converter and S2P can only do a one-way conversion, and cannot convert Paramics networks into other formats. While both Paramics Converter and S2P can be valuable tools for building Paramics networks, they are not geared towards transferring signal timing data; converted networks only contain basic skeleton information and geometric data.

This chapter will introduce an approach for integrating Paramics with two prominent signal timing optimization tools, Synchro and TRANSYT-7F. Two sets of converters were developed as Paramics plug-ins, one for Synchro and the other for TRANSYT-7F. The approach is geared towards signal timing optimization and performs a two way conversion between Paramics and the signal optimization tool. It capitalizes on the user's knowledge in both platforms; after converting the Paramics network into the signal optimization tool's format, the user can use the powerful features of the signal timing optimization tool to design signal timing plans. Signal timing plan data can then be transferred back to the Paramics model.

The next section will present the overall architecture of the integrated Paramics - Synchro/TRANSYT-7F application. The remainder of this chapter is dedicated to the Paramics - Synchro integration process; the process is detailed in section 3.3 and followed by a step by step procedure in section 3.4. A step-by-step tutorial is included in section 3.5. Finally, a conclusion and approach limitations are presented in section 3.6. The Paramics - TRANSYT-7F integration process is presented and detailed with a tutorial in chapter 4.

3.2 Paramics – Synchro/TRANSYT-7F Application Architecture

The architecture of the integrated Paramics – Synchro/TRANSYT-7F application requires two additional Paramics plug-ins; one to facilitate actuated signal operation in Paramics, and the other to facilitate the conversion of network elements into

Synchro/TRANSYT-7F input. Although the actuated signal plug-in facilitates actuated signal logic in Paramics, it cannot optimize the signal timing plans for isolated intersections or arterials. The second additional plug-in builds a skeleton network file that provides a simplified version of the Paramics link/node structure.

In general, the integrated Paramics – Synchro/TRANSYT-7F application architecture comprises two major tiers, the microsimulation tier and the signal optimization tier. All of the plug-ins are implemented as in-process dynamic link library (DLL) components that reside in the microsimulation tier as shown below in Figure 3.1.

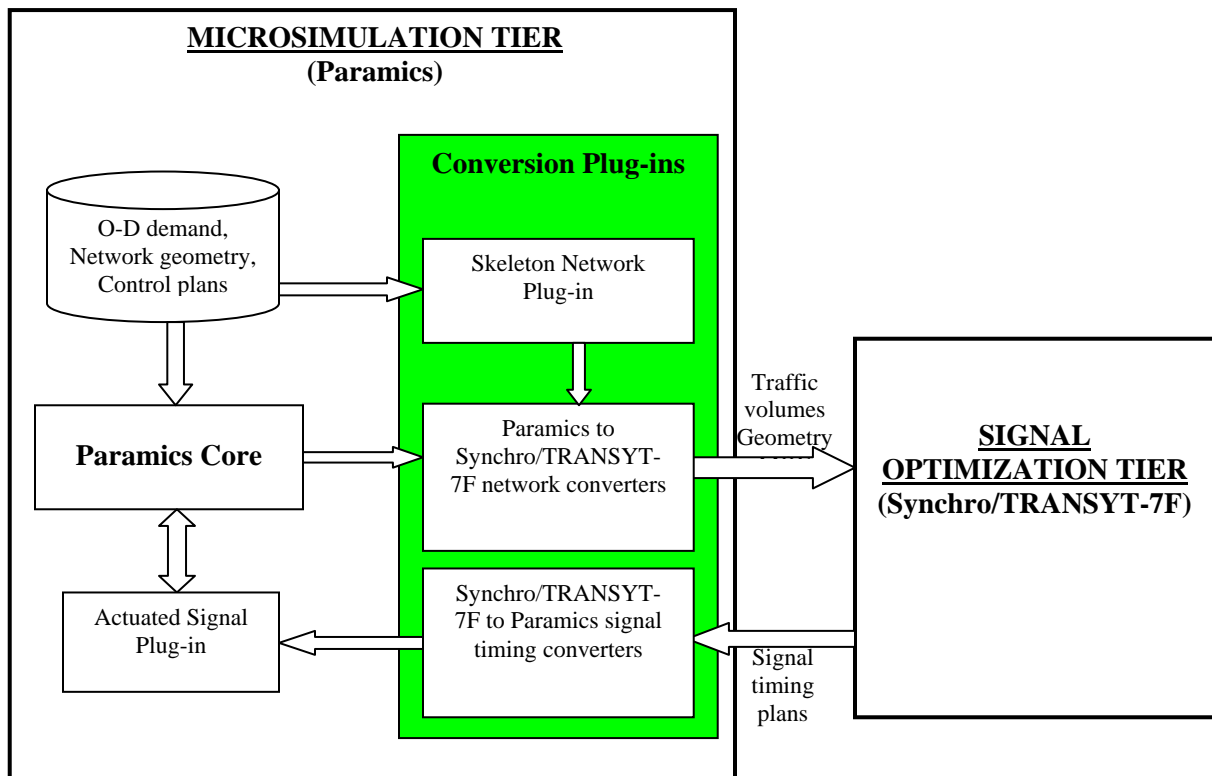


Figure 3.1 Integrated Paramics - Synchro/TRANSYT-7F Application Architecture

The developed Paramics – Synchro/TRANSYT-7F integration tool utilizes a rich Application Programming Interface (API) provided by Paramics, which allows the user to customize many features of underlying simulation model. Two main sets of plug-ins have been developed, the Paramics to Synchro/TRANSYT-7F network converters and the Synchro/TRANSYT-7F to Paramics signal timing plan converters.

The skeleton network plug-in, shown in figure 3.1 above, builds a skeleton file including network skeleton nodes, which represent major junctions in the network, and skeleton links, which define how the skeleton nodes are linked. The reason behind building the skeleton network file is to simplify the complex link/node structure of a Paramics model. Geometric features such as roadway horizontal curvature, widening, and storage lanes may necessitate coding multiple links in Paramics in order to properly replicate the real life network. Networks in Synchro and TRANSYT-7F only require certain geometric features of links at the interface between the link and the intersection node. Figure 3.2 below shows a snapshot of the skeleton file.

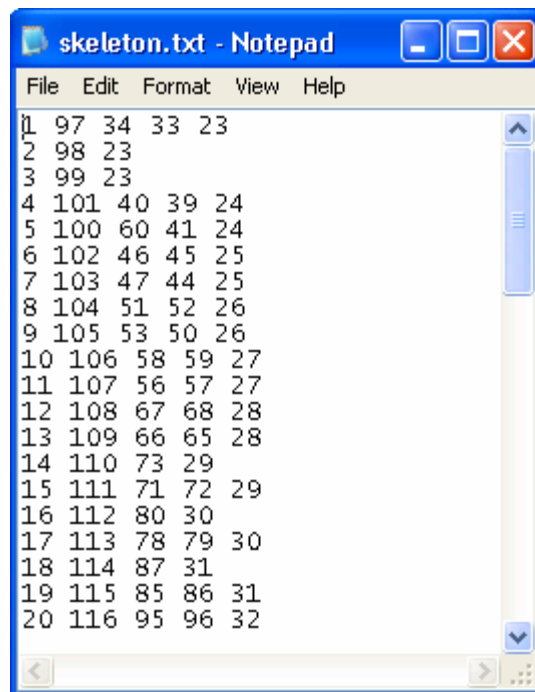


Figure 3.2 Skeleton File Snapshot

Each line in the skeleton file, shown in Figure 3.2 above, represents a skeleton link. For example, the first line in the file is a skeleton link that starts at major junction node 1; links to nodes 97, 34, 33; and ends with major junction node 23. This information is crucial when building the Synchro/TRANSYT-7F input file.

3.3 Implementation of an Integrated Paramics - Synchro Application

The integration of Paramics with Synchro for the purpose of traffic signal optimization requires a two-way conversion process. Two converter applications were developed; the first builds Synchro input files based on the Paramics model and the second transfers optimized signal timing data back to the Paramics model. A flowchart of the process is shown in Figure 3.3, followed by a description of the two conversion procedures.

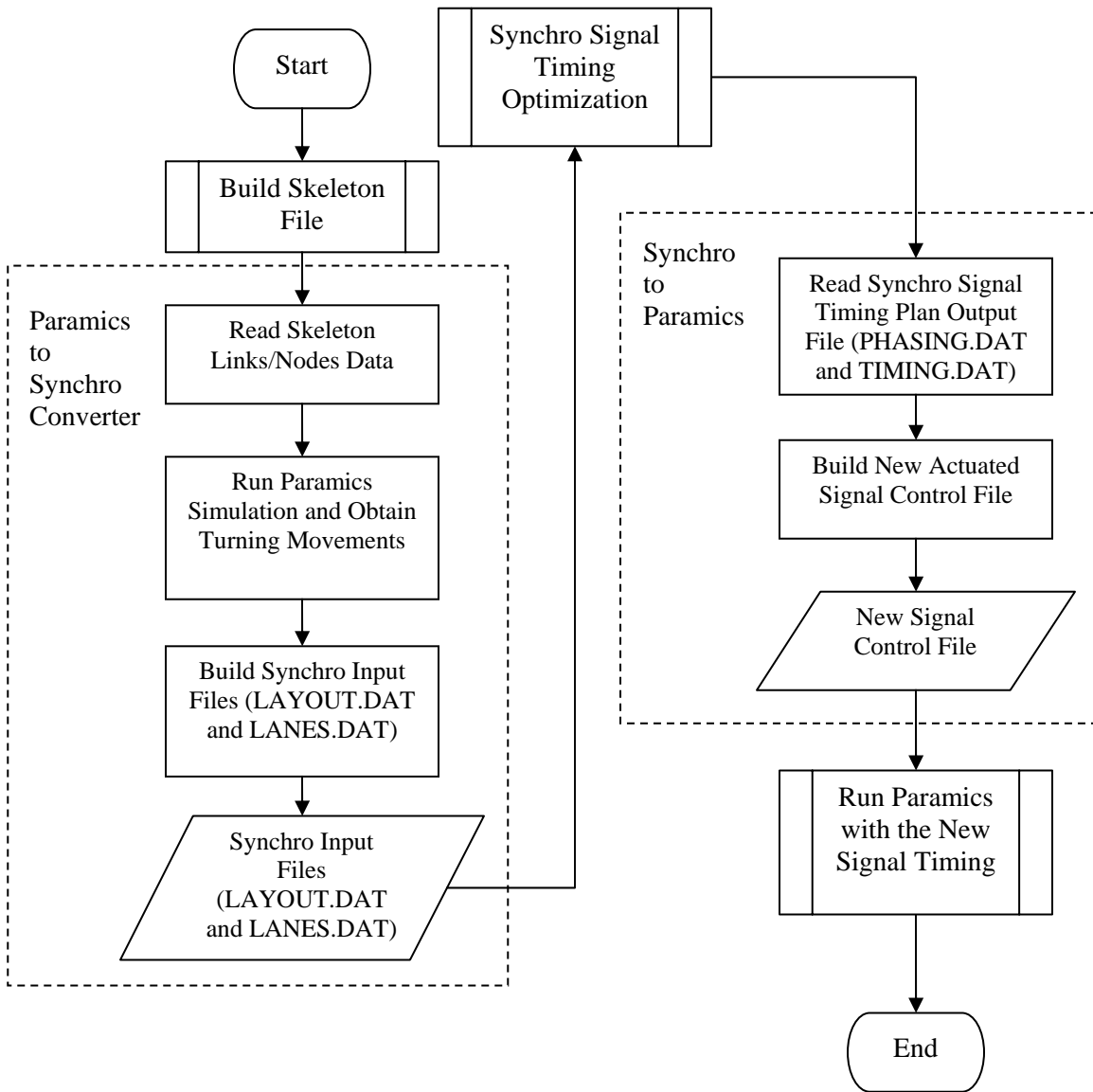


Figure 3.3 The Paramics – Synchro Two-Way Conversion Process Flowchart

3.3.1 The Paramics to Synchro Network Converter

The main goal of this conversion process is to build two Synchro data files, a network layout data file (LAYOUT.DAT), and lanes and volumes data file (LANES.DAT). The first step in the process is to read the skeleton network file to obtain network layout information. For traffic volumes, Synchro requires turning volume data at each signalized intersections. However, only demand data between O-D pairs can be directly read from the Paramics network inputs. It is, therefore, necessary to run the network in Paramics and generate the intersection turning volumes using the Paramics API. Nodes and links data from the existing Paramics network are used to provide network geometry information to Synchro, such as node coordinates and link speed limits. With this information, the Paramics to Synchro converter can generate the two Synchro data files with the necessary information for signal optimization.

3.3.2 The Synchro to Paramics Signal Timing Converter

The Synchro to Paramics signal timing converter was developed to transfer optimized signal timing data from Synchro to Paramics by modifying the Paramics signal control file. This converter is invoked after optimizing the network in Synchro. Optimized signal timing data are contained in two Synchro output files; the first file is the “TIMING.DAT” file, which provides signal splits, cycles and offsets; the second file is the “PHASING.DAT” file, which holds actuated signal data such as minimum green, maximum green and extension.

Actuated signal control logic is maintained by the actuated signal plug-in “actuated_signal.dll”. Signal control data is stored in the “signal_control” file under the Paramics network directory. This is the file that the Synchro to Paramics converter modifies based on the obtained signal control data from Synchro.

3.4 Step-By-Step Procedure

The two-way conversion procedure for Paramics – Synchro integration is divided into the following six steps:

1. Generating skeleton network data from Paramics
2. Generating the Synchro input files
3. Loading the network in Synchro
4. Optimizing signal timings with Synchro
5. Importing signal timing data from Synchro to Paramics
6. Run the network in Paramics with the new signal timing data

The six steps are detailed below.

Step 1: Generating skeleton network data from Paramics

This step is a pre-requisite step to the conversion process. It involves first indicating the location of the “skeleton.dll” plug-in in the “programming.modeller” file for Paramics Modeller simulations. Alternatively, this may be specified in the “programming.processor” file or the “programming.simulator” file if the user prefers to use Paramics Processor or Paramics Processor from the command line, respectively. This is followed by loading the network in Paramics to generate “skeleton.txt” file under the network folder.

Step 2: Generating the Synchro input files

Prior to running the converter, the “skeleton.dll” plug-in should be de-referenced in the Paramics model. The next step is to indicate the locations of the “Paramics_to_Synchro.dll” plug-in and the “actuated_signal.dll” plug-in in the Paramics model, in addition to any other necessary plug-ins for the simulation. The network is to then be loaded and run in Paramics. The “LAYOUT.DAT” and “LANES.DAT” files will be generated under the network folder after the simulation finishes.

Step 3: Loading the network in Synchro

Before loading the data files, a new Synchro project is to be created by selecting “File -> New” in Synchro. The network data can then be loaded by selecting “Transfer -> Data Access...” option; this will open the “UTDF Database Access” window. Under the “Layout” tab the “LAYOUT.DAT” file is to be selected and read by clicking “Select”, browsing to the file location and selecting “Read”. These steps are to also be followed to import the “LANES.DAT” file under “Lane” tab ensuring the “Include Volume related data” option is toggled on.

Step 4: Optimizing signal timings with Synchro

At this point, the network has been converted from Paramics to Synchro. The user can design a signal timing plan using the features provided by Synchro.

Step 5: Importing signal timing data from Synchro to Paramics

After the signal timing plan has been designed by the user in Synchro, the user is to export the Synchro signal timing data by selecting the “Transfer -> Data Access...” option in Synchro; the “UTDF Database Access” window will open. Under the “Phasing” tab, the user is to write a “PHASING.DAT” file to the Paramics network directory. The same steps are followed under the “Timing” tab to export a “TIMING.DAT” file to the existing Paramics network directory.

Before opening Paramics and importing the timing data, the user should ensure that the “signal_control” file exists under existing Paramics the network folder. The “Synchro_to_Paramics.dll” plug-in is to then be referenced in the Paramics model. The network is to then be loaded in Paramics; this will modify the “signal_control” file in accordance with optimized signal timing data generated by Synchro.

Step 6: Run the network in Paramics with the new signal timing data

De-reference the “Synchro_to_Paramics.dll” plug-in in the Paramics model. This step concludes the two-way conversion process and the network is ready to run with new optimized signal timing data.

3.5 Step-By-Step Tutorial

The purpose of this tutorial is to provide the user with a step-by-step example that covers the conversion process from start to end. The example Paramics network used is based on a main street arterial network in Logan, Utah. It consists of one arterial with ten signalized intersections as shown in Figure 3.4 below. The project name is “Tutorial_Synchro” and the project directory for the tutorial is “C:\Tutorial_Synchro”.

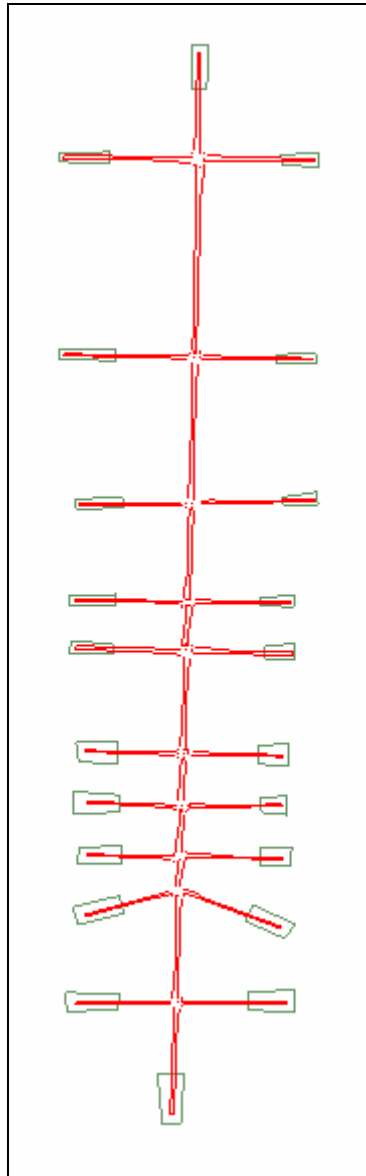


Figure 3.4 Tutorial_Synchro Project Layout

Before using the converters, it is necessary to ensure that the actuated signal control plug-in (actuated_signal.dll) is available and an associated actuated signal control file (signal_control) is also available. The reader is referred to the actuated controller plug-in documentation [12] for further details. It is important to note that initial timing data in the signal control file are not transferred to Synchro by the converter. It is also necessary to have the skeleton network plug-in.

1- Prepare the plug-ins for use in Paramics:

Paramics allows for more than one way to use plug-ins. For Tutorial_Synchro, all plug-in DLL files will be placed in the project directory and referenced in the “programming.modeller” file. The first plug-in that will be used is the skeleton network plug-in. The remaining three plug-ins are de-referenced at this point in the process by simply placing ‘##’ at the beginning of line calling them in the “programming.modeller” file as shown in Figure 3.5.

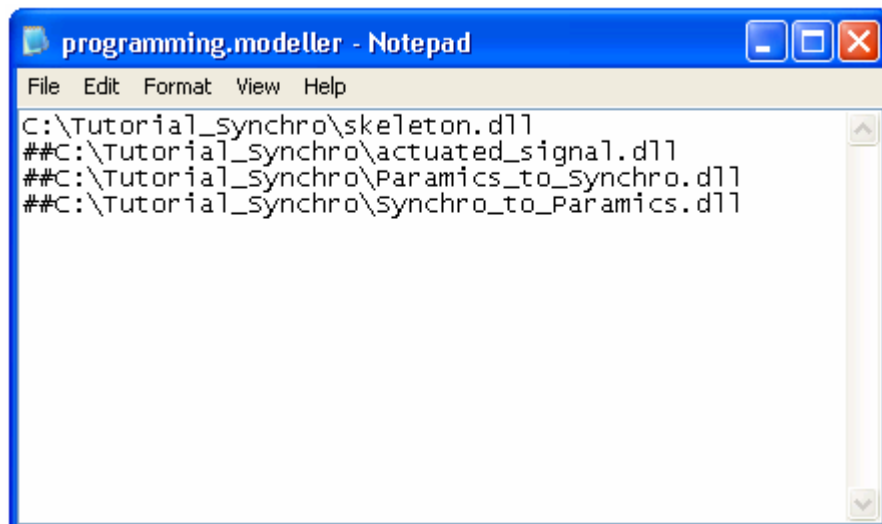


Figure 3.5 Plug-ins Initial Setup

2 - Build the skeleton network file:

After preparing, saving, and closing the “programming.modeller” file, the network is loaded in Paramics Modeller, and then closed without running a simulation. This

step will build the “skeleton.txt” file in the project directory that contains all the skeleton link data necessary for the conversion.

- 3 - Prepare the network for the first conversion process:

In this step, the conversion plug-in will be loaded by opening the “programming.modeller” file and modifying it as shown in Figure 3.6.

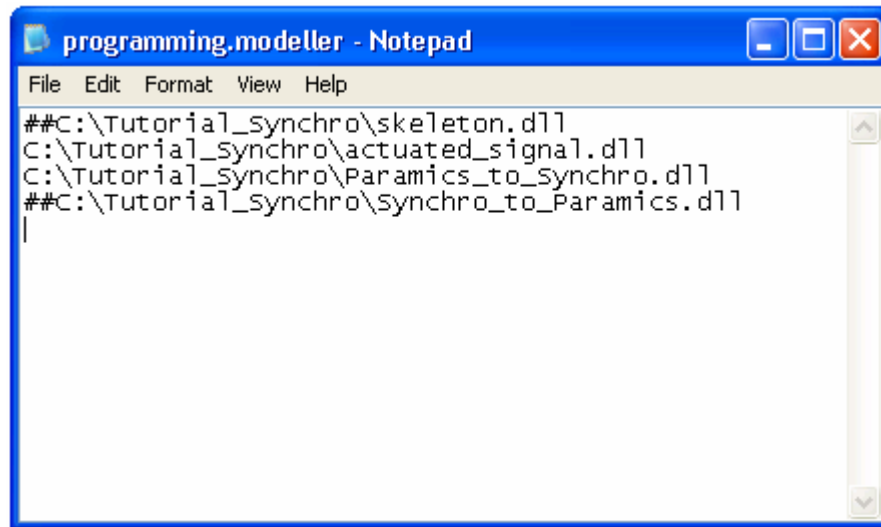


Figure 3.6 Preparing for First Conversion

- 4 - Convert the network:

Re-opened the network in Paramics Modeller and run a simulation. After the simulation is done, the two Synchro network data files (LAYOUT.DAT and LANES.DAT) will be generated is built in the project directory (C:\Tutorial_Synchro). A snapshot of the two files is shown below in Figures 3.7 and 3.8.

INTID	TYPE	X	Y	NID	SID	EID	WID
1	1	-5013	-4514	23			
2	1	-6283	-3033			23	
3	1	-3501	-3025				23
4	1	-6169	-1869			24	
5	1	-3569	-2027				24
6	1	-6139	-1067			25	
7	1	-3554	-1105				25
8	1	-6147	-364			26	
9	1	-3561	-402				26
10	1	-6162	315			27	
11	1	-3554	255				27
12	1	-6283	1691			28	
13	1	-3425	1615				28
14	1	-6290	2326			29	
15	1	-3448	2303				29
16	1	-6237	3604			30	
17	1	-3138	3649				30
18	1	-6442	5584			31	
19	1	-3161	5524				31
20	1	-6442	8207			32	
21	1	-3138	8169				32
22	1	-4658	9606		32		
23	0	-4960	-3016	24	1	3	2
24	0	-4928	-1526	25	23	5	4
25	0	-4911	-1085	26	24	7	6
26	0	-4888	-417	27	25	9	8
27	0	-4865	284	28	26	11	10
28	0	-4826	1646	29	27	13	12

Figure 3.7 LAYOUT.DAT Snapshot

RECORDNAME	INTID	NBL	NBT	NBR	SBL	SBT	SBR	EBL	EBT	EBR	WBL	WBT	WBR
Lanes	23	1	2	1	1	2	1	0	1	0	0	1	0
Shared	23		0			0			3			3	
Speed	23		39			35			29			29	
Volume	23	3	1989	44	13	1702	18	14	56	9	34	68	12
Lanes	24	1	2	1	1	2	1	1	1	0	1	1	1
Shared	24		0			0			2			0	
Speed	24		35			29			29			29	
Volume	24	25	1957	4	3	1752	27	10	18	2	4	49	1
Lanes	25	1	2	1	1	2	1	1	1	1	1	2	0
Shared	25		0			0			0			3	
Speed	25		29			29			29			29	
Volume	25	12	1937	19	9	1720	19	15	173	7	61	279	22
Lanes	26	1	2	1	1	2	1	1	1	1	1	1	1
Shared	26		0			0			0			0	
Speed	26		29			29			29			29	
Volume	26	5	1952	10	5	1746	4	4	30	1	7	47	8
Lanes	27	1	2	1	1	2	1	1	1	1	1	1	1
Shared	27		0			0			0			0	
Speed	27		29			29			29			29	
Volume	27	33	1911	12	10	1750	67	39	114	14	5	132	5
Lanes	28	1	2	1	1	2	1	1	2	1	1	2	1

Figure 3.8 LANES.DAT Snapshot

5- Open the network in Synchro:

Open Synchro and create a new file by selecting the “File -> New” option. Select the “Transfer -> Data Access...” option; this will open the “UTDF Database Access” window. Click on the “Layout” tab and under “Active File” at the top of the form, click “Select”. Browse to the network folder (C:\Tutorial_Synchro) and select the “LAYOUT.DAT” file and click “Read”. The network becomes visible in Synchro as shown in Figure 3.9.

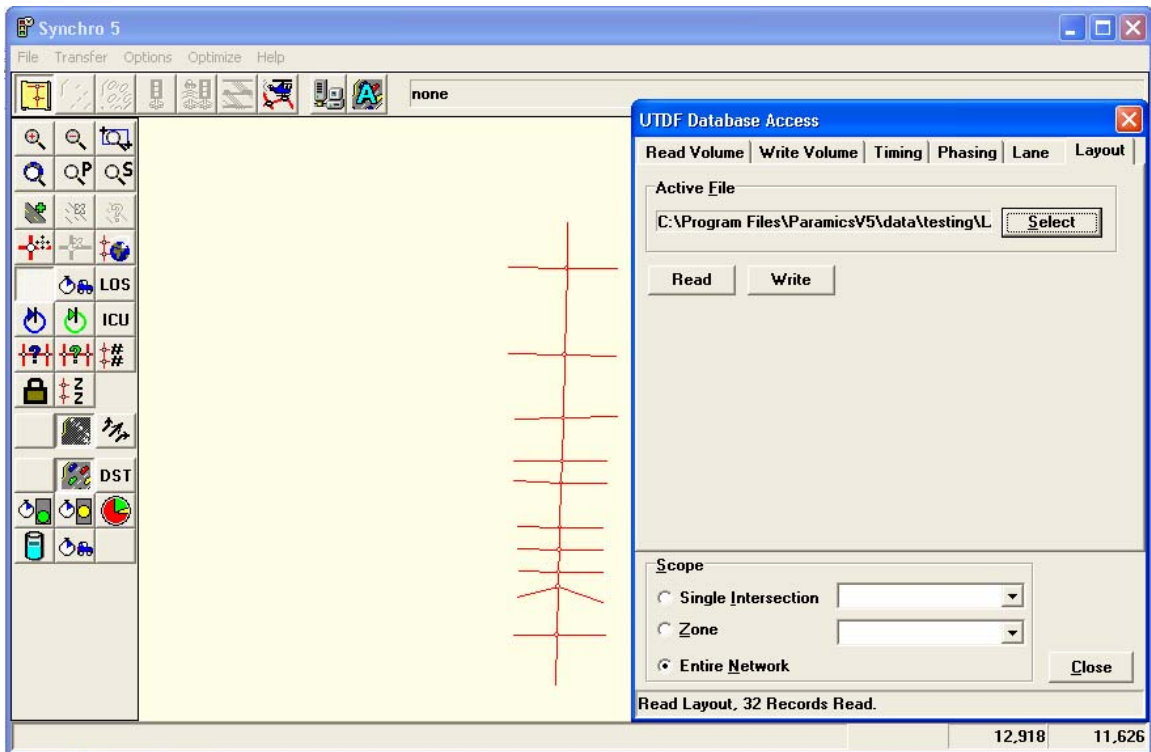


Figure 3.9 Importing the Synchro LAYOUT.DAT File

Next, import the lanes and volumes data by first clicking on the “Lane” tab in the “UTDF Database Access” window. Ensure that the “Include Volume related data” check box is check. Select and read the “LANES.DAT” file located in the project directory. Volume data is now included in the Synchro network as shown in Figure 3.10. Close the “UTDF Database Access” window and save the Synchro project.

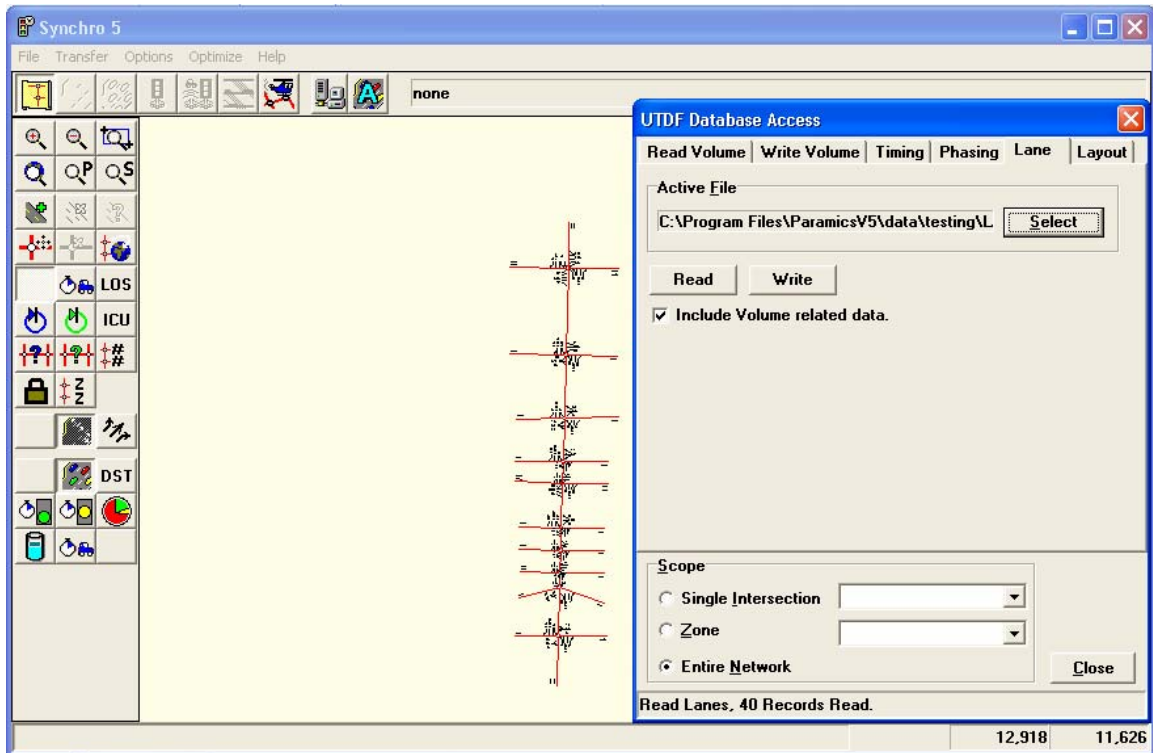


Figure 3.10 Importing the Synchro LANES.DAT File

6 - Optimize the network traffic signals:

Select the “Optimize -> Partition Network” option. Accept the default “One System (0)” partitioning strategy in the Partition Network window and click OK. When the network partitioning is complete, select the Optimize -> Network Cycle Lengths...” option. If asked to do so save changes. Accept the defaults in the “Optimize Cycle Lengths” window and click “Automatic”. When cycle lengths optimization is complete, save the project and select the “Optimize Network Offsets” option. Accepts the defaults, click OK, and after the network offsets optimization is complete save the network.

7 - Export the Synchro signal timing file:

Select the “Transfer -> Data Access...” option; this will open the “UTDF Database Access” window. Click on the “Phasing” tab and under “Active File” at the top of the form, click “Select”. Browse to the network folder (C:\Tutorial_Synchro), save the “PHASING.DAT” file and click “Write” as shown in Figure 3.11.

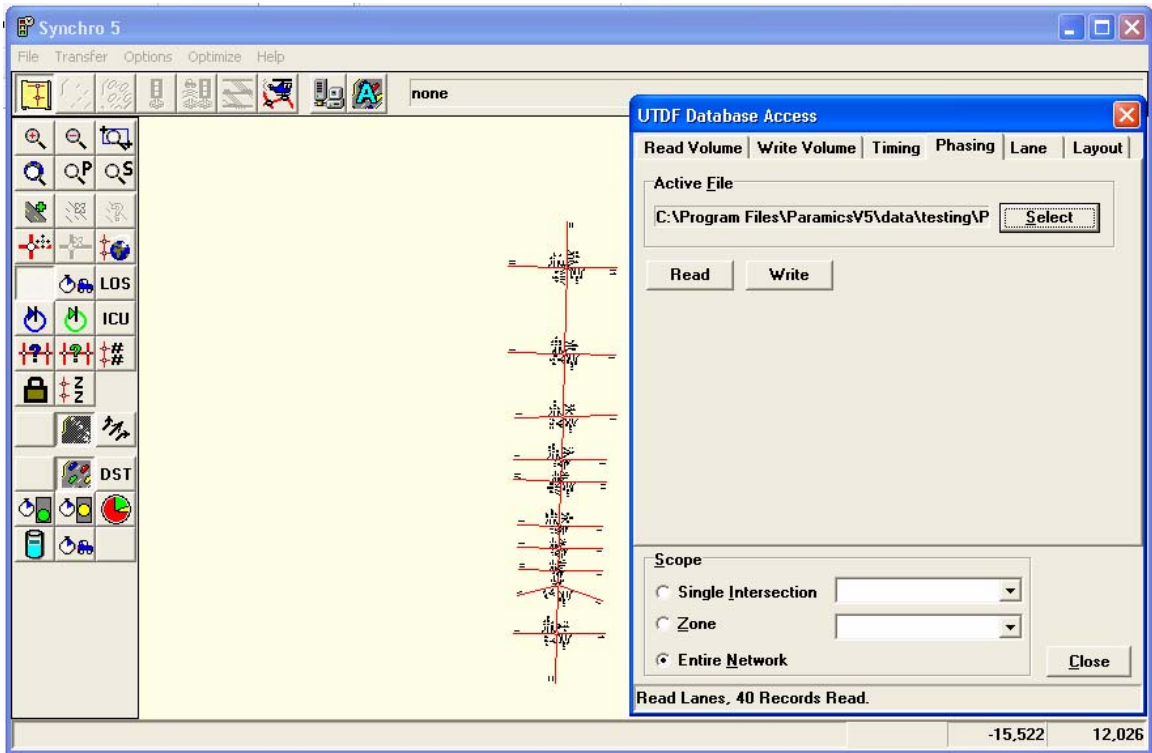


Figure 3.11 Exporting PHASING.DAT File from Synchro

Similarly, export the “TIMING.DAT” file under the “Timing” tab as shown in Figure 3.12 below.

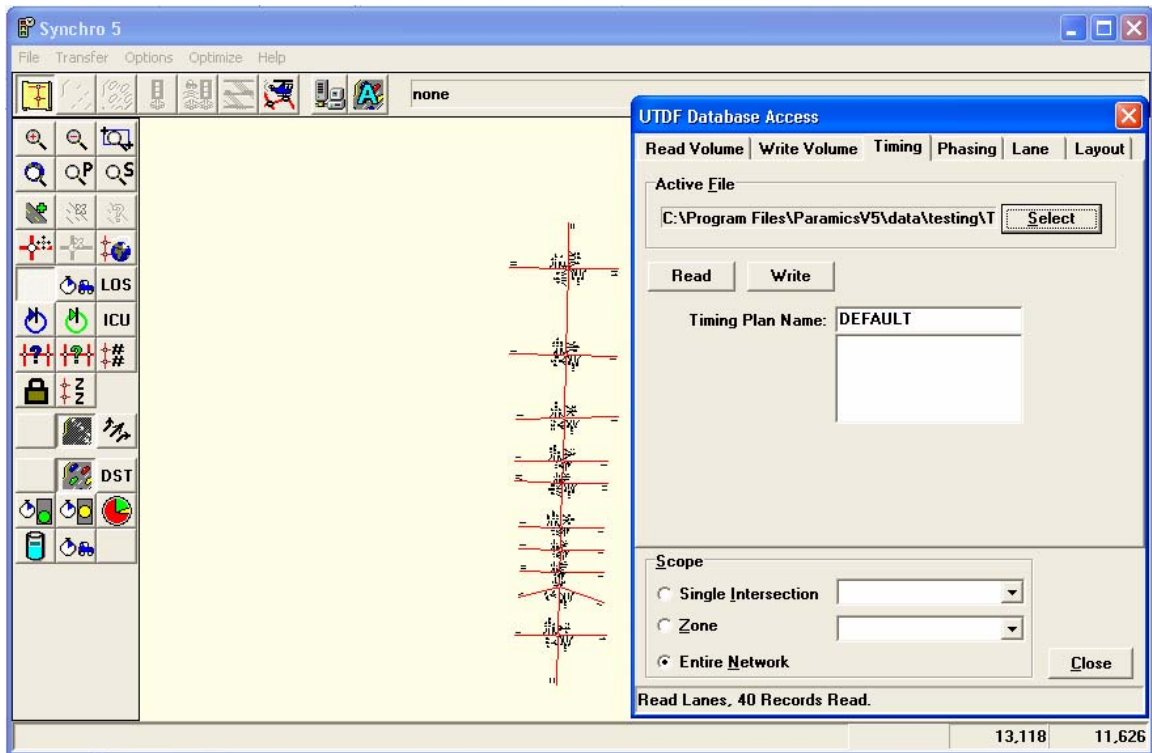


Figure 3.12 Exporting TIMING.DAT File from Synchro

- 9 - Prepare the Paramics network for the second conversion process:
Adjust the programming.modeller file as shown in Figure 3.13 below.

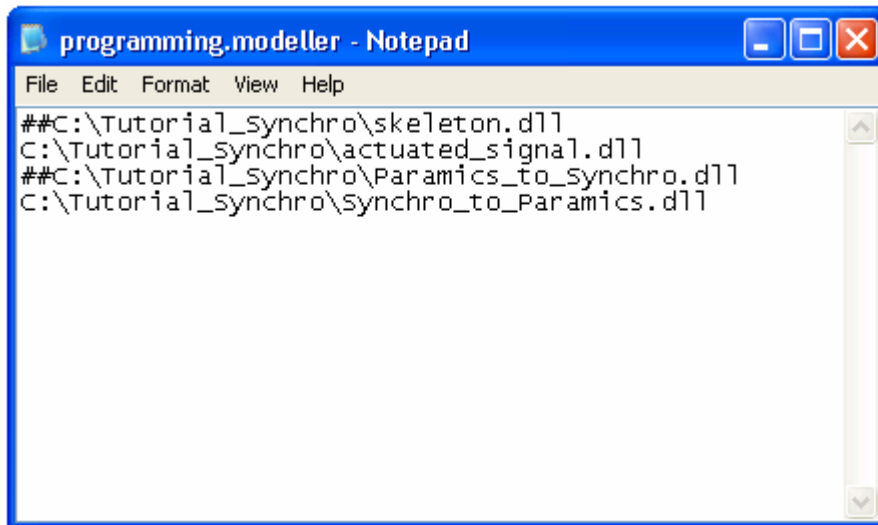


Figure 3.13 Preparing for Second Conversion

10 - Re-open the network in Paramics Modeller:

Load the Tutorial_Synchro network in Modeller. This will cause the second conversion process to take place, where the “signal_control” file is modified according to the optimized signal timing data. This step concludes the second conversion process.

11 - Run the network in Paramics using the new signal timing data:

Before running the network in Paramics, make sure to de-reference the second converter “Synchro_to_Paramics.dll” and re-load the network.

3.6 Conclusion and Approach Limitations

In this chapter we present an approach for integrating microscopic traffic simulation with signal optimization tools. The approach is used to build a two-way conversion procedure between Paramics and Synchro by developing two data conversion plug-ins. The first plug-in reads Paramics network data and traffic volume data and generates two Synchro data files; the second converter transfers optimized actuated signal timing data back to Paramics. These tools are developed to assist traffic engineers in optimizing, evaluating and refining signal timing plans for arterial traffic operations. This chapter also includes a step-by-step tutorial to enable the user to become familiar with the new plug-ins.

The developed Paramics - Synchro optimization tool works well for arterial networks; however, there are still some limitations that need to be considered prior to implementation.

1. It is assumed that the signalized intersections that need to be optimized are all actuated intersections. Therefore, the signal control data in Paramics are stored in the “signal_control” file, and the previous designed “actuated_signal.dll” plug-in [1] must be incorporated within the Paramics simulation.
2. Coordination along the signalized network is not considered. Synchro can generate the coordinated optimizing signal plans; however, the coordinated signal plug-in in

Paramics requires different input files other than the actuated signal plug-in.

Coordination will be considered in further research.

3. The developed tool can not be applied for ramp metering, though it can be applied on a freeway-arterial corridor network.

3.7 References

- 3-1 Liu, H., Chu, L., and Recker, W. Paramics API Development Document for Actuated Signal, Signal Coordination and Ramp Control, California PATH Program Working Paper, UCB-ITS-PWP-2001-11, 2001.
- 3-2 Bullock, D. et al. Hardware-in-the-Loop Simulation. Transportation Research, Vol. 12C, 2004, pp.73-89.
- 3-3 Stallard C. and Owen, L. E. Evaluating Adaptive Signal Control Using CORSIM. Proceedings of the 1998 Winter Simulation Conference, pp. 1147-1153, 1998.
- 3-4 Park B. et al. Evaluating Reliability of TRANSYT-7F Optimization Schemes. Journal of Transportation Engineering, Vol. 127, No.4, pp.319-326, 2001.
- 3-5 Saiyed S. and Stewart J. An Assessment of Pre-timed, Actuated and Adaptive Signal Control Strategies for Unsaturated and Saturated Arterial Network. Paper presented at the 83rd Transportation Research Board Annual Meeting, Washington, D.C., 2004.
- 3-6 Trafficware. Synchro 6 User Guide. 2003.
- 3-7 Hale, D. Traffic Network Study Tool, TRANSYT-7F, United States Version. McTrans Center, University of Florida, 2003.
- 3-8 TSS – Transportation Simulation Systems. AIMSUN Microscopic Traffic

- Simulator: A Tool for the Analysis and Assessment of ITS Systems.
<http://www.tss-bcn.com>, 2002.
- 3-9 PTV America, Inc. VISSIM: Traffic & Transit Simulation. Available at:
<http://www.itc-world.com/vissim.html>.
- 3-10 Quadstone Ltd. Paramics Converter. Available at: http://www.paramics-online.com/products/product_converter.htm
- 3-11 Converting Shape Files for use with Paramics. Available at:
<http://www.ncgia.ucsb.edu/ncrst/research/microsimulation/first.html>
- 3-12 Chu, L., Liu, H., Smolke, B., Recker, W. Paramics Plugin Document – Actuated Signal Control, PATH ATMS Center, University of California, Irvine, 2003.
Available at:
<http://www.its.uci.edu/~lchu/documents/PARAMICS%20Plugin%20-%20signal.pdf>

4 INTEGRATION OF PARAMICS WITH TRANSYT-7F AND VERIFICATION

4.1 Introduction

Chapter 3 introduced some of the benefits of integrating signal timing optimization with microscopic traffic simulation in addition to some of the previous work done on the subject. Also in chapter 3, Plug-ins were developed to integrate Paramics with Synchro and the conversion procedure was detailed.

In this chapter, an approach for building an interface between Paramics and TRANSYT-7F will be presented. The approach is similar to that presented in chapter 3 for interfacing between Paramics and Synchro. Two converters were built, one that creates a TRANSYT-7F network based on the Paramics model, and another one that transfers actuated signal timing data back to Paramics after signal timing optimization in TRANSYT-7F is done. It is worth noting that no such interfaces between Paramics and TRANSYT-7F were found in the literature.

The Paramics rich Application Programming Interface (API) was crucial tool for building the conversion plug-ins. For a detailed description of Paramics and the Paramics API, the reader may refer to the Paramics user guide [4-1].

A TRANSYT-7F network is built as a text-based input file to define the network parameters and settings; the input file format is based on “record types” with certain records that define network elements and other records that define network settings. The features of Paramics and TRANSYT-7F enable communication between the two with relative ease. However, there are conceptual differences between the two with respect to network elements despite the fact that they both use a link/node structure to define their networks. A link in Paramics, for example, may carry traffic with different destinations and may not necessarily be associated with an intersection; nodes in Paramics simply represent connectors between links and do not necessarily represent intersections. In TRANSYT-7F, nodes always represent intersections, and links represent different

movements at the intersection; for example, a northbound through movement and a northbound left-turn movement must be coded as a separate links. Additionally, a Paramics model typically replicates the geometry of the network and the geographic orientation of the links; whereas TRANSYT-7F only builds a simplified representation of the network. To illustrate the difference, Figures 4.1 and 4.2 below show a snapshot of the same intersection in Paramics and TRANSYT-7F, respectively.

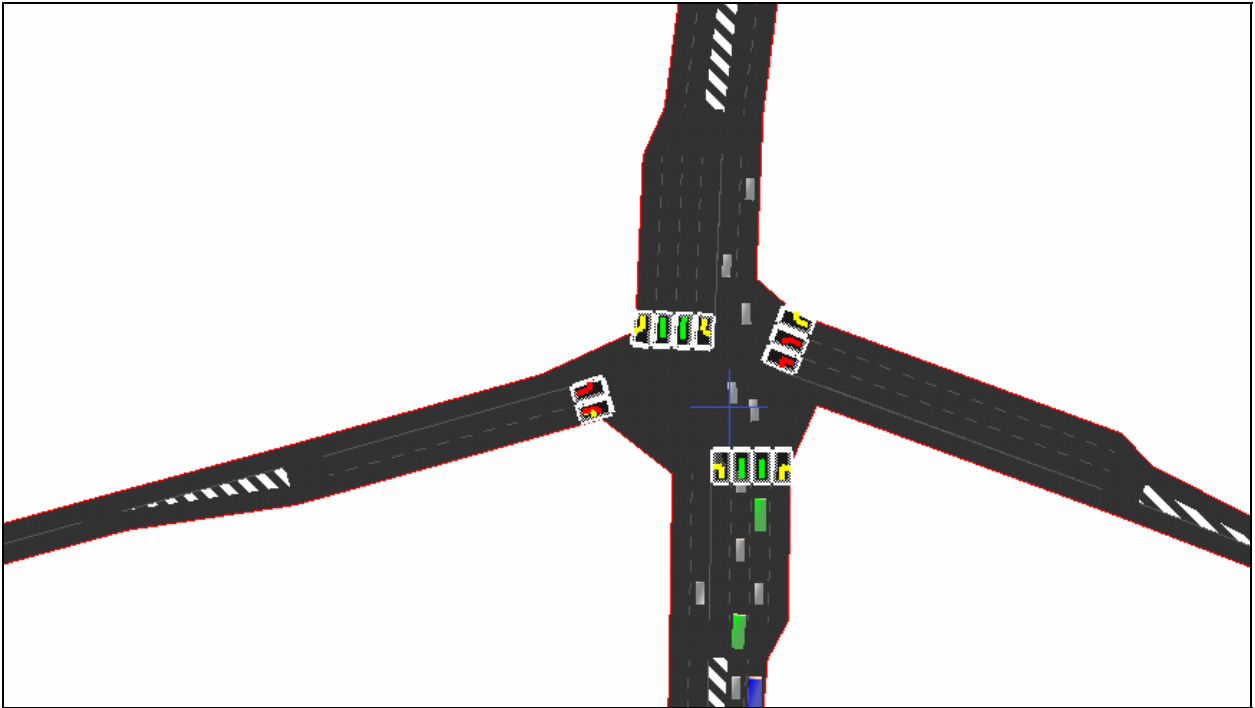


Figure 4.1 Intersection Snapshot in Paramics

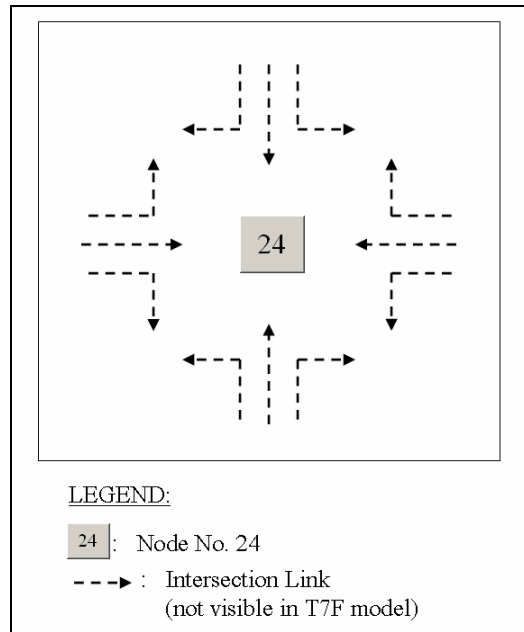


Figure 4.2 Intersection Snapshot in TRANSYT-7F

These differences pose a challenge when translating network elements from a Paramics model into a TRANSYT-7F network, which becomes more difficult with more complex Paramics network geometries. This is due to the link naming convention in TRANSYT-7F, which varies as more turning movements occupy more lanes while also having lanes that are shared with through movements. This link naming convention becomes more complex with intersections that have more than four legs. With complex scenarios such as these, it may be necessary to make some manual adjustments in TRANSYT-7F by the user.

After the conversion takes place and signal timing optimization is carried out, the signal timing plans are transferred back to the Paramics model. The necessary information for this conversion is found in a TRANSYT-7F output text file. Two aspects of this output file make it difficult for conversion to take place. First, actuated signals in the Paramics model use a NEMA phasing scheme, which is not followed by TRANSYT-7F¹. Second, TRANSYT-7F may omit necessary information for some of the movements if they share

¹ The NEMA naming scheme in TRANSYT-7F differs from typical NEMA phasing in that it is associated with links and not signal phases, it extends the standard NEMA convention to include separate links for right-turn movements, and phase 2 is always assigned to the eastbound through movement regardless of the traffic volumes entering the intersection.

phases with other movements. In order to overcome the first difficulty it will be necessary to use a fixed NEMA phase naming scheme in the Paramics model (phase 2 representing northbound through and the remaining 7 are named accordingly) for all signalized intersections in the model regardless of traffic volumes. To overcome the second challenge, the converter simply assumes the same values for the omitted movements as those of the movements they share phases with.

The overall architecture of the integrated Paramics - TRANSYT-7F application was presented in chapter 3. This chapter will focus on the integrated Paramics - TRANSYT-7F application. Section 4.2 will detail the two-way conversion process and discuss implementation of the two converters. This is followed by a step-by-step procedure in section 4.3, which will present in detail how the two converters are used. A step-by-step tutorial is also included in section 4.4 of this chapter. Finally, the limitations of the approach and a conclusion are presented in section 4.5.

4.2 Implementation of an Integrated Paramics – TRANSYT-7F Application

The integration of Paramics with TRANSYT-7F requires a two-way conversion process. Two converter applications were developed; the first builds a TRANSYT-7F input file based on the Paramics model and the second transfers optimized signal timing data back to the Paramics model. A flowchart of the process is shown in Figure 4.3, followed by a description of the two conversion procedures.

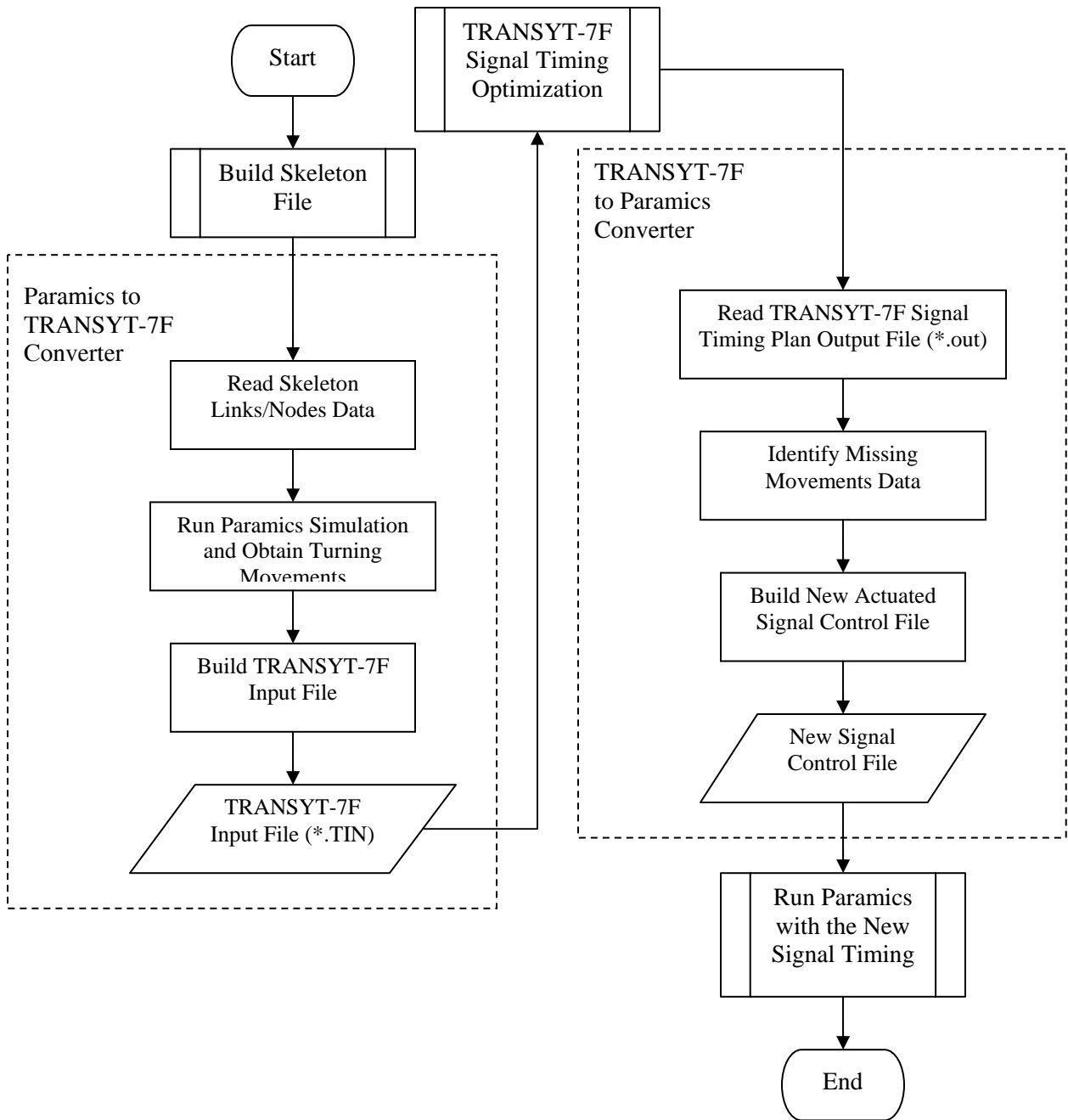


Figure 4.3 Paramics to TRANSYT-7F Conversion Process

4.2.1 The Paramics to TRANSYT-7F Converter Plug-in

The main goal of this conversion process is to build the TRANSYT-7F input file (*.TIN). The first step in the process is to read the skeleton network file to obtain the skeleton nodes and the skeleton links. For further information on the skeleton network, refer to

chapter 3. After reading the skeleton links and the skeleton nodes, the converter builds a signalized intersection information table. It is important to note that this part of the process is internal to the converter and the table is not written to an external file. The table lists the signalized intersection nodes, their coordinates, their associated inbound links, and inbound link attributes such as link length and number of lanes. The next part of the process is to run Paramics and obtain turning movements for each of the signalized intersections. The turning movements are then appended to the intersection information table. Finally, the information table is used by the converter to build the TRANSYT-7F input file.

4.2.2 The TRANSYT-7F Input Data File

A TRANSYT-7F input file is text-based file that is formatted as a series of records that define the network elements, their attributes, the relationships between network elements, and network run settings. The reader is referred to the TRANSYT-7F user manual [\[4-2\]](#) for a detailed description of the record types. The following lists the main TRANSYT-7F record types:

- Run Controls
- Optimization Node List
- Link Numbering Scheme
- Shared Lanes
- Network Parameters
- Signal Timing
- Link Data
- Run Specifications
- Node Coordinates
- Termination

TRANSYT-7F requires these input records in order to build the network. Other record types are available, but not necessary to build a network. The converter transfers the network elements and traffic volumes data to the input file. It does not transfer any

signal timing data from the Paramics model and sets signal timing settings and certain network parameter global settings to TRANSYT-7F defaults. The following lists TRANSYT-7F record issues that the user should note before optimizing the signal timing plans:

1. The user should review the “Run Controls” record after conversion; some of the default global settings may need to be adjusted for certain projects.
2. The user should note that the converter uses the default TRANSYT-7F link numbering scheme instead of a user-defined scheme to enable TRANSYT-7F to attempt to handle unusual scenarios on its own.
3. The user should review the “Network Parameters” record after conversion. The converter assumes a default saturation flow rate of 1,900vplvhg. There are also other global settings in this record that may need to be adjusted for certain projects.
4. The converter sets the network setting units in accordance with the Paramics model. However, if the Paramics model is in metric units, the user should revise some of the global settings in the “Network Parameters” record; namely, “External Approach Speed” and “Vehicle Spacing”.
5. For the “Link Data” record, the user should note that the converter transfers the total traffic volumes for all signalized intersections over the entire simulation period and sets all Peak Hour Factors (PHF) to 1.00. If the simulation period is greater than one hour, it is may be advised to break the simulation period into one-hour periods and optimize signal timing for each of the periods separately.
6. The converter uses TRANSYT-7F defaults for the “Run Specifications” record; for example, simulation parameters are set to multi-cycle step wise simulation.

4.2.3 The TRANSYT-7F to Paramics Signal Timing Converter

The TRANSYT-7F to Paramics converter plug-in was built specifically to transfer optimized signal timing plans to Paramics. This converter is invoked after optimizing the network in TRANSYT-7F. It first reads optimized initial green times, maximum green times, and green extension times for each of the signalized intersections from the

TRANSYT-7F output file. The signal timing output file omits some of the intersection movements if they share phases with other movements. The converter, hence, assumes the same timing plans for these movements as those they share phases with. Actuated signal control data is processed by the actuated signal controller plug-in and is stored in an actuated signal control file. The last step in the second conversion process is to build a new signal control file.

It is important to note that the TRANSYT-7F to Paramics converter only works for actuated traffic signals. If the network traffic signals are all pre-timed, the user will have to manually adjust the signal timing parameters in Paramics based on the design carried out in TRANSYT-7F.

4.3 Step-By-Step Procedure

The two-way conversion procedure for Paramics – TRANSYT-7F integration is the same as that presented in chapter 3 for Paramics – Synchro integration. It is divided into the following six steps:

7. Generating skeleton network data from Paramics
8. Generating the TRANSYT-7F input file
9. Loading the network in TRANSYT-7F
10. Optimizing signal timings with TRANSYT-7F
11. Importing signal timing data from TRANSYT-7F to Paramics
12. Run the network in Paramics with the new signal timing data

The six steps are detailed below.

Step 1: Generating skeleton network data from Paramics

This step is a pre-requisite step to the conversion process. It involves first indicating the location of the “skeleton.dll” plug-in in the “programming.modeller” file for Paramics Modeller simulations. Alternatively, this may be specified in the “programming.processor” file or the “programming.simulator” file if the user prefers to

use Paramics Processor or Paramics Processor from the command line, respectively. This is followed by loading the network in Paramics to generate “skeleton.txt” file under the network folder.

Step 2: Generating the TRANSYT-7F input file

Prior to running the converter, the “skeleton.dll” plug-in should be de-referenced in the Paramics model. The next step is to indicate the locations of the “Paramics_to_T7F.dll” plug-in and the “actuated_signal.dll” plug-in in the Paramics model, in addition to any other necessary plug-ins for the simulation. The network is to then be loaded and run in Paramics. The TRANSYT-7F input file is generated by the converter, “CONVERT_TO_T7F.TIN”, under the network folder after the simulation finishes. At this point, the network has been converted from Paramics to Synchro

Step 3: Loading the network in TRANSYT-7F

Open the TRANSYT-7F input file (CONVERT_TO_T7F.TIN) in TRANSYT-7F.

Step 4: Optimizing signal timings with Synchro

The user can now design a signal timing plan TRANSYT-7F.

Step 5: Importing signal timing data from TRANSYT-7F to Paramics

After the signal timing plan has been designed by the user in TRANSYT-7F, the user is to export the TRANSYT-7F signal timing data by selecting the “Edit -> Analysis” option in TRANSYT-7F, selecting “Estimation” in the “Run Instructions” window, and running TRANSYT-7F. This will generate the “t7fact.out” file, which contains the optimized signal timing data.

Before opening Paramics and importing the timing data, the user should ensure that the “signal_control” file exists under existing Paramics the network folder. The “T7F_to_Paramics.dll” plug-in is to then be referenced in the Paramics model. The network is to then be loaded in Paramics; this will modify the “signal_control” file in accordance with optimized signal timing data generated by TRANSYT-7F.

Step 6: Run the network in Paramics with the new signal timing data

De-reference the “T7F_to_Paramics.dll” plug-in in the Paramics model. This step concludes the two-way conversion process and the network is ready to run with new optimized signal timing data.

4.4 Step-By-Step Tutorial

The purpose of this tutorial is to provide the user with a step-by-step example that covers the conversion process from start to end. The example Paramics network used is based on a main street arterial network in Logan, Utah. It consists of one arterial with ten signalized intersections as shown in Figure 4.4 below. The project name is “Tutorial_T7F” and the project directory for the tutorial is “C:\Tutorial_Synchro”.

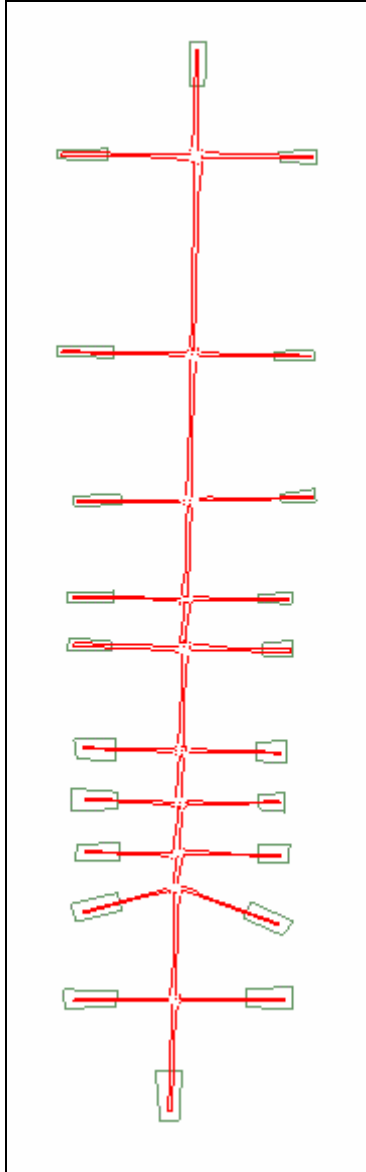
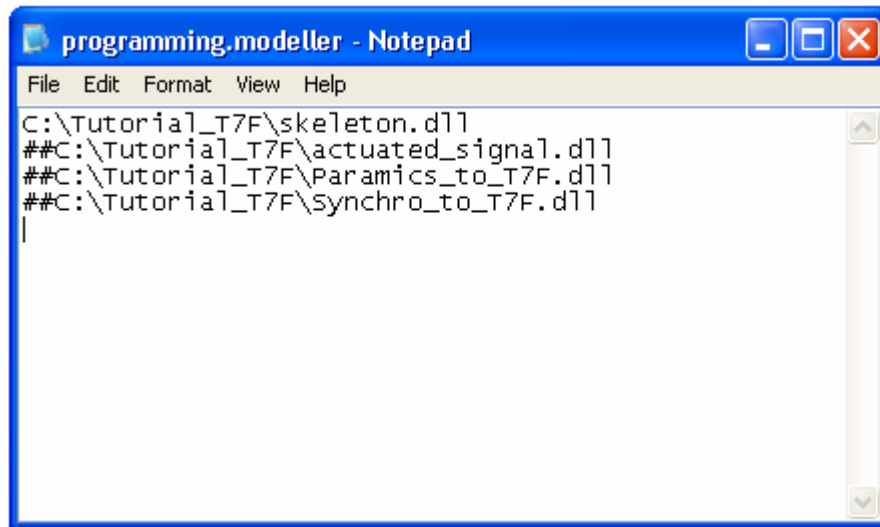


Figure 4.4 Tutorial_T7F Project Layout

Before using the converters, it is necessary to ensure that the actuated signal control plug-in (actuated_signal.dll) is available and an associated actuated signal control file (signal_control) is also available. The reader is referred to the actuated controller plug-in documentation [4-3] for further details. It is important to note that initial timing data in the signal control file are not transferred to TRANSYT-7F by the converter. It is also necessary to have the skeleton network plug-in.

- 1- Prepare the plug-ins for use in Paramics:

Paramics allows for more than one way to use plug-ins. For Tutorial_T7F, all plug-in DLL files will be placed in the project directory and referenced in the “programming.modeller” file. The first plug-in that will be used is the skeleton network plug-in. The remaining three plug-ins are de-referenced at this point in the process by simply placing ‘##’ at the beginning of line calling them in the “programming.modeller” file as shown in Figure 4.5.



```
programming.modeller - Notepad
File Edit Format View Help
C:\Tutorial_T7F\skeleton.dll
##C:\Tutorial_T7F\actuated_signal.dll
##C:\Tutorial_T7F\Paramics_to_T7F.dll
##C:\Tutorial_T7F\synchro_to_T7F.dll
|
```

Figure 4.5 Plug-ins Initial Setup

- 2 - Build the skeleton network file:

After preparing, saving, and closing the “programming.modeller” file, the network is loaded in Paramics Modeller, and then closed without running a simulation. This step will build the “skeleton.txt” file in the project directory that contains all the skeleton link data necessary for the conversion.

- 3 - Prepare the network for the first conversion process:

In this step, the conversion plug-in will be loaded by opening the “programming.modeller” file and modifying it as shown in Figure 4.6.

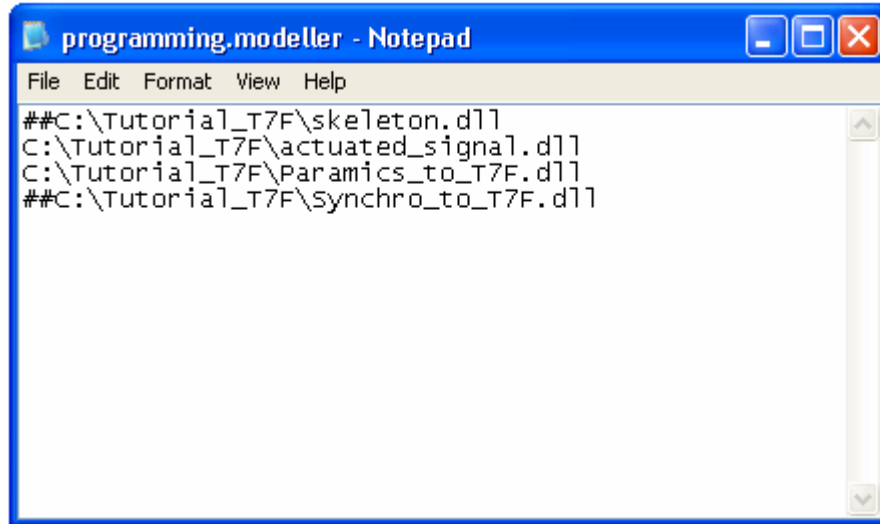


Figure 4.6 Preparing for First Conversion

4 - Convert the network:

Re-opened the network in Paramics Modeller and run a simulation. After the simulation is done, a file named "CONVERT_TO_T7F.TIN" is built in the project directory. This is the TRANSYT-7F input file. At the end of the simulation, Paramics can be closed.

5 - Open the "CONVERT_TO_T7F.TIN" file in TRANSYT-7F:

Open TRANSYT-7F and load the "CONVERT_TO_T7F.TIN" file. To view the TRANSYT-7F network, the user can go to map view and obtain the screen shown in Figure 4.7. The user should not try to optimize the network at this point as this may generate errors.

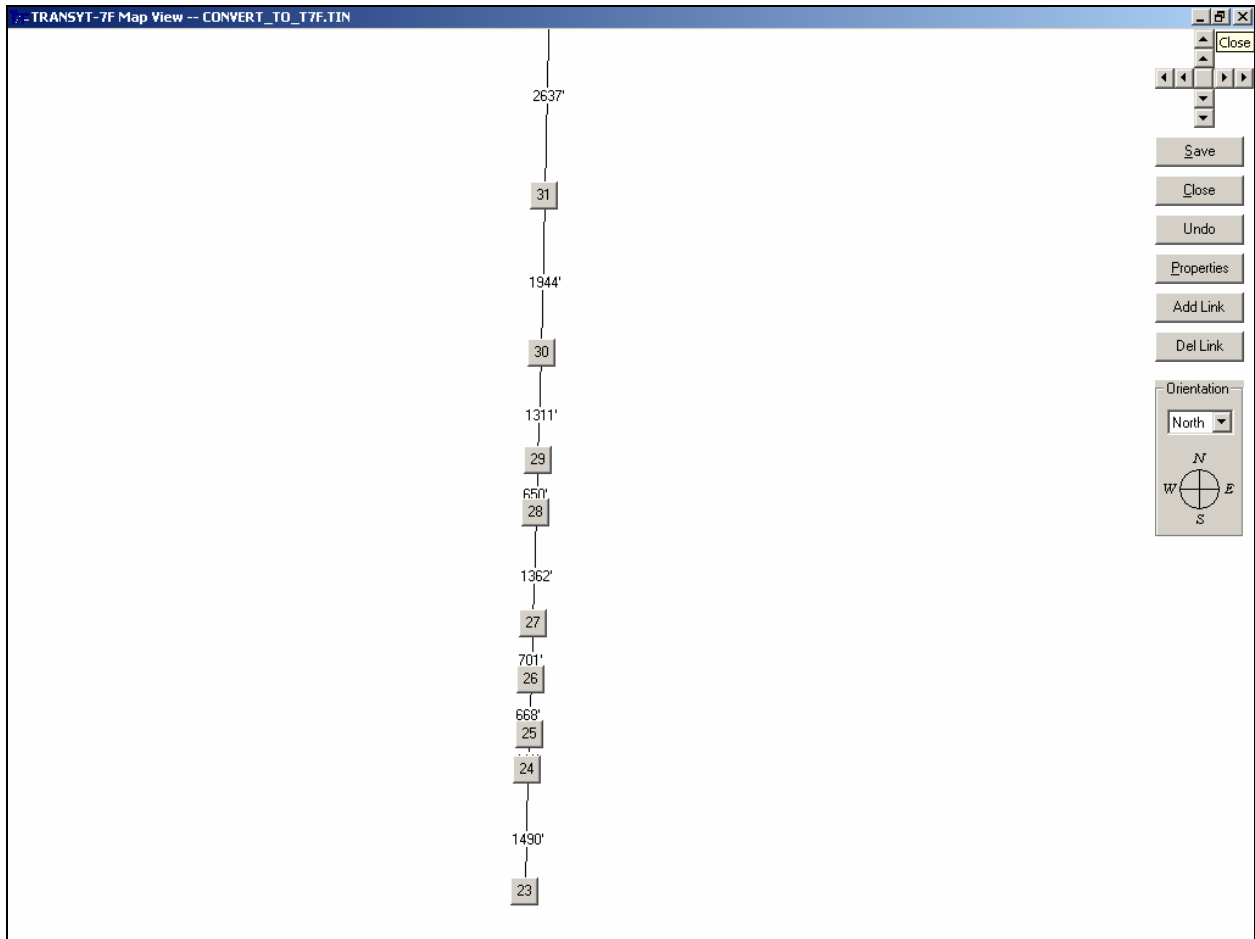


Figure 4.7 TRANSYT-7F Converted Network

6 - Prepare preliminary signal timing settings in TRANSYT-7F:

For the purposes of this tutorial the ‘Preset’ signal timing feature will be used for simplification. This part of the process is left to the user to design and optimize a signal timing plan according to their project requirements. For the Tutorial_T7F network, the signal timing window is launched using the “Edit -> Timing” option in TRANSYT-7F. Under Phase Data in the upper right corner of the timing form, check the “Actuated” and the “Preset” checkboxes and select “LT” in the combo box next to the Preset checkbox. Then also under Phase Data, go to Phase # 3, check the “Actuated” and the “Preset” checkboxes. In the combo box next to the “Actuated” checkbox, select EB; in the combo box next to the Preset checkbox, select “LT”. This step is to be repeated for all network intersections. The user can toggle through

network intersections by using the Node Number Jump combo box in the bottom left of the Timing form. A snapshot of the timing form is shown in figure 4.8.

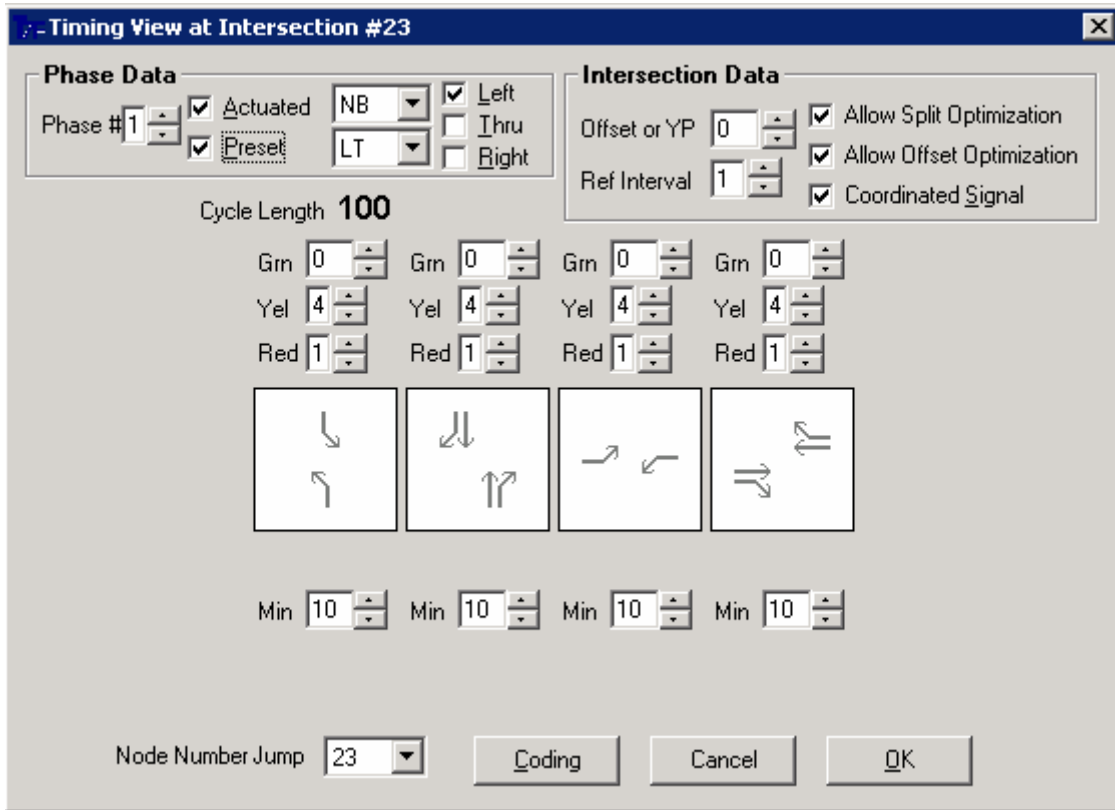


Figure 4.8 TRANSYT-7F Preliminary Timing Settings

7 - Simulate and Optimize the TRANSYT-7F network:

This step involves using the default settings in addition to the preliminary signal timing settings to run a TRANSYT-7F simulation. Where errors are generated, they are to be fixed before attempting to run the optimization. Warnings will be ignored for the purposes of this tutorial. Simulations create punch files in TRANSYT-7F, in this case the file is named “CONVERT_TO_T7F.PUN”. The user can either use these result for actuated signal timing plans estimation or optimize the network first. For the purposes of the tutorial, an optimization step will be carried out using the TRANSYT-7F defaults first. Optimization settings in TRANSYT-7F can be found under “Edit -> Analysis” as shown in Figure 4.9.

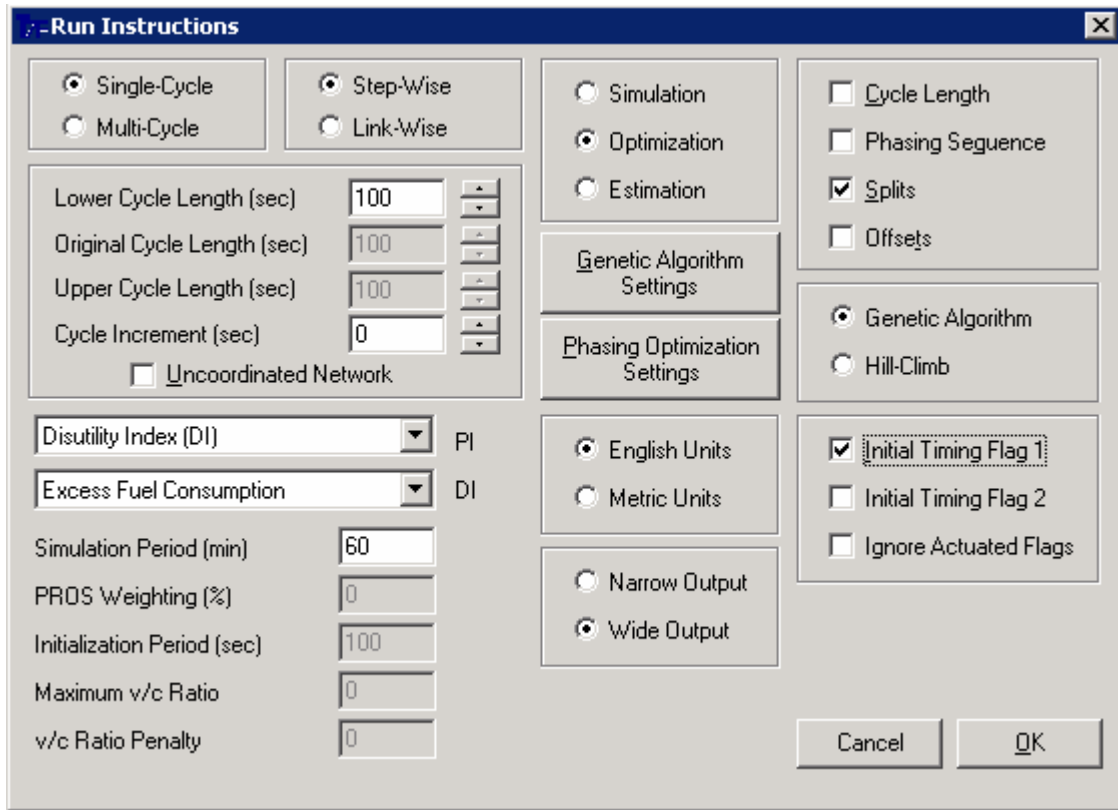


Figure 4.9 TRANSYT-7F Run Instructions

8 - Estimate actuated signal timing plans from the optimized network:

Before moving on the second conversion process, it is important to generate the TRANSYT-7F signal timing data through the estimation output file. First, open the “CONVERT_TO_T7F.PUN” file in TRANSYT-7F and save it as “CONVERT_TO_T7F_01.TIN”. The TRANSYT-7F “Run Instructions” are then adjusted for estimation as shown in Figure 4.10. When this step is completed a new file named “t7fact.out” is created in the project directory. It is essential for the next step of the procedure that the file name and location remain the same.

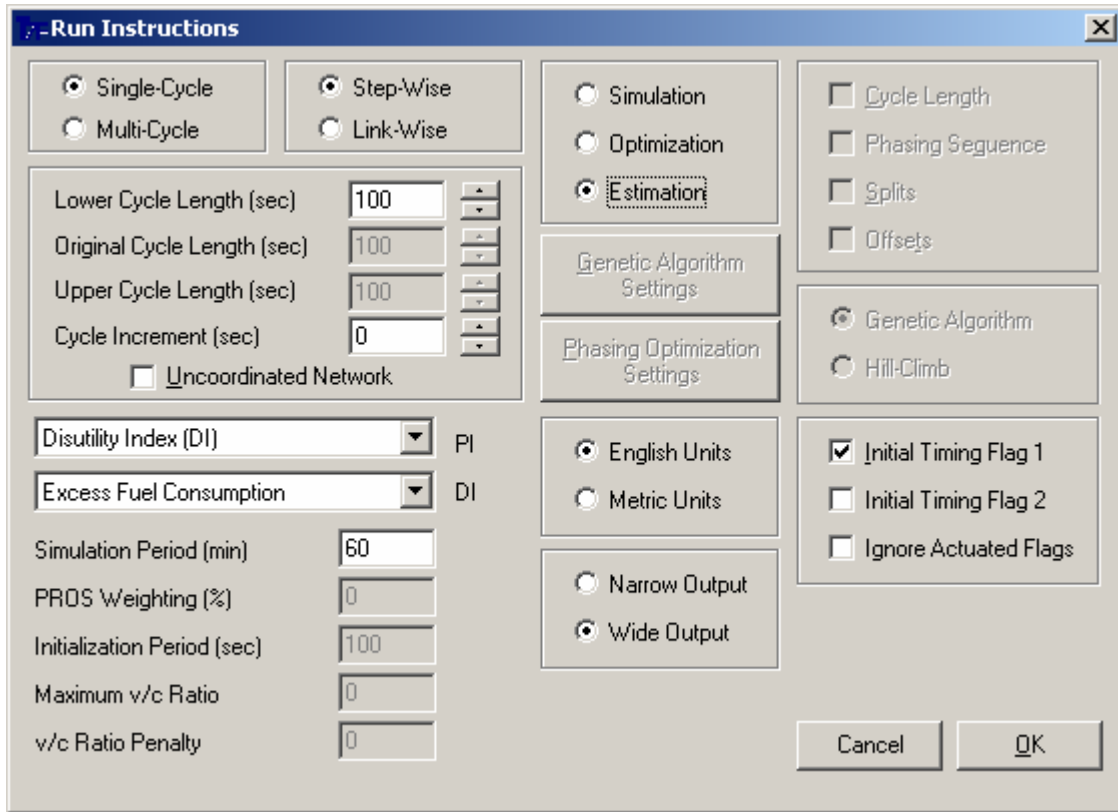


Figure 4.10 Signal Timing Estimation Run Instruction Settings

9 - Prepare the Paramics network for the second conversion process:

Open the programming.modeller file and modify it as shown in Figure 4.11 below.

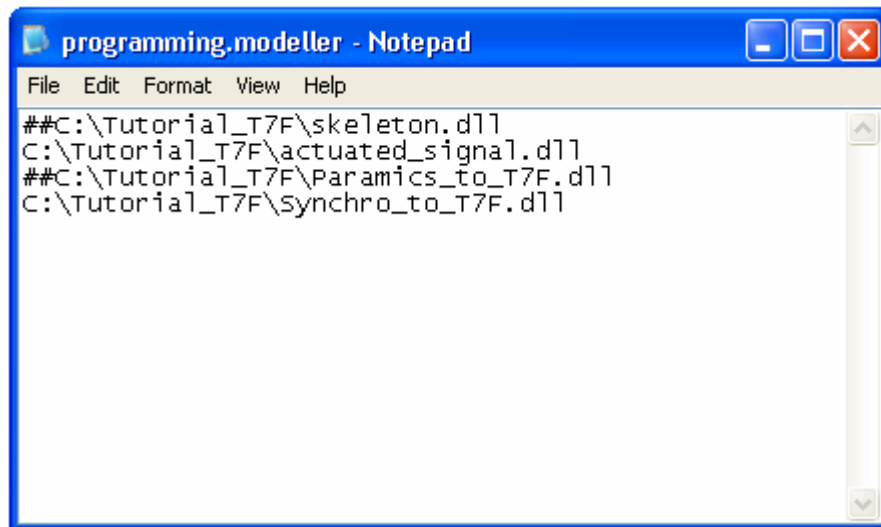


Figure 4.11 Preparing for Second Conversion

10 - Re-open the network in Paramics Modeller:

Load the Tutorial_T7F network in Modeller. This will cause the second conversion process to take place, where the “signal_control” file is modified according to the optimized signal timing data. This step concludes the second conversion process.

11 - Run the network in Paramics using the new signal timing data:

Before running the network in Paramics, make sure to de-reference the second converter “T7F_to_Paramics.dll” and re-load the network.

4.4 Conclusion and Approach Limitations

In this chapter we present an approach to integrate Paramics with TRANSYT-7F. Two converters are built as software plug-ins for use with Paramics. The first plug-in converts Paramics network data and traffic volume data into a TRANSYT-7F input file; the second converter transfers optimized actuated signal timing data back to Paramics. These tools are developed to assist traffic engineers in optimizing, evaluating and refining signal timing plans for arterial traffic operations. This chapter also included a step-by-step tutorial to enable the user to become familiar with the new plug-ins.

There are some limitations to this two-way conversion approach that are mainly either due to the conceptual differences between Paramics and TRANSYT-7F, or certain limitations in TRANSYT-7F that had to be overcome by the converters. The following lists some of these limitations:

1. Due to differences in link and phase movement naming conventions between Paramics and TRANSYT-7F, the user must remember to use a fixed NEMA phase naming scheme in the Paramics model when building the signal_control file in the first step of the process. The converter naming scheme starts with phase 2 representing northbound through and the remaining seven phases are named accordingly for all signalized intersections.

2. The converters are limited to four-leg intersections. Intersections with 5 or 3 legs may be converted to TRANSYT-7F, but with inconsistencies. The converter may still be used for such scenarios; however, the user will be required to manually adjust such intersections in the TRANSYT-7F model and also manually transfer some of the signal timing data back to the Paramics model.
3. Certain situations such as the existence highway off-ramps between signalized intersections that are to be optimized may not be coded correctly in the TRANSYT-7F network. With such scenarios, the intersections will be transferred into the TRANSYT-7F network as isolated intersections. The user will be required to manually connect the isolated intersections in TRANSYT-7F.
4. Node coordinates in TRANSYT-7F can only be whole numbers and only 5-digits in length including negative signs, where negative coordinates are present. This is a major drawback when converting networks from different sources to TRANSYT-7F. It is important that the user checks the coordinates of the network before converting it to TRANSYT-7F. If longer node coordinates are provided, TRANSYT-7F will automatically change the coordinates of the node by truncating some of the digits. This is a TRANSYT-7F limitation and not a converter limitation. Paramics models with node coordinates that are longer than 5-digits will need to be moved to a different reference point. Large Paramics models that cannot be moved in order to adjust all of the node coordinates will need to be split into two or more models and then moved.
5. Another TRANSYT-7F limitation is that it can only handle a maximum number of 99 intersections in a single network. Although this is a large number of signalized intersections for a single network, it is essential that the user takes note of it.
6. TRANSYT-7F link traffic volumes cannot be less than 10, regardless of the traffic volumes collected during the Paramics simulation. The first converter adjusts any TRANSYT-7F link volume less than 10 to 10.

7. In TRANSYT-7F, when connecting intersections in a network, feeder movements are defined for each link. The total volumes of the feeder links should not exceed that of the receiving link. With microsimulation, inconsistencies may arise when vehicles have traversed upstream intersections but have not yet traversed the receiving intersection. Additionally, due to the TRANSYT-7F minimum link volume constraint, some of the total feeder volumes may exceed the receiving link traffic volumes after adjustments are made. In order to overcome this limitation, for such cases the converter increases the receiving link volumes to match those of the total feeder link volumes.
8. For situations where the total feeder volumes are less than the receiving link volume, TRANSYT-7F does automatic volume adjustment; hence, such situations are not handled by the converter.
9. When converting Paramics model data to build a TRANSYT-7F input file, the converter does not transfer signal timing parameters to the TRANSYT-7F network; it is left to the user to do so. It is important to note here that signal timing settings in TRANSYT-7F are sensitive to intersection geometry and unless the geometry is taken into consideration, errors may be generated during this step of the process. The user should run a TRANSYT-7F simulation, check, and fix potential errors.

4.5 References

- 4-1 Quadstone Limited. Quadstone Paramics V5.1 Manual, 2005.
- 4-2 Hale, D. Traffic Network Study Tool, TRANSYT-7F, United States Version. McTrans Center, University of Florida, 2003.
- 4-3 Chu, L., Liu, H., Smolke, B., Recker, W. Paramics Plugin Document – Actuated Signal Control, PATH ATMS Center, University of California, Irvine, 2003.

Available At:

<http://www.its.uci.edu/~lchu/documents/PARAMICS%20Plugin%20-%20signal.pdf>

5 ROBUST TIMING PLANS FOR ISOLATED SIGNALIZED INTERSECTIONS

5.1 Introduction

Many of state-of-the-practice pre-timed systems are operated in a time-of-day mode in which a day is segmented into a number of time intervals, and a signal timing plan is predetermined for each time interval. Typically three to five plans are run in a given day. The basic premise is that the traffic pattern within each interval is relatively consistent and the predetermined timing plan is best suited for the condition of this particular time of day. The timing plan is often obtained by applying Webster's formula [5-1] or using optimization tools such as TRANSYT [5-2] or TRANSYT-7F [5-3], with the inputs of design flows, the mean values of traffic arrivals, for the time-of-day intervals.

Real-world travel demands are intrinsically fluctuating, and traffic arrivals to intersections may vary significantly even for the same time of day and day of week. An issue that traffic engineers may be confronted with is to determine what flows to use to optimize signal timings. This issue was hardly a concern in old days since the data collection used to be resource demanding, and traffic data were only collected for a couple of days. As the advancement of portable-sensor and telecommunications technologies make high-resolution traffic data more readily available, chances for traffic engineers to raise such a question become more prevalent. This is particularly true in re-timing efforts for the closed-loop control systems with fiber optic connections. For example, in California, second-by-second returns of loop data can be obtained and archived via using AB3418 (The California legislature passed legislation, Assembly Bill 3418, requiring all signal controllers purchased in the state after January 1, 1996, to be compliant with a standardized communication protocol).

Use of the average arrivals may not be a sensible choice. Heydecker [5-4] pointed out that if the degree of variability of traffic arrivals is significant, optimizing signal timing with respect to the average arrivals may incur considerable additional delay, compared with the timing obtained by taking this variability into account. For small degree of

variability, use of the average arrivals in conventional calculation methods will only lead to small losses in performance (efficiency). However, as observed later in this paper, it may still result in considerable losses in stability of performance (robustness), thereby causing motorists' travel times unpredictable and unreliable. On the other hand, if the observations of highest flows are used instead, the resulting timing plans may be over-protective and unjustifiably conservative. The average performance (efficiency) is very likely to be inferior. Smith et al. [5-5] suggested using 90th percentile volumes as the representative volumes to generate optimal timing plans and further advised that if time permits, other percentile volumes should be used to compare the results.

This chapter intends to answer the question of what flows to use for signal optimization. More rigorously, this chapter is to investigate methods of signal optimization for pre-timed control under demand fluctuations. In view of the conflicting nature of efficiency and robustness of signal control, a tradeoff would be inevitable to make. The chapter attempts to develop a timing plan whose performance is near optimal in an average sense, and is fairly stable under any realization of uncertain traffic flows.

Such a timing plan also allows a slower deterioration of performance. We note that the signal timing process is normally time-consuming, and thus it is rarely repeated unless changes in traffic conditions are so significant that the system begins performing poorly. It has been estimated that traffic experiences an additional 3%-5% delay per year as a consequence of not retiming signals as the conditions evolve over time [5-6]. Therefore, it would be more desirable to have timing plans that accommodate or tolerate these changes in traffic to a great extent.

Since the seminal work of Webster [5-1], significant efforts have been devoted to improving signal timing for saturated isolated intersections, coordinated arterials and grid networks etc (see, e.g., [5-7], [5-8] and [5-9]). However, only a few studies have been conducted to address signal timing under flow fluctuations for pre-timed control systems. Heydecker [5-4] investigated the consequences of variability in traffic arrivals and saturation flows for the calculation of signal settings and then proposed an optimization

formulation that minimizes the mean rate of delay over the observed arrivals and saturation flows. Following the same notion, Ribeiro [5-10] proposed a novel technique called Grouped Network for using TRANSYT to calculate timing plans that are efficient even when demand is variable.

In this chapter, we present two approaches, scenario-based and min-max, to determine robust optimal signal timings that minimize the mean of delays per vehicle under day-to-day or within-day demand fluctuations as well as maintain a fairly stable performance under those fluctuations. These two robust optimization approaches are simple in structure, and tractable in computation. We demonstrate both approaches on an isolated fixed-time signalized intersection, but the principles and methodology involved are applicable more widely, and can be applied to fixed-time or actuated controls² for arterials and grid networks.

5.2 Scenario-Based Optimization

The scenario-based robust optimization model is a straightforward extension to the principles developed in [5-4] and [5-10]. The model represents uncertainty of traffic flows via a limited number of discrete flow scenarios associated with strictly positive probability of occurrence, and then attempts to optimize signal timings across these scenarios for solutions that are near-optimal and robust with respect to the population of all possible realizations of uncertainty. The concept has been discussed extensively in [5-11], and has been applied to different domains, such as finance, electric utilities and telecommunications (e.g., [5-12]).

5.2.1 Model Formulation

Without loss of generality, we consider an isolated fixed-time signalized intersection. To represent the uncertainty of traffic arrivals to the intersection, a set of scenarios $\Omega = \{1, 2, 3, \dots, K\}$ is introduced. For each scenario $k \in \Omega$, the probability of occurrence is

² Although actuated signals can respond to traffic fluctuations to a certain degree, the underlying timing plan still plays an important role in the determining the efficiency and robustness of the control.

π^k , the traffic flow at lane group i is q_i^k . Given a signal timing plan and a flow scenario k , in order to allow for transient surges of traffic flows, we use the delay equation in Highway Capacity Manual (HCM) to estimate the delay per vehicle:

$$d^k = \frac{\sum_{i=1}^N \frac{C(1-\lambda_i)^2 q_i^k}{2(1-\lambda_i \min(1, x_i^k))} + 900Tq_i^k \left(x_i^k - 1 + \sqrt{(x_i^k - 1)^2 + \frac{4x_i^k}{c_i T}} \right)}{\sum_{i=1}^N q_i^k} \quad (5-1)$$

where d^k denotes the delay per vehicle (sec); N is the number of lane groups; C is cycle length (sec); λ_i is effective green split for lane group i ; x_i^k represents the degree of saturation for lane group i under flow scenario k , equal to $q_i^k / \lambda_i s_i$ and s_i is the saturation flow for the lane group i (veh/h); T is duration of analysis period, 0.25 (h) used in this paper, and c_i is the capacity for lane group i (veh/h), equal to $\lambda_i s_i$. Note that other delay formulae can be applied as well. Dion et al. [5-13] compared several methods for delay estimates at fixed-time signalized intersections, and concluded that the equations defined in 1997 HCM, 1995 Canadian Capacity Guide, and 1981 Australian Capacity Guide and INTEGRATION microscopic simulation produce consistent results under both under-saturated and over-saturated conditions.

Given a set of flow scenarios, we now seek for a robust signal timing plan that minimizes the mean of delays per vehicle across all of the scenarios as well as minimizes the variability of the performance. Since these two objectives conflict with each other, a tradeoff is needed. In this paper we use standard deviation (SD) to represent the variability of performance, and then establish a mean-SD tradeoff. The scenario-based optimization model for determining the cycle length and green times is shown below:

$$\min_{g,C} Z = (1-\alpha) \sum_{k \in \Omega} \pi^k d^k + \alpha \sqrt{\sum_{k \in \Omega} \pi^k (d^k - \sum_{k \in \Omega} \pi^k d^k)^2} \quad (5-2)$$

subject to linear constraints on g and C .

where α is a weighting parameter, $0 \leq \alpha \leq 1$ and g denotes the vector of effective green time for each lane group (sec).

It is easy to see that the first component of the objective function (Equation 5-2) is the mean of the delays per vehicle across all of flow scenarios while the second represents the SD of the delays per vehicle. The parameter α reflects the tradeoff between mean (efficiency) and SD (robustness), varying from zero (minimizing the mean only) to one (minimizing the SD only). Note that when α equals zero, the above problem is equivalent to the model developed in Heydecker [5-4].

5.2.2 Discussions for Model Implementation

Solution algorithm

Note that Equation (5-2) is continuous and the feasibility set defined by linear constraints on g and C is nonempty, closed and bounded. According to Weierstrass' Theorem, there exists an optimal solution to the optimization problem. However, since Equation (5-2) is not convex, it may be difficult to obtain the global optimum. Despite this, the simple structure of the problem (linear constraints only) allows many existing algorithms to solve efficiently for local optima, which often serves well the purpose for engineering applications. A sequential quadratic programming (SQP) subroutine with finite-differencing derivatives in Matlab is used in this paper to solve the optimization problem.

Scenarios

There is no doubt that the scenarios in Ω are only one possible set of realizations of the uncertain traffic flows. Two important questions about the scenario-based robust optimization are: 1) how many scenarios should be included in order to find a solution that is robust across the population of all possible realizations of uncertainty, and 2) how to specify these scenarios and their associated probabilities. Intuitively, the more scenarios we include, the more robust solution we are likely to obtain. However, as the

number of scenarios increases, the problem may become prohibitively large. Fortunately, prior studies (e.g., [5-11] and [5-12]) have shown, confirmed by our computation experiments, that relatively small number of scenarios will be able to produce near-optimal policies. In regard to the other question, as Mulvey *et al.* [5-11] pointed out importance sampling in stochastic simulation can be applied to generate the representative scenarios, if the distributions are known. For real-world applications where the distributions of flows are normally unknown, we suggest selecting 20 to 200 flow scenarios from the field data for the same time-of-day interval, with assuming equal probability of occurrence. The observations of flows can be sorted by their resulting saturation degrees, and then flow scenarios can be determined as observations in the appropriate percentiles.

Choice of weighting parameter

The weighting parameter α represents the tradeoff between efficiency and robustness. Our computation experiments show that although a large value of α leads to more robust signal control, the average performance is far from optimal. Considering traffic engineers may favor efficiency over robustness, we suggest selecting a value between 0 and 0.5. Certainly if time permits, a series of values can be used to create a frontier of non-dominated solutions, from which a compromise solution can be determined eventually.

5.3 Min-Max Optimization

One of the major drawbacks of the scenario-based approach is the additional efforts required to specify the scenarios. Moreover, as the number of scenarios increases, the problem may become computationally demanding. Here we propose another more sensible way to determine a robust optimal timing plan.

The key point is that uncertain traffic flows are assumed to be bounded by a likelihood region. To specify the region, traffic engineers only need to provide their estimates of maximum and minimum likely flows for each lane group (denoted as q_i^{\max} and q_i^{\min}

respectively). Within the region, we then seek for robust timing plans that tolerate changes in traffic arrivals up to the given bound. More specifically, we optimize the signal timing against the worst-case scenario realized within the region. It will be shown later that by selecting an appropriate geometry of the region, we can avoid being either careless (without considering variability of flows at all) or overly conservative. Consequently we will obtain a timing plan that is less sensitive to the fluctuations of traffic flows without losing much optimality.

We further note that the new approach is also attractive in situations on the other extreme where agencies do not have enough resources to do a meaningful data collection but have some prior knowledge about traffic conditions of the intersections. As shown later in the paper, even if the specified maximum and minimum flows are biased, the robust timing approach is able to produce reasonably good and stable timing plans.

The above concept is the basic notion behind a stream of recent research in the area of robust optimization (see [5-14] for an overview). Successful applications of robust optimization can be found in areas such as finance, telecommunication and structural engineering (e.g., [5-15]).

5.3.1 Likelihood Region of Flows

The geometry of the flow likelihood region actually reflects preference or tradeoff that traffic engineers may have between efficiency and robustness. It may be straightforward to assume that traffic flow of each lane group independently varies within the interval of $Q_i = [q_i^{\min}, q_i^{\max}]$. When combined together, the whole likelihood region becomes $\hat{Q} = Q_1 \times Q_2 \times \dots \times Q_N$, where N is the number of lane groups. \hat{Q} is a box centered at the nominal (average) traffic flows $q^0 \in R^N$, a vector with elements of $(q_i^{\max} + q_i^{\min})/2$. See Figure 1 for an illustrative example for an intersection with two one-way approaches N-S and E-W. Since we attempt to optimize signal timing against the worst-case scenario, the corresponding flows would be the one with the highest arrival q_i^{\max} for each lane group.

Obviously it is too conservative, because in the real world it is rare to see that all lane groups experience their corresponding highest flows simultaneously.

Instead, we use the following ellipsoidal region to confine traffic arrivals at intersections:

$$Q = \{q \in R^N \mid q = q^0 + M \cdot u, \|u\|_2 \leq 1\} \quad (5-3)$$

where $M \in R^{N \times N}$, a diagonal matrix with elements of $\theta \cdot (q_i^{\max} - q_i^{\min})/2$. The region can be also written as:

$$Q = \{q \in R^N \mid \sum_{i=1}^N \left(\frac{q_i^{\max} - q_i^{\min}}{2} \right)^{-2} \cdot (q_i - q_i^0)^2 \leq \theta^2\} \quad (5-4)$$

θ is a parameter, reflecting attitudes of traffic engineers towards robustness. The larger is θ , the more preference to robustness. When $\theta = 0$, Q contains q^0 only, suggesting that the signal timing is optimized with respect to the nominal flow, the conventional approaches. When $\theta = 1$, Q would be the largest volume ellipsoid contained in the box region \hat{Q} specified above (See Figure 5.1).

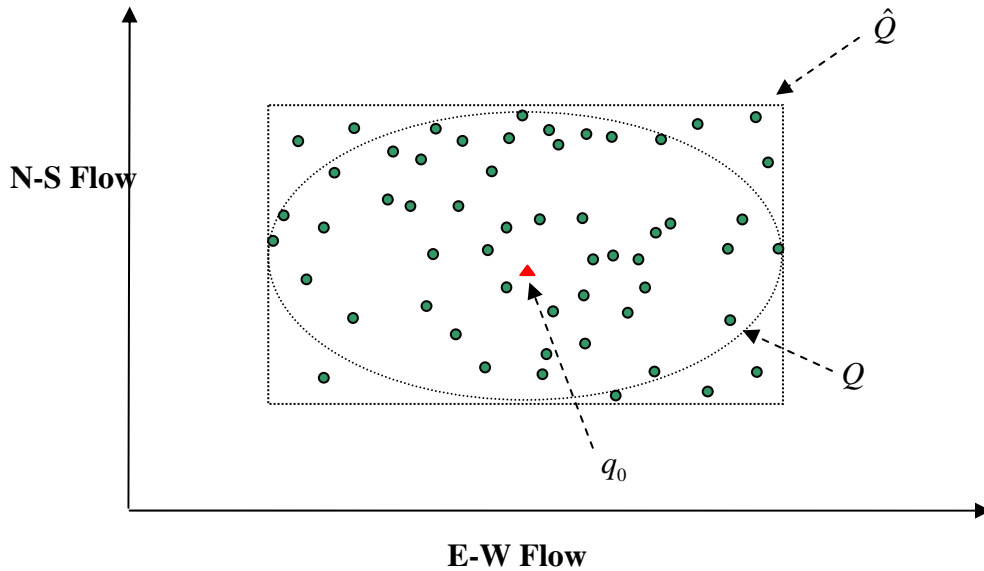


Figure 5.1 Illustrative Example for Flow Likelihood Region

The ellipsoidal region is much more realistic than the box region in the sense that not all lane groups achieve their corresponding highest flows at the same time. However, with an ellipsoidal likelihood region, it is not straightforward to identify the worst-case flow vector, and the vector could be different for different signal settings. The choice of likelihood region will affect the optimality and robustness of the resultant timing plan. However, it should be stressed that when applying the min-max concept, we do not intend to incorporate all the possible realizations of traffic flows into the likelihood region. The minimum or maximum flows by no means are the least or highest possible realizations of uncertain flows. Indeed, it has been shown that even though the region does not contain a single realization of the random vector, the min-max concept still results in a meaningful robust solution [5-15].

5.3.2 Model Formulation

Within the likelihood region of traffic flows, we seek for a robust timing plan that minimizes the delay per vehicle under the worst-case demand scenario bounded by the likelihood region. Mathematically, the robust optimal signal timing for an isolated fixed-time signal can be obtained by solving the following min-max optimization problem:

$$\min_{C, g} \max_{q \in Q} d = \frac{\sum_{i=1}^N \frac{C(1-\lambda_i)^2 q_i}{2(1-\lambda_i \min(1, x_i))} + 900Tq_i \left(x_i - 1 + \sqrt{(x_i - 1)^2 + \frac{4x_i}{c_i T}} \right)}{\sum_{i=1}^N q_i} \quad (5-5)$$

subject to linear constraints on g and C .

In Equation (5-5), the decision variables for the minimization problem is the cycle length and effective green times, and the maximization problem is to determine the worst-case flow vector. Essentially the objective function is to minimize the maximum delay per vehicle possibly incurred within the likelihood region.

5.3.3 Solution Algorithm

There is no readily-available solution algorithm for the above min-max problem. Below we propose a cutting-plane solution algorithm (see, e.g., [5-16]). The algorithm generates the worst-case flow vectors one a time, each of which produces a constraint that cuts away part of the region not feasible to the original problem. Mathematically, the algorithm can be stated as follows:

Step 0: (Initialization) Choose an initial timing plan $g^{(1)}$ and set the iteration counter, n to 1.

Step 1: Given $g^{(n)}$, solve the following (sub) problem:

$$q^{(n)} = \arg \max \{d : q \in Q\}.$$

Step 2: Using $q^{(1)}, \dots, q^{(n)}$ generated in Step 1, formulate and solve the following (master) problem:

$$(g^{(n+1)}, C^{(n+1)}, y^{(n+1)}) = \arg \min_{(g,C,y)} y$$

subject to linear constraints on g and C , and

$$d^k \leq y, \quad k = 1, 2, \dots, n$$

where d^k is the delay per vehicle incurred by $q^{(k)}$, calculated using Equation (5-1).

Step 3: If $\max_i (|g_i^{(n+1)} - g_i^{(n)}|) \leq \delta$, then stop, where δ is a predetermined error tolerance (e.g., one second). Otherwise, let $n = n+1$, go to Step 1.

At the current timing plan $g^{(n)}$, the subproblem in Step 1 finds the flow vector $q^{(n)}$ that yields the maximum delay per vehicle. The subproblem is a quadratically-constrained problem, which can be solved efficiently by using an iterative descent scheme. At each iteration, a localized linear approximation can be formulated using the finite-differencing derivatives: $\max_{\|u\| \leq 1} \nabla d_q^T \cdot M \cdot u$, where ∇d_q is the vector of derivatives of the delay per

vehicle with respect to traffic flows. The optimal solution of this quadratic problem is

$$M \cdot \nabla d_q / \sqrt{(M \cdot \nabla d_q)^T \cdot (M \cdot \nabla d_q)} .$$

The master problem in Step 2 finds the timing plan $g^{(n+1)}$ that minimizes the maximum delay per vehicle among n flow vectors $q^{(1)}, \dots, q^{(n)}$. The problem can be solved efficiently by a SQP algorithm with finite-differencing derivatives.

The proposed algorithm essentially solves a sequence of two nonlinear programming problems. Therefore, the computation effort for obtaining robust signal timings only increases in a polynomial manner. Our computation experiments show that it normally takes about five iterations for the algorithm to converge.

5.4 Numerical Examples

We apply the proposed approaches to an isolated fixed-time four-stage signalized intersection. The input data are shown in Table 5.1 for both under-saturated and over-saturated conditions. A specific lead-lag phasing sequence is used in the example, and the resulting constraints for optimization problems (5-2) and (5-5) are:

$$g_1 + g_2 + g_3 + g_4 + L = C \quad (5-6)$$

$$g_1 = g_6, g_2 = g_5, g_3 = g_7, g_4 = g_8 \quad (5-7)$$

$$g_i \geq g_{\min}, i = 1, 2, \dots, 8 \quad (5-8)$$

$$C_{\min} \leq C \leq C_{\max} \quad (5-9)$$

where L is total lost time per cycle, 14 seconds used in the example; g_{\min} is minimum green time, 8 seconds used, and C_{\min} and C_{\max} are the minimum and maximum cycle length, specified as 50 and 140 seconds respectively.

The computation experiments are conducted as follows:

Signal timing optimization

For the scenario-based approach, 2000 flow samples are first generated assuming traffic arrivals follow normal distributions with means and SDs reported in Table 5.1. These 2000 samples serve as the observations from the field, and are then sorted by their saturation degrees. Flow scenarios are determined as samples in the appropriate percentiles with equal probability of occurrence. Different numbers of scenarios are tested in the computation experiments. Moreover, a series of value of α are used to reflect different attitudes towards robustness and efficiency. Tables 2 and 4 report selected plans resulted by using 500 scenarios and α equal to 0.0 or 0.5 (indicated as scenario-0.0 and scenrio-0.5).

Table 5.1 Input Data

		Under-saturated				Over-saturated			
		Scenario-based		Min-Max		Scenario-based		Min-Max	
Lane Group	Saturation flow rate	Average flow	SD of flows	Minimum flow	Maximum flow	Average flow	SD of flows	Minimum flow	Maximum flow
1	1900	225	65	100	350	275	90	100	450
2	3800	400	100	200	600	525	140	250	800
3	3800	650	125	400	900	875	160	550	1200
4	1900	275	65	150	400	275	60	150	400
5	1900	250	25	200	300	350	75	200	500
6	3800	500	100	300	700	650	175	300	1000
7	3800	650	75	500	800	900	150	600	1200
8	1900	170	25	120	220	250	65	120	380
Saturation degree	-	0.59	-	0.39	0.82	0.80	-	0.47	1.13

For the min-max approach, the minimum and maximum flows in Table 5.1 are specified (the interval is approximately the 95% confidence interval, if flows follow normal distribution specified as above). Given the minimum and maximum flows, the min-max model is solved with values of θ varying from 0 to 1. Tables 5.2 and 5.3 report two selected plans with θ equal to 0.5 or 1.0 (indicated as minimax-0.5 and minimax-1.0).

For the purpose of comparisons, we also use the average flows, 75%, 90% and 100% percentile flows from the generated samples to optimize the signal settings using conventional calculation methods. More specifically, the timing plans are calculated by solving the following problem:

$$\min_{C,g} d = \frac{\sum_{i=1}^8 \frac{C(1-\lambda_i)^2 q_i}{2(1-\lambda_i \min(1, x_i))} + 900Tq_i \left(x_i - 1 + \sqrt{(x_i - 1)^2 + \frac{4x_i}{c_i T}} \right)}{\sum_{i=1}^8 q_i}$$

subject to constraints (5-6)-(5-9).

Tables 5.2 and 5.3 report the resulting timing plans for both under-saturated and over-saturated cases. The plans appear quite different from each other.

Table 5.2 Resulting Timing Plans (Under-Saturated)

Lane Group	1	2	3	4	5	6	7	8	Cycle length
Average	9	9	11	11	9	9	11	11	54
Minimax-0.5	10	9	13	12	9	10	12	13	59
Minimax-1.0	13	11	16	14	11	13	14	16	68
Scenario-0.0	11	10	14	13	10	11	13	14	62
Scenario-0.5	13	11	16	15	11	13	15	16	68
75% percentile	9	9	13	13	9	9	13	13	59
90% percentile	9	12	14	15	12	9	15	14	64
100% percentile	16	13	18	18	13	16	18	18	79

Monte-Carlo simulation

The efficiency and robustness of the resultant timing plans are tested by a macroscopic Monte-Carlo simulation, which is intended to replicate the real-world traffic conditions. Samples of traffic flows are drawn from normal distributions with means and SDs reported in Table 5.1, and for each sample the delay per vehicle resulted by each timing

plan (reported in Tables 5.2 and 5.3) is computed using Equation (5-1). After drawing 5000 samples, the means and SDs of the delays per vehicle incurred by all samples are calculated for all timing plans, and are reported in Tables 5.4 and 5.5.

Table 5.3 Resulting Timing Plans (Over-Saturated)

Lane Group	1	2	3	4	5	6	7	8	Cycle length
Average	16	15	21	21	15	16	21	21	87
Minimax-0.5	20	18	25	25	18	20	25	25	102
Minimax-1.0	24	19	29	29	19	24	29	29	116
Scenario-0.0	19	18	23	24	18	19	24	23	99
Scenario-0.5	22	19	27	27	19	22	27	27	109
75% percentile	30	17	25	23	17	30	23	25	108
90% percentile	20	17	29	33	17	20	33	29	114
100% percentile	25	21	39	40	21	25	40	39	139

Table 5.4 Comparisons of Timing Plans via Monte-Carlo Simulation (Under-Saturated)

	Mean of delays per vehicle	SD of delays per vehicle	Change of mean	Change of SD
Average	37.3	7.8	0.0%	0.0%
Minimax-0.5	36.4	5.6	-2.4%	-28.2%
Minimax-1.0	36.9	4.3	-1.2%	-44.6%
Scenario-0.0	36.1	5.4	-3.4%	-30.8%
Scenario-0.5	36.5	4.4	-2.3%	-43.5%
75% percentile	37.5	6.9	0.5%	-11.6%
90% percentile	40.0	8.9	7.2%	13.8%
100% percentile	38.4	4.1	2.9%	-47.5%

Table 5.5 Comparisons of Timing Plans via Monte-Carlo Simulation (Over-Saturated)

	Mean of delays per vehicle	SD of delays per vehicle	Change of mean	Change of SD
Average	75.9	20.6	0.0%	0.0%
Minimax-0.5	75.3	18.3	-0.7%	-11.3%
Minimax-1.0	77.5	17.1	2.2%	-16.7%
Scenario-0.0	74.8	18.5	-1.4%	-10.0%
Scenario-0.5	75.6	17.4	-0.4%	-15.4%
75% percentile	92.5	23.2	21.9%	12.8%
90% percentile	84.8	21.1	11.8%	2.6%
100% percentile	89.2	19.7	17.6%	-4.5%

By examining Tables 5.4 and 5.5, it is interesting to observe that:

- The plan optimized against the average flows presents a good average performance. This observation is consistent with the conclusion made in Heydecker [5-4] that for small degrees of variability use of mean values in conventional calculation methods will lead only to small loss in efficiency.
- Compared with the one with the average flow pattern, the timing plans resulted from the two approaches (minimax-0.5; minimax-1.0; scenario-0.0 and scenario-0.5) have similar levels of efficiency, but are much more robust. The resulting SDs of the delays per vehicle are reduced by 10% to 16.7%, and 28.3% to 44.6% under over-saturated and under-saturated conditions respectively. This demonstrates that the proposed approaches are able to produce timing plans whose performances are less sensitive to fluctuations of traffic flows without losing optimality.

- There exists a tradeoff between efficiency and robustness. The computation experiments confirm that a larger value of θ or α leads to a more robust but less efficient timing plan. The proposed approaches are flexible in representing different preferences traffic engineers may have.
- Simply using one specific percentile of flows to optimize signals may not be sensible. In our experiments, for the under-saturated cases, higher percentiles of flows result in more robust timing plans without losing much efficiency. However, the results are also mixed. 90%-percentile plan actually increases both mean and SD. More importantly, when over-saturated, use of higher percentiles of flows actually fails to produce neither robust nor efficient timing plans.

To further demonstrate the usefulness of the min-max approach, we conduct another experiment for the situations where traffic engineers do not have enough resources for data collection but rather determine the likely minimum and maximum flows based on their prior knowledge about the intersection. To reflect their biases, which they are very likely to have, we randomly generate their estimates by assuming the deviations to the “unbiased” minimum and maximum flows reported in Table 5.1 are independently uniformly distributed between $[-75, 75]$. Table 5.6 reports only three estimates (randomly generated and selected), for the over-saturated case. Given these estimates, the min-max approach is then used to optimize the timing plans, which are then evaluated using the same Monte-Carlo simulation. Table 5-6 reports the evaluation results, compared with the one using the true average flow (reported in Table 5-5). It can be seen that even with biased estimates of minimum and maximum flows, the min-max approach is able to generate timing plant that performs reasonably well and stably.

Table 6 Impacts of Specified Minimum and Maximum Flows

Lane Group	Case 1			Case 2			Case 3		
	Minimum flow	Maximum flow	Timing	Minimum flow	Maximum flow	Timing	Minimum flow	Maximum flow	Timing
1	137	441	26	29	395	23	133	466	21
2	215	800	18	222	756	18	279	727	19
3	541	1157	28	477	1216	28	488	1128	26
4	215	422	30	133	420	28	143	354	27
5	227	473	18	227	481	18	191	513	19
6	257	1069	26	239	1011	23	278	934	21
7	651	1234	30	530	1193	28	548	1180	27
8	139	367	28	137	312	28	146	400	26
Cycle length	116			112			106		
Mean of delays per vehicle	81.4			77.8			75.9		
SD of delays per vehicle	18.5			17.7			18.2		
Change of mean	7.3%			2.6%			0.1%		
change of SD	-10.3%			-13.9%			-11.8%		

5.5 Conclusion

We have presented two approaches, scenario-based and min-max respectively, to determine robust optimal timing plans, and demonstrated both approaches on an isolated fixed-time signalized intersection. It has been shown that the proposed approaches are able to produce timing plans whose performances are less sensitive to fluctuations of traffic flows without losing optimality. Both approaches are practical, and simple to implement. The min-max approach may be more sensible in the sense that it requires neither an extensive data collection effort nor a prior knowledge of distributions of traffic flow. The model simply works with engineers' estimates on minimum and maximum possible traffic arrivals. Moreover, the results from the model are not so sensitive to those estimates.

Future research is needed to demonstrate the proposed approaches with more sophisticated signal control systems for corridors and grid networks. Moreover, it may be necessary to conduct microscopic traffic simulation studies to further verify the proposed approaches.

5.6 Reference

- 5-1 Webster, F.V. (1958), *Traffic Signal Settings*, Road Research Technical Paper No.39, Her Majesty's Stationery Office.
- 5-2 Vincent, R.A., Mitchell, A.I. and Roberson, D.I. (1980) *User Guide to TRANSYT Version 8*. TRRL Report LR888, Transport and Road Research Laboratory, Crowthorne, 1980.
- 5-3 Wallace, C. E., Courage, K. G., Hadi, M. A. and Gan, A. G. (1998) *TRANSYT-7F User's Guide*, University of Florida, Gainesville, FL, 1998.
- 5-4 Heydecker, B. (1987) Uncertainty and variability in traffic signal calculations. *Transportation Research*, Part B, Vol.21, 79-85.
- 5-5 Smith, B. L., Scherer, W. T., Hauser, T. A., and Park, B. B. (2002) Data-driven methodology for signal timing plan development: a computational approach, *Computer-Aided Civil and Infrastructure Engineering*, 17, 387-395.
- 5-6 Luyanda, F., Gettman, D., Head, L., Shelby, S., Bullock, D. and Mirchandani, P. (2003). ACS-Lite algorithmic architecture: applying adaptive control system technology to closed-loop traffic signal control systems. Design guidelines for deploying closed loop systems. *Transportation Research Record 1856*, TRB, National Research Council, Washington, D.C., 175-184.

- 5-7 Gazis, D. C. (1964) Optimum control of a system of oversaturated intersections. *Operations Research*, 12, 815-831.
- 5-8 Robertson, D.I. and Bretherton, R.D. (1991) Optimizing networks of traffic signals in real-time: the SCOOT method. *IEEE Trans. Vehicular Tech*, Vol.40, No.1, 11-15.
- 5-9 Gartner, N. H. (2002) Development and implementation of an adaptive control strategy in a traffic signal network: the virtual-fixed-cycle approach. In *Proceedings of the 15th International Symposium on Transportation and Traffic Theory* (edited by A.P. Taylor), 137-155.
- 5-10 Ribeiro, P.C.M. (1994) Handling traffic fluctuation with fixed-time plans calculated by TRANSYT. *Traffic Engineering and Control*, Vol.35, 362-366.
- 5-11 Mulvey, J.M., Vanderbei, R.J. and Zenios, S.A. (1995) Robust optimization of large-scale systems. *Operations Research*, Vol.43, No.2, 264-281.
- 5-12 Laguna, M. (1998) Applying robust optimization to capacity expansion of one location in telecommunications with demand uncertainty. *Management Science*, Vol.44, No.11, 101-110.
- 5-13 Dion, F., Rakha, H. and Kang, Y-S. (2004) Comparison of delay estimates at under-saturated and over-saturated pre-timed signalized intersections. *Transportation Research*, Vol.38B, 99-122.
- 5-14 Ben-Tal, A. and Nemirovski, A. (1999) Robust solutions to uncertain linear programs. *Operations Research Letters*, 25, pp. 1-13.

- 5-15 Ben-Tal, A. and Nemirovski, A. (2002) Robust optimization – methodology and applications, *Mathematical Programming*, Ser.B. 92, 453-480.
- 5-16 Bazaraa, M.S., Sherali, H.D. and Shetty, C.M. (1993) *Nonlinear Programming: Theory and Algorithms*. Second Edition, John Wiley & Sons, New York, New York.

6 SIGNAL TIMING STRATEGIES FOR TIME-OF-DAY CONTROL

6.1 Motivation

Availability of real-time loop traffic data and signal status data provides traffic engineers opportunities to further improve the effectiveness of current closed-loop control systems. Given that the TOD controls currently used by Caltrans perform well when timed properly, we will not develop traffic response plan selection algorithms because of the highly transition costs involved [6-1].

This chapter presents a methodology that further improves the ability for generating efficient TOD plans for arterial corridors, by developing a systematic approach making use of archived real-time traffic data. We do not intend to develop a new signal optimization model, but rather to optimally determine TOD intervals and the corresponding representative traffic volumes.

Once TOD intervals have been formed, the second step of the timing process is to identify the representative volumes for the TOD intervals. For each TOD interval, we actually have numerous archived traffic volumes, characterized by an uncertainty set. The cluster centroid volumes and 90th percentile volumes have been recommended as the representative volumes to generate optimal timing plans. As pointed out in Chapter 5, optimizing signal timing with respect to these two volumes may incur considerable additional delay, compared with the timing obtained by taking this variability into account. Therefore, a more sensible way is to apply models developed in Chapter 5 to seek for a robust plan that tolerates changes in traffic demands and remain “close” to its *designated* performance under any realization of traffic fluctuations.

6.2 Background

6.2.1 Problem Formulation

Consider an arterial corridor with I intersections. Each intersection is equipped with loop detectors as in figure 6.1. Loops may be advanced or presence loops, and measure the flow $q_{ij}(t)$ on every approach ij in Figure 6.1, and also the occupancy $o_{ij}(t)$. (Vehicles on approach ij are approaching intersection i from direction j .)

It is further assumed that turning proportions are available.

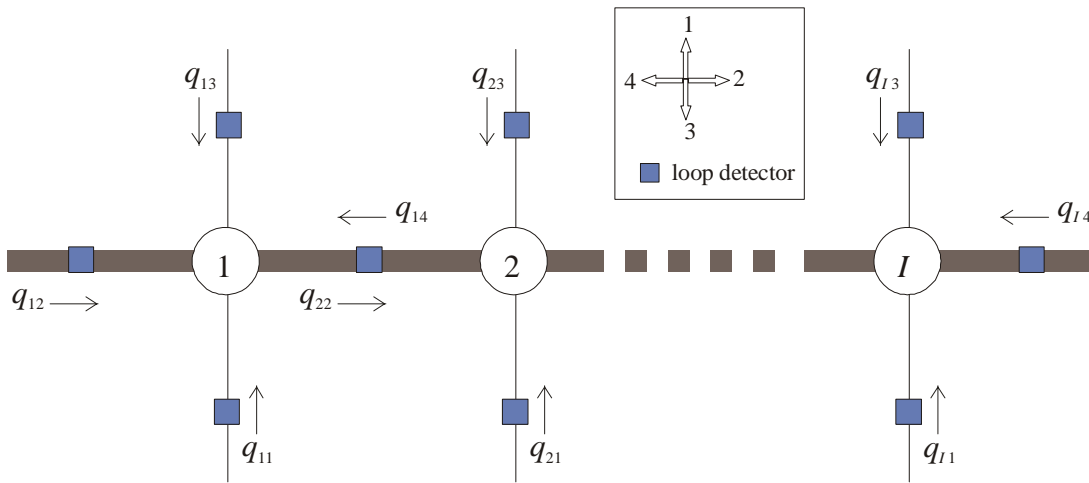


Figure 6.1: Arterial Corridor Configuration

The problem is then to find an optimal way to divide a whole day into n consecutive TOD intervals so that *traffic conditions* in each interval are “clustered” as closely as possible for a set of coordinated signalized intersections. In this way, in each TOD interval traffic conditions will be more or less homogeneous and signal-timing parameters can be optimized separately.

We also need to find the optimal number of TOD intervals n^* that would best represent the entire analysis period, subject to the constraint of a maximum number of TOD intervals. Normally, the maximum number of TOD pre-determined, considering the

capability of the controllers and signal plan transition problem (a typical closed-loop system in Caltrans District 4 stores the combination of cycle, splits and offsets as a pattern, and a 170 controller can store nine patterns).

6.2.2 Existing Methods

Up to recently, no automated tools exist to help engineers determine appropriate TOD intervals. Current practice uses single day, hand counted volumes at the critical intersection of the corridor to define the state for TOD plan development. This is generally done by plotting aggregate traffic volumes over the course of a day, and then using judgment in the identification of significant changes in traffic volume. Notice that the volumes used to identify TOD intervals are bi-directional aggregate volume values from the critical intersection.

Recently, only to research efforts have been developed to automate the process using a data mining approach [6-2,6-3].

Statistical clustering and classification analyses was proposed in [6-3]. We believe that the problems with this for formulation are:

- Definition of *traffic conditions*: they used the standardized flows and occupancies of every phase at every intersection in the system as the state vector. As discussed earlier, demands should be used instead of flows in order to insure the consistency of the process.
- Including occupancies does not add information given that they are correlated with flows, as per the lane fundamental diagram.
- Finally, standardization has the effect of weighting observations according to their deviations from the phase's average; it is more reasonable that the weight be proportional to the magnitude of observation.

This second point motivated the authors of [6-2] to investigate a method based on a genetic algorithm (GA) that optimizes TOD breakpoints with explicit consideration of

signal timing performance at a representative intersection. This method implements a two-stage optimization: an outer loop for TOD breakpoints and an inner loop for timing plans then resulting delays of corresponding TOD intervals. Unfortunately, the formulation in [6-2] is very incomplete and seems to have fundamental flaws. For example:

- This method will yield TOD intervals consistent with homogeneous delays but not necessarily homogeneous flows. We believe it is not a good choice because delays are certainly dependent on the signal timing parameters and on the history of the system when it is oversaturated.
- The authors choose the “optimal” signal timing parameters at a representative intersection according to the HCM formulas. The resulting delays may have nothing to do with the delays on the actual corridor.
- It is not clear how to choose the "representative intersection". The critical intersection seems to be the best choice, but this is not explicit from their analysis.
- The state variables are not defined.

In summary, we conclude that past efforts may yield inconsistent results because they are based on endogenous traffic conditions (flows, densities and delay). Therefore, the control parameters of the system (e.g., saturation flows, signal timing parameters) influence the optimal TOD plan, and the results would depend on control parameters in place at the time of the data collection. In these circumstances, one would need elaborate modeling of Traffic Dynamics in the network to take those factors into account, which is computationally prohibitive.

6.2.3 Clustering Algorithm

The term cluster analysis (first used by Tryon, 1939) encompasses a number of different algorithms and methods for grouping objects of similar kind into respective categories. Cluster analysis is an exploratory data analysis tool which aims at sorting different objects into groups in a way that the degree of association between two objects is maximal if they belong to the same group and minimal otherwise.

Clustering algorithms may be classified as:

- Hierarchical Clustering
- K -means clustering

6.2.3 The K -Means Clustering Algorithm

The K -means method [6-4] partitions N data points into K disjoint subsets S_j containing N_j data points so as to minimize the sum-of-squares criterion

$$J = \sum_{j=1}^K \sum_{x \in S_j} |x_n - \mu_j|^2,$$

where x_n is a vector representing the n^{th} data point and μ_j is the geometric centroid of the data points in S_j . The algorithm consists of a simple re-estimation procedure as follows.

1. the data points are assigned at random to the K sets.
2. the centroid is computed for each set.
3. repeat steps 1 and 2 until a no further change in the assignment of the data points.

In general, the K -means method will produce exactly K different clusters of greatest possible distinction. It should be mentioned that the best number of clusters K leading to the greatest separation (distance) is not known as *a priori*.

6.2.4 Hierarchical Clustering

Given a set of N items to be clustered, and an $N \times N$ distance (or similarity) matrix, the basic process hierarchical clustering is this:

1. Start by assigning each item to its own cluster,
2. Find the closest (most similar) pair of clusters and merge them into a single cluster,
3. Compute distances (similarities) between the new cluster and each of the old clusters.
4. Repeat steps 2 and 3 until all items are clustered into a single cluster of size N .

Table 6.1 presents the computation time and memory requirements for K-means and Hierarchical Clustering methods, where:

- t : number of iterations for K-means
- $k \approx 6$
- n : number of 15-min time intervals in one day (≈ 70)
- m : number of approaches ($\approx 2I$)

Table 6.1: K-means vs. Hierarchical Clustering

Method	Computation Time	Memory Requirements
Hierarchical clustering	$O\{ m n^2 \log(n) \}$	$O\{ n(m + n) \}$
K-means clustering	$O\{ k t m n \}$	$O\{ n(m + k) \}$

Since $n \gg k$ in our application, the K-means method requires the least memory and should run faster.

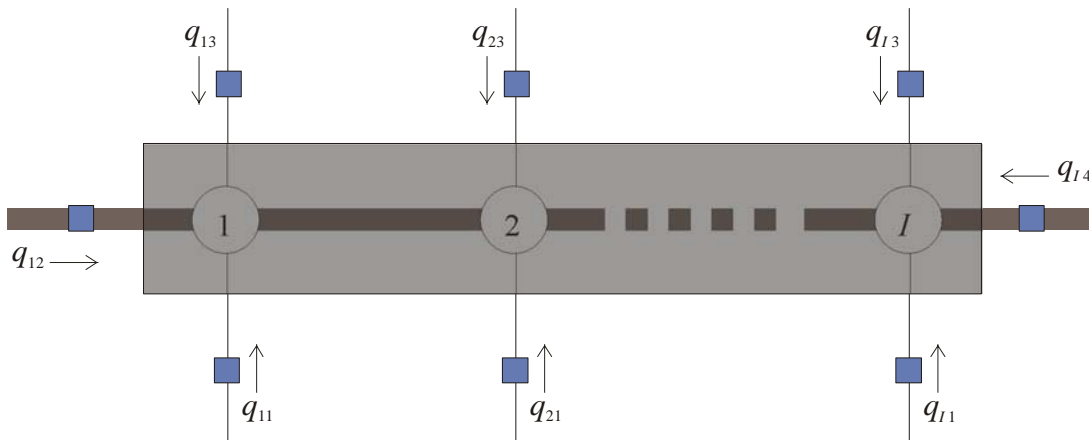


Figure 6.2: Measurement Locations for Proposed Method

However, the problem with these methods is that the optimal number of clusters n^* is not known as a priori, and determining it may become time consuming and subjective. The method presented next determines n^* automatically.

6.3 Proposed Method

The proposed method is based on exogenous state variables of the problem. In this way we insure the consistency of the results because the system control parameters in place at the time of data collection do not affect the outcome of the method.

6.3.1 System Definition

The proposed system definition is shown in Figure 6.2. This definition recognizes that only the measurement locations shown in the figure are necessary, given that any other link flow in the system can be obtained from these. For example, q_{22} in Figure 6.1 is a result of q_{11} , q_{12} , q_{13} and the signal timing parameters at intersection 1.

6.3.2 State Variables

The proposed definition of state variables (or traffic conditions) is based on the *desired inflows* or *demands* to the system shown in Figure 6.2. Demand equals the flow measured by the loop detector when there is no congestion. In the presence of a *permanent queue*, however, demand will be higher than the flow on the loop because it will be dictated by the discharge rate of the downstream bottleneck. In such a case, one has to expand the analysis area until reaching uncongested intersections.

The state of the system at time t , $Y(t)$, is defined as a vector containing all the *flow ratios* in the system; i.e.:

$$Y(t) = \{ y_{12}, y_{14}, y_{ij} \}, \quad \text{for all } i = 1, 2 \dots I \text{ and } j = 1, 3$$

$$y_{ij}(t) = q_{ij}(t) / s_{ij}$$

where s_{ij} is the saturation of flow on approach ij . In this manner, we insure that intervals will be homogeneous in terms of flows *and* delays. The former is true since intersection delay depends on the degree of saturation, which is proportional to the flow ratios.

6.3.3 Solution Methods

The most efficient method to solve our problem is *dynamic programming*. The problem can be divided into stages with a decision required at each stage. Each stage has a number of states associated with it. In our case stages are the TOD intervals and the states the time of which they end.

But the problem can be solved using standard shortest path algorithms over the following network:

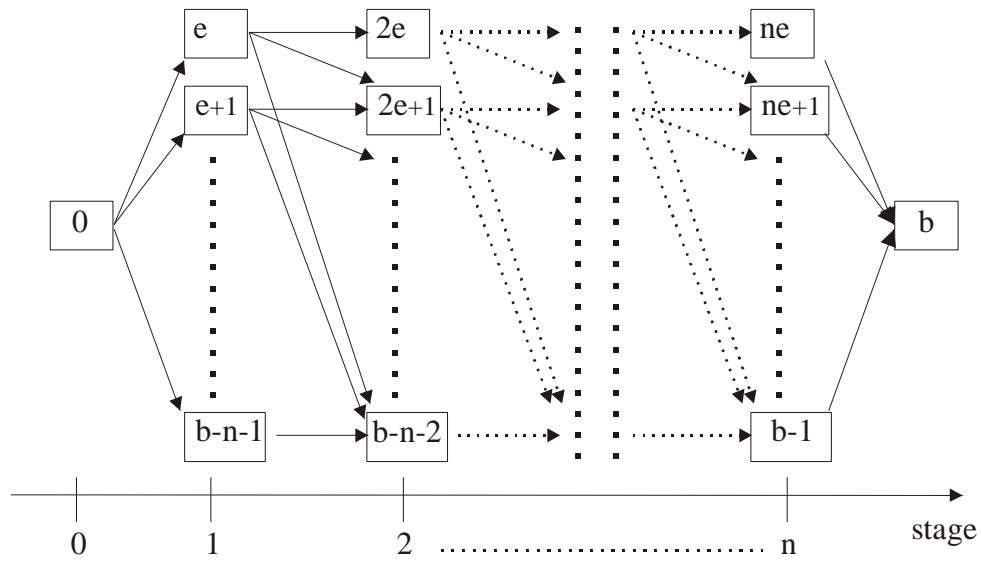


Figure 6.3 Solution Network

where n is the maximum number of TOD intervals, e is the minimum duration of a TOD interval (eg, 1 hour) and b is the last time interval of the day (eg, 96 when using 15 min intervals). The nodes of this network represent the time interval at which the TOD interval (stage) ends. Notice that the optimal number of clusters n^* can be determined automatically simply by introducing links connecting every pair of non-consecutive stages.

The cost on any link $j-k$ can be computed as the area shown in Figure 6.4. This area corresponds to the cumulative difference between the actual demand curve and the demand used for signal timing.

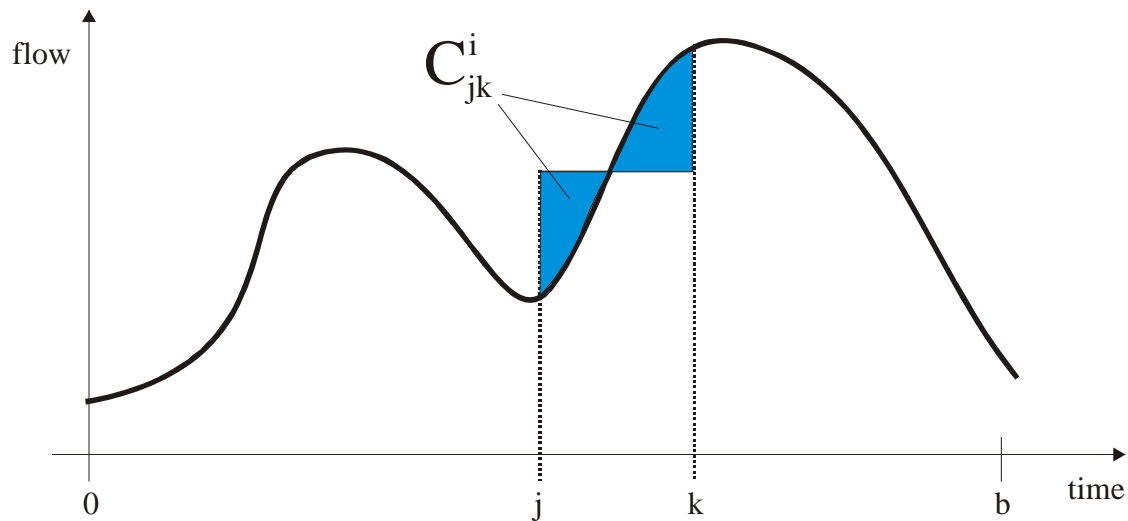


Figure 6.4: Definition of Cost on Link $j-k$.

6.4 Conclusions

The proposed method is based on exogenous state variables of the problem. This improves current practice since we insure that the resulting TOD definition is independent of the system control parameters in place at the time of data collection. Additionally, the proposed dynamic programming approach to solve the problem is fast and gives the optimal number of clusters as an output.

6.5 References

- 6-1. Mussa, R. and M.F. Selekwa, *Optimization of signal timing transition period*. Urban transport X: urban transport in the 21st century., 2004: p. 689-697.

- 6-2. Park, B., et al., *Optimization of time-of-day breakpoints for better traffic signal control*. Freeway Operations and Traffic Signal Systems 2004, 2004(1867): p. 217-223.
- 6-3. Smith, B., et al., *Data-Driven Methodology for Signal Timing Plan Development: A Computational Approach*. Computer-Aided Civil and Infrastructure Engineering, 2002. 17: p. 387-395.
- 6-4. MacQueen, J.B., *Some Methods for classification and Analysis of Multivariate Observations*. Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, 1967. 1(281-297).

7 DEVELOPMENT OF A PROTOTYPE SIGNAL OPERATIONS MONITORING SYSTEM

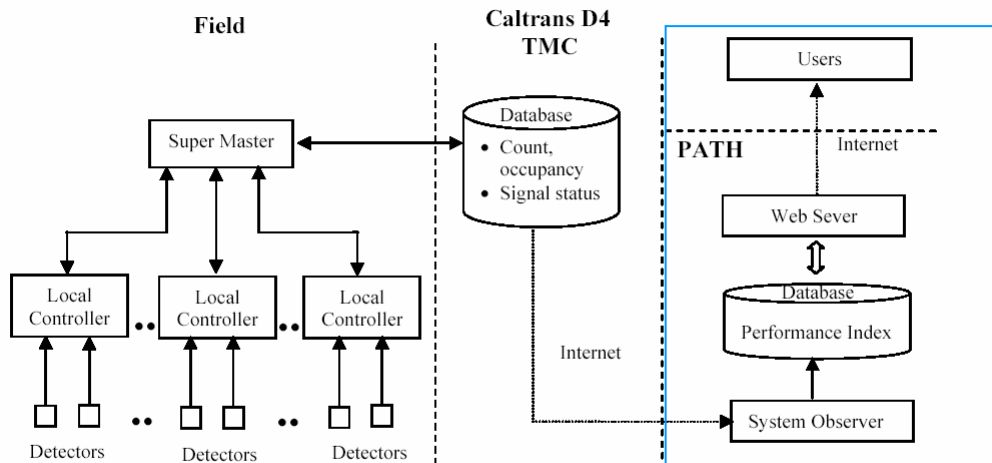
7.1 Introduction

The main objective of this chapter is to develop a prototype signal operation monitoring system that enables the Caltrans traffic operation staffs to monitor the performances of signal control operations. The testing site of this demonstration is the “Arterial Traffic Lab” on El Camino Real, California, which has been set up during the pervious transit signal priority projects at California PATH. Currently the loop traffic data and signal status data are being collected.

Since real-time data retrieving has been conducted by other projects, this task mainly involves real-time software development. The software needs to have the following functionalities:

- (1) To be run in QX 6.0 operating system;
- (2) To automatically read-in text type data files according to the order of time recorded in the frequency of 15 minutes which is the time interval for real-time data retrieving;
- (3) To automatically carry out data processing by calculating expected traffic parameters and output results to certain text format data file;
- (4) To display the calculated results on Website which involves two steps:
 - i. Use Java script to pick up data from data file (Text format) and put to internet website
 - ii. Refresh the website every 15 minutes

The main task of this chapter is depicted as the block on the right of in Figure 7.1, which is the overall picture of the monitoring system.



Physical Architecture of the Monitoring System

Figure 7.1 Functionality of the Software Part

7.2 Document Review

Performance measurement of traffic system is a large topic for traffic engineering. Since this task is focusing on some parameter calculation instead of discussing how to measure the performance, the previous results are directly used. However, several sources of methods are briefly reviewed here.

The main source of methods for performance parameter calculation is the Highway Capacity Manual [7-3] which provides most performance parameter definition and the ways for calculation. The work of [7-6] mainly uses stops and delay as performance measurement for both fixed-time and adaptively actuated control signals. [7-4] Average delay (Veh/s) is used in the evaluation of traffic control at maintenance & reconstruction zones. Travel time, speed and delay are used for arterial performance evaluation and the number of conflict point is used for safety related evaluation in [7-2]. [7-5] mentioned that although the data from loops such as: speed, vehicle counts, occupancy, surrogate of density are widely used for measurement, they are weak because of point measure characteristics. AVT (Advanced Vehicle Tracking) → OD calculation → Performance evaluation. Point measurement based evaluation → interval measurement based

evaluation. [7-1] pointed out that the most influential factor in the intersection performance for heavy flows is achieved by **reducing the number of phases** in the signal cycle. The CFI especially is finding increasing acceptance in the United States lately. Performance criteria for the intersection design include: *average delay time per vehicle, average stop time per vehicle, average number of stops per vehicle, average queue length, and maximum queue length*. A pedestrian volume of 75ped/hr is assumed on each approach (Eastbound, Southbound, Westbound, and Northbound) and as there are three possible directions in which each of these volumes can be assigned, the directional volumes are equal to 25ped/hr. (e.g. pedestrian trips generated at South approach is 75ped/hr, the volume of trips towards East, West, and North is 25ped/hr each). The walking speed of pedestrians is assumed to be 4ft/sec.

Terminologies

The following is a list terminologies used for the calculation and display.

- **Cycle:** A cycle is a complete sequence of intervals of all the signals in all directions of an intersection. In the example provided, the sequence would go from interval one to nine and then another cycle would start with interval one.
- **Cycle length:** A cycle length is the time it takes to complete one cycle. In the table this would be the sum of the interval times, or 65 seconds. The minimum time for a cycle length is generally 45 seconds, to limit the time lost starting and stopping traffic.

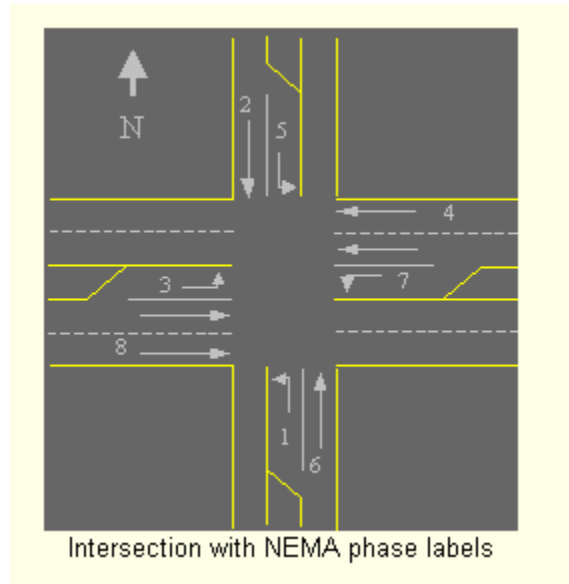


Figure 7.2 A Typical Intersection with 8 Movements

Interval	Time (sec)	NBLT	NBTH	SBLT	SBTH	WBLT	WBTH	EBLT	EBTH
1	12	R	R	R	R	G	R	G	R
2	4	R	R	R	R	Y	R	Y	R
3	1	R	R	R	R	R	R	R	R
4	25	R	R	R	R	R	G	R	G
5	4	R	R	R	R	R	Y	R	Y
6	1	R	R	R	R	R	R	R	R
7	14	G	G	G	G	R	R	R	R
8	3	Y	Y	Y	Y	R	R	R	R
9	1	R	R	R	R	R	R	R	R

Figure 7.3 Typical Phase in a Cycle for Traffic Signal

- Phase:** A phase is the part of the cycle assigned to a fixed set of traffic movements. When any of these movements change, the phase changes. The first phase in the above table is comprised of intervals one, two, and three. The third

interval is included even though no traffic movements have the right-of-way.

Characteristic: Each phase ends with **ALL RED** in all directions - Red Clearance Signal.

- **Yellow Change interval:** This is an interval in which yellow indications tell drivers in the phase with the right-of-way that their movement is about to lose its right-of-way. An example of this is interval five.
- **Red Clearance interval:** This describes the interval when all of the indications are red and is a safety measure designed to give the oncoming traffic enough time to clear the intersection before the next phase begins. An example of this is interval six.
- **Intergreen time:** This is the summation of the time allocated to the change and clearance intervals for a given phase (yellow and all red time).
- **Force-off:** Signal feature which, when asserted, shall cause termination of the current phase, provided that phase is in the extension portion. In no case shall assertion of force-off cause termination in a clearance interval or during a minimum Green for vehicles or pedestrians.
- **Max-out:** Phase end by reaching its maximum green time;
- **Gap-out:** When the vehicle approach arrive with longer headways than the extension interval; Inter-vehicle distance determined by loop signal.
- **Synch Phase:** Ends at the yield point if a call has been placed from the conflicting phases; If no call received, the controller will remain in the sync phase for the rest of the cycle;
- **Pedestrian call:** Signal given by pedestrian;
- **Off-set:** The difference between the zero points of Master Clock (MC) and Local Clock (LC): Usually, 4~6 intersections are coordinated with a MC and LC. All the MC and LCs are of the same cycle length which is fixed. For all green through, the off-set point of the LCs has a difference which is approximately the average travel time (related to intersection length). This is determined by the green plan. Next phase can start earlier but cannot be late as planned. Cycle length may be set different according traffic situation for Time-of-day.

- **Cycle length, speed and off-set** are the three parameters used for optimization; Cycle length are fixed for each intersection, which is not allowed to change. However, time interval for each phase within a cycle can change.
- **Signal for a movement:** Only 3 cases: **green, yellow and red;**
- Signal change is basically caused by either of the three triggers: Force-Off, Max-Out, or Gap-Out
- **Capacity:** throughput of the number of vehicle in hourly rate
- **Demand:** number of vehicle arriving
- **Volume:** number of vehicle discharging
- **FFS** – free-flow speed;
- **Running speed:** Intersection length divided by average speed also account for stop-time delay; (Traffic Manual 10-3); Posted speed limit is the default. Low volume (< 200 veh/h/ln) under all green condition.
- **Average travel speed:** The one we need to calculate; Performance index: LOS

7.3 List of Parameters for Performance Measurement:

- (1) **Average travel speed:** characteristic captured the effect of traffic control; Length of the segment (or entire street in consideration) divided by average travel time including stop-time delay;
- (2) **Total delay:** Travel time experienced – reference travel time
- (3) **Control Delay:** portion of total delay caused by signal operation; Including
 - i. Initial deceleration delay
 - ii. Stop-time Delay
 - iii. Queue move-up delay
 - iv. Final acceleration delay
- (4) **v/c ratio:** Approximate of overall efficiency of an intersection
- (5) **Average back of the queue:** The number of vehicles queued and not cleared (stop and wait);
- (6) **Arrive Type:** Quantitative measure of progression; 6 Levels determined by platoon length/density and arrive time: in which part of the read or green intervals.

(7) **Progression efficiency:** Green Arrival Ratio = # of vehicle arrive at green / Total number of vehicle within a cycle;

(8) **LOS:** A-F estimation over 15[min] period

Lane Division at an Intersection into lane groups (case sensitive):

Exclusive left turn;

Exclusive right turn;

All through lanes;

Mixed left (right) turn and through should be weighted according traffic flow;

7.4 Performance Parameter Calculation

The set of parameters used for evaluation of the performance of traffic control signals are listed and described briefly below.

The data file from the loops is updated for every 15 minutes from network real-time data retrieving system. For future uses the parameters are calculated and progressively-averaged over the number of phases passed or the number of cycles passed.

Typically, each intersection has 8 movements as shown in Figure 7.2. Corresponding to each movement, there may be zero or multiple lanes depending on the geometry of the intersection. For each lane or movement, loop sensor availability may have different situations:

- departure loop only
- arrival loop only
- both departure and arrival loop

The parameter computation needs to consider those differences.

All the 17 parameters are calculated for each movement of an intersection if applicable. Otherwise, zero value is assumed.

Due to progressive average, the calculated parameters can be output at any time point during the 15 minute period. However, three time points are chosen for output: 5, 10, 15 minute respectively at this stage. Thus the website will update at this rate, but it can update at any rate in principle in the future if necessary.

(1) **Traffic flow (each phase)** : number of vehicle discharged per hour, which is directly obtained from departure loop count.

(2) **Flow ratios (V/C, each phase)** :

$$vc_ratio = \frac{volume}{capacity}$$

which approximately represents the overall efficiency of an intersection

- **Capacity**: throughput of the number of vehicle in hourly rate; Actual value used: 500 vehicles per hour per lane
- **Volume**: number of vehicle discharged per hour

(3) **Saturation degree**: Calculated for each cycle and then progressively averaged over the number of cycles passed.

Saturation degree is calculated as follows:

$$Saturation_Degree = \frac{Volume * Cycle_Time}{Saturation_Flow * Green_Time}$$

$$Saturation_Flow = 1800 \text{ vehs / per_lane_per_hour}$$

$$Cycled_Time = 90[s]$$

(4) **Average Delay is calculated for three cases**. Each case is calculated for each cycle and then progressively averaged over the number of cycles passed.

Case 1: With both arrival and departure loop count:

$$\bar{T}_d = \frac{\sum_{All_Arrival} UTC - \sum_{All_Departure} UTC}{Number_of_Vehicle_Arrived}$$

Case 2: With only arrive loop counts

$$\bar{T}_d = \frac{\int_0^T f(t)dt - \int_{t_1}^T g(t)dt}{N}$$

$t = 0$: starting Red

$t = t$: starting green

T – clear up time instant after green ($t_1 \leq T$)

$f(t)$ – Arrival flow starting from Red

$g(t)$ – Saturation flow starting from green

N – Total vehicle count corresponding to the time point when cleaned.

Case 3: Only have departure count, no arrival count, using the Webster formula:

$$d_1 = \frac{0.5c \left(1 - \frac{g}{c}\right)^2}{1 - \left[\min(1, X) \frac{g}{c}\right]}$$

where d_1 = uniform control delay assuming uniform arrival (s/veh);

C = cycle length (s): cycle length used in pre-timed signal control, or average cycle length for actuated control;

g = effective green time for lane group (s); green time used in pre-timed signal control, or average lane group effective green time for actuated control;

X = v/c ratio or degree of saturation for lane group.

- (5) **Progression efficiency - Arrival ratio in green:** Calculated for each cycle and then progressively averaged over the number of cycles passed.

$$\text{Arrival ratio in green} = \text{Green_Arrival} / \text{All_Arrival}$$

- (6) **Percentage of gap out :** Number of gap out / Number of phases passed
Gap-out: When the vehicle approaching with longer headways than the extension interval; Inter-vehicle distance determined by loop signal.
- (7) **Percentage of max out:** Number of max out / Number of phases passed
Max-out: Phase end by reaching its maximum green time;
- (8) **Percentage of force off:** Number of force off / Number of phases passed
Force-off: Signal feature which, when asserted, shall cause termination of the current phase, provided that phase is in the extension portion. In no case shall assertion of force-off cause termination in a clearance interval or during a minimum Green for vehicles or pedestrians.
- (9) **Number of cycles which skipped this phase**
- (10) **Percentage of cycles which skipped this phase:** Number of cycles which skipped this phase divided by number of cycles passed
- (11) **Phase time durations**
- (12) **Number of phases with pedestrian interval**
- (13) **Percentage of phases with pedestrian interval:** Number of phases with pedestrian interval divided by the number of phases passed for that movement
- (14) **Number of preemption intervals**
- (15) **Percent of preemption intervals:** Number of preemption intervals divided by the number of phases passed for that movement

- (16) **Detector failure:** If loop detector data have not been updated for certain period of time, it is announced failure;
- (17) **Signal data failure :** If signal data has not been updated for certain period of time, it is announced failure. Here 5 minutes is used as a threshold.

7.5 Software Structure

The real-time software is developed in C code. The overall structure of the software is shown in Figure 7.4 and the C code structure is shown in Figure 7.5. It is noted that, for simplicity, the interface between the interface between the output of the C code (parameter calculation) and the Java code (for display the results on the Website) is text format data file instead of data base.

Overall Software Structure and Data Flow

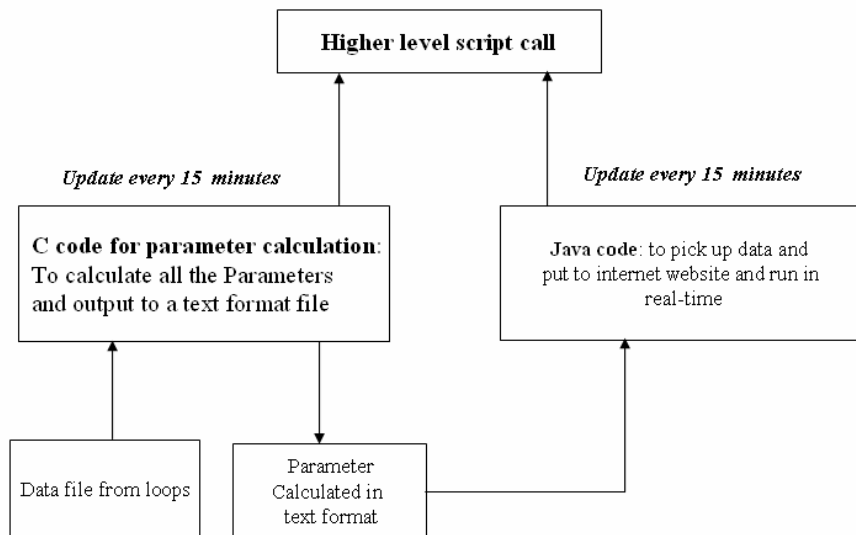


Figure 7.4 Overall Real-time Software Structure and Data Flow

C Code for Parameter Calculation

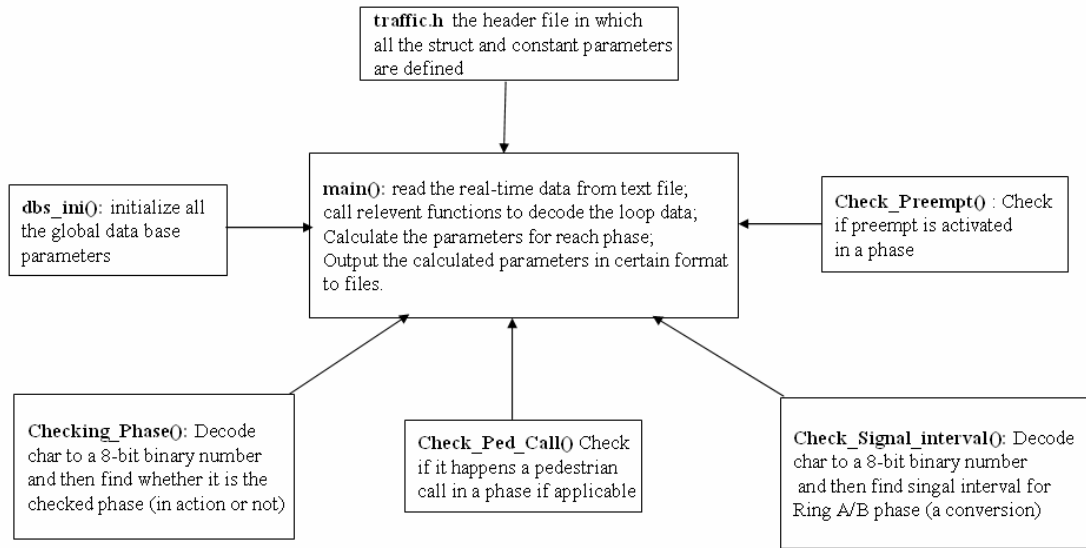


Figure 7.5 Structure of C Code and Data Flow

A sample of real-time display by Java code on the website is shown in Figure 7.6.

Intersection 1 (28th and El Camino)

UTC time	PST time	Phase	Preemption										
23:44:38	11:44:38	34	0										
Pedestrian	Local Cycle Clock	Master Cycle Clock	Pattern										
0	66	101	3										
Loop Detector Counts													
0	6	7	29	32	16	2	1	4	8	12	0	0	11

Figure 7.6 Website Real-time Display by Java code

7.6 References

- 7-1 Joe G. Bared, Praveen K. Edara and Ramanujan Jagannathan, Design and Operational Performance of Double Crossover Intersection and Diverging Diamond Interchange, TRB-05, Paper #05-1075
- 7-2 William L. Eisele and William E. Frawley, Estimating the Safety and Operational Impacts of Raised Medians and Driveway Density: Experiences from Texas and Oklahoma Case Studies, TRB-05, Paper #05-0719
- 7-3 Highway Capacity Manual, 2000
- 7-4 Ahmed Al-Kaisy & Eric Kerestes, Evaluation of the Effectiveness of Single-Lane Two-Way Traffic Control at Maintenance & Reconstruction Zones, TRB-05, Paper #05-1507
- 7-5 Cheol Oh and Stephen G. Ritchie, development of a methodology for designing advanced traffic surveillance systems toward origin-destination based traffic information, TRB-05, Paper #05-0998
- 7-6 Skabardonis, A., Fixed-time vs. actuated control in coordinated signal systems, TRB-05, Paper #05-2529

8 AN OFFSET REFINER FOR COORDINATED ACTUATED SIGNAL CONTROL SYSTEMS

8.1 Introduction

It has been a common practice to operate traffic-actuated controllers in coordinated systems to provide progression for major traffic movements along arterials and networks. Compared with fixed-time coordinated systems, these semi-actuated coordinated systems offer additional flexibility in responding to fluctuations in traffic demand. Under signal coordination, traffic actuated signals operate on a common background cycle length. Coordination is provided through an offset that defines the start of the local clock. The start of local clock, depending on the offset reference point can be set to the start of green, end of green or yield point for the sync phase(s). The rest of the phases are actuated and their duration varies between a minimum and a maximum green time. These phases may terminate at fixed force-off points in the background cycle or terminate early (gap-out). Typically, when the actuated phases terminate early, the spare green time in the cycle is received by the sync phase (Note that the NTCIP ASC specification allows the user to control the spare time going to either the next phase in the sequence or the sync phase).

To ensure operation efficiency of coordinated actuated systems, attention should be paid to determining appropriate signal settings, particularly offsets due to the fact that the start of green of the coordinated phases (typically Phases 2 and 6) are not fixed. Several approaches have been proposed in the literature to address such a so-called “early return to green” problem in the determination of offsets. Jovanis and Gregor [8-1] suggested adjusting the end of green of the sync phase to the end of the through-band for non-critical signals. Skabardonis [8-2] proposed three methods for determining offsets from the optimal fixed-time splits and offsets. Although the three methods differ in the procedure and applicable situation, the concepts are essentially the same: making a best estimate on average starting point of the sync phase and then optimizing the offset based on the estimate. Chang [8-3] offered a similar suggestion for obtaining the offsets from a

second optimization run that uses the *anticipated* green times on the noncoordinated phases, as constraints on their maximum green times.

The above prior studies have focused on determination of appropriate offsets in the stage of design of the signal timing plans. Certainly after implementing the timing plans in the field, there are still opportunities for fine-tuning. Shoup and Bullock [8-4] examined a concept of using the link travel times observed for the first vehicle in a platoon to adjust offsets. The concept could lead to an online offset refiner, if vehicle identification technologies had been deployed in arterial corridors. Abbas et al. [8-5] developed an online real-time offset transitioning algorithm that continually adjusts the offsets with the objective of providing smooth progression of a platoon through an intersection. More specifically, the objective was achieved by moving the green window so that more of the current occupancy actuation histogram is included in the new window. A greedy search approach was used to determine the optimal shift of the green window. In the ACS-Lite system developed by FHWA [8-6], a run-time refiner can modify in an incremental way the cycle, splits and offsets of the plan that is currently running based on observation of traffic conditions.

The advancement and deployment of telecommunication and ITS technologies make high-resolution signal operation data more readily available. This chapter presents the basic concept of an offline offset refiner for coordinated actuated control systems. The tool is not intended for taking the place of traditional signal optimization methodologies or software. Presumably the signal settings are optimized by using these methodologies or programs, probably following the guidelines set up in [8-2], [8-3] and [8-7]. The refiner is supposed to be used after implementing the resulting timings for a certain period of time (say, two weeks or longer). If the performance of the signal reported from the monitoring system developed in Chapter 7 is not satisfactory, the refiner can be called for refinement. Making use of a large amount of signal status data obtained from the field operation, it refines the offsets to provide smoother progression by re-maximizing the bandwidth and minimizing the red-meeting probability.

8.2 Premise of the Offset Refiner

The premise of the offset refiner is that day-to-day traffic and pedestrian flows are realization of uncertain traffic and pedestrian demands, which are certain types of stochastic processes. As a consequence, the resulting responses of actuated signals, such as phase durations or starts/ends of green of phases would follow certain types of stochastic distributions. These distributions can be estimated based on a large amount of signal status data from the field operation, thereby providing a better knowledge about the uncertainty of starts/ends of green for coordinated phases. The refiner will then apply the knowledge to adjust the offsets to improve the coordination. In real-world implementation, the refiner may be applied periodically to fine-tune the system until a satisfactory performance has been achieved. Note that the concept presented in this paper uses signal status data only. Our experience with field signal operation data reveals that signal status data are often far more accurate and robust than loop detector data (count and occupancy). The latter are prone to be inaccurate or missing due to inappropriate setting of loop sensitivity, loop malfunction, communication conflict, and weather condition etc. Certainly if high-quality loop data can be made available, the information would definitely help refine the system, as suggested later in this paper. Our ongoing research is looking into the opportunities.

To facilitate the presentation of the concept, we use the field data from a real-world network illustrated in Figure 8.1 (the figure is a snapshot of the simulation using Paramics produced by Quadstone, 2003 [8-8]). The network is a segment of El Camino Real, a major arterial in the San Francisco Bay Area. The study section is located in the city of Palo Alto, California, consisting of five signalized intersections whose cross streets are Churchill, Serra, Stanford, Cambridge and Page Mill from north to south respectively (there is another signalized intersection named as California between Cambridge and Page Mill where the field master is located. We were not able to obtain its signal status data since a special program was needed. We ignore this intersection in this paper, without impairing the validity of the concept presented herein.) The average travel times between intersections are also given in Figure 8.1. The signal status data are

pulled by the field master every two seconds and are stored in a roadside computer, and are further retrieved regularly via a dial-up connection. The data used in this paper were collected from 11:00 am - 3:00 pm, during the period of February 6, through March 7, 2005, for a total of 14 weekdays with more than 1,600 cycles. Moreover, the corridor is in two-way coordination, and the signal settings were recently optimized by using the Synchro software [8-9]. The signal settings for the time period of analysis are given in Table 8.1.

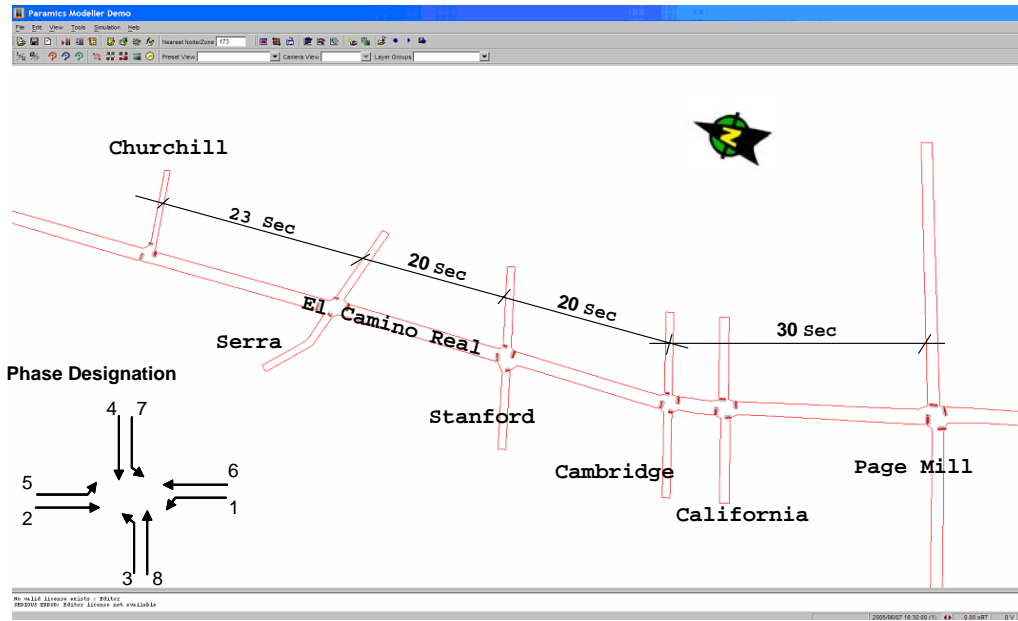


Figure 8.1 Study Corridor, El Camino Real, Palo Alto, CA

Table 8.1 Signal Settings for Each Intersection

Intersection	Cycle Length (sec)	Force-off								Offset (sec)
		1	2	3	4	5	6	7	8	
Churchill	120	N/A	0	33	N/A	63	0	N/A	33	84
Serra	120	59	0	N/A	30	51	0	N/A	N/A	72
Stanford	120	81	25	N/A	58	25	0	N/A	N/A	62
Cambridge	120	21	0	N/A	57	82	21	N/A	57	1
Page Mill	120	106	19	43	83	19	0	43	83	108

The premise of the offset refiner can be justified by examining the signal status data. Figure 8.2 presents the empirical cumulative distribution functions for starting points of Phase 2 at the El Camino/Page Mill intersection. Visually, it can be observed that the more days of data we used, the closer to each other the resulting distributions would be, which suggests that the empirical distributions tend to be stable.

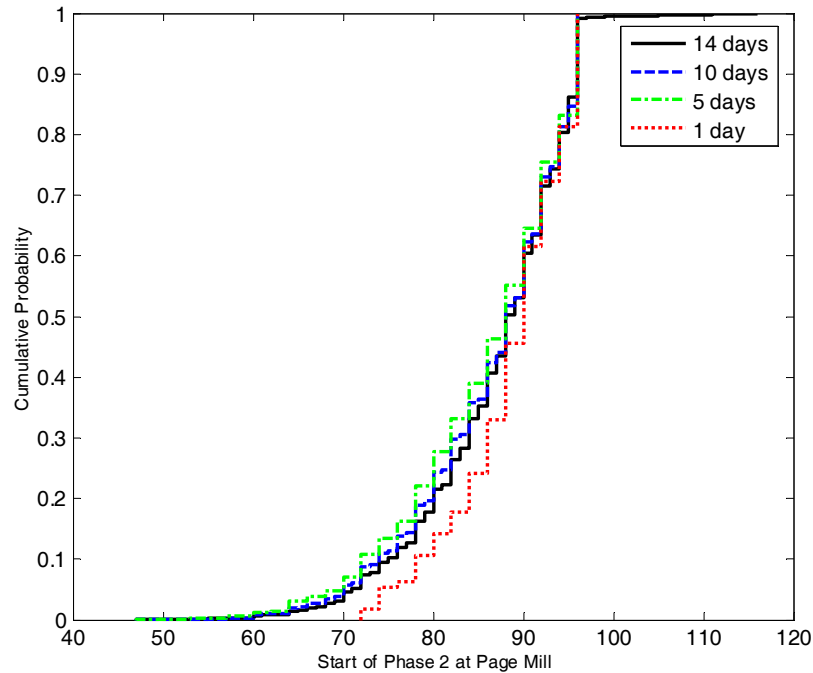


Figure 8.2 Empirical Cumulative Distributions for Start of Green of Phase 2 at Page Mill

We performed a Kolmogorov-Smirnov test to examine the null hypothesis that 10 days of data and 14 days of data have the same distribution. The observed Kolmogorov-Smirnov statistic is 0.033 and p value is 0.4711. Clearly the difference between their distributions is not significant at the 5% level, and thus we cannot reject the null hypothesis. The statistical test verifies the above observation that the empirical distributions would eventually become stable, and 14 days of data may be sufficient to estimate the distribution. Be aware of that the conclusion is only valid for this specific site and the

time period (11:00 am – 3:00 pm), and separate tests would be necessary for other implementation sites or different times of day.

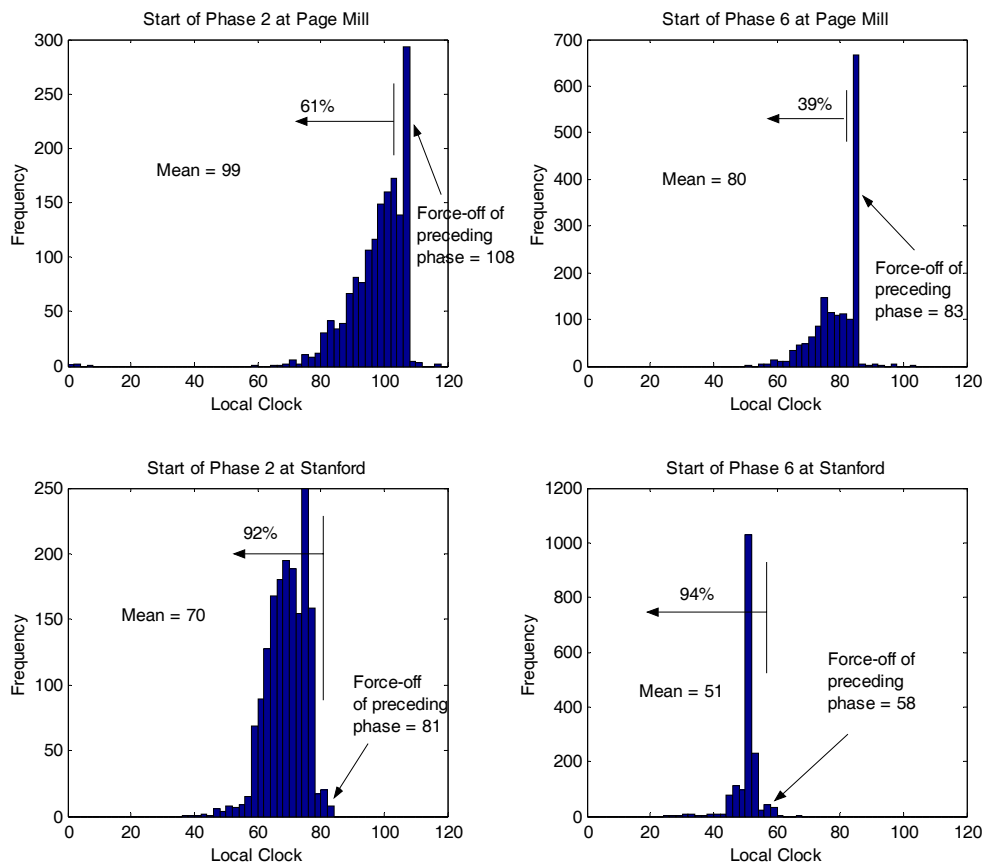


Figure 8.3 Probability of Early-Return-to-Green at Selected Intersections

To illustrate how prevailing the problem of “early return to green” is, Figure 8.3 depicts the histograms for starts of green of Phase 2 and 6 at two selected intersections along El Camino: Page Mill and Stanford. Page Mill is a critical intersection for the corridor with almost equal amounts of mainline and cross-street traffic. Still, the probabilities of “early return to green” are 61% for Phase 2 and 39% for Phase 6. Stanford has low volume of minor-phase traffic, thus the probabilities are as pretty high as 92% for Phase 2 and 94% for Phase 6 respectively. The histograms confirm the assertion made in the previous studies that the problem of “early return to green” should be recognized and explicitly addressed in the timing of coordinated actuated control.

Note that in addition to uncertainty of start of green, end of green is also uncertain, especially under lead-lag phase sequence, due to skip or gap-out of the left-turn phase. Figure 8.4 presents the histograms for green terminations of Phase 2 and 6 at Page Mill and Stanford. It can be seen that compared with starts of green, terminations of green have much narrower spans. Under many circumstances, the termination is the force-off point. In addition, the starts of Phase 2 and 6 are typically affected by traffic actuations on the cross streets, while the terminations of green often result from traffic actuations on main-street left-turns. Since traffic flows of these two streams can be viewed independent, the start and termination of green are likely independent, and thus can be treated separately.

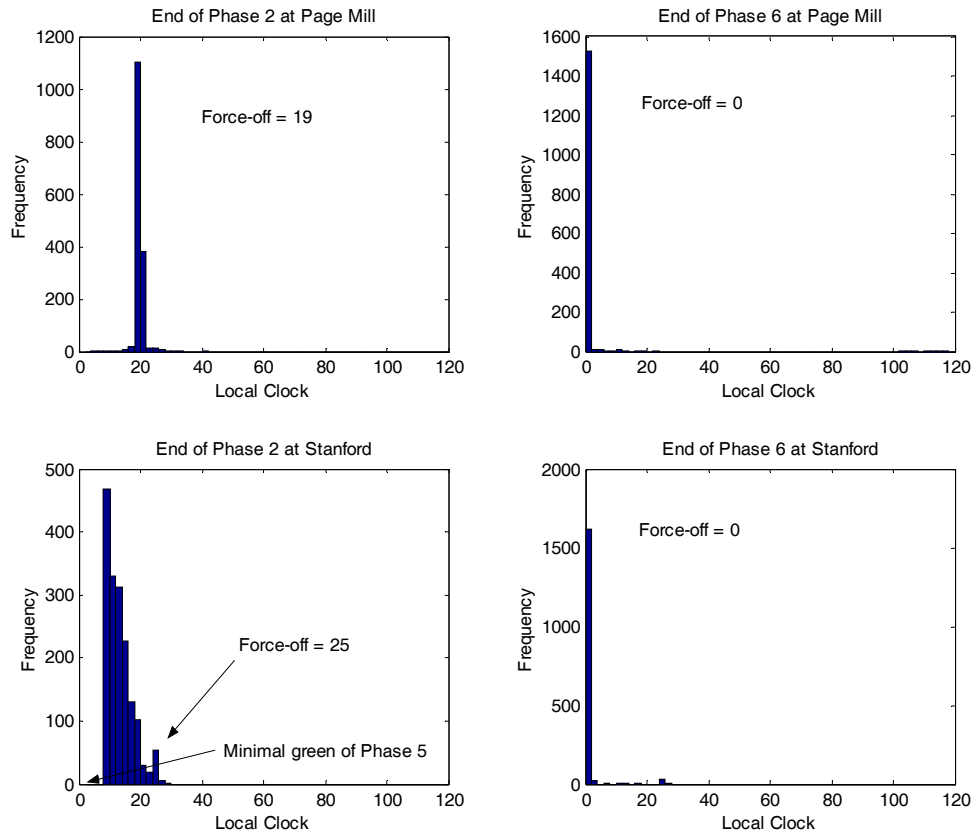


Figure 8.4 Uncertain Termination of Green at Selected Intersections

8.3 Maximization of Expected bandwidth

Maximizing bandwidth is often one of the objectives in determination of optimal offsets, especially for two-way coordination. Therefore the refiner aims to marginally adjust offsets to maximize the bandwidth or the sum of the bandwidths in the two arterial directions.

Since the start and end of major green are random variables, the bandwidth will also be random. Therefore, the objective is then to maximize the expected bandwidth. Before proceeding to discuss the bandwidth maximization, we conduct a coordinate transformation in order to facilitate the calculation. The coordinate transformation is to simply shift forward/backward the local clocks (coordinates) of the downstream/upstream intersections by a “distance” of the corresponding average travel time from the reference intersection. After the transformation, the vehicle trajectories will become vertical [8-10]. Consequently, the expected one-way bandwidth can be calculated as below:

$$W = E(\min(e_1, \dots, e_i, \dots, e_n)) - E(\max(s_1, \dots, s_i, \dots, s_n)) \quad (8-1)$$

Where:

W = the expected one-way bandwidth

$E(\bullet)$ = the expected value

e_i = the end of green in the transformed coordinate at intersection i

s_i = the start of green in the transformed coordinate at intersection i

The start and end of green, s_i and e_i , are discrete and independent random variables, whose distributions are estimated from the empirical data. However, analytical derivation of the expected bandwidth using Equation (8-1) based on the estimated distributions is quite tedious. In view of that the minimum/maximum operation is a convex/concave function, we have:

$$\begin{aligned} W &= E(\min(e_1, \dots, e_i, \dots, e_n)) - E(\max(s_1, \dots, s_i, \dots, s_n)) \\ &\geq \min(E(e_1), \dots, E(e_i), \dots, E(e_n)) - \max(E(s_1), \dots, E(s_i), \dots, E(s_n)) = \hat{W} \end{aligned} \quad (8-2)$$

The interpretation of Equation (8-2) is straightforward: the expected values of starts and ends of green may be used to roughly estimate the bandwidth, and the estimator \hat{W} is actually a lower bound of the expected bandwidth. The tightness of this lower bound depends on the difference among the distributions of start/end of green of each intersection.

It is easy to see that the variables s_i and e_i are function of the offset o_i , and thus the bandwidth W (\hat{W}) are affected by a set of offsets $\{o_i\}$. Changing offsets is like doing another coordinate transformation to shift ends and starts of arterial green, therefore changing the value of the bandwidth. The objective here is to determine a set of offsets to maximize the one-way bandwidth or the sum of the bandwidths in the two arterial directions, which is a constrained optimization problem whose constraints are $-C \leq o_i \leq C, i = 1, 2, \dots, n$, where C is the cycle length. This optimization problem can be readily solved by a suite of efficient algorithms.

We demonstrate the concept in the El Camino Real study corridor. A sequential quadratic programming (SQP) subroutine with finite-differencing derivatives in Matlab was adopted to solve for the optimal offsets for maximum two-way bandwidth. The SQP algorithm is one of the most efficient solution approaches for constrained nonlinear programming problems. It mimics Newton's method for unconstrained optimization in that it finds a step away from the current point by minimizing a quadratic model of the problem. Applying the refiner, we obtained the optimal marginal offsets (in addition to the current offsets) as $\{5, 0, 0, 0, 4\}$ seconds respectively for the intersections from north to south. The resulting through-bands are shown in Figure 8.5 and 8.6. The refiner increased the expected two-way bandwidth from 72 seconds to 76 seconds, and all of the improvements were obtained from Phase 2. The reason for achieving such a limited improvement (5%) is that the corridor is constrained by one critical intersection Page Mill whose green intervals of Phase 2 and 6 are very small compared with those of other intersections, and thus the largest bandwidth we can possibly obtain is the sum of these two green intervals. If the corridor had multiple critical intersections where the situation

is more complicated and the tradeoff is trickier, it is expected that the resulting improvement would be more significant.

Phase 2

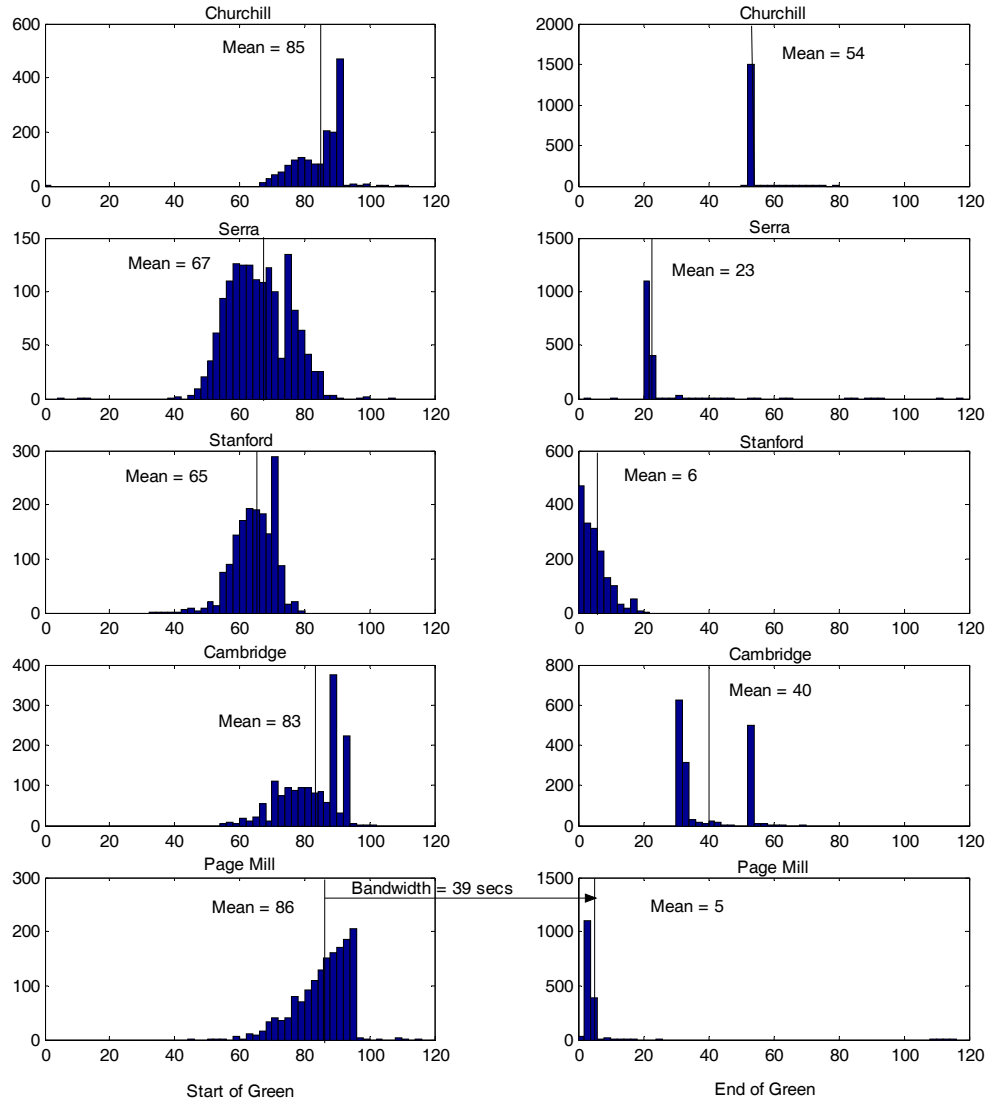


Figure 8.5 Optimized Through-Band for Phase 2—Two Way Coordination

Phase 6

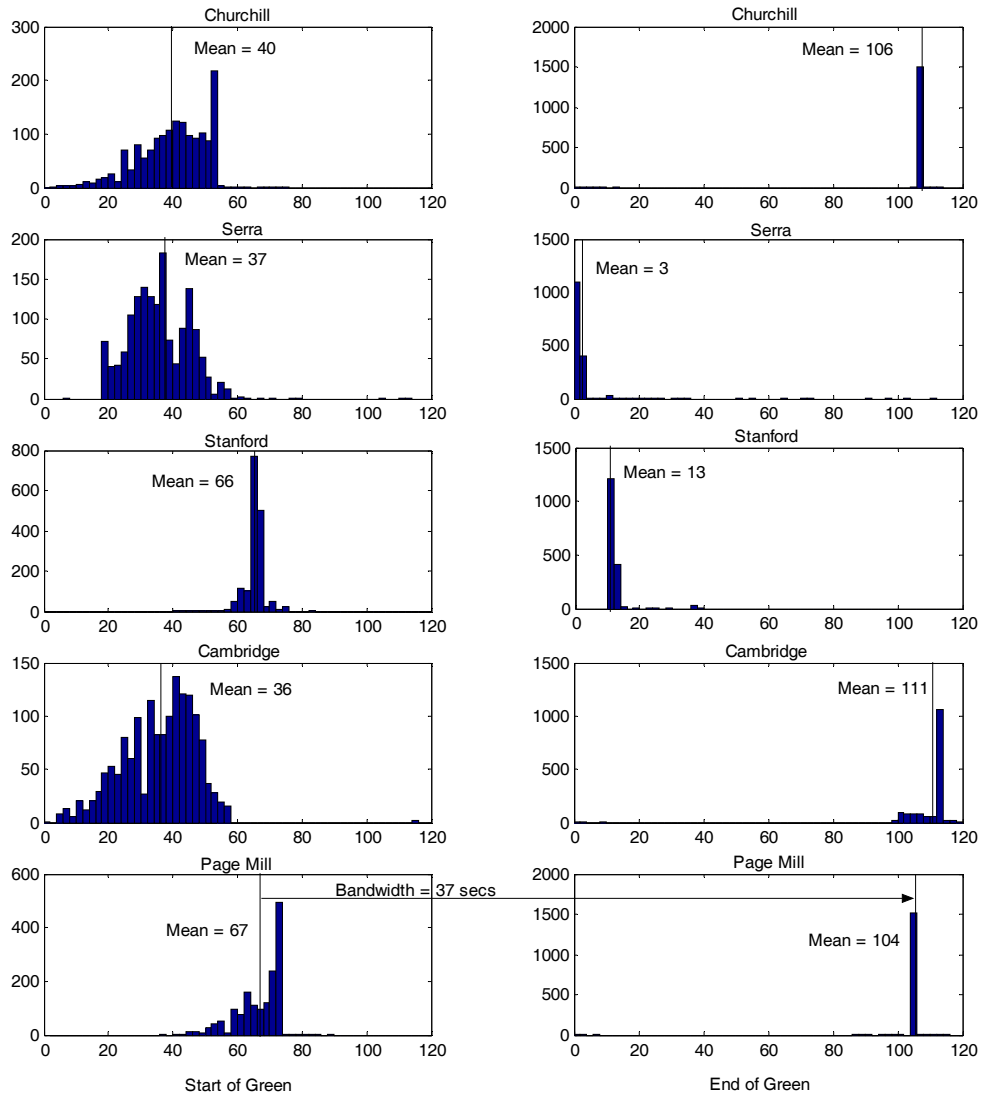


Figure 8.6 Optimized Through-Band for Phase 6—Two Way Coordination

Note that the optimal set of offsets is not unique. For example, the set of marginal offsets of $\{2.5, 0, -3.5, 0, 1\}$ is another optimal solution. If the loop detector data can be made available, the count and occupancy information could be used to choose the set of offsets that also minimizes the total delay of traffic.

In summary, the procedure described above is to first obtain the empirical signal status data to estimate the average starts/ends of green, and then use an optimization module or existing bandwidth programs to further optimize the offsets. The concept is exactly the same as what Skabardonis [8-2] and Chang [8-3] have previously suggested, but this paper further notes that:

- A large amount of empirical signal data is able to provide a more realistic estimate of starts/ends of green for the coordinated phases;
- The expected values of starts/ends of green may be used as reference points to optimize the bandwidth. The resulting bandwidth is the lower bound of the actual expected bandwidth.

8.4 Minimization of Red-Meeting Probability

Maximum bandwidth does not necessarily provide a better progression perceived by drivers. It has been pointed out that the two-way bandwidth maximization in no way guarantees perceived driver progression even for fixed-time signal controls [8-11]. However, it remains feasible to change the offsets to reduce the probability of a vehicle departing from an intersection to stop at another downstream intersection. As aforementioned, Abbas et al. [8-5] and Luyanda et al. [8-6] have used cyclic platoon pattern from detector data to dynamically adjust offsets for a smoother progression.

For one-way coordination, it has been known for long that the last vehicle in a platoon should be guaranteed to clear all intersections in the coordination such that the total traffic delay can be minimized [8-10]. However, this way the lead vehicles released from one intersection may have to stop on red at downstream intersections, giving drivers a perception of bad progression. In view of this, traffic engineers still tend to do the coordination for the lead vehicle, more specifically with reference to starts of green, even confronted with the problem of “early return to green”.

Once the reference points are determined, one-way coordination is generally straightforward and the bandwidth is the minimal green interval along the corridor.

However, the uncertainty of starts of green in actuated signal makes the task tricky, because the vehicles released earlier than those reference points may stop at the red again at the downstream signals. One way to mitigate or even eliminate this adverse effect is to set the offsets so that the earliest starting point of green at the first intersection is later than the latest starting point of green at any of downstream intersections (all these points are in the transformed coordinates). However, in this case the effective bandwidth could be significantly reduced. Therefore, there are actually two objectives among others in designing one-way or two-way progression for actuated signals. One is to provide enough bandwidth, and the other is to make sure early-released vehicles not to meet red lights at the downstream intersections. These two objectives are conflicting to a certain extent, and both need to be addressed in offset settings.

To represent the second objective, we define a new metric as the probability of the lead vehicle from the first intersection in coordination to meet a red light at any of the downstream intersections. Note that if the information of average queue length can be made available, the metric could be the lead vehicle from the first intersection in coordination not to *stop* at any of the downstream intersections. The calculation of the probability would be the same as that presented in the paper, except adding additional queue clearance times to the starts of green. Let $P_{i \rightarrow i+1}^t$ denote the probability for a vehicle departing at time t from intersection i to meet a red light at the next intersection $i+1$ or any of further downstream intersections, we have:

$$P_{i \rightarrow i+1}^t = \Pr(s_{i+1} > t) + \Pr(s_{i+1} \leq t \leq e_{i+1}) \cdot P_{i+1 \rightarrow i+2}^t \quad (8-3)$$

Note that Equation (8-3) is recursive. It means that the probability $P_{i \rightarrow i+1}$ consists of two components: the first is the probability of the vehicle meeting a red light at intersection $i+1$, and the second is for the case that the vehicle passes through intersection $i+1$ with a green light but meets a red light at a further downstream intersection; the probability for the occurrence of this case is the probability of that vehicle meeting a green light at intersection $i+1$ times the probability of meeting a red light at intersection $i+2$ or any

further downstream intersections. The concept is illustrated by a time-space diagram in Figure 8.7.

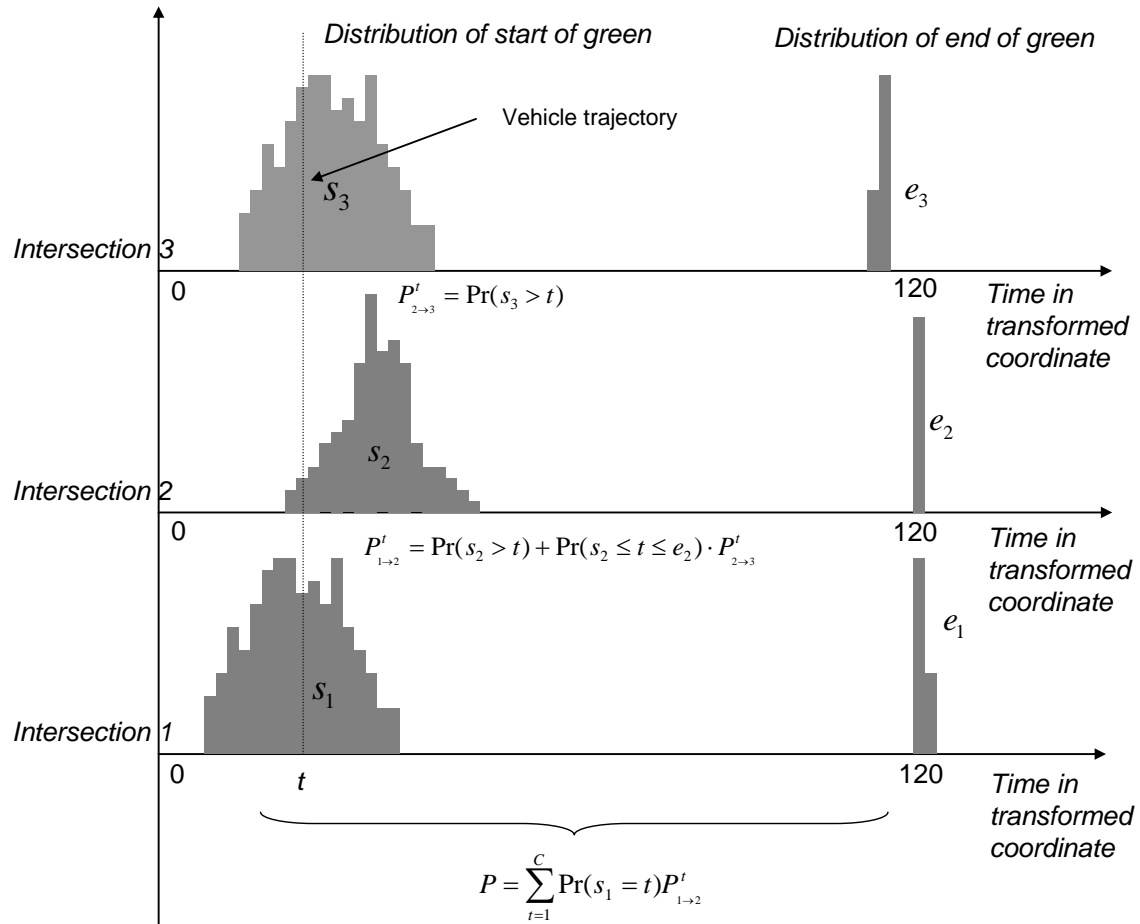


Figure 8.7 Red-Meeting Probability

And the boundary condition for the last intersection is:

$$P^t_{n-1 \rightarrow n} = \Pr(s_n > t) \quad (8-4)$$

which implies that the probability for a vehicle departing at time t from intersection $n-1$ to meet a red light at the last intersection is equal to the probability that the start of green is greater than the time t .

Consequently, summing up the probability for all discrete departure times t from the beginning of the cycle to the end of the cycle yields the probability for the first vehicle

released from intersection 1 to meet a red light at any of the downstream intersections. Therefore the new metric defined can be mathematically expressed as:

$$P = \sum_{t=1}^C \Pr(s_1 = t) P_{1 \rightarrow 2}^t \quad (8-5)$$

To illustrate the proposed metric, we applied it to Phase 6 of the study corridor. The full mathematical expression of the probability for the corridor is:

$$P = \sum_{t=1}^C \Pr(s_{\text{Page Mill}} = t) \cdot [\Pr(s_{\text{Cambridge}} > t) + \Pr(s_{\text{Cambridge}} \leq t \leq e_{\text{Cambridge}}) \cdot [\Pr(s_{\text{Stanford}} > t) + \Pr(s_{\text{Stanford}} \leq t \leq e_{\text{Stanford}}) \cdot [\Pr(s_{\text{serra}} > t) + \Pr(s_{\text{serra}} \leq t \leq e_{\text{Serra}}) \cdot \Pr(s_{\text{Churchill}} > t)]]]] \quad (8-6)$$

With the current setting of offsets, the probability for the first vehicle departing from Page Mill to meet a red light at any of the four downstream signals is 21.8% while the expected bandwidth for Phase 6 is 37 seconds. The probability is quite high. However, had the one-way coordination been made with reference to the expected values of starts of green for those five intersections, the probability would be 83.4% and the bandwidth remain the same. On the other extreme, if the sole objective is to minimize the red-meeting probability, then the set of marginal offsets of $\{0, 0, 6, 0, -26\}$ provides a minimum value of the probability of 0.3%, but the bandwidth is significantly reduced to 14 seconds. This is because to minimize the probability, the earliest starting point of green at the first intersection could be moved to be time point later than the latest starting point of green at any of downstream intersections, which obviously reduces the effective bandwidth.

Clearly we now have a bi-objective optimization problem whose two objectives are conflicting with each other. We thus are not able to find an unambiguous optimal solution. Rather, we will seek Pareto optimal or nondominated solutions, which are optimal in the sense that no improvement can be achieved in any objective without degradation in others. Several methods, such as the weighted sum method and the ε -constraint method,

have been proposed in the literature to solve the multi-objective optimization problem for the Pareto optimal solutions.

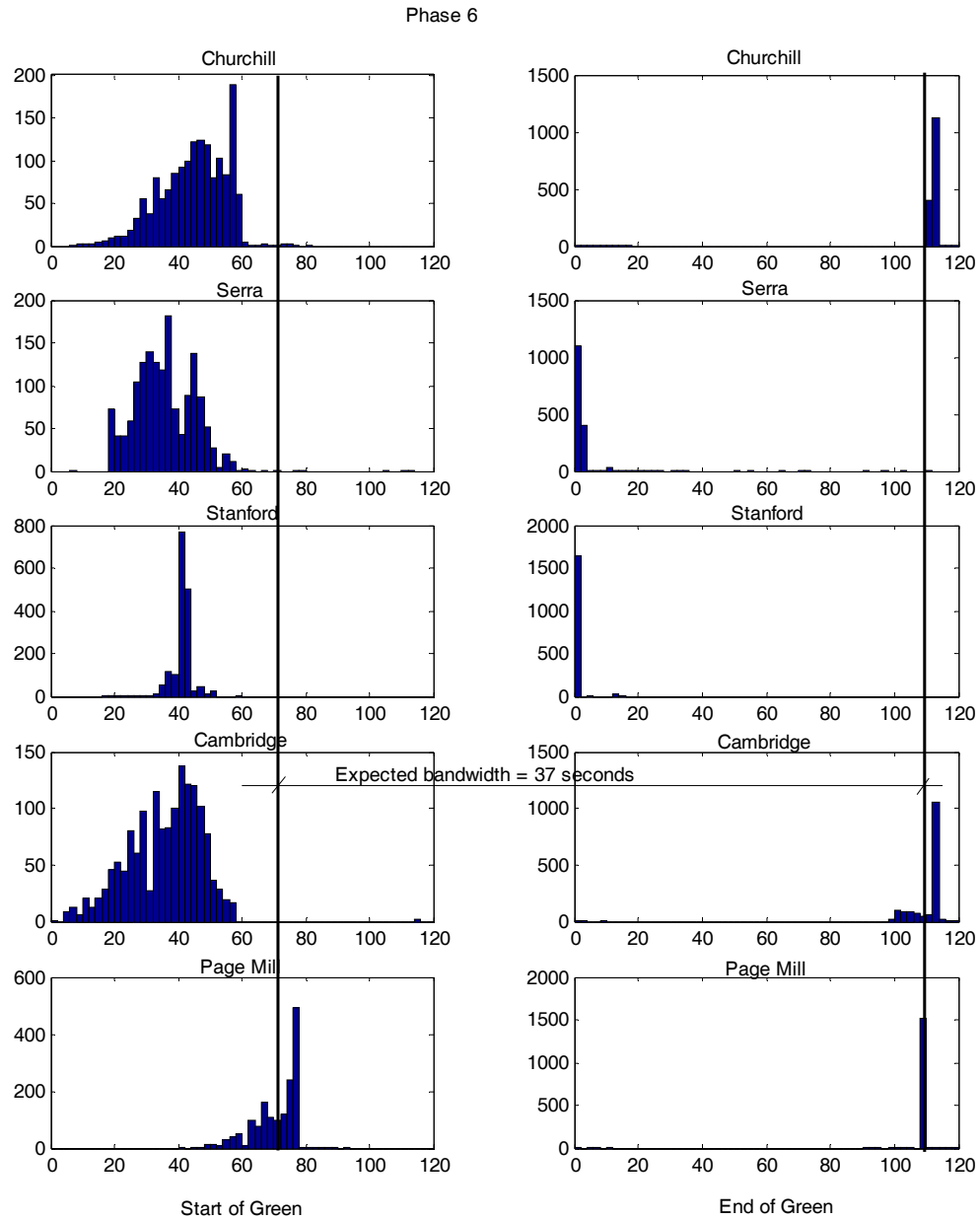


Figure 8.8 Optimized Through-Band for Phase 6-One Way Coordination

A subroutine of the goal attainment method for multi-objective optimization in Matlab was adopted for the case study. Applying the refiner to the example corridor, we obtained a Pareto optimal marginal offsets as $\{0, 0, 24, 0, 0\}$ seconds respectively, which result in a

probability of 2.3% and a bandwidth of 37 seconds. The resulting through-band is shown in Figure 8.8.

The solution of $\{0, 0, 6, 0, -26\}$ with a resulting probability of 0.3% and a bandwidth of 14 seconds and that of $\{0, 0, 24, 0, 0\}$ with a probability of 2.3% and a bandwidth of 37 seconds are both Pareto optimal and non-dominated. Since the difference of the red-meeting probability is quite trivial, it is expected that the latter solution will be favored by the practitioners. This offset setting successfully reduces the red-meeting probability from 21.8% to 2.3% with the bandwidth unchanged. Again, if a corridor has multiple critical intersections, the trade-off between the two objective functions would become more complex and profound. In this case, it would be necessary to produce a set of Pareto optimal solutions that forms an efficient frontier for traffic engineers to make the final selection based upon their preferences over these two objectives.

Note that the refiner presented in this paper uses signal status data only in view of inaccuracy or frequent loss of loop detector data that we have experienced. Consequently, the refiner is concerned with bandwidth maximization and red-meeting probability minimization, without accounting for delays or travel times. However, since the refinement is often marginal, we expect the impacts on these measures of performances will be also marginal. To verify this, we conducted a simulation study using Paramics to evaluate each of the following scenarios: baseline scenario with current offsets and the two scenarios with refined offsets for two-way coordination and one-way coordination respectively. Vehicle intersection delay in the simulation was defined as the travel time for a vehicle departing from an arrival loop detector to a departure loop detector minus its free-flow travel time. During the simulation period, all vehicles moving through intersections were traced by a tool developed through Paramics Application Programming Interface. Table 8.2 compares resulting average corridor delays for the three scenarios. To test the significance of the changes in delay, two-sided t -tests were performed. The null hypothesis is that the means of corridor delays with refined offsets are different from those in the baseline scenario. Given the size of population, the critical value with 95% confidence level is ± 1.68 . As shown in Table 8.2, all of the t -statistics

are smaller than 1.68. Therefore, both null hypotheses are rejected, implying that changes in delay incurred by the refiner are statistically insignificant. In summary, the simulation study has verified our expectation that the refiner may impose statistically insignificant impact on travel time or delay.

Table 8.2 Simulation Analysis of Traffic Corridor Delay

		Baseline		After		Change	<i>t-stat</i>
		Mean (sec)	SD (sec)	Mean (sec)	SD (sec)	Mean (sec)	
Two-Way	Major	37.805	27.864	39.348	28.440	1.544	1.385
	Minor	55.329	37.362	53.971	39.549	-1.358	-1.161
One-Way	Major (NB)	38.532	30.784	35.950	46.493	-2.581	-0.902
	Major (SB)	37.385	25.888	35.087	26.943	-2.298	-1.630
	Minor	55.329	37.362	54.029	36.943	-1.300	-1.150

Note: the delay for major phases is the average corridor delay while the delay for minor phases is the average delay of each intersection.

8.5 Concluding Remarks

We have presented the basic concept of an offline offset refiner, which attempts to address the problem of uncertain starts/ends of green in determination of offsets for coordinated actuated signal control. It has been shown that a large amount of archived signal status data is able to provide a more realistic estimate of distributions of starts/ends of green of the coordinated phases. The refiner will take advantage of this knowledge to fine-tune the implemented offsets in the field to provide smoother progression.

We have discussed two objectives in determination of offsets for coordinated actuated signal control. The first one is maximization of expected bandwidth. For the purpose, it is

acceptable to use means of starts/ends of green to do the bandwidth optimization. We have shown that the estimated bandwidth is the lower bound of the expected bandwidth.

We have defined a new metric to represent the smoothness of the progression, the probability of the lead vehicle departing from the first intersection in the coordination to meet a red light at any of downstream intersections. Minimization of this probability can serve as one of the objectives in designing signal coordination plans in addition to maximization of the bandwidth. Because these two objectives are often conflicting with each other, a bi-objective optimization procedure has to be adopted to seek the Pareto optimal offsets.

The offset refiner presented in this paper readily works with the current signal control system, and is easy to implement. It can also serve as a stand-alone tool available at Traffic Management Center, built upon commercial optimization software or using self-programmed codes to solve the resulting optimization problems. The refiner could be run periodically or together with an online progression monitor. If the signal performance degrades, then the refiner can be called to fine-tune the offsets for better coordination.

8.6 References

- 8-1 Jovanis, P. P. and Gregor, J. A (1986). Coordination of actuated arterial traffic signal systems. *Journal of Transportation Engineering*, 112(4), 416-432.
- 8-2 Skabardonis, A. (1996) Determination of timings in signal systems with traffic-actuated controllers. *Transportation Research Record 1554*, TRB, National Research Council, Washington, D.C.,18-26.
- 8-3 Chang, E. C. P. (1996). Guidelines for actuated controllers in coordinated systems. *Transportation Research Record 1554*, TRB, National Research Council, Washington, D.C., 61-73.

- 8-4 Shoup, G. E., and Bullock, D. (1999). Dynamic offset tuning procedure using travel time data. *Transportation Research Record 1683*, TRB, National Research Council, Washington, D.C., 84-94.
- 8-5 Abbas, M., Bullock, D. and Head, L. (2001). Real-time offset transitioning algorithm for coordinating traffic signals. *Transportation Research Record 1748*, TRB, National Research Council, Washington, D.C., 26-39.
- 8-6 Luyanda, F., Gettman, D., Head, L., Shelby, S., Bullock, D. and Mirchandani, P. (2003). ACS-Lite Algorithmic Architecture: Applying Adaptive Control System Technology to Closed-Loop Traffic Signal Control Systems. Design Guidelines for Deploying Closed Loop Systems. *Transportation Research Record 1856*, TRB, National Research Council, Washington, D.C., 175-184.
- 8-7 Nichols, A. and Bullock, D. (2001) *Design Guidelines for Deploying Closed Loop Systems*. Final Report FHWA/IN/JTRP-2001/11, Indiana Department of Transportation & FHWA, U.S. Department of Transportation.
- 8-8 Quadstone Limited (2003) Quadstone Paramics V4.2 User Guide, Edinburgh, Scotland, UK.
- 8-9 Trafficware Corporation (2003). *Traffiware Synchro 6.0 User Guide*. Albany, California.
- 8-10 Newell, G. F. (1989). *Theory of Highway Traffic Signals*. UCB-ITS-CN-89-1, ISSN 0129-5911, University of California at Berkeley.
- 8-11 Wallace, C. E. and Courage, K.G. (1982). Arterial Progression – New Design Approach. *Transportation Research Record 881*, TRB, National Research Council, Washington, D.C., 53-59.

9 CONCLUSIONS AND NEXT STEPS

9.1 Conclusion

We have achieved two project objectives of integrating Paramics with Synchro and TRANSYT-7F, and developing a systematic approach to improve efficiency of closed-loop signal control systems based on archived traffic and signal status data.

Two sets of Paramics plug-ins have been developed to facilitate the two-way data conversion between Paramics and Synchro or TRANSYT-7F. The first set of plug-ins read Paramics network data and traffic volume data and generate Synchro or TRANSYT-7F data files while the second transfer optimized actuated signal timing data back to Paramics. These tools may be able to assist traffic engineers in optimizing, evaluating and refining signal timing plans for arterial traffic operations. Step-by-step tutorials are also included in the report to teach the user how to use the new plug-ins.

The systemic approach we have proposed includes three components: timing, monitoring, and plan adjustment. For the timing component, we have developed two innovative models, scenario-based and min-max, to determine robust optimal signal timings that are less sensitive to fluctuations of traffic flows at the same time minimizing the mean of the delays per vehicle across all realizations of uncertain traffic flows. We also have investigated how to make use of a large set of archived traffic data to produce TOD intervals for TOD controls. For the monitoring component, a prototype signal performance monitoring system has been developed to report performance measures of signal control operations and help traffic operation staffs make the decision whether a retiming or fine-tuning effort is needed or not. Finally, we have delivered a proof of concept of an offset refiner that fine tunes signal offsets to provide smoother progression in either one-way or two-way coordination. The offset refiner is easy to implement, and could be run periodically or together with the performance monitoring system. If the signal performance degrades, then the refiner can be called to fine-tune the offsets for better coordination.

9.2 Next Steps

Future research can be conducted in the following three directions:

- 1) Address the limitations of the developed integrated Paramics-Synchro /TRANSYT-7F optimization tool documented in Chapters 3 and 4, and enhance the tool to be applicable to coordinated controls for arterials and freeway-arterial corridors.
- 2) The integrated tool does not work directly with real-time “live” traffic data. A third product or self-developed codes is needed to estimate O-D tables from archived real-time traffic data and then feed these tables into the tool. Therefore, one of the immediate next steps will be to incorporate real-time "live" field traffic data into the integrated tool, which will further empower the integrated tool to replicate the reality more accurately and produce traffic signal control strategies in a near real-time manner.
- 3) This project has developed a systemic approach to make closed-loop control systems be more adaptive in both strategic and tactic levels. Although the approach has been verified via simulation studies, a field operational test is necessary to further examine and demonstrate its effectiveness, particularly the offset refiner presented in Chapter 8 that appears appealing to many practitioners. At the same time, the notion of robust timing approach presented in Chapter 5 can be further extended to designing robust coordination plans for coordinated actuated control systems.