INTELLIGENT DYNAMIC SIGNAL TIMING OPTIMIZATION PROGRAM

BY
ALI HAJBABAIE

DISSEPTION

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Civil Engineering in the Graduate College of the University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Professor Rahim F. Benekohal, Chair and Director of Research
Professor Sheldon H. Jacobson
Associate Professor Yanfeng Ouyang
Associate Professor Dusan M. Stipanovic
ABSTRACT

In this research the development of a signal timing optimization model for oversaturated urban traffic networks with stochastic driver behavior and vehicle arrival headway is presented. The model is called Intelligent Dynamic Signal Timing Optimization Program or IDSTOP. IDSTOP is formulated as a dynamic optimization problem whose objective is to maximize the number of weighted completed trips in the network (weighted by the length of the shortest route available for that trip). The model aims at managing transportation supply by optimizing signal timing parameters and simultaneously managing transportation demand by redirecting vehicles to less congested routes.

Solving IDSTOP is a very complicated task since it is a nonlinear optimization program with no closed form formulation for the objective function in terms of the decision variables; and has an extremely large decision space. Therefore, a meta-heuristic algorithm is developed. It creates a population of candidate solutions and improves their quality over different generations. To reduce the runtime, a heuristic method was developed to create feasible solutions for the first population. The feasibility of candidate solutions was first checked using a macroscopic approach. A microscopic approach was also used to check all the solutions that were marked feasible by the macroscopic approach. To account for stochastic driver behavior and vehicle arrival headway, several microscopic simulation replications were made. The fittest individual of each population was chosen for traffic assignment. Assigning traffic for the fittest individual not only significantly reduced the runtime, but also insured not using inefficient signal timing parameters.

IDSTOP solutions were compared to Direct-CORSIM solution using a realistic case study network and four demand patterns covering both undersaturated and oversaturated conditions for symmetric and asymmetric traffic demands. Findings indicated that IDSTOP solutions resulted in significantly more efficient network performance than Direct-CORSIM solutions. IDSTOP solutions increased the number of
completed trips by 2.0% to 19.6% and at the same time reduced average delay by 8.9% to 30.8% for different demand patterns in the case study network. These figures indicated significant improvement in the network performance.

Simple GA, Elitist simple GA, Micro-Elitist GA, self-adaptive ES, and Elitist self-adaptive ES (ES+) were used to solve IDSTOP. In general, ES+ outperformed the rest of algorithms in reaching most different levels of the upper-bounds. In addition, ES+ was very efficient in oversaturated conditions especially when demand was symmetric. Micro-Elitist GA was very quick in early improvements in the fitness value. However, in most of the cases it was outperformed by ES+ in reaching higher levels of fitness value except for asymmetric undersaturated conditions.

Using IDSTOP, Optimal Left Turn Management Program (OLTMP) was developed. OLTMP improves network performance by prohibiting the left turns at certain intersections of the network. Numerical findings indicated that OLTMP had great potential to improve network performance efficiency by optimizing the policies on the left turns. When left turn volume was low (up to 7.5% of the capacity of a lane), none of the left turns were prohibited since left-turners had enough opportunity to make their turning maneuver in permitted phases. When left turn volume was very high (20% of the capacity of a lane), none of the left turns were prohibited as well because doing so resulted in rerouting too many vehicles and overcrowding other intersections. However, for moderate left turn volumes (10% to 17.5% of the capacity of a lane) left turns were prohibited in one or two intersections of the network.

A method was proposed to determine the policy that resulted in a more efficient network performance among variable cycles and common cycle policies. Our findings in a case study network (symmetric oversaturated demand pattern) that was suitable for signal coordination indicated the variable cycle length strategy has great potential to improve network performance compared to common cycle strategy. The improvement is achieved by using more suitable signal timing parameters
for each intersection and only coordinating them when needed. In the case study, variable cycle lengths strategy reduced total delay by 7.5%, and improved the number of completed trips by 1.0% compared to common cycle length strategy. Therefore, using variable cycle lengths significantly improved network performance efficiency in symmetric oversaturated conditions.

IDSTOP was used to develop Optimal Network Metering Program (ONMP). ONMP improved network performance by metering traffic at entry points of the network. ONMP was formulated and a meta-heuristic algorithm was developed to solve it. The numerical findings showed that optimized metering strategy reduced total delay by 10.6% and total travel time by 6.7% compared to no metering strategy. Therefore, optimal metering has significantly improved network performance in the case study. In addition, optimized metering strategy reduced total delay by 4.5% and total travel time by 2.7% compared to the best uniform metering strategy. This indicated that ONMP solution significantly improved network performance compared to the best uniform metering strategy.
TO MY MOM, DAD, AND BROTHER
I would like to express my deepest gratitude to my advisor and mentor, Professor Rahim F. Benekohal, for his continuous guidance, support, and encouragement throughout my graduate studies at the University of Illinois at Urbana-Champaign. It would have not been possible to complete this dissertation without his help. His wisdom, deep knowledge of transportation operation, commitment to the highest standards, and devotion and enthusiasm to education and research, have been and will always be a source of inspiration in my career.

My thanks also go to Professor Sheldon H. Jacobson, Professor Yanfeng Ouyang, and Professor Dusan M. Stipanovic for their participation in my thesis committee. Their valuable comments had a significant impact on my work.

This research has greatly benefited from the courses I have taken from several departments at the University of Illinois. I would like to thank all the professors who have taught me during my graduate studies. I would also like to appreciate Carol Czajkowski, Mary Pearson, and Sheree Waltz for their invaluable help and support.

I greatly enjoyed the opportunity to work and be friend with my colleagues at UIUC. I always appreciate Varun Atluri, Kivanc Avrenli, Dr. Madhav Chitturi, Leila Hajibabai, Juan Medina, Hani Ramezani, Jarice Rodriguez, and Dr. Ming-heng Wang, for their contributions to my efforts. Special thanks to Leila Hajibabai, Juan Medina, and Hani Ramezani for their helps, long deliberations, useful comments, and their encouragement in the course of conducting this research.

None of this would have been possible without the support and love from my family. I would like to greatly appreciate my mom, dad, and brother, Pouran, Hossein, and Amin for their understanding, unconditional support, and always being there for me.
# TABLE OF CONTENTS

LIST OF FIGURES ................................................................................................. xi
LIST OF TABLES .................................................................................................... xii

CHAPTER 1 INTRODUCTION ...................................................................................... 1
  1.1 Problem Statement and Research Motivation .................................................. 1
  1.2 Research Objectives ......................................................................................... 4
  1.3 Research Tasks ................................................................................................ 5
  1.4 Scope and Contributions of Research .............................................................. 6
  1.5 Thesis Organization ......................................................................................... 7
  1.6 References ...................................................................................................... 8

CHAPTER 2 BACKGROUND ....................................................................................... 11
  2.1 Urban Traffic Control ...................................................................................... 11
    2.1.1 Fixed-Time Signal Timing ......................................................................... 11
    2.1.2 Real-Time Signal Timing .......................................................................... 14
    2.1.3 Adaptive Signal Control ........................................................................... 18
  2.2 Dynamic Traffic Assignment .......................................................................... 24
    2.2.1 Mathematical Programming ...................................................................... 24
    2.2.2 Optimal Control Formulation .................................................................... 27
    2.2.3 Variational Inequality ............................................................................... 27
    2.2.4 Simulation-Based Methods ....................................................................... 28
  2.3 Summary ......................................................................................................... 29
  2.4 References ...................................................................................................... 30

CHAPTER 3 IDSTOP FORMULATION ...................................................................... 36
  3.1 Introduction .................................................................................................... 36
  3.2 Decision Variables ......................................................................................... 37
  3.3 Objective Function ........................................................................................ 38
    3.3.1 Delay Minimization .................................................................................. 38
    3.3.2 Travel Time Minimization ........................................................................ 39
    3.3.3 Throughput-Minus-Queue Maximization ................................................... 40
    3.3.4 Trip Maximization .................................................................................... 40
    3.3.5 Weighted Trip Maximization .................................................................... 41
    3.3.6 Choosing the Objective Function .............................................................. 42
  3.4 Constraints .................................................................................................... 48
    3.4.1 Cycle Length Constraints .......................................................................... 49
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.4.2</td>
<td>Green Time Constraints</td>
<td>50</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Offset Constraints</td>
<td>55</td>
</tr>
<tr>
<td>3.4.4</td>
<td>Queue Length Constraints</td>
<td>56</td>
</tr>
<tr>
<td>3.4.5</td>
<td>Gridlock Constraints</td>
<td>56</td>
</tr>
<tr>
<td>3.4.6</td>
<td>De-Facto Red Constraints</td>
<td>57</td>
</tr>
<tr>
<td>3.4.7</td>
<td>Ideal Offset Constraints</td>
<td>58</td>
</tr>
<tr>
<td>3.4.8</td>
<td>Route Delay Constraints</td>
<td>59</td>
</tr>
<tr>
<td>3.4.9</td>
<td>System Optimum Dynamic Traffic Assignment Constraints</td>
<td>60</td>
</tr>
<tr>
<td>3.4.10</td>
<td>Summary of Formulation</td>
<td>62</td>
</tr>
<tr>
<td>3.5</td>
<td>References</td>
<td>64</td>
</tr>
<tr>
<td>4.1</td>
<td>IDSTOP SOLUTION TECHNIQUE</td>
<td>65</td>
</tr>
<tr>
<td>4.2</td>
<td>Introduction</td>
<td>65</td>
</tr>
<tr>
<td>4.3</td>
<td>Genetic Algorithms</td>
<td>67</td>
</tr>
<tr>
<td>4.4</td>
<td>Evolution Strategies (ES)</td>
<td>68</td>
</tr>
<tr>
<td>4.4.1</td>
<td>IDSTOP Architecture</td>
<td>70</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Signal Timing Optimization</td>
<td>77</td>
</tr>
<tr>
<td>4.4.2</td>
<td>System Optimum Traffic Assignment</td>
<td>81</td>
</tr>
<tr>
<td>4.5</td>
<td>Summary</td>
<td>83</td>
</tr>
<tr>
<td>4.6</td>
<td>References</td>
<td>84</td>
</tr>
<tr>
<td>5.1</td>
<td>IDSTOP IMPLEMENTATION AND PERFORMANCE</td>
<td>86</td>
</tr>
<tr>
<td>5.2</td>
<td>Introduction</td>
<td>86</td>
</tr>
<tr>
<td>5.3</td>
<td>Case Study Network</td>
<td>86</td>
</tr>
<tr>
<td>5.4</td>
<td>Demand Patterns</td>
<td>88</td>
</tr>
<tr>
<td>5.5</td>
<td>Coding IDSTOP</td>
<td>89</td>
</tr>
<tr>
<td>5.5.1</td>
<td>Handling the Constraints</td>
<td>93</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Type-I Constraints</td>
<td>93</td>
</tr>
<tr>
<td>5.5.2</td>
<td>Type-II Constraints</td>
<td>94</td>
</tr>
<tr>
<td>5.6</td>
<td>Generating the First Population</td>
<td>98</td>
</tr>
<tr>
<td>5.7</td>
<td>Fitness Function Evaluation</td>
<td>99</td>
</tr>
<tr>
<td>5.8</td>
<td>Using Cache</td>
<td>102</td>
</tr>
<tr>
<td>5.9</td>
<td>Obtaining an Upper Bound</td>
<td>102</td>
</tr>
<tr>
<td>5.9.1</td>
<td>Oversaturated Conditions</td>
<td>102</td>
</tr>
<tr>
<td>5.9.2</td>
<td>Undersaturated Conditions</td>
<td>106</td>
</tr>
<tr>
<td>5.10</td>
<td>Comparing Solutions</td>
<td>108</td>
</tr>
</tbody>
</table>
5.11 Numerical Findings .................................................................................. 110
  5.11.1 FTNTA-IDSTOP vs. Direct-CORSIM .................................................. 110
  5.11.2 RTNTA-IDSTOP vs. FTNTA-IDSTOP .............................................. 115
  5.11.3 RTNTA-IDSTOP vs. DIRECT-CORISM .......................................... 119
  5.11.4 IDSTOP vs. RTNTA-IDSTOP .......................................................... 119
  5.11.5 IDSTOP with time Variant Traffic Demand ........................................ 122
5.12 Summary .................................................................................................. 125

CHAPTER 6 EVOLUTIONARY ALGORITHMS ......................................................... 127
6.1 Introduction .............................................................................................. 127
6.2 Genetic Algorithm .................................................................................... 128
  6.2.1 Simple Genetic Algorithms ................................................................. 129
  6.2.2 Elitist GA ............................................................................................ 134
  6.2.3 Micro-Elitist GA ................................................................................ 135
6.3 Evolution strategies .................................................................................. 136
  6.3.1 Recombination .................................................................................... 138
  6.3.2 Mutation .............................................................................................. 139
  6.3.3 Selection .............................................................................................. 140
6.4 Efficiency of Different Algorithms .......................................................... 141
  6.4.1 SGA vs. ESGA ..................................................................................... 143
  6.4.2 ES vs. ES+ ........................................................................................... 144
  6.4.3 ES+ vs. ESGA ..................................................................................... 146
  6.4.4 ES+ vs. MEGA ................................................................................... 147
  6.4.5 Summary ............................................................................................. 148
6.5 References .................................................................................................. 148

CHAPTER 7 A PROGRAM FOR OPTIMAL LEFT TURN MANAGEMENT ............... 150
7.1 Introduction .............................................................................................. 150
7.2 Optimal Left Turn management Problem Formulation .............................. 152
7.3 Solving OLTMP ......................................................................................... 152
7.4 Case Study Network .................................................................................. 156
7.5 Numerical Findings ................................................................................... 157
7.6 Summary .................................................................................................... 163

CHAPTER 8 DYNAMIC INTELLIGENT SIGNAL COORDINATION IN OVERSATURATED TRANSPORTATION NETWORKS ............................................. 164
8.1 Introduction .............................................................................................. 164
### LIST OF FIGURES

| Figure 3.1 | Phases in an intersection | ................................................................. | 37 |
| Figure 3.2 | Different cases for statistical analysis | ......................................................................... | 43 |
| Figure 3.3 | Methodology of choosing IDSTOP objective function | ................................................................... | 44 |
| Figure 3.4 | Network PM for each demand pattern and each objective function | ........................................ | 45 |
| Figure 3.5 | A coordinated arterial | ................................................................... | 51 |
| Figure 3.6 | Sensitivity analysis on $\delta$ | .................................................................. | 53 |
| Figure 3.7 | Gridlock avoidance | .................................................................. | 57 |
| Figure 4.1 | Schematic IDSTOP solution technique | .................................................................. | 76 |
| Figure 4.2 | IDSTOP phase sequence | .................................................................. | 79 |
| Figure 5.1 | Modified Springfield network | .................................................................. | 87 |
| Figure 5.2 | Demand changes over a study period of 60 minutes | ........................................ | 89 |
| Figure 5.3 | Schematic optimization process | .................................................................. | 92 |
| Figure 5.4 | Number of randomly created individuals to obtain 450 feasible solutions for different generations | ........................................... | 98 |
| Figure 5.5 | Finding the number of fitness function evaluations throughout the optimization | ........................................ | 101 |
| Figure 5.6 | Schematic urban traffic network | .................................................................. | 104 |
| Figure 5.7 | Distribution of average saturated headway at exit intersections of the network | ........................................ | 105 |
| Figure 5.8 | Traffic demand over time in modified Springfield network | ........................................ | 123 |
| Figure 5.9 | Entry traffic demand and network-wide average queue length for modified Springfield network | ........................................ | 124 |
| Figure 5.10 | Number of completed trips, average cycle length, average delay and travel time for modified downtown Springfield network over time | ........................................ | 125 |
| Figure 6.1 | Binary representation of decision variables in GA | .................................................. | 129 |
| Figure 6.2 | Truncation selection | .................................................................. | 131 |
| Figure 6.3 | Schematic single point crossover | .................................................................. | 133 |
| Figure 6.4 | Schematic two point crossover | .................................................................. | 133 |
| Figure 6.5 | Discrete recombination | .................................................................. | 139 |
| Figure 7.1 | Starvation for east-bound through | .................................................................. | 150 |
| Figure 7.2 | Solution process for solving OLTMP | .................................................................. | 155 |
| Figure 7.3 | Modified Springfield network | .................................................................. | 157 |
| Figure 7.4 | Number of intersections with prohibited left turns vs. left turn percentage | ........................................ | 162 |
| Figure 8.1 | The effects of common and variable cycles on the offsets | ........................................ | 164 |
| Figure 8.2 | Effects of using a common cycle in an arterial on queue length | ........................................ | 165 |
| Figure 8.3 | Schematic methodology to find the optimal strategy | ........................................ | 169 |
| Figure 8.4 | Case study network | .................................................................. | 170 |
| Figure 9.1 | ONMP solution technique | .................................................................. | 180 |
| Figure 9.2 | Case study network with gating signals | .................................................................. | 182 |
| Figure 9.3 | Network performance under different metering levels | ........................................ | 186 |
LIST OF TABLES

Table 3.1 Performance Measures for Different Objective Functions under Different Demand Patterns.....47
Table 5.1 PM for FTNTA-IDSTOP vs. Direct-CORSIM ..................................................................113
Table 5.2. PM for RTNTA-IDSTOP vs. FTNTA-IDSTOP ..................................................................118
Table 5.3. PM for IDSTOP vs. RTNTA-IDSTOP ..............................................................................122
Table 6.1. Number of Fitness Function Evaluations Required to Reach a Certain Level of the Theoretical
Upper Bound. ........................................................................................................................................145
Table 7.1. PM for OLTMP and LT Allowed Policy..............................................................................160
Table 8.1. Performance Measures for Variable Cycle Length (IDSTOP) and Common Cycle (CC-IDSTOP)
Strategies ..............................................................................................................................................172
Table 9.1 PM for Different Uniform Metering Levels and Optimized Metering Strategy...............184
CHAPTER 1

INTRODUCTION

1.1 Problem Statement and Research Motivation

Traffic congestion in urban areas is a huge problem. In 2000, travel delay in US urban areas was 4.0 billion hours, a total of 1.6 billion gallons of fuel was wasted, and congestion cost was $79 billion. In 2010, travel delay was increased to 4.8 billion hours, 1.9 billion gallons of fuel were wasted, and total congestion cost was increased to $101 billion [1]. In addition, traffic congestion is a major contributing factor to greenhouse gas emissions and consequently environmental pollutants. Proper management of traffic supply in urban areas could potentially reduce some of these costs (delay, fuel consumption, etc.) and improve their livability, safety, and economic competitiveness. This can be achieved by optimizing traffic signal timing parameters in these areas.

In fact, much research is devoted to remedy traffic congestion in urban networks. Studies such as [2], [3], [4], [5], [6], [7], and [8] developed signal control schemes for oversaturated conditions using fixed-time plans. During oversaturated periods, traffic flow condition changes over time. Therefore, the application of fixed-time signal timing plans to oversaturated condition results in sub-optimal signal timing, and consequently sub-optimal network performance.

The next step in urban traffic management was the introduction of real-time signal strategies. Using real-time approach can overcome the problem of fixed-signals. In a real-time method, signal timing parameters change over time in response to a time-variant demand. Studies such as [9], [10], [11], [12], [13], [14], [15], [16], and [17] developed real-time signal plans for oversaturated condition. However, these studies are either only applicable to very small and simplified networks or use simplified
traffic flow propagation models that are not capable of accurately addressing oversaturated conditions. This may result in sub-optimal network performance.

These simplified traffic flow propagation models are deterministic while traffic related problems are stochastic. Deterministic approaches cannot accurately model a stochastic problem. For instance, macroscopic models neglect different drivers’ behaviors in following their leader, acceleration, deceleration, lane changes, etc. In fact, they implicitly assume that all drivers act identically, accelerate and decelerate similarly, keep identical headways from their leader, travel with identical speeds, do not change lanes, do not block the intersections, etc. These methods, provide valuable insights about the problem, however due to their simplistic nature, complex system dynamics and random driver behavioral tendency, along with the inherent ill-behaved nature of traffic related problems, their prediction of the state of a transportation network may be significantly different than reality. For instance, they may find relatively short queue lengths while queues are long enough to block upstream intersections. As such, their application to signal timing optimization problem may result in finding sub-optimal solutions.

Adaptive signal control methods are known to be effective tools to control traffic congestion in urban areas. They adjust signal timing parameters in response to a time-variant traffic demand. Compared to fixed-time methods, adaptive systems improve network performance especially in undersaturated conditions [18], [19], [20], [21], [22], [23]. However, in oversaturated conditions their benefits are limited due to the following reasons:

a) Adaptive methods use simplified traffic flow propagation models. These models are not capable of accurately predicting traffic condition in oversaturated conditions.

b) Adaptive methods do not dynamically coordinate signals. In fact, user needs to specify the corridors where signal coordination is needed.
c) Adaptive methods do not optimize signal timing parameters in combination to each other since it significantly enlarges the decision space and complicates the process of finding a globally optimal solution in real-time.

d) Adaptive methods use several heuristics in the process of signal timing optimization that can potentially result in finding sub-optimal solutions.

e) Adaptive methods do not assign traffic.

None of the existing signal timing optimization algorithms has all of the following capabilities together:

1- accurately addressing oversaturated conditions
2- accounting for stochastic driver behavior’s and arrival headways
3- being applicable to more realistic traffic flow and network geometric conditions
4- managing traffic supply and demand in combination to each other
5- dynamically optimizing signal timing parameters (cycle length, green splits, offsets) in combination to each other

The main objective of this research is to develop a dynamic signal timing optimization program that has all the above-mentioned capabilities. In order to be able to accurately address oversaturated conditions, account for stochastic driver behavior and arrival headways, and be applicable to realistic traffic flow and network geometric conditions, using microscopic traffic flow propagation models is required. These capabilities are achieved at the expense of increasing the complexity of the problem. When microscopic models are used, the structure of the objective function is unknown. As such, the optimization techniques that rely on information on the structure of the objective function cannot be used anymore.
In addition, managing traffic supply and demand together, and dynamically optimizing signal timing parameters in combination to each other significantly enlarges the decision space of the problem. The decision space is so large that traditional search techniques (e.g. exhaustive search) or methods such as dynamic programming cannot be used.

As such, the second objective of this research is to develop an efficient solution technique to solve the problem. For this purpose several decomposition techniques and heuristics will be used to find near optimal solutions in reasonable amount of time.

Finally, the third objective of this study is to use the developed method to optimize network-level policies on prohibiting or allowing left turns, and on traffic metering. Optimizing these two is not possible without developing an efficient signal timing optimization algorithm. Both problems will be formulated and solution techniques will be developed to solve them.

### 1.2 Research Objectives

The main goal of this research is to formulate and develop a meta-heuristic algorithm to solve dynamic signal timing optimization problem for urban traffic networks with oversaturated intersections. The specific objectives are as follows:

a) Develop analytical formulation for dynamic signal timing optimization and system optimal traffic assignment algorithms

b) Develop solution techniques for the proposed model

c) Compare the developed algorithm solutions to solutions of Direct-CORSIM optimizer

d) Determine the most efficient algorithm in solving the problem among simple GA, Elitist simple GA, Micro-Elitist GA, self-adaptive ES, and Elitist self-adaptive ES

e) Develop a program for optimal left-turn management in urban transportation networks
f) Study the effects of using a common cycle and variable cycles on network performance efficiency

g) Develop a program for optimal traffic metering strategy in urban transportation networks

1.3 Research Tasks

The following tasks were performed to achieve the objectives:

a) Review the existing literature on
   a. Signal timing optimization
   b. Dynamic traffic assignment
   c. Evolutionary algorithms (different variations of GA and ES)

b) Formulate the Intelligent Dynamic Signal Timing Optimization Program (IDSTOP)

c) Develop the solution technique for IDSTOP
   a. Develop code for different variations of GA and ES
   b. Incorporate the code with microscopic traffic flow propagation models

d) Compare IDSTOP solutions to Direct-CORSIM solutions on a realistic case study network

e) Study the effects of different optimization techniques in solving the problem

f) Develop a program for optimal left-turn management strategy to improve network performance

g) Study the effects of using common cycle and variable cycles strategies on network performance efficiency

h) Develop a program for optimal network metering strategy to improve network performance
1.4 Scope and Contributions of Research

This study developed an intelligent dynamic signal timing optimization program (IDSTOP) for urban transportation networks with oversaturated intersections. IDSTOP was formulated and a new objective function was introduced that was maximizing the weighted number of completed trips. The number of completed trips for each origin destination pair was multiplied by the length of shortest path connecting the two pairs to distinguish longer trips from shorter ones. A set of constraints were developed to ensure that the solutions are feasible/reasonable. Transportation supply was managed simultaneously with transportation demand. This was done by including traffic assignment constraints in the formulation of IDSTOP.

In this study, origin destination demand is given for the entire study period. IDSTOP does not consider stochastic traffic demand. IDSTOP is not aimed at finding solutions on-line, to be implemented in actual network. Instead, it creates off-line solutions that can be used to study signal timing optimization procedure especially in oversaturated conditions. IDSTOP takes stochastic drivers’ behaviors (in acceleration and deceleration rates, lane changes, joining the back of queue in an almost full link) and stochastic vehicular arrival headways into account but, is not designed to account for other stochastic events such as traffic incidents, vehicle failures, traffic signal failures, emergency vehicles, etc.

The main contributions of this research include:

1) The expansion of existing deterministic signal optimization models to probabilistic case

2) The expansion of existing signal optimization models to more realistic network geometry, and traffic conditions

3) Developing an efficient method to solve IDSTOP including the introduction of a new objective function, new constraints and heuristics

4) Developing a program for optimal left-turn management to improve network performance
5) Developing a methodology to study the effects of common cycle to variable cycles policies on network performance efficiency

6) Developing a program for optimal network metering to improve network performance

As mentioned above, IDSTOP is not designed for real-world operation. In fact, its runtime does not meet on-line application requirements; however, the outcome of IDSTOP can be used to improve the performance of adaptive signal timing tools by reducing their search space, or providing them with the best strategies (e.g. coordinating the signal, using long or short cycles) to be implemented for recurrent conditions in transportation networks. When fixed-time signal plans are used in real-world, IDSTOP can find solutions for recurring traffic demand in the network and its solutions can be implemented. In addition, IDSTOP can be used for planning and design purposes. It can be used to predict network performance efficiency in (near or far) future by using forecasted traffic demands. For planning purposes, it could be used to find out whether or not the current transportation supply is enough to meet the demand by a more efficient supply management (e.g. signal timing optimization), or by a more efficient simultaneous demand and supply management (e.g. signal timing optimization and traffic assignment), or by changes in network management policies (prohibiting or allowing left turns, traffic metering, making some streets one-way). If none of these methods are sufficient, IDSTOP can be used to find those locations in the network for which transportation supply needs to be increased (e.g. adding a lane).

1.5 Thesis Organization

This document is divided into 10 chapters. Chapter 2 contains a critical review of relevant literatures. Chapter 3 presents the mathematical formulation of the problem and explains the objective function and constraints of the problem in details. Chapter 4 explains the proposed procedure (IDSTOP) to optimize signal timing parameters in the network. It also describes development of Genetic
Algorithm, and Evolution Strategy methods that will be used. Chapter 5 discusses the comparison process of IDSTOP to Direct-CORSIM optimizer using a realistic sample transportation network. Chapter 6 compares the efficiency of different optimization method to each other, determines the most efficient one(s) and provides useful information on running each method. Chapter 7 describes the development of a program for optimal left turn management in transportation networks and Chapter 8 compared variable cycle lengths strategy to common cycle length strategy. In chapter 9 the development of program for optimal traffic metering in urban transportation networks is explained. Finally, chapter 10 contains the concluding remarks and recommendation for future studies.

1.6 References


CHAPTER 2

BACKGROUND

In this chapter a critical review of the literature on urban traffic control and dynamic traffic assignment is presented. The majority of the discussion is on urban traffic control which is divided into three sections: a) fixed-time signal timing, b) real-time signal timing, and c) adaptive traffic control. The discussion is followed by reviewing the most relevant studies on system optimal dynamic traffic assignment which is divided into four sections: a) mathematical programming, b) optimal control formulation, c) variational inequality, and d) simulation-based methods.

2.1 Urban Traffic Control

2.1.1 Fixed-Time Signal Timing

Much research is devoted to remedy traffic congestion in urban networks. Many of the early studies developed signal control schemes for oversaturated conditions using fixed-time signal timings. This means that signal timing parameters were constant over time and did not change in response to a time-variant demand.

Gazis was one of the pioneers to study signal control in oversaturated conditions. He proposed a method to control two closely located oversaturated intersections and used service rates as control variables and minimized delay. He did not consider left turns in his study and determined service rates based on the available green time between the two traffic directions. He proposed a 3-stage traffic light operation scheme to obtain the optimal control of the two intersections. At each stage the service rate of each approach was either at its maximum or minimum. The service rate was set to its other boundary as soon as capacity reached demand. In the other word as soon as queue dissipated, green signal was ended and switched to red. This strategy resulted in no waste in green times that was important since
any wasted green translated in some loss in transportation supply and consequently a drop in network capacity. It is possible to extend this method to more than two intersections however, it increases the number of control variables that can cause some problems. This method is the first method that takes the issue of queues in oversaturated condition into account however, does not explicitly use a constraint for queue length in the analysis [1] [2].

Michalopoulos and Stephanopoulos (1977) used control theory to propose a strategy to minimize delay on a single, and on two oversaturated intersection(s) of one-way streets. Their study considered queue constraints, travel time between the two intersections, and turning movements. Their objective was to find the optimal switchover point during the oversaturated period to switch the signals. They found that in the oversaturated periods, they had to allocate the maximum green to the approach with the highest traffic demand. This resulted in some queue build up in the minor street over time. Therefore, at the switchover point, they allocated the maximum green duration to the approach with the minimum traffic demand (now with long queue), and the minimum green duration to the approach with the highest demand [3].

D’ans and Gazis (1976) extended the work of Gazis (1964) to more than one intersection. In addition, instead of a limited study period that was as long as a cycle length, they extended the study period to more than one cycle. They used fixed time signals and minimized the lost time by vehicles in queues over the entire study period. They stated that oversaturated network problems were dynamic and complex optimization problems. The complexity was due to taking a larger number of control variables into account. They found that solving oversaturation problems required optimum allocation of routes to drivers, and optimum signal switching at each intersection, simultaneously [4].

Rathi (1986) developed a method to limit or prevent the occurrence of upstream or downstream intersection blockages. He used the concept of spillback avoidance to reduce both frequency and
duration of queue spillback formed on crossing streets. He developed a model to find solutions with near-optimal offsets and splits for the major arterials that facilitated traffic flow on cross-streets. There were several assumptions in this model: a) queue storage and receiving links must be known and constant; b) the procedure is not dynamic, it uses historic traffic arrival data; and c) it assumes continuous congested condition and does not work for uncongested conditions [5].

Gal-Tzur (1993) method metered entry traffic and adjusted that to the capacity of the critical intersection. This method prevented blockages inside the network and enables the relocation of queues to the links with higher capacity inside the network. As a result, the method converted an oversaturated network to an undersaturated one. Then the available methods for undersaturated conditions were used to solve the problem. This method however, might result in extremely large queues at the boundaries of the network since those queues were not taken into account [6].

Yuan et al. (2006) determined optimal signal timing in a network of three intersections for an oversaturation period of ten minutes. They used cell transmission model, and Genetic Algorithms (GA) to find the optimal signal timing. They used a fixed cycle strategy where their algorithm determined the cycle length, green splits, and the offset for each intersection. They found out that the best signal timing with fixed cycle strategy has a cycle length that is less than the maximum cycle length. This finding was not supporting the results of other studies [7].

Zhang et al. (2010) proposed an off-line method to determine signal timing for a pre-timed two-way arterial of five oversaturated intersections. Their method determined fixed signal timing for their study period. They also formulated a scenario-based stochastic programming model to optimize signal timing along an arterial under day-to-day demand variations. They introduced a set of demand scenarios and their corresponding probabilities of occurrence. They used cell transmission models and determined cycle length, green splits, phase sequence, and offsets to minimize the expected delay incurred by “high-
consequence” demand scenarios. They used genetic algorithms to obtain signal timing on their case study arterial. They found their method working better against high-consequence demand scenarios without losing optimality in the average sense [8].

When oversaturation occurs, traffic flow condition changes over time. As a result, the application of a fixed-time signal timing plan to oversaturated condition results in sub-optimal signal timing, and consequently a sub-optimal network performance. In addition, all studies listed above used deterministic approaches to model traffic flow propagation inside the network which may result in sub-optimal performance in real-world conditions.

2.1.2 Real-Time Signal Timing

In real-time methods, signal timing parameters change over time in response to a time-variant demand. This distributes queues spatially over different links of the network and also temporally over different cycles of the study period.

Longley (1968) proposed a method that was only applicable to oversaturated and saturated conditions. His method managed queues so that a minimum number of secondary intersections were blocked. Longley’s method only dealt with congestion in secondary intersections but not with congestion in primary intersections. In the other words, it controlled blockage of secondary intersections but not blockage of primary intersections. He used queue ratio as a performance criteria and defined “queue unbalanced” as a measure of queue ratio deviation, and assumed that adjacent intersections were coordinated (this was not an output of his algorithm). His algorithm worked by changing the green split between a maximum and a minimum so that the queue unbalanced was reduced to zero. Simulation studies found Longley’s algorithm effective in saturated or oversaturated condition however, if any of the intersections became undersaturated, the algorithm would not be applicable anymore [9].
Singh and Tamura (1974) used optimal control theory to control traffic in oversaturated condition. They defined oversaturated period as a period of time when the queues remained at the intersections after the end of green signal. They used explicit constraints to control formation of queues thus, prevented heavy congestion. Their method did not take the interference of downstream queues with upstream discharge into account. This could be a reasonable assumption if the queue length were short enough to prevent spillover. They assumed that the offsets were known. This assumption could be a limitation of their study since in oversaturated condition when queues were formed the interference with the upstream signal was not avoidable. Therefore, the offsets should be changed based on the queue lengths [10].

Michalopoulus and Stephanopoulos (1978) developed a real-time strategy and compared it to their fixed-time strategy. They concluded that the real-time timing resulted in a more efficient network performance compared to the fixed-time signal timing when the traffic volume was high [11].

Pignataro et al. (1978) developed a method to manage traffic queue in oversaturated conditions by switching the green when queues reached a certain threshold. It should be noted that the method manages the queue rather than finding an optimal solution [12].

Abu-lebdeh and Benekohal (1999) developed a dynamic traffic signal control procedure for oversaturated arterials. Their method produced real-time signal timings that dynamically managed queue formation and dissipation. They assigned different priorities to arterial and cross-streets traffic for a given queue management strategy. They formulated this problem where their objective function was maximizing system throughput and penalized it by a disutility function that specified the relative importance of an arterial and cross-streets for a given queue management strategy. Their method took both one-way and two-way arterials into account however, it was restricted to a single arterial, and only two-phase signals. For a one-way arterial, their method provided dynamic time-dependent traffic
Offsets and green times were dynamically changed as a function of demand and queue lengths. They found similar results for a two-way arterial however, as expected, for the secondary direction their algorithm could not provide all the capabilities associated with the primary direction. For the secondary direction, it managed the queues so that the occurrence of queue spillback was minimized [13] [14].

Girianna and Benekohal (2002) expanded Abu-Lebdeh and Benekohal’s algorithm and proposed dynamic signal coordination models for oversaturated transportation networks. They formulated the model as a dynamic optimization problem with the objective of maximizing the total number of vehicles released by the network and penalizing it by queue accumulation along the arterials and used genetic algorithms to find the near optimal signal timing. They developed a cycle based, and a discrete-time based, network loading model. In the cycle based model, they assumed equal cycle length for all intersections of the network while in the discrete-time based model they relaxed this assumption. They used CORSIM to validate their model. They found that their model successfully managed queues along the coordinated arterials and also created opportunity for traffic progression in specified directions. Their algorithm managed local queues by spatially distributing them over some signalized intersections and by temporarily spreading them over signal cycles. If a critical signal was located at an exit point, the algorithm protected that signal from becoming excessively loaded. On the other hand, if a critical signal was located at an entry point, the algorithm reduced the queues at downstream intersections and then released the platoon from the critical intersection. This study did not take left turns into account [15].

Chang and Sun (2003) proposed a method to dynamically control an oversaturated traffic signal network by using a bang-bang like model for oversaturated intersections, and TRANSYT-7F for undersaturated intersections. They called their method maximal progression probability algorithm. Their model had two different operating procedures one for saturated and one for undersaturated conditions. They formulated the problem and proposed a heuristic method to find the signal timing. They suggested that the most congested intersection had to be chosen as the pivot intersection. At that cycle step, they
set the cycle length of all oversaturated intersections equal to the cycle length of the pivot intersection that was found by the bang-bang like control model. Then they assigned the offsets to the maximal flow rate approach at all intersections. After completing the cycle, a new pivot intersection was selected. They tested their model in a network of 12 oversaturated intersections that were surrounded by 13 undersaturated intersections and they allowed turning movements and compared it to TRANSYT-7F. They found that their method provided better results than TRANSYT-7F [16].

Lo and chow (2004) applied their Dynamic Intersection Signal Control Optimization (DISCO) method to a one-way arterial of three intersections and compared three control strategies. These strategies were: fixed-cycle or fixed-time plan, variable green split in a fixed cycle, and variable-green-no-cycle-plan. DISCO uses cell transmission model by Daganzo (1992) and simple genetic algorithms to find the near-optimal signal timing. They found out that the most flexible strategy plan, variable-green-no-cycle, did not necessarily result in the best answer under the limitations of solution heuristics, especially when there was no good initial solution. However, with good initial signal timing, this plan outperformed other plans. They supported Lo’s (2001) previous findings that the results of a variable green no cycle plan is only a few percent better than the other two cycle timing plans. They stated that the variable-green-no-cycle plan cannot contribute too much since most of the streets operated in a state of de facto red. They concluded that a dynamic plan could only result in slightly better signal timing if only used a good initial solution that was produced by a fixed-cycle plan. They stated that the reason was a larger feasible area for the dynamic plan compared to the static plan that made finding a high quality solution much harder [17] [18].

Sun and Benekohal (2006) developed a bi-level programming formulation and a heuristic solution for traffic control in an oversaturated network with dynamic demand and stochastic route choice. They formulated the problem for networks of one-way streets with turning movements with two-phase signal plans. In their bi-level programming model, the upper level represented the signal
optimization that was controlled by system manager. The lower level, modeled the traveler’s behavior. They used genetic algorithms and a cell transmission based incremental logit assignment to solve the problem and tested their method on two transportation networks. Using dynamic signal timing, reduced the average link travel time by 5-8% and up to 14% compared to a static signal timing [19].

Putha et al. (2010) used ant colony optimization to solve signal coordination problem for an oversaturated network. They formulated the problem and used ant colony to solve it and compared its results to simple genetic algorithms results. Their formulation and case study network was very similar to Girianna and Benekohal’s (2002) formulations and case study. They maximized the total number of vehicles processed by network during the saturation period and used a disutility function to penalize the occurrence of queues at the end of green signal. Similarly they used ideal offset, de facto red, coordinated loops, queue storage capacity, network flows, and control variable constraints. Their case study network had 20 intersections and one-way arterials. They did not report much detail on the signal timing that was found by ant colony and genetic algorithms however, they compared the performance of these two methods by comparing the average value of fitness function over 30 runs. They found that for most of the cases ant colony provided higher fitness compared to simple genetic algorithm except for the case with 400 population size/ants and 50 generations/trials. Although their comparison showed that ant colony optimization outperformed simple genetic algorithm in most of the cases, it did not provide details on the output signal timing to show if it was reasonable or not. In addition, they did not report any details on calibration of simple genetic algorithm they used [20].

### 2.1.3 Adaptive Signal Control

Several adaptive signal control tools have been developed to optimize and coordinate signals in realistic networks. These systems monitor traffic condition inside the network using vehicular detectors and find signal timing in response to that. Consequently, the signal timing changes over time. Adaptive
systems are either reactive, or proactive. Reactive systems react to the current traffic condition in the network. On the other hand, proactive systems predict traffic condition in near feature and take preventive actions to avoid traffic congestion.

Sydney Coordinated Adaptive Traffic System (SCATS) was developed in early 1970s in Australia. SCATS utilizes a partially decentralized architecture and relies on detectors at stop bar locations to predict downstream arrival using vehicle departures and a platoon dispersion factor. SCATS finds signal timings for background plans using the existing demands at critical intersections, and these set the base for coordination with intersections belonging to a predefined subsystem around it. However, the offsets should be provided for SCATS to use them at later times. SCATS does not optimize the offsets. It uses a feature known as marriage/divorce to dynamically group adjacent subsystems of intersections for coordination, each subsystem varying in size from one to ten intersections (NCHRP Report 340, 1991). At peak hours, cycle lengths in each subsystem are found using Webster’s method and offsets provide coordination for the direction with the highest demand. Webster’s method does not result in finding reasonable cycle length at saturation level. Therefore, SCATS uses an upper bound to limit the value of the cycle length. At off-peak hours, a cycle length is selected to provide better coordination for both directions and the objective is to minimize stops. In undersaturated conditions, the goal of SCATS is to reduce stops and delay, and near saturation it maximizes throughput and controlled queues (Traffic Detector Handbook, 2006) [21].

Split, Cycle, Offset Optimization Technique (SCOOT) is another well-known adaptive (reactive) signal control, developed by the Transport Research Laboratory (TRL) in the U.K. SCOOT is a centralized traffic-responsive system that minimizes stops and delay by optimizing cycle, splits, and offsets. The system uses detectors upstream from the intersections to predict vehicle arrivals downstream at the stop bar, and update its predictions every few seconds. The optimization is performed using heuristics from TRANSYT considering only small changes in the signal settings (given that the solution needs to be
obtained in real-time), and also not to significantly disrupt coordination in a single step. However, this limits the changes to gradual modifications over time that may be slower than what is needed under unusual circumstances (e.g. incidents), and it indicates that the optimization is local rather than global. In addition, using TRANSYT for optimization can be a limitation since its solution is local as well. SCOOT has been deployed in more than 200 cities worldwide [22], [23].

Optimized Policies for Adaptive Control (OPAC) minimizes a function of total intersection delay and stops for predetermined time horizons. Four versions of OPAC are available. OPAC I uses dynamic programming to determine globally optimal signal timing parameters for a “single” intersection. The second optimization algorithm that was developed, OPAC II, consists of a simplification of the OPAC-I algorithm. It was designed to serve as a building-block in the development of a distributed online strategy. In OPAC III, signal timings are optimized using a rolling horizon (typically as long as an average cycle) and a simplified dynamic programming approach based on detector data and predictive traffic models, but only the “head” portion of the prediction is implemented. The “head” prediction is based on actual detector information (not on the predicted demand). The system can make decisions every 1 or 2 seconds, and phase sequencing is not free but based on the time of day, skipping phases if there is no demand for such movements. It is noted that all phases are also constrained by maximum and minimum green times. The OPAC IV (or RT-TRACS) version is intended to incorporate explicit coordination and progression in urban networks and is known as the virtual-fixed-cycle OPAC. The virtual-fixed-cycle restricts the changes in cycle lengths at intersections around a given primary signal, so that they can fluctuate only in small amounts to maintain coordination. This may result in finding a local optimal solution rather than a global one. There are three control layers in the OPAC architecture: 1) local control (using OPAC III), 2) coordination (offset optimization), and 3) synchronization (network-wide virtual-fixed-cycle). The upcoming OPAC V will include dynamic traffic assignment in the optimization of the signal timings [24].
Real-time Hierarchical Optimized Distributed Effective System (RHODES) developed at the University of Arizona starting in 1991 [25]. RHODES has three hierarchical levels: 1) intersection control, 2) network flow control, and 3) network loading. RHODES optimizes different measures of effectiveness such as delay, number of stops, or throughput [26] by using real time input from vehicle detectors. It predicts traffic fluctuations in the short and medium terms to find the following phases and their duration. At the intersection control level, an optimization is carried out with the dynamic programming routine “COP” that uses a traffic flow model (called PREDICT) for a horizon that rolls over time (e.g. 20 to 40 seconds). The solution for the first phase is implemented and the optimization is performed again based on updated information. The network flow control uses a model called REALBAND to optimize the movement of platoons identified and characterized by the system (based on size and speed). It creates a decision tree with all potential platoon conflicts and finds the best solution using results from APRES-NET, which is a simplified model to simulate platoons through a subnet of intersections (similar to PREDICT). The rolling horizon at this level is in the order of 200-300 seconds. Finally, the network loading focuses on the demand on a much longer prediction horizon (in the order of one hour). Some of the limitations of RHODES arise with oversaturated conditions, under which the queue estimations may not be properly handled by PREDICT. Also, the predictions consider signal timing plans for upstream intersection, which may change at any point in time creating deviations between the estimated and actual arrival times at the subject intersection. Lastly, there are several parameters used in the queue predictions such as queue discharge speeds that should be calibrated to field conditions, and the fact that an upper layer is used for network coordination demands additional infrastructure.

Real-Time Traffic Adaptive Control Logic (RTACL) was derived from OPAC and specifically designed for urban networks. This system uses macroscopic model to select the next phases. Most of the logic is based on local control at the intersection level, and the predictions are found for the next two cycles (short term), leading to recommendations for the current and the next phase, and long-term
estimations for the following phases. These recommended actions (short and long term) to generate estimates of demand that are used at the network level by nearby intersections, which can adjust their decisions based on the new predictions [27]. RTACL may be more suitable for undersaturated conditions since its macroscopic model may not be able to properly handle oversaturated conditions. In addition, its solution is local rather than global optimum.

Programmation Dynamique (PRODYN) was developed by the Centre d’Etudes et de Recherches de Toulouse (CERT), France. PRODYN uses a rolling horizon for the optimization and predicts vehicle arrivals and queues at each intersection every five seconds and for periods of 140 seconds. At the intersection level, it minimizes delay by forward dynamic programming with minimum and maximum green time constraints. At the network level it simulates and propagates the outputs to downstream intersections for future forecasting [28]. It has a centralized (PRODYN-H) and a decentralized version (PRODYN-D). PRODYN-H has shown better performance, but due to its complexity is limited to a very low number of intersections. PRODYN-D comes in two versions: one with information exchange between intersections (better suitable for networks), and one with information from the immediate links.

Urban Traffic Optimization by Integrated Automation/Signal Progression Optimization Technology (UTOPIA/SPOT) was developed by Mizar Automazione in Italy. It has a module for optimization of a given criteria (e.g. delay or stops) at the intersection level (SPOT) and one module for dealing with area-wide coordination between intersections (UTOPIA), with the objective of improving mobility for both public and private transport. Intersections with SPOT share signal strategy and platoon information with their neighbors for better network operation, but UTOPIA is needed for an increased number of intersections linked together, allowing for area-wide predictions and optimization. The predictions at the network level (and the optimized control) are made for a horizon of 15 minutes, and individual intersections compute their own predictions (for the next two minutes) using local data.
Adjustments to the signal strategies can be made every three seconds. Deviations with the network-level predictions are sent to the central controller so that better predictions for other intersections are available [29].

Adaptive traffic control tools described above have the potential to improve system-wide performance and they use real-time data for determining a control policy. Some of them have been proved in field installations with successful results and have been distributed extensively around the world. They are flexible in the sense that they can frequently change cycle times (or they are acyclic) and have the capability to adjust the signal strategy based on predictions every few seconds. However, as it has been pointed out [30], they have some limitations in terms of uncertainty in the predictions of traffic flow and arrival times, and their lack of evolving mechanisms for self-adjusting or learning over time. In addition, some of the current adaptive control systems (OPAC, PRODYN, and RHODES) use recursions based on dynamic programming or enumeration of a reduced version of the available space for a given rolling horizon, but with the shortcoming that the best solutions are based for the most part on predicted traffic, which may not be accurate enough to obtain optimal behavior (it is also recalled that the forward dynamic programming recursions find the optimal values and then move backward in time to estimate the optimal policy, from the end of the horizon, which has the most uncertainty). Overall, the adaptive system reviewed above, significantly improve network performance compared to previous systems however, they are not aimed at finding globally optimal signal timing for the network due to their real-time constraints. In addition, they are not able to determine which movements to coordinate. In fact, this is one of their input data. Moreover, they also do not optimize all decision variables simultaneously and in combination with each other. In addition, these methods use models to predict traffic condition in future that are very simplified (to reduce runtime) and are usually not capable of accurately modeling oversaturated conditions. Finally, adaptive models do not simultaneously manage transportation supply and demand. It should also be noted that these adaptive
methods minimized delay alone, or in combination with stop minimization or speed maximization. As shown in different studies, in oversaturated conditions improving the capacity of the network is more important than reducing delay [31] [32] [33] [13].

2.2 Dynamic Traffic Assignment

Static-demand and deterministic user equilibrium and system optimal problems can be easily solved by Frank-Wolf algorithm. However, complex system dynamics, random driver behavior, and the inherent ill-behaved nature of DTA problem, results in complicated modeling issues associated with analytical methods [34]. A lot of formulations and solution approaches have been introduced since the pioneering work of Merchant and Nemhauser in 1978. These works can be categorized into four groups based on their methodology:

1) Mathematical programming,
2) Optimal control formulation,
3) Variational inequality, and
4) Simulation-based methods

In the rest of this chapter, these methods are briefly explained and their strengths and drawbacks are highlighted.

2.2.1 Mathematical Programming

Mathematical programming DTA models discretize the time and formulate the problem in that discretized time-setting. The first attempt to formulate the DTA problem as a mathematical program (by Merchant and Nemhauser) was limited to the deterministic, fixed-demand, single-destination, single-commodity, system optimal case [35] [36]. The model was based on link exit function to propagate traffic, and a static link performance function to find travel cost as a function of link volume. The
formulation was a flow-based, discrete time, non-convex nonlinear mathematical program. This model provided a proper generalization of the conventional static system optimum assignment problem, and the global solution was obtained by solving a piecewise linear version of the model. Later, in 1980, Ho proved that such global optimum could be determined by solving a sequence of at most $N + 1$ linear programs, where $N$ is the number of time periods [37].

Carey proved that the Merchant-Nemhauser model satisfies the linear independence constraint qualification because the proposed exit function was continuously differentiable [38]. Carey manipulated the exit functions to obtain mathematical and algorithmic advantages over the original formulation and make it a well-behaved convex nonlinear program [39]. This mathematical program could be solved by regular mathematical programming software. The formulation was extended to handle multiple destinations instead of one. The formulation had the non-convexity issues resulted from First-In-First-Out (FIFO) property. This problem exists in all mathematical programming approaches for both user equilibrium and system optimum cases. The FIFO requirement can be easily satisfied in a single destination DTA however, in general networks it requires adding additional constraints to the formulation that results in a non-convex feasible area. This non-convex feasible area significantly increases the computational requirements to solve the problem and usually makes it impossible to get real-time results [40].

In addition, in system optimal DTA, “holding-back” issue is likely to be observed. This means that the traffic in minor roads may be hold for unusually long periods of time to reduce system’s total delay. Some of the issues relate to FIFO property, and holding back are presented by Carey and Subrahmanian (2000) [41].

Later in 1993, Birge and Ho extended Merchant-Nemhauser model to stochastic demand. They relaxed the assumption that the O-D is known for to entire study period by developing a multistage
stochastic mathematical programming formulation that was neither linear nor convex. Their model assumed a finite number of scenarios of random variable realizations. This formulation assumed that current assignment decisions were independent of future O-D [42].

Based on the cell transmission model (Daganzo, 1994), Ziliaskopoulus (2000) developed a linear programming formulation for single destination system optimal DTA. Using cell transmission model, he obtained link volumes and travel cost in each time step of the study period. The model was more sensitive to traffic realities and provided some insights on the DTA problem properties but was not an operational model for real-world applications [43].

Li et al. (1999) modeled system optimal dynamic assignment as a linear programming with multi-origin multi-destination. They used cell-transmission model and observed that FIFO constraints were generally satisfied [44].

Abdul Aziz and Ukkusuri (2011) proposed a bi-level and a single-level formulation to simultaneously manage traffic supply and demand. They used cell transmission model and solved the single-level program. They optimized phase durations and found system optimal traffic assignment. They concluded that their model found a better solution than fixed-time signals [45].

An issue in this part is the trade-off mathematical tractability with traffic realism. For example, to represent the FIFO property, non-convex constraints are needed. Non-convexity in DTA results in loss of analytical and computational tractability for deployment in general networks. In addition, this method usually has problems related to: the used of link performance/exit functions; tracking back of queues; efficient solutions for real-time purposes in large-scale networks; and a clear understanding of solution properties for realistic scenarios.
2.2.2 Optimal Control Formulation

In constrained optimal control theory DTA formulations, it is assumed that O-D demands are known as a continuous function of time, and link flows are determined as continuous functions of time as well. Constrains are similar to those for the mathematical programming; however, instead of being defined for discrete time intervals, they are defined for continuous time setting.

Link-based optimal control formulation for a single destination case was introduced by Friesz et al. in 1989. The model included both system optimal and user equilibrium objectives and assumed that adjustments from one state to another may occur concurrently as the network condition changes. The system optimal model is a temporal extension of the static system optimal model [46].

Using optimal control theory, Ran and Shimazaki (1989) developed a link-based system optimal model for urban transportation network with multiple origins and destinations. To reduce the computational complexity of the problem, they used linear exit functions and quadratic link performance functions. Their model had two issues: a) unrealistic modeling of the congestion, and b) not taking the FIFO constraint into account [47].

Optimal control theory was an attractive method to describe dynamic systems, however negative factors still exist: 1) The lack of explicit constraints to ensure FIFO and holding of vehicle at nodes 2) The lack of realistic modeling of congestion and over-saturation 3) The lack of solution approach for general networks.

2.2.3 Variational Inequality

Variational Inequality (VI) provides a general formulation platform for several classes of problems in DTA context like: optimization, fixed point, and complementarity. VI handles more realistic traffic scenarios and sensitivity analysis and extensions can be easily performed. This approach is more general than other two approaches and provides greater analytical flexibility and convenience in
addressing various DTA problems. VI highlights the inability of the mathematical programming approaches in addressing scenarios with asymmetric Jacobean matrices for the travel cost functions. However, VI methods are more computationally intensive than other two methods. In addition, traffic realism issue exists in these models. Several studies have used the concept of VI for DTA [48] [49].

2.2.4 Simulation-Based Methods

In contrast to analytical DTA models, simulation based DTA models use a traffic simulator to capture the system dynamic and drivers’ behavior on route choice. Traffic simulator is flexible to replicate the traffic propagation, holding back, congestion and physical queue impact, signal coordination, and randomness of drivers’ behavior. Simulation based DTA model have gained greater acceptability for real-world deployment due to its flexibility and fidelity.

In simulation based models, simulator is dedicated to determining the shortest path and search for optimal solution, in addition to propagating the traffic. Mahmassani and Peeta ([50] [51] [52]) used a mesoscopic traffic simulator, DYNASMART, as part of an iterative algorithm to solve System Optimal (SO) and User Equilibrium (UE) solution of their DTA models. From a computational standpoint, further modification is still required to deploy their deterministic DTA models to real-time environment. CONTRAM simulator ([53] [54] [55]) was implemented to address SO and UE DTA problems by Ghali and Simith (1992) [56] and Smith (1994). Rolling horizon DTA models was developed by Peeta and Mahmassani (1995) to improve computational efficiency. The rolling horizon DTA can use the current information and near-term forecast for a solution in quasi real-time situation. DynaMIT was introduced by Ben-Akiva et al. (1997) [57] to approximate real time traffic condition in a dynamic traffic assignment system, which consists of two simulators, demand and supply simulator. Vehicles are moved in packets in these mesoscopic simulators to reduce computational load. However, it is incapable of handling the
randomness of driver’s behavior and vehicle characteristics, the impact of physical queue at signalized intersection, and gap acceptance behavior (similar to the other simulation models mentioned above).

2.3 Summary

In this chapter previous studies in the area of urban traffic control and dynamic traffic assignment were reviewed. In summary, both fixed-time and real-time signal timing optimization approaches are based on deterministic and oversimplified models to represent traffic propagation in transportation networks. These methods, provide valuable insights about the problem, however due to their simplistic nature, complex system dynamics and random driver behavioral tendency, along with the inherent ill-behaved nature of traffic related problems, their optimal solution may result in sub-optimal network performance in real world.

Adaptive traffic control overcomes some of these issues; however, they are not capable of finding an optimal solution since they do not optimize all signal timing parameters. In fact, since doing so requires extensive computations that usually exceed the real-time constraints, only some of the signal timing parameters are optimized. Use of simplified prediction models is another limitation of adaptive models especially in oversaturated condition. In addition, none of the adaptive models manage both traffic supply and demand simultaneously. Finally, adaptive systems need to know the corridors were signal coordination is desired as an input.

Based on the review of the state of the art in urban traffic control we identified lack of a signal timing optimization method that simultaneously manages traffic supply and demand, optimize all signal timing parameters, considers complications that occur in oversaturated conditions, and account for different driver behaviors and vehicle specifications. The main objective of this study is to formulate and develop a solution technique to solve such problem.
2.4 References


CHAPTER 3

IDSTOP FORMULATION

3.1 Introduction

Finding signal timing parameters that result in an efficient network performance can be formulated as an optimization problem. The decision variables of this problem are signal timing parameters (i.e. number of phases, cycle length, green splits, and offsets) and the objective function is to optimize one or several Performance Measures (PM) of the network (i.e. number of trips, network throughput, vehicle-mile travelled, average speed, delay, travel time, emissions, etc.). In addition, several constraints are needed to ensure that the solution is feasible and/or desired (e.g. a solution that creates excessively long delays at a minor road may not be considered desired, a very short cycle length of for example 10 seconds may not be considered feasible).

Network performance may be further optimized if drivers are dynamically routed to the paths with lower traffic congestion. This can be achieved by dynamically assigning traffic in the network. There are two major traffic assignment approaches: user equilibrium and system optimal. Since the focus of this study is to identify the optimal network performance, system optimal concept is used. In this case, the entire problem of signal timing optimization and traffic assignment could be formulated in a single level optimization program. The decision variables are signal timing parameters, and turning volumes at each intersection over time.

In the rest of this chapter, the decision variables, objective functions, and the constraints of the problem will be introduced. At the end a method to obtain an upper-bound to IDSTOP objective function is introduced.
3.2 Decision Variables

The decision variables of the problem are signal timing parameters of all intersections of the network in each time interval. That is, number of phases (phase plan), the cycle lengths, green splits, and start time of the first green of all intersections at each time interval. For traffic assignment purpose, turning volumes at each intersection at each time interval are also decision variables. The list of all decision variables and their notation is presented below:

\[
\begin{align*}
\Phi_i^t & = \text{number of phases at intersection } i \text{ at time interval } t \\
C_i^t & = \text{cycle length of intersection } i \text{ at time interval } t \\
s_{i,k}^t & = \text{split for green for phase } k \text{ of intersection } i \text{ at time interval } t \text{ (see Figure 3.1)} \\
sof \ g_i^t & = \text{start time of the first phase of intersection } i \text{ at time interval } t \\
y_{rs,ij}^t & = \text{turning traffic volume at upstream intersection } i \text{ moving towards downstream intersection } j \text{ on a path from source node } r \text{ to a sink node } s \text{ at time step } t
\end{align*}
\]

Figure 3.1. Phases in an intersection
3.3 Objective Function

The ultimate goal of signal timing optimization is to improve transportation network performance. This can be achieved by selecting different network Performance Measures (PM) to be optimized as the objective function of the problem. Proper selection of the objective function (i.e. which PM of the network to be optimized) is extremely important due to the following reasons: a) Optimizing different objective functions may result in finding different solutions; b) Optimizing some objective functions may require adding extra constraints to the problem; c) Optimizing some objective functions may require a larger area of the network to be simulated which is computationally expensive; and d) Different objective functions may have different convergence speeds. Some candidate objective functions are:

1- Delay minimization (OB1),
2- Travel time minimization (OB2),
3- Throughput minus queue maximization (OB3),
4- Trip maximization (OB4), and
5- Weighted trips maximization (OB5).

3.3.1 Delay Minimization

For a specific trip, travel delay is the time difference between the actual travel time, and the hypothetical ideal travel time (under free flow conditions and the absence of traffic control devices). Therefore, travel delay minimization (i.e. reducing total travel delay for all vehicles for the entire study duration), on the average reduces travel times and brings them closer to the ideal travel time. This is desired however, to get the best results one needs to pay attention to the following point. When no vehicle enters the study area (i.e. the network) travel delay is at its lowest level (i.e. zero). Therefore, delay minimization may found solutions that keep many vehicles outside of the study area (i.e. where
delays are not calculated) and let only a small number of them enter the network. This results in lower travel delays inside the network at the expense of excessive delays at the boarders. This could be prevented by:

a- Expanding the study area such that delay at the boarders of the network is taken into account. This ensures that not too many vehicles are metered (for the purpose of improving interior network performance). However, this method is computationally expensive especially in oversaturated conditions. In these conditions, queues at the boarders may become too long (due to the large traffic demand) and require substantial length of entry links to be modeled (computationally expensive especially when microscopic models are used).

b- A set of constraints may be used to ensure that all traffic demand is entered the network. This strategy will work for undersaturated conditions where traffic demand is below network capacity. However, in oversaturated conditions, it is not possible to process all traffic demand because network does not have enough capacity to do so. Therefore one needs to decide how much of traffic demand should be entered into the network which is a very challenging task and has significant influence in the solution of the problem.

### 3.3.2 Travel Time Minimization

Another objective function that has great potential to improve network performance is travel time minimization that is minimizing total travel time for all vehicles in the network for the study period. Travel time, is one of the most direct costs experienced by users of a transportation network. It simply is equivalent to the time needed to process a vehicle in the network. Therefore, when it is minimized, total process time for vehicles is reduced and as such, the network performance is improved. However, similar to delay minimization, not letting vehicles into the network results in lowest possible total travel time (i.e. zero). To prevent holding vehicles at the entry points, one needs to either expand the entry
links of the network to more accurately estimate travel time (which is computationally expensive) or need to add constraints to the problem to ensure all demand is satisfied (which is not possible in oversaturated conditions).

3.3.3 Throughput-Minus-Queue Maximization

Throughput maximization increases the capacity of the system in processing more vehicles [2], [3], [4], [5], etc. When throughput (i.e. sum of vehicles released from each link of all intersections of the network over the entire study period) maximization is used, queues may grow in some certain cases especially when a downstream intersection has less capacity than its upstream intersection. This queue growth may create a gridlock which should always be avoided. To take care of this issue, Abu-Lebdeh and Benekohal, and Girianna and Benekohal added a disutility function to their objective function that penalized the value of throughput based on the queue lengths in different links. This penalty took care of the issue of long queues in the network. Their studies indicated that throughput-minus-queue maximization was a very reasonable objective function in oversaturated conditions. However, when system optimum traffic assignment is performed, maximizing throughput alone may result in circulating vehicles inside the network since this circulation can increase the value of the objective function. This circulation can be prevented by adding some constraints; however, adding such constraints introduces more complexity to the problem. In addition, since the queue length at entry links are also important, an extended length of entry links may be needed to be modeled to accurately estimate queue lengths at the boarders of the network. This is computationally expensive especially when a microscopic model is used.

3.3.4 Trip Maximization

Another objective function that has benefits similar to throughput-minus-queue maximization concept, but does not circulate vehicles inside the network is maximizing the number of completed trips
hereafter we call it trip maximization. It ensures that vehicles have exited the network which is desired since the more vehicles exit the network in a time interval, the more efficient the performance of the network during that time. In addition, when system optimum traffic assignment is performed, trip maximization does not encourage vehicle circulation in the network since circulations delay the exit time of vehicles from the system and as a result reduce the number of completed trips in a time interval. One significant benefit of trip maximization concept is that it does not require modeling extended lengths of entry and exit links. The reason is that the lengths of these links do not change the number of completed trips inside the network which is the only parameter that determines that number of completed trips in the system (since traffic is only controlled inside the network and not on entry and exit links). Therefore, one only needs to model the interior of the network to solve the problem. This is computationally more efficient than modeling the network and extended portions of entry/exit links.

One drawback of trip maximization concept is that it does not distinguish between very short and very long trips. Therefore, it may maximize the value of the objective function by processing too many short trips and not many long ones. This is not desired.

3.3.5 Weighted Trip Maximization

As mentioned above, trip maximization treats short and long trips equally (however, longer trips produce more negative effects than what shorter trips do). To avoid this, each trip is weighted by the length of shortest path (in terms of distance) from its origin to its destination and hereafter this objective function is called weighted trip maximization. Note that the actual length of trips should not be used since it encourages using longer routes and potentially circulates vehicles inside the network. Weighted trip maximization aims at improving the capacity of the network, gives more opportunity to trips with longer shortest paths, and does not encourage increasing the length of trips inside the network.
network (since traveling in longer routes delays finishing the trips and consequently reduces the value of objective function).

### 3.3.6 Choosing the Objective Function

The objective functions discussed above, can be categorized into two groups: a) minimizing travel cost in the network (i.e. travel delay and travel time minimization), and b) maximizing network capacity (i.e. throughput-minus-queue, trip, and weighted trip maximization). It is shown that in oversaturated conditions, increasing system capacity to process more vehicles is more important than reducing travel time or travel delay [1], [2], [3], [4]. Among the objective functions that aim at improving network capacity, weighted trip maximization has the following benefits:

1- for a single trip, does not encourage longer routes and do not circulate vehicles in the network
2- gives more opportunity to trips with longer shortest-path (i.e. trips that require travelling more in the network),
3- does not require to model extended length of entry and exit links

As such, weighted trip maximization offers great potential for efficient network performance especially in oversaturated conditions. To make sure that this is true, a simulation based method is used that compares the effects of optimizing each objective function on network performance. In a realistic case study network for four different demand patterns, signal timing optimization problem is solved using each objective function (a total of $4 \times 5 = 20$ optimization runs), see in Figure 3.2. The four different demand patterns are:

1- Symmetric undersaturated demand pattern (DP1)
2- Symmetric oversaturated demand pattern (DP2)
3- Asymmetric undersaturated demand pattern (DP3)
4- Asymmetric partially oversaturated demand pattern (DP4)
After finishing each optimization run, the optimized signal timing parameters and turning percentages were coded in microscopic traffic simulation model (CORSIM) and 250 simulation runs with different seeds were made to cover a wide range of vehicle arrival headways and driver behaviors and to account for internal variability of CORSIM (details on the number of runs is available in Chapter 5). Eight following PM were collected during the 250 microscopic replications of all 20 combinations of different objective functions and demand patterns:

1- Travel delay inside the network
2- Travel time inside the network
3- Throughput-minus-queue
4- Number of completed trips
5- Weighted number of completed trips
6- Delay at the boarders
7- Total delay (travel delay inside the network plus delay at the borders for each seed)
8- Average speed

For each demand pattern, for each PM, Least Significant Difference (LSD) test with 95% significance level was performed across the five different objective functions to show any statistical difference between the values of each PM for different objective functions (eight LSD tests for each demand pattern, total of $8 \times 4 = 32$ tests for all four demand patterns). The procedure of choosing the most appropriate objective function is shown in Figure 3.3. Set $OF$ is set of all objective functions and set $DP$ is set of all demand patterns used to test different objective functions.

![Flowchart of Methodology of choosing IDSTOP objective function](image-url)
To choose the best objective function for each demand pattern, we looked at total delay in the entire system (sum of delay inside the network and delay at its borders). The objective function that results in the lowest total delay is selected as the most appropriate objective function for each demand pattern. If total delays happen to be similar, average speed is used as the second criteria. If both total delay and average speed were similar, weighted number of completed trips is used as the third criteria.

**Figure 3.4.** Network PM for each demand pattern and each objective function

a) Delay (inside, at the borders, and in the entire system), travel time inside the network, and speed inside the network

b) Throughput-minus-queue, number of completed trips, and number of weighted completed trips
Note that neither a set of constraints was used to ensure that all traffic demand is satisfied (since it is not possible in oversaturated condition) nor the lengths of entry links were extended during the optimization process (since significantly increases the runtime). In all four demand patterns, as expected, travel delay and travel time minimization objective functions resulted in overall shorter delays and travel times inside the network, respectively; however, this was achieved at the expense of keeping more vehicles at the boarders of the network (compared to methods who aimed at maximizing the capacity of the network). This was confirmed by looking at delays at the borders that were longer and number of completed trips that were lower for delay and travel time minimization objective functions, see in Figure 3.4 a-b.

In oversaturated conditions, for all four demand cases delay at the borders for throughput-minus-queue, trip, and weighted trip maximization objective functions were significantly less than that for travel delay and travel time minimization objective functions. In fact, this resulted in statistically significantly lower total delay for the objective functions that aimed at maximizing network capacity compared to those that aimed at reducing travel cost (except for asymmetric undersaturated condition in which trip maximization and travel time minimization resulted in similar total delays), see in Table 3.1. This indicated the advantage of the objective functions that aimed at maximizing network capacity over those who aimed at reducing travel cost in oversaturated conditions. It should be noted that we expect that travel delay and travel time minimization objective functions found much more efficient solutions if extended length of entry links were modeled. However, we did not perform the optimization with long entry links since it was extremely computationally expensive.
Table 3.1 Performance Measures for Different Objective Functions under Different Demand Patterns

<table>
<thead>
<tr>
<th>Demand Pattern</th>
<th>Objective Function</th>
<th>Network Performance Measure (PM)</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Delay Inside (h)</td>
<td>Travel Time Inside (h)</td>
<td>Throughput Minus Queue (veh)</td>
<td>Trips (veh)</td>
<td>Weighted Trips (veh)</td>
<td>Total Delay (h)</td>
</tr>
<tr>
<td>Symmetric Undersat.</td>
<td>Min Delay</td>
<td>68.3 A</td>
<td>123.8 A</td>
<td>12107 A</td>
<td>2790 A</td>
<td>1494 A</td>
<td>22.5 A</td>
</tr>
<tr>
<td></td>
<td>Min Travel T.</td>
<td>70.2 B</td>
<td>122.4 B</td>
<td>11829 B</td>
<td>2754 B</td>
<td>1484 B</td>
<td>23.6 B</td>
</tr>
<tr>
<td></td>
<td>Max T-Q</td>
<td>68.4 A</td>
<td>123.9 A</td>
<td>12620 C</td>
<td>2875 C,D</td>
<td>1557 C</td>
<td>20.9 C</td>
</tr>
<tr>
<td></td>
<td>Max Trip</td>
<td>68.5 A</td>
<td>123.2 C</td>
<td>12401 D</td>
<td>2880 C</td>
<td>1562 D</td>
<td>20.8 C</td>
</tr>
<tr>
<td></td>
<td>Max Wt. Trip</td>
<td>68.4 A</td>
<td>123.3 C</td>
<td>12532 E</td>
<td>2870 D</td>
<td>1632 E</td>
<td>20.8 C</td>
</tr>
<tr>
<td>Symmetric Oversat.</td>
<td>Min Delay</td>
<td>111.1 A</td>
<td>170.6 A</td>
<td>10113 A</td>
<td>2991 A</td>
<td>1611 A</td>
<td>65.7 A</td>
</tr>
<tr>
<td></td>
<td>Min Travel T.</td>
<td>112.5 B</td>
<td>165.9 B</td>
<td>8463 B</td>
<td>2799 B</td>
<td>1491 B</td>
<td>75.5 B</td>
</tr>
<tr>
<td></td>
<td>Max T-Q</td>
<td>111.7 A</td>
<td>174.0 C,D</td>
<td>11928 C</td>
<td>3204 C</td>
<td>1731 C</td>
<td>49.3 C</td>
</tr>
<tr>
<td></td>
<td>Max Trip</td>
<td>111.4 A</td>
<td>174.3 C</td>
<td>11856 C</td>
<td>3249 D</td>
<td>1757 D</td>
<td>49.0 C</td>
</tr>
<tr>
<td></td>
<td>Max Wt. Trip</td>
<td>111.1 A</td>
<td>173.7 D</td>
<td>11896 C</td>
<td>3206 C</td>
<td>1835 E</td>
<td>47.3 D</td>
</tr>
<tr>
<td>Asymmetr. Undersat.</td>
<td>Min Delay</td>
<td>60.8 A</td>
<td>111.4 A</td>
<td>11593 A</td>
<td>2511 A</td>
<td>1408 A</td>
<td>20.5 A</td>
</tr>
<tr>
<td></td>
<td>Min Travel T.</td>
<td>61.4 B</td>
<td>111.2 A</td>
<td>11488 A</td>
<td>2506 A</td>
<td>1415 B</td>
<td>20.5 A</td>
</tr>
<tr>
<td></td>
<td>Max T-Q</td>
<td>61.0 A,B</td>
<td>112.1 B</td>
<td>12155 B</td>
<td>2540 B</td>
<td>1422 C</td>
<td>19.7 B</td>
</tr>
<tr>
<td></td>
<td>Max Trip</td>
<td>62.2 C</td>
<td>113.7 C</td>
<td>12024 B</td>
<td>2585 C</td>
<td>1459 D</td>
<td>19.6 B</td>
</tr>
<tr>
<td></td>
<td>Max Wt. Trip</td>
<td>62.5 C</td>
<td>113.6 C</td>
<td>11473 A</td>
<td>2554 D</td>
<td>1524 E</td>
<td>17.9 C</td>
</tr>
<tr>
<td>Asymmetr. Partially Oversat.</td>
<td>Min Delay</td>
<td>62.6 A</td>
<td>110.8 A</td>
<td>10849 A</td>
<td>2477 A</td>
<td>1312 A</td>
<td>35.6 A</td>
</tr>
<tr>
<td></td>
<td>Min Travel T.</td>
<td>64.5 B</td>
<td>109.9 B</td>
<td>9391 B</td>
<td>2380 B</td>
<td>1276 B</td>
<td>40.8 B</td>
</tr>
<tr>
<td></td>
<td>Max T-Q</td>
<td>70.8 C</td>
<td>120.8 C</td>
<td>11822 C</td>
<td>2573 C</td>
<td>1363 C</td>
<td>21.5 C</td>
</tr>
<tr>
<td></td>
<td>Max Trip</td>
<td>70.1 D</td>
<td>120.8 C</td>
<td>11310 A,C</td>
<td>2694 D</td>
<td>1453 D</td>
<td>23.2 D</td>
</tr>
<tr>
<td></td>
<td>Max Wt. Trip</td>
<td>65.2 E</td>
<td>117.2 D</td>
<td>11198 A</td>
<td>2662 E</td>
<td>1531 E</td>
<td>17.4 E</td>
</tr>
</tbody>
</table>

Min Delay: Objective Function is Delay Minimization
Min Travel T.: Objective Function is Travel Time Minimization
Max T-Q: Objective Function is Throughput-Minus-Queue Maximization
Max Trip: Objective Function is number of completed Trips Maximization
Max Wt. Trip: Objective Function is Number of Weighted Completed Trips Maximization

Among all objective functions, weighted trip maximization resulted in statistically lower total delays in the system (except for symmetric undersaturated demand) compared to the other objective functions. Therefore, for asymmetric undersaturated, symmetric oversaturated, and asymmetric partially oversaturated demand patterns, weighted trip maximization resulted in the most efficient network performance among other objective functions and is used as IDSTOP objective function. In symmetric undersaturated conditions, throughput minus queue, trip, and weighted trip maximization objective functions yielded similar total delays in the entire system. However, the average speed, and
the number of weighted completed trips for weighted trip maximization objective function was statistically significantly more than the other two objective functions. Therefore for symmetric undersaturated demand pattern (similar to the other demand patterns) weighted trip maximization was used as the objective function of IDSTOP.

Overall, weighted trips maximization resulted in shorter delays at the borders, shorter total travel delays in the system, processing higher number of vehicles with longer shortest-path, and faster average speed inside the network. The objective function is formulated as follows:

\[
\text{Maximize } \sum_{\text{SET}} \sum_{\text{SR}} \sum_{\text{ES}} \eta_{rs} x_{rs}^t, \quad \forall t \in T_i
\]  

\( x_{rs}^t = \text{number of completed trips from source node } r \text{ to sink node } s \text{ during time interval } t \)

\( \eta_{rs} = \text{length of the shortest distance path from source node } r \text{ to sink node } s \)

\( T_i = \text{set of discrete time intervals (in the order of minutes)} \)

\( R = \text{set of source nodes} \)

\( S = \text{set of sink nodes} \)

### 3.4 Constraints

If no constraint is used in the problem formulation, the solution may not be desired or feasible. For instance, a solution that creates long queues in the system is not desired while a solution that has a very long cycle length is not feasible. All the constraints are introduced in the rest of this section. It should be noted that some of the constraints should not be avoided in any circumstances. An example for them is the gridlock constraints. A solution should not create a gridlock under any condition. On the other hand, some of the constraints may be violated in certain conditions. An example is de-facto red constraints.
3.4.1 Cycle Length Constraints

In each phase change, some part of the cycle length is wasted due to the yellow and all red signals and also the delays incurs in acceleration and decelerations of the vehicles. This wasted amount of time is called lost time. When the cycle length is short, the phases change more frequently and as a result the lost time is more. Therefore, delay increases and a larger proportion of the green time is wasted. This reduces network capacity and consequently results in less efficient network performance. As a result very short cycle lengths should be avoided. On the other hand, when the cycle length is longer, the phases change less frequently. Therefore, total lost time is less that results in reduction in total delay (compared to shorter cycle length); however, when the cycle length is long, the duration of red is also longer. This means that for a longer period of time the vehicles are not processed. As such, queue lengths may considerably grow. Long queues increase the probability of queue spillovers, de-facto reds, and gridlocks. Therefore, they also should be avoided. In addition, excessively long red signals result in driver frustration.

As a result the cycle length at each intersection at each time interval should be bounded by a lower and an upper bound. This is shown in Equation 3.2 as follows:

\[ C_{min}^i \leq C_t^i \leq C_{max}^i, \quad \forall t \in T_i, \forall i \in I \]  

\( C_t^i \) = cycle length of intersection \( i \) at time interval \( t \)

\( C_{min}^i \) and \( C_{max}^i \) = minimum and maximum allowed cycle length at intersection \( i \) at time interval \( t \), respectively

\( I \) = set of all intersections of the network

It is noted that in this study a minimum value of 40 seconds and a maximum value of 160 seconds were used for cycle length.
3.4.2 Green Time Constraints

Split for green for each phase is the ratio of green time to the cycle length. As a result, the sum of all splits for greens at an intersection in each time period follows Equation 3.3:

$$\sum_{k \in K_i} s_{i,k}^t = 1 - \frac{L_i^t}{C_i^t}, \quad \forall t \in T, \forall i \in I,$$

(3.3)

$s_{i,k}^t =$ split for green associated with phase $k$, at intersection $i$ at time period $t$

$K_i =$ set of all phases available at intersection $i$

Equation 3.3 also indicates that the sum of splits for greens should be equal to the ratio of the effective greens ($C_i^t - L_i^t$) to the total cycle length ($C_i^t$). Green time associated with each phase is obtained by multiplying the associated split for green by the cycle length as shown in the following equation:

$$g_{i,k}^t = s_{i,k}^t C_i^t, \quad \forall t \in T, \forall i \in I, \forall k \in K_i,$$

(3.4)

$g_{i,k}^t =$ green duration for phase $k$, at intersection $i$ at time period $t$

Similar to cycle length, green times should also be bounded since too short and too long green times result in non-efficient network performance. This is shown by the following equation:

$$g_{i,k}^{min} \leq g_{i,k}^t \leq g_{i,k}^{max}, \quad \forall t \in T, \forall i \in I, \forall k \in K_i,$$

(3.5)

$g_{i,k}^{min}$ and $g_{i,k}^{max} =$ minimum and maximum green time associated with phase $k$, at intersection $i$ at time period $t$

It is noted that in this study, a minimum value of 20 seconds and a maximum value of 80 seconds were used for green durations of through movements. For the left turns, these values were 5 and 20 seconds, respectively. Whenever, the algorithm finds left turn green duration less than five seconds, the left turn phase is omitted.
At an isolated intersection, splits for greens of different phases are expected to be proportional to the volume-to-saturated-flow-rate-ratio of the critical movements. In general, this strategy works in a network as well. However, in some cases it may cause extremely long queues, spillovers, and/or possibly gridlocks.

![Figure 3.5. A coordinated arterial](image)

For instance, in the coordinated arterial shown in Figure 3.5, assigning green durations proportional to the volume-to-saturated-flow-rate-ratios of critical movements results in an east-bound through green duration at intersection 2 that is longer than that at intersection 3 while the cycle lengths are the same. Therefore, intersection 2 releases more vehicles than what can be processed at intersection 3. If this setting is maintained long enough, it yields long queues at intersection 3, and in extreme cases, upstream intersection blockage (that should be avoided). As such, in general, setting the green splits identical to the volume-to-saturated-flow-rate-ratio for critical movements of each phase may result in non-efficient network performance. On the other hand, optimizing them completely regardless of volume-to-saturated-flow-rate-ratios of critical movements significantly enlarges the feasibility area. Therefore, optimization algorithms require considerably longer time to find near-optimal splits for green. In addition, if not enough computational resources are available, the algorithms may find sub-optimal splits for greens. Therefore, for each phase, rather than assigning the splits for greens proportional to volume-to-saturated-flow-rate-ratios, or completely regardless of them, an interval centered in that ratio is used in which, splits for greens are optimized. Equation 3.6 formulates these constraints:
\[
\frac{V/Scr_{i,k}^t}{\sum_{v \in K_i} V/Scr_{i,k}^t} - \frac{\delta}{2} \leq \frac{g_{i,k}^t}{\sum_{v \in K_i} g_{i,k}^t} \leq \frac{V/Scr_{i,k}^t}{\sum_{v \in K_i} V/Scr_{i,k}^t} + \frac{\delta}{2}, \quad \forall t \in T_i, \forall i \in I, \forall k \in K_i,
\]

\[
0 \leq \frac{\delta}{2} < \min \left( \frac{V/Scr_{i,k}^t}{\sum_{v \in K_i} V/Scr_{i,k}^t}, 1 - \frac{V/Scr_{i,k}^t}{\sum_{v \in K_i} V/Scr_{i,k}^t} \right)
\]

(3.6)

\[
V/Scr_{i,k}^t = \text{volume-to-saturated-flow-rate-ratio for phase } k, \text{ at intersection } i \text{ at time period } t
\]

If enough amount of computational resource is available, the value of \(\delta\) can be very close to its upper bound, see Equation 3.6. This is equivalent to optimizing the splits for greens regardless of the volume-to-saturated-flow-rate-ratios. In this case it is expected that after enough search, the optimization algorithm finds efficient green splits in each intersection in each time interval (provided that the algorithm can avoid local optimums and find a global optimal or near-optimal solution). However, when the computational resources are limited (in most cases) searching through all possible green split ratios is not efficient. The role of parameter \(\delta\) is to narrow down the search for green split ratios to an interval around volume-to-saturated-flow-rate-ratios for the critical movements. It should be noted that \(\delta\) does not have a unit.

To determine appropriate values for \(\delta\) a series of sensitivity analysis is performed. Different values for parameter \(\delta\) (0 to 0.5 with increments of 0.05) are used and signal timing parameters are optimized in a case study network with four different demand patterns (see Chapter 5 for details on the case study). For all cases, the numbers of fitness function evaluations were identical (22500 fitness function evaluations) to ensure that the same amount of computational resources are consumed. A value of zero for \(\delta\) means that the splits are not optimized by the algorithm and were simply set equal to the volume-to-saturated-flow-rate-ratios for critical movements. A value of 0.1 for \(\delta\) (as an example) means that the green splits are optimized in an interval with length of 0.1 centered on the ratio of volume-to-saturated-flow-rates for critical movements (0.05 to the left and 0.05 to the right of the
Number of completed weighted trips for each value of parameter $\delta$ is shown in Figure 3.6 as well as the observed values for $\delta$ along with their average over 20 intersections of the case study network.

As shown in Figure 3.6 a-d, for all demand patterns, as the specified values for $\delta$ increased, the range of observed values for $\delta$ increased; however, the observed ranges were always smaller than the specified range. In addition, as the specified values for $\delta$ increased the difference between the observed range and specified range enlarged as well (general trend). These two mean that even though the algorithm was allowed to look for green split ratios in a wider range, it yielded green split ratios in a much narrower range indicating that a wide range was not needed. In addition, the wider range resulted
In reductions in the value of objective function meaning that the algorithm did not have enough amount of computational resources to optimize the splits for greens.

In undersaturated conditions (see Figure 3.6 a and c), for all $\delta$ values up to 0.25, the number of weighted completed trips were similar and higher than those for larger values of $\delta$. In addition, for values up to 0.2, the observed ranges for $\delta$ were always at most 0.1. As such, in undersaturated condition, there is no need to provide a large range for $\delta$ to optimize green splits. In fact, setting the green split ratios identical to volume-to-saturated-flow-rate-ratios for critical movements results in solutions as efficient as using a value of up to 0.25 for $\delta$. Therefore, in our case study network, in undersaturated conditions in both symmetric and asymmetric demand patterns, there is no need to optimize the green splits throughout the optimization. Their ratios can simply be equal to volume-to-saturated-flow-rate ratios for critical movements (this was expected) and the extra computational resources can be allocated to optimizing other signal timing parameters.

In symmetric oversaturated demand conditions, $\delta$ values between 0.1 and 0.25 resulted in similar number of weighted completed trips that were higher than those for the rest values for $\delta$. When $\delta = 0.10, 0.15,$ and $0.20$, the observed range for $\delta$ was around 0.1 indicating that there was no need to set the range bigger than 0.1. Not only the observed range was 0.1, a specified range of 0.1 resulted in the most efficient network performance (similar to $\delta = 0.15$ and 0.20). As such, in our case study, in symmetric oversaturated demand conditions, $\delta = 0.10$ should be used which was also expected.

In asymmetric partially oversaturated conditions, $\delta$ values between 0.15 and 0.25 resulted in similar number of completed trips that were higher than those for the rest of $\delta$ values. When specified $\delta$ equaled 0.15, 0.2, and 0.25, the observed $\delta$ were 0.14, 0.19, and 0.19, respectively. Therefore, a range
of 0.2 was needed to optimize signals in our case study network for asymmetric partially oversaturated conditions.

It is noted that findings for values for $\delta$ are network-specific and different values of $\delta$ may yield more efficient performance in other networks. Conducting similar sensitivity analysis is suggested to find the most appropriate values of $\delta$ for other network.

### 3.4.3 Offset Constraints

At two consecutive signals, the offset between two coordinated phases is the time difference between the onsets of green signal for those phases. If the start time of the first phase of each intersection (according to a reference clock), and the duration of the green times is known, the offsets between each two movements can be found by Equation 3.7. Note that this is not a constraint. It is used to find the offsets between two movements based on the green times, start of greens, and the phase sequences. The equation is as follows:

$$
sof g_i^t + \sum_{k=1}^{\varphi_{i-1}} g_{ik}^t + offset_{im,\varphi_{i}\varphi_{m}} = sof g_m^t + \sum_{k=1}^{\varphi_{m-1}} g_{mk}^t, \quad \forall t \in T, \forall i, \forall m \in M_i
$$

- $sof g_i^t$: start of the first green at intersection $i$ at time interval $t$
- $\varphi_m$: number of coordinated phase at intersection $m$
- $\varphi_i$: number of coordinated phase at intersection $i$
- $M_i$: set of all intersections downstream of $i$
- $offset_{im,\varphi_{i}\varphi_{m}}$: offset between the phase $\varphi_i$ of intersection $i$ (upstream), and phase $\varphi_m$ of intersection $m$ (downstream)

The first phase at an intersection may be started when the reference clock is just started $t_0 = 0$ or any time later. However, it cannot be larger than the cycle length of that intersection because it
simply shows that the first green is started \( \text{mod}(sog_i^t, C_i^t) \) seconds later than the start of reference clock. This is shown in Equation 3.8 as follows:

\[
0 \leq sog_i^t \leq C_i^t, \quad \forall t \in T, \forall i \in I
\]  

(3.8)

### 3.4.4 Queue Length Constraints

In oversaturated conditions, due to excessive traffic demand, it is very likely that queues start to grow at the intersections of the network. If the queues are not properly managed, they may block upstream intersections. This reduces the capacity of the intersections and consequently deteriorates network performance. Therefore, the queue length should be controlled to be always shorter than the capacity of the link, or to be more conservative, a proportion of that. This is shown in Equation 3.9.

\[
q_{i,k}^t \leq \omega q_{max_{i,k}}^t, \quad 0 < \omega \leq 1, \forall t \in T_S, \forall i \in I, \forall k \in K_i
\]  

(3.9)

- \( T_S \) = set of discrete time steps (in the order of seconds)
- \( q_{i,k}^t \) = queue length associated with phase \( k \), at intersection \( i \) at time period \( t \)
- \( q_{max_{i,k}}^t \) = maximum allowed queue length associated with phase \( k \), at intersection \( i \) at time period \( t \)

### 3.4.5 Gridlock Constraints

Gridlocks significantly reduce network performance efficiency and consequently, increase total travel time. They have to be always avoided. A gridlock happens when in an immediate loop of several adjacent intersections, in either directions in the loop ("1" denotes clockwise direction, "2" denotes counterclockwise direction), the queue from each downstream intersection blocks the upstream one, see in Figure 3.7. When this occurs, none of the vehicles move and the gridlock may remain effective for a significant amount of time. Therefore, if a solution creates long queues in all intersections of an immediate loop of several intersections along either directions of the loop it must be discarded.
Figure 3.7. Gridlock avoidance

\[ qg^t_{ik} \leq l_{ik}, \quad \forall t \in T_s, \forall i \in B_z, \forall z \in Z, \forall k \in \{1, 2\} \quad (3.10) \]

\( qg^t_{ik} \) = queue length at intersection \( i \) along loop direction \( k \) at time \( t \)

\( l_{ik} \) = length of the link at intersection \( i \) along loop direction \( k \)

\( B_z \) = set of all intersections creating immediate loop \( z \)

\( Z \) = set of all immediate loops of the network

3.4.6 De-Facto Red Constraints

If during a green signal the receiving link is full, no vehicle can be discharged from the intersection to that link. This condition is defined as de-facto red since the signal is actually green but performs as a red signal (due to lack of capacity at receiving link). De-facto red should be avoided since it wastes the green time that could have been allocated to competing phases; however, in some cases it may not be possible to prevent it. For example, assume an intersection whose all receiving links are completely filled with vehicles. In this case, regardless of how green time is allocated, de-facto red occurs. In some cases that it is possible to avoid de-facto red, doing so may not result in a more efficient network performance. For example de-facto red in a minor street should not be eliminated if doing so
significantly deteriorates performance in a major arterial. Since in some certain cases it may not be feasible, or it may not be beneficial to completely avoid de-facto red, we penalize the objective function whenever de-facto red occurs rather than discarding the solution. Equation 3.11 ensures the elimination of de-facto red:

\[
g_{i,\varphi_i}^t \leq g_{m,\varphi_m}^t + of_{im,\varphi_i\varphi_m}^t + \beta_{im}^t = g_{m,\varphi_m}^t + of_{im,\varphi_i\varphi_m}^t + \frac{d_{im} - q_{im,\varphi_m}^t l_{veh}}{\lambda}, \forall t \in T, \forall i \in I, \forall m
\]

\[
\in M_i
\]

\[
\beta_{im}^t = \text{time needed for the stopping shockwave in link } i - m \text{ to reach intersection } i \text{ (upstream) from the end of queue at intersection } m \text{ (downstream) associated with phase } \varphi_m
\]

To avoid de-facto red, the effective green for upstream signal should be less than or equal to the sum of effective green at the downstream intersection, the offset between the two movements, and the time needed for the stopping shock wave to propagate upstream from the end of queue in the receiving link. It should be noted that if the downstream receiving link is already full, the time for the shockwave to reach upstream intersection is zero since the distance between the end of the queue and the upstream intersection is zero.

### 3.4.7 Ideal Offset Constraints

If coordination between two particular movements of two consecutive intersections is desired, the offset for those two movements has to be set equal to the ideal offset. If cars leaving the upstream intersection arrive at the downstream intersection when the tail of the queue at downstream intersection is moving with the speed of arriving vehicles, the offset is ideal. Girianna and Benekolah (2002) have described this concept in details. This constraint may not hold for all of the movements. For example, in a two-way arterial it is not always possible to have the ideal offset for both directions. Thus, IDSTOP uses this constraint only if signal coordination is required over a path. Otherwise, it lets the
optimization engine determine the offsets. These offsets may be ideal or not. The ideal offset constraint is formulated in Equation 3.12 as follows:

\[ o f f_{im,\varphi_m}^t = \tau_{im,\varphi_m}^t - \alpha_{m,\varphi_m}^t = \left( \frac{d_{im} - q_{m,\varphi_m}^t l_{veh}}{v_{im,\varphi_m}^t} \right) - \alpha_{m,\varphi_m}^t l_{veh} \]

\[ = \frac{d_{im}}{v_{im,\varphi_m}^t} - \left( \frac{v_{im,\varphi_m}^t + \lambda}{\lambda} \right) l_{veh} \frac{q_{m,\varphi_m}^t}{\lambda}, \quad \forall t \in T_s, \forall i \in I, \forall m \in M_i \]  

(3.12)

\( d_{im} \) = distance between intersection \( i \) and \( m \)

\( q_{m,\varphi_m}^t \) = queue length associated with phase \( \varphi_m \) at intersection \( m \) at time \( t \)

\( l_{veh} \) = average vehicle length

\( \lambda \) = starting shockwave speed

\( \tau_{im,\varphi_m}^t \) = time required for the first vehicle of the released platoon from intersection \( i \) to join the tail of platoon at intersection \( m \) that is served by phase \( \varphi_m \) at time \( t \)

\( \alpha_{m,\varphi_m}^t \) = time needed for the tail of queue associated with phase \( \varphi_m \) at intersection \( m \) to start moving at time \( t \)

### 3.4.8 Route Delay Constraints

A good solution should result in reasonable travel time in all routes of the network. Still there might be long delays at some intersections but overall travel time should be reasonable. If such a constraint is not used, a large number of vehicles may be processed by the network at the expense of excessively increasing delay at some routes. To avoid such a condition, constraints on travel time over routes of the network should be used not to let extremely long travel times in the network as formulated in Equation 3.13 as follows:

\[ \Delta_{rs,t}^t \leq \Delta \max_{rs,t}^t, \quad \forall t \in T_I, \forall r \in R, \forall s \in S, \forall l \in L_{rs} \]  

(3.13)
\[ \Delta t_{rs,l} = \text{delay on route } l, \text{ connecting source node } r \text{ to sink node } s, \text{ at time } t \]

\[ \Delta \text{max} t_{rs,l} = \text{maximum acceptable delay on route } l, \text{ connecting source node } r \text{ to sink node } s, \text{ at time } t \]

\[ R = \text{set of source nodes} \]

\[ L_{rs} = \text{set of routes connecting source node } r \text{ to sink node } s \]

3.4.9 System Optimum Dynamic Traffic Assignment Constraints

The following constraints are formulated to ensure that origin-destination demand is met while the traffic is assigned to routes such that the number of trips generated in the network is maximized.

The number of vehicles that leave link \( i - j \), at time \( t \) on their way from source node \( r \) to sink node \( s \) \( (x_{rs,ij}^t) \) is equal to that in the previous time interval \( (x_{rs,ij}^{t-1}) \), plus the number of vehicles entered link \( i - j \) from all predecessor nodes during the green signals \( \left( \sum_{k \in P(i)} g_{k \varphi_{ki}}^{t-1} y_{rs,ki}^{t-1} \right) \), minus those who have left link \( i - j \) to the successor nodes during the green intervals at the previous time interval \( \left( \sum_{k \in S(j)} g_{k \varphi_{jk}}^{t-1} y_{rs,jk}^{t-1} \right) \). These constraints are formulated in Equation 3.14 as follows:

\[
x_{rs,ij}^t = x_{rs,ij}^{t-1} + \sum_{k \in P(i)} g_{k \varphi_{ki}}^{t-1} y_{rs,ki}^{t-1} - \sum_{k \in S(j)} g_{k \varphi_{jk}}^{t-1} y_{rs,jk}^{t-1}, \quad \forall t \in T_l, \forall r \in R, \forall s \in S, \forall i, j
\]

\[ \in I/(R, S), \]  \hspace{1cm} (3.14)

\[ g_{k \varphi_{ki}}^{t-1} = 1, \text{ when traffic signal at intersection } k \text{ associated with the phase feeding node } i \text{ shows green signal at time step } t - 1, 0 \text{ otherwise} \]

\[ y_{rs,ki}^{t-1} = \text{number of vehicles that can travel from node } k \text{ to node } i \text{ at time step } t - 1, \text{ associated with source node } r, \text{ and sink node } s \]
It is also needed to make sure that the number of vehicles leaving a link is not more than the number of vehicles that were present in the link. These constraints are formulated in Equation 3.15 as follows:

\[ \sum_{k \in P(i)} G_{ik}^t y_{rs,ik}^t \leq x_{rs,ij}^t \quad \forall t \in T_i, \forall r \in R, \forall s \in S, \forall i, j \in I, \quad (3.15) \]

In addition, the number of vehicles leaving a link cannot be greater than the capacity of the receiving link. Equation 3.16 formulates these constraints:

\[ \sum \sum \sum G_{jk}^t y_{rs,kj}^t \leq N_{ij}^t \quad \forall t \in T_i, \forall i, j \in I/(R,S), \quad (3.16) \]

where \( N_{ij}^t \) is the capacity of link \( i \) to \( j \).

In addition, the number of vehicles leaving a link cannot be more than the discharge capacity of the intersection. These constraints are formulated in Equation 3.17 and 3.18 as follows:

\[ \sum \sum \sum G_{jk}^t y_{rs,kj}^t \leq Q_{ij}^t \quad \forall t \in T_i, \forall j \in I/S, \quad (3.17) \]

\[ \sum \sum \sum G_{ik}^t y_{rs,ik}^t \leq Q_{ij}^t \quad \forall t \in T_i, \forall i \in I/S, \quad (3.18) \]

Demand also has to be forced into the network by the following constraints:

\[ x_{rs,ij}^t = x_{rs,ij}^{t-1} + d_{rs}^{t-1} - y_{rs,ij}^{t-1} \quad \forall t \in T_i, \forall j \in S(r) \cap R, \forall r \in R, \quad (3.19) \]

\[ y_{rs,js}^t = d_{rs}^t \quad \forall t \in T_i, \forall j \in P(s) \cap S, \quad (3.20) \]

Finally it is needed to make sure that the demand from one source node is not met by another source node:

\[ y_{r1s,r2j}^t = 0 \quad \forall t \in T_i, r_1 r_2 \in R, s \in S, r_1 \neq r_2, (r2j) \in E, \quad (3.21) \]
\( E = \text{set of all links} \)

At the end, it is needed to make sure that number of vehicles in the links, and the number of vehicles moving from a link to the other one cannot be negative:

\[
x_{rs,ij}^t \geq 0, \quad \forall \ t \in T, \forall r \in R, \forall s \in S, \forall (ij) \in E, \tag{3.22}
\]

\[
y_{rs,ij}^t \geq 0, \quad \forall \ t \in T, \forall r \in R, \forall s \in S, \forall (ij) \in E, \tag{3.23}
\]

### 3.4.10 Summary of Formulation

In summary, IDSTOP formulation can be represented as follows:

Maximize \[\sum_{v \in V} \sum_{t \in T} \sum_{r \in R} \sum_{s \in S} \eta_{rs} x_{rs}^t, \quad \forall t \in T_i \] (3.1)

s.t.

\[
C_{min}^t \leq C_i^t \leq C_{max}^t, \quad \forall t \in T_i, \forall i \in I
\]

\[
\sum_{v \in V} s_{i,k}^t = 1 - \frac{L_i^t}{C_i^t}, \quad \forall t \in T_i, \forall i \in I
\]

\[
g_{i,k}^t = s_{i,k}^t C_i^t, \quad \forall t \in T_i, \forall i \in I, \forall k \in K_i
\]

\[
g_{min}^t \leq g_{i,k}^t \leq g_{max}^t, \quad \forall t \in T_i, \forall i \in I, \forall k \in K_i
\]

\[
\frac{(1 - \delta) Vcr_{i,k}^t}{\sum_{v \in V} cr_{i,k}^t} \leq \frac{g_{i,k}^t}{\sum_{v \in V} g_{i,k}^t} \leq \frac{(1 + \delta) Vcr_{i,k}^t}{\sum_{v \in V} cr_{i,k}^t}, \quad \forall t \in T_i, \forall i \in I, \forall k \in K_i
\]

\[
sof g_i^t + \sum_{k=1}^{\varphi_{i-1}} g_{ik}^t + of_{f,im,\varphi_{i}}^t = sof g_m^t + \sum_{k=1}^{\varphi_{m-1}} g_{mk}^t, \quad \forall t \in T_i, \forall i \in I, \forall m \in M_i
\]

\[
0 \leq sof g_i^t \leq C_i^t, \quad \forall t \in T_i, \forall i \in I
\]

\[
q_{i,k}^t \leq \omega q_{max}^t, \quad 0 < \omega \leq 1, \forall t \in T, \forall i \in I, \forall k \in K_i
\]
\[ q_{l_{ik}}^t \leq l_{ik}, \quad \forall i \in B_z, \forall z \in Z, \forall k \in \{1,2\} \]  \tag{3.10}

\[ g_{i,\varphi_i}^t \leq g_{m,\varphi_m}^t + \text{off}_{im,\varphi_i}^t \frac{d_{km}^t q_{m,\varphi_m} l_{veh}}{\lambda} + \frac{d_{im}^t q_{m,\varphi_m} l_{veh}}{\lambda}, \quad \forall t \in T, \forall i \in I, \forall m \in M_i \]  \tag{3.11}

\[ \text{off}_{im,\varphi_i}^t = \left( \frac{d_{im}^t q_{m,\varphi_m} l_{veh}}{\lambda} \right) - \left( \frac{q_{m,\varphi_m}^t l_{veh}}{\lambda} \right) \]  \tag{3.12}

\[ \Delta_{rs,i}^t \leq \Delta_{\max}^t, \quad \forall t \in T, \forall r \in R, \forall s \in S, \forall i \in L_{rs} \]  \tag{3.13}

\[ x_{rs,ij}^t = x_{rs,ij}^{t-1} + \sum_{k \in P(i)} G_{k,\varphi_{ki}}^t y_{rs,kl}^{t-1} - \sum_{k \in S(j)} G_{k,\varphi_{jk}}^t y_{rs,jk}^{t-1}, \quad \forall t \in T, \forall r \in R, \forall s \in S, \forall i, j \]  \tag{3.14}

\[ \sum_{k \in P(i)} G_{k,\varphi_{ki}}^t y_{rs,ik}^t \leq x_{rs,ij}^t, \quad \forall t \in T, \forall r \in R, \forall s \in S, \forall i, j \in I, \]  \tag{3.15}

\[ \sum_{r \in R} \sum_{s \in S} x_{rs,ij}^t \leq N_{ij}^t - \sum_{r \in R} \sum_{s \in S} x_{rs,ij}^t, \quad \forall t \in T, \forall i, j \in I/(R,S), \]  \tag{3.16}

\[ \sum_{r \in R} \sum_{s \in S} G_{j,\varphi_{js}}^t y_{rs,kj}^t \leq Q_{ij}^t, \quad \forall t \in T, \forall j \in I/S, \]  \tag{3.17}

\[ \sum_{r \in R} \sum_{s \in S} G_{k,\varphi_{ks}}^t y_{rs,ik}^t \leq Q_{ij}^t, \quad \forall t \in T, \forall i \in I/S, \]  \tag{3.18}

\[ x_{rs,j}^t = x_{rs,j}^{t-1} + d_{rs,j}^{t-1} - y_{rs,j}^{t-1}, \quad \forall t \in T, \forall j \in S(r), \forall r \in R, \]  \tag{3.19}

\[ y_{rs,js}^t = d_{rs,s}^t, \quad \forall t \in T, \forall j \in P(s), \forall s \in S, \]  \tag{3.20}

\[ y_{r1,r2}^t = 0, \quad \forall t \in T, \forall r, \forall s \in S, \forall r, r_1 \neq r, (r2) \in E, \]  \tag{3.21}

\[ x_{rs,ij}^t \geq 0, \quad \forall t \in T, \forall r \in R, \forall s \in S, \forall (ij) \in E, \]  \tag{3.22}
\begin{equation}
\gamma_{rs,ij}^{t} \geq 0, \quad \forall \ t \in T_i, \forall r \in R, \forall s \in S, \forall (ij) \in E,
\end{equation}

\section{3.5 References}


CHAPTER 4

IDSTOP SOLUTION TECHNIQUE

4.1 Introduction

There are two main complexities associated with solving IDSTOP. First, the decision space of the problem is extremely large. Second, there is no closed-form formulation to represent the value of the IDSTOP’s objective function in terms of its decision variables.

IDSTOP’s solution space follows a power relationship with the number of intersections. If there are \( x \) different possible decisions for each intersection at one time interval, solution space has \( x^n \) components for \( n \) intersections. It is noted that \( x \) can be as large as \( 1.8 \times 10^8 \) for one intersection with four phases for a single time interval. This number is obtained by multiplying the total number of possible values each decision variable can take. A minimum of 15 seconds and a maximum of 80 seconds for through traffic green signal duration (total of \( 80 - 15 + 1 = 66 \) decision for each direction), a minimum of 7 seconds and a maximum of 20 seconds for left turn arrow green signal duration (total of \( 20 - 7 + 1 = 14 \) decision for each direction), and a minimum of zero and a maximum of 214 seconds for offset (total of \( 214 - 0 + 1 = 215 \) decision for the offset), results in \( 66 \times 66 \times 14 \times 14 \times 215 = 1.8 \times 10^8 \) decisions. Due to this extremely large solution space, traditional methods such as exhaustive search or dynamic programming will not lead to a near-optimal solution in a reasonable amount of time even when the fitness function evaluations requires a fraction of a second.

In addition, IDSTOP is a nonlinear non-convex optimization problem without a closed-form formulation to represent its objective function in terms of the decision variables. Therefore, none of the methods that rely on knowing detailed relations between the decision variables and the objective function such as deepest descent can be used.
These two together, extremely complicate the process of solving IDSTOP. The reason is that there is no information available on the structure and the behavior of the objective function in terms of decision variables that could be used to facilitate finding an optimal solution. In addition, the large size of the decision space makes it impossible to exhaustively search the space.

All these, limit the optimization techniques to those that fall in the category of heuristic (i.e. methods that aim at finding a feasible solution) and meta-heuristic methods (i.e. methods that optimize a problem by iteratively trying to improve a candidate solutions with respect to a measure of quality) among them, two families of evolutionary algorithms are chosen:

a) Genetic Algorithms (GA)
b) Evolution Strategies (ES)

Evolutionary Algorithms are population based meta-heuristic optimizations that utilize biology-inspired operators such as mutation, crossover, selection, and survival of the fittest to improve the quality of a set of solutions. One important advantage of evolutionary algorithms compared to other optimization algorithms is their so called “black box” feature that enables them to carry out the optimization process without knowing exact structure of the objective function based on the decision variables. IDSTOP takes full advantage of this feature of evolutionary algorithms. In fact, evolutionary algorithms only need to know the value of objective function for a set of decision variables but not any more information. This makes them a suitable pick for solving IDSTOP since it is possible to (accurately enough) estimate the value of the objective function for a candidate solution. This is can be done by using a traffic simulation model.

Genetic Algorithms and specifically simple GA have been extensively used to optimize signal timing in urban networks (e.g. [1], [2], [3], [4], [5]). In this study, several variations of genetic algorithm will be used to solve IDSTOP. These variations are as follows:
a) Simple GA (as a benchmark)
b) Elitist GA
c) Micro-Elitist GA

Among different Evolution Strategy methods two of them that are widely accepted in other fields of science are selected:

a) Self-adaptive ES
b) Self-adaptive elitist ES

In the rest of this chapter, each method is briefly described. The IDSTOP structure is explained. Later both signal timing optimization and dynamic traffic assignment modules are explained. The discussion is followed by explaining how the constraints are taken into account and finally a summary of the chapter is presented.

4.2 Genetic Algorithms

Genetic Algorithms (GA) are search techniques to find optimal or near-optimal solutions to an optimization or a search problem. GA are global search heuristics and are known to be less likely trapped in a local optimum. GA are a specific class of evolutionary algorithms and use techniques inspired by evolutionary biology like inheritance, selection, crossover, and mutation.

GA are implemented in a computer simulation environment where a population of candidate solutions are created and evolved towards better solutions over different generations. Unlike other well-known optimization techniques that start the search with one feasible solution, GA start the search with several candidate solutions, called population. The initial population can be created randomly or by using some heuristics. Each population member is called an individual or a chromosome, and has a fitness value that represents the value of the objective function for that individual. For example, if the
objective function is to maximize $f(x) = x^2$, the fitness of one of the individuals e.g. $x = 3$ will be $3^2 = 9$. Based on the fitness values, GA stochastically select some individuals of the population where individuals with higher fitness values are more likely to be selected (for a maximization problem). The selected individuals form a mating pool where they are crossed over and mutated to form some new individuals for the new population in the next generation. GA continue to select new individuals as parents until enough individuals for the next generation are created. As soon as a new individual is created its fitness value is evaluated. It is noted that in this study, the feasibility of that individual is checked before determining its fitness value (details available in chapter 5). The whole process of selection, crossover, and mutation is continued until the termination criteria are met. Usually a maximum number of generations, or a threshold for the relative difference between the maximum fitness value and average fitness value of a population are chosen as the termination criterion.

Traditionally, binary coding was used to represent each feasible solution in GA; however, other methods of coding exist such as real-coding. In binary coding each 0 or 1 of the chromosome is called a genome. Several variations of GA exist. In this study three of them are used to solve IDSTOP: a) simple GA, b) Elitist GA, and c) Micro-Elitist GA. Comprehensive details on GA can be found in Goldberg (1989) [6].

4.3 Evolution Strategies (ES)

Evolution Strategies (ES), genetic algorithms, and evolutionary programming are the main three paradigms of Evolutionary Computation (EC). In general, these three methods are based on iterative birth and death, variation, and selection. The first ES had only two rules: 1) slightly change all variables at a time at random, 2) if this set of variables leads to better results keep them otherwise, keep the original ones. As it is apparent from the rules, this ES worked with only two individuals per iteration: one old individual or parent, and one new individual or offspring. This ES was later called 1+1-ES meaning
that out of a single parent, one offspring is generated and among these two individuals, the best is chosen. The 1+1-ES with binomially distributed mutations on a two dimensional parabolic ridge was studied by Schwefel in 1965 [7]. The study showed that 1+1-ES is very likely to find a local optimal answer rather than a global one. In this case, larger mutations were needed to escape from this local optimum. To solve this problem, instead of using discrete variables, using continuous variable with Gaussian distributions was suggested. Rechenberg presented approximate analyses of the 1 + 1 − ES with Gaussian mutation on two different functions (hyper sphere, and rectangular corridor models). He found that the convergence was inversely proportional to the number of variables; linear convergence might be obtained if the mutation step size was set to the proper order of magnitude; and the optimal mutation strength was in the order of one fifth for both models. In addition, instead of using a single parent, he used μ parents, crossed them over, and generated one offspring. He concluded that this method could speed up the evolution if the speed was measured per generation; and the population might learn by itself how to adjust the mutation step size. This method of ES was called μ + 1 − ES since among μ + 1 individuals the best μ individuals were selected or in other words, the worst individual is extinct. Later, μ + 1 − ES was expanded to μ + λ − ES. In this method instead of creating a single offspring out of the μ parents, λ descendants are created. Then among these μ + λ individuals the fittest μ individuals are selected to form the next population. Another variation of ES with μ > 1 parents and λ > 1 descendants exists. In this method, after creating the new λ descendants, all parents are discarded. Out of the λ descendants, the fittest μ are chosen to form the next population. Thus, λ has to be strictly larger than μ. This method is called μ, λ − ES. In general, μ + λ − ES and μ, λ − ES generate better results than 1 + 1 − ES and μ + 1 − ES do. Although intuitively it is believed that μ + λ − ES generates better results than μ, λ − ES does, for small μ and λ − to − μ ratio, μ, λ − ES generates better results. When μ and λ − to − μ ratio increase, both algorithms perform similarly.
All variations of ES with \( \mu > 1 \) parents and \( \lambda > 1 \) descendants have three different operators that are recombination, mutation, and selection. ES has the following steps:

0) Initialization: the first population is generated randomly or by means of some heuristics

1) Regeneration: next population is produced

   1-1) Recombination: randomly select \( \rho \) parents and recombine them to generate a new offspring

   1-2) Mutation: mutate the new offspring

   1-3) Fitness function evaluation: evaluate the fitness of the generated offspring

2) Selection: select new parents with respect to “+” or “,” scenario

3) Termination criteria: stop if termination criteria are met otherwise continue by going to step 1

ES could be self-adaptive. This means that as the populations evolve, the strategy parameters evolve as well. This is done by coupling endogenous strategy parameters with the objective parameters. In other words, the decision vector contains decision variables as well as endogenous strategy parameters. This is shown in Equation 4.1.

\[
\vec{\alpha}_k = (y_{1k}, y_{2k}, ..., y_{Dk}, s_{1k}, s_{2k}, ..., s_{Dk})
\]  

(4.1)

Where: \( y_{ij} \): the \( i^{th} \) component of decision variable \( j \), and

\( s_{ij} \): the \( i^{th} \) component of endogenous strategy parameter \( j \).

More information on ES could be found in Schwefel (1965).

### 4.4 IDSTOP Architecture

IDSTOP, as mentioned before, is formulated as a signal timing optimization program that dynamically finds signal timing parameters (i.e. phase plan, cycle length, splits, and the offsets) for an
urban traffic network over the study period. It also dynamically reroutes drivers to less congested routes to further increase the number of completed trips by using its system optimal traffic assignment feature.

IDSTOP considers stochasticities that are involved in the car following and lane changing behavior as well as vehicle arrival to the network. Details on how it accounts for them are explained later in this chapter. IDSTOP takes into account stochastic behavior of drivers in car following, in speeding up after a red signal turns green, and in slowing down to stop before a red signal, and also in lane changing. In addition, it considers different types of vehicles in the network that significantly changes acceleration and deceleration behavior. Also, it takes into account combinations of different drivers and vehicles that bring more stochasticity into the problem. IDSTOP also considers different distributions for vehicle arrival to the network and unlike deterministic models, does not assume that vehicles keep constant headways from each other, have identical acceleration and deceleration rates, and drivers behave identically in accelerating, decelerating, and deciding to stop or to proceed for a yellow signal, or join the back of queue when the receiving links is almost full. Modeling all these stochasticities makes the solution technique extremely more complicated but, enables IDSTOP to find solutions that more accurately depict what happens in the real world. For example, based on the constraints of the cell transmission model, it is assumed that no vehicle joins the back of queue when the receiving links is full; however, this does not happen in the real world as one driver may join the back of queue and one may not. If cell transmission based solutions are used in real-world application, and a driver decides to join the back of queue in a link which is already filled with vehicles, upstream signal may be blocked and gridlock may happen while IDSTOP finds a solution that prevents them. To handle these stochasticities, IDSTOP runs microscopic simulation model with several replications with different parameters to account for different scenarios that may occur in real-world conditions. This will be discussed later in this chapter. However, it should be noted that IDSTOP is not designed to handle neither the uncertainties associated with origin-destination (o-d) demand (it needs to know the demand
for the upcoming time interval up front) nor some other stochastic events such as accident occurrence, vehicle break down, or traffic signal failure.

The main idea to solve the problem is to discretize the time period into smaller time intervals and optimize signal timing and turning percentages in each interval. There are three main reasons for this: 1) it significantly reduces the complexity of the problem; 2) it results in more efficient network performance; 3) if the study period is long enough, eventually all vehicles will complete their trips.

The decision space is significantly smaller when the problem is solved sequentially. In fact, instead of being the combination of the possible decision spaces of all time intervals, it is the summation of the decision spaces of all time intervals. Therefore, it is computationally less expensive to solve the problem.

IDSTOP’s objective function (as described in Chapter 3 Equation 3.1) may result in keeping vehicles in the network during one time interval and releasing them in another one. For instance assume a study period of ten minutes with two 5-minute time intervals with a uniform traffic demand of 50 vehicles per each time interval. The optimal policy that results in lowest delay and best network performance is to process 50 vehicles in each time interval. In that case, the total number of completed trips in the entire study period is 100 vehicles which is the maximum possible. If the objective function is to maximize the sum of completed trips in both time intervals together, processing 20 vehicles in the first and 80 in the second time interval (as well as any combination of two numbers that summing up to a hundred) is also a valid solution. However, it is not as efficient as processing 50 vehicles in each time interval, is not desired, and should be avoided. In fact, it is preferred to maximize the number of completed trips in each time interval rather than in the whole study period. In other word, instead of solving a single non-linear problem for the entire study period, solve one non-linear problem for each
time interval is more desired. In that case, \(|T_i|\)-many non-linear problems are solved. The new objective function is shown in Equation 4.2 as follows:

Maximize \( \sum_{r \in R} \sum_{s \in S} \eta_{rs} x_{rs}^t \), \( \forall t \in T_i \) \hspace{1cm} (4.2)

\( x_{rs}^t = \) number of completed trips from source node \( r \) to sink node \( s \) during time interval \( t \)

\( \eta_{rs} = \) length of the shortest distance path from source node \( r \) to sink node \( s \)

\( T_i = \) set of discrete time intervals (in the order of minutes)

\( R = \) set of source nodes

\( S = \) set of sink nodes

The time intervals are selected such that the origin-destination demand in each is approximately constant. Based on the constant o-d demand in each time interval, fixed signal timing parameters and system optimum traffic assignment for the network is found. It is noted that the signal timing is fixed for an intersection in a time interval, but changes from one intersection to another within the same time interval. In addition, for each intersection, signal timing parameters change from one time interval to another in response to time-variant demand. For each o-d pair, the routes are fixed for each time interval, but they change from one time interval to another. Routes are assigned to vehicles when they enter the system and they are not allowed to change their routes at different intersections (since doing so significantly enlarges IDSTOP’s solution space). The final state of time interval \( t \) is used as the initial state of the time interval \( t + 1 \). The state of the system at a time is location (longitudinal and lateral), speed, and acceleration/deceleration rates of all vehicles in the network as well as the state of the signal at each approach of each intersection.

A meta-heuristic algorithm is developed to find near-optimal signal timing parameters and system optimum traffic assignment in each time interval. As mentioned earlier, the state of the system
at the end of current time interval is carried over as the initial state of the system in the next time interval. The following steps are taken to simultaneously optimize signal timing and drivers’ routes inside the network:

Step 0) Initialization:

a) a set of feasible candidate solutions are generated either randomly or by using some heuristics
b) the fitness of each individual is evaluated using microscopic simulation model
c) system optimum traffic assignment is performed for the fittest individual,
d) link volumes and turning percentages are updated if in step 0-c fitness value was improved

Step 1) Regeneration:

a) selection: parents are selected
b) regeneration: new individuals are created using the selected parents assuming the link and turning volumes obtained in the previous generation
c) evaluation: fitness function is evaluated for each new individual

Step 2) System Optimum Traffic assignment:

d) traffic is assigned for the fittest individual created in Step 1, and link and turning volumes are obtained
e) link volumes and turning percentages are updated if in step 2-a fitness value was improved

Step 3) termination criteria: if termination criteria are met stop; otherwise go to step 1.

As it is presented in the algorithm, signal timing optimization and system optimum traffic assignment are not found in combination to each other. Finding them in combination to each other significantly enlarges the solution space and makes finding a near-optimal solution almost impossible
unless a deterministic approach is used to provide enough information about the structure of the objective function (see [8]). Instead of optimizing them in combination to each other, a semi-sequential method is used. In each generation of the proposed algorithm, first the signal timing parameters are improved for all intersection for the current time interval. Then for the best solution available, system optimal traffic assignment is performed to optimize link and turning volumes. If traffic assignment improves the fitness value, new link and turning volumes are used for the next step of signal timing. Otherwise old link and turning volumes are used in the next step. This new solution is added to the individuals that were created in the most recent generation while the link and turning volumes for all individuals are updated. Then in the next generation, similarly, first the signal timing and then vehicle routes are optimized. This algorithm is shown in Figure 4.1.

The proposed algorithm has the following advantages:

a) does not need to know the structure of the objective function to find a solution
b) extremely smaller decision space
c) flexibility to optimize different objective function
d) flexibility to use different forms of evolutionary algorithms
e) flexibility to use different microscopic simulation models to obtain the fitness of each individual
Initialization:
Generate the first population according to feasibility area
For all chromosomes, compute the fitness value: $\sum_{s \in S} \sum_{r \in R} \eta_r x_s^T$

Assign traffic for best individual and obtain Fitness in CORSIM

Fitness improved?

Yes
Update turning percentages for the next generation

No
Regeneration:
Generate new solution according to the feasibility area:
Selection, Xover, Mutation

Obtain fitness value from CORSIM

Fitness improved?

Yes
Update turning percentages for the next generation

No
Increase individual count

Enough individuals?

Yes
Assign traffic for best individual and obtain fitness in CORSIM

Fitness improved?

Yes
Update turning percentages for the next generation

No
Increase generation count

Termination criteria?

Yes
Output the Best Solution

No
Increase individual count

Enough individuals?

Yes
4.4.1 Signal Timing Optimization

The discussion in this section is centered on the following four items in signal timing optimization process:

1) Traffic flow propagation
2) Constraints satisfaction
3) Decision variables
4) Accounting for stochasticities

4.4.1.1 Traffic flow propagation

As previously mentioned, one of the objectives of this study is to develop a signal timing optimization method that considers stochasticities associated with traffic flow propagation such as: different driver behavior, different vehicles types, different headway distributions, etc. In order to have the capability to take them into account, IDSTOP has to be able to model them to begin with. As a result, IDSTOP simulation model has to be able to model different headway distributions, different driver behavior (in car following, acceleration, deceleration, lane changes, joining back of queue, etc.), different vehicle types, etc. Macroscopic and mesoscopic traffic simulation models are not capable of modeling all these stochasticities. On the other hand, microscopic traffic simulation models are capable of modeling these stochastic events.

Among the most widely used microscopic traffic simulation packages, two of them were tested in this study. These two were CORSIM developed by Federal Highway Administration, and VISSIM developed by Planung Transport Verkehr AG in Germany. Both packages were capable of modeling different network, traffic, and geometric conditions as well as modeling different driver behaviors, vehicles characteristics, and entry headways. It is noted that CORSIM can model all details that is needed in this study and is considerably faster (in terms of runtime) than VISSIM. Therefore, it is
selected to propagate traffic flow inside the network. There was an option of developing a new microscopic model for the purpose of this study. However, since the focus of the study was on the optimization part, the widely accepted CORSIM was used.

4.4.1.2 Constraints Satisfaction

As previously mentioned, signal timing optimization process starts with generating a population of potential solutions that are produced either randomly or by using some heuristics (e.g. optimal solution of commercial software). These solutions are created such that they satisfy the constraints on the minimum and maximum values of the decision variables. However, making sure that the rest of the constraints are satisfied requires complicated calculations. Eventually, all constraints are checked during the microscopic simulation run when the fitness value of each solution is obtained. If a solution does not satisfy any of the constraints it will be discarded. However, since running a microscopic simulation model requires a significant amount of CPU time, it is extremely important to identify the infeasible solutions before running the microscopic model. Thus, using a less computationally expensive model (i.e. a macroscopic or a mesoscopic model), the infeasible solutions need to be identified. It should be noted that it is still possible that some of the solutions that were identified as feasible using the faster model, be infeasible when microscopic model is used. This may happen due to all simplifying assumptions that exist in the fast (macroscopic) model. As a result, even when a faster model for pre-scrutiny is used, all constraints will be rechecked when the microscopic model is running. For the pre-scrutiny purpose a macroscopic model developed by Girianna and Benekohal (2002) is used. This model uses shockwave theory to find the queue length in each link over time [3]. Using this model, queue length constraints, de-facto red constraints, ideal offset constraints, and gridlock constraints are checked. If any of the solutions does not satisfy any of the constraints, the solution is discarded and a new solution is created. This pre-scrutiny part is continued until enough individuals are created. This step is used in initialization step as well as the regeneration step. For each individual that satisfied all the
constraints in this section, microscopic model is called. Delay constraints in addition to all other constraints are checked and if any constraint is violated, the solution is discarded. If the solution is identified as feasible, its fitness value is obtained.

4.4.1.3 Signal Timing optimization Decision Variables

The parameters that are associated with signal timing optimization are phase plan, cycle length, green splits, and the offsets for all intersection in each time interval. IDSTOP optimizes cycle length, green splits, and the offsets. The phase sequence is optimized based on the optimized green splits. This means that if IDSTOP allocates a green duration of less than five second to a left turn movement, that phase is omitted. Through movement phases are never omitted. IDSTOP allows a maximum of four phases per cycle with the widely known Lead-Lead Left-Turn Phase Sequence: lead-lead left turn green signal for direction one, through traffic green signal for direction one, lead-lead left turn green signal for direction two, and through traffic green signal for direction two. As a result, the number of phases varies between a minimum of two (when both left turns were omitted) and a maximum of four phases (when none of the left turns were omitted). This sequence is shown in ring format if Figure 4.2.


Figure 4.2. IDSTOP phase sequence

IDSTOP’s decision vector for signal timing optimization for each intersection consists of five components as follows:

1. Cycle length
2. Green split for phase one (left turn movement for direction one)
3. Green split for phase two (through movement for direction one)

4. Green split for phase three (left turn movement for direction two)

5. The start time of the green of the first phase according to a time reference point

It should be noted that the green split for phase four is found based on the splits of the other three phases, the cycle length, and the lost time as follows:

\[
s_{i,4}^t = 1 - \frac{L_i^t}{C_i^t} - \sum_{k=1,2,3} s_{i,k}^t, \quad \forall t \in T_i, \forall i \in I,
\]  \hspace{1cm} (4.3)

Where:

\(s_{i,k}^t\) = green split associated with phase \(k\), at intersection \(i\) at time period \(t\)

\(L_i^t\) = lost time at intersection \(i\) at time period \(t\)

\(C_i^t\) = cycle length of intersection \(i\) at time interval \(t\)

\(T_i\) = set of discrete time intervals

\(I\) = set of all intersections of the network

The decision vector for all intersections consists of all decision vectors for each intersection of the network followed by each other as follows:

\[
\vec{q}^t = (C_1^t, s_{1,1}^t, s_{1,2}^t, s_{1,3}^t, sof g_1^t, C_2^t, s_{2,1}^t, s_{2,2}^t, s_{2,3}^t, sof g_2^t, \ldots , C_{|I|}^t, s_{|I|,1}^t, s_{|I|,2}^t, s_{|I|,3}^t, sof g_{|I|}^t )
\]  \hspace{1cm} (4.4)

Where:

\(\vec{q}^t\) = IDSTOP signal timing decision variable at time interval \(t\)

\(sof g_i^t\) = start of the first green at intersection \(i\) at time interval \(t\)
4.4.1.4 Taking Stochasticities into Account

In order to be able to model stochastic events (e.g. driver behavior in acceleration, deceleration, lane change, and joining back of queue in an almost full link; and vehicle arrival headway to the network) in traffic, a microscopic simulation model was needed. Each run of this model simulates a certain set of stochastic events that occurred in the network. If the optimization is carried based on a single simulation run, the optimal solution provides the best performance for a real-world network only if real drivers always behave identical to those in the simulated network, headway between vehicles are identical to the simulated network, and similarly all other parameters are identical. However, this is not likely to happen. Instead of finding such a solution, IDSTOP finds a solution that provides an efficient network performance under different driver behaviors, vehicle headways, etc. This is achieved by making several simulation runs for a candidate solution and finding the fitness value by averaging the fitness value for each run. It is noted that a certain seed for each run needs to be used which has to be different than the other seeds that are used in the other runs. This is to avoid creating identical conditions. To find the fitness value of each individual a total of ten runs are made and the average fitness value is obtained. Details on finding the number of replication are available in chapter 5. It is noted that if any of the constraints are violated in any of these replications, the solution is discarded and a new solution is created.

4.4.2 System Optimum Traffic Assignment

The main objective of this research is to develop dynamic stochastic signal timing optimization algorithms for urban traffic network with oversaturated intersections. Optimizing transportation supply and demand together has potential to further improve network performance. Sun and Benekohal (2004), and Abdul Aziz and Ukkusuri (2011) developed algorithms for managing transportation demand and supply at the same time based on deterministic models to move vehicles inside the network [4] [8]. In this research simultaneous demand and supply management is performed to further improve
network performance. However, since the main focus of the study is to find optimal or near-optimal network performance, system optimum traffic assignment is used. Although system optimal flows are not likely to be observed in the real-world conditions (since drivers choose their routes in order to minimize their own travel cost rather than a total system cost), still knowing the best performance possible can be helpful in making decisions. For example, the traffic flows can be used in network design or can be sought by introducing tolls on different links of the network to match user equilibrium and system optimal flows. Two different System Optimum Dynamic Traffic Assignment (SODTA) methods are considered. These methods are:

a) CORSIM’s system optimum traffic assignment

b) Cell Transmission based SODTA developed by Li, Ziliaskopoulos, and Travis Waller (1999) [9]

As mentioned previously, the study period is discretized with respect to o-d demand such that in each time interval o-d demand variations are negligible. In each time interval, a static system optimum traffic assignment is performed except for method b where traffic assignment is dynamic within each time interval as well. Each method is explained next.

4.4.2.1 CORSIM System Optimum Traffic Assignment

The least computationally expensive method was the one implemented in CORSIM. This method used Frank-Wolf algorithm to find user optimal traffic assignment. Travel costs were estimated based on using so-called BPR equations. The parameters of the BPR function, and number of iterations, as well as the o-d demand were the inputs to the traffic assignment module. CORSIM’s default values for BPR function parameters were used ($a = 0.6, b = 4.0$) and the number of iterations was set to the maximum possible of 20 iterations.

The most important benefit of this model was its extremely short runtime. However, due to its oversimplified method its solution may not always result in an improvement in the value of the
objective function. The main reason is that its solution is aimed at reducing the travel time based on BPR equations; however, these equations cannot accurately determine the travel time, and do not directly take intersection delay into account. Thus, the solution of this approach, although finds shorter total system travel time based on BPR equations, may not always reduce travel time and may not always increase number of completed trips in CORSIM. If traffic assignment does not increase the number of completed trips, the new link and turning volumes are discarded and the old ones are used.

4.4.2.2 Cell-Transmission based SODTA

The cell transmission based DTA, as used by Li et al., models SODTA as a linear programming that can be solved using different solvers. This method requires about two hours to find optimal solution for a network of 20 intersections and a study period of 15 minutes which is significantly longer than previous approach and makes this approach less suitable for IDSTOP. Assuming that IDSTOP has a total of 30 generations, total runtime for DTA will be around 60 hours. This method is capable of finding optimal solution in the network; however, has some limitations. First, it does not model any of the stochastic effects that IDSTOP is designed to take them into account such as different vehicle types, different drivers, and non-constant headway. Second, it cannot consider more than one lane for each street. Third, it cannot take permissive left-turns into account. Finally, since it uses a different logic to move the vehicles than CORSIM, even if calibrated, there is no guaranty the its optimal solution results in less travel time and higher number of completed trips when the solution is used in CORSIM. Adding all these to its long runtime makes the algorithm less-suitable for IDSTOP.

4.5 Summary

In this chapter IDSTOP solution technique was explained. The main idea to solve the problem and account for the known time-variant demand was to discretize the study period to shorter time interval in which o-d demand is approximately constants. Then near-optimal signal timing parameters as
well as system optimal traffic assignment were found in each time interval. Two families of metaheuristic approaches were explained and the reason of choosing a microscopic traffic simulation model was discussed. Finally, taking the stochasticities into account was explained and two different traffic assignment methods were discussed. In the following chapter, IDSTOP implementation, verification, and validation, details on how the method is implemented, how constraints were checked, how the objective function was evaluated, and how the algorithm was verified and validated will be discussed.

4.6 References


CHAPTER 5

IDSTOP IMPLEMENTATION AND PERFORMANCE

5.1 Introduction

In this chapter, details on IDSTOP implementation are discussed. The explanation starts with introducing the case study network, the demand patterns, how IDSTOP is coded, how the constraints are checked, and how the fitness function evaluation is performed to account for stochasticities. Upper bounds on the number of completed trips for all demand patterns are determined; and discussion is continued with explaining the performance of IDSTOP and its comparison to a state-of-the practice signal timing optimization package.

5.2 Case Study Network

IDSTOP was tested using several case study networks. All results presented in this chapter are based on a realistic case study network that was adopted from downtown Springfield in Illinois. The main idea was to test IDSTOP under a more diverse set of conditions, closer to real world operations. The case study network has 20 intersections and a combination of one-way and two-way streets with different number of lanes. It comprised the area between 5th and 11th street from west to east, and between Jefferson and Capitol streets from north to south in Springfield, Illinois.

A few modifications were made to the real network in Springfield because of the higher vehicular demand used in the test case compared to the actual demand in the field: 1) most of the left-turn lanes in the network were shared by through movement, but this was changed by adding exclusive left-turn pockets, 120ft in length; and 2) if there was a lane drop or a lane addition on an arterial, the model maintained the same number of lanes along the arterial. The test network is called modified Springfield network, and is shown in Figure 5.1.
In this area of downtown Springfield, actual traffic volumes are not high enough to create oversaturated conditions. Since, we are interested in finding solutions for oversaturated conditions traffic volumes at different links of the case study network were increased. In addition, all traffic signals in the portion of downtown Springfield use only two phases. In the case study, the possibility of having up to four phases was considered.

Figure 5.1: Modified Springfield network

As mentioned before, at each intersection a minimum of two phases and a maximum of four phases are allowed. When two one-way streets intersect, only two phases can be used. This was the case for intersections number 1, 2, 5, 6, 9, and 10 in the modified Springfield network, see in Figure 5.1. When a one-way and a two-way street intersect, the number of phases is optimized by IDSTOP and it
can be either two or three phases. This was the case for intersections number 3, 4, 7, 8, 13, 14, 17, and 18 in Figure 5.1. Finally, when two two-way streets intersect, the number of phases can be two, three, or four which is optimized by IDSTOP. This was the case for intersections number 15, 16, 19, and 20, see in Figure 5.1.

5.3 Demand Patterns

Four different fixed-demand traffic pattern cases were used on the modified Springfield network:

*Case-a*) Undersaturated network with symmetric traffic demand (750 vphpl in each entry links)

*Case-b*) Oversaturated network with symmetric traffic demand (1000 vphpl in each entry links)

*Case-c*) Undersaturated network with asymmetric traffic demand, high volume in east-west streets (1000 vphpl), low volume in north-south streets (500 vphpl)

*Case-d*) Partially oversaturated network with asymmetric traffic demand, 1000 vphpl in corridors P-G and B-L7; 700 vphpl corridors A-M, R-E, and F-Q; 600 vphpl in corridors O-H and N-I; and 500 vphpl in corridors C-K and D-J (see Figure 5.1).

For each case, it is assumed that whenever possible, 10% of the traffic in the right-most lane makes a right turn, 10% of traffic in the left-most lane makes a left turn, and the remaining vehicles go straight. For example, for a single lane street with possible left and right turns, 10% of traffic turns left, 10% turns right and 80% goes straight. For a two-lane street with possible right and left turns, 5% of the total incoming traffic turns right, 5% turns left, and 90% goes straight. It is noted that whenever a left or right turn is not possible the turning vehicles go straight. These turning percentages are only used when traffic is not assigned. When traffic assignment feature is on, the o-d demand is needed and is estimated based on the turning percentages mentioned above. This is done to make sure the same test bed is used
for future comparisons (traffic assignment versus no traffic assignment). To estimate the o-d for each of
the four demand cases, all routes from each origin to each destination needed to be found. For each
route, based on the traffic demand and turning percentages, the number of vehicles that reached each
destination node was calculated. Therefore, the o-d demand was found for all o-d pairs.

In addition to the fixed-demand traffic pattern, a dynamic-demand traffic patterns is used. For
this case traffic demand gradually changes in increments of five minutes from symmetric
undersaturated to symmetric oversaturated, asymmetric partially oversaturated, and asymmetric
undersaturated conditions in a 60-minute study period. Demand changes are shown in Figure 5.2.

5.4 Coding IDSTOP

All algorithms (i.e. Simple GA, Elitist Simple GA, Micro-Elitist GA, ES, and ES+) were coded using
Matlab software. Code for simple GA was obtained from Illinois GA Lab and was modified to add elitism,
micro-elitism, and to put CORSIM and Traffic Assignment module in the loop. Code for ES and ES+ was
specifically developed for this study.
IDSTOP flowchart is shown in Figure 5.3 a-b. The flowchart is divided into two sections: a) initialization; and b) regeneration. In initialization, the first population is generated. After creating each individual, the constraints were checked. If they were not satisfied, the solution was discarded; otherwise, its fitness value was obtained. This was done by generating an input file for CORSIM containing the newly generated signal timing parameters. After making a certain number of replications (to account for stochasticities), the output file was read and the value of fitness function was obtained. After generating the entire initial population, the fittest individual was selected. System optimal traffic assignment was found for that individual. The fitness value was again determined after traffic assignment. If the fitness value was improved, the updated link and turning volumes were used for the next generation by coding them into CORSIM input file. If the fitness value was not improved, the old link and turning volumes were used. This process insured that at each iteration the fitness value can only be improved. Entire process is shown in Figure 5.3.

The initial population is used to regenerate new populations using GA or ES operators. As soon as a new individual is created, its feasibility is checked. If the new individual is feasible, its fitness value is determined by creating a CORSIM input file and calling the software and making several replications. If the new individual happens to be infeasible it is discarded and another individual is created. This process is continued until the entire new population is created. Similarly, the fittest individual is selected for traffic assignment and if traffic assignment yielded improvement in the fitness value, the new turning percentages will be used for the next generation. Otherwise the turning percentages will not be updated. Similarly new generations are created until termination criteria are met.
Figure 5.3 (cont. on next page)
b) Regeneration process

Figure 5.3. Schematic optimization process
5.5 Handling the Constraints

The constraints of the problem are divided into two groups. The first group includes those constraints that are directly enforced by GA. These constraints limit the value of decision variables and are called Type-I constraints. The second group includes those constraints that cannot be directly enforced by GA. Checking these constraints requires running macroscopic or microscopic traffic simulation. These constraints are called Type-II constraints. For instance, to obtain the value of queue length in each link over time, simulation run is needed. After the values of queue length are obtained, the constraints can be checked and proper action is taken if they are violated.

5.5.1 Type-I Constraints

Type-I constraints (i.e. constraints on cycle length, splits, and offsets for each intersection) are handled through the encoding and decoding of the binary chromosomes. For example, assume a chromosome with 5 binary genes is used to denote a random number between minimum cycle length \( C_{\text{min}} \) and maximum cycle length \( C_{\text{max}} \).

The real value of this 5-digit binary string is a real number, \( DV \), within the range of \([0, 31]\). The value of cycle length is determined (in GA to be used in CORSIM) using the following equation:

\[
C = C_{\text{min}} + \frac{(C_{\text{max}} - C_{\text{min}})}{2^d - 1}DV
\]  

(5.1)

Where:

\( C \) = the cycle length

\( C_{\text{min}} \) = minimum cycle length

\( C_{\text{max}} \) = maximum cycle length

\( d \) = number of strings of the decision variable
\[ DV = \text{decoded value of the string in base 10} \]

The minimum value of C is obtained by assuming \( DV = 0 \). The value of \( DV \) equals zero when all genomes of the string are zero. In that case:

\[ C = C_{\text{min}} \frac{(C_{\text{max}} - C_{\text{min}})}{2^{n-1}} \times 0 = C_{\text{min}} \quad (5.2) \]

The maximum value is obtained by assuming \( DV = 31 \). The value of \( DV \) equals 31 when all genomes of the string are one, which results in:

\[ C = C_{\text{min}} + \frac{(C_{\text{max}} - C_{\text{min}})}{2^5 - 1} \times 31 = C_{\text{min}} + (C_{\text{max}} - C_{\text{min}}) = C_{\text{max}} \quad (5.3) \]

This illustrates how GA encoding and decoding procedure enforces the satisfaction of the constraints on the decision variables.

### 5.5.2 Type-II Constraints

On the other hand, taking type-II constraints (i.e. constraints on queue length, de-facto red, gridlock, delay, etc.) into account needs complicated calculations and simulation runs. As mentioned before, all type-II constraints are eventually checked during microscopic simulation run. If the constraints are violated in any of the replications, the solution is discarded. However, running the microscopic simulation model is computationally expensive. Therefore, it is needed to identify infeasible solutions before running the microscopic simulation run. For this purpose a macroscopic approach is used. It uses shockwave theory to estimate the queue length inside the network and tallies the number of vehicles in each link. The model is adopted from Girianna and Benekohal (2002) and expanded to more than two phases. However, it assumes one lane for each turning movement and assumes that the queue length in each lane does not interfere with the vehicles in other lanes. This is equivalent to assuming exclusive left-turn and right-turn lanes as long as the length of the links. It is noted that this assumption makes the model less realistic and may results in marking an infeasible solution as a feasible
one (this assumption does not result in marking a feasible solution as an infeasible one). This infeasible solution will be identified during the microscopic simulation model run.

### 5.5.2.1 **Green Time Constraints**

Two sets of constraints were used to limit the value of green times. The first was shown by Equation 3.5 and the second was shown by Equation 3.6. The discussion here is centered on green time constraints formulated in Equation 3.6.

To obtain minimum and maximum green time values, traffic volumes (for critical movements) are needed. These volumes are available after performing traffic assignment and are plugged into Equation 3.6 and consequently the upper and lower bounds on green times can be found based on parameter $\delta$.

In the first generation, volume data are obtained by using either historic data, or by performing a system optimal traffic assignment based on the available o-d demand with assuming no intersection delays.

There is death penalty associated with green time constraints formulated in Equation 3.6. This means that whenever, the generated green times did not satisfy the constraints, the solution is discarded and anew solution is regenerated. These constraints are first checked using the macroscopic simulation model and then by microscopic simulation model. As soon as they are violated the solution is discarded and a new one is regenerated.

### 5.5.2.2 **Queue Length Constraints**

Violation of queue length constraints has a death penalty in this study since under no circumstances a solution with queue spillover should be used. Queue length constraints (shown in Equation 3.9) are first checked by the macroscopic model. If the queue length in any of the links becomes longer than the capacity of the link, the solution is discarded.
During the microscopic simulation model run, the queue lengths are also checked and if they grow longer than the capacity of their receiving link the solution is discarded.

5.5.2.3 Gridlock Constraints

There is death penalty associated with violating these constraints (Equation 3.10). The reason is that gridlocks significantly deteriorate the performance of a network and significantly increase delay and consequently travel time. The violation of gridlock constraints under any circumstance results in discarding the solution. Similar to previous constraints, gridlock constraints are checked in both macroscopic and microscopic simulation models. As soon as they are violated, the simulation run is terminated and the solution is discarded. The queue length constraints only consider the queue length in each link. Gridlock constraints monitor the queue length along an immediate closed loop and if the backs of queues get close to all upstream intersections the solution is discarded.

5.5.2.4 De-Facto Red Constraints

De-facto red constraints (Equation 3.12) were checked similar to queue length constraints using both macroscopic and microscopic simulation models; however, it should be noted that in some cases it is not possible to avoid de-facto red. For example, when all receiving links are almost full, regardless of the direction of green signal, there will be some de-facto reds. Whenever a preventable de-facto red occurs (other receiving links are not full) the solution is discarded. If a non-preventable de-facto red occurs (i.e. all receiving links of an intersection are full) the solution is not discarded.

5.5.2.5 Ideal Offset Constraints

There is no death penalty associated with ideal offset constraints (Equation 3.13). These constraints are used only when on a certain corridor signal coordination is needed regardless of its effect on the objective function. In other words, if it is required to have signal coordination in a corridor, ideal offset constraints are used. When they are violated on that corridor the objective function is
penalized. For the other corridors IDSTOP determines optimal offsets. The penalty is in the following form:

$$
\mu \left( \text{off}_{im,\varphi_i\varphi_m} \left( \frac{d_{im}}{v_{im,\varphi_m}} - \left( \frac{v_{im,\varphi_m} + \lambda}{\lambda v_{im,\varphi_m}} \right) q_{im,\varphi_m} \right) \right)
$$

(5.4)

Where:

$$\mu = \text{weight factor for the penalty function, and all parameters are introduced previously.}$$

Girianna and Benekohal (2002) suggested the use of saturation flow rate for the penalty. This results in similar units for the penalty and the objective function and is used as the weight factor for the penalty function. Since the purpose of this study is to let IDSTOP to optimize the offsets ideal offset constraints were not used.

5.5.2.6 Route Delay Constraints

As mentioned before, total delay on each route should be reasonable (see Equation 3.14). If no constraints to limit the average route delay are used in the network, some drivers may experience significantly long delays so that the network processes more vehicles. This does not satisfy equity requirements. The upper bound on the value of delay should be selected based on the traffic demand at the entry links. For very low traffic demands, the upper bound on the route average delay should be small compared to high traffic demands. Since the main purpose of this study is to address oversaturation, it is assumed that a level of service of E has to be maintained at all lane groups associated with the route. This means that if a route goes through four intersections, the associated lane group in each of those intersections has to maintain a LOS of E. According to HCM 2010, the maximum accepted control delay per vehicle for level of service E is 80 seconds. Therefore, if a route goes through $n$ intersections, the maximum acceptable delay per vehicle for that route is $80 \times n$ seconds. It should be
noted that the lane groups associated with entry links should be excluded since it is very likely to observe longer queues in them during to metering effect.

Delay constraints are checked while microscopic simulation model is run. If a solution violates delay constraints the solution is discarded, and a new solution is generated.

5.6 Generating the First Population

In the initialization step, generating the first population can be purely random; however, in that case most of candidate solutions do not satisfy some of the constraints. In fact, in order to create 450 feasible solutions, approximately 1,000,000 individuals were needed. It is noted that this number significantly reduced in the following generations. In Figure 5.4 the number of required individuals to obtain 450 feasible solutions is shown for different generations. Note that a logarithmic scale is used for y-axis on the left.

![Figure 5.4. Number of randomly created individuals to obtain 450 feasible solutions for different generations](image)

As mentioned before, checking all constraints during microscopic simulation run requires a significant amount of time. The reason is that simulating each time interval of our case study network requires one second of CPU time. Therefore, even if the microscopic simulation is run only one time,
checking the constraints requires 1 seconds of CPU time. Using the macroscopic approach reduces the CPU time to 0.0012 seconds which is significantly less than one second.

To further reduce the runtime, a heuristic approach is used to generate the first population. This heuristic is designed to reduce the possibility of generating an infeasible solution. For this purpose, cycle lengths at all the intersections of the network are chosen to be not much different from each other. This increases the possibility of signal coordination in the network. For each individual an interval of 10 seconds is stochastically assigned and all the cycle lengths are selected from that interval for all intersections of the network. Therefore, for a solution, signals at different intersections of the network can be at most 10 seconds different. The intervals are from 40-50 seconds up to 170-180 seconds. To define each interval, a uniform random number between 45 and 175 is generated, $x$. The interval is defined as $[x - 5, x + 5]$. It is assumed that offset between different intersections is zero. This is a fair assumption since intersections of the case study network are not far from each other. Finally the green splits are assigned proportional to the volume-to-saturated-flow-rate ratios of critical movements. When this heuristic was in used, the number of required individuals to obtain 450 feasible solutions was reduced to around 6200.

5.7 Fitness Function Evaluation

As mentioned before, throughout the optimization process, both GA and ES create new solutions using their operators and evaluate the fitness value of each solution (i.e. the value of the objective function for a set of signal timing parameters and turning percentages). Using these fitness values they choose the fitter individuals for the next generation and carry the optimization. Both algorithms use CORSIM for fitness function evaluation. However, to evaluate the fitness value of a solution more than one simulation run is needed to account for internal variability of CORSIM. These replications take into account the stochasticities associated with traffic flow propagation. Increasing the
number of replications improves the accuracy of fitness value estimation, yields a more robust solution, but increases the runtime. Since replications are done for each fitness function evaluation throughout the optimization, they increase the runtime almost linearly. Therefore, the number of replications needs to be carefully determined. Equation 5.5 has been extensively used for this purpose:

\[ n = \left( \frac{Z_{0.975} \sigma}{e} \right)^2 \]  

Where:

\( e \) = maximum allowed error in fitness value for a certain set of signal timing parameters

\( n \) = number of replications required in CORSIM

\( \sigma \) = standard deviation of fitness value for a certain set of signal timing parameters

Equation 5.5 can be used when the variance of fitness values for a solution is known; however, since the variance is not known it needs to be estimated using a sample. In that case, a try-and-error approach is used to determine the number of runs. For a certain solution, the fitness value is determined using a few replications (e.g. 5). Sample variance is determined using this sample of five fitness values and is plugged into equation 5.5 along with error and standard z-score and the number of replications is calculated. The sample size is big enough if it is larger than the calculated number of replications. In this case, the sample size can be reduced. If the sample size is smaller than the calculated number of replications, the sample size is not enough and needs to be increased. The process needs to be continued until the sample size is slightly smaller than the calculated number of replications.

Using this process, required sample size is determined for a certain solution. However, a different solution, as expected, results in different fitness values and different sample variance. Therefore, to determine the number of replications we did this process for 15 randomly picked solutions
with higher fitness values. For each one, this process was performed and the numbers of replications required are shown in Figure 5.5.

![Graph showing finding the number of fitness function evaluations throughout the optimization](image)

**Figure 5.5. Finding the number of fitness function evaluations throughout the optimization**

Values on y-axis represents the required number of replications which are determined by Equation 5.5. Values on x-axis represents the sample size that is used to determine variance. Any point that falls above line $y = x$ indicates that the required number of replication is higher than the starting sample size. As such, the number of replications is not enough. As shown, for up to a starting sample size of 8, required numbers of replications for at least some solutions are more than the starting sample size. Therefore, the number of replications cannot be less than or equal to 8. From staring sample size of 9 and higher, the required number of replications is always less than starting sample size. This indicated that any number of replication equal to or larger than 9 is enough. Since for starting sample size of 9, the required number of replications for one case was very close to the starting sample size (8.5), we decided to set the number of replications equal to 10. As shown in Figure 5.5, for other staring sample sizes, the maximum number of required replications is always around 10 indicating that 10 replications is reasonable. It is noted that in all computations it was assumed that $\alpha = 0.05$ and maximum tolerable error in fitness value evaluation was 2%. 

101
5.8 Using Cache

All solutions that satisfy all constraints in macroscopic approach are recorded in cache. After a solution is created and satisfied all constraints in macroscopic approach, IDSTOP searches the cache before calling microscopic simulation model. If the new individual already existed in cache, the microscopic simulation model is not called anymore. Instead, from cache data it is determined if the solution has satisfied all the constraints in microscopic mode. In that case, its fitness is obtained from cache as well. If this solution has not satisfied some of the constraints based on cache data, it is discarded. If the new solution was not existed in cache it would be added to cache as well as its feasibility (in microscopic approach) and fitness (if it was feasible).

5.9 Obtaining an Upper Bound

As mentioned before, a microscopic traffic simulation model is used to move vehicles inside the network. There are several constraints to load the network and to move vehicles in it that are handled by this model and have not been presented in our formulation. For example, car following constraints and those preventing discharge of vehicles from an upstream intersection when the receiving link is full are not shown. In the following of the text we explain how relaxing some of these constraints can help find an upper bound to the number of completed trips. To obtain the upper bound, two conditions need to be studied:

1. Oversaturated conditions
2. Undersaturated conditions

5.9.1 Oversaturated Conditions

In oversaturated conditions, the upper bound of the number of completed trips cannot be determined by simply summing up traffic demand for all origin destination pairs. In fact, the number of
completed trips in these conditions is controlled by the capacity of the intersections. Therefore, to find
the highest number of completed trips, one needs to find the capacity of the network in processing
vehicles. In oversaturated conditions, since traffic demand is more than the capacity of the network,
queues are very likely to grow and may eventually block upstream intersections. When this happens, the
capacity of an intersection does not only depend on its own signal timing parameters; it also depends on
the signal timing parameters of its downstream intersections. Therefore, the capacity of an upstream
intersection may be “lowered” by the capacity of its downstream intersections. At the best case, when
there is no interference from the queue at the downstream intersections or when this interference is
“ignored”, the capacity of the upstream intersection is at its highest level (for a set of signal timing
parameters at upstream intersection). Therefore, to get the upper bound on the number of trips
completed in a network, the interference between the upstream and downstream intersections needs
to be ignored. This is equivalent to relaxing some of the network loading constraints that do not let an
upstream intersection discharge any vehicle when a receiving link is full. By relaxing these constraints,
we have assumed that vehicles can be discharged even when the receiving links are full. Therefore, we
have to assume that the vehicles are stacked on top of each other when a link is full. Relaxing these
constraints ensures that for a set of signal timing parameters, each intersection operates at its highest
capacity. Any feasible solution to this relaxed problem is an upper bound to the original problem. To
further increase the capacity of each intersection, its signal timing parameters are the only remaining
parameters to be optimized. Since long queues are not of any concern with the relaxed constraints, the
cycle length of each intersection should be as high as possible. This reduces the lost time (by having
fewer phase changes) and consequently increases the capacity of each intersection.

It should be noted that in this study the capacity of an intersections denotes the sum of volumes
of all critical movements that can be processed. If the saturation flow headway is $h_s$, a total of $\frac{3600}{h_s}$
vehicles pass a point during an hour of uninterrupted traffic flow. However, at an intersection, the
control device results in some wasted time in each cycle during which, no vehicle is processed. This period is called lost time and is denoted by $L$. Thus, the number of vehicles that are released from competing phases of an intersection is a portion of $\frac{3600}{h_s}$. In fact, since during each cycle $L$ seconds is wasted, the sum of vehicles discharged from all critical movements of an intersection can at most be $\frac{C-L \cdot 3600}{C \cdot h_s}$.

When traffic demand in all links are high enough to make the network oversaturated (around 1000 vphpl or more), it is valid to assume that system optimal traffic assignment should allocate traffic to different links such that traffic congestion levels at different links of the network are approximately similar. Therefore, similar green durations can be used at different intersections of the network including those intersection located at the boundaries of the network. As such, one can assume that the number of discharged vehicles from east bound and southbound of intersection no.1 are approximately similar and are also equal to the number of discharged vehicles at the east bound of intersection no. 2 (and the rest of intersections at the boundaries), see Figure 5.6.

![Figure 5.6. Schematic urban traffic network](image)

Therefore, the number of vehicles exited from each exit link is:
Therefore, the upper bound to the number of completed trips is:

\[
\frac{C - L}{3600} \times \frac{1}{h_s} \times \frac{1}{2} \times H
\]

(5.6)

Therefore, the upper bound to the number of completed trips is:

\[
\frac{C - L}{3600} \times \frac{1}{h_s} \times \frac{1}{2} \times H
\]

(5.7)

\(H\) is the number of exit links, and

\(h_s\): saturated headway

Smallest value of \(h_s\) yields the highest value of equation 5.7 which is the upper bound on the number of completed trips. As such, if a value of zero is used for \(h_s\), the upper bound to the number of completed trips will be infinity; however, we know that vehicles cannot reduce their headway from a certain value (vehicle length over travelling speed). In fact, in real world, vehicles maintain longer headways from each other than the minimum possible. To obtain a reasonable value for this headway, we estimated the average headway of vehicles at the exit links of the case study network in 250 different replications when very efficient signal timing parameters and traffic assignment were used. The distribution of these average headways is shown in Figure 5.7.

Figure 5.7. Distribution of average saturated headway at exit intersections of the network
Each run out of 250 runs, gave us more than 1800 exit headways whose average was one observation in the Figure. As shown in Figure 5.7, out of 250 average saturated headways, only 10 resulted in average saturated headways between 1.91 and 1.93 seconds. Only one observation had an average saturated headway between 1.87 and 1.89 seconds. However, to be on the safe side and reduce the possibility of having an average saturated headway smaller than $h_s$, we chose a value of 1.85 for $h_s$.

By assuming that around 10% of the cycle length is wasted due to the lost time, and knowing $h_s = 1.85$, the sum of volumes of the critical movements at each intersection can be at most $0.9 \times 3600 \times \frac{0.9}{1.85} = 1751 \text{ vph}$ (assuming each leg has one lane).

For case-b, symmetric oversaturated conditions, the combined traffic demand for each intersection is around $2 \times 1000 = 2000 \text{ vphpl}$ which is more than 1751 vphpl and network is uniformly oversaturated. Therefore, at each intersection a maximum of 1751 vphpl can be processed (assuming that the effect of turning vehicles is minimal). Since the traffic demand is symmetric, it is fair to assume that identical splits are used and consequently $\frac{1751}{2} = 876 \text{ vphpl}$ are at most processed at each leg of each intersection. Therefore, upper bound of the number of trips for a time interval of 5 minutes is $27 \times 876 \times \frac{5}{60} = 1971 \text{ vehicles}$. Note that there are 27 exit lanes in the case study network.

### 5.9.2 Undersaturated Conditions

In undersaturated conditions, when traffic demand is lower than the capacity of the network, the number of trips completed in the network can be at most as much as total traffic demand. The reason is that in undersaturated conditions, good signal timing parameters ensures that all demand is processed in the network. Therefore, the upper bound can be determined by the following equation:

$$
\sum_{r \in \mathcal{R}, s \in \mathcal{S}} d_{rs}^t, \quad \forall t \in T_t
$$

(5.8)

\[d_{rs}^t = \text{traffic demand from origin node } r \text{ to destination node } s \text{ at time } t\]
This upper bound is valid even when some of the corridors are oversaturated but the entire network has some capacity left since system optimum traffic assignment module reroutes the vehicles in the network such that all the capacity is used.

Similar to oversaturated condition, arrival headway of vehicles in CORSIM is stochastic and is generated using a specific headway distribution. Default distribution is Erlang with a shape parameter equal to 1. The average headway is \( \frac{3600}{V} \) where, \( V \) is entry traffic volume. However, since the headways generated randomly, it is possible that for a sequence of random numbers CORSIM generates headways such that their average is less than \( \frac{3600}{V} \). In that case average entry volume would be more than \( V \) and therefore, the number of completed trips may exceed the upper bound. Therefore, similar to oversaturated condition, the distribution of the average entry headways is found and entry headway is determined such that the probability of exceeding the upper bound is small. The average number of vehicles entering the network for a period of five minutes for the case studies is 1690 vehicles. This means that the arrival headway for each replication needs to be generated 1690 times. For Erlang distribution, the average headway and its variance for our case study are 4.8. According to central limit theorem, the average entry headway has a normal distribution with average of 4.8 and standard deviation of 0.053. Therefore, if we assume that the probability of exceeding the upper bound is 0.001, the entry headway should be \( 4.8 - 3.1 \times 0.053 = 4.64 \). Therefore, a value of \( \frac{3600}{4.64} = 776 \) vphpl should be used for the entry volume.

For Case-a, symmetric undersaturated conditions, the combined traffic demand for each intersection is around \( 2 \times 750 = 1500 \) vph which is less than 1751 (due to the turning movements the volume may not be exactly 750 vphpl). Thus, all traffic demand can be processed. In the worst case the average entry traffic volume in the network may reach 776 vphpl. The network has a total of 27 entry
links, and study period is 5 minutes. As a result, the upper bound to the number of completed trips is \(27 \times 776 \times \frac{5}{60} = 1746\) vehicles.

For Case-c, asymmetric undersaturated conditions, each intersection serves a major street with a volume of 1000 vphpl and a minor street with volume of 500 vphpl. The sum of critical movements is 1500 vph that is less than 1751. Thus, all traffic demand can be processed. Using the same approach for case-a, results in an upper bound of 1578 trips.

Finally for case-d, asymmetric partially oversaturated conditions, for all intersections except for 10, where critical volume of each direction is 1000 vphpl, sums of critical movements are less than 1751. Thus, those intersections could process all vehicles. But in intersection number 10, the sum of critical movements can be at most 1751 (assuming that the effect of turning vehicles is minimal). Since the demand at this intersection is symmetric, the capacity of each direction is \(\frac{1751}{2} = 876\) vphpl. It should be noted that the capacity of corridors B-L and P-G will be 876 vphpl. However, since for case-d there is unused capacity in alternative routes, all entry traffic demand can be processed at corridors B-L and P-G, see in Figure 5.1. Therefore, using the same approach used for case-a and case-c, the upper bound to the number of completed trips is 1628 vehicles.

### 5.10 Comparing Solutions

Both IDSTOP and Direct-CORSIM are used to optimize signal timing on the case study network. After each one finds its final solution, both solutions are simulated in CORSIM and several network Performance Measures (PM) are determined. To account for internal variability of CORSIM a certain number of replications is needed to provide enough accuracy in estimation of each performance measure. The number of replications can be determined using the following equations:

\[
n = \frac{s^2 x^2}{\varepsilon^2}
\]

(5.9)
Where:

\[ n = \text{number of required replications} \]
\[ S = \text{standard deviation of each PM}, \]
\[ z = \text{critical value of normal distribution (or student t distribution if N is less than 30) for a certain confidence level, and} \]
\[ \varepsilon = \text{maximum accepted error}. \]

In this study the error in estimation of each PM was limited to a maximum of 1% and \( \alpha = 0.05 \) was used. The same approach used to determine the number of replications for fitness function evaluation was used here. For final solutions, first the PM were obtained by 10 replications and if required number of replications was more than specified one, the number of replications were increased.

In general, estimating total delay required around 80-100 replications while to estimate the number of completed trips around 20-30 replications were enough. Therefore, a maximum of 100 replications was enough to accurately estimate all important PM. However, a total of 250 replications were made to add more accuracy to the estimation of the PM although was not needed.

In addition, when 250 replications are made, each solution is tested under a vast set of diverse traffic conditions (i.e. arrival headways, and drivers’ behavior) to identify if it results in a sustainable network performance under these different conditions. At the end, student t-test is used to statistically compared PM of different solutions. Based on the equality or inequality of the variance of the PM, proper student t-test is performed.
5.11 Numerical Findings

IDSTOP’s performance evaluation included the following steps:

1. Comparing Fixed-Time-No-Traffic-Assignment-IDSTOP (FTNTA-IDSTOP) solution to Direct-CORSIM optimizer solution under several traffic demand scenarios
2. Comparing Real-Time-No-Traffic-Assignment-IDSTOP (RTNTA-IDSTOP) solution to FTNTA-IDSTOP solution under several traffic demand scenarios (similar to step 1)
3. Comparing RTNTA-IDSTOP solution to Direct-CORSIM solution
4. Comparing IDSTOP (real-time with traffic assignment) solutions to RTNTA-IDSTOP solution under several traffic demand scenarios

5.11.1 FTNTA-IDSTOP vs. Direct-CORSIM

Direct-CORSIM optimizer uses a GA code developed in TRANSYT7F and can optimize signals using CORSIM (as well as a macroscopic approach). When GA generates a solution, the signal timing parameters are transmitted to CORSIM, to simulate the network and find PM for the network. This PM is the fitness value for the solution. Direct-CORSIM optimizer does not change signal timing over time even if traffic demand changes. As a result, to make a “fair” comparison between Direct-CORSIM and IDSTOP we have to modify IDSTOP to generate fixed-time solutions. In addition, Direct-CORSIM cannot optimize the phase sequence. Therefore, IDSTOP was altered not to optimize it either. Finally, Direct-CORSIM is not capable of traffic assignment. As such, traffic assignment feature of IDSTOP was turned off. This resulted in having a Fixed-Time-No-Traffic-Assignment IDSTOP (FTNTA-IDSTOP). The comparison was performed for all four fixed-demand traffic patterns.

5.11.1.1 Case-a, Undersaturated Network with Symmetric Traffic Demand

For Case-a, FTNTA-IDSTOP and DIRECT-CORSIM solutions had the following similarities: They both found a common cycle for the network and coordinated several movements. They also allocated
the green times proportional to the traffic demand. However, FTNTA-IDSTOP found a common cycle of 80 seconds with green splits of 36 seconds while DIRECT-CORSIM found a common cycle of 60 seconds with identical green splits of 26 seconds for all intersections (three seconds of yellow indication and one second of all-red indication).

Under DIRECT-CORSIM, 4930 trips were completed in the network. This was only 5.9% below the theoretical upper bound of 5238 (3*1746) trips, indicating that the solution was very close to the theoretical optimal. FTNTA-IDSTOP solution significantly increased the number of completed trips to 5030 (with $\alpha = 0.01$).

Delay for FTNTA-IDSTOP solution was 10.3% less than that for DIRECT-CORSIM. The difference was statistically significant with $\alpha = 0.01$. A reduction of 10.3% in delay was a great improvement by itself. It is important to note that the FTNTA-IDSTOP solution not only increased the number of completed trips and pushed it very close to the upper bound, but also at the same time significantly reduced total delay. These two together indicate a significant improvement in network performance. This improvement was possible by efficient signal coordination. FTNTA-IDSTOP managed to coordinate signals such that in east-west direction vehicles could go through the arterials with only one stop. Similar coordination was observed in the north-south direction. This was achieved by using a longer cycle length and a better optimization of the offsets.

5.11.1.2 Case-b, Oversaturated Network with Symmetric Traffic Demand

When volume was increased to 1000 vphpl, high enough to make the network oversaturated in Case-b, both FTNTA-IDSTOP and DIRECT-CORSIM found common cycle lengths. DIRECT-CORSIM’s common cycle and green splits were identical to those in undersaturated condition (cycle length of 60 and splits of 26 seconds); however, the offsets were different. On the Other hand, FTNTA-IDSTOP increased the cycle length by 40 seconds resulting in a common cycle of 120 and green splits of 56
seconds. The splits were also proportional to the traffic demand (on average). IDSTOP managed to coordinate the east-west and north-south direction arterials so that vehicles could travel through them with only one stop. DIRECT-CORSIM on the other hand used a different strategy. By using a short common cycle, it tried to let fewer vehicles enter the network in order to keep it less congested.

When compared to DIRECT-CORSIM solution, FTNTA-IDSTOP solution significantly ($\alpha = 0.01$) increased the number of completed trips (by 18.1%). This indicated a considerable improvement in the network performance for oversaturated condition. Number of completed trips was only 5.6% below the upper bound, see in Table 5.1. In addition, FTNTA-IDSTOP solution significantly reduced total delay by 14.8% compared to DIRECT-CORSIM solution. Increasing the number of trips together with reducing delay was a major improvement in the network performance efficiency that was possible by a more efficient signal coordination. This was achieved by using a longer cycle length and a better optimization of the offsets.

Comparing the FTNTA-IDSTOP and DIRECT-CORSIM solution in undersaturated to those in oversaturated condition, suggests that a longer cycle in oversaturated condition and a shorter cycle in undersaturated condition can improve network performance efficiency. A longer cycle in oversaturated condition reduces the lost time thus, more green is available. In addition, it increases the possibility of coordination. These two together can significantly improve network performance. It should be noted that using an excessively long cycle length in oversaturated condition can potentially result in long queues in the network that may increase the possibility of queue spillovers, de-facto reds, and gridlocks. Using long cycles in undersaturated conditions may increase delay as well. Delay can increase when there is no additional demand for the current green signal and the signal is not switched due to the long cycle length while in the competing direction, there is demand for green signal.
Table 5.1 PM for FTNTA-IDSTOP vs. Direct-CORSIM

<table>
<thead>
<tr>
<th>Measure of Performance</th>
<th>Statistic</th>
<th>Case-a FTNTA</th>
<th>Case-b FTNTA</th>
<th>Case-c FTNTA</th>
<th>Case-d FTNTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DirectCOR.</td>
<td>4691</td>
<td>4900</td>
<td>4290</td>
<td>5301</td>
</tr>
<tr>
<td># of Trips Completed</td>
<td>Min</td>
<td>4691</td>
<td>4900</td>
<td>4290</td>
<td>5301</td>
</tr>
<tr>
<td></td>
<td>10th %</td>
<td>4545</td>
<td>4975</td>
<td>4726</td>
<td>5583</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>4930</td>
<td>5030</td>
<td>4726</td>
<td>5583</td>
</tr>
<tr>
<td></td>
<td>90th %</td>
<td>4995</td>
<td>5078</td>
<td>4886</td>
<td>5694</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>5072</td>
<td>5127</td>
<td>5059</td>
<td>5905</td>
</tr>
<tr>
<td>Total Delay (h)</td>
<td>Min</td>
<td>111</td>
<td>107</td>
<td>225</td>
<td>197</td>
</tr>
<tr>
<td></td>
<td>10th %</td>
<td>120</td>
<td>113</td>
<td>242</td>
<td>208</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>129</td>
<td>117</td>
<td>257</td>
<td>219</td>
</tr>
<tr>
<td></td>
<td>90th %</td>
<td>138</td>
<td>121</td>
<td>274</td>
<td>232</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>157</td>
<td>140</td>
<td>308</td>
<td>249</td>
</tr>
<tr>
<td>Average Delay (s)</td>
<td>Min</td>
<td>68</td>
<td>66</td>
<td>130</td>
<td>105</td>
</tr>
<tr>
<td></td>
<td>10th %</td>
<td>74</td>
<td>69</td>
<td>138</td>
<td>108</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>79</td>
<td>72</td>
<td>147</td>
<td>114</td>
</tr>
<tr>
<td></td>
<td>90th %</td>
<td>85</td>
<td>74</td>
<td>158</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>97</td>
<td>85</td>
<td>181</td>
<td>129</td>
</tr>
<tr>
<td>Vehicles in the Network</td>
<td>Min</td>
<td>793</td>
<td>780</td>
<td>1341</td>
<td>1229</td>
</tr>
<tr>
<td></td>
<td>10th %</td>
<td>851</td>
<td>805</td>
<td>1456</td>
<td>1279</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>917</td>
<td>846</td>
<td>1558</td>
<td>1345</td>
</tr>
<tr>
<td></td>
<td>90th %</td>
<td>980</td>
<td>890</td>
<td>1661</td>
<td>1410</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>1142</td>
<td>948</td>
<td>1784</td>
<td>1531</td>
</tr>
<tr>
<td>Average Travel Time (s)</td>
<td>Min</td>
<td>122</td>
<td>120</td>
<td>180</td>
<td>157</td>
</tr>
<tr>
<td></td>
<td>10th %</td>
<td>128</td>
<td>123</td>
<td>188</td>
<td>160</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>133</td>
<td>126</td>
<td>197</td>
<td>166</td>
</tr>
<tr>
<td></td>
<td>90th %</td>
<td>139</td>
<td>129</td>
<td>207</td>
<td>173</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>149</td>
<td>139</td>
<td>228</td>
<td>181</td>
</tr>
</tbody>
</table>

# Upper bound to # of trips Completed

5238  5913  4734  4884

FTNTA: Fixed-Time-No-Traffic-Assignment-IDSTOP
DirectCOR.: Direct-CORSIM stochastic optimizer

5.11.1.3 Case-c, Undersaturated Network with Asymmetric Traffic Demand

For Case-c, where traffic volume was 1000 vphpl on east-west streets and 500 vphpl on north-south streets, it was expected that FTNTA-IDSTOP allocated a longer green time to east-west direction, and also coordinated most of the movements along this direction. In fact, this was what happened as FTNTA-IDSTOP assigned around 2/3 of the available green to the east-west and the remaining to the north-south direction (on average). FTNTA-IDSTOP found a common cycle of 62 seconds and offsets such that the signals along the major streets were coordinated. This also resulted in signal coordination in minor streets of the network. DIRECT-CORSIM solution was different from FTNTA-IDSTOP solution. DIRECT-
CORSIM found a common cycle of 30 seconds for the network and on the average assigned around 64% of the effective green to the direction with more demand. By using a short cycle length, Direct-CORSIM limited the number of vehicles entering into the network to keep its interior less congested. No more than a few coordinated phases were observed in the network.

As shown in Table 5.1, when FTNTA-IDSTOP solution was used, on the average 4444 trips were completed which was only 6.1% less than the upper bound. This indicated that for Case-c, FTNTA-IDSTOP could find a solution that was very close to theoretical optimal. Around 11.8% more trips (statistically significantly more with $\alpha = 0.01$) were completed by FTNTA-IDSTOP solution as opposed to DIRECT-CORSIM. Similar to previous two cases, FTNTA-IDSTOP significantly increased the number of trips, meanwhile significantly reduced total delay (11.9%).

5.11.1.4 Case-d, Partially Oversaturated Network with Asymmetric Traffic Demand

For Case-d where traffic demand was not symmetric, FTNTA-IDSTOP found a common cycle of 112 seconds for the network with identical green splits. This may sound counterintuitive at the first glance since for most of the intersections, the green splits are not proportional to traffic volumes. However, as mentioned previously this prevented long queues in the network and in addition coordinated the signals. DIRECT-CORSIM also found a common cycle of 60 seconds for the network; however, the green splits where not assigned proportional to the volume ratios of the critical movements to prevent queues at some links.

When FTNTA-IDSTOP found the solution, on the average 4230 trips were completed which was 13.4% below the theoretical upper bound. Larger distance from the upper bound was expected for asymmetric partially oversaturated conditions since without traffic assignment FTNTA-IDSTOP could not utilize the unused capacity of the network. When DIRECT-CORSIM solution was in effect 3930 trips were completed that was significantly fewer than those for FTNTA-IDSTOP with $\alpha = 0.01$. Similar to previous
case, not only FTNTA-IDSTOP managed to process more vehicles in the network (7.6% more), it significantly reduced total delay (by 13.7%).

5.11.2 RTNTA-IDSTOP vs. FTNTA-IDSTOP

In this part RTNTA-IDSTOP is compared to FTNTA-IDSTOP. Again for a “fair” comparison, static traffic demand was used during the study period. Therefore, any change in RTNTA-IDSTOP solution over time is not due to changes in traffic demand. These changes are the result of queue build up and vehicle accumulation inside the network that trigger the change in signal timing parameters over time. Modified Springfield network with the four demand patterns were used.

5.11.2.1 Case-a, Undersaturated Network with Symmetric Traffic Demand

When RTNTA-IDSTOP found a solution for this case, it was identical to FTNTA-IDSTOP solution. The reason was that the solution that was obtained by the FTNTA-IDSTOP (and also DIRECT-CORSIM) was already extremely close to theoretical optimal. In addition, in undersaturated condition, queues did not build up and vehicles were not accumulated in the network. Thus, the state of the network did not change from one time interval to the other one. As a result, signal timing parameters did not change.

5.11.2.2 Case-b, Oversaturated Network with Symmetric Traffic Demand

RTNTA-IDSTOP solution varied over time. For the first five minutes, the signal timing parameters were identical to FTNTA-IDSTOP solution. However, since no protected left-turn phase was used, left turn queues started to build up. This resulted in using four-phase signal plans at intersections number 15, 16, 19, and 20, and using three-phase signal plans at intersections number 3, 4, 7, 8, 11, 12, 13, 14, 17, and 18 during the second time interval. Phase sequence at the rest of intersections remained the same. For intersections with two phases, through movement green duration for each direction was 70 seconds. For intersections with three phases, through movement green duration was 65 seconds and left turn green duration was 5 second. Note that if IDSTOP finds a left turn green duration of less than 5
seconds, it omits the left turn phase. For intersection with four phases, through movement green duration was 60 seconds and left turn movement green duration was 5 seconds for each direction. This solution resulted in processing the left-turners and eliminating left-turn queues in the network. Therefore, there was no left turn queue at the start of the third time interval. As such, protected left turn phases were omitted and a two-phase signal plan was used for the third time interval. However, since in the second time interval the share of through movements from green time was reduced, a longer common cycle was used in the third time interval and again the signals were coordinated. In the third time interval the cycle length was 130 seconds with 61 second of green duration for through movement for each direction.

RTNTA-IDSTOP solution resulted in a total of 5650 completed trips in the network that was 1.2% more than FTNTA-IDSTOP see in Table 5.2. This difference was statistically significant with $\alpha = 0.01$. RTNTA-IDSTOP solution did not reduce average delay and average travel time however, processed more vehicles in the network. Note that number of completed trips for FTNTA-IDSTOP solution was already very close to the theoretical upper bound indicating that there is a very limited room for further improvements. That is why the 1.2% improvement in the number of completed trips is very important. In fact, this improvement resulted on number of completed trips that was only 4.4% below the theoretical upper bound.

The main reason of more completed trips is the changes in the signal timing parameters according to the state of the network. RTNTA-IDSTOP could process more vehicles by processing left turners when there was considerable queue for them by allocating protected left turn phases.

5.11.2.3 Case-c, Undersaturated Network with Asymmetric Traffic Demand

Similarly for Case-c, RTNTA-IDSTOP solution varied over time. The solution for the first and second time interval used two phases while for the third time interval, the number of phases was
increased to three where a protected left-turn phase was allocated to the major streets where left turn movements were allowed. In the first and second time intervals, a common cycle of 74 seconds was used. Higher proportion of the green time (2/3) was allocated to the approaches with 1000 vphpl and smaller proportion of that (1/3) was allocated to the approaches with 500 vphpl. In the third time interval, again a common cycle was used (76 seconds). RTNTA-IDSTOP used three-phase at intersections number 3, 4, 7, 8, 11, 12, 15, 16, 19, and 20. At these intersections a protected left phase was allocated to the major streets. When the minor streets were two-way, a leading protected left-turn was used. When one of the minor streets was one-way, through and the left-turn movement on the majors were combined. For both cases, the duration of the phase containing the left turns was 5 seconds. Duration of through movement green signal was 37 seconds on the major streets and 23 seconds on the minors. In case of two phase signal plans, 42 seconds of green was allocated to through movement on the major, and 26 seconds of that was allocated to through movement on the minor. Signals were coordinated in all three time intervals.

When RTNTA-IDSTOP solution was in effect, 4524 trips were completed in the network that shows a 1.8% increase when compared to FTNTA-IDSTOP. The difference was statistically significant with $\alpha = 0.01$. Not only the number of completed trips was increased, the average delay per vehicle was significantly reduced by 16.5%. Looking at these two MP at the same time indicates a considerable improvement in network performance.

5.11.2.4 Case-d, Partially Oversaturated Network with Asymmetric Traffic Demand

RTNTA-IDSTOP coordinated the signals in all three time intervals. In the first time interval it used a common cycle of 120 seconds with only two-phases, with identical splits of 56 seconds. However, in the second time interval it used two different cycle lengths in the network. For all intersection along corridors carrying a volume of 1000 vphpl, it used identical cycles of 128 seconds. By optimizing the offsets, the signals were coordinated in each cycle. For the rest of the intersections identical cycles of 64
seconds were used which was half of 128 seconds. As a result, when the offsets were optimized, along minor corridors, the signals were coordinated every other cycle. By doing so, signals along major corridors were always coordinated. Along minor corridors, signals were coordinated every other cycle, but with a much shorter cycle that resulted in significant reduction in wasted green. In the third time interval a similar strategy was used. However, the cycle lengths were shortened. Along major corridors, a common cycle of 96 seconds, and along minor corridors (except for when they intersect the majors) a common cycle of 48 seconds were used. In all three time interval, the offsets were optimized to coordinate the signals along the major corridors such that vehicles arriving from an upstream intersection, go through the intersection without a stop.

Table 5.2. PM for RTNTA-IDSTOP vs. FTNTA-IDSTOP

<table>
<thead>
<tr>
<th>Measure of Performance</th>
<th>Statistic</th>
<th>Case-a FTNTA</th>
<th>Case-b FTNTA</th>
<th>Case-c FTNTA</th>
<th>Case-d FTNTA</th>
</tr>
</thead>
<tbody>
<tr>
<td># of Trips Completed</td>
<td>Min</td>
<td>4900</td>
<td>5301</td>
<td>4333</td>
<td>4149</td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>4975</td>
<td>5466</td>
<td>4394</td>
<td>4192</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>5030</td>
<td>5583</td>
<td>4444</td>
<td>4230</td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>5078</td>
<td>5694</td>
<td>4495</td>
<td>4262</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>5127</td>
<td>5905</td>
<td>4542</td>
<td>4287</td>
</tr>
<tr>
<td>Average Delay (s)</td>
<td>Min</td>
<td>66</td>
<td>105</td>
<td>84</td>
<td>72</td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>69</td>
<td>108</td>
<td>91</td>
<td>73</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>72</td>
<td>114</td>
<td>97</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>74</td>
<td>121</td>
<td>103</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>85</td>
<td>129</td>
<td>108</td>
<td>77</td>
</tr>
<tr>
<td>Vehiciles in the Network</td>
<td>Min</td>
<td>780</td>
<td>1229</td>
<td>929</td>
<td>656</td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>805</td>
<td>1279</td>
<td>997</td>
<td>684</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>846</td>
<td>1345</td>
<td>1053</td>
<td>701</td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>890</td>
<td>1410</td>
<td>1109</td>
<td>722</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>948</td>
<td>1531</td>
<td>1162</td>
<td>742</td>
</tr>
<tr>
<td>Average Travel Time (s)</td>
<td>Min</td>
<td>120</td>
<td>157</td>
<td>138</td>
<td>127</td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>123</td>
<td>160</td>
<td>145</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>126</td>
<td>166</td>
<td>150</td>
<td>129</td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>129</td>
<td>173</td>
<td>156</td>
<td>130</td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>139</td>
<td>181</td>
<td>160</td>
<td>132</td>
</tr>
<tr>
<td>Upper bound to #of trips Completed</td>
<td>5238</td>
<td>5913</td>
<td>4734</td>
<td>4884</td>
<td></td>
</tr>
</tbody>
</table>

RTNTA: Real-Time-No-Traffic-Assigment-IDSTOP
FTNTA: Fixed-Time-No-Traffic-Assigment-IDSTOP
When RTNTA-IDSTOP solution was in effect, 4287 trips were completed in the network that shows a 1.3% increase when compared to FTNTA-IDSTOP. The difference was statistically significant with $\alpha = 0.01$. Not only the number of completed trips was increased, the average delay per vehicle was significantly reduced by 4.1%. Looking at these two PM at the same time indicates a considerable improvement in network performance, see in Table 5.2.

5.11.3 RTNTA-IDSTOP vs. DIRECT-CORISM

In the previous sections it was established that FTNTA-IDSTOP solutions resulted in a more efficient network performance than Direct-CORISM solutions. It was also shown that RTNTA-IDSTOP solutions improved network performance when compared to FTNTA-IDSTOP solutions. Consequently, RTNTA-IDSTOP solutions will result in more efficient network performance than Direct-CORISM solutions. Therefore, no further comparisons between RTNTA-IDSTOP solutions and Direct-CORISM solutions are needed.

5.11.4 IDSTOP vs. RTNTA-IDSTOP

In this section, IDSTOP with all its features functional (including traffic assignment) is compared to RTNTA-IDSTOP (without traffic assignment). For a “fair” comparison, and to keep the test bed unchanged, it is assumed that traffic demand stays the same in this section as well. Both IDSTOP and RTNTA-IDSTOP may change signal timing parameters from one time interval to another. It should be noted that this change is due to vehicle accumulation and queue build up inside the network that varies over time and results in continuous changes in the state of the network.

As mentioned earlier, in the previous comparisons certain turning percentages were used throughout the network. It was assumed that at each intersection, 10% of traffic of the right-most lane makes a right turn, 10% of the traffic of the left-most lane makes a left turn, and the remaining vehicles go straight. However, to assign traffic an o-d demand is needed. As a result, to keep the test case study
network similar to previous cases, o-d demand was estimated based on the turning percentages that were assumed in previous steps of verification and validation. To estimate the o-d demand for each of the four demand cases, all routes from each origin to each destination needed to be found. For each route, based on the traffic demand and turning percentages, the number of vehicles that reached each destination node was calculated. Therefore, the o-d demand was found for all o-d pairs.

5.11.4.1 Case-a, Undersaturated Network with Symmetric Traffic Demand

Similar to previous steps, for case-a, IDSTOP found a solution that was identical to RTNTA-IDSTOP in terms of signal timing parameters. Turning percentages were also very similar to those used in previous cases. The reason was that previous solutions were already very close to theoretical optimal and the traffic assignment used in previous steps is very close to optimal. This was observed since in most of the steps, performing traffic assignment did not result in significant improvement in the value of objective function.

5.11.4.2 Case-b, Oversaturated Network with Symmetric Traffic Demand

Simultaneous transportation supply and demand management did not significantly improve network performance (compared to only supply management) in this case mainly due to the following three reasons: 1) In case-b, the network was oversaturated and there was no unused capacity available. As a result, when signal timing parameters were optimized together with link and turning percentages, the solution did not improved compared to RTNTA-IDSTOP solution. 2) As mentioned before, the turning percentages that were used in previous steps of validation were very close to optimal. 3) RTNTA-IDSTOP solution was already very close to theoretical upper bound.

5.11.4.3 Case-c, Undersaturated Network with Asymmetric Traffic Demand

Similar to previous cases, in case-c, simultaneous transportation supply and demand management did not significantly improve network performance of the case study network. The reason
was that the traffic assignment that was used in previous steps was very close to optimal. As a result, IDSTOP was not able to improve the value of the objective function by assigning traffic in the network.

5.11.4.4 Case-d, Partially Oversaturated Network with Asymmetric Traffic Demand

Similar to previous cases, the origin and destination demand was estimated based on the turning percentages. In this case since not all the corridors of the network were oversaturated traffic assignment could potentially improve network performance.

When transportation demand and supply were managed together, IDSTOP rerouted some of the traffic from corridors B-L and P-G through minor corridors, see in Figure 5.1. This was done to reduce the sum of volumes for critical movements to the capacity or below that at intersection number 10. In the first time interval among the vehicles driving from origin node “P” to destination node “G”, around 160 of them were rerouted through intersections 9-5-1-2-3. Half of this many vehicles reached their destination by going through nodes 7-11-12 and the rest went through nodes 4-8-12.

Among the vehicles travelling from node “B” to node “L” around 130 vphpl were rerouted through nodes 2-3. These vehicles were divided into four groups and got to their destination through routes 7-11-15-14-18, 7-11-15-19-18, 4-8-10-16-15-14-19, or 4-8-12-16-20-19-18. These reroutes reduced the sum of critical volumes to around 1700 vphpl at intersection 10 and in none of the other intersection this summation exceeded 1700 vphpl. Along the major corridors a common cycle of 96 seconds and along minor corridors a common cycle of 48 seconds (except for when they intersect a major) were used. Therefore, signals along major corridors were always coordinated while along minors they were coordinated in every other cycle. Similar routes were used in time interval two and three; however, the signal timing parameters were slightly changed.
Table 5.3. PM for IDSTOP vs. RTNTA-IDSTOP

<table>
<thead>
<tr>
<th>Measure of Performance</th>
<th>Statistic</th>
<th>Case-d</th>
<th>RTNTA-IDSTOP</th>
<th>IDSTOP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># of Trips Completed</td>
<td>Min</td>
<td>4227</td>
<td>4568</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>4261</td>
<td>4600</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>4287</td>
<td>4637</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>4313</td>
<td>4668</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>4347</td>
<td>4728</td>
<td></td>
</tr>
<tr>
<td>Average Delay (s)</td>
<td>Min</td>
<td>68</td>
<td>68</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>70</td>
<td>71</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>71</td>
<td>74</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>73</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>75</td>
<td>80</td>
<td></td>
</tr>
<tr>
<td>Vehicles in the Network</td>
<td>Min</td>
<td>728</td>
<td>789</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>737</td>
<td>806</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>745</td>
<td>823</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>754</td>
<td>839</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>773</td>
<td>864</td>
<td></td>
</tr>
<tr>
<td>Average Travel Time (s)</td>
<td>Min</td>
<td>123</td>
<td>122</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10th perc</td>
<td>125</td>
<td>126</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>126</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90th perc</td>
<td>127</td>
<td>131</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Max</td>
<td>130</td>
<td>134</td>
<td></td>
</tr>
</tbody>
</table>

When traffic demand and supply were managed at the same time 4637 trips were completed, see in Table 5.3. This was extremely close to the theoretical upper bound (5.1% below). When the traffic was optimally assigned, the number of completed trips was increased by 8.2% indicating significant improvement in network performance. Average delay and average travel time increased since many more vehicles could travel through the network. Increase in delay does not indicate less efficient network performance since the number of completed trips is increased by 8.2% at the expense of increasing average travel time by only 1.6%.

5.11.5 IDSTOP with time Variant Traffic Demand

In this section a time variant demand is used to study the performance of IDSTOP. It was already established that for a constant traffic demand, IDSTOP solution results in a more efficient network performance than FTNTA-IDSTOP and Direct-CORSIM solutions. Consequently, when demand varies over
time IDSTOP solution has to be better than FTNTA-IDSTOP and Direct-CORSIM solutions since it changes over time. Therefore, IDSTOP is not compared with Direct-CORSIM and FTNTA-IDSTOP when demand varies.

In a one-hour-long study period, traffic demand is gradually increased in the network from 600 vphpl up to 1000 vphpl in increments of 100 vphpl from one time interval to the next one (in four time intervals). Each time interval is five minutes. These high entry volumes are maintained for two time intervals and then gradually decreased to 600 vphpl in increments of 100 vphpl in all corridors of the network except for P-G and B-L. Then entry volumes in these two corridors are reduced to 600 vphpl in two increments. The changes in traffic volume are presented in Figure 5.8.

![Figure 5.8. Traffic demand over time in modified Springfield network](image)

IDSTOP optimized signal timing parameters and traffic assignment in the network. The study period was divided into twelve 5-minute time intervals totaling a study period of 60 minutes. Near-optimal solutions were found for each time interval, and the final state of the network at the end of each time interval was used as initial state of the network for the next time interval. Figure 5.9 presents average traffic demand on the entry links as well as average network-wide queue length over time. As shown, when traffic demand increases over time, average queue length increases in the network. The
highest average queue length in the network was five vehicles which occurred during the fifth time interval when the entry volumes were increased to 1000 vphpl and protected left turn phases were in use. IDSTOP used protected left-turn phases to reduce queue length for left-turn movements that were accumulated in the network during the previous time interval. After processing these left-turners, for the next time interval IDSTOP used 2-phase signal plan.

![Diagram showing traffic demand and network-wide average queue length for modified Springfield network.](image)

**Figure 5.9. Entry traffic demand and network-wide average queue length for modified Springfield network**

IDSTOP changed the cycle length based on traffic demand. When traffic demand (on the average) was 600 vphpl, IDSTOP used a common cycle of 48 seconds in the network and coordinated the signals. IDSTOP responded to the increasing traffic demand by using longer common cycles as the cycle length increased to 148 seconds when traffic demand was 1000 vphpl. As expected, when traffic demand was reduced, IDSTOP’s average cycle length was reduced accordingly.
This significantly increased number of completed trips and reduced average delay in the network. In Figure 5.10 the number of completed trips, average cycle length, average delay, and average travel time are presented for all time intervals. For time intervals 9, and 10, where entry traffic volume in corridors P-G and B-L were 1000 vphpl, and in the rest of entry links 700 and 600 vphpl, respectively, IDSTOP used a common cycle of 96 seconds along the two corridors. It optimized the offsets such that upcoming vehicles drove through the downstream intersection without reducing their speed as the queue was just dissipated. Along other corridors, a common cycle of 48 second were used for all intersections except for when they intersected the major corridors. Therefore, along minor corridors, signals were coordinated in every other cycle.

5.12 Summary

IDSTOP was tested in several steps in this chapter. The results indicated that on the case study network, FTNTA-IDSTOP solution resulted in significantly more efficient network performance than Direct-CORSIM solution as the numbers of completed trips were increased by 2.0% to 18.8% and at the same time average delay was reduced by 10.3% - to 13.7%. Real-time strategy (RTNTA-IDSTOP) as
expected, improved FTNTA-IDSTOP solution. The number of completed trips was increased by up to 1.8% and average delay was reduced by up to 16.5%. Finally demand and supply management improved the number of completed trips by 8.2% for asymmetric partially oversaturated conditions.

At the end, IDSTOP was used to find optimal signal timing parameters and turning percentages for a period of 60 minutes with time variant demand. Results indicated that IDSTOP managed the queues inside the network and dynamically optimized phase sequence, cycle length, green splits, and turning percentages in response to the time variant demand.
CHAPTER 6

EVOLUTIONARY ALGORITHMS

6.1 Introduction

In this chapter, the effects of using different evolutionary algorithms in solving IDSTOP are studied in terms of their runtime. Two families of evolutionary algorithms that are widely used in different fields of science are chosen:

1- Genetic Algorithms (GA)
2- Evolution Strategy (ES)

Genetic Algorithms and specifically simple GA have been extensively used to optimize signal timing parameters in urban networks (e.g. [1], [2], [3], [4], [5]). In this study, three variations of genetic algorithm will be used to solve IDSTOP. These variations are as follows:

d) Simple GA
e) Elitist GA
f) Micro-Elitist GA

Among different Evolution Strategy methods two of them that are widely used in other fields of science are selected:

c) Self-adaptive ES
d) Self-adaptive elitist ES

In the rest of this chapter each method is introduced and important operators and parameters are explained. Then the integration of each model into IDSTOP is explained. The algorithms are used to find optimal signal timing parameters and traffic assignment on a case study network for all four cases of traffic demand patterns (details are available in chapter 5). First it is checked if each algorithm was
able to find a near optimal solution for different cases. Then, their efficiency (in terms of speed of convergence) in finding solutions is compared to each other. In the rest of this chapter, first, each evolutionary algorithm is discussed, their ability to find a near-optimal solution is studied and the efficiency of the algorithms in finding such a solution is compared. Finally, important parameters in calibration of each algorithm are discussed.

6.2 Genetic Algorithm

Genetic Algorithms (GA) are search techniques to find optimal or near-optimal solutions to an optimization or a search problem. GA are global search heuristics and are known to be less likely trapped in a local optimum. GA are a specific class of evolutionary algorithms and use techniques inspired by evolutionary biology like inheritance, selection, crossover, and mutation.

GA are implemented in a computer simulation environment where a population of candidate solutions are created and evolved towards better solutions over different generations. Unlike other well-known optimization techniques that start the search with one feasible solution, GA start the search with several points in the feasible area (a population of candidate solutions). The initial population can be created randomly or by using some heuristics. Each population member is called an individual or a chromosome, and has a fitness value that represents the value of the objective function for that individual. For example, if the objective function is to maximize $f(x) = x^2$, the fitness of one of the individuals say $x = 3$ is $3^2 = 9$. Based on the fitness values, GA stochastically select some individuals of the population where individuals with higher fitness values are more likely to be selected (for a maximization problem). The selected individuals form a mating pool where they are crossed over and mutated to form some new individuals for the new population in the next generation. GA continue to select new individuals as parents until enough individuals for the next generation are created. As soon as a new individual is created the fitness value of that individual is evaluated. It is noted that in this
study, the feasibility of that individual is checked before determining its fitness value. The whole process of selection, crossover, and mutation is continued until the termination criteria are met. Usually a maximum number of generations, or a threshold for the relative difference between the maximum fitness value and average fitness value of a population are chosen as the termination criterion.

Traditionally, binary coding was used to represent each feasible solution in GA however, other methods of coding exist. In binary coding each 0 or 1 of the chromosome is called a genome. Several variations of GA exist. In this study three of them are used to solve IDSTOP: a) simple GA, b) Elitist GA, and c) Micro-Elitist GA. Comprehensive details on GA can be found in Goldberg (1989) [6]. In the rest of this section, a brief review of different GA methods that are used in this study is provided.

6.2.1 Simple Genetic Algorithms

Simple GA use binary coding to represent decision variables. This means that a decision variable in the form of $\boldsymbol{x} = (x_1, x_2, x_3, ..., x_{m-1}, x_m)$ is represented as the following chromosome:

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>. . .</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>. . .</th>
<th>0</th>
<th>. . .</th>
<th>. . .</th>
<th>. . .</th>
<th>. . .</th>
<th>. . .</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$x_2$</td>
<td>$x_3$</td>
<td>. . .</td>
<td>$x_{m-1}$</td>
<td>$x_m$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.1. Binary representation of decision variables in GA.

Simple GA have three operators: Selection, Crossover, and Mutation. Selection operator stochastically selects two parents with bias towards fitter individuals, crossover operator exchanges useful information between the two parents, and mutation operator introduces diversity to the algorithm to search for unsearched parts of the feasible area. These three operators will be explained in details:

6.2.1.1 Selection

Selection is one of the GA operators that leads the search to more desired parts of the feasible area. It simply selects the individuals with higher values; however, the process of selection is stochastic
rather than deterministic. This process is not purely random but, is biased towards individuals with higher fitness values. There are three main variations of selection:

6.2.1.1.1 Proportionate Selection

In proportionate selection, a probability of selection is assigned to each individual. Then based on this probability, the individuals are selected to form the mating pool. The probability of selection for each individual is simply the ratio of the fitness value of that very individual to the sum of the fitness values of the whole population. Thus, an individual with higher fitness value is more likely to be selected. The probability of selection is formulated as follows:

\[
P_i = \frac{f_i}{\sum_{j=1}^{n} f_j}
\]

(6.1)

Where:

- \( P_i \) = is the probability of selecting individual \( i \),
- \( f_i \) = is the fitness value of individual \( i \), and
- \( n \) = is the population size.

To select any of the individuals, a uniformly distributed random number between zero and one is generated. If this random number is between 0 and \( P_1 \), individual number 1 is selected. If this random number is between \( P_1 \) and \( P_2 \), individual number 2 is selected and so on.

Proportionate selection is very easy to implement however, when the difference between the fitness values of individuals is much smaller than the value of fitness, it may not provide good results because it assigns almost similar selection probabilities to all individuals. In this case the search becomes more like a random search. Due to this problem and the fact the fitness values of different solutions of IDSTOP follow the above mentioned trend we will not use proportionate selection in our GA.
6.2.1.1.2 Truncation Selection

Truncation selection is one of the easiest selection methods to implement. In truncation selection, the individuals are sorted based on their fitness value. Based on a selection pressure, \( s \frac{1}{s} \) of the population with highest fitness values are selected. This proportion of the population gets \( s \) copies and forms the mating pool for the next generation. This process is shown in Figure 6.2. Truncation selection is not used in this study since in certain cases it may significantly reduce diversity and result in premature convergence.

![Figure 6.2. Truncation selection.](image)

6.2.1.1.3 Tournament Selection

In tournament selection two individuals are selected randomly. Among these two individuals, the one with highest fitness value is selected as one of the parents. The second parent is selected similarly and then the two parents are crossed over and mutated. Other variations of tournament selection exist. For example after selecting two individuals one may not always choose the one with higher fitness value. On the other hand, a probability could be assigned based on which a parent is selected. For example, with probability of 80% the fitter individual is selected and with the probability of 20% the individual with the lower fitness value is selected. Assigning these probabilities brings more diversity to the search process.
In addition, a selection pressure, \( s \), could be used in tournament selection. This means that instead of randomly selecting two individuals, \( s \) individuals are randomly selected. Then the best individual (based on the fitness value) is selected as one of the parents. The second parent is selected similarly and then crossover and mutation operations are performed.

Tournament selection may be with or without replacement. In tournament selection with replacement, each of the individuals of the population may be selected many times. On the other hand, in tournament selection without replacement, each individual that is selected as one of the parents will be removed from the population and consequently could not be chosen again. Tournament selection without replacement results in more diversity in search process while tournament selection with replacement directs the search towards the areas with higher fitness values more. In this study a tournament selection with replacement is used. In addition, always more than two individuals are selected as parents.

### 6.2.1.2 Crossover

The main purpose of crossover operator is creating new individuals by exchanging information between available individuals. In GA, crossing over two parents leads to two new individuals that could potentially be fitter than their parents. To generate two new individuals by crossover, two parents are randomly selected from the mating pool. Then cross sites are selected and based on them some pieces of each parent are cut and then spliced in the other one. Several variations of cross over exist four of which will be discussed here:

#### 6.2.1.2.1 Single-Point Crossover

In single-point crossover, two parents are randomly selected from the mating pool. The cross location is randomly selected as well. To do so, a uniform random number between 1 and maximum number of cross locations is generated. Then both parents are cut in that location. The two new
individuals are created by keeping the first slice of each parent and attaching the second slice of the other parent. This process is shown in Figure 6.3.

![Crossover location](image)

**Figure 6.3. Schematic single point crossover.**

6.2.1.2.2 Two-Point Crossover

Two-point crossover is very similar to single-point crossover but, instead of cutting and splicing the parents at a single point, they are cut and spliced at two points. For two-point crossover, first two parents are randomly selected. Then two locations for crossover are randomly selected. The new individuals keep the first and last parts of their first parent and the middle part of the other parent. This procedure is shown in Figure 6.4.

![Crossover locations](image)

**Figure 6.4. Schematic two point crossover.**

6.2.1.2.3 Multi-Point Crossover

When the chromosome is long, single-point or two-point crossover may not be able to generate enough diversity for the next population. This may result in being trapped in a local maximum. To overcome this issue, multi-point crossover could be used. It is similar to single-point and two-point crossovers however, instead of crossing over at one or two locations, the new individuals are generated by crossing parents over at several points.

6.2.1.2.4 Uniform Crossover

To introduce even more diversity, uniform crossover could be used. In uniform crossover, two parents are randomly selected from the mating pool. Each bit of the first offspring is randomly selected
from one of the two parents. In other words, to select the first bit, a uniform random number between 0 and 1 is generated. If this number is less than or equal to 0.5 the first bit of the first offspring would be equal to the first bit of the first parent. If the random number is larger than 0.5, the first bit of the first offspring would be equal to the first bit of the second parent. Other bits of the first offspring are generated similarly. To generate the second offspring, the same method will be used. It is noted that instead of using equal probabilities to choose the genomes from first or second parent, they could be chosen with probability \( p \) from the first parent and probability \( q = 1 - p \) from the second parent. In this case, a higher probability could be assigned to the parent with higher fitness value. This variation of uniform crossover is sometimes called parameterized uniform crossover. In this study, uniform crossover with \( p = 0.5 \) is used.

6.2.1.3 Mutation

Mutation is used in GA to introduce more diversity to the search. In addition, if the parents are similar, crossing them over does not produce a new individual. In this case mutation is needed to generate a new offspring. In bitwise mutation, each bit of the chromosome is flipped with respect to the probability of the mutation. That means that, each bit of a chromosome is flipped with probability of \( P_m \) that is probability of mutation. Different methods of mutation exist. But in this study bitwise mutation is used.

6.2.2 Elitist GA

To make sure that from one generation to another GA do not loose fit individuals, elitism can be used. Elitism works better when the population size is large. For small population sizes elitist GA may be trapped in a local maximum. This happens due to using the fittest individual over and over that reduces the diversity and limits the search to a smaller portion of the feasible area. On the other hand, when the population size is large it is possible that either fittest individual are not selected as parents, or they are
destroyed by cross over and/or mutation. Elitist GA avoid this problem. Two methods exist to introduce elitism to GA. In the first one, the fittest or some of the fittest individuals of population in generation $i$ will be selected and without crossover or mutation will be directly put in the population for generation $i + 1$. Doing so insures that the maximum fitness value at generation $i + 1$ will be at least as well as that of generation $i$.

The second method of elitism is quite different. In simple GA two individuals are generated by stochastically selecting two parents, crossing them over, mutating them and put these two newly generated individuals in the next population. In elitist GA, after generating these to individuals, they are compared to their parents. Among these four individuals (two parents and two descendants) the two individuals with highest fitness values are selected and put in the next population. Thus, if the descendants are not as fit as their parents, they will be discarded. The rest of population is generated the same way. This method also ensures that the maximum fitness of population at generation $i + 1$ is at least as well as that in population in generation $i$. However, this method of elitism is known to significantly reduce diversity and result in genetic drift. On the other hand, it could be useful when a quick convergence to a local optimum is needed. In this research the first Elitism method is used.

6.2.3 Micro-Elitist GA

It has been shown that regular GA are useful search techniques in many optimization problems. However, due to evaluating the objective function to determine the fitness value of each individual, regular GA may involve a significant time penalty. This time penalty becomes more serious especially for large populations when a higher number of fitness function evaluations is needed as well as when fitness function evaluation is complicated (e.g. determining the objective function of IDSTOP that involves microscopic simulation run). Small populations can be effectively used in GA if the population is restarted sufficient times [7]. In micro-GA, a small population of size “$n$” is generated and converged. After convergence, “$n - 1$” new individuals are randomly created and plus the best individual resulted
from the previous run of micro-GA, form the initial population. This population is converged and again the best individual is selected to form a new population with “$n - 1$” newly generated individuals. This process is continued until the termination criteria are met. Abu-Lebdeh and Benekohal (1999) have suggested how to choose the population size and number of generations for micro-GA in case of signal timing optimization problem in an arterial.

6.3 Evolution strategies

Evolution Strategies (ES), genetic algorithms, and evolutionary programming are the main three paradigms of Evolutionary Computation (EC). In general, these three methods are based on iterative birth and death, variation, and selection. The first ES had only two rules: 1) slightly change all variables at a time at random, 2) if this set of variables leads to better results keep them otherwise, keep the original ones. As it is apparent from the rules, this ES worked with only two individuals per iteration: one old individual or parent, and one new individual or offspring. This ES was later called 1+1-ES meaning that out of a single parent, one offspring is generated and among these two individuals, the best is chosen. The 1+1-ES with binomially distributed mutations on a two dimensional parabolic ridge was studied by Schwefel in 1965 [9]. The study showed that 1+1-ES is very likely to find a local optimal answer rather than a global one. In this case, larger mutations were needed to escape from this local optimum. To solve this problem, instead of using discrete variables, using continuous variable with Gaussian distributions was suggested. Rechenberg presented approximate analyses of the $1+1-ES$ with Gaussian mutation on two different functions (hyper sphere, and rectangular corridor models). He found that the convergence was inversely proportional to the number of variables; linear convergence might be obtained if the mutation step size was set to the proper order of magnitude; and the optimal mutation strength was in the order of one fifth for both models. In addition, instead of using a single parent, he used $μ$ parents, recombined them, and generated one offspring. He concluded that this
method could speed up the evolution if the speed was measured per generation; and the population might learn by itself how to adjust the mutation step size. This method of ES was called $\mu + 1 - ES$ since among $\mu + 1$ individuals the best $\mu$ individuals were selected or in other words, the worst individual was extinct. Later, $\mu + 1 - ES$ was expanded to $\mu + \lambda - ES$. In this method instead of creating a single offspring out of the $\mu$ parents, $\lambda$ descendants are created. Then among these $\mu + \lambda$ individuals the fittest $\mu$ individuals are selected to form the next population. Another variation of ES with $\mu > 1$ parents and $\lambda > 1$ descendants exists. In this method, after creating the new $\lambda$ descendants, all parents are discarded. Out of the $\lambda$ descendants, the fittest $\mu$ are chosen to form the next population. Thus, $\lambda$ has to be strictly larger than $\mu$. This method is called $\mu, \lambda - ES$. In general, $\mu + \lambda - ES$ and $\mu, \lambda - ES$ generate better results than $1 + 1 - ES$ and $\mu + 1 - ES$ do.

All variations of ES with $\mu > 1$ parents and $\lambda > 1$ descendants have three different operators that are recombination, mutation, and selection. ES has the following steps:

4) Initialization: the first population is generated randomly or by means of some heuristics

5) Regeneration: next population is produced

   1-4) Recombination: randomly select $\rho$ parents and recombine them to generate a new offspring

   1-5) Mutation: mutate the new offspring

   1-6) Fitness function evaluation: evaluate the fitness of the generated offspring

6) Selection: select new parents with respect to “+” or “,” scenario

7) Termination criteria: stop if termination criteria are met otherwise continue by going to step 1

ES could be self-adaptive. This means that as the populations evolve, the strategy parameters evolve as well. This is done by coupling the endogenous strategy parameters with the objective
parameters. In other words, the decision vector contains object parameters as well as endogenous strategy parameters. This is shown in Equation 6.2.

\[
\vec{a}_k = (y_{1k}, y_{2k}, ..., y_{Dk}, s_{1k}, s_{2k}, ..., s_{Dk})
\] (6.2)

Where: \(y_{ij}\): the \(i^{th}\) component of decision variable \(j\), and \(s_{ij}\): the \(i^{th}\) component of endogenous strategy parameter \(j\).

More information on ES could be found in Schwefel (1965). ES operators are described in the rest of this section.

6.3.1 **Recombination**

In recombination, \(\rho \geq 1\) individuals are selected among parents and then recombined. When \(\rho = 1\), the new offspring is simply equal to its parent meaning that no recombination is done. There are two main methods of recombination: discrete, and intermediate.

Assume that a parental vector (a decision variable that is selected to be one of the \(\rho\) parents) is: \(\vec{a} = (a_1, a_2, ..., a_D)\), and the recombinant is \(\vec{r} = (r_1, r_2, ..., r_D)\).

6.3.1.1 **Discrete Recombination**

In discrete recombination \(\rho\) parents are randomly selected. Among the \(\rho\) parent one is randomly selected. The first component of the offspring will be equal to the first component of this randomly selected individual. To select the second component of the offspring, another parent is randomly selected and the value of its second component is chosen as the value of second component of the offspring. The value of each component is selected the same way.
Discrete recombination is shown by Equation 6.3:

\[(r)_k = (a_{mk})_k \quad \text{with} \quad m_k = \text{Rand} \{1,2,...,\rho\} \quad (6.3)\]

### 6.3.1.2 Intermediate Recombination

In intermediate recombination, \(\rho\) parents are randomly selected. The offspring will simply be the center of mass of the \(\rho\) parent vectors. In other words, each component of the offspring will be equal to the average of that component of the \(\rho\) parents:

\[(r)_k = \frac{1}{\rho} \sum_{m=1}^{\rho} (a_m)_k \quad (6.4)\]

### 6.3.2 Mutation

Mutation is the main source of genetic variation in ES. The design of mutation operator is problem dependent. It is suggested that each mutation operator has to have reachability, unbiasedness, and scalability [10].

Reachability means that from each parental state, any other state should be reachable in a finite number of mutations. Mutation operator should be completely unbiased toward individuals with higher fitness values. Instead, selection operator is biased towards fitter individuals. Scalability means that mutation step size should be tunable in order to adapt to the properties of the fitness landscape.

In general, the new individual, \(\tilde{y}\), is generated by mutating the recombinant, \(y\), as shown in Eq.6.5:

\[\tilde{y} = y + z \quad (6.5)\]
To determine \( z \), three different equations may be used:

\[
    z = \sigma \times (N_1(0,1), ..., N_D(0,1))
\]  

(6.6)

This method of mutation is called single component mutation that results in concentric spheres around the parental state \( y \). This operator is easy to use since it has only one endogenous strategy parameter; however, in some situations it is beneficial to have a vector of endogenous strategy parameters. For those cases, \( z \) is determined using the Equation 6.7:

\[
    z = (\sigma_1 \times N_1(0,1), ..., \sigma_D \times N_D(0,1))
\]  

(6.7)

This equation results in ellipsoidal surfaces around the parental state \( y \).

In the most general case, when the ellipsoid needs to be arbitrary rotated in the search space, Equation 6.8 should be used to determine \( z \).

\[
    z = M(\sigma_1 \times N_1(0,1), ..., \sigma_D \times N_D(0,1))'
\]  

(6.8)

Where \( M \) is an orthogonal rotation matrix. This matrix introduces correlations between the components of \( z \).

### 6.3.3 Selection

The selection operator \( a_{m; q} \) takes the \( m^{th} \) best individuals out of a population of size \( q \). There are two variations of selection based on using “plus” or “comma” strategies. In case of using “plus” strategy, after generating \( \lambda \) descendants out of \( \mu \) parents, the best \( \mu \) are selected among \( \mu + \lambda \) individuals. In case of “comma” strategy, after generating the \( \lambda \) descendants, the best \( \mu \) are selected among the \( \lambda \) descendants.
6.4 Efficiency of Different Algorithms

In this section the efficiency of different ES in solving IDSTOP is compared to each other. For this purpose, all five EA were used to solve IDSTOP in the case study network for all four demand patterns. Details on the case study and demand patterns are available in Chapter 5. All EA were provided with identical computational resources to make sure that possible changes in the solutions are only due to using different algorithms and not different amount of allocated resources.

For each EA, the number of Fitness Function Evaluations (FFE) that was required to reach to a certain level of the theoretical upper-bound was recorded. The following eight levels of the upper bound were used:

1- 80% of the upper-bound,
2- 82.5% of the upper-bound,
3- 85% of the upper-bound,
4- 87.5% of the upper-bound,
5- 90% of the upper-bound,
6- 92.5 of the upper-bound,
7- 95% of the upper-bound, and
8- 97.5% of the upper-bound

Note that the upper bound was found for the number of completed trips in the network. Also note that this number of FFE is required to improve the average number of completed tips of the entire population to that level and not only the fittest individual. As such, while the number of completed trips of a population is for example at 92.5% of the theoretical upper-bound, the maximum number of completed tips of that population might be at a higher level. The comparison process has the following steps:
1- Simple GA (SGA) vs. Elitist Simple GA (ESGA)

2- ES vs. ES+

3- ES+ vs. ESGA

4- ES+ vs. Micro-Elitist GA (MEGA)

There are several points that need to be discussed before starting the comparisons. SGA was not able to improve the average number of completed tips to 97.5% of the upper-bound in any of the cases, see in Table 6.1. In addition, in symmetric oversaturated conditions, it could not reach to 92.5% of the upper-bound. Finally, in most of the cases, it required more FFE to reach to different levels of the upper bound than all other EA (except for 92.5% and 95% of the upper-bound in symmetric undersaturated condition when compare to ESGA). Therefore, among all five EA, SGA appears to have the least efficient performance on the case study.

Similar to SGA, ES was also not able to reach to 97.5% of the upper-bound in any of the four demand patterns. However, unlike SGA, ES was among the fastest algorithms to reach up to 95% of the upper-bound.

None of the algorithms reached 97.5% of the upper bound in all four cases of the demand patterns. ESGA could not reach it in two cases: symmetric undersaturated and symmetric oversaturated demand conditions. In addition, in symmetric oversaturated conditions it could not reach to 95% of the upper-bound. On the other hand, ES+ and MEGA reached up to 97.5% of the upper bound in three out of the four demand cases indicating their ability for further improvements in the signal timing parameters for the case study network.

Looking at all three variations of GA reveals very interesting findings. In cases with symmetric demand, for most of different levels of the upper-bounds, GA variations are slower than ES variations. The difference is more pronounced in symmetric oversaturated conditions in which not only GA
variations are slower for all different levels, they also never reach to 95% of the upper-bound. ES variations reach this level fairly quickly. In addition, in symmetric oversaturated conditions, ES+ reaches to 97.5% of the upper-bound quickly. One reason for this is that ES operators are more likely to create equal green times for different directions compared to GA operators. This results in longer time for GA variations to reach to these approximately equal green times that are required for symmetric conditions. Therefore, ES variations could find high quality answers much faster than GA variations.

On the other hand, in asymmetric conditions, in reaching to 97.5% of the upper-bound, ESGA and MEGA are considerably faster than ES variations since their operators can create different green times easier than ES operators.

6.4.1 SGA vs. ESGA

In symmetric undersaturated conditions, SGA and ESGA needed the same number of FFE to improve the population to reach to 90% of the theoretical upper-bound. For 92.5% and 95%, SGA needed fewer FFE (3600 vs. 4050 and 7200 vs. 10350, respectively). None of the algorithms could reach to 97.5% of the theoretical upper-bound. In this case, SGA was slightly more efficient than ESGA in reaching the higher levels of the theoretical upper-bound.

In Symmetric oversaturated conditions, ESGA required fewer number of fitness function evaluations for all levels of fitness function evaluations except for 85% for which both algorithms needed 6750 FFE. In addition, ESGA was capable to improve the average number of completed trips for the population up to 92.5% while SGA could improve it up to 90%. Therefore, in this case, ESGA clearly outperformed SGA.

In asymmetric undersaturated conditions, both algorithms required similar number of FFE to reach to 87.5% of the upper-bound. For higher levels, ESGA required slightly fewer number of FFE (1350 vs. 1800, 3150 vs. 3600, and 5400 vs. 6300, for 90%, 92.5%, and 95% of the upper-bound, respectively).
In addition, ESGA could reach up to 97.5% of the upper-bound while SGA reached up to 95% of that. As such, in asymmetric undersaturated conditions ESGA outperformed SGA.

Finally, for asymmetric partially oversaturated conditions, ESGA required fewer number of FFE to reach to all different levels of the upper-bound (except for 87.5% where both required 3150 FFE) than SGA. In addition, ESGA could improve the average number of completed trips up to 97.5% of the upper-bound while SGA could improve it up to 95%. Therefore, in asymmetric partially oversaturated conditions, ESGA outperformed SGA.

In summary, our findings indicated that in oversaturated conditions (cases b and d) ESGA consistently outperforms SGA. In undersaturated conditions (cases a and c), both algorithms were similar in reaching up to 90% of the upper-bound. In symmetric demand condition (case a) SGA was more efficient in reaching to 92.5% and 95% of the upper-bound while in asymmetric condition (case c) ESGA was more efficient in reaching 92.5%, 95%, and 97.5% of the upper bound. Considering the small difference between the two algorithms in symmetric undersaturated conditions, overall, ESGA was more efficient than SGA.

6.4.2 ES vs. ES+

In Symmetric undersaturated conditions, both algorithms required similar number of FFE to reach up to 92.5% of the upper-bound. ES+ required considerably fewer FFE than ES to reach to 95% of the upper-bound (1865 vs. 7715). None of the algorithms could reach to 97.5% of the upper-bound.

In symmetric oversaturated conditions, both algorithms were the same efficient in reaching up to 85% of the upper-bound. ES+ was more efficient than ES in reaching to higher levels of the upper bound except for 90% for which both EA needed 1415 FFE. It is noted that ES+ reached 97.5% of the upper-bound fairly quickly while ES could not reach it. Overall, in symmetric oversaturated conditions, ES+ was more efficient than ES in reaching higher levels of the upper-bound.
Table 6.1. Number of Fitness Function Evaluations Required to Reach a Certain Level of the Theoretical Upper Bound.

<table>
<thead>
<tr>
<th>Demand Pattern</th>
<th>% of the Theoretical Upper-Bound</th>
<th>Number of Required Fitness Function Evaluations to Reach a Certain Level of the Upper-Bound</th>
<th>Simple GA</th>
<th>Elitist GA</th>
<th>Micro-Elitist GA</th>
<th>ES</th>
<th>ES+</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symmetric Undersaturated Demand Pattern: Case a</td>
<td>80</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>82.5</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>85</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>87.5</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>1350</td>
<td>1350</td>
<td>225</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>92.5</td>
<td>3600</td>
<td>4050</td>
<td>825</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>95</td>
<td>7200</td>
<td>10350</td>
<td>8625</td>
<td>7715</td>
<td>1865</td>
<td></td>
</tr>
<tr>
<td></td>
<td>97.5</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td></td>
</tr>
<tr>
<td>Symmetric Oversaturated Demand Pattern: Case b</td>
<td>80</td>
<td>5400</td>
<td>4050</td>
<td>375</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>82.5</td>
<td>6300</td>
<td>4500</td>
<td>1425</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>85</td>
<td>6750</td>
<td>6750</td>
<td>1500</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>87.5</td>
<td>9450</td>
<td>8100</td>
<td>2625</td>
<td>1415</td>
<td>965</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>14400</td>
<td>9900</td>
<td>2850</td>
<td>1415</td>
<td>1415</td>
<td></td>
</tr>
<tr>
<td></td>
<td>92.5</td>
<td>&gt;22500</td>
<td>12600</td>
<td>4125</td>
<td>2765</td>
<td>1865</td>
<td></td>
</tr>
<tr>
<td></td>
<td>95</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>7265</td>
<td>2765</td>
<td></td>
</tr>
<tr>
<td></td>
<td>97.5</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td>7715</td>
<td></td>
</tr>
<tr>
<td>Asymmetric Undersaturated Demand Pattern: Case c</td>
<td>80</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>82.5</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>85</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>87.5</td>
<td>450</td>
<td>450</td>
<td>75</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>1800</td>
<td>1350</td>
<td>300</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>92.5</td>
<td>3600</td>
<td>3150</td>
<td>450</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>95</td>
<td>6300</td>
<td>5400</td>
<td>3150</td>
<td>6815</td>
<td>965</td>
<td></td>
</tr>
<tr>
<td></td>
<td>97.5</td>
<td>&gt;22500</td>
<td>8100</td>
<td>16200</td>
<td>&gt;22500</td>
<td>&gt;22500</td>
<td></td>
</tr>
<tr>
<td>Asymmetric Partially Oversaturated Demand Pattern: Case d</td>
<td>80</td>
<td>1350</td>
<td>900</td>
<td>225</td>
<td>65</td>
<td>65</td>
<td></td>
</tr>
<tr>
<td></td>
<td>82.5</td>
<td>2250</td>
<td>1800</td>
<td>225</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>85</td>
<td>3150</td>
<td>2700</td>
<td>450</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>87.5</td>
<td>3150</td>
<td>3150</td>
<td>900</td>
<td>515</td>
<td>515</td>
<td></td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>4500</td>
<td>3600</td>
<td>2250</td>
<td>965</td>
<td>1415</td>
<td></td>
</tr>
<tr>
<td></td>
<td>92.5</td>
<td>7200</td>
<td>4050</td>
<td>5850</td>
<td>4115</td>
<td>1865</td>
<td></td>
</tr>
<tr>
<td></td>
<td>95</td>
<td>10350</td>
<td>7200</td>
<td>6750</td>
<td>17165</td>
<td>3215</td>
<td></td>
</tr>
<tr>
<td></td>
<td>97.5</td>
<td>&gt;22500</td>
<td>9000</td>
<td>19350</td>
<td>&gt;22500</td>
<td>13565</td>
<td></td>
</tr>
</tbody>
</table>
In asymmetric undersaturated conditions, both algorithms required similar number of FFE to reach up to 92.5% of the upper-bound. For higher levels ES+ was more efficient than ES. ES could not reach 97.5% of the upper-bound. It is noted that findings for this condition was very similar to symmetric undersaturated conditions.

Finally in asymmetric partially oversaturated conditions both algorithms were similar in reaching up to 87.5% of the upper-bound. At 90% ES was more efficient; however, at 92.5%, 95%, and 97.5% ES+ was the more efficient one. ES could not reach 97.5% of the upper-bound. Overall, ES+ was more efficient than ES in reaching higher levels of the upper-bound in asymmetric partially oversaturated conditions.

Our findings indicated that overall, ES+ was more efficient than ES in reaching higher levels of the upper-bound. The difference was smaller in undersaturated conditions but become more noticeable in oversaturated conditions.

6.4.3 ES+ vs. ESGA

Since in general, ES+ and ESGA were more efficient than ES and SGA, respectively, they are compared to each other in this section to choose the more efficient algorithm.

In symmetric undersaturated conditions, ES+ consistently required fewer number of FFE to reach to all different levels of the upper-bound. It is noted that neither of the algorithms could reach to 97.5% of the upper-bound.

In symmetric oversaturated conditions, ES+ consistently required significantly fewer number of FFE to reach to all different levels of the upper-bound than ESGA. Therefore, for symmetric oversaturated conditions ES+ was more efficient than ESGA.
In asymmetric undersaturated conditions, ES+ consistently outperformed ESGA in all different levels except for 97.5% of the upper-bound for which, ESGA required fewer number of fitness function evaluations (8100) compared to ES+ (22115).

Similarly, for asymmetric partially oversaturated conditions, ES+ is considerably more efficient than ESGA up to 95% of the upper-bound; however, ESGA is significantly faster in reaching to 97.5% of the upper-bound (9000 FFE for ESGA vs. 13565 for ES+).

Our findings indicated that when demand is symmetric (cases a and b) ES+ is consistently more efficient than ESGA in reaching all levels of the upper-bounds. When demand is not symmetric, ES+ is again more efficient in reaching up to 95% of the upper-bound; however, it was outperformed by ESGA in reaching to 97.5% of the upper-bound.

6.4.4 ES+ vs. MEGA

In symmetric undersaturated conditions, ES+ outperforms MEGA in reaching up to 95% of the upper-bound except for 90% for which MEGA required fewer number of FFE (225 vs. 515). However, MEGA could reach to 97.5% of the upper-bound while ES+ never reached that.

In symmetric oversaturated conditions, ES+ consistently outperformed MEGA in reaching all levels of the upper-bound except for reaching to 80% of that for which MEGA required 375 FFE and ES+ required 515 FFE.

In asymmetric undersaturated conditions, ES+ was consistently more efficient than MEGA for different levels up to 95% of the upper bound except for 92.5% for which MEGA required 450 and ES+ required 515 FFE. In reaching to 97.5% of the upper-bound, MEGA (16200 FFE) was more efficient than ES+ (22115 FFE).

In asymmetric partially oversaturated conditions, ES+ consistently outperformed MEGA for all levels except for 82.5 and 85% of the upper-bound for which MEGA required fewer number of FFEs.
6.4.5 Summary

In general ES+ outperformed the rest of algorithms in reaching most different levels of the upper-bounds. In addition, ES+ was very efficient in oversaturated conditions especially when demand was symmetric.

MEGA was very quick in early improvements in the fitness value. However, in most of the cases it was outperformed by ES+ in reaching higher levels of fitness value except for asymmetric undersaturated condition.

In symmetric oversaturated demand conditions, ES+ was consistently faster than ESGA. This was the case for other cases as well; however, in reaching to 97.5% of the upper-bound ESGA was much more efficient than ES.

SGA was the least efficient algorithm among the other in all four different demand patterns. In general, it was slowest in reaching different levels of the upper-bound except for two levels in symmetric undersaturated condition. In addition, in none of the cases it could reach to 97.5% of the upper-bound.

6.5 References


CHAPTER 7
A PROGRAM FOR OPTIMAL LEFT TURN MANAGEMENT

7.1 Introduction

In oversaturated conditions, left turners can hardly find gaps in the opposing traffic that are large enough to perform their left turn maneuvers. As such, if no protected left turn phase is used, left turners may encounter longer delays. In addition, left turn queues may start to grow during several cycles and block through traffic lanes and create starvation and waste green duration, see in Figure 7.1 for eastbound.

![Figure 7.1. Starvation for east-bound through](Image)

Adding a protected left-turn phase helps solve this problem; however, it increases the lost time due to the added phase, and reduces the share of through movements (that usually have considerably higher demand than left-turns) from total green time in a cycle. This may result in longer travel times in the network and may reduce its capacity.

It is also possible to prohibit left turns at certain intersections of the network (especially when demand is not high) and reroute the left turners through other routes. Although travel distance for
these vehicles will be increased, total savings in travel time (network-wide) as well as total improvement in the number of completed trips may be large enough to justify prohibiting some left turns.

Prohibiting left turns at some intersections may sound very promising but, it has to be done very carefully. It may significantly improve traffic condition at a congested intersection. However, this might be at the cost of deteriorating traffic condition at other areas of the network such that overall network performance worsens. This may happen by overcrowding a downstream intersection by sending too many vehicles, un-coordinating the signals by changing the cycle lengths and the offsets, and not properly rerouting the left turners to their destinations. In addition, which intersection to select for prohibiting the left turns is extremely important.

The main objective of this chapter is to introduce a program to intelligently select some intersections for left turn prohibition, formulate this problem, and develop a solution method for it. This program is aimed at maximizing the total number of completed weighted trips (weighted by the lengths of the shortest path from each origin to each destination node) in the network. Its decision variables are either to prohibit or to allow the left turns at each direction of each intersection of the network. It is noted that in each direction, left turn movement is either allowed or prohibited at both approaches together (i.e. eastbound and westbound together as well as northbound and southbound together).

It is also noted that value of the objective function for a candidate solution (i.e. a vector indicating in which intersections left turns are either allowed or prohibited) depends on the signal timing parameters used inside the network. In other words, for a specific solution, depending on the signal timing parameters, numerous values for the objective function exists (one for each set of signal timing parameters).

Efficient signal timing results in a larger value for the objective function, while non-efficient signal timing results in a smaller value. For each candidate solution (i.e. at which intersection left turns
are prohibited), those values of the objective function that correspond to a non-efficient set of signal timing parameters are not of interest since they do not lead to the most efficient network performance. On the other hand, for the same candidate solution, those values of the objective function that correspond to the most efficient set of signal timing parameters are of significant interest since they might result in the best network performance possible. Therefore, to accurately determine the value of objective function for each candidate solution, near-optimal signal timing parameters and their corresponding routes needs to be used. This involves solving IDSTOP for each candidate solution.

The rest of this chapter introduces the formulation and solution technique that were developed for the problem. The algorithm is tested on a case study network, the findings are discussed and finally, the concluding remarks are presented.

### 7.2 Optimal Left Turn management Problem Formulation

As mentioned before, Optimal Left Turn Management Problem (OLTMP) is formulated as a maximization program. The objective function aims at maximizing total number of completed weighted trips in the network. The objective function is formulated as follows:

$$
\text{Maximize } \sum_{r \in R} \sum_{s \in S} \sum_{t \in T_i} \eta_{rs} \chi_{rs}^t (a_{ik}^t, \phi_i^t, C_i^t, s_{i,k}, sof_{i,j}^t, y_{rs,ij}^t), \quad \forall t \in T_i, \forall i, j \in I/(R,S), \forall k \in K_i
$$

Where:

- $T_i =$ set of discrete time intervals (in the order of minutes)
- $I =$ set of all intersections of the network
- $K_i =$ set of all phases available at intersection $i$
- $\chi_{rs}^t =$ number of completed trips from source node $r$ to sink node $s$ during time interval $t$
- $\eta_{rs} =$ length of the shortest distance path from source node $r$ to sink node $s$
\( \partial_{ik}^t \) = decision on prohibiting or allowing left turn at intersection \( i \) for direction \( k \) at time \( t \)

\( \partial_{ik}^t \in \{0,1\} \quad \forall t \in T_i, \forall i, k \in D \)

\( \varnothing_i^t \) = number of phases at intersection \( i \) at time interval \( t \)

\( C_i^t \) = cycle length of intersection \( i \) at time interval \( t \)

\( s_{ik}^t \) = split for green for phase \( k \) of intersection \( i \) at time interval \( t \)

\( sof g_i^t \) = start time of the first phase of intersection \( i \) at time interval \( t \)

\( y_{rs,ij}^t \) = turning traffic volume at upstream intersection \( i \) moving towards downstream intersection \( j \) on a path from source node \( r \) to a sink node \( s \) at time step \( t \)

The problem has two sets of constraints. The first set indicates that each decision variable has to be either 1, meaning that left turn is allowed, or 0, meaning that left turn is prohibited as shown below:

\[ \partial_{ik}^t \in \{0,1\} \quad \forall t \in T_i, \forall i, k \in D \]  \hspace{1cm} (7.2)

Where:

\( I \) = set of all intersections of the network where it is physically possible to make a left turn

\( D \) = set of traffic directions at each intersection, east-west or north-south directions

The second constraints set enforces that the value of the objective function is obtained when near-optimal signal timing parameters are used for variables \( \varnothing_i^t, C_i^t, s_{ik}^t, sof g_i^t \), and \( y_{rs,ij}^t \). This can be achieved by solving IDSTOP for each set of decision variables on left turn strategies. Details on IDSTOP formulation are available in chapter 3. In summary, the formulation is as follows:

\[
\text{Maximize} \sum_{r \in R} \sum_{s \in S} \eta_{rs} x_{rs} \tilde{X}_{rs}(\partial_{ik}^t, \varnothing_i^t, C_i^t, s_{ik}^t, sof g_i^t, y_{rs,ij}^t), \quad \forall t \in T_i \\
\text{s.t.} \]  \hspace{1cm} (7.1)
\[ \theta_{ik}^t \in \{0,1\} \quad \forall t \in T_i, \forall i \in I, k \in D \]  \hspace{1cm} (7.2)

Where \( \theta_{ik}^t, C_{ik}^t, s_{ik}^t, sof_{ik}^t, \) and \( y_{rs,ij}^t \) are found by solving IDSTOP.

There are two main reasons for not optimizing the decisions on the left turn policies and signal timing parameters at the same time:

a) Simultaneously optimizing both sets of decision variables significantly changes the shape of the objective function. The change in the shape is very likely to produce numerous local optimal points that can potentially make finding a global solution very hard.

b) Optimizing both sets of decision variables at the same time significantly enlarges the decision space. This enlarged decision space is very hard to be intelligently searched. This can make finding a global solution almost impossible.

### 7.3 Solving OLTMP

Similar to IDSTOP, there is no closed-form formulation to represent the objective function of OLTMP in terms of its decision variables. Therefore, optimization techniques that rely on knowing such relationships and the structure of the objective function cannot be used to solve the problem. In addition, the solution space is large. For each intersection, at most two decision variables exist, each of which can take two values. This means that the solution space of each intersection has at most four elements. As a result, the solution space for a network of \( n \) intersection has a decision space as large as \( 4^n \). This solution space can be large enough to make traditional search method such as exhaustive search or dynamic programming unsuccessful especially when the number of intersections increases. Therefore, a meta-heuristic approach is developed to solve the problem.

As established in chapter 6, Micro-Elitist GA and ES+ were the two algorithms that outperformed the rest of algorithms in reaching to most levels of the upper-bound. Moreover, since the
decision variables of OLTMP can take only zero or one, binary coding (as used in Micro-Elitist GA) is very suitable to represent the decision variables. Therefore, Micro-Elitist GA is used to solve the problem.

Figure 7.2. Solution process for solving OLTMP

The first population can be generated either randomly or by means of some heuristics. To get the fitness of each individual, the decisions on prohibiting or allowing left turns at each direction of all intersections are transmitted to IDSTOP. IDSTOP codes CORISM input file with prohibited left turns and optimizes the signal timing parameters and the turning percentages. This optimized decision variables
and their corresponding objective function value are used in Micro-Elitist GA as the fitness associated with the decision variable that were just created. This process is shown in Figure 7.2.

Solving OLTMP requires a lot of IDSTOP runs. As mentioned in previous sections, solving IDSTOP is by itself very time consuming. Therefore, it is extremely important to solve both OLTMP and IDSTOP very efficiently. Micro-Elitist GA (which was fast in convergence), as mentioned before, was used to solve OLTMP. As shown in Chapter 6, ES+ was the quickest algorithm in reaching to most levels of the upper-bound in solving IDSTOP. As such, it was used to solve IDSTOP.

It is not needed to run ES+ for 50 generations to solve IDSTOP. In fact, ES+ found a solution that was at most less than 5% below our theoretical upper-bound for the objective function in less than 6 generations for all four case studies. As a result, to solve IDSTOP, ES+ was used and as soon as it reached 95% of the upper-bound it was stopped. In addition a maximum number of 10 generations was enforced for those cases where 95% of the upper-bound was not reachable within 10 generations. This could happen due to using very unrealistic decisions on prohibiting or allowing left turns in the network. However, these solutions will not be selected to generate new solutions due to their low fitness value.

To further speed up IDSTOP’s run time, instead of making 10 replications to obtain the fitness value of each candidate solution, only 5 replications were made. With 5 replications, a maximum of 5% error in estimating the fitness value is tolerated. These two actions together, reduced the runtime of OLTMP by a factor of at least 10.

### 7.4 Case Study Network

OLTMP was tested on modified downtown Springfield network for symmetric oversaturated condition, see in Figure 7.3. For this case, traffic demand at all entry links was 1000 vphpl, and it was assumed that 10% of the traffic in the left-most lane turns left, 10% of the traffic in the right-most lane turns right, and the remaining of traffic goes straight. Details on the case study network and the demand
pattern are available in chapter 5. Symmetric oversaturated demand pattern was selected since the main focus of this chapter is to improve traffic condition in oversaturated conditions. In fact, in undersaturated condition, there is no need to prohibit the left turns and reroute the vehicles inside the network.

Figure 7.3: Modified Springfield network

7.5 Numerical Findings

In the case study network at intersections number 1, 2, 5, 6, 9, and 10, only two phases were possible since both directions were one-way, see in Figure 7.3. Therefore, there was no decision variable associated with these intersections. At intersections number 3, 4, 7, 8, 11, 12, 13, 14, 17, and 18, only
one of the streets was two-way. As such only one decision on prohibiting or allowing left turns was possible at these intersections. At intersections number 15, 16, 19, and 20, since both streets were two-way, two decisions on left-turns were possible. As such, the decision variable of OLTMP includes a total of $10 \times 1 + 4 \times 2 = 18$ components for the case study network. It is noted that a left turn in a one-way street is never prohibited since this maneuver is performed simultaneously with through movement. Each component of the decision vector is coded using a single bit. Therefore, the chromosome is 18 bits long. As a result, a population size of $\sqrt{18} \approx 4$ was used in Micro-Elitist GA. The number of generations within each epoch was 5 and a total of 10 epochs was used.

In order to study the effects of OLTMP on network performance, OLTMP was applied to our case study network for symmetric oversaturated demand conditions. Its solution was compared to a solution for which signal timing parameters were optimized without optimizing left turn policies, hereafter called No-OLTMP. To make statistical comparisons, both solutions were simulated in CORSIM with 250 replications to account for internal variability and increased accuracy in estimating different PM (i.e. total delay, average speed, travel time, and number of completed trips).

It is noted that for symmetric oversaturated demand pattern, 10% of the vehicles in the left-most lane made a left turn. To study the effects of different left turn percentages on OLTMP solution eight different left turn percentages were used for the case study for each of which two solutions were found: a) No-OLTMP solution (i.e. signal timing parameters were optimized assuming left turns were always allowed), and b) OLTMP solution (i.e. OLTMP was solved to optimize left turn policies in the network.). The following eight left turn percentages were used:

1- 2.5% left turns,
2- 5% left turns,
3- 7.5% left turns,
Findings are shown in Table 7.1. For each left turn percentage, left turn volume per cycle is found by multiplying the capacity of a lane (876 vphpl found in Chapter 5) by left turn percentage, and dividing the outcome by the number of cycles in each hour (average cycle length was approximately 120 s). When left turn percentage is low (i.e. 2.5%, 5%, and 7.5%) OLTMP did not prohibit the left turns in the network. In addition, never a protected left turn phase was used in the network. The reason was that for 2.5%, 5%, and 7.5% the number of left turners was on the average 0.7, 1.5, and 2.2 per cycle, respectively. Therefore, the left turners could perform their left turn maneuver during the yellow indication for through movement. Therefore, for low left turn percentages (2.5%, 5%, and 7.5%) neither a left turn phase nor prohibiting left turns was needed.

On the other hand, when left turn percentage was increased to 10%, 12.5%, 15%, and 17.5%, OLTMP prohibited the left turns at one or two intersections of the network. By doing so, as presented in table 7.1 OLTMP reduced delay, increased average speed, reduced travel time, and increased number of completed trips compared to No-OLTMP solutions. All differences were statistically significant with \( \alpha = 0.05 \) (p-values were always smaller than 0.05) indicating that optimizing left turn policies significantly improved network performance.

For 10% left turn, when left turn policies were not optimized, 5667 trips were completed in the network with a total delay of 220.11 hours. The average speed was 9.46 mph and total travel time was 321.48 hours in the network. When left turn policies were managed, left turns were prohibited at both
directions of intersection 15 and vehicles were rerouted through their destinations. Due to left turn prohibition, left turn volume increased at intersections 19 and 11 for which protected left turn phases were used. Managing the left turns increased the number of completed trips by 63 vehicles (it reached 5730 vehicles) which was very close to the theoretical upper-bound. OLTMP also yielded lower delay, higher average speed, and lower travel time in the network (all statistically significant). This indicated that when 10% of traffic turned left in symmetric oversaturated conditions, managing the left turns significantly improved network performance.

Table 7.1. PM for OLTMP and LT Allowed Policy

<table>
<thead>
<tr>
<th>Left Turn %</th>
<th>Strategy</th>
<th>Performance Measure</th>
<th>Left Turn Volume per Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Delay (h)</td>
<td>Average P-value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Average Speed (mph)</td>
<td>Average P-value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Travel Time (h)</td>
<td>Average P-value</td>
</tr>
<tr>
<td></td>
<td></td>
<td># of Trips (veh)</td>
<td>Average P-value</td>
</tr>
<tr>
<td>2.5% *</td>
<td>No OLTMP</td>
<td>191.39</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>10.61</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>296.08</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5872</td>
<td>N/A</td>
</tr>
<tr>
<td>5% *</td>
<td>No OLTMP</td>
<td>205.28</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>10.1</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>309.37</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5847</td>
<td>N/A</td>
</tr>
<tr>
<td>7.5% *</td>
<td>No OLTMP</td>
<td>214.17</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>9.77</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>317.6</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5780</td>
<td>N/A</td>
</tr>
<tr>
<td>10.0%</td>
<td>No OLTMP</td>
<td>220.11</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>9.46</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>321.48</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5667</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>12.5%</td>
<td>No OLTMP</td>
<td>238.04</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>8.89</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>338.26</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5605</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>15.0%</td>
<td>No OLTMP</td>
<td>249.41</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>8.46</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>347.32</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5458</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>17.5%</td>
<td>No OLTMP</td>
<td>255.84</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>8.21</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>352.05</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5363</td>
<td>0.0178</td>
</tr>
<tr>
<td>20% *</td>
<td>No OLTMP</td>
<td>259.7</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td>OLTMP</td>
<td>7.46</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>357.818</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5297</td>
<td>N/A</td>
</tr>
</tbody>
</table>

OLTMP did not prohibit any left turns in the network

When 12.5% of traffic turned left and left turns were not managed, protected left turn phase was used at intersections 3, 4, 7, 8, 11, 12, 13, 14, 15, 16, 17, 18, 19, and 20. A total of 5605 trips were completed in the network with a total delay of 238.04 hours, see in Table 7.1. OLTMP prohibited both left turns at intersection 15, reduced the number of phases to two at that intersection, and rerouted
vehicles in the network. This resulted in longer left turn green times at intersection 19 and 11 due to additional left turn volume. Managing the left turns increased the number of completed trips by 45 vehicles and significantly reduced total delay, increased average speed, and reduced total travel time. These indicated that OLTMP significantly improved network performance efficiency for 12.5% left turn in symmetric oversaturated conditions.

For 15% left turn, when left turns were not managed, protected left turn phase was used at intersections 3, 4, 7, 8, 11, 12, 13, 14, 15, 16, 17, 18, 19, and 20. A total of 5458 trips were completed in the network with a total delay of 249.41 hours. OLTMP prohibited the left turns at intersections 15 and 14, removed the protected left turn phase at these two intersections and rerouted vehicles in the network. It increased left turn phase durations at intersections 11 and 19. This resulted in an increase of 62 vehicles in the number of completed trips, significantly shorter delay, faster speed, and shorter travel time. It is noted that this higher percentage of left turns forced the algorithm to prohibit the left turns at two intersections in order to achieve an efficient network performance.

For 17.5% left turn, when the left turns were not managed, protected left turn phase was used at intersections 3, 4, 7, 8, 11, 12, 13, 14, 15, 16, 17, 18, 19, and 20. OLTMP prohibited the left turn at intersection 15 and removed left turn phases from this intersection, and rerouted vehicles in the network. This increased number of completed trips by 21 vehicles, reduced total delay and total travel time, and increased average speed (all statistically significant). Note that the improvement in the number of completed trips is smaller than the previous cases. The reason is that the number of left turners has started to become big enough to require left turn phase at most of the intersections as OLTMP only prohibited it at one intersection.
When the left turn percentage was increased to 20, OLTMP did not prohibit left turns at any intersection since rerouting them resulted in problems and congestion at other intersections. Its solution was identical to when the left turns were not managed.

![Graph showing number of intersections with prohibited left turns vs. left turn percentage](image)

**Figure 7.4. Number of intersections with prohibited left turns vs. left turn percentage**

Number of intersections with prohibited left turns is plotted in Figure 7.4 for different left turn percentages. It is noted that prohibiting the left turns was possible at 14 out of 20 intersections of the case study network. As shown in the Figure, for low left turn percentages (up to 7.5%) the left turns were not prohibited at any intersection as there was enough opportunity for the left turners to complete their maneuver during the permitted phases. When the left turn percentage increased to 10% and 12.5%, not all the left turners could complete their maneuver during the permitted left turn phases. As such, at one intersection the left turns were prohibited and vehicles were rerouted to other intersections to make their left turn maneuvers in a protected phase. For 15% left turn, at two intersections the left turns were prohibited and vehicles were rerouted. By increasing the left turn percentage from 15% to 17.5%, the benefits of left turn management started to decrease since left turn volume was high enough that required left turn phases at all intersections of the network. As such, the left turns were prohibited only at one intersection. Finally for 20% left turn, at none of the intersections
the left turns were prohibited. This was due to the large left turn volume that required a protected phase at all intersections of the network. Rerouting the left turns could significantly overcrowd other intersections and deteriorate overall network performance.

7.6 Summary

In this chapter, a program for optimal left turn management in oversaturated urban transportation networks was developed. Its formulation and solution technique was described and the algorithm was tested on a case study network under several traffic conditions. Numerical findings indicated that OLTMP has great potential to improve network performance efficiency by optimizing the policies on the left turns.

When left turn volume was low (up to 7.5% of the capacity of a lane), none of the left turns were prohibited. This was expected since the left-turners had enough opportunity to make their turning maneuver in a timely manner with no need to have a protected phase.

When left turn volume was very high (20% of the capacity of a lane), none of the left turns were prohibited as well. This was also expected since doing so resulted in rerouting too many vehicles and overcrowding the other intersections.

However, for moderate left turn volumes (10% to 17.5% of the capacity of a lane) left turns were prohibited in some intersections of the network. This happened since rerouting this many vehicles and adding them to the rest of intersections resulted in more efficient network performance at the intersections that the left turns were prohibited in them such that overall network performance was improved.
CHAPTER 8

DYNAMIC INTELLIGENT SIGNAL COORDINATION IN OVERSATURATED TRANSPORTATION NETWORKS

8.1 Introduction

Traffic signal coordination, when done properly, improves intersection traffic operation and safety. In traffic signal coordination, it is tried to synchronize green durations at two or more closely located traffic lights such that vehicles passing through them experience a minimum number of stops. This means that for two coordinated intersections, vehicles released from the upstream intersection (during the green signal of a coordinated phase) do not stop for a red light at the downstream intersection in most of the times. It is extensively believed that signal coordination is only possible when the cycle lengths of the coordinated signals are identical, or one is \( k \) times as long as the other \((k \) is an integer number). This strategy has led to very desirable results.

![Diagram of traffic signal coordination](image)

Figure 8.1. The effects of common and variable cycles on the offsets

It is very well established that a common cycle length is needed to coordinate the signals over all cycles. This is required since if the cycle lengths change at different intersections, the beginnings of the coordinated phases start to roll over time relative to each other; consequently, the offsets change over time, and signals become un-coordinated. This phenomenon is shown in Figure 8.1.
However, it is still unknown if using a common cycle length results in optimal network performance as opposed to using variable or approximately equal cycle lengths at intersections. Hajbabaie and Benekohal (2011) showed on a case study network that using variable cycle lengths has potential to improve network performance [70]. This was achieved by establishing signal coordination only when needed and not in all cycles. In fact, it is quite possible that using different cycle lengths along an arterial results in a more efficient network performance by reducing wasted greens and de-facto reds and coordinating the signals only when needed. To illustrate this, assume the arterial shown in Figure 8.2. As shown in the Figure, traffic volume at the minor street at intersection number 2 is half of that at intersection number 3. In this condition, cycle length needed to accommodate traffic demand at intersection no. 2 is shorter than that at intersection no. 3. Using a common cycle length along this corridor yields waste in green time (even when green splits are proportional to traffic volumes). In addition, it can result in sending too many vehicles to intersection no. 3. This happens since green time for through movement along the arterial at intersection no. 2 is longer than that at intersection no. 3. Therefore, more vehicles are released along the arterial at intersection no. 2 than what can be processed at intersection no. 3. This can potentially create long queues at eastbound approach at intersection no.3, and in extreme cases, it may create spillovers.

![Figure 8.2](image)

**Figure 8.2. Effects of using a common cycle in an arterial on queue length**

This example shows that using a common cycle length along an arterial (for signal coordination purpose) does not always result in the most efficient network performance possible. In fact, using different cycle lengths at different intersections may result in a more efficient network performance. Therefore, in an urban transportation network, before choosing a common cycle length for signal
coordination, it is needed to find out if using a common cycle length strategy yields the most efficient network performance. The answer to this question leads to choosing either a common cycle or different cycle lengths at different intersections of the network. It should be noted that the best strategy is both network- and traffic pattern-dependent. This means that one strategy may result in the most efficient network performance under certain conditions while it might not be the best strategy to use under different conditions. Therefore, the optimal strategy needs to be determined for different networks and traffic conditions. IDSTOP can be used to find this optimal strategy. In fact, IDSTOP can be used to find optimal signal timing for a certain network and traffic condition. For identical conditions, IDSTOP with added constraints to ensure identical cycle length at all intersections of the network, can be used to find another set of optimal solutions when using a common cycle is enforced. The solutions can be compared to each other in terms of several network PM to identify the best strategy. The rest of this chapter explains the methodological framework of choosing the best strategy among using a common cycle length or variable cycle lengths.

8.2 Methodology

As mentioned before, two strategies for choosing cycle length a transportation network are compared to each other to find out which one results in the most efficient network performance. These two strategies are:

1- Using a common cycle length in the network
2- Using different cycle lengths in the network.

Network performance efficiency is determined using the following PM:

1- Number of completed trips
2- Total travel time
3- Total delay
4- Total stopped delay
5- Number of phase failures
6- Average speed
7- Percentage of stopped vehicles, and
8- Storage percentage

PM 4 to 8 are used since they indicate how efficiently signals are coordinated. Shorter stopped delay, higher average speed, and lower percentage of stopped vehicles indicate more efficient signal coordination.

To make a valid comparison, for both strategies optimal signal timing parameters and turning percentages are needed. It is necessary to make sure that under each strategy, the most efficient network performance is used. This can be achieved by using IDSTOP to optimize signal timing parameters and turning percentages for the network for both strategies. For different cycle lengths strategy, IDSTOP can be used without any modification; however, IDSTOP does not necessarily find a common cycle in a network. Therefore, it has to be modified to ensure finding a common cycle for common cycle strategy. For this purpose the following constraints need to be added to IDSTOP and hereafter this modified version of IDSTOP is called Common-Cycle-IDSTOP (CC-IDSTOP):

\[ C_i^t = C_j^t \quad \forall i, j \in I, \forall t \in T_i \]  

(8.1)

\[ I = \text{set of all intersections of the network} \]

\[ T_i = \text{set of discrete time intervals (in the order of minutes)} \]

Constraints 8.1 can be enforced in two ways:

a) Discarding any solution that does not satisfy the constraints

b) Creating the solutions such that they cannot violate the constraints
The first way of enforcing the constraints is very easy to implement; however, it is not efficient. It is very likely that for obtaining a feasible solution, numerous candidates need to be created. This requires a significant amount of runtime. The second method is computationally very efficient and needs some changes in the structure of the chromosome. In that case, decision variable on cycle length of each intersection is removed. Instead, a single decision variable on the common cycle length is used for the entire network. The decision variables for each intersection are green splits and the offsets. Using this method, CC-IDSTOP cannot generate any solution without a common cycle. This method is used to enforce constraints 8.1.

To select the best strategy both IDSTOP and CC-IDSTOP are used to find optimal solution for the network and traffic conditions of interest. After obtaining the solutions, they are replicated in CORSIM to account for its internal variability and consider different driver behaviors and vehicle arrival to the network. The number of runs can be determined using the following equation based on the accuracy needed in estimating the value of each PM:

\[
\# \text{ of Replications} = \frac{S^2 \cdot z^2}{\varepsilon^2}
\]  

(8.2)

Where:

\( S \): standard deviation of each PM,

\( z \): critical value of normal distribution (or student t distribution if \( N \) is less than 30) for a certain confidence level, and

\( \varepsilon \): the acceptable error in mean estimation.
It is noted that the maximum required number of replications was 100; however, to be consistent with the other chapters and increase the accuracy of estimating different PM, a total of 250 runs were made. The process of determining the optimal cycle length strategy is presented in figure 8.3.

8.3 Case Study

In undersaturated condition, using a common cycle length has shown desired network performance. In addition, the two issues discussed in the illustrative example are very unlikely to happen in undersaturated conditions. As a result, using variable cycle length strategy may at most have a very negligible benefit. Therefore, in the case study network, undersaturated demand is not used.
We test the approach on symmetric oversaturated condition that was introduced in chapter 5. This case was suitable for signal coordination because a) the intersections were closely located; b) traffic demand was symmetric that increases the opportunity of coordinating the signals along both directions of an arterial; and c) majority of traffic goes straight instead of making a turn. Since it is possible to coordinate the signals along all corridors, a common cycle for all intersections is used. The case study network is shown in Figure 8.4.
8.4 Numerical Results

CC-IDSTOP found a common cycle of 120 seconds with an average of 56 seconds of green for each direction. The offsets were optimized such that good signal coordination was observed in the network. The quality of signal coordination was checked by: a) Looking at the animated output of CORSIM for 10 different randomly picked seeds; and b) Looking at stopped delay, average speed, stopped vehicle percentage, storage percentage, and number of phase failures in the network. CC-IDSTOP coordinated the intersections along both east-west and north-south direction corridors such that vehicle travelled through these arterials with only a single stop.

On the other hand, IDSTOP found a solution whose cycle lengths ranged from 116 seconds to 124 seconds with an average of 119 seconds. It is noted that this average cycle length is very close to the common cycle length found by CC-IDSTOP. Splits, on average, were allocated proportional to entry volumes. Since the cycle lengths were not equal at different intersections of some corridors, the offsets along them varied over time.

Table 8.1 presents several PM that are obtained by making 250 microscopic traffic simulation runs (in CORSIM) using IDSTOP and CC-IDSTOP solutions. For both strategies, the optimized solution resulted in no spillovers, no de-facto reds, and no gridlocks in the network. IDSTOP solution processed 5643 vehicles with a total travel time of 301.80 hours in the network. CC-IDSTOP processed 5585 vehicles with a travel time of 318.15 hours that were both statistically different from number of completed trips and travel time for IDSTOP solution. This shows that by not enforcing equal cycle length at all intersections of the network not only the number of completed tips increased, total travel time was significantly reduced as well. These two together indicated that using variable cycle lengths resulted in a more efficient network performance (in terms of number of completed trips and travel time) than using a common cycle in our case study network.
Table 8.1. Performance Measures for Variable Cycle Length (IDSTOP) and Common Cycle (CC-IDSTOP) Strategies

<table>
<thead>
<tr>
<th>Performance Measure</th>
<th>Algorithm</th>
<th>Min</th>
<th>Average</th>
<th>Max</th>
<th>P-value</th>
<th>Percentage of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Trips (veh)</td>
<td>CC-IDSTOP</td>
<td>5367</td>
<td>5585</td>
<td>5904</td>
<td>&lt;0.0001</td>
<td>1.0%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>5261</td>
<td>5643</td>
<td>5797</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Travel Time (hr)</td>
<td>CC-IDSTOP</td>
<td>297.15</td>
<td>318.15</td>
<td>346.27</td>
<td>&lt;0.0001</td>
<td>-5.1%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>283.56</td>
<td>301.80</td>
<td>332.37</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Delay (hr)</td>
<td>CC-IDSTOP</td>
<td>198.06</td>
<td>218.52</td>
<td>246.89</td>
<td>&lt;0.0001</td>
<td>-7.5%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>182.78</td>
<td>202.05</td>
<td>235.72</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total Stopped Delay (min)</td>
<td>CC-IDSTOP</td>
<td>9632.4</td>
<td>10498.2</td>
<td>11644.2</td>
<td>&lt;0.0001</td>
<td>-10.2%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>8580.8</td>
<td>9422.5</td>
<td>10991.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average Speed (mph)</td>
<td>CC-IDSTOP</td>
<td>8.54</td>
<td>9.40</td>
<td>10.00</td>
<td>&lt;0.0001</td>
<td>5.5%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>8.72</td>
<td>9.92</td>
<td>10.66</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stopped Vehicles Percentage</td>
<td>CC-IDSTOP</td>
<td>33.48</td>
<td>36.03</td>
<td>38.90</td>
<td>0.13921</td>
<td>0.3%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>33.59</td>
<td>36.13</td>
<td>40.69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Storage Percentage</td>
<td>CC-IDSTOP</td>
<td>29.26</td>
<td>31.34</td>
<td>33.81</td>
<td>&lt;0.0001</td>
<td>-5.6%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>27.84</td>
<td>29.60</td>
<td>32.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Phase Failures</td>
<td>CC-IDSTOP</td>
<td>39.00</td>
<td>57.34</td>
<td>79.00</td>
<td>&lt;0.0001</td>
<td>-13.1%</td>
</tr>
<tr>
<td></td>
<td>IDSTOP</td>
<td>25.00</td>
<td>49.81</td>
<td>95.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Similar to travel time, total delay for variable cycle lengths strategy in the case study network was significantly lower than that for common cycle strategy (202.05 hour vs. 218.52 hour, respectively). This indicated that using variable cycles significantly improved network performance in terms of total delay. The same trends were observed for total stopped delay in the case study network.

Using variable cycles resulted in statistically larger average speed, smaller storage percentage, and fewer phase failures; however, stopped vehicles percentages were not statistically different. These indicated that using variable cycle lengths can result in more efficient signal coordination in the case study network.

In summary, variable cycle strategy statistically significantly improved network efficiency in terms of number of completed trips, travel time, delay, stopped delay, average speed, storage percentage and number of phase failures. Therefore, one can conclude that variable cycle lengths improved network performance efficiency compared to common cycle length strategy in our case study.
network. This indicates great potential for using variable cycles to further improve network performance.

8.5 Discussion

When IDSTOP was not forced to use a common cycle, the range of cycles was very narrow. This range was only eight seconds (116 - 124 seconds). Different cycle lengths strategy found more suitable signal timing parameters for each intersection, and the narrow range (of cycle lengths) provided enough opportunity for signal coordination when needed. In addition, along the east-west direction in one-way corridors with three lanes, the cycle lengths were at most six seconds different. Along the north-south direction in one-way corridors with three lanes, the cycles were similar but, changing from one corridor to another. In addition, the offsets were optimized along these corridors to coordinate the signals. Along the two-way two-lane corridors, the cycles were up to eight seconds different.

In addition to analyzing the PM, we looked at the animated output of CORSIM for ten randomly select seeds. Along north-south one-way corridors with three lanes, signals were coordinated. The offsets were optimized such that vehicles released from an upstream intersection reach a downstream intersection right after the back of queue at the downstream intersection traveled with a speed approximately close to the speed of arriving vehicles. This means that the signals turned green at downstream intersection well before vehicles from upstream intersection arrive to downstream intersection. This was observed for all ten seeds. Signal coordination on the east-west one-way corridors with three lanes was also observed.

8.6 Summary

In this chapter we proposed a method to determine the policy that results in a more efficient network performance among using variable cycle length and using common cycle strategies. The
methodology uses IDSTOP to optimize signal timing parameters for both policies and compares them by looking at different PM. Our findings in a case study network that was suitable for signal coordination indicated that the variable cycle length strategy has great potential to improve network performance compared to common cycle strategy. The improvement is achieved by using a more suitable signal timing for each intersection and only coordinating them when needed. In the case study, using variable cycle lengths reduced total delay by 7.5%, and increased the number of completed trips by 1.0%. Therefore, using variable cycle length strategy significantly improved network performance efficiency in symmetric oversaturated conditions.

The outcome of our method is problem specific. Therefore, for different networks and different traffic demand patterns, the optimal strategy might be different. However, since we found variable cycle strategy beneficial in a case study network that was suitable for signal coordination, we can conclude that variable cycles have great potential to improve network performance efficiency in networks that are less suitable for signal coordination. The outcome of this method can be used as an input to adaptive signal timing methods to reduce their search space. For example for some recurrent traffic patterns one can determine which policy results in a more efficient network performance and stop searching not-promising parts of the feasible area in the adaptive methods.

8.7 References

CHAPTER 9
A PROGRAM FOR OPTIMAL TRAFFIC METERING

9.1 Introduction

In undersaturated conditions when demand is fixed, the number of completed trips in a certain time interval is at most equal to the number of vehicles that enter the network. This means that if efficient signal timing parameters and routes are used, no vehicle is stored in the network.

However, in oversaturated conditions when demand is fixed, the number of completed trips is considerably lower than the number of vehicles that enter the network. If efficient signal timing parameters and routes are used, the number of completed trips is at most equal to the capacity of the network. In these conditions, queues start to grow and yield longer delays, slower speeds, and possibilities of upstream intersection blockages, de-facto reds, and even gridlocks.

Traffic metering in oversaturated conditions improves traffic condition inside the network at the expense of deteriorating it at the borders of the network. This is achieved by sending fewer vehicles inside the network and keeping them at the borders. This reduces delay and travel time inside the network while increases delay and travel time at the borders since more vehicles are kept there.

Deciding how much of the traffic to meter is a very important decision and needs to be made very accurately. Too much metering results in very short travel times inside and very long travel times outside of the network such that total travel time in the entire system (inside plus outside) is long. On the other hand, not enough metering yields very long travel times inside and very short travel times at the borders. This also may result in long total travel time in the entire system. This suggests that a certain level of metering may exist that results in lowest total travel time in the entire system by reasonably low travel times both inside the network and at its borders.
The main objective of this chapter is to formulate Optimal Network Metering Problem (ONMP) and develop a solution technique for it. However, optimizing metering level for a network independent of its traffic signal timing parameters and traffic assignment is not very likely to yield most efficient network performance. The reason is that traffic signal timing uses links’ volumes as input and yields changes in travel times on different links and routes of the network. Traffic assignment and Traffic metering alter the links’ volumes based on their travel times. The obvious interdependency between these three means that a change in one may result in significant changes in the rest. As such, the proposed formulation should account for this interdependency to ensure efficient network performance. In the rest of this chapter optimal network metering problem formulation and solution technique is presented, its performance is tested on a case study network, and concluding remarks are presented.

9.2 Optimal Network Metering Problem Formulation

Optimal Network Metering Problem (ONMP) is formulated as a minimization problem. The objective is to minimize total travel time both inside the network and at its borders. The objective function is formulated as follows:

\[
\text{Min } \psi^t(\alpha_i, \varphi_i, C_i, s_{ik}, sf g_{ij}, y^{c}_{rs,ij}), \quad \forall t \in T_i, \forall i, j \in I/(R, S), \forall k \in K_i, \forall r \in R
\]  

Where:

\(T_i\) = set of discrete time intervals (in the order of minutes)

\(I\) = set of all intersections of the network

\(K_i\) = set of all phases available at intersection \(i\)

\(R\) = set of source nodes

\(\psi^t\) = total travel time in the network at time interval \(t\)
\( a_{rt}^t \) = number of vehicle that are let into the network at entry link \( r \), at time interval \( t \)

\( \emptyset_i^t \) = number of phases at intersection \( i \) at time interval \( t \)

\( C_i^t \) = cycle length of intersection \( i \) at time interval \( t \)

\( s_{i,k}^t \) = split for green for phase \( k \) of intersection \( i \) at time interval \( t \)

\( sof \ g_i^t \) = start time of the first phase of intersection \( i \) at time interval \( t \)

\( y_{rs,i}^t \) = turning traffic volume at upstream intersection \( i \) moving towards downstream

As will be explained later in the chapter, the entry links are extended to accurately determine travel time on them. The decision variable of the problem is the metering level at each entry link of the network at each time interval and is denoted by \( a_{rt}^t \). The value of the decision variable denoted the number of vehicles per hour that are let into the network from each metering point. As such, \( a_{rt}^t \) can take any values between 0 and the traffic demand at the entry points. Therefore, the first set of constraints is as follows:

\[
0 \leq a_{rt}^t \leq \sum_{s \in S} d_{rs}^t, \quad \forall r \in R, \forall t \in T_i
\]  \hspace{1cm} (9.2)

Where:

\( S \) = set of sink nodes

It is noted that the value of the objective function for a certain decision variable vector depends on the signal timing parameters and traffic assignment used in the network. In other words, for a decision variable vector, numerous values for objective functions exist depending on what signal timings and turning percentages are used in the network. As expected, efficient signal timing and traffic assignment in the network results in lower travel time inside the network and even at its borders (since more vehicles can be processed by the network). On the other hand, non-efficient signal timing
parameters and traffic assignment increases the travel time inside the network and consequently at its borders by reducing its capacity. Since we are only interested in the most efficient network performance for each metering strategy, near optimal signal timing parameters and turning percentages need to be found for each decision on metering strategy. This involves solving IDSTOP for each decision on metering strategies. Therefore, the second set for constraints enforces to use signal timing parameters and traffic assignment that results in the most efficient performance possible inside the network. This also addresses the interdependency between traffic metering and signal timing and traffic assignment:

\[
(\phi^t_i, c^t_i, s^t_{i,k}, sof g^t_i, y^t_{rs,ij}) = \text{Argmax (IDSTOP)}, \quad \forall t \in T_i, \forall i, j \in I/(R,S), \forall k \in K_i \quad (9.3)
\]

In summary, the formulation of the problem is as follows:

\[
\begin{align*}
\text{Min } & \psi^t(\alpha^t_r, \phi^t_i, c^t_i, s^t_{i,k}, sof g^t_i, y^t_{rs,ij}), & \forall t \in T_i, \forall i, j \in I/(R,S), \forall k \in K_i, \forall r \in R & \quad (9.1) \\
\text{s.t.} & \\
0 \leq \alpha^t_r \leq \sum_{s \in S} d^t_{rs}, & \forall r \in R, \forall t \in T_i & \quad (9.2) \\
(\phi^t_i, c^t_i, s^t_{i,k}, sof g^t_i, y^t_{rs,ij}) = \text{Argmax (IDSTOP)}, & \forall t \in T_i, \forall i, j \in I/(R,S), \forall k \in K_i & \quad (9.3)
\end{align*}
\]

### 9.3 Solving ONMP

Similar to IDSTOP, there is no closed-form formulation to represent the objective function of ONMP in terms of its decision variables. Therefore, optimization techniques that rely on knowing such relationships and the structure of the objective function cannot be used. In addition, the solution space is large. At each entry link, one decision variable exists. Assuming \( x \) different values for each decision variable and a total of \( n \) entry links, the decision space of ONMP is as large as \( x^n \). Based on the value of \( x \), and the number of entry links, the decision space can be extremely large. This solution space is
large enough to make traditional search methods such as exhaustive search and dynamic programming unsuccessful. Therefore, a meta-heuristic approach is developed to solve the problem.

Searching through all possible values for the decision variable (number of vehicles entering the system from entry links) has two problems:

1- Not all the values for the decision variables $x$ are meaningful. For example, it is never accepted to meter all traffic at the entry links. In addition sending many vehicles more than the capacity of the intersections does not make sense either. Therefore, we limited the values of the decision variables to vary between 625 vphpl and 1000 vphpl. Note that the capacity of entry links as established in chapter 5 is around 876 vphpl.

2- These many different values make the decision space significantly large. This may result in extremely long run time. In addition, it is very unlikely that a difference of one, five, or even ten vehicles in the entry volume results in a significant difference in the network performance. Therefore, we decided to use increments of 25 vehicles for ONMP decision variables.

Therefore the decision variable of ONMP at each entry link can take the following values:

$$\omega_r^t = \{625, 650, 675, 700, 725, 750, 775, 800, 825, 850, 875, 900, 925, 950, 975, 1000\}, \quad \forall r \in R, \forall t \in T_r$$

It is noted that the assumptions on the value of the decision variable do not limit the generality of the approach; however, it may change the final solution of the algorithm. Micro-Elitist GA is used to solve ONMP. Binary coding was used where each decision variable was coded using four bits; therefore, total chromosome length was $4n$ where $n$ was the number of entry links. It should be noted that it was assumed that the same metering level was used on different lanes of each entry link.
In the developed meta-heuristic approach, the first population can be generated either randomly or by means of some heuristics. To get the fitness of each individual, the decisions on metering level at each entry link are transmitted to IDSTOP. IDSTOP coded CORISM input file with different metering levels and optimized the signal timing parameters and the turning percentages based on new number of vehicles entered the network. These optimized decision variables were used to determine the fitness value of the decision variable. CORSIM was run and total travel time inside and at the borders of the network with optimized signal timing parameters and turning percentages were
obtained. This value was used in Micro-Elitist-GA as the fitness associated with the ONMP candidate solution. The framework of this process is shown in Figure 9.1.

Solving ONMP requires a lot of IDSTOP runs. As mentioned in previous chapters solving IDSTOP is by itself very time consuming. Therefore, the runtime of ONMP can be extremely long since in needs several IDSTOP runs. To reduce ONMP runtime, IDSTOP has to be solved much faster. As shown in Chapter 6, ES+ was the quickest algorithm in reaching to most levels of the upper-bound in solving IDSTOP. As such, it was used to solve IDSTOP. There is no need to run ES+ for 50 generations. In fact, ES+ found a solution that was at most less than 5% below our theoretical upper bound for the objective function in less than 6 generations for all four case studies. As a result, to solve IDSTOP, ES+ was used with a maximum number of generations equal to 10. In addition, as soon as the solution was less than 5% away from the theoretical upper bound, ES+ was stopped and IDSTOP solution was transmitted to ONMP.

To further speed up IDSTOP’s run time, instead of making 10 replications to obtain the fitness value of each candidate solution, only 5 replications were made. Making 5 replications resulted in less than 5% error in estimating the number of completed trips in the network for good and average solutions. These two measures together, reduced IDSTOP runtime by a factor of 10.

9.4 Case Study Network

ONMP was tested on modified downtown Springfield network with symmetric oversaturated demand pattern. This demand pattern was selected since traffic metering in undersaturated condition is not needed. In this case traffic demand at all entry links was 1000 vphpl that was high enough to make the network oversaturated. The capacity of entry links was on average 876 vphpl (obtained in chapter 5). The metering decision variable could take 16 values that were explained before.
Traffic was metered using gating signals that were placed 550 feet (average link length in the network) upstream of each entry intersection, see in Figure 9.2. The gating signals had only green and red durations similar to on-ramp metering signals. To monitor travel time and delay at the boundaries, the length of the entry links was increased to 2000 feet upstream of the gating signals. An entry link of 2000 feet was long enough to ensure that it never will be filled with vehicles at the borders. If the entry links are filled with vehicles, CORSIM underestimates travel time and delay since the vehicles have not entered the system. That is why each entry link has to be long enough to accommodate all incoming traffic.

![Gating Signals](image)

Figure 9.2. Case study network with gating signals

9.5 Numerical Results

ONMP was used to optimize metering level at different entry links of the network. In addition, for 17 different uniform metering strategies, signal timing parameters were optimized. For each uniform metering level, identical entry volumes were used at all entry point, see in Table 9.1. These 16 uniform metering strategies were used to a) study the effects of different metering levels on traffic conditions
both inside and at the borders of the network, and b) to compare ONMP solution to them to determine if ONMP is capable of finding a more efficient solution. For each uniform metering strategy and for ONMP solution the following network Performance Measures (PM) were determined:

1- delay inside the network,
2- travel time inside the network,
3- average speed inside the network,
4- delay at the borders,
5- travel time at the borders,
6- total delay in the entire system,
7- total travel time in the entire system, and
8- number of completed trips

To determine these PM, each solution was coded in CORSIM and 250 replications were made to account for internal variability of CORSIM and test the solution for a vast set of different driver behaviors and vehicles arrival to the network (a maximum of 100 replications was enough for all cases). Different metering levels and different network PM are shown in Table 9.1. To statistically compare the PM over different metering strategies Least Significant Difference (LSD) test with $\alpha = 0.05$ was used. For Each PM, having similar letters for two different metering levels indicates no statistical difference between the metering levels while having different letters indicates that the value of PM is statistically different for the two metering levels.

ONMP solution let an average of 913 vphpl enter the network. This was achieved by setting the entry volume equal to 950 vphpl on north-south and equal to 850 vphpl east-west directions. Note that north-south direction had a total of 17 entry lanes ($3 + 3 + 3 + 4 + 4$ from west to east) while east-west direction had only 10 entry lanes ($3 + 3 + 2 + 2$ from north to south). Also note that the average
entry volume was more than the capacity of the links (876 vphpl). This higher volume ensured a higher
green utilization in the network since several vehicles were always present at different intersections
before the signals turned green. It should be noted that the sum of volumes for critical movements was
950 + 850 = 1800 vphpl that was only slightly higher than the capacity of the intersections (1752
vphpl). The optimized metering strategy yielded a travel time of 296.8 hours inside and 282.3 hours at
the borders of the network that were both statistically lower than travel times and delays under all
uniform metering strategies, see in Table 9.1.

Table 9.1 PM for Different Uniform Metering Levels and Optimized Metering Strategy

<table>
<thead>
<tr>
<th>Metering Strategy</th>
<th>Inside the Network</th>
<th>At the Borders</th>
<th>Entire System</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay (hr)</td>
<td>Speed (mph)</td>
<td>Travel Time (hr)</td>
</tr>
<tr>
<td>ONMP Solution</td>
<td>196.0 A</td>
<td>10.2 A</td>
<td>296.8 A</td>
</tr>
<tr>
<td>625 vphpl Enter</td>
<td>84.3 B</td>
<td>14.1 B</td>
<td>158.6 B</td>
</tr>
<tr>
<td>650 vphpl Enter</td>
<td>90.7 C</td>
<td>13.8 C</td>
<td>168.2 C</td>
</tr>
<tr>
<td>675 vphpl Enter</td>
<td>95.7 D</td>
<td>13.7 D</td>
<td>175.8 D</td>
</tr>
<tr>
<td>700 vphpl Enter</td>
<td>103.4 E</td>
<td>13.3 E</td>
<td>186.2 E</td>
</tr>
<tr>
<td>725 vphpl Enter</td>
<td>109.4 F</td>
<td>13.2 F</td>
<td>195.3 F</td>
</tr>
<tr>
<td>750 vphpl Enter</td>
<td>118.6 G</td>
<td>12.8 G</td>
<td>207.2 G</td>
</tr>
<tr>
<td>775 vphpl Enter</td>
<td>133.2 H</td>
<td>12.2 H</td>
<td>224.4 H</td>
</tr>
<tr>
<td>800 vphpl Enter</td>
<td>147.7 I</td>
<td>11.6 I</td>
<td>240.8 I</td>
</tr>
<tr>
<td>825 vphpl Enter</td>
<td>155.5 J</td>
<td>11.4 J</td>
<td>250.7 J</td>
</tr>
<tr>
<td>850 vphpl Enter</td>
<td>166.0 K</td>
<td>11.1 K</td>
<td>263.5 K</td>
</tr>
<tr>
<td>875 vphpl Enter</td>
<td>179.7 L</td>
<td>10.7 L</td>
<td>278.8 L</td>
</tr>
<tr>
<td>900 vphpl Enter</td>
<td>195.2 A</td>
<td>10.2 A</td>
<td>295.6 A</td>
</tr>
<tr>
<td>913 vphpl Enter</td>
<td>201.7 M</td>
<td>10.0 M</td>
<td>302.4 M</td>
</tr>
<tr>
<td>925 vphpl Enter</td>
<td>204.8 N</td>
<td>9.9 N</td>
<td>305.7 N</td>
</tr>
<tr>
<td>950 vphpl Enter</td>
<td>211.8 O</td>
<td>9.7 O</td>
<td>312.9 O</td>
</tr>
<tr>
<td>975 vphpl Enter</td>
<td>215.6 P</td>
<td>9.6 P</td>
<td>316.8 P</td>
</tr>
<tr>
<td>1000 vphpl Enter</td>
<td>218.5 Q</td>
<td>9.5 Q</td>
<td>319.7 Q</td>
</tr>
</tbody>
</table>

ONMP solution resulted in the lowest travel time at the borders. This was achieved by letting
more vehicles enter the network from the north-south direction with higher number of entry lanes (17
out of 27). Therefore, on 17 lanes of the network ONMP solution yielded considerably lower travel times
than what other 17 uniform metering strategies did. Travel time inside the network was only slightly
higher than that for when 900 vphpl were sent into the network. These two together resulted in shortest total travel time in the entire system for ONMP solution. Similar trends were observed for delay inside the network, at its borders, and in the entire system. At the same time, ONMP solution resulted in the highest number of completed trips in the network. Therefore, we concluded that ONMP solution resulted in the most efficient network performance among all the strategies that were tested. This indicates great potential for ONMP to further improve network efficiency in congested urban areas.

Looking at the PM for different uniform metering strategies (i.e. different entry volumes) reveals interesting findings. As the entry volume increased, delay inside the network, travel time inside the network, and the number of completed trips increased while average speed was decreased. This was expected since more vehicles entered the network. Number of completed trips continued its increasing trend up to an entry volume of 900 vphpl for which 5645 trips were completed. Increasing the entry volume beyond this level did not statistically significantly increase the number of completed trips; however, delay and travel time inside the network kept increasing, see in Table 9.1. Thus, network performance was deteriorated. This indicated that more vehicles were stored in the network (especially in the entry links) which worsened network performance. These trends are also shown in Figure 9.3.

Delay and travel time at the borders were decreased as the entry volume increased. This trend was observed up to an entry volume of 900 vphpl. Similar to the number of completed trips, increasing the entry volume beyond this level did not statistically significantly reduce delay and travel time at the borders. The reason was that not much more than 900 vphpl could enter the network as there was not much capacity left for incoming vehicles. In fact, although the metering signals allowed more vehicles enter the network, the first set of interior signals of the network did not have enough capacity to let much more than 900 vphpl in the network. As such, these vehicles were mostly store in the 550 ft long links of the network. These links connected the metering signals to the first signals of the network.
Among different uniform metering strategies, sending 900 vphpl vehicles into the network resulted in the most efficient network performance as it resulted in lowest total delay and total travel time in the entire system (statistically significant) while the number of completed trips was among the highest. This finding also supports ONMP solution that resulted in an average entry volume of 913 vphpl.

Since ONMP found an average entry volume of 913 vphpl (that was not uniform among different entry links), a uniform entry volume of 913 was used as well. Letting 913 vphpl to uniformly enter the network from all entry links yielded less efficient network performance compared to ONMP solution, see in Table 9.1. The reason was that this uniform entry volume resulted in a sum of volumes for critical movements equal to 1826 vphpl that was larger than the capacity of 1752 vphpl (and sum of critical volumes for ONMP solution, 1800 vphpl). This volume resulted in less efficient interior network performance compared to ONMP solution and uniformly sending 900 vphpl inside, while at the borders, its efficiency was similar to them. As such, overall network performance was less efficient compared to the other two strategies.
9.6 Summary

In this chapter Optimal Network Metering Program was introduced, formulated, and a meta-heuristic algorithm was developed to solve it. ONMP provides a framework for optimizing transportation network performance that accounts for interdependencies that exists between traffic signal timing, traffic assignment, and traffic metering.

The numerical findings on a case study network indicated that the proposed method had great potential to improve network performance efficiency by optimizing the number of vehicles that were allowed to enter the network and accordingly optimizing traffic signal timing parameters and turning percentages for them. In addition, they indicated that optimal metering had great potential to significantly improve network performance efficiency in terms of travel time and delay in the entire system. It also indicated that optimal metering did not reduce the number of completed trips but reduced delay and travel time in the system.

The numerical results indicated that optimized metering strategy reduced total delay by 10.6% and total travel time by 6.7% compared to no metering strategy. Therefore, optimal metering has significantly improved network performance in the case study. In addition, optimized metering strategy reduced total delay by 4.5% and total travel time by 2.7% compared to the best uniform metering strategy. This indicated that ONMP solution significantly improved network performance compared to the best uniform metering strategy.

We tested ONMP in a single case-study network for a specific traffic demand pattern and found very promising results. To fully discover potential benefits of traffic metering in urban transportation networks, we suggest solving ONMP for different case study networks and for a diverse set of traffic conditions.
Our numerical findings indicated that when links with enough capacity to hold vehicles are available, this capacity could be used to meter traffic travelling to downstream intersections to potentially improve network performance efficiency. Therefore, we suggest developing a method to study the effects of internal traffic metering in urban transportation networks to fully discover the benefits of traffic metering.
CHAPTER 10

CONCLUSIONS AND RECOMMENDATIONS

10.1 Conclusions

This research presented the development of a signal timing optimization model for oversaturated urban traffic networks with stochastic driver behavior and stochastic vehicular arrival headways. Intelligent Dynamic Signal Timing Optimization Program (IDSTOP) was formulated as a dynamic optimization problem whose objective was to maximize the number of weighted completed trips in the network (weighted by the length of the shortest route available for that trip). This objective function was selected among several objective functions (i.e., delay minimization, travel time minimization, throughput-minus-queue maximization, trip maximization, and weighted trip maximization) by using a simulation-based approach and resulted in the most efficient network performance among all. This was achieved by sending as many vehicles as possible out of the network with prioritizing those trips that required longer minimum travel distances in the network. The model aimed at managing transportation supply by optimizing signal timing parameters (i.e., phase plans, cycle lengths, green splits, and offsets), and simultaneously managing transportation demand by redirecting vehicles to less congested routes using system optimal traffic assignment. In addition to introducing an innovative objective function, several constraints were introduced to reduce the size of the search space and direct the search toward more promising parts of the feasibility area.

Solving IDSTOP is a very challenging task due to the following two reasons. First, IDSTOP is a nonlinear optimization problem without a closed-form formulation for the objective function in terms of the decision variables. Therefore, methods that rely on knowing the structure of the objective function cannot solve it. Second, IDSTOP has an extremely large decision space that makes the search very complicated. As such, traditional search methods such as exhaustive search cannot be used to solve it as
well optimization techniques such as dynamic programming. Therefore, a meta-heuristic algorithm was developed to solve it.

Rather than optimizing signal timing parameters for different time intervals in combination to each other, they were optimized sequentially. There were two main reasons for this: first, sequential optimization significantly reduced the size of decision space and consequently the runtime; and second, it yielded a more efficient network performance by ensuring that the highest possible number of vehicles has exited the network in each time interval.

The developed meta-heuristic approach generated a population of candidate solutions and improved their quality (i.e. value of the objective function or the fitness value) over different generations. This improvement was expected by selecting fitter solutions to generate new ones. The selection was stochastic but biased towards fitter solutions. New solutions were obtained by exchanging information between fitter solutions and introducing some random search. To obtain sufficient feasible solutions for the first generation, a huge number of candidate solutions were needed (this was significantly reduced for the rest of generations). As such, a heuristic method was developed to reduce this number. The feasibility of each candidate solution was checked in two steps. In the first step, a macroscopic approach was used. If the macroscopic approach found a solution feasible, its feasibility was checked using a microscopic traffic simulation approach as well. There are two main reasons for this: first, it is possible that a solution satisfies all the constraints in macroscopic approach but violates some in microscopic simulation, and second, macroscopic approach is considerably faster than microscopic simulation and help identify infeasible solution very quickly.

Microscopic simulation models were required to accurately address oversaturated conditions as well as stochastic driver behavior and vehicular arrival headway. To account for these stochasticities in the optimization, the fitness of each feasible solution was determined by making several simulation
runs. This was necessary to create different combination of these stochastic events. The fittest individual of each population was chosen for traffic assignment. Assigning traffic for the fittest individual not only significantly reduced the runtime, but also insured not using inefficient signal timing parameters for traffic assignment.

IDSTOP solutions were compared to Direct-CORSIM solution using a realistic case study network and four demand patterns covering both undersaturated and oversaturated conditions for symmetric and asymmetric demands. Findings indicated that IDSTOP solutions resulted in significantly more efficient network performance than Direct-CORSIM solutions. IDSTOP solutions increased the number of completed trips by 2.0% to 19.6% and at the same time reduced average delay by 8.9% to 30.8% for different demand patterns in the case study network. These figures indicated significant improvement in the network performance.

Simple GA, Elitist simple GA, Micro-Elitist GA, self-adaptive ES, and Elitist self-adaptive ES (ES+) were used to solve IDSTOP. In general, ES+ outperformed the rest of algorithms in reaching most different levels of the upper-bounds. In addition, ES+ was very efficient in oversaturated conditions especially when demand was symmetric. Micro-Elitist GA was very quick in early improvements in the fitness value. However, in most of the cases it was outperformed by ES+ in reaching higher levels of fitness value except for asymmetric undersaturated condition.

Building upon IDSTOP, a program for optimal left turn management in oversaturated urban transportation networks was developed. Optimal Left Turn Management Program (OLTMP) improved network performance by prohibiting the left turns in certain locations of the network. If a left turn was prohibited, the left turners were rerouted in the network towards their destinations. To optimize left turn strategies (i.e. where to prohibit the left turns and where to allow them) optimized signal timing parameters and turning percentages were needed. Therefore, without IDSTOP it was not possible to
develop OLTMP. Numerical findings indicated that OLTMP had great potential to improve network performance efficiency by optimizing the policies on the left turns. When left turn volume was low (up to 7.5% of the capacity of a lane), none of the left turns were prohibited since left-turners had enough opportunity to make their turning maneuver in permitted phases. When left turn volume was very high (20% of the capacity of a lane), none of the left turns were prohibited as well because doing so resulted in rerouting too many vehicles and overcrowding other intersections. However, for moderate left turn volumes (10% to 17.5% of the capacity of a lane) left turns were prohibited in one or two intersections of the network.

A method was proposed to determine the policy that results in a more efficient network performance among variable cycles and common cycle policies. IDSTOP was needed to make sure that for both policies efficient signal timing parameters are used. Our findings in a case study network that was suitable for signal coordination indicated the variable cycle length strategy had great potential to improve network performance compared to common cycle strategy. The improvement is achieved by using a more suitable signal timing for each intersection and only coordinating them when needed. In the case study, using variable cycle lengths reduced total delay by 7.5%, and increased the number of completed tips by 1.0% compared to using a common cycle. Therefore, using variable cycle length strategy significantly improved network performance efficiency in the case study.

IDSTOP was used to develop Optimal Network Metering Program (ONMP). ONMP aimed to improve network performance by metering traffic at entry points of the network. To choose the optimal metering policy, optimized signal timing parameters and turning percentages were needed. As such, it was not possible to develop ONMP without having IDSTOP. ONMP was formulated and a meta-heuristic algorithm was developed to solve it. The numerical results indicated that optimized metering strategy reduced total delay by 10.6% and total travel time by 6.7% compared to no metering strategy. Therefore, optimal metering has significantly improved network performance in the case study.
addition, optimized metering strategy reduced total delay by 4.5% and total travel time by 2.7% compared to the best uniform metering strategy. This indicated that ONMP solution significantly improved network performance compared to the best uniform metering strategy.

10.2 Recommendations

The research presented in this study can be further extended into the following areas:

10.2.1 Signal Optimization Methods

IDSTOP took stochasticities associated with drivers’ behaviors (in acceleration and deceleration rates, joining back of queue, and lane changes) and arrival headway into account; however, it did not account for stochasticities associated with traffic incidents, traffic signal failures, vehicles failures, emergency vehicles passages, etc. Microscopic traffic simulation packages have started to model traffic incidents but not the rest of stochastic event mentioned above. Developing models to account for these stochastic events is recommended. Such models can improve the robustness of signal timing parameters in transportation networks.

10.2.2 Variations of Evolutionary Algorithms

In this research we looked at several variations of GA and ES for signal timing optimization purpose. Self-adaptive ES showed very promising performance by preventing premature convergence. Self-adaptive GA is worth exploring to help prevent premature convergence during the search process. In that case, GA parameters including population size, selection method, and crossover and mutation rates can vary in the course of optimization. In addition using hybrid techniques such as GA-ES, or combination of them with greedy algorithms has great potential to improve the efficiency of the optimization.
10.2.3 Prohibiting or Allowing Left Turns in Oversaturated Conditions

Optimal left turn management indicated great potential to further improve network performance efficiency. The method developed in this study was tested on a single case study network. Exploring the benefits of managing the left-turn policies for different networks and different demands for left turns is recommended.

10.2.4 Traffic Metering in Oversaturated Conditions

We tested optimal network metering program in a single case-study network for a specific traffic demand pattern and found very promising results. To fully discover potential benefits of traffic metering in urban transportation networks, we recommend testing different case study networks and diverse set of traffic conditions.

When links with enough capacity (to hold vehicles) are available, this capacity could be used to meter traffic travelling to downstream intersections to potentially improve network performance efficiency. Therefore, we recommend developing a method to study the effects of internal traffic metering in urban transportation networks to fully discover the benefits of traffic metering.

10.2.5 Distributed Signal Control System

This research focused on centralized signal control strategy; however, when large urban networks are studied, distributed signal control should be explored as it may be necessary to decompose large networks into smaller sub-networks.
REFERENCES


[54] B. Ran, R. W. Hall and D. E. Boyce, "A Link-Based Variational Inequality Model for Dynamic


