Adaptive Timing Optimization for Cyber-Physical Traffic Control Systems

Wen-Hao Wu

A Thesis Submitted to the
Institute of Computer Science and Information Engineering of
National Chung Cheng University for the
Degree of Master in
Computer Science and Information Engineering

June, 2015

摘要

隨著人口的增加，城市中的車輛也隨之增加，進而造成交通壅塞與空氣汙染。因此，智慧型運輸系統 (intelligent transportation systems, ITS) 被提出來改善交通壅塞的問題，其中之一為交通號誌控制系統 (traffic signal control systems)，透過調整交通號誌來達到改善交通壅塞。現今的交通號誌控制系統主要根據過去的歷史車流量資料來調整各個時段的交通號誌，但此方法仍無法動態且自適應地根據車流量來進行號誌的調整。

為了能動態且自適應地調整號誌，我們提出了一個整合計算單元與實體單元的系統稱為「網宇實體交通控制系統 (Cyber-Physical Traffic Control Systems)，簡稱 (CPTCS)」以取得即時的道路資料並優化號誌。此外，我們也針對 CPTCS 提出了一個名為「自適應號誌優化 (Adaptive Timing Optimization)，簡稱 (ATO)」的方法，當中包含「基於基因演算法之號誌優化 (GA-based signal timing optimization)」與「自適應優化調整 (adaptive adjustment of optimization)」兩個部分。基於基因演算法之號誌優化以即時的道路資料對號誌進行優化，而自適應優化調整則是在 CPTCS 有限的計算效能下透過調整優化的閥值與頻率來增加計算的效率。

在優化效果的實驗結果顯示出在一天的情況下，ATO 比起固定調整的方法在平均上能減少 34% 的等待車輛，且與基於賽局理論之號誌優化相比，能減少將近一倍的等待車輛。在優化次數的實驗結果顯示出在一天的情況下，使用自適應優化調整比未使用在平均上能減少 21% 的優化次數並達到近乎相同的優化效果。

關鍵字: *交通壅塞, 智慧型運輸系統, 網宇實體系統, 號誌優化, 基因演算法, 自適應優化,*

Abstract

With the increase in population, the number of vehicles on the road has increased rapidly, which causes traffic congestion and air pollution. To solve this issue, intelligent transportation systems (ITS) were proposed.

One kind of ITS is traffic signal control systems which adjust signal timing configurations at intersections then the problem of traffic congestion is alleviated . Existing traffic signal control systems adjust signal timing configurations according to historical traffic volumes. However, such an approach is still unable to react dynamically and adaptively to real-time traffic volume.

In order to alleviate traffic congestion dynamically and adaptively, we propose a *Cyber-Physical Traffic Control Systems* (CPTCS) that integrates the computational elements and physical entities to obtain real-time traffic data and optimize signal timing configurations. Further, we also propose an *Adaptive Timing Optimization* (ATO) for CPTCS including GA-based signal timing optimization and adaptive adjustment of optimization. The GA-based signal timing optimization tries to optimize signal timing configurations according to the real-time traffic data. To increase scalability of CPTCS, adaptive adjustment of optimization is proposed, which includes adjusting optimization threshold and optimization frequency.

Experiments conducted on optimization show that compared with the fixed timing method, ATO reduces number of waiting vehicles by 34% and incurs reduction in number of waiting vehicles almost double compared with the game theory-based method. Experiments conducted on optimization times show that compared with signal timing optimization only, signal timing optimization with adaptive adjustment of optimization can reduce the optimization times by 21% for a full day traffic.

Keywords: *Traffic Congestion*, *Intelligent Transportation Systems*, *Cyber-Physical Systems*, *Signal Timing Optimization*, *Genetic Algorithm*, *Adaptive Optimization*,

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the increase in population, the number of vehicles on the road has increased rapidly, which cause traffic congestion in urban areas, especially during the peak hours. This situation leads to drivers wasting a lot of time during travel, while causing air pollution. As a result, traffic congestion is a critical issues in urban traffic.

To solve this issue, existing traffic control systems adjust signal timing configurations according to the different hours of a day. However, such an approach is still unable to react dynamically and adaptively to the traffic volume because the configurations are determined by historical traffic volume rather than real-time.

In order to obtain the real-time traffic information of urban areas, a traffic control system must be integrated with vehicle detectors. Such a system when equipped with capability to react dynamically and adaptively to the traffic volume is called a *Cyber-Physical Traffic Control Systems*(CPTCS).

In this Thesis, we want to alleviate traffic congestion, so we proposed an *Adap-*

tive Timing Optimization (ATO) for CPTCS, including GA-based signal timing optimization and adaptive adjustment of optimization. In ATO, it considers the traffic volume and queue length of roads to calculate the most appropriate signal timing for intersection to alleviate traffic congestion. Furthermore, due to the limited computing power of CPTCS, ATO also adjusts the optimization threshold and optimization frequency to increase computation efficiency.

1.1 Background

1.1.1 Traffic Signal Timing Optimization

In order to make the signal timing more appropriate to intersection, the traffic signal timing optimization was proposed. The traffic signal timing optimization is to adjust the length of green time of each road according to the traffic information such as traffic volume, queue length, waiting time of vehicles and so on.

Traffic signal timing optimization has demonstrated it effective for in reducing traffic congestion [3] and a large number of traffic signal timing optimization methods have been proposed to address the traffic signal timing issue, including fuzzy-logic [1], neural networks [4], particle swarm optimization [5][6], Petri-nets [7], game theory [8], and genetic algorithm [9][10].

1.1.2 Cyber-Physical System

Cyber-Physical Systems (CPS) integrate computing, networking, data storage, and physical entities (e.g. sensors, device controller, etc.) for monitor or control the physical entities [11][12].

The difference between and traditional embedded system is the emphasis in CPS on the interaction between systems, instead of standalone. Nowadays, CPS has played an important role in smart-grid [13], medical devices [14], *Intelligent Transportation Systems* (ITS) [15] and so on.

1.2 Motivation

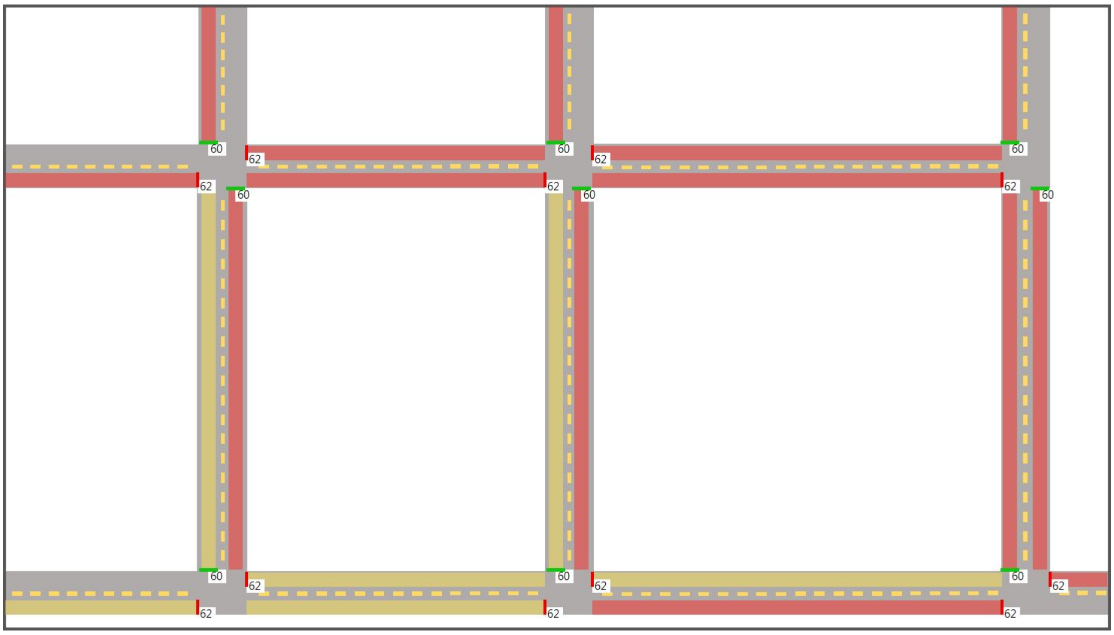

Figure 1.1: Traffic congestion status before optimization

In this section, we present an example to show our goal. We simulate an urban city with high traffic volume and optimize the traffic with CPTCS.

The degree of congestion as shown in Table 1.1, is estimated by the *Intersection Average Waiting Rate* (IAWR) which will be explained in chapter 3. According

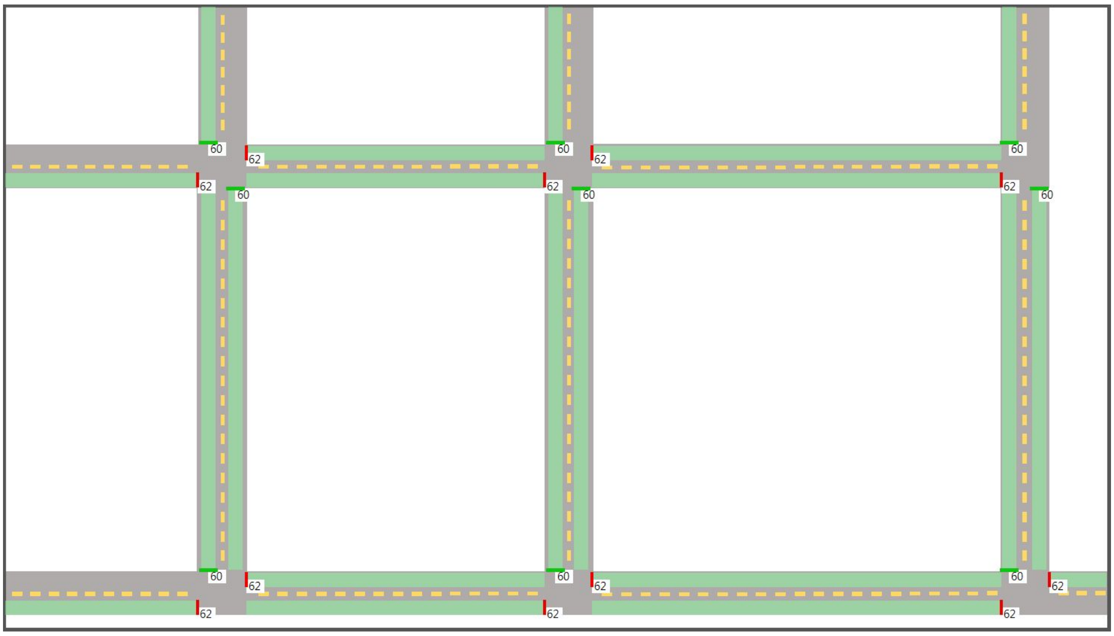

Figure 1.2: Traffic congestion status after optimization

to the IAWR, we mark the roads with colors to represent the congestion degree clearer, that is IAWR under 50 with white, between 50 and 70 with gray and above 70 with black.

Figure 1.1 shows the IAWR of each intersection, where all intersections are heavily congested. This situation implies that current signal timing configurations cannot reduce the traffic volumes at the congested intersections. To mitigate traffic congestion, the signal timing configurations should be adjusted via optimization that is, IAWR must be decreased. Figure 1.2 shows the changed traffic congestion degree after optimization. Thus, heavily congested intersections are now less congested. The average IAWR of all intersections is now reduced from 67.77 to 42.91 (by 36.34%).

Table 1.1: Optimization result of example

Intersection	Unoptimized IAWR	Optimized IAWR	Reduce (%)
1	68.39	46.65	31.78
2	71.62	50.53	29.45
3	60.60	50.03	17.45
4	68.55	39.66	42.15
5	69.51	33.53	51.76
6	67.96	37.07	45.45
Average	67.77	42.91	36.34

Table 1.2: Optimization times of example

Time	No adjustment		Adaptive adjustment of optimization	
	Average IAWR	Times	Average IAWR	Times
Peak hours (morning)	37.63	34.4	37.57	17.1
Off-peak hours	33.24	21.9	33.32	24.4
All day	34.44	75.3	34.59	59.9

In order to react to the traffic congestion rapidly and prevent the unnecessary optimization and over-optimization, ATO adjusts the optimization threshold and optimization frequency according to traffic conditions. Table 1.2 gives the number of times optimization was performed for all day. For all day, the resulting IAWR values in the “No adjustment” case and in the “Adaptive adjustment of optimization” case are approximately the same; however, in the latter case the numbers of optimization times are much smaller. The average number of optimization times is reduced from 75.3 (no adjustment) to 59.9 (with adaptive adjustment), that is, the reduction is by 21%. This means the adaptive adjustment of optimization can reduce the optimization times and maintain almost the same optimization result. In other words, the computation efficiency is increased by adaptive adjustment of optimization.

1.3 Thesis Organization

The rest of this Thesis is organized as follows. Chapter 2 introduces related work on traffic signal timing and the optimization algorithm. Chapter 3 illustrates the proposed system architecture, how it works, and the definitions used in the system. Chapter 4 explains the optimization algorithm in our system. Chapter 5 presents the simulations and analyzes the results. Chapter 6 gives the conclusion of the Thesis.

Chapter 2

Related Work

In this chapter, we introduce the issue in traffic signal timing and research previous approaches. Further, we organize the approaches to find out the crucial traffic informations that used in previous works and the result as Table 2.3.

2.1 Traffic Signal Control

Traffic signal control (TSC) is important and effective to urban city traffic and solving traffic congestion. Thus, there are several methods have been proposed for the TSC optimization. TSC can be classified to three stages [3]:

1. Pre-timed control [16][17] : Using the predetermined signal timing for corresponding time or traffic flow.

2. Traffic-responsive control [18] : Depending on the real-time traffic data which from sensors to adjust signal timing.

3. Intelligent control : By implementing the computational intelligence like

neural networks, fuzzy system, genetic algorithm, etc. on TSC to found the optimal or suboptimal signal timing.

Because the uncertainty and complexity of traffic, we focus on the intelligent control approaches and use it to solving signal timing.

2.2 Single Approach on Signal Control

Lertworawanich Ponlathep proposed a Adaptive Signal Control Algorithm for isolated intersections [19]. They use the shockwave theory-based method to construct the time-space diagrams from occupancy information that obtained from detectors. After constructing the time-space diagrams of all phase, the delay per cycle can be evaluated then optimizes the splits of each phase by increase or decrease green time to reduce total delay. Due to the increase or decrease of green time, the cycle length of the intersection is adjusted too.

Khanjary et al. proposed a game model that based on Cournot's oligopoly game to optimize single intersection [8]. In this model, each direction of intersection are regarded as a player and the queue length as the payoff. All players is compete for the shared time to increase their green light time. If the green time of the player is long enough, the payoff will achieve the highest when it equal to zero that means no vehicles in the queue. Therefore, the queue model was proposed and it determine the queue length by arrival rates α, departure rates β, current queue q, and green time t of the phase that can be expressed as $q_i + \alpha T - t_i(\alpha + \beta)$. By draw the diagram of the equation which shown in Figure 2.1, the Nash equilibrium

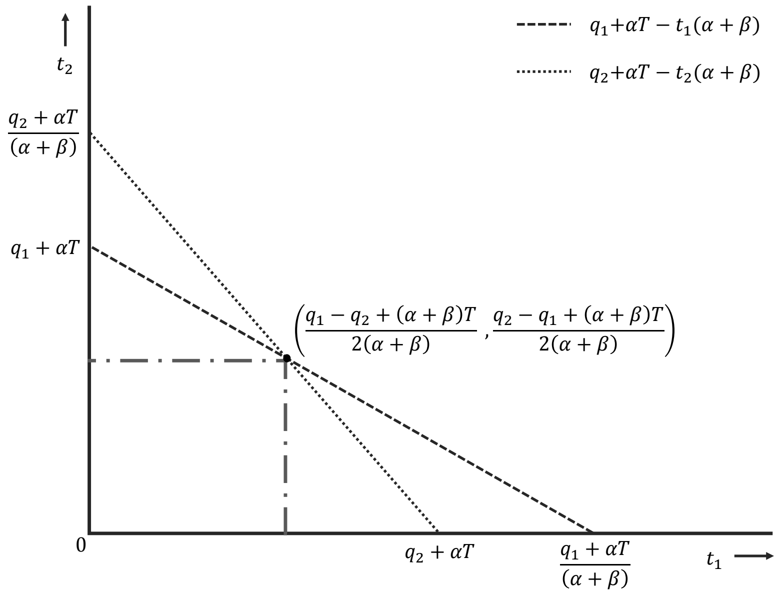

Figure 2.1: Nash equilibrium of two phase

could be figured out and then to decide the green time for next green phase.

Tung et al. proposed a traffic signal timing adjustment strategy based on genetic algorithm [9]. The strategy is based on the cellular automata model and it uses the informations like starting and destination points of all vehicles to optimize the signal to minimize travel time. In the genetic algorithm (GA) of the strategy, the chromosome is defined as the traffic information of whole intersections which includes the vertical green time, horizontal green time, current signal state, and remain time. The fitness value of the chromosome is defined as the average traffic time, thus the lower average traffic time will be higher fitness value. After genetic algorithm processing, the chromosome with the lowest average traffic time would be obtained that is the new signal timing of whole intersections. However, retrieving the informations of all vehicles is difficult in practice, the authors also proposed a improved expectation maximization (EM) method. By tuning the EM solution which based on GA solution with linear learning model, the significant features are observed that is vertical green time and horizontal green time of self and adjacent intersection.

2.3 Hybrid Approach on Signal Control

Yi et al. proposed a fuzzy logic controller with adaptive dynamic programming for isolated intersection signal optimization [1]. The strategy of this controller is to add extended green time to current phase depending on the information from vehicle detectors. The fuzzy logic controller has two input and one output, the inputs are number of vehicles in green phase (Q_Green) and other red phases (Q_Red), output is the action of the controller that extending or terminating the green phase. The membership functions is shown in Figure 2.2 and rule shown in Table 2.1. By optimizing the membership functions of the fuzzy logic controller with adaptive dynamic programming, the average delay time can be reduced especially in heavy traffic volume and traffic sudden changing.

Table 2.1: Fuzzy rule of [1]

Outpur		Q_Red			
		Zero	Small	Medium	Big
Q_Green	Zero	T	T	T	T
	Small	E	E	T	T
	Medium	E	E	E	T
	Big	E	E	E	E

Bi et al. proposed a Type-2 fuzzy logic controller (T2FLC) for single intersection [2] that combining fuzzy logic and genetic algorithm. The method considers

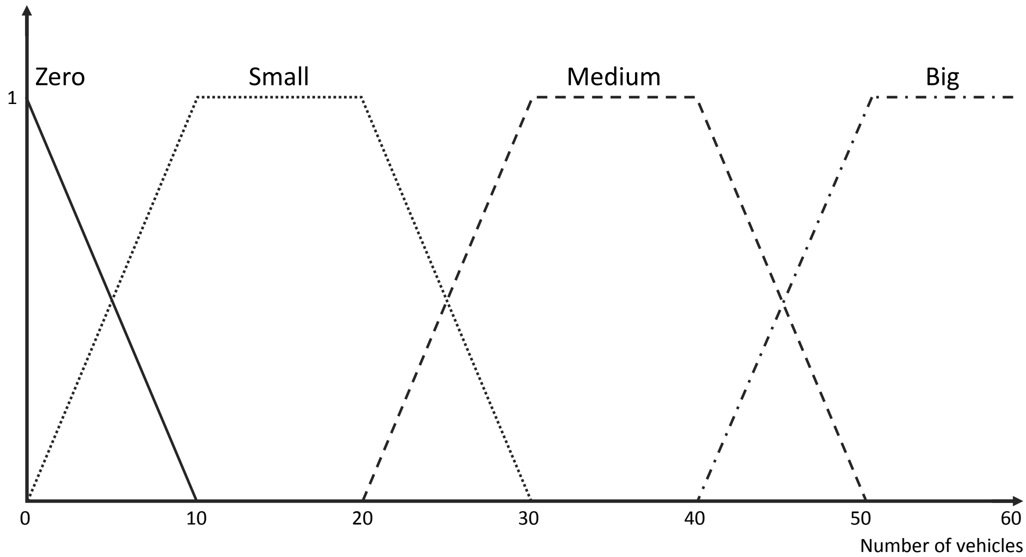

Figure 2.2: Membership functions of [1]

the queue length of the roads and divided them into three fuzzy language partitions ”short”, ”medium, ”long”. Then use queue length of next green phase (QG) and queue length of red phase (QR) as the two inputs of T2FLC to obtain the green time (T) of next phase. The membership functions is shown in Figure 2.3 and rule shown in Table 2.2. In order to validity of the T2FLC, the authors select the queue length and average delay as criterion and propose queue length model and delay model. Finally, the authors use genetic algorithm to minimize the average delay that based on those models and obtain suitable green time.

Table 2.2: Fuzzy rule of [2]

Output (T)		QG		
		Short	Medium	Long
QR	Short	S	M	L
	Medium	S	M	L
	Long	S	S	M

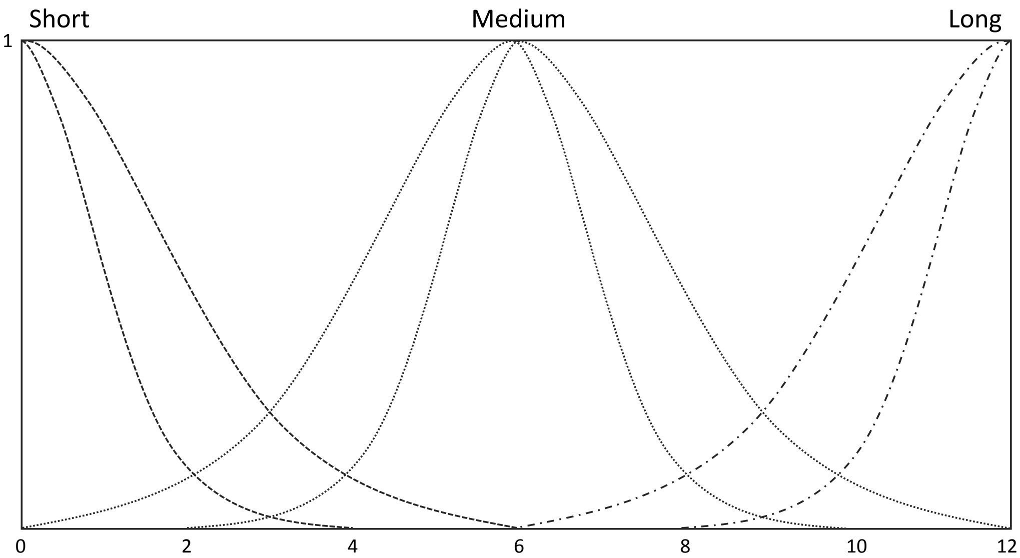

Figure 2.3: Type-2 Gaussian membership functions of [2]

Table 2.3: Traffic signal timing approaches

Year	Literature	Scheme	Used Parameters	Objectives
2010	[19]	Time-space diagrams	Queue length Vehicle speed	Minimize queue length Minimize delay time
2013	[8]	Game theory	Queue length Arrival rates Departure rates	Minimize queue length
2014	[9]	Genetic algorithm	Starting and destination points of vehicle Average signal timing	Minimize travel time
2008	[1]	Fuzzy logic Adaptive Dynamic Programming	Queue length Arrival vehicles	Minimize delay time
2013	[2]	Fuzzy logic Genetic algorithm	Queue length Arrival vehicles Departure rates	Minimize delay time

Chapter 3

Proposed System Architecture

In this chapter, we introduce the traffic definitions and explain the proposed Cyber-Physical Traffic Control Systems (CPTCS) architecture as shown in Figure 3.1.

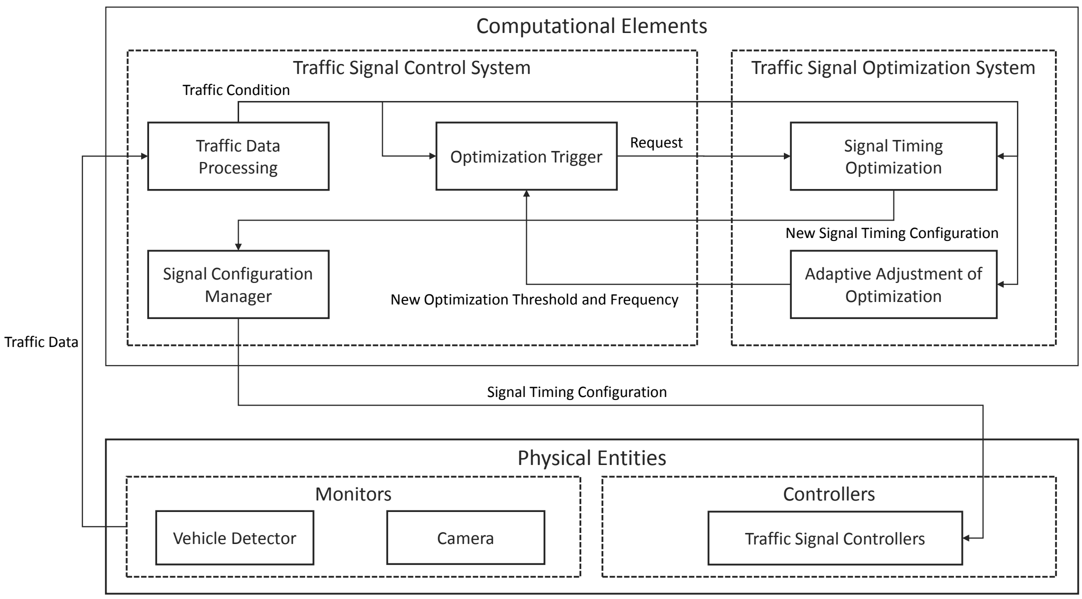

Figure 3.1: Architecture of Cyber-physical Traffic Control Systems

3.1 Definitions

In this section, we give the definitions for road, intersection, and signal in our system.

3.1.1 Traffic Configuration

Road

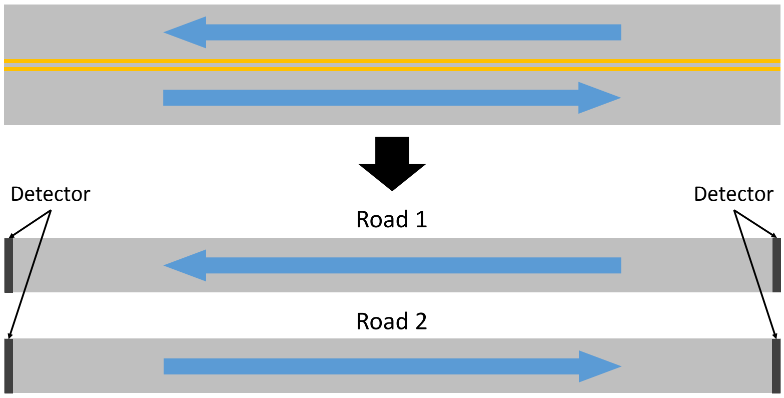

Figure 3.2: Straight road with two directions

The common road is shown in Figure 3.2 which has two opposite directions. In our system, for simplicity only one-way road are considered. Thus a two-way road can be considered as two roads, such as road 1 is east to west and road 2 is west to east. For each monitored road, we assume they have detectors in both the upstream and the downstream to detect the vehicles entering and leaving the road.

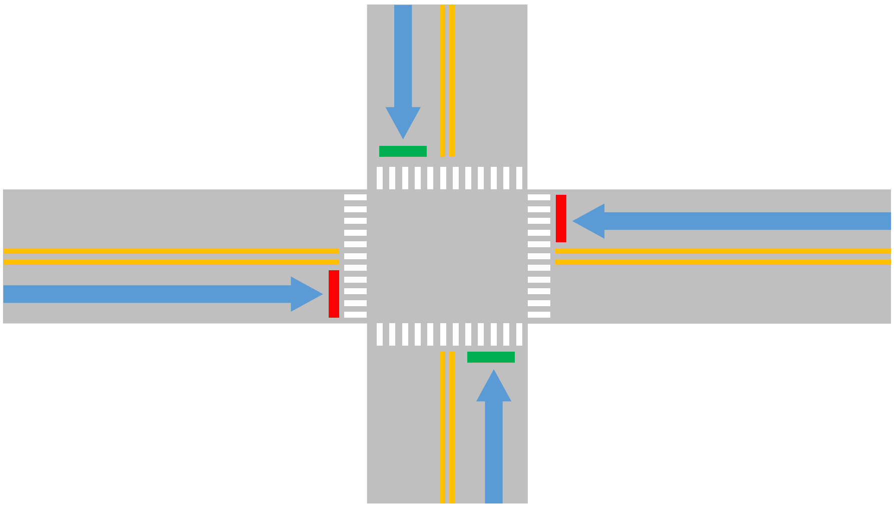

Figure 3.3: Four-way intersection (crossroads)

Table 3.1: Intersection parameters

Parameter	Description
Minimum Green Time (GT_m)	Minimum green time allowed to set.
Maximum Green Time (GT_M)	Maximum green time allowed to set.
Basic Optimization Interval (OI_B)	Basic optimization interval of the intersection
Maximum Optimization Interval (OI_M)	Maximum optimization interval allowed to set
Optimization Threshold (OT)	The value against which the current IAWR is checked
Optimization Interval (OI)	The number of cycles between two successive optimization check
Stability (S_{self})	The value to represent the stability of the intersection, the range of stability is from 0 to 10

Intersection

The common four-way intersection is shown in Figure 3.3, which is composed by four roads with vehicles entering the intersection. For each intersection, it has own intersection parameters that shown in Table 3.1.

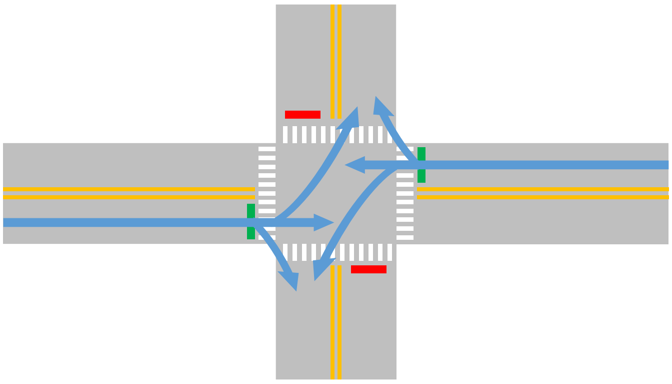

Figure 3.4: Phase 1 of four-way intersection

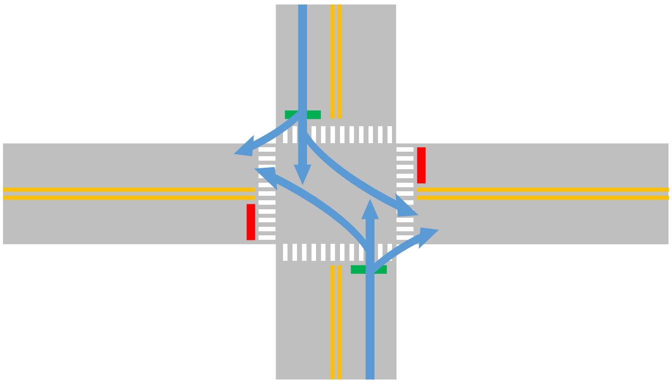

Figure 3.5: Phase 2 of four-way intersection

Signal Timing Configuration

A signal timing configuration includes the signal timing of every phases and each phase has its own time of green light, yellow light, and red light that we called green time, yellow time, and red time respectively. For the intersection shown in Figure 3.3, signal timing configuration can be divided into two phases as shown in Figure 3.4 and Figure 3.5. In the intersection, each road belongs to a phase and each phase has its own green time and yellow time to allow vehicle passing, as shown in Figure 3.6. The intersection signals continuously countdown so that during green light the vehicles of each phase can take turns to pass the intersection. A cycle is defined as the time period from the start time to the end

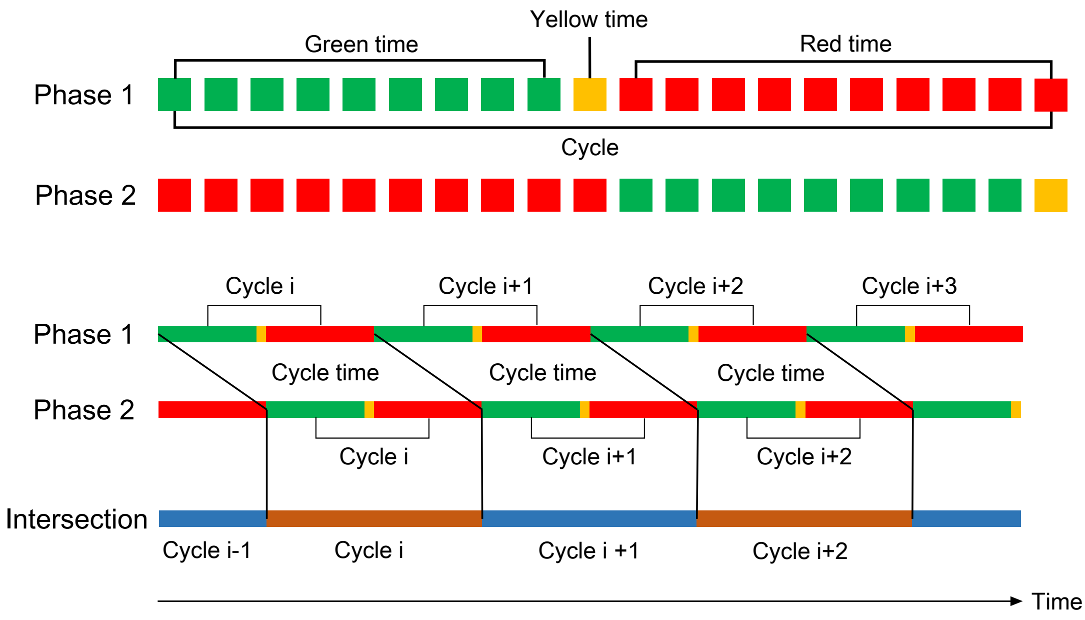

Figure 3.6: Signal timing configuration of an intersection

time of a phase, all phases in an intersection have same cycle time. In order to recognize the cycle of intersection, the cycle number of intersection is same to the cycle number of last phase.

3.2 Physical Entities

In this section, we explain the following of each physical entity.

3.2.1 Monitors

Monitors are used to detect and record information on the number of vehicles entering and leaving a road. Detection is performed using vehicle detector or a camera with image processing.

At the end of a cycle, the recorded traffic data as shown in Table 3.2 will be sent to the traffic signal control system and stored.

Table 3.2: Data of single cycle

Name	Description
Cycle Time (CT)	Cycle time of this cycle
Arrived vehicles (V)	Number of arrived vehicles in the cycle.
Arrived Rate (VR)	$\frac{V}{CT} \times 60$
Passed vehicles (PV)	Number of passed vehicles during green light and yellow light.
Waiting Vehicles (WV)	Number of vehicles waiting at a red light.
Waiting Time (WT)	Total time of all vehicles spent to wait.
Waiting Rate (WR)	$\frac{WV}{V}$

3.2.2 Controllers

Controllers are responsible for controlling the traffic signal at an intersection. The traffic signal configuration manager issue commands as shown in Table 3.3

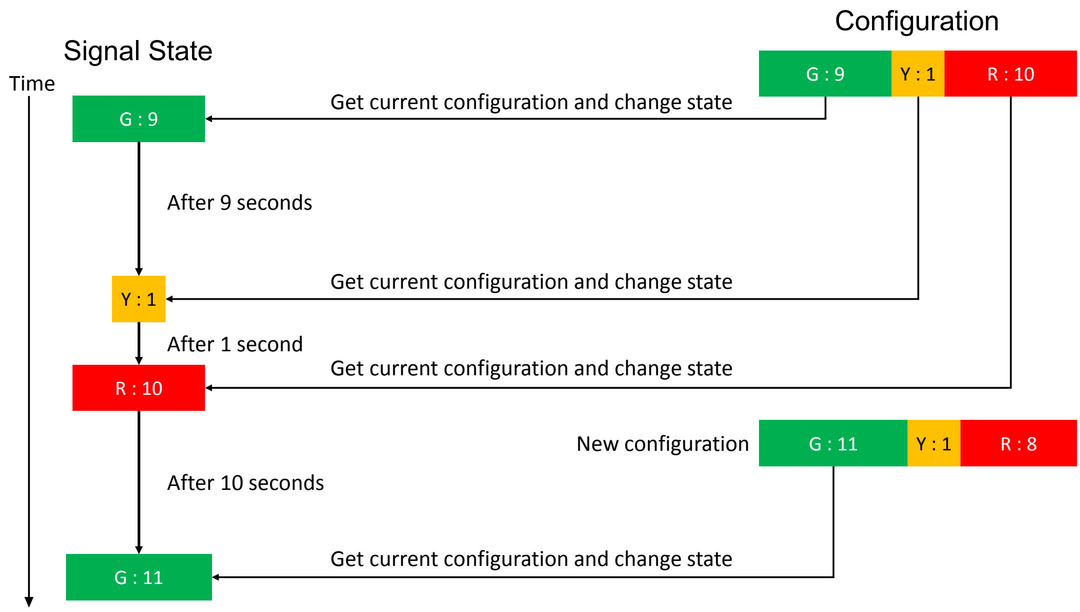

Figure 3.7: Mechanism of controller in signal display

to control the signal controller. After the signal timing configuration of the controllers are initialized, the signal controller will continuously countdown to display the signals of every road. When the controller receives the set signal configuration command and the new signal timing configuration, the current configuration is replaced by the new one, without affecting the current signal display. For safety, the current signal countdown will continue until the end of current light. Thus

Table 3.3: Commands of signal configuration manager sent to signal controller

Command	Description
Initialize	Signal configurations manager sends the intersection signal timing configuration to the controller.
Set Signal Configuration	Controller applies the new signal timing configuration of the intersection.
Synchronize	Controller directly replaces the current signal state of each phase with the specified signal state, the signal will display the new state immediately.

means new signal timing configuration are only applied staring from the next displayed light. The mechanism of controller in signal display is shown in Figure 3.7. In addition, regularly synchronization is performed so as to prevent the deviation of signal state between controller and system.

3.3 Computational Elements

In this section, we explain the function of all computational elements. The flow chart of the system operation process is as shown in Figure 3.8.

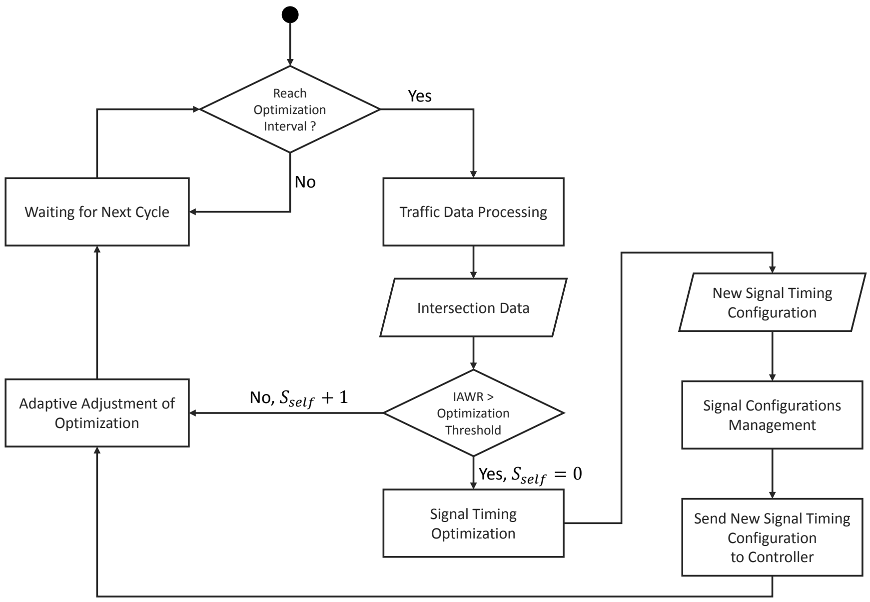

Figure 3.8: Flow chart of the system operation process

3.3.1 Traffic Data Processing

Traffic data processing is performed for multiple cycles. The average traffic data of roads and intersection data are calculated for C cycles. For intersection data calculation, we use a weighted average because roads with higher volumes are more important than the others. Definitions for average traffic data and intersection data are give in Table 3.4.

Note that we choose the intersection average waiting rate (IAWR) as the basic optimization criterion in our proposed traffic control system.

Table 3.4: Data of multiple cycles

Name	Description
Calculated Cycles (C)	The number of calculated cycles for estimating the traffic condition during these cycles.
Road Average Data	
Average Arrived Vehicles (V_{avg})	$\frac{\sum_{i=1}^{C} V^i}{C}$
Average Arrived Rate (V_{min})	$\frac{\sum_{i=1}^{C} VR^i}{C}$
Average Waiting Vehicles (WV_{avg})	$\frac{\sum_{i=1}^{C} WV^i}{C}$
Average Waiting Time (WT_{avg})	$\sum_{i=0}^{C} \frac{WT^i}{V^i}/C$
Average Waiting Rate (WR_{avg})	$\frac{\sum_{i=1}^{C} WR^i}{C}$
Intersection Data	
Composed Roads (R)	The number of composed roads
Intersection Arrived Vehicles (V_{inte})	$\sum_{j=1}^{R} V_{avg}^j$
Intersection Average Waiting Time ($IAWT$)	$\sum_{j=1}^{R} \frac{V_{avg}^j \times WT_{avg}^j}{V_{inte}}$
Intersection Average Waiting Rate ($IAWR$)	$\sum_{j=1}^{R} \frac{V_{avg}^j \times WR_{avg}^j}{V_{inte}}$

3.3.2 Optimization Trigger

The optimization trigger is responsible for determining whether the intersection should be optimized or not. If the current IAWR is greater than the optimization threshold than optimization performed and set the S_{self} to 0, otherwise the stability counter increment by 1. Adaptively changing optimization threshold and optimization interval is very important for big data analysis in smart cities. This work proposes a method for adaptive adjustment of optimization.

3.3.3 Signal Timing Optimization

Signal timing optimization calculates a new signal timing configuration according to the average traffic data of the composed roads. The detail calculation will described in the following Chapter 4. After completing the calculation, the new signal timing configuration will be sent to the signal configurations manager.

3.3.4 Adaptive Adjustment of Optimization

Based on the collected and calculated data of the target intersection and they of adjacent intersections, an appropriate optimization threshold and an optimization interval are dynamically determined, which we call adaptive adjustment of optimization. Detailed calculations are given in the following Chapter 4.

3.3.5 Signal Configurations Manager

Signal configuration manager is responsible for storing signal timing configurations of all intersections and interaction with the traffic signal controllers. If a new signal timing configuration is received from the signal timing optimization, it will be stored and sent to corresponding traffic signal controller.

Chapter 4

Optimization Methods

In this chapter, we introduce the two parts of our proposed adaptive timing optimization, namely the signal timing optimization and the adaptive adjustment of optimization.

4.1 Signal Timing Optimization

In the section, we explain our proposed genetic algorithm-based signal timing optimization, including chromosome design, fitness function, and optimization steps.

4.1.1 Chromosome Design

In our optimization method, the chromosome represents an intersection signal timing configuration which consists only of green time of each phase. Since the yellow time is very short compared to a full cycle time of intersection we can ignore

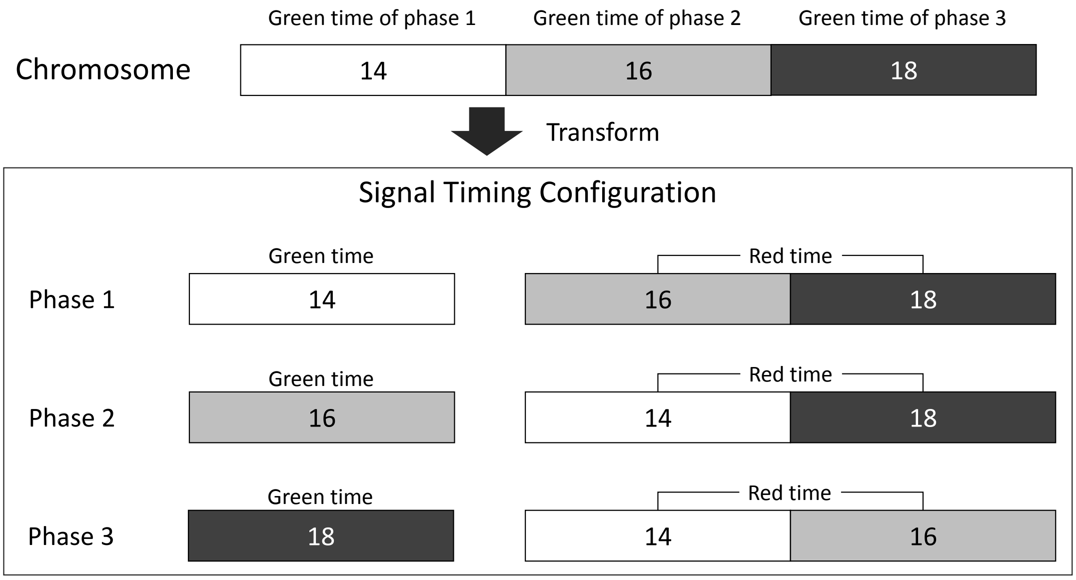

Figure 4.1: Chromosome of three phases intersection

it. The red time of each phase is the sum of the green times of other phases thus there is no need to explicitly record it.

The chromosome length is the number of green times, if the chromosome has three green times that the chromosome length is three. The maximum and minimum of the green time are determined by the intersection parameters. For example, if the intersection has three phases and the maximum and minimum of green times are 30 and 10, the chromosome will have three values that represent the green time of the three phases and each value would be between the maximum and minimum values as shown in Figure 4.1 shows.

4.1.2 Fitness function

Our goal is to reduce the IAWR of an intersection, thus we directly use IAWR as the fitness value and select the chromosome with the lowest fitness value. As equation 4.1, the fitness function use the estimated IAWR $IAWR_e$ means the IAWR that if we apply the new signal timing configuration which transform by the estimated chromosome to the current traffic volumes of the intersection.

$$Fitness = IAWR_e \tag{4.1}$$

To calculate the $IAWR_e$, we using the new signal timing configuration and road average data obtained from traffic data processing which shown as Table 3.4. The $IAWR_e$ can be expressed by Equation 4.2 that is similar to the depiction of IAWR except WR_{avg} is replaced by WR_e, which is the estimated waiting rate of a road and it can be expressed as Equation 4.3, where V_{min} is the average arrived rate of road average data , T_G and T_R are the green time and red time which will be applied to the road, and RT is the reservation time. The reservation time of a road is defined as the total tome required for all waiting vehicles in the previous cycle to leave the road.

$$IAWR_e = \sum_{j=1}^{R} \frac{V_{avg}^j \times WR_e^j}{V_{inte}} \tag{4.2}$$

In the WR_e calculation, we assume the vehicle arrivals are uniform distribution, so the total number of vehicles arrived in a cycle can be estimated as $V_{min} \times (T_G + T_R)$ The number of waiting vehicles in a cycle can be estimated as $V_{min} \times T_R$.

However, at the start of the current cycle, the waiting vehicles from the previous cycle will need the RT to leave from the road. During this time RT leaving the previous waiting vehicles, some of the newly arrived vehicles in the current cycle need to wait during the green time of the current cycle. For this reason, the number of waiting vehicles add includes the vehicles during RT $V_{min} \times RT$.

$$WR_e = \frac{WaitingVehicle}{TotalVehicle} = \frac{V_{min} \times RT + V_{min} \times T_R}{V_{min} \times (T_G + T_R)} \tag{4.3}$$

The design of the fitness function given in Equation 4.1 allows large deviations in the green time although the difference between traffic volumes is extremely small. Accordingly, we add the signal deviation factor (SDF) to prevent the situation as shown in Equation 4.4, where α and β are weight for $IAWR_e$ and SDF respectively. When the difference between traffic volumes is extremely small, the difference of $IAWR_e$ between the chromosome with large deviations and the chromosome with small deviations are extremely small that almost between 0.01 to 0.02. In this situation, the SDF has highly influential in determining the chromosome.

$$Fitness = IAWR_e \times \alpha + SDF \times \beta \tag{4.4}$$

SDF can be expressed as in Equation 4.5, where σ is the standard deviation of the green time and MD is the difference between the maximum and the minimum green time of the intersection parameters.

$$SDF = \frac{\sigma}{MD} \tag{4.5}$$

After tuning, we set the weights α to 1 and β to 0.1.

Reservation Time Calculation

For the reservation time calculation, we use the Intelligent Driver Model (IDM) which is a car-following model [20] to simulate vehicles behavior on the road. The IDM function can be expressed as Equation 4.6 and the parameters of IDM are as given in Table 4.1. Using IDM, we can get the next acceleration or deceleration value of a vehicle.

$$IDM(\alpha, \alpha - 1) = a\left(1 - \left(\frac{v_\alpha}{v_0}\right)^\delta - \left(\frac{s_0 + v_\alpha T}{s_\alpha} + \frac{v_\alpha \Delta v_\alpha}{2\sqrt{ab}s_\alpha}\right)^2\right) \tag{4.6}$$

Table 4.1: Parameters of IDM

Parameter	Description
v_α	The velocity of vehicle
x_α	The position of vehicle
l_α	The length of vehicle
Δv_α	The velocity difference between vehicle α and $\alpha - 1$ $\Delta v_\alpha = v_\alpha - v_{\alpha-1}$
s_α	The net distance between vehicle α and $\alpha - 1$ $S_\alpha = x_{\alpha-1} - x_\alpha - l_{\alpha-1}$
v_0	The desired velocity, we set it to the speed limit of located road
s_0	The minimum desired net distance between two vehicles
T	The desired time headway
a	The acceleration
b	The braking deceleration
δ	The exponent that is usually set to 4

The reservation time calculation is as shown in Algorithm 1. Since the reservation time calculation is to calculate the total time required for all waiting vehicles to leave the road, we can assume the velocity of each waiting vehicle is 0 and the net distance between each vehicle is minimum. Further, we iteratively calculate the velocity of each vehicle and according to the velocity, each vehicle is moved until the last vehicle leaves the road. The total time required for this process is called the reservation time.

Algorithm 1: Reservation Time Calculation

Input:
WV : Number of waiting vehicles;
l_v : Length of vehicle;
s_0 : Minimum desired net distance;
v_0 : Desired velocity;
Output:
RT : Reservation time;
Variable:
G : Position of goal;
Q : Queue of waiting vehicles;
V_i : i-th waiting vehicles;
v_c : Current velocity;
v_n : Velocity of next second;

$Q = \emptyset$;
$RT = 0$;
$G = WV \times (l_v + s_0) - l_v$;
for $i = 1$ *to* WV **do**
    Create a new vehicle V_i and set its position at $G - (i \times (l_v + s_0) - l_v)$;
    Add V_i to Q;
while $V_{WV}.position < G$ **do**
    for $i \leftarrow 1$ *to* WV **do**
        $v_c = V_i$.velocity;
        if $i == 1$ **then**
            $v_n = v_c + accelerationofV_i$;
            if $v_n > v_0$ **then**
                $v_n = v_0$;
        else
            $v_n = v_c + IDM(V_i, V_{i-1})$; //Equation 4.6
        V_i.position $+= (v_c + v_n)/2$;
    $RT += 1$;
return RT;

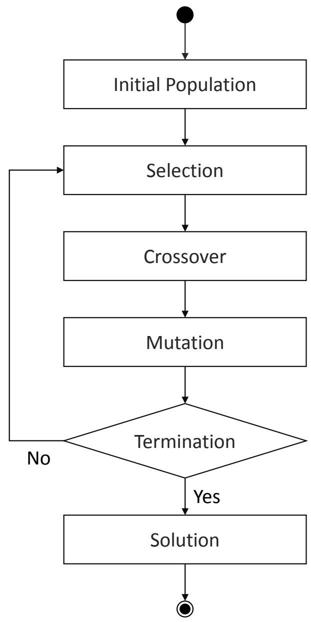

Figure 4.2: Flow chart of the genetic algorithm-based signal timing optimization

4.1.3 Optimization Steps

The flow chart of GA-based optimization steps is as shown in Figure 4.2. The parameters of the genetic algorithm are as shown in Table 4.2. The steps from selection to mutation are defined as a generation and are explained in the following.

Table 4.2: Parameters of genetic algorithm

Parameter	Value
Population Size (PS)	50
Generation (G)	50
Selection Operator	Tournament
Crossover Probability (P_{cro})	0.7
Mutation Probability (P_{mut})	0.05

Initial Population

At first, we generate the chromosomes and put them into the genetic pool, the number of generated chromosomes is equal to the population size. The initial population step is as shown in Algorithm 2.

Algorithm 2: Initial Population

Input:
p : Number of phases;
GT_M : Maximum of green time;
GT_m : Minimum of green time;
PS : Population size;
Output:
GP : Genetic pool, a set of chromosomes

```
GP = ∅;
for i = 1 to PS do
    Generate a chromosome c with p random values between GT_m and GT_M;
    Add c to GP;
return GP;
```

Selection

In the selection step, we use tournament selection to choose chromosomes with better fitness from the genetic pool. For each tournament selection, we randomly select 5 chromosomes from the genetic pool and copy the best fitness chromosome into the new genetic pool. The selection is repeated until the population size of new genetic pool a pre-defined limit. At the end of selection, we replace the genetic pool with the new one to be used in the next step. The selection step is as shown in Algorithm 3.

Algorithm 3: Selection

Input:
GP : Genetic pool;
PS : Population size;
Output:
GP : Genetic pool, a set of chromosomes;
Variable:
GP_n : New genetic pool;

```
GP_n = ∅;
for i = 1 to PS do
    Randomly select 5 chromosomes from GP and copy the best fitness one
    into GP_n;
Replace GP with GP_n, GP = GP_n;
return GP;
```

Crossover

In the crossover step, we randomly select two chromosomes from the genetic pool as parent chromosomes and according to the crossover probability to determine whether to crossover or not. If crossover is performed, two child chromosomes will be produced and put into the new genetic pool, while the parent chromosomes are removed from the genetic pool. If no crossover to be performed, the parent chromosomes are directly put into the new genetic pool. At the end of the crossover, the genetic pool is replaced with the new one to be used in the next step. The crossover step is as shown in Algorithm 4.

Mutation

At the last step, we determine if each chromosome in the genetic pool is to be mutated or not according to the given mutation probability. If it is to be mutated, all the green times in the chromosome will be randomly incremented by -5 to 5

Algorithm 4: Crossover

Input:
GP : Genetic pool;
PS : Population size;
P_{cro} : Crossover probability;
CL : Chromosome length;
Output:
GP : Genetic pool;
Variable:
GP_n : New genetic pool;
r : A float between 0 and 1;;
C_{p1} : Parent chromosome 1;
C_{p2} : Parent chromosome 2;
C_{c1} : Child chromosome 1;
C_{c2} : Child chromosome 2;
cp : Crossover point;

$GP_n = \emptyset$;
for $i = 1$ *to* $\frac{PS}{2}$ **do**
  Randomly select 2 chromosomes from GP as parent chromosomes C_{p1} and C_{p2};
  r = random float between 0 and 1;
  if $r \leq P_{cro}$ **then**
    cp = random integer between 2 and CL;
    for $j = 1$ *to* CL **do**
      if $j < cp$ **then**
        C_{c1}.phase j = C_{p1}.phase j;
        C_{c2}.phase j = C_{p2}.phase j;
      else
        C_{c1}.phase j = C_{p2}.phase j;
        C_{c2}.phase j = C_{p1}.phase j;
    Put the child chromosome C_{c1} and C_{c2} into GP_n;
  else
    Put the parent chromosomes C_{p1} and C_{p2} into GP_n;
Replace GP with GP_n, $GP = GP_n$;
return GP;

seconds. The mutation step is as shown in Algorithm 5.

Algorithm 5: Mutation

Input:
GP : Genetic pool;
PS : Population size;
P_{mut} : Mutation probability;
CL : Chromosome length;
GT_M : Maximum of green time;
GT_m : Minimum of green time;
Output:
GP : Genetic pool;
Variable:
r : A float between 0 and 1;;
t : A integer between -5 and 5;
GP^i : The i-th chromosome in genetic pool;

```
for i = 1 to PS do
    r = random value between 0 and 1;
    if r ≤ P_mut then
        for p = 1 to CL do
            t = random integer between 5 and -5;
            GP^i.phase p = GP^i.phase p + t ;
            if GP^i.phase p > GT_M then
                GP^i.phase p = GT_M;
            if GP^i.phase p < GT_m then
                GP^i.phase p = GT_m;
return GP;
```

Termination

After completion of the above steps, the optimization method will check if the termination condition is satisfied or not. Here, in this work, the number of generations is used the termination condition. Therefore, the optimization will stop when the number of generations has reached the upper bound; otherwise, the next generation will be produced by repeating the above steps from selection to mutation.

Solution

Finally, we will obtain a best fitted chromosome as the solution which will be sent to the signal configurations manager.

4.2 Adaptive Adjustment of Optimization

In adaptive adjustment of optimization, we use the stability of the intersection (S_{self}) and the stability of adjacent intersections to calculate the threshold and interval. Since an intersection usually has 3 or more adjacent intersections, the average stability value is used to represent the stability of adjacent intersections.

The average stability of adjacent intersections (S_{adj}) can be expressed as Equation 4.7, where N is the number of adjacent intersections, V^i_{avg} is the average number of arrived vehicles which come from i-th adjacent intersection, V_{inte} is the total number of arrived vehicles from adjacent intersections, and S_i is the stability of i-th adjacent intersection. V_{avg} and V_{inte} are obtained from the traffic data processing. Figure 4.3 shows an example for the average stability of four adjacent intersections.

$$S_{adj} = \sum_{i=1}^{N} \frac{V^i_{avg} \times S_i}{V_{inte}} \tag{4.7}$$

4.2.1 Threshold Adjustment

For threshold adjustment, we want to make the optimization able to react rapidly when the traffic volume is changed, and prevent unnecessary optimization when the traffic volume is stable. For this reason, we set the threshold close to IAWR but higher and gradually decreased when the traffic volume is stable.

If the optimization was just performed, the next optimization will only be performed when the IAWR increased significantly to prevent unnecessary opti-

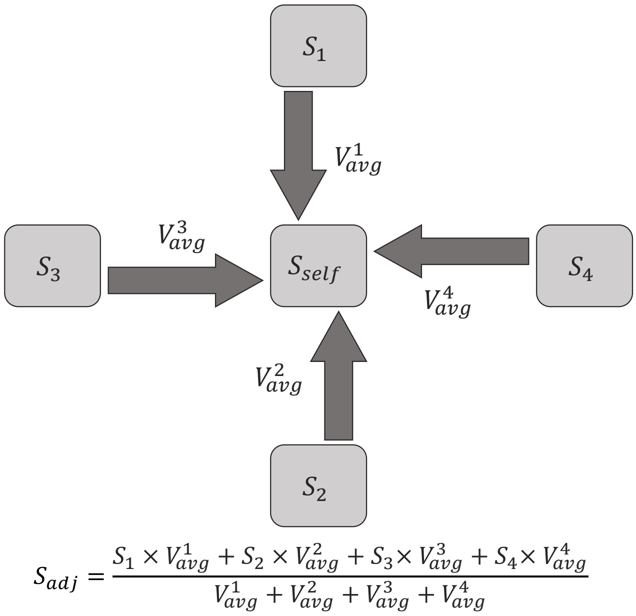

Figure 4.3: Average stability of four adjacent intersections

mization. On the contrary, when the traffic volume is stable, a slight increment of IAWR would trigger the optimization to prevent IAWR continually increasing.

The proposed threshold adjustment method is as shown in Algorithm 6. The new threshold is calculated by weighted average calculation, so it will be between the current threshold and IAWR. With an increase in stability (S_self), the new threshold will be set gradually closer to the current IAWR until the new threshold is the average of current threshold and IAWR.

4.2.2 Interval Adjustment

For interval adjustment, we want to reduce the frequency of optimization checking because frequent checking will waste the computational power and decrease computational efficiency especially when traffic volume is stable.

If the stability of the intersection is high, there will be no significant changes in

Algorithm 6: Threshold Adjustment

Input:
$IAWR$: Current IAWR;
OT_c : Current threshold, $T_c \in [0,100]$;
S_{self} : Stability of the intersection, $S_{self} \in [0,10]$;
R_M : Maximum allowable increased rate of IAWR, $R_M \in [0,1]$;
Output:
OT_n : New threshold;
Variable:
α : Weight value ;

```
if S_self == 0 then
    OT_n = Min(IAWR × (1 + R_M), 100);
else
    α = Min( S_self / Max(S_self), 0.5 )
    OT_n = α × IAWR + (1 − α) × OT_c;
return OT_n;
```

if $S_{self} == 0$ **then**
  $OT_n = Min\left(IAWR \times (1 + R_M)\right, 100);$
else
  $\alpha = Min\left(\frac{S_{self}}{Max(S_{self})}, 0.5\right)$
  $OT_n = \alpha \times IAWR + (1 - \alpha) \times OT_c;$
return OT_n;

traffic volume and thus the frequency of optimization checking should be reduced. In constant, if the stability of the intersection is low, the frequency of optimization checking should be increased to ensure optimization is timely such that traffic congestion can be alleviated. Besides, we also consider the stability of adjacent intersections because the changes in traffic volumes of adjacent intersections will affect the target intersection and cause the changes in traffic volume.

Therefore, we adjust the optimization interval according to the stability of the target intersection (S_self) and the stability of adjacent intersections (S_{adj}). The calculation of new optimization interval (OI_n) is as shown in Algorithm 7.

Algorithm 7: Interval Adjustment

Input:

S_{self} : Stability of the intersection, $S_{self} \in [0,10]$;

S_{adj} : Average stability of adjacent intersections, $S_{adj} \in [0,10]$;

OI_B : Basic optimization interval ;

OI_M : Maximum optimization interval ;

Output:

OI_n : New optimization interval;

Variable:

C : Curve factor ;

RI : Rate of increment ;

RI_M : Maximum rate of increment ;

Calculate the maximum rate of increment RI_M ;

$RI_M = \frac{OI_M}{OI_B} = \frac{\left(Max\left(S_{self}\right)+1\right)\times\left(Max\left(S_{adj}\right)+1\right)+C}{\left(Max\left(S_{self}\right)+1\right)+C}$

Use the equation of RI_M to find the curve factor C ;

$C = \frac{\left(Max\left(S_{self}\right)+1\right)\times\left(Max\left(S_{adj}\right)+1\right)\times I_B-\left(Max\left(S_{self}\right)+1\right)\times OI_M}{OI_M-OI_B}$

Calculate the rate of increment RI and new optimization interval ;

$RI = \frac{\left(S_{self}+1\right)\times\left(S_{adj}+1\right)+C}{\left(S_{self}+1\right)+C}$

$OI_n = \lfloor OI_B \times RI \rfloor$;

return OI_n;

Chapter 5

Experiments

This chapter presents the experimental results of the proposed ATO. We introduce the experiment environment, compared methods, and experimental results.

5.1 Experimental Environment

In this section, we describe the simulation tool and simulation cases that we used for experiments.

5.1.1 Simulation Tool

In our experiments, we want to simulate the traffic of a part of the city, thus we need a microscopic traffic simulator. A microscopic traffic simulator SUMO [21] is used in many works [5][6][4]. In SUMO, it has car-following models, lane-change models, etc. to make the simulation more realistic. However, it is very complicated to implement our proposed CPTCS architecture in SUMO by modifying its source code.

Thus, we build a relatively simple traffic simulator which is written in C# language and implement CPTCS architecture and IDM [20] car-following model in the simulator. In the simulator, we can customize the road map and the traffic volumes to create a desired traffic for a part of the city. The traffic volume value is the average arrived vehicles per minute and according to this value, the simulator can randomly generate vehicles based on the Poisson distribution.

The source code of the simulator is released on GitHub [22].

5.1.2 Simulation Case

The simulation map as shown in Figure 5.1. The left side of the map is urban and right side of the map is outskirts, so most of the vehicles from the left side move to the right side and vice versa.

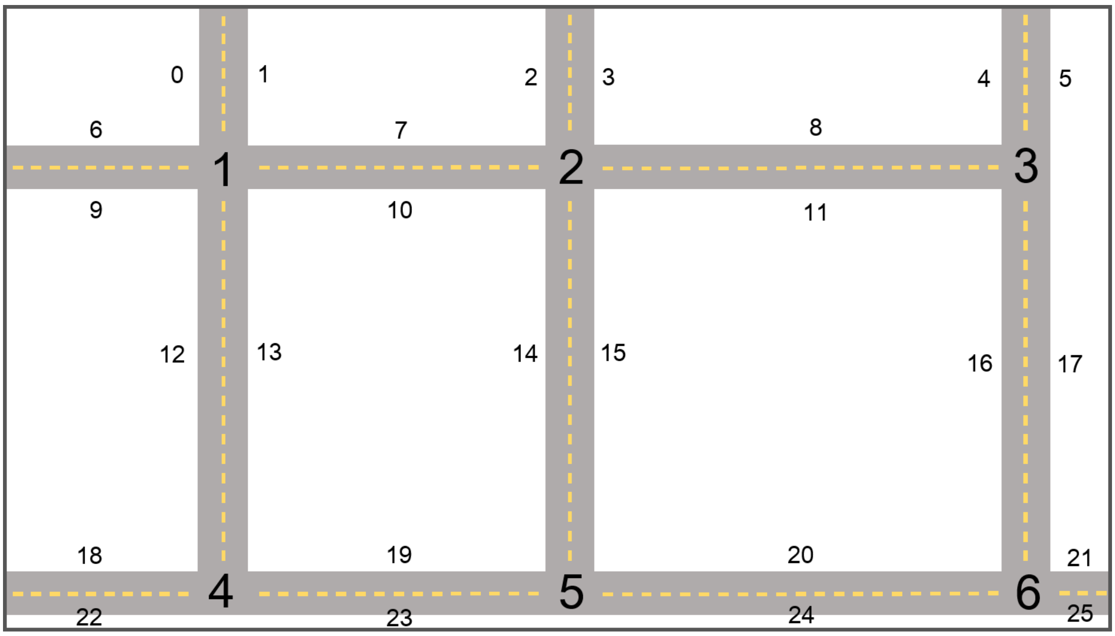

Figure 5.1: Simulation map

For this map, we create all day traffic volume simulation cases which start time from the AM 5:00 to PM 9:00. In this case, the time from AM 6:00 to AM:10:00 are the morning peak hours and time from PM:5:00 to PM 9:00 are evening peak hours, and the other time period are off-peak hours. The roads 0, 2, 4, 9, 21, 22 are roads with vehicles entering, so we set the traffic volumes of these roads. The traffic volume values as shown in Table 5.1, where each traffic volume value represents the arrived vehicles per minute.

Finally, the intersection parameters in this case as shown in Table 5.2.

5.2 Tuning for Adaptive Timing Optimization

In the threshold adjustment method, we must determine the maximum allowable increased rate of IAWR (R_M) of the Algorithm 6, so we perform an experiment to find a better value for R_M. In this experiment, we set R_M to 0.05, 0.1, and 0.15 and compare the optimization results.

5.2.1 Experimental Results

The IAWR of all day as shown in Figure 5.2, and the optimization results are shown in Table 5.3. The optimization gives the best result when R_M is set to 0.1, so we use the value 0.1 for R_M in our other experiments.

Table 5.1: Traffic volume values in all day case

Traffic Volume						
Time	Road 0	Road 2	Road 4	Road 9	Road 21	Road 22
AM 05:00 ~ AM 6:00	1	1	1	1	1	1
AM 06:00 ~ AM 6:30	3	3	3	2	3	2
AM 06:30 ~ AM 7:00	3	3	4	2	4	2
AM 07:00 ~ AM 7:30	5	5	6	3	5	3
AM 07:30 ~ AM 8:00	5	5	8	3	6	3
AM 08:00 ~ AM 8:30	7	7	7	3	9	3
AM 08:30 ~ AM 9:00	7	7	6	4	8	4
AM 09:00 ~ AM 9:30	5	5	5	4	6	4
AM 09:30 ~ AM 10:00	5	5	5	4	4	4
AM 10:00 ~ PM 5:00	3	3	3	4	3	4
PM 5:00 ~ PM 5:30	3	3	3	6	3	6
PM 5:30 ~ PM 6:00	3	3	3	6	3	6
PM 6:00 ~ PM 6:30	3	3	3	8	3	8
PM 6:30 ~ PM 7:00	3	3	3	10	3	10
PM 7:00 ~ PM 7:30	3	3	3	8	3	8
PM 7:30 ~ PM 8:00	3	3	3	6	3	6
PM 8:00 ~ PM 8:30	3	3	3	4	3	4
PM 8:30 ~ PM 9:00	3	3	3	4	3	4

Table 5.2: Intersection parameters in all day case

Parameter	Value
Minimum Green Time (GT_m)	30
Maximum Green Time (GT_M)	90
Basic Optimization Interval (I_B)	5
Maximum Optimization Interval (I_M)	40

5.3 Comparison of Optimization Results

In order to prove our proposed ATO method is effective in solving traffic congestion, we perform an experiment to compare the proposed ATO method with two

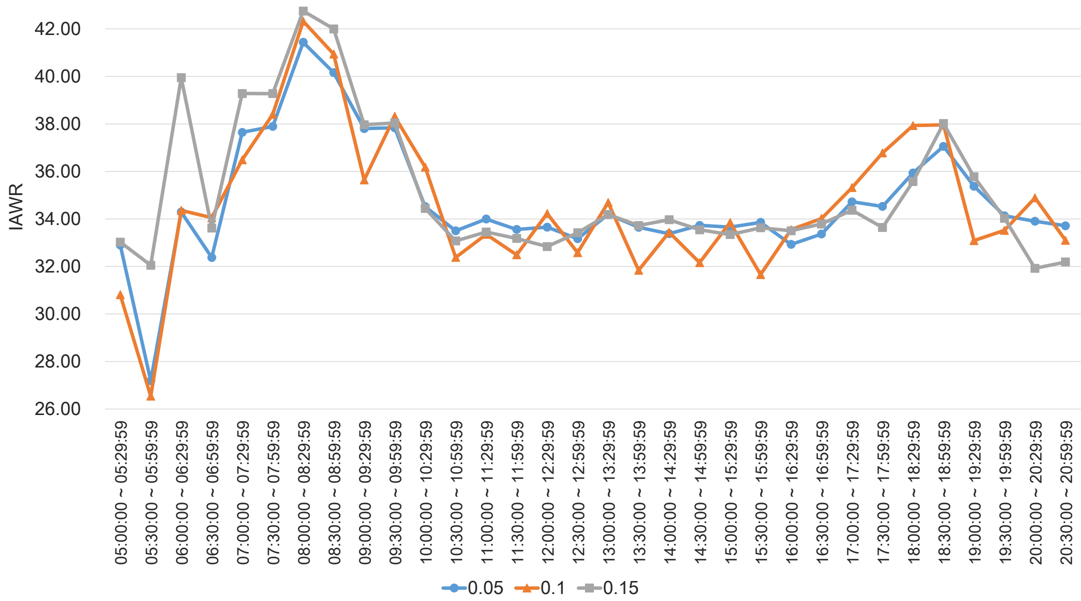

Figure 5.2: IAWR of all day with different R_M values

Table 5.3: IAWR comparison of different R_M values

Time	R_M	Average IAWR
All day (AM 5:00 $\sim$ PM 9:00)	0.05	34.69
	0.1	34.59
	0.15	35.34

other two methods. The first method for comparison is the fixed timing (FT) that is no optimization. The second method for comparison is the game theory-based signal timing optimization (GT) introduced in Chapter 2 [8].

In addition, we perform another experiment to find the optimization limit of ATO.

5.3.1 Experimental Results

In this experiment, we discuss the experimental results of morning peak hours, nigh peak hours, off-peak hours and all day separately.

Peak Hours

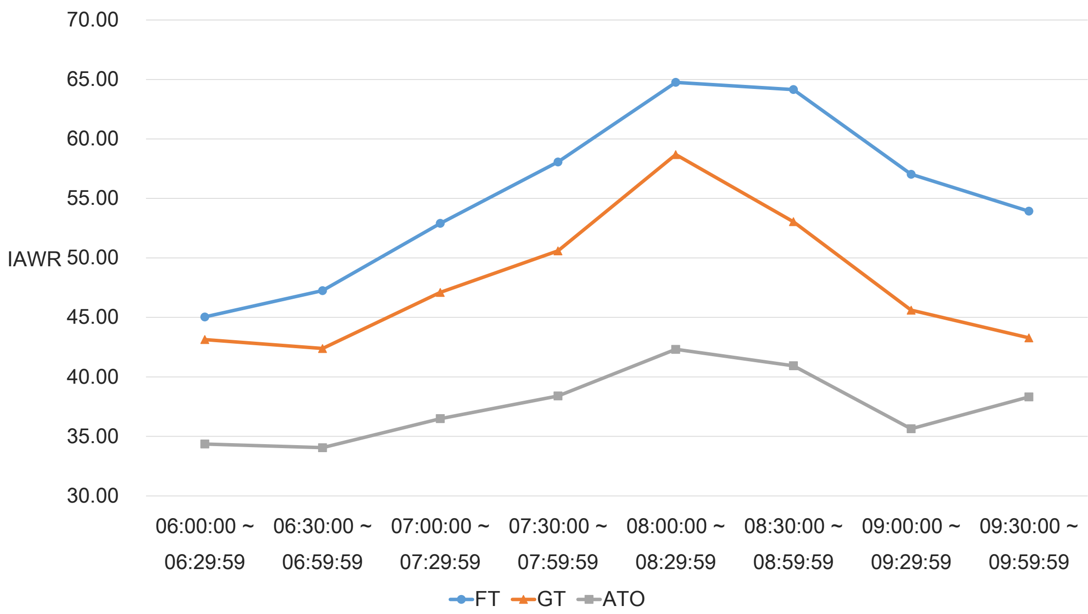

Figure 5.3: IAWR of morning peak hours

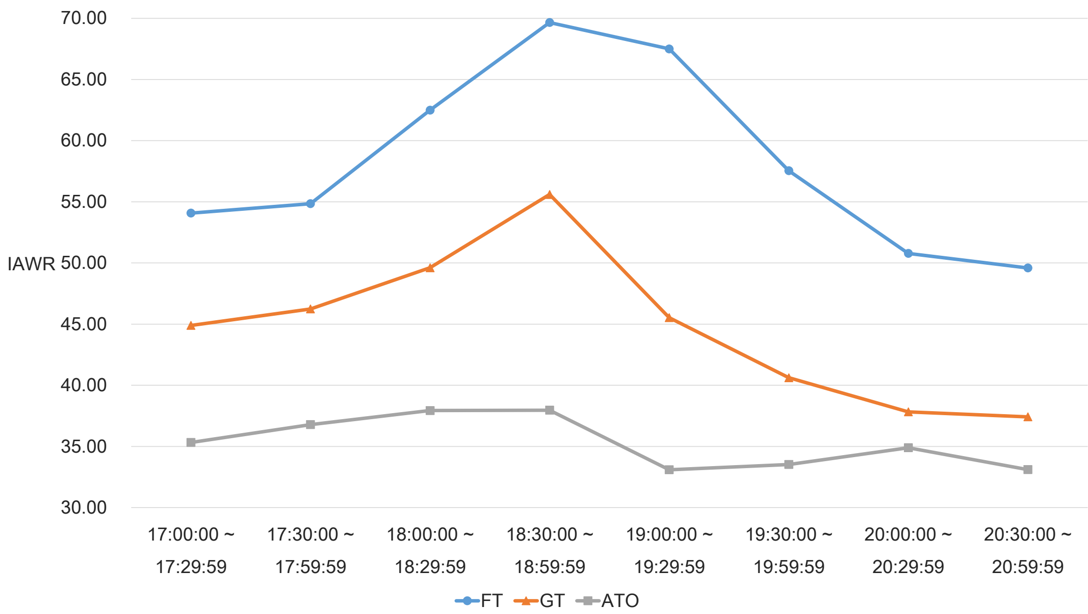

Figure 5.4: IAWR of evening peak hours

The IAWR of the two peak hours are as shown in Figure 5.3 and Figure 5.4, respectively. The IAWR comparison of the two peak hours is as shown in Table

5.4. For the two peak hours, both GT and ATO incur reductions in IAWR compared with FT. GT reduces the IAWR by 13% and 23% respectively in morning peak hours and evening peak hours. ATO reduces the IAWR by 32% and 39% respectively in morning peak hours and evening peak hours. As the results, we can observe ATO incurs a reduction in IAWR almost double that by GT.

Table 5.4: IAWR comparison of peak hours

Time	Method	IAWR	Reduce (%)
Morning peak Hours (AM 6:00 ∼ AM 10:00)	FT	55.4 0	0
	GT	47.98	13
	ATO	37.57	32
Evening Peak Hours (PM 5:00 ∼ PM 9:00)	FT	58.31	0
	GT	44.71	23
	ATO	35.32	39
Average	GT	46.35	18
	ATO	36.44	36

Off-peak Hours

The IAWR of off-peak hours is as shown in Figure 5.5, and the IAWR comparison of off-peak hours is as shown in Table 5.5. For the off-peak hours, both GT and ATO incur reductions in IAWR that compare with FT. GT reduces the IAWR by 15% and ATO reduces the IAWR by 33%. Both reductions in IAWR are a little less than that at peak hours, because the traffic volumes of off-peak hours are small, so the benefit of optimization is less obvious.

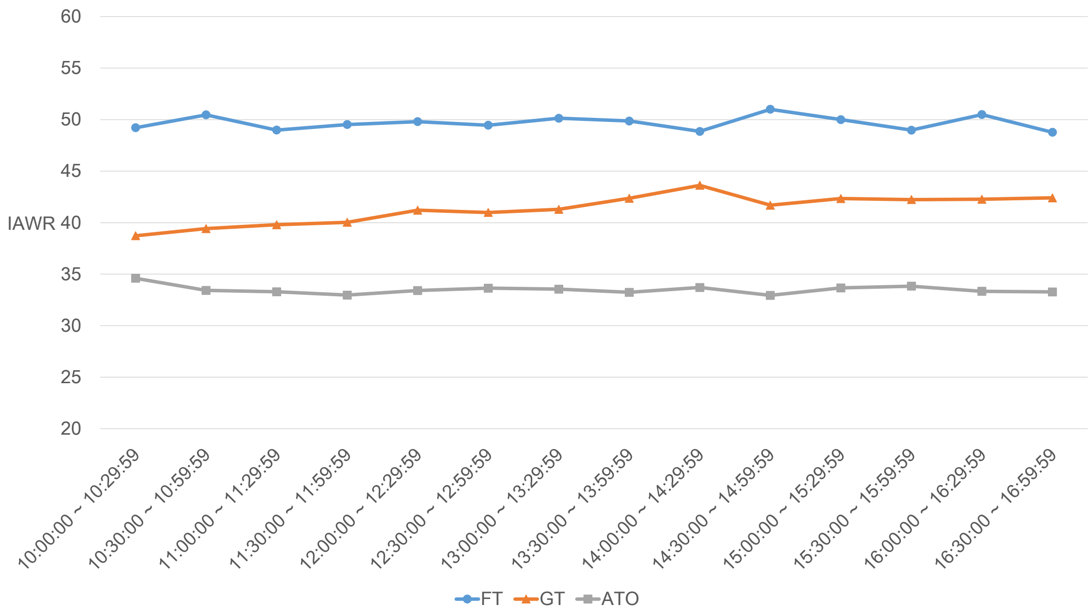

Figure 5.5: IAWR of off-peak hours

Table 5.5: IAWR comparison of off-peak hours

Time	Method	IAWR	Reduce (%)
Off-peak Hours (AM 10:00 ∼ PM 5:00)	FT	49.75	0
	GT	42.10	15
	ATO	33.53	33

All Day

The IAWR of off-peak hours are as shown in Figure 5.6, and the IAWR comparison of off-peak hours are as shown in Table 5.6. For the full day, both GT and ATO always incur reductions in IAWR especially during peak hours. On average, GT reduces the IAWR by 17% and ATO reduces the IAWR by 34%.

Further, ATO always incurs higher reductions in IAWR than GT. This means our proposed method is more efficient in solving traffic congestion.

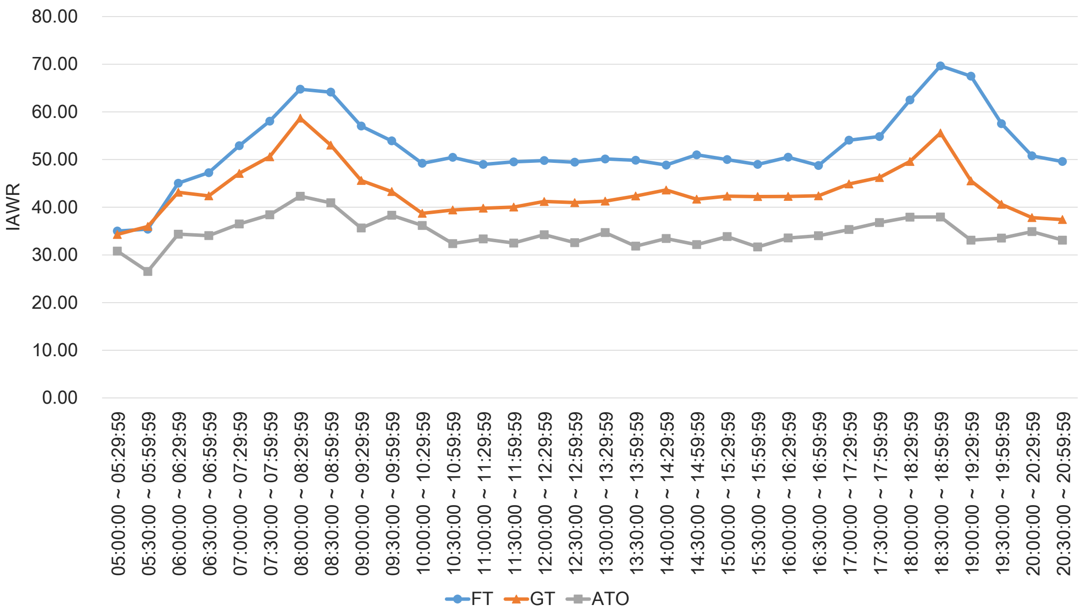

Figure 5.6: IAWR of all day

Table 5.6: IAWR comparison of all day

Time	Method	IAWR	Reduce (%)
All day (AM 6:00 ∼ PM 9:00)	FT	52.36	0
	GT	43.44	17
	ATO	34.59	34

5.3.2 Optimization Limit

For the experiment on finding the optimization limit of ATO, we used 5 different sets of traffic volume data. One set is the all day case shown in Table 5.1. The other four sets of data were generated by incrementing the values of traffic volume in Table 5.1 by 4, 8, 12, and 16, that is, the second set had all traffic volumes in Table 5.1 increased by 4, the third set by 8, the fourth set by 12, and the fifth set by 16, the most congested traffic pattern.

The IAWR of FT and ATO are as shown in Table 5.7. In cases 1 and 2, ATO reduces the IAWR by about 34%, however, with the all traffic volume values increased, the reduction in IAWR is smaller. In cases 4 and 5, ATO only reduces the IAWR by 11%, it means in extremely high traffic volume, the reduction in IAWR will be slight.

Table 5.7: IAWR of different all day case

Case	FT	ATO	Reduce (%)
1 - All day	52.36	34.59	34
2 - All day + 4	72.61	46.98	35
3 - All day + 8	85.61	68.89	20
4 - All day + 12	89.84	79.54	11
5 - All day + 16	91.60	81.74	11

5.4 Comparison of Optimization Times

In this experiment, we compare the optimization times with different configurations of our proposed adaptive adjustment of optimization to show the improve in computational efficiency and discuss the reduction of optimization time in real-world situations. The methods for comparison as shown in Table 5.8, the four methods are described below. GA method is only optimization without threshold adjustment and interval adjustment that is like traditional GA [9][10]. GA DTFI method is optimization with only threshold adjustment and fixed optimization interval. GA FTDI method is optimization with only interval adjustment and fixed optimization threshold. ATO method is optimization with both threshold adjustment and interval adjustment. The four methods are use same GA-based signal timing optimization which our proposed, only different in threshold adjustment method and interval adjustment.

Table 5.8: Methods for comparison of optimization times

Method		Threshold	
		Fixed	Dynamic
Interval	Fixed	GA	GA DTFI
	Dynamic	GA FTDI	ATO

Before this experiment, we must determine the value of fixed threshold and the value of fixed interval. The value of fixed interval we can directly use the basic optimization interval of intersection parameters, so we just need to determine the value of fixed threshold. Thus, we perform another experiment to find the value of fixed threshold which with better result. Since the average IAWR of all day is 52.36, we select three values 45,50, and 55 of fixed threshold for the experiment of fixed threshold.

The experimental result of different fixed threshold is as shown in Table 5.9. As the result, we select 45 as the value of fixed threshold in following experiment.

Table 5.9: Comparison of fixed threshold

Time	Original IAWR	Threshold	Optimization Result
Peak Hours (AM 6:00 $\sim$ AM 10:00)	55.4	45	37.63
		50	39.20
		55	42.08
Off-peak Hours (AM 10:00 $\sim$ PM 5:00)	49.68	45	33.24
		50	33.72
		55	35.06

5.4.1 Experimental Results

In this experiment, we discuss the experimental results of morning peak hours, off-peak hours, and all day separately.

Peak Hours

Due to the results of evening peak hours is similar to results of morning peak hours, we only discuss the results of morning peak hours. The IAWR and optimization times of morning peak hours are as shown in Figure 5.7. The IAWR of GA, GA DTFI, and ATO are approximate, but the optimization times of ATO is significantly less than the two methods especially in comparison with GA, ATO reduces optimization times by 51%.

In addition, although the optimization times of GA FTDI is less than GA and GA DTFI, but the IAWR is significantly higher than that optimized by the other two methods. The reason is in the fixed threshold method, it is hard to detect the increase in IAWR, thus causing the optimization interval continuously to increase. Thus, when the IAWR is finally greater than the threshold and optimization performed, a high IAWR continued already for some time.

Off-peak Hours

The IAWR and optimization times of off-peak hours are as shown in Figure 5.8. For off-peak hours, the IAWR of GA and ATO are approximate, but the optimization times incurred by is 10% more than that by GA. That means the adaptive adjustment of optimization has lower benefits during off-peak hours.

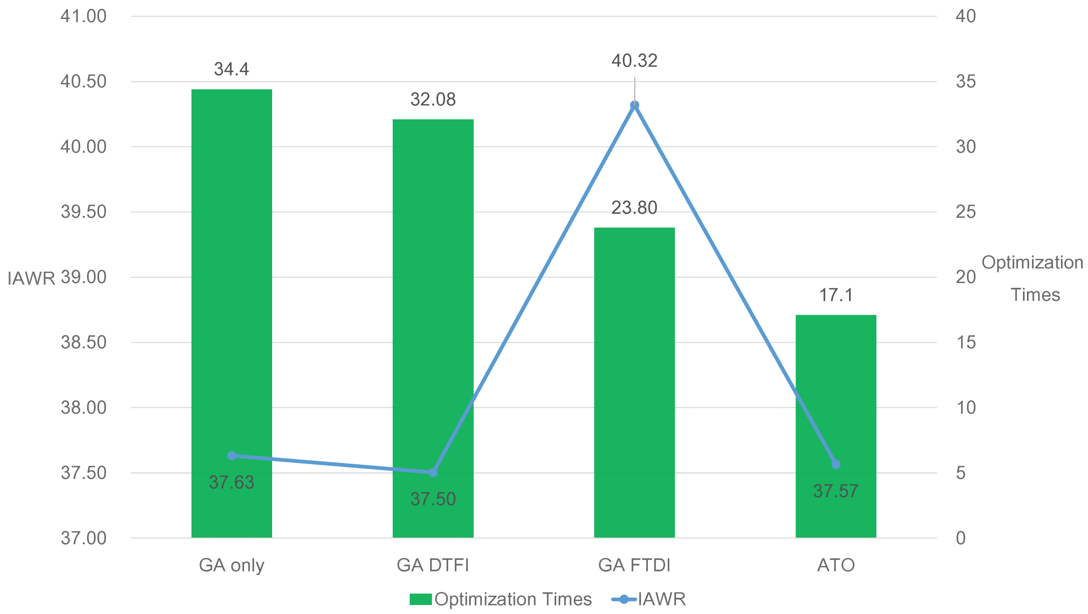

Figure 5.7: IAWR and optimization times of morning peak hours

In addition, the optimization times of GA DTFI is significantly more than other methods. The reason is the dynamic threshold adjustment sets the threshold gradually closer to IAWR but the optimization interval is fixed and short. Therefore, the threshold rapidly become closer to the IAWR, thus causing optimization to be easily triggered in spite of the increment of IAWR due to slight changes in traffic volumes.

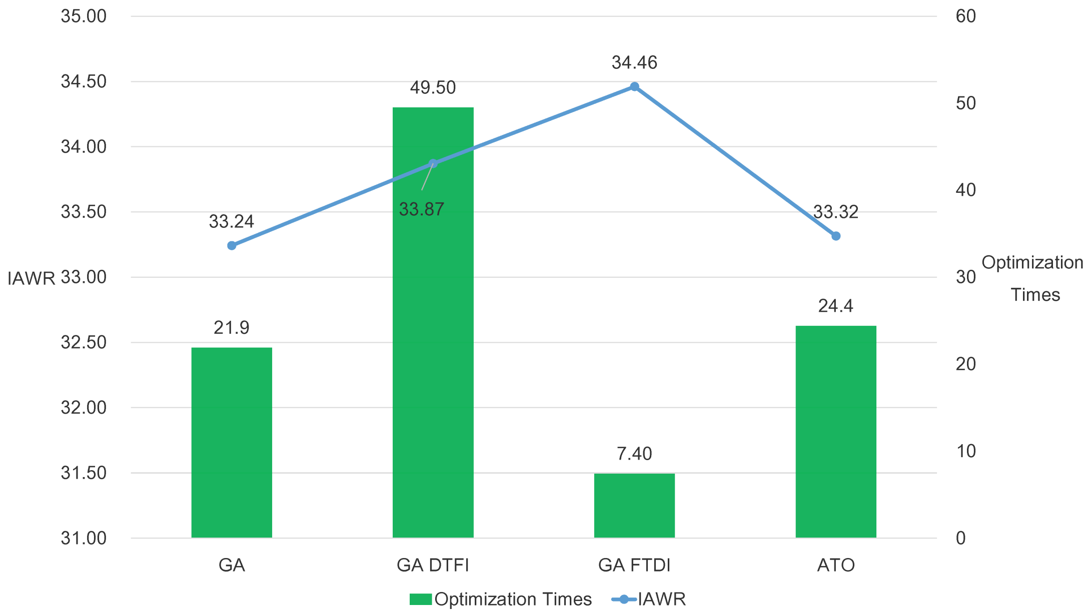

Figure 5.8: IAWR and optimization times of off-peak hours

All Day

The IAWR and optimization times of all day are as shown in Figure 5.8. For the all day, the IAWR of GA, GA DTFI, and ATO are approximate and ATO has the least optimization times compared with GA, ATO reduces optimization times by 21%.

In summary, the adaptive adjustment of optimization improves the computational efficiency in signal timing optimization of all day.

5.4.2 Reduction in Optimization Times in Real-World Situations

Based on the above experiments, we have demonstrated that the adaptive adjustment of optimization can effectively reduce the number of optimization times throughout the full day. However, the examples considered only few road inter-

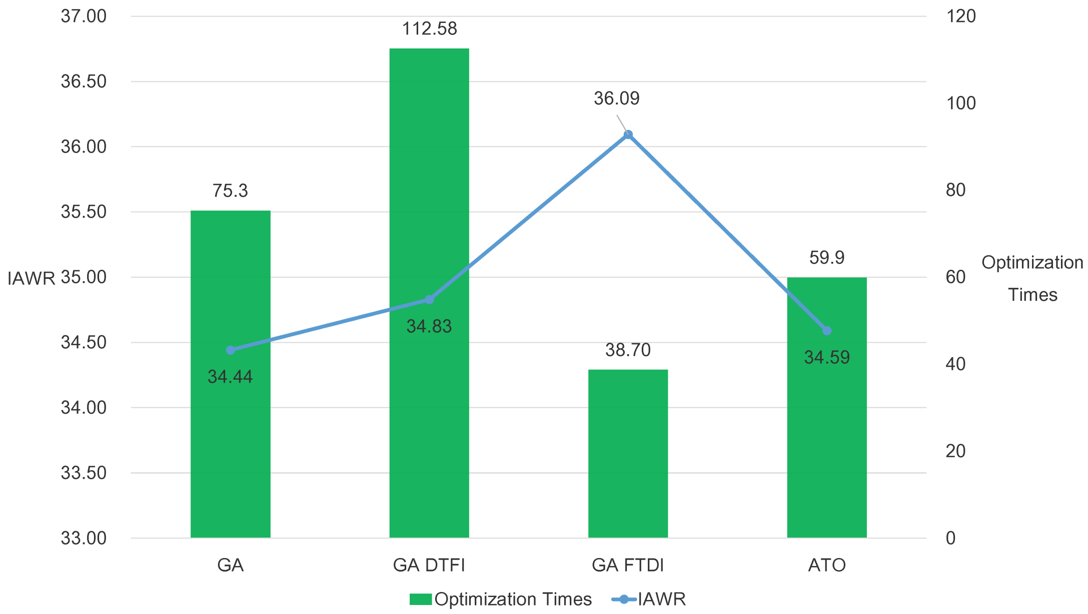

Figure 5.9: IAWR and optimization times of all day

sections.

In this section, we consider and discuss real-world situations. We take the example of the Taipei City, where the total number of intersections is 2,428 [23]. The maximum number of optimization times is 2,428 that may be performed during peak hours. Further, we consider the traffic optimization to be performed in an embedded system environment, rather than on a PC. Thus, we measure the execution time of a single time of optimization on the PC and also estimate the execution time in the embedded system environment based on the Intel Galileo board [24]. The execution times are shown in Table 5.10.

With the increase in phases, the number of generations and the population size both need to be increased to obtain better results. Therefore, we used 75 generations and a population size of 75 for 3-phase road intersections and 100 generations and a population size of 100 for 4-phase road intersections. In the peak hours, the optimization interval is usually set to the minimum (i.e., 5 cycles). The time of optimization intervals are as shown in Table 5.11.

Finally, we use the execution time to calculate the optimization time for programs running on the Intel Galileo platform during peak hours and compare the execution time of the GA method (no adaptive adjustment of optimization) and the ATO method. The comparison results are shown in Table 5.12, where the optimization time of the GA method is always larger than the time of optimization interval, while the optimization time of the ATO method is always within the time of optimization interval. It means in real-world situations, the GA method need more a more compute-intensive environment to ensure the current optimization can be completed before the next optimization. However, the ATO method need lesser amount of computation devices to achieve the same goal.

Table 5.10: Execution time of PC and Galileo

Case		PC CPU = 3.6 GHz	Galileo CPU = 400 MHz
GA Parameters	Phases	Measured	Estimated
Generation = 50 Population Size = 50	2	30 ms	270 ms
	3	35 ms	315 ms
	4	41 ms	369 ms
Generation = 75 Population Size = 75	2	65 ms	585 ms
	3	75 ms	702 ms
	4	92 ms	828 ms
Generation = 100 Population Size = 100	2	117 ms	1,053 ms
	3	140 ms	1,260 ms
	4	167 ms	1,503 ms

Table 5.11: Time of optimization interval

Phases	One cycle	Minimum optimization interval (5 cycles)
2	60 ∼ 180 sec	300 ∼ 900 sec
3	90 ∼ 270 sec	450 ∼ 1,350 sec
4	120 ∼ 360 sec	600 ∼ 1,800 sec

Table 5.12: Optimization time of Galileo

Maximum number of optimization		2,428
Reduction in peak hours		51 %
Optimization time		
Phases	Origin	ATO
2	656 sec	322 sec
3	1,705 sec	836 sec
4	3,650 sec	1,789 sec

Chapter 6

Conclusions

In this Thesis, we proposed an architecture for Cyber-Physical Traffic Control Systems (CPTCS) with Adaptive Timing Optimization (ATO), including GA-based signal timing optimization and adaptive adjustment of optimization. We adapt the GA-based signal timing optimization to reduce the number of waiting vehicles at intersections. Further, we use adaptive adjustment of optimization to reduce the number of times optimization is performed, while maintaining the same optimization effect, thus improves greatly the computational efficiency of traffic signal control.

Comparing optimization results shows that our proposed ATO can reduce IAWR by 34% in the all day case and the maximum reduction in IAWR is 39% during the evening peak hours. Compared with the Game theory-based method, ATO incurs a reduction in IAWR which is almost double for both the peak hours and the off-peak hours.

Comparing optimization times shows that compared with signal timing optimization only, signal timing optimization with adaptive adjustment of optimization result in 51% reduction in optimization times during peak hours and a 10% increment in optimization times during off-peak hours. Overall, the adaptive adjustment of optimization can reduce the optimization times by 21% for the full day.

Bibliography

[1] Yi Tian, D. Zhao, and J. Yi. A Fuzzy Logic Controller with Adaptive Dynamic Programming Optimization for Traffic Signals. In *Proceedings of the Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, volume 3, pages 191–195. IEEE, October 2008.

[2] Y. Bi, D. Srinivasan, X. Lu, and Z. Sun. Single intersection signal control based on Type-2 fuzzy logic. In *Proceedings of the Symposium on Computational Intelligence in Vehicles and Transportation Systems*, pages 25–31. IEEE, April 2013.

[3] D. Zhao, Y. Dai, and Z. Zhang. Computational intelligence in urban traffic signal control: A survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 42(4):485–494, July 2012.

[4] M. BW De Oliveira and A. De A. Neto. Optimization of traffic lights timing based on multiple neural networks. In *Proceedings of the 25th International Conference on Tools with Artificial Intelligence*, pages 825–832. IEEE, November 2013.

[5] Z. Kai, T. J. Gong, and J. Zhang. Real-time traffic signal control with dynamic evolutionary computation. In *Proceedings of the 3rd International Conference on Advanced Applied Informatics*, pages 493–498. IEEE, August 2014.

[6] R.K. Abushehab, B.K. Abdalhaq, and B. Sartawi. Genetic vs. particle swarm optimization techniques for traffic light signals timing. In *Proceedings of the 6th International Conference on Computer Science and Information Technology*, pages 27–35. IEEE, March 2014.

[7] G. F. List and M. Cetin. Modeling traffic signal control using Petri nets. *IEEE Transactions on Intelligent Transportation Systems*, 5(3):177–187, September 2004.

[8] M. Khanjary. Using game theory to optimize traffic light of an intersection. In *Proceedings of the 14th International Symposium on Computational Intelligence and Informatics*, pages 249–253. IEEE, November 2013.

[9] H.-Y. Tung, W.-C. Ma, and T.-L. Yu. Novel traffic signal timing adjustment strategy based on Genetic Algorithm. In *Proceedings of the Congress on Evolutionary Computation*, pages 2353–2360. IEEE, July 2014.

[10] G. Tong, F. Cui, and C. Fan. Genetic Algorithm and Its Application in the Realtime Traffic Signal Optimization Control. In *Proceedings of the 6th International Conference on ITS Telecommunications*, pages 86–89. IEEE, 2006.

[11] J. Shi, J. Wan, H. Yan, and H. Suo. A survey of cyber-physical systems. In *Proceedings of the International Conference on Wireless Communications and Signal Processing*, pages 1–6. IEEE, November 2011.

[12] R. H. Rawung and A. G. Putrada. Cyber physical system: Paper survey. In *Proceedings of the International Conference on ICT For Smart Society*, pages 273–278. IEEE, September 2014.

[13] Y. Pi, Y. Zhang, X. Wang, and H. Huang. A cyber-physical system framework for smart grid wireless communications. In *Proceedings of the International Conference on ICT Convergence*, pages 179–184. IEEE, October 2013.

[14] D. Arney, M. Pajic, J. M. Goldman, I. Lee, R. Mangharam, and O. Sokolsky. Toward patient safety in closed-loop medical device systems. In *Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems*, pages 139–148. ACM, April 2010.

[15] S. El-Tawab, S. Olariu, and M. Almalag. Friend: A cyber-physical system for traffic flow related information aggregation and dissemination. In *Proceedings of the International Symposium on World of Wireless, Mobile and Multimedia Networks*, pages 1–6. IEEE, June 2012.

[16] F. V. Webster. Traffic signal settings. Technical report, London:H.M.S.O, 1958.

[17] A. J. Miller. Settings for fixed-cycle traffic signals. *OR*, 14(4):373–386, December 1963.

[18] P. Mirchandani and F.-Y. Wang. RHODES to intelligent transportation systems. *Intelligent Systems, IEEE*, 20(1):10–15, January 2005.

[19] L. Ponlathep. A simple adaptive signal control algorithm for isolated intersections using time-space diagrams. In *Proceedings of the 13th Conference on Intelligent Transportation Systems*, pages 273–278. IEEE, September 2010.

[20] M. Treiber, A. Hennecke, and D. Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805, 2000.

[21] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker. Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138, December 2012.

[22] Smart Traffic Simulator. `https://github.com/bluewinter0124/SmartTrafficSimulator.git`, 2015.

[23] Transportation Facilities in Taipei City. `http://www.bote.taipei.gov.tw/GIPDSD/xslGip/xslExport/117031/logodownload/104.06traffic.pdf`, 2015.

[24] Intel Galileo Spec. `https://www.arduino.cc/en/ArduinoCertified/IntelGalileo`, 2015.