RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université de Sétif 1 - Ferhat ABBAS

Faculté des Sciences

Département d'Informatique

Synchronisation des feux de circulation dans les villes intelligentes

Thèse

Présenté par

Rafik ZERROUG

Pour L'obtention du diplôme de

Doctorat 3éme cycle LMD en Informatique

Option : Sureté de fonctionnement des réseaux dynamiques

19 Avril 2025

ALGERIENNE REPUBLICAIN DEMOCRATIC AND POPULAIRE

Ministry of High Education and Scientific Research

Setif 1 University - Ferhat ABBAS

Sciences Faculty

Computer Science Department

Synchronization of traffic lights in smart cities

Thesis

Presented by

Rafik ZERROUG

As a Requirement to aim for the degree of

Doctorate 3rd cycle LMD in Computer Science

Option: Reliable operation of dynamic networks

Sustained before a jury composed of:

Houssem	MANSOURI	Prof. University of Ferhat Abbas Sétif 1	President
Zibouda	ALIOUAT	Prof. University of Ferhat Abbas Sétif 1	Reporter
Mohamed	BENMOHAMMED	Prof. University of Constantine 2	Examiner
Hamza	REFFAD	MCA. University of Ferhat Abbas Sétif 1	Examiner
Samir	FENANIR	MCA. University of Ferhat Abbas Sétif 1	Examiner
Makhlouf	ALIOUAT	Prof. University of Ferhat Abbas Sétif 1	Invited
Adel	ALTI	MCA. University of Ferhat Abbas Sétif 1	Invited

April 19, 2025

بسم الله الرحمن الرحيم

Acknowledgements

First of all, I thank the great ALLAH our Creator.

A thesis is a subject, and a support team. I would like to express my respects to my supervisors **Pr. Makhlouf and Pr. Zibouda ALIOUAT** for having followed and guided me during the development of this project, always present, it is with great pleasure that I have worked with them.

I would like thank all persons who have contributed in any way to the completion of this work, especially **Dr Adel ALTI**.

I would also like to thank each member of the jury for honoring me and agreeing to judge my work.

My thanks also to all my teachers from primary school to university, who have participated in their noble task of teaching us and enriching our knowledge.

My thanks go to my little family, my wife and children, who have always supported me on this path and to whom I have not always given all the time it deserved, with a special mention to my mother and father, my brother and sisters.

Thanks again to everyone.

Rafik.

Abstract

Nowadays, STLSs (Smart Traffic Light Systems) are widely adopted by smart cities to control traffic lights. They often rely on dedicated equipment like camera and sensors to collect traffic data. However, STLSs in urban areas are not flexible enough to efficiently manage traffic tricky problems such congestions, emergencies, etc. Therefore, the increasing need of automatic synchronized STLS system of various traffic controllers at different intersections which performed with reduced occupancy and waiting time benefits becomes a major concern. This is crucial for meeting the optimized traffic parameters of multiple intersections and driver's needs. Our objective is to develop a new system, called ADSTLS (Adaptive and Dynamic Smart Traffic Light System) to address traffic management at an intersection, solving the challenging problem of traffic congestion while prioritizing emergency vehicles. Therefore, we have proposed a system with new hybrid traffic flow model that combines a cycle model and a phase model for optimizing and efficiently managing traffic light planning, along with a decision-making approach focused on reducing congestion and average vehicle waiting time. By collecting traffic data, the system automatically extracts useful traffic information using computer vision and computing standard traffic metrics. Moreover, we proposed two-traffic modes for regular and emergency vehicles to achieve an optimal decision-making process. The first mode, the dynamic mode, select the best phase using the Weight Chicken Swarm Optimization (WCSO) algorithm to ensure an optimal vehicle waiting time and queue occupancy at the city’s intersection. The second mode, the adaptive mode, determines the priority direction based on the distance of emergency vehicles and their priority levels. We have also proposed extending ADSTLS system using the multi-agent paradigm to improve the system’s performance in terms of execution time. To demonstrate our approach, we have presented a simulator applied to a real case study of EL-Hidhab Setif city intersection. The experimental results showed a decrease in the average vehicle waiting time (31 s) and queue occupation rate (33.82%) across all simulated traffic scenarios. Furthermore, compared to other car types, emergency vehicles usually had much shorter wait times.

Keywords: Smart cities, Traffic Controllers, Congestion, ADSTLS, Optimization, WCSO, Priority, Chronological Coordination.

List of Content

List of Content i
List of Figures vi
List of Tables viii
List of Algorithms ix
List of Abbreviations x
General Introduction 1
Part 1: Domain Background and Related Approaches 9
Chapter 1: Artificial Intelligence, Swarm Optimization and Computer Vision in Smart Traffic Light Systems: Background and Main Approaches 10
1. Introduction 11
2. Smart Traffic Light System in Smart Cities 12
    2.1 Evolution of Traffic Light Control System: From Manual to Smart Control 12
    2.2 Architecture of Smart Traffic Light System 12
    2.3 Components of Smart Traffic Light System 14
        2.3.1 Roads and Intersection 14
        2.3.2 Traffic Lights 15
        2.3.3 Cycle and Phase 15
    2.4 Sensors and Detectors 15
        2.4.1 Sensors Technology 16
        2.4.2 Motivation for Using Wireless Sensors 16
    2.5 Smart Traffic Light Management Process 17
        2.5.1 Traffic Data Collection 17
        2.5.2 Traffic Data Processing 17
        2.5.3 Decision-Making 18
    2.6 Emergency Vehicles Signal Management Scenarios 18
3. Computer Vision in Smart Traffic Light Systems 18
    3.1 Introduction to Computer Vision 18
    3.2 Traffic Video Images Processing Methodology 19
        3.2.1 Video Images Traffic Acquisition and Pre-Processing 19
        3.2.2 Object Detection and Traffic Feature Extraction 19
        3.2.3 Traffic Flow Metrics Calculation 20

4. Swarm Optimization Metaheuristics in Smart Traffic Light Systems 21
4.1 Concepts of Swarm Optimization 21
4.2 Standard Swarm Optimization Algorithms and Applications 24
4.2.1 Particle Swarm Optimization (PSO) Algorithm 24
4.2.2 Chicken Swarm Optimization (CSO) Algorithm 24
4.2.3 Ant Colony Optimization (ACO) Algorithm 25
4.2.4 Artificial Bee Colony (ABC) Algorithm 26
4.2.5 Firefly Algorithm (FA) 26
4.2.6 Bat Algorithm (BA) 27
4.2.7 Cuckoo Search (CS) Algorithm 28
4.2.8 Grey Wolf Optimizer (GWO) Algorithm 28
4.2.9 Whale Optimization Algorithm (WOA) 29
4.2.10 Genetic Swarm Optimization (GSO) Algorithm 30
4.2.11 Swarm Optimization Applications 31
4.2.12 Main Advantage and Disadvantage of Swarm Optimization Algorithms 32
4.3 Benefits of Swarm Optimization in Smart Traffic Light Systems 33
5. Artificial Intelligence in Smart Traffic Light System 34
5.1 Introduction to Artificial Intelligence (AI) 34
5.2 The Role of AI in Smart Traffic Light Systems 34
5.3 Types of AI-Powered Traffic Control Management Systems 35
5.3.1 AI-Powered Centralized Traffic Control Management System 35
5.3.2 AI-Powered Decentralized Traffic Control Management System 36
5.3.3 AI-Powered Distributed Traffic Control Management System 37
5.3.4 Centralized, Decentralized and Distributed Traffic Control Management Systems . 37
5.4 Coordination and Cooperation of Traffic Control Mechanisms 38
5.4.1 Coordination of Traffic Control Mechanisms 38
5.4.2 Cooperation of Traffic Control Mechanisms 38
5.4.3 Coordinative vs. Cooperative Traffic Control Mechanisms 39
5.5 Existing AI Approaches in Smart Traffic Light System 39
5.5.1 Multi-Agent Based Traffic Flow System 39
5.5.2 Machine Learning Approaches and Optimization Algorithms 39
5.5.3 Deep Learning Approaches 40
5.5.4 Hybrid Approaches 40

6. Combine Swarm Optimization with AI in STLS 40
6.1 Principles of Combining of Swarm Optimization and AI...................................41
6.1.1 Integration of Sensors and Cameras for Data Collection.............................41
6.1.2 Integration of Edge Computing for Data Processing...................................41
6.1.3 Swarm Optimization for Signal Timing 41
6.1.4 Machine Learning and Deep Learning for Traffic Flow Prediction.............42
6.1.5 Swarm Algorithms for Traffic Flow Optimization......................................42
6.1.6 Intelligent Agents for Dynamic Routing and Adaptation 42
6.1.7 Emergency and Incident Management..42
6.2 Benefits of Combining of Swarm Optimization with AI....................................43
7. Conclusion...43
Chapter 2: A Survey on Smart Traffic Light Systems: Models and Approaches 44
1. Introduction 45
2. Main Works in the STLS Area 46
3. STLS Approaches for a Single Intersection Model...50
3.1 Heuristics based Approaches..50
3.2 Sensor-based Approaches 56
3.3 Hybrid Approaches..59
3.4 Synthesis and Comparison 62
4. STLS Approaches for a Set of Intersections 63
4.1 Heuristics based Approaches..63
4.2 Sensor-based Approaches 70
4.3 Hybrid Approaches..71
4.4 Synthesis and Comparisons..73
5. Global Smart Traffic Light System and Internet of Things 73
6. STLS Simulation Methods...75
7. Conclusion...75
Part 2: Contributions...76
Chapter 3: ADSTLS: Adaptive and Dynamic Smart Traffic Light System for Single and Multiple Intersections...77
1. Introduction 78
2. System Architecture...78
2.1 Single Intersection 79
2.1.1 Main Video Processing Unit (MVPU)...79

2.1.2 Main Traffic Signal Control Unit (MTSCU) 79
2.2 Multiple Intersections 80
3. Assumptions 81
4. System Models and Traffic Metrics 81
4.1 Intersection Model 81
4.2 Cycle and Phase Models 82
4.3 ADSTLS's Traffic Metrics 83
4.3.1 Vehicle Metrics 83
4.3.2 Queue Metrics 84
4.3.3 Intersection Metrics 85
4.3.4 Other Metrics 86
5. ADSTLS Control System for Single Intersection 86
5.1 Fault Tolerance 87
5.2 Traffic Data Collection and Preprocessing 88
5.3 Traffic Data Privacy and Security 89
5.3.1 The Proposed Protocol 89
5.3.2 GDPR Policies and Regulations 90
5.4 Computation of the System Metrics 91
5.5 Decision Making in ADSTLS Modes 92
5.5.1 Phase Selection and Weight Optimization 92
5.5.2 WCSO based Dynamic and Intelligent Mode 98
5.5.3 WCSO based Adaptive Emergency Mode 104
5.6 Traffic Light Control Execution 107
6. Chronological Coordination of ADSTLS at Multiple Intersections 107
6.1 WCSO-Based Agent Intersection Controller 107
6.2 Chronological Coordination of WCSO Agent Controllers 108
6.2.1 Coordination Process in Congestion Mode 108
6.2.2 Coordination Process in Weather Disasters 111
7. Conclusion 112
Chapter 4: Smart Traffic Flow Simulation System and Evaluation 113
1. Introduction 114
2. Simulator Presentation 114
3. Simulator Demonstration 116

3.1 Presenting the Case Study 116
3.2 Case Study Simulation 117
3.3 Implementation and Deployment Challenge 118
4. ADSTLS Performance Evaluation for a Single Intersection 119
4.1 Evaluation Metrics 120
4.2 Optimization Performance Comparison 123
4.3 Performance Comparison of ADSTLS Models in Dynamic Mode 126
4.4 Performance Comparison of ADSTLS Models in Adaptive Mode 129
4.5 Comparison of Traffic Signal Plan and Performance Optimization 131
4.5.1 Comparison of Traffic Signal Plans 131
4.5.2 Comparison of Performance Optimization with Relative Models 133
4.6 Results Analysis and Key Observations 135
4.6.1 Optimization Performance Comparison of Different Swarm Intelligence Algorithms 135
4.6.2 Fixed-weights Approach vs Weight-based Chicken Swarm Optimization Approach 136
4.6.3 Dynamic intelligent vs fixed-time traffic control models 136
4.6.4 DIM Models Performance Comparison: WCSO Phase vs WCSO Cycle vs WCSO Hybrid 136
4.6.5 Comparison of Average Waiting Time of Emergency Vehicles vs Regular Vehicles in Adaptive Phase Model 137
5. CCADSTLS Performance Evaluation for a Set of Intersections 137
5.1 Traffic Congestion Scenario 139
5.2 Weather Disaster Scenario 139
5.3 Performance Evaluation Metrics 140
5.4 Simulation Results and Discussion 141
6. Conclusion 142
General Conclusion 144
References 147

List of Figures

Figure 1 Map of the intersection for EL-Hidhab, Setif city case study. 5
Figure 1.1 Smart traffic light system research domains..11
Figure 1.2 Smart traffic light system architecture. 13
Figure 1.3 Model of a four-way signaled intersection [35]...14
Figure 1.4 Two-cycle models with four phases..15
Figure 1.5 Intrusive and non-intrusive sensor models [38]...16
Figure 1.6 Traffic image processing methodology. 19
Figure 1.7 Computer vision for vehicles detection [44]. 21
Figure 1.8 Concepts of swarm optimization. 23
Figure 1.9 Evolution of different traffic control management systems. 35
Figure 2.1 STLS models and approaches..46
Figure 2.2 TSPN sub-model for movement (ij) [16]...53
Figure 2.3 Image processing process [23]..61
Figure 2.4 Solution coding, phase duration and traffic light offsets for two intersections [11]..64
Figure 2.5 DQN architecture [17]...67
Figure 2.6 Optimization model [103]. 69
Figure 2.7 Traffic light control new proposal [9]. 71
Figure 2.8 Example of a solution for eight junctions [9]. 72
Figure 2.9 Example of domains influenced by IoT. 74
Figure 2.10 IoV sustained by IoT. 74
Figure 3.1 General architecture of ADSTLS. 79
Figure 3.2 The multi-agent architecture for traffic-light control. 80
Figure 3.3 Intersection model. 82
Figure 3.4 Cycle and phase models. 83
Figure 3.5 Functional model of ADSTLS. 87
Figure 3.6 ADSTLS fault tolerance process...87
Figure 3.7 Traffic data collection and preprocessing process. 88
Figure 3.8 Flowchart of WCSO-based phase selection. 97
Figure 3.9 ADSTLS decision making in dynamic mode process..98
Figure 3.10 Signal planning in WCSO dynamic intelligent phase model. 100

Figure 3.11 Signal planning in WCSO dynamic intelligent cycle model. 101
Figure 3.12 Signal planning in WCSO dynamic intelligent hybrid model. 102
Figure 3.13 ADSTLS decision making in adaptive mode process. 104
Figure 3.14 Modules of WCSO based agent intersection controller. 108
Figure 3.15 Chronological coordination process in congestion mode. 110
Figure 3.16 Chronological coordination process in weather disasters mode. 111
Figure 4.1 Main user interface of STLS simulator. 115
Figure 4.2 Description of cycle, green and orange time. 117
Figure 4.3 Model configuration parameters and simulation results. 117
Figure 4.4 Comparison of model's performance under different vehicles started time in dynamic mode. 129
Figure 4.5 Comparison of traffic signal plans. 132
Figure 4.6 Comparison of performance optimization. 134
Figure 4.7 Comparison of utilities between WCSO dynamic models. 137
Figure 4.8 CCADSTLS based 4×4 traffic network with sixteen intersections. 138
Figure 4.9 Number of congested paths detected and resolved using CCADSTLS. 141

List of Tables

Table 1.1 Swarm optimization algorithm applications 32
Table 1.2 Majors' advantages and disadvantages of swarm optimization algorithms 32
Table 1.3 Centralized, decentralized and distributed traffic control management systems......38
Table 1.4 Coordinative vs. cooperative traffic control management.................................39
Table 1.5 Comparison of AI approaches in STLS. 40
Table 2.1 (a) Component definition of the TSPN Sub-model in Figure 2.2. (b) Activation conditions for the decision events in Figure 2.2. [16]...54
Table 2.2 Comparison of different STLS approaches for a single intersection. 62
Table 2.3 Comparative study of STLS approaches for an intersection set. 73
Table 4.1 Economic cost-benefit analysis for ADSTLS system implementation..............119
Table 4.2 Simulation parameters in optimization performance. 124
Table 4.3 Comparison of best fitness values and convergence speed. 124
Table 4.4 Accuracy comparison of fixed weight vs optimized weight approaches using WCSO. 125
Table 4.5 Simulation parameters in dynamic mode. 126
Table 4.6 Comparison of model's utility under different vehicles started time in dynamic mode. 128
Table 4.7 Simulation parameters in adaptive mode. 130
Table 4.8 Simulation of different start time intervals for emergency vehicles. 130
Table 4.9 Comparison of CCADSTLS utility under different vehicles arrival time and coordinate rate value in high vehicles density. 142

List of Algorithms

Algorithm 1.1 Particle swarm optimization (PSO).....24
Algorithm 1.2 Chicken swarm optimization (CSO).....25
Algorithm 1.3 Ant colony optimization (ACO).....25
Algorithm 1.4 Artificial bee colony (ABC).....26
Algorithm 1.5 Firefly algorithm (FA).....27
Algorithm 1.6 Bat algorithm (BA).....27
Algorithm 1.7 Cuckoo search (CS).....28
Algorithm 1.8 Grey wolf optimizer (GWO).....29
Algorithm 1.9 Whale optimization algorithm (WOA).....30
Algorithm 1.10 Genetic swarm optimization (GSO).....31
Algorithm 3.1 Computation of the system metrics.....91
Algorithm 3.2 Decision-making in dynamic intelligent mode based WCSO.....103
Algorithm 3.3 Decision-making in adaptive emergency mode based WCSO.....106

List of Abbreviations

Acc	Accuracy
ADSTLS	Adaptive and Dynamic Smart Traffic Light System
AEM	Adaptive Emergency Mode
ANPR	Automatic Number Plate Recognition
Aor	Average Occupancy Rate
Aqwt	Average Queue Waiting Time
Awt	Average Waiting Time
Bf	Best Fitness
CCADSTLS	Chronological Coordination of ADSTLS
CP	Congested Path
CS	Congested Scenario
Cs	Convergence Speed
CSO	Chicken Swarm Optimization
Cwtv	Current Waiting Time Vehicle
DIM	Dynamic Intelligent Mode
Dlv	Distance Last Vehicle
DM	Decision Making
Dv	Distance Vehicle
Es	East to South
Ew	East to West
f	Fitness
fs	Selection Function
Fu	Utility Function
GDPR	General Data Protection Regulation
HDV	Hight Density of Vehicles
Hr	Hybrid Rate
In	Input Flow
In_tv	Input Time Vehicle
IoT	Internet of Things
LDB	Local Database
LSTI	Large Start Time Interval
MDV	Medium Density of Vehicles
MSTI	Medium Start Time Interval
MTC	Main Telecommunication Component

MTSCU	Main Traffic Signal Control Unit
MVCC	Main Video Capture Component
MVPC	Main Video Processing Component
MVPU	Main Video Processing Unit
NDV	Normal Density of Vehicles
Ne	North to East
Ns	North to South
Nvp	Number of Vehicles Priority
Nv	Number of Vehicles Simulated
Occupr	Occupancy Rate
Ofr	Output Flow Rate
Out	Output Flow
Out_tv	Out Time Vehicle
P2BAC	Privacy Policy Based Access Control
Ph	Phase
Plv	Priority Level of the Vehicle
PSO	Particle Swarm Optimization
Qsize	Queue Size
Qsizemax	Queue Size Maximum
RSU	Road Side Unit
Sn	South to North
Sr	Simulation Time to Real Time Ratio
STC	Secondary Telecommunication Component
SSTI	Small Start Time Interval
STLS	Smart Traffic Light System
STSCU	Secondary Traffic Signal Control Unit
Sv	Speed of Vehicles During the Simulation
SVCC	Secondary Video Capture Component
SVPC	Secondary Video Processing Component
SVPU	Secondary Video Processing Unit
Sw	South to West
Tc	Time Current
TCM	Traffic Communication Module
TDC	Traffic Data Collector
Tg	Time Green
To	Time Orange
Ts	Time Scale

Tsti	Start Time Interval
VMS	Vehicle Message Scheduling
W	Inertia Weight
WCSO	Weight Chicken Swarm Optimization
WDS	Weather Disaster Scenario
We	West to East
Wm	Way Movement
Wn	West to North
WPSO	Weight Particle Swarm Optimization
Wq	Weight Queue
Wt	Weight Time
Wtv	Waiting Time Vehicle

General Introduction

Smart cities represent an emerging paradigm in which advanced technologies, particularly Information and Communication Technologies (ICT), are harnessed to improve the quality of life in urban areas. One of the significant challenges facing modern cities is urban traffic congestion, leading to longer journey times and increased pollution levels. Synchronizing traffic lights is crucial for managing vehicle flow and reducing delays, playing a key role in addressing these challenges. As cities become smarter, traditional traffic management systems are evolving towards more efficient intelligent traffic systems, enabling for the dynamic adjustment of traffic lights based on real-time traffic conditions. This advance empowers cities to respond proactively to shifting traffic patterns. Smart cities aim to enhance the sustainability of urban mobility by optimizing existing infrastructure, reducing both travel times and greenhouse gas emissions, as well as reducing driver stress at peak traffic times.

Traffic light synchronization is one of the critical issues in traffic management in urban environments. Well-synchronized traffic signals can improve traffic flow, reduce waiting times at intersections and minimize fuel consumption. However, coordinating these signals across an entire city presents a complex problem, particularly in cities with varying traffic patterns.

Traditional traffic lights systems operate according to fixed cycles and predefined schedules, which do not adapt to real-time traffic conditions. Although still widely used, these systems lack efficiency, especially in urban areas where traffic volumes fluctuate significantly. In traditional systems, traffic lights switch after a fixed interval, regardless of the number of vehicles waiting at a junction. This can lead to unnecessary delays for vehicles and contribute to congestion. The main limitation of traditional traffic lights is their inability to respond to real-time traffic conditions. For example, during low-traffic periods, vehicles may be forced to wait unnecessarily at red lights, wasting time and fuel. Similarly, at peak times, traditional traffic lights often fail to reduce congestion, as their fixed timetables are unable to cope with sudden traffic surges.

Traffic light synchronization aims to optimize the flow of vehicles in urban areas by minimizing unnecessary stops and reducing journey times. Traditionally, traffic light systems operated on fixed cycles, often ignoring real-time traffic variations. However, with the advent of intelligent sensors, connected vehicles and advanced communication infrastructures, it is now possible to design dynamic systems, capable of adjusting to current conditions. These systems, known as Smart Traffic Lights System (STLS), are at the heart of initiatives to improve traffic efficiency in smart cities.

STLS are designed to respond to traffic data in real-time by using sensors, cameras and communication modules to dynamically adjust signal timings according to actual traffic conditions. By optimizing traffic flow at intersections, these intelligent traffic lights can reduce vehicle idling time, shorten overall journey times, and reduce the environmental impact of vehicle emissions. STLSs are vital components in smart city infrastructure. They are connected to a centralized or distributed control system that collects data from various sources across the city. This data is processed with advanced algorithms, which then adjust signal timings at different intersections to create a more efficient traffic flow. Additionally, smart traffic lights can communicate with connected vehicles, providing drivers with real-time updates on upcoming signal changes, further optimizing traffic flow and reducing fuel consumption.

Several approaches have been developed for synchronizing traffic lights in smart cities. These approaches utilize different optimization techniques aimed at improving traffic flow while minimizing delays and reducing vehicle emissions. These techniques include Genetic Algorithms (GAs) [1], Fuzzy logic control [2], Reinforcement Learning (RL) [3], Multi-Agent Systems (MAS) [4] and Artificial Neural Networks (ANN) [5]. These techniques highlight how STLS leverages technology to improve urban mobility while addressing environmental and safety concerns. Each method offers different advantages depending on the size and complexity of the traffic network. However, despite technological advances, implementing these systems raises several challenges. The integration of real-time data from a variety of sources such as vehicle sensors, and surveillance cameras, and making rapid decisions in complex urban environments, requires robust, high-performance systems. Additionally, when synchronizing traffic lights, it is primordial to consider the conflicting needs of different road users, including motorized vehicles, cyclists, pedestrians, and public transport. Moreover, optimizing traffic lights in an urban setting poses problems of computational complexity, requiring scalable solutions to handle complexity effectively [6].

1. Problematic

Traditional urban traffic management methods based on fixed green light times often prove inefficient for effective traffic management and cannot be applied to high-traffic flow density scenarios. As a result, large smart cities have adopted STLSs to find appropriate and enduring ways to overcome the increasing demand for effective traffic management. STLS has been proposed to minimize the average waiting time of vehicles and queue length while maximizing overall traffic flow using advanced optimization methods to achieve good performance in real-world environment. These optimization methods include dynamic algorithm [7], artificial neural networks [8], gaussian mixture model [9], bee colony optimization [10], multi-objective diversity-based evolutionary genetic algorithms [11], multi-agent [12], fuzzy logic [13], reinforcement learning [14], game theory [15], and Petri nets [16]. Sometimes, a combination of two techniques such as multi-agent and deep learning [17], is also used.

Other complementary methods provide significant benefits when collecting traffic data using cameras and sensors. Few solutions have benefited from the vast data available from modern wireless sensors. In this context, some STLS systems have used either optimization approaches [14,16,18,19] or sensor-based decision [7,20-22] while others have adopted a hybrid approach combining sensors and optimization techniques [8,23-26]. For instance, Hosur *et al.* [20] proposed a system based on sensors with a defined threshold distance using Internet of Things (IoT) technology. Joo *et al.* [14] introduced a traffic signal control system at an isolated intersection to maximize traffic flow and minimize queue size with a standard deviation. Some works in [7,8,14,19,20,22-24,26] used the phase model for moving traffic instead of the cycle model, which requires high computation time. In contrast, other works [16, 18, 21, 25] used the cycle model for dense traffic situations instead of the phase model, which reduces system efficiency. In addition, current research lacks a suitable approach for determining the next phase while balancing solution quality. However, this problem can be solved within an acceptable time by utilizing a hybrid dynamic traffic lighting controller that combines cycle and phase models. The hybrid model switches from a cycle model to a phase model when the average queue occupancy rate is higher and vice versa.

Currently, most traffic management research has focused on improving vehicle waiting times and reducing problems at intersections. Promising solutions have been proposed to satisfy drivers' requirements without considering the evolution of incoming vehicles, advanced information, and communication technologies. Prioritizing certain vehicles is essential for effective traffic management. The most striking case is high traffic volume, where the movement of emergency vehicles can cause a delay in finding optimal solutions. A few studies, such as [24-26] have shown that the adaptive traffic management model achieves an optimal waiting time for emergency vehicles. The focus should also be on high-priority vehicles, such as ambulances and firemen, avoiding long waiting times to respect their emergency status. Automatic detection of emergency vehicles by traffic signals to ensure fast intervention. This objective can be achieved if the traffic signal management system receives the traffic data necessary to process it appropriately and optimally.

To reduce traffic congestion during weather fluctuations and various scenarios, the traffic light system uses STLS, which is equipped with cameras and sensors to collect real-time traffic data. This enables the system to regulate traffic flow by adjusting signal timing and providing drivers with accurate information about traffic conditions [27]. However, a significant issue with STLS is that the time interval is not adapted to dynamically changing traffic data during bad weather, or when the presence of emergency vehicles increases traffic flow. As a result, intersection controllers must target short-term goal and manage the most congested path to maximize benefits.

Although current agent-based STLS systems are reliable [28], they are still time-consuming and often require extensive message exchanges which could affect the traffic flow dynamics at intersections and negatively affect the expected traffic performance of the system. By providing decentralized decision-making through the STLS approach, our system addresses these challenges by dynamically controlling traffic flow at multiple intersections.

2. Motivating Case Study

The decision to use a large city intersection as a real-life case study for an STLS is motivated by several key factors. Firstly, these intersections are critical points where traffic jams are frequent, and implementing an intelligent system would improve traffic flow by dynamically adjusting traffic lights in real time, thereby reducing waiting times and congestion. Furthermore, reducing frequent stops and fuel consumption and greenhouse gas emissions should contribute to a cleaner and more sustainable urban environment. Additionally, the system could improve road safety by predicting risk behavior and preventing accidents through optimal management of vehicle flows. Moreover, a suitable urban intersection is an ideal testing environment for collecting valuable data to support intelligent algorithms, enabling continuous improvement of the system. Finally, our work can serve as a model for wider implementations across other areas of the city or different cities, paving the way for seamless integration into a broader smart city strategy. Therefore, we have selected a crossroads in El-Hidhab, Setif city, as a real-world traffic case study.

Setif is a town in northeastern Algeria, that is one of its largest cities. As the population of the city increases, the Setif authorities will rely on a novel extension of the urban perimeter for quality assurance. Therefore, a new city of El-Hidhab was created in the northeast of Setif Town. A city that has a simple intersection, as shown in Figure 1, is used to demonstrate the proposed approach. El-Hidhab’s intersection was considered to have heavy traffic. It includes four directions, each equipped with a traffic light.

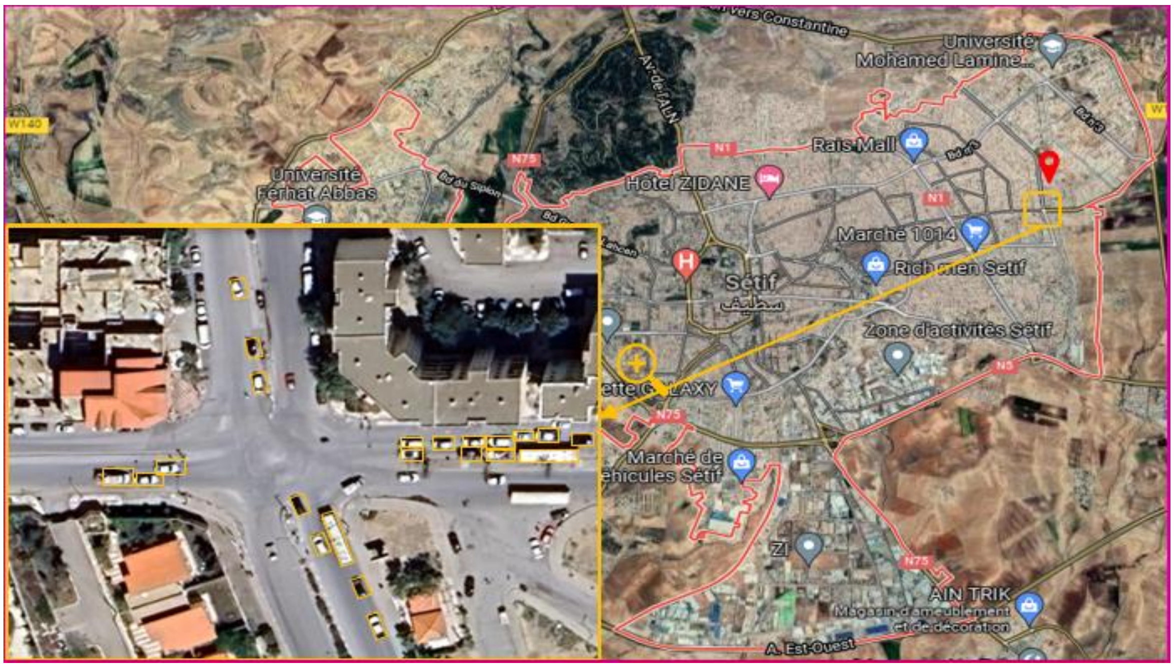


Figure 1 Map of the intersection for EL-Hidhab, Setif city case study.

3. Objectives and Contributions

The central aim of this thesis is to explore the power and immense benefits of smart sensor technology and intelligent decision-making processes to build a robust and efficient hybrid traffic signal management model. Our goal is to design and develop a new intelligent, dynamic, and adaptive traffic management system that optimizes various traffic parameters while remaining flexible to prioritize emergency vehicles. Indeed, the intersections in the city are experiencing a rise in vehicle numbers at different times, leading to traffic congestion. The system should be dynamic to adapt quickly to these evolutions. For this reason, we have adopted an intelligent traffic control system called ADSTLS (Adaptive and Dynamic Smart Traffic Light System) based on video sensors to collect and process traffic data, enabling us to calculate standard traffic metrics (e.g., average occupancy rate, output flow rate, and average waiting time of vehicles). This system aims to reduce traffic congestion and management of emergency vehicles.

The traffic controller automatically generates an intersection management plan with an optimal green time for the phases based on a combination of system metrics. The traffic controller operates in two modes, dynamic and adaptive. The dynamic mode consists of three models for traffic signal management: cycle, phase, and hybrid models. The adaptive emergency mode enables effective management of emergency vehicles with an optimal waiting time for priority information. The system recognizes any priority vehicle using sensory identification, such as intelligent cameras, and can adjust to integrate new priority types. By integrating the collective intelligence of chicken behavior, we can create a more dynamic, adaptive, and efficient traffic management system to meet the evolving needs of modern cities. To the best of our knowledge, this is the first study that combines cycle and phase models for traffic signal planning. This is expected to leverage the additional optimization of parameters at a single traffic intersection and associated weights, such as vehicle waiting time and queue occupancy rate, using a novel Weight-based Chicken Swarm Optimization (WCSO) algorithm.

An extended version of the ADSTLS system, known as CCADSTLS (Chronological Coordination of ADSTLS), aims to minimize congestion at multiple intersections. It enables traffic control agents to autonomously identify the source of congestion early, allowing for efficient and cost-effective decisions to keep traffic flow smoothly. The system prioritizes the safety and effectiveness of intersections, especially during severe weather fluctuations.

Moreover, ADSTLS is based on cycle and phase models, fault tolerance, bio-inspired optimization, and potential traffic metrics, which are more useful and efficient in case of unexpected events. The key contributions of this study are summarized as follows.

- **Fault tolerance**: Existing works do not propose a fault tolerance strategy. ADSTLS uses a heartbeat mechanism to ensure reliability without losing data or functionality.
- **Hybrid optimal WCSO-based traffic flow management**: None of the existing works combine cycle and phase models to optimize traffic flow. We use WCSO to select the best phase or cycle that optimizes traffic parameters by finding optimal weights that could be achieved by processing possible traffic data.
- **Flexibility**: Despite the regular and emergency vehicles, existing approaches have adopted only one strategy. To solve this issue, we use sensors to detect emergency vehicles and automatically switch between dynamic intelligent mode and adaptive emergency mode depending on the situation.
- **Vehicle density and scalability**: Most existing approaches lack the support to efficiently deal with high traffic density. Moreover, they are strongly bounded by normal vehicle density, such as [7,14,20,22]. We opt for a bio-inspired optimization approach to deal with low, medium, and high vehicle density.

- **Priority-based emergency vehicles**: Existing approaches do not cover all criteria relevant to the management of emergency vehicles including, the vehicle trust, priority levels and vehicle location.
- **Real case study**: We investigated the usefulness of leveraging traffic information in controlling traffic lights in a real case study of EL-Hidhab Setif city intersection, and in optimizing queue occupancy rate, average waiting time, and average output flow in adaptive and dynamic modes.
- **Coordination for multi-intersection**: Introducing CCADSTLS, a novel chronological coordination of ADSTLS for multi-intersection system to reduce congestion and improve traffic flow during disasters and weather fluctuations. Based on a multi-agent approach and WCSO algorithm, CCADSTLS determines the congested path by asynchronously transmitting traffic flow messages between neighboring intersections.

4. Organization of Thesis

This thesis is organized into four chapters. The first two chapters are devoted to the background of the domain and a literature review relevant to our research, while chapters 3 and 4 detail our contributions. An overview of each chapter is provided below:

- **Chapter One** gives an overview of smart traffic light systems, covering their evolution, architecture, and components. It also explores the role of computer vision in STLS, as well as the applications of swarm optimization and artificial intelligence within these systems. Additionally, it discusses the main types of traffic control management systems, highlighting their advantages and disadvantages, and concludes with existing AI approaches in STLS.
- **Chapter Two** presents a comprehensive survey that examines different STLS models and approaches. It reviews key works in the STLS domain, compares the different approaches for managing both a single intersection and a set of intersections, and offers a synthesis of these approaches. The chapter also gives an overview of the role of the Internet of Things (IoT) in STLS and ends with a discussion of different simulation methods used to validate these approaches.
- **Chapter Three** details our contribution, the Adaptive Dynamic Traffic Signal System (ADSTLS) for both single and multiple intersections. It presents the system's architecture, models, and standard traffic metrics. The chapter also presents the

control system for managing a single intersection and details the chronological coordination of ADSTLS across multiple intersections.

– **Chapter Four** presents the developed simulator, which is based on the weight sensor-based collective intelligence model and various implemented ADSTLS models. It includes the demonstration of the simulator using a real-world case study EL-Hidhab Intersection in Setif city. This chapter also examines the performance evaluation and experimental results for both single and multiple intersections.

This thesis concludes with a summary of the research, highlighting its contributions, and limitations, and offering suggestions, along with potential directions for future studies.

Part 1: Domain Background and Related Approaches

Chapter 1: Artificial Intelligence, Swarm Optimization and Computer Vision in Smart Traffic Light Systems: Background and Main Approaches

Content

1. Introduction 11
2. Smart Traffic Light System in Smart Cities 12
   2.1 Evolution of Traffic Light Control System: From Manual to Smart Control 12
   2.2 Architecture of Smart Traffic Light System...12
   2.3 Components of Smart Traffic Light System 14
   2.4 Sensors and Detectors...15
   2.5 Smart Traffic Light Management Process 17
   2.6 Emergency Vehicles Signal Management Scenarios...18
3. Computer Vision in Smart Traffic Light Systems 18
   3.1 Introduction to Computer Vision..18
   3.2 Traffic Video Images Processing Methodology..19
4. Swarm Optimization Metaheuristics in Smart Traffic Light Systems....................................21
   4.1 Concepts of Swarm Optimization...21
   4.2 Standard Swarm Optimization Algorithms and Applications 24
   4.3 Benefits of Swarm Optimization in Smart Traffic Light Systems..................................33
5. Artificial Intelligence in Smart Traffic Light System 34
   5.1 Introduction to Artificial Intelligence (AI)...34
   5.2 The Role of AI in Smart Traffic Light Systems 34
   5.3 Types of AI-Powered Traffic Control Management Systems..35
   5.4 Coordination and Cooperation of Traffic Control Mechanisms.....................................38
   5.5 Existing AI Approaches in Smart Traffic Light System..39
6. Combine Swarm Optimization with AI in STLS..40
   6.1 Principles of Combining of Swarm Optimization and AI..41
   6.2 Benefits of Combining of Swarm Optimization with AI 43
7. Conclusion...43

1. Introduction

The congestion of urban traffic is becoming one of the crucial issues with increasing population and number of vehicles. Traffic jams not only cause stress and delays for the drivers but also increase transportation costs and incidents. There can be different causes of congestion in traffic such as manual control lights, red light delays, and increased demands. Thus, the conventional management of traffic lights is one of the critical factors affecting poor traffic flow. To this end, transport organizations shift out to Smart Traffic Light System (STLS) as a solution that could make a significant contribution to manage a wide range of events. Besides, it could occur frequently at intersections and dynamically control them to guarantee optimal congestion.

In this chapter, we present some background and preliminaries necessary to understanding the related works and our contributions detailed in the rest of this thesis in the domain of STLS. It is important to underline that our thesis is located at the intersection of three main areas of research which are: Smart Traffic Light System, Computer Vision, Swarm Optimization, Traffic flow Management using Wireless Sensors Networks (WSN) and Artificial Intelligence (AI) approaches. Figure 1.1 shows the STLS research domains. First, we provide the fundamental preliminary definitions of concepts related to STLS and give an overview of different challenges in this research area. After that, we present the use of computer vision domain in STLS. Then, we present some standard approaches of STLS systems and their drawbacks related to traffic flow management systems. Finally, we conclude this chapter and present some lessons learned.

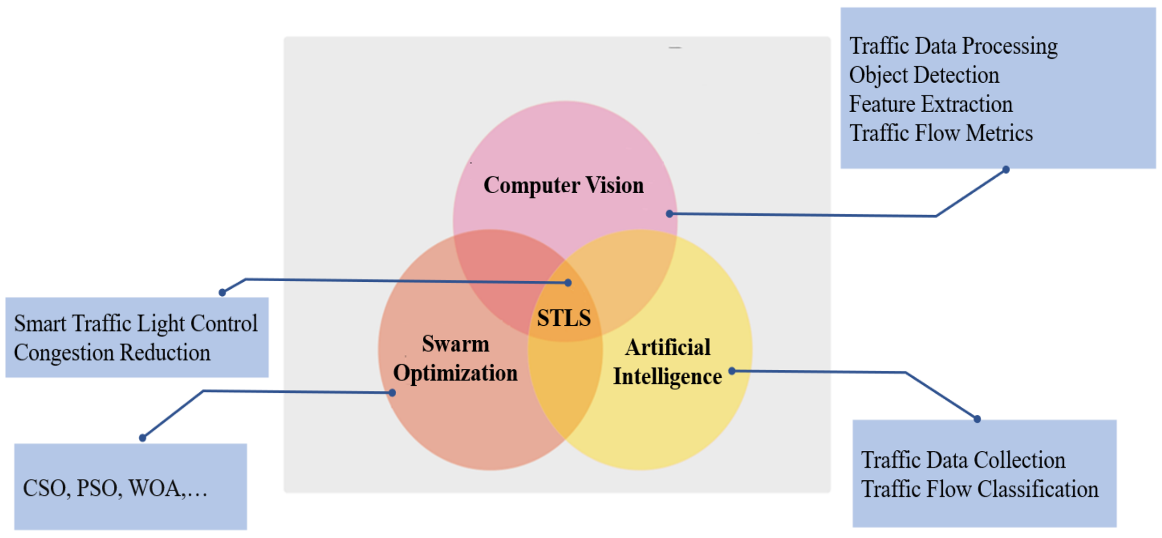


Figure 1.1 Smart traffic light system research domains.

2. Smart Traffic Light System in Smart Cities

2.1 Evolution of Traffic Light Control System: From Manual to Smart Control

Over the past few decades, the control of vehicle traffic has been one of the success keys factors in the development of a country. The vehicle traffic system being used first manual controlling as and even suggests require manpower to control the traffic flow. Policemen are allotted to a required area to control traffic using signboards, sign lights, and whistles. The second one is current automatic traffic light is controlled by timers and sensors. In traffic light, a fixed value is loaded in the timer. The lights are automatically switched to red and green based on the timer value. The last one is almost an intelligent traffic light system.

In the last decade, Wireless Sensors Networks became mandatory support to link different vehicles and witnessed a fast development process in various areas of daily live. Smart and modern cities target this technology to achieve full control of vehicular traffic. However, intelligent strategies have become indispensable to ensure city advances and quality of life. This involves the provision of appropriate road structures and efficient management of intersections. Therefore, vehicular traffic congestion at road intersections is a serious obstacle to the expected development of smart cities. With proper integration of STLS, most congestion problems can be solved. Indeed, multiple solutions have been proposed in the STLS specialized literature, either for a single intersection [7, 8, 14, 16,23] or for a set of intersections [9, 11, 17] and that according to different approaches [7-17].

2.2 Architecture of Smart Traffic Light System

Smart Traffic Light System (STLS) is a modern system, involving multimedia sensors, data processing, communication infrastructure, remote control, and interactive human-machine interfaces to monitor, analyze and efficiently improve traffic flow management. This is expected to reduce congestion and contribute to the appropriate development of more intelligent and resource-efficient cities. These technologies could include surveillance cameras and sensors to measure vehicle flow, and driver guide system. We show an example of STLS in Figure 1.2. Generally, the system STLS consists of the following components to ensure its effectiveness in aiding traffic management:

- **Sensors and data collection:** STLS uses technologies such as video cameras, infrared sensors, and radar systems for real-time traffic data collection. Besides, IoT devices gather information about the current traffic density, vehicle types and pedestrian movements [29].

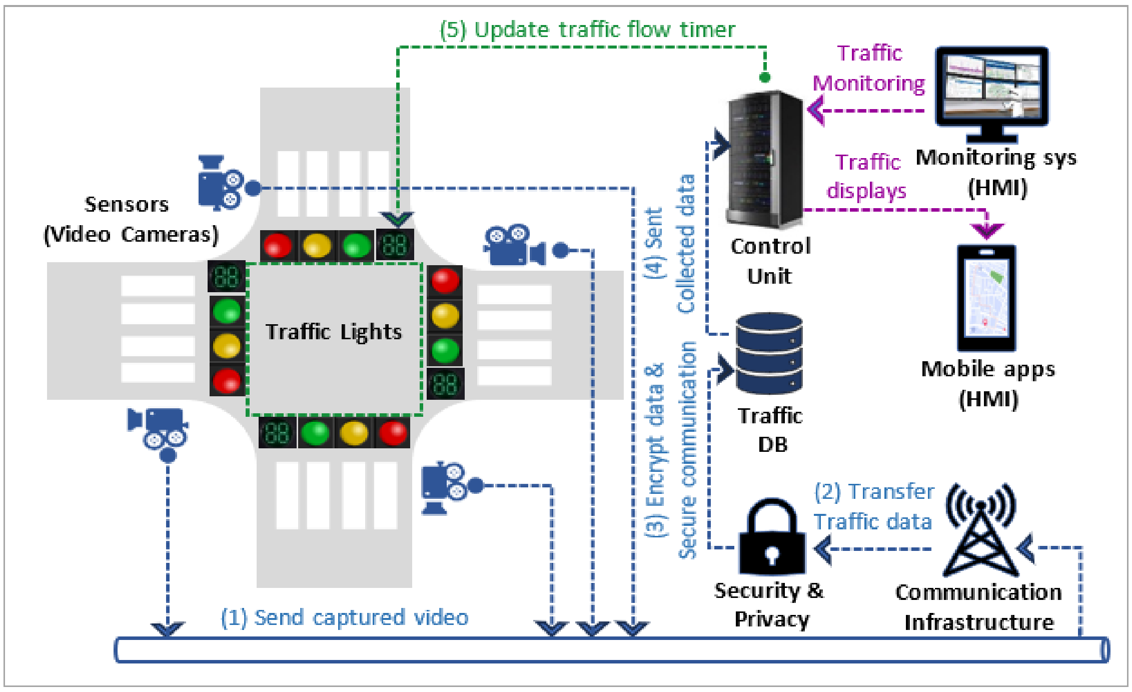


Figure 1.2 Smart traffic light system architecture.

- **Communication infrastructure:** consists of high-speed communication networks (5G, Wi-Fi) to ensure seamless traffic data transfer. It is also defined as dedicated communication protocols which provide secure and reliable information exchange. It also leverages the existing Intelligent Transportation Systems (ITS) networks [30].
- **Data processing and analysis:** operated by edge computing nodes to process data in real-time. It uses artificial intelligent algorithms that enable data analysis in real-time and dynamically adjust traffic signal timings based on current traffic conditions. It refers also to Cloud-based analytics for historical data analysis [31].
- **Control Unit**: can either be centralized, decentralized or distributed for real-time decision-making. It is based on adaptive control algorithms to dynamically adjust signal timings based on traffic conditions. Such control unit is the main part of traffic management systems that ensures citywide control centers [32].
- **Security and data privacy**: are parts of STLS systems that encrypt traffic data and ensure secure communication protocols. It is based on anonymization techniques for protecting personal traffic data privacy. Regular security audits and updates to safeguard against cyber threats [33].

– **Human-Machine Interface (HMI)**: there are two types of HMIs, namely, User Interface for operators and Mobile Apps and public displays defined as follows:
  - **User Interface for operators**: a graphical interface for traffic management operators to monitor the system, manually intervene if necessary, and analyze historical data [34].
  - **Mobile apps and public displays**: information about the current traffic conditions and current suggested routes which can be communicated to drivers through mobile apps or electronic displays [34].

2.3 Components of Smart Traffic Light System

2.3.1 Roads and Intersection

An intersection is a place where two or more roadways meet or cross, managed by several traffic lights located on the different lanes entering the intersection [35]. In other words, an intersection can be defined as a junction where two or more ways intersect. There are several categories of intersections: *controlled*, *signaled, and uncontrolled*. *Controlled intersections* are equipped with a stop sign, marking, or managed by authorized personnel. *Signalized intersections* are provided with automatic traffic lights. Finally, *uncontrolled intersections*, where priorities and traffic flow are left to the appreciation of the users.

For the STLS domain, the focus is on signalized intersections. Figure 1.3 shows an example of a four-way signalized intersection.

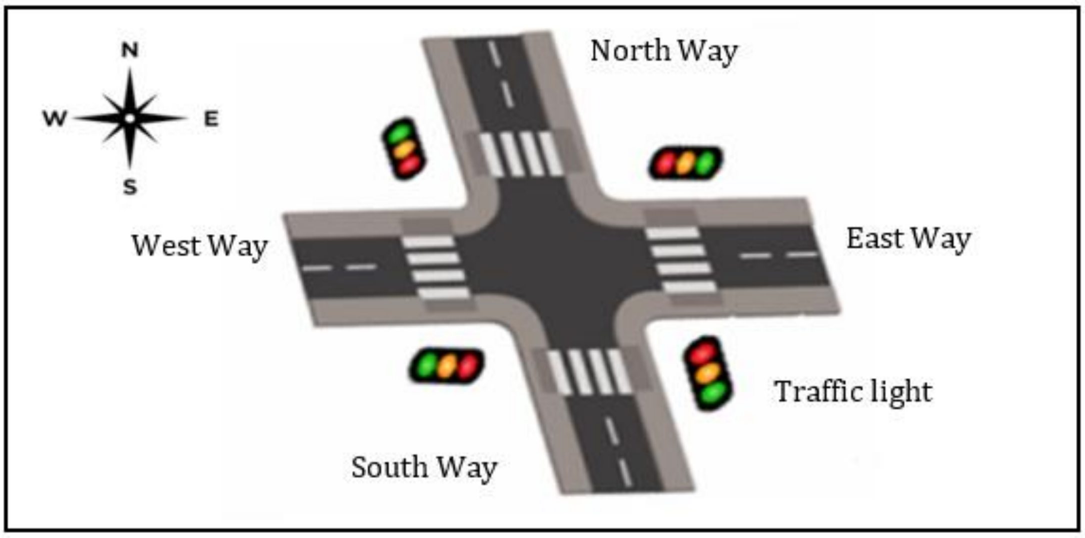


Figure 1.3 Model of a four-way signaled intersection [35].

2.3.2 Traffic Lights

Traffic lights are devices installed at intersections or other road junctions to control the flow of vehicular and pedestrian traffic. In STLS, these traditional traffic lights are equipped with intelligent capabilities, such as real-time monitoring, adaptive control algorithms and connectivity to a centralized control system [36].

Several types of traffic lights in a STLS, such as [36]:

- **Standard traffic lights**: are traditional traffic lights with intelligent capabilities. They generally consist of red, yellow and green lights to control vehicle traffic.
- **Pedestrian lights**: are traffic signals designed specifically to regulate pedestrian movements at intersections. In STLS, pedestrian signals can include functions such as countdowns and audible signals.
- **Bicycle signals:** In areas with high bicycling levels, dedicated bicycle signals can be installed to provide safe and efficient passage for cyclists. These signals can be integrated with STLS to coordinate vehicle and pedestrian movements.
- **Priority signals:** are used to give priority to specific modes of transport, such as buses, emergency vehicles, or pedestrians. In STLS, priority signals can be dynamically activated according to traffic conditions and real-time demand.

2.3.3 Cycle and Phase

- **Phase:** a phase of a traffic light is a period during which one or more coherent flows area allowed into the intersection [35].
- **Cycle:** a cycle of a traffic light represents the time between two identical phases of the intersection. It is defined by a sequence of phases [35]. Figure 1.4 illustrates two traffic light cycles composed of four phases.

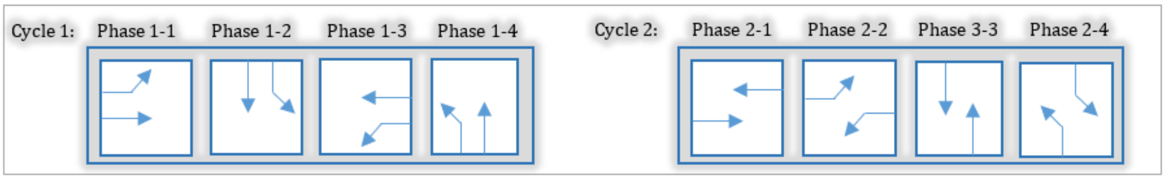


Figure 1.4 Two-cycle models with four phases.

2.4 Sensors and Detectors

The use of sensors in the field of road traffic occurred a century ago, around the 1920s in the United States of America, when traffic light controllers replaced manual signals at fixed intervals. In the same period, two engineers pioneered traffic monitoring, the first, Charles

Adler, Jr. designed a sensor-activated traffic control system that modified steering based on the presence of a vehicle horn, and the second, Henry A. Haugh, developed a pressure-sensitive sensor on the roadway, using two metal plates that functioned as electricity connections. For recognizing the presence of cars on the road, the latter design was more successful than the horn sensor and became the main sensor for decades [37].

2.4.1 Sensors Technology

For vehicle monitoring in the domain of traffic flow management, two sensor technologies are used:

- Intrusive sensors are introduced on the road, such as inductive loop sensors Figure 1.5, magnetic sensors, pneumatic sensors and moving weight sensors. Some of the advantages of this sensor model are surveillance of traffic flow for a regular period, and insensitivity to weather conditions such as rain, fog, and snow. Among their disadvantages are high installation and maintenance costs, due to the difficulty of installation which requires the removal of existing road surfaces and the closure of the road [38].
- Non-intrusive sensors, which can be installed out of the road and demand little upkeep, include radar microwave system, video and image process system, ultrasonic passive control and infrared passive control system Figure 1.5. The advantages of this model are easy installation and maintenance, and low-cost technology except for the video camera system. The major disadvantage of these sensors is the sensitivity to metrological conditions, such as weather variation and wind conditions, which can influence their functioning [38].

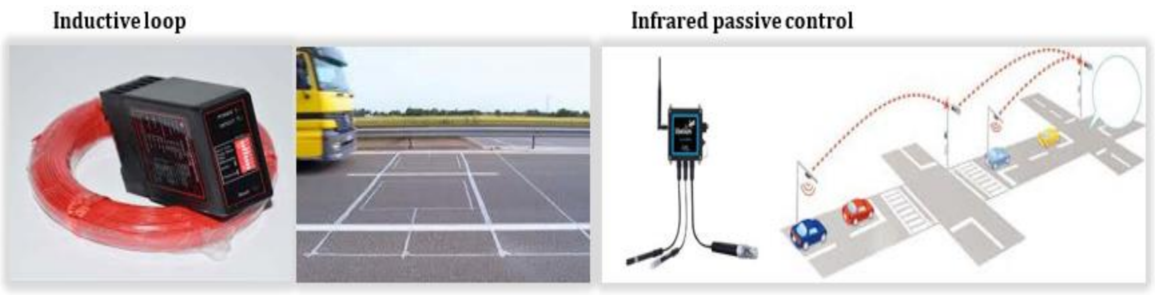


Figure 1.5 Intrusive and non-intrusive sensor models [38].

2.4.2 Motivation for Using Wireless Sensors

The convergence of technological advances in the wireless communication domain and the miniaturization of electronic components have given rise to devices of remarkable interest whose use has affected many industrial and socio-economic fields. These tiny wireless devices or sensors, often grouped to form wireless sensor networks, provide important services via distributed applications that can be integrated into almost all areas of everyday life.

The main reasons for introducing wireless sensors in the field of road traffic are the following:

– Flexibility, maintainability, manageability, and inexpensive components.
– Analysis, improvement and evolution of STLS.
– Facilitating the use of transport modes by users.
– Real-time detection and evaluation of expected events: incidents, accidents, etc.
– Automating access management to parking areas.
– Etc.

It should be noted that wireless sensors play a fundamental role in the conception of digital platforms for autonomous and connected vehicles. Whether for reasons of safety of vehicle movements and passengers or comfort needs on board vehicles. Sensors and more precisely the network that includes them, is an essential element within a vehicle and in general within an intelligent transport system.

2.5 Smart Traffic Light Management Process

The smart traffic light management process includes several key steps to effectively monitor, analyze and control traffic flow at intersections. Here, we describe these steps as follows:

2.5.1 Traffic Data Collection

Various types of sensors such as video cameras, radar sensors, and inductive loops, are deployed at intersections to collect data about the current traffic volume, vehicle speed, and queue length. These sensors constantly monitor traffic conditions and provide real-time data to the traffic management system [39].

2.5.2 Traffic Data Processing

The collected data is pre-processed and analyzed using computer vision and intelligent algorithms to extract significant and useful information about traffic patterns, and congestion rates at different times of the day. Machine learning and artificial intelligence techniques can be used to predict traffic behavior and optimize traffic light timings [39].

2.5.3 Decision-Making

The smart traffic management system dynamically adjusts signal times to improve traffic flow and reduce congestion based on real-time traffic analysis. This system uses adaptive control algorithms to adjust green time in real-time based on current traffic conditions including bad weather, emergencies and incidents [39].

2.6 Emergency Vehicles Signal Management Scenarios

Emergency vehicle signal management involves giving priority to emergency vehicles such as ambulances, fire engines and police cars at intersections to ensure their timely passage through traffic as well as improving traffic performance. The adaptive signal control real-time data based on current traffic conditions to adjust signal timings and traffic flow optimization. In the presence of an approaching emergency vehicle, the system can dynamically modify signal timings to allow the emergency vehicle to proceed on green while minimizing disruption to other traffic [40].

3. Computer Vision in Smart Traffic Light Systems

Computer vision plays a vital role in helping STLS collect real-time traffic data effectively and accurately to detect vehicle and pedestrian movement. Modern traffic flow management extracts all the significant traffic information from images and exploits computer processing capabilities with the ultimate goal of improving traffic flow control and lowering the overall cost.

3.1 Introduction to Computer Vision

Computer vision is an Interdisciplinary domain that enables computers to interpret, process, and analyze visual data from the real world. It includes a wide range of methods and algorithms intended to extract meaningful information from images or videos. With computer vision, it recognizes objects, detects patterns and understands the content of visual data by imitating the human visual system. Numerous applications including autonomous vehicles, medical imaging, surveillance and security, augmented reality, robotics, industrial automation and e-commerce can benefit from computer vision [41].

Computer vision algorithms are often comprised of several tasks, image pre-processing, feature extraction, object detection, segmentation, classification, and scene understanding. These algorithms have evolved from traditional image processing techniques to advanced machine and deep learning models [41].

3.2 Traffic Video Images Processing Methodology

The traffic image processing methodology involves the application of computer vision techniques. It analyses and extracts valuable traffic information from different images or videos captured at traffic intersections or roadsides, computes traffic flow metrics, and then integrates them into the traffic management system. Figure 1.6 shows a typical methodology for processing traffic images.

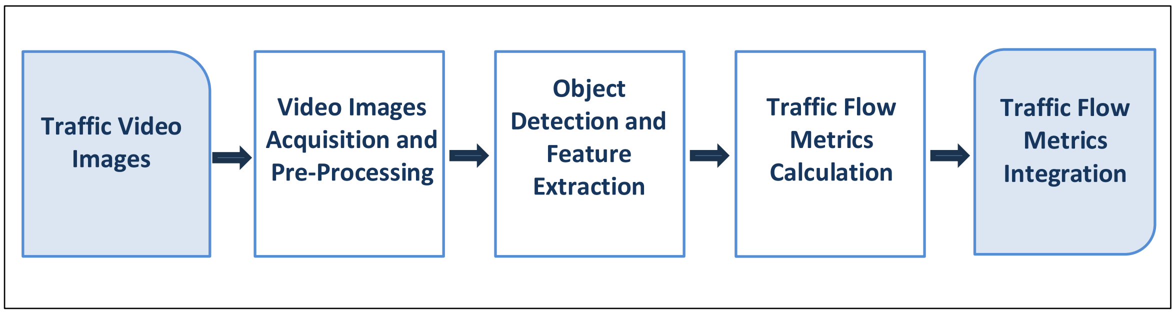

Traffic Video Images
Video Images Acquisition and Pre-Processing
Object Detection and Feature Extraction
Traffic Flow Metrics Calculation
Traffic Flow Metrics Integration

Figure 1.6 Traffic image processing methodology.

3.2.1 Video Images Traffic Acquisition and Pre-Processing

Video images are captured by traffic surveillance cameras installed at various intersections on roadside poles. These cameras capture clear and detailed objects of the traffic landscape including vehicles, pedestrians, and road markings. Pre-processing techniques are applied to images or video traffic for quality improvement and traffic flow analytics. They applied multiple image treatment methods including noise reduction, image stabilization, color correction, and image resizing [42].

3.2.2 Object Detection and Traffic Feature Extraction

Computer vision algorithms are essential for efficient smart traffic management systems. Through computer vision algorithms, the smart traffic management system can detect and track objects (i.e., vehicles, peoples) in the traffic scene. Generally, the Haar cascades, HOG (Histogram of Oriented Gradients), or deep learning-based approaches such as YOLO (You Only Look Once), are targeted for this particular purpose. Visual features such as edges, corners, textures and shapes are extracted from detected objects to characterize their appearance and properties, enabling further traffic analysis [42].

3.2.3 Traffic Flow Metrics Calculation

There are many different traffic flow metrics in today's smart traffic management system. Traffic metrics can evaluate the number of vehicles, speed of vehicles, density of vehicles and vehicle occupancy rates based on detected objects and their trajectories. These metrics provide information about traffic jams, travel patterns, and general traffic conditions. To evaluate such metrics, several computer vision techniques are widely used. These techniques are summarized as follows [43]:

- **Vehicle detection:** computer vision techniques perform vehicle detection and localization based on traffic camera images or videos. These techniques include background subtraction, followed by object detection and accurate vehicle detection using deep learning approaches.
- **Vehicle tracking:** once vehicles have been detected, computer vision techniques also track the vehicle trajectories over time, which alerts the flow traffic controller about the vehicle movement changes and interactions with other vehicles to improve the traffic flow management. Tracking algorithms detect and identify the vehicles from one image to another, and predict the vehicle's travel time from its origin to its destination based on the vehicle's current speed and directions.
- **Traffic density:** is calculated based on the number of vehicles available in a particular area or lane of urban road. It can be measured by the number of vehicles per unit area, or by other metrics such as occupancy rate, which represents an interval of time that vehicles are taken in a particular area.
- **Traffic speed:** is calculated based on the analysis of movement of vehicles between two consecutive video images. By tracking the movement of vehicles over time, their mean speed can be estimated over different traffic segments or lanes.
- **Traffic flow:** represents the speed at which vehicles pass a given point or segment of urban road. It is calculated as the product of traffic density and speed. It is often expressed as the number of vehicles per unit of time (*e.g. vehicles per hour*).
- **Queue length:** This is calculated based on the number of waiting vehicles in queues at intersections. Computer vision algorithms analyze the spatial distribution of vehicles and identify the vehicle queues waiting.

Traffic flow measurements are usually integrated with traffic management systems to ensure easy real-time decision-making by optimizing signal timings and implementing adaptive traffic control strategies to reduce congestion and improve traffic flow efficiency.

Figure 1.7 shows examples of vehicle detection using computer vision techniques.

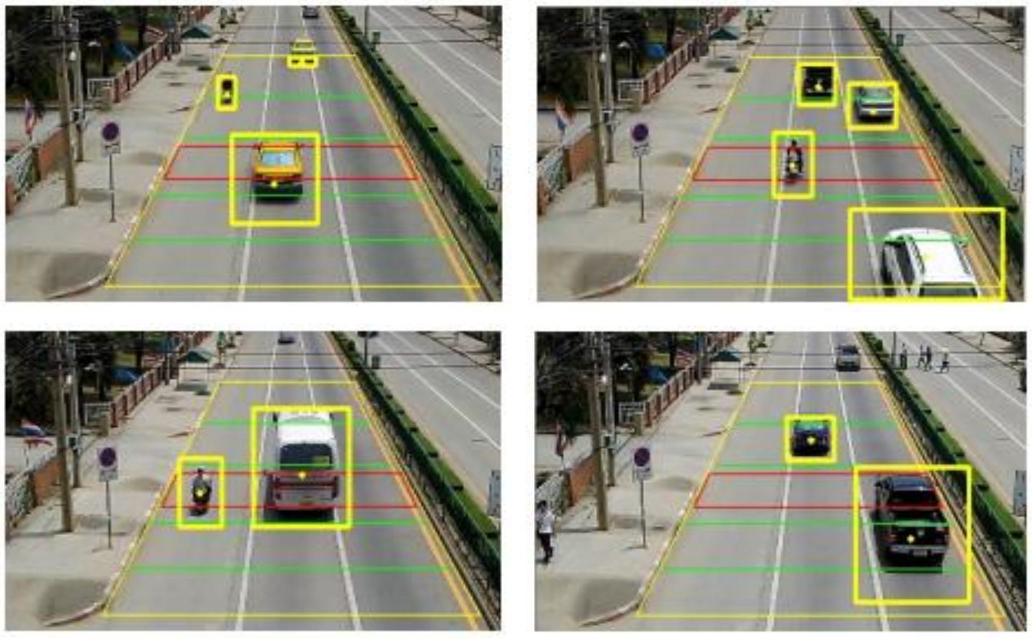


Figure 1.7 Computer vision for vehicles detection [44].

4. Swarm Optimization Metaheuristics in Smart Traffic Light Systems

Swarm optimization in STLS is an emerging technique that has gained significant attention in the field of traffic flow control and optimization. It refers to the collective behavior exhibited by a set of agents, each following simple rules, which collectively result in complex and intelligent global patterns. The phenomenon takes inspiration from the behavior of ant colonies, flocks of birds or schools of fish, where individual agents interact with their environment and each other to achieve common goals without any central control or coordination. In the context of traffic light systems, swarm optimization algorithms, such as Particle Swarm Optimization (PSO) [45] or Ant Colony Optimization (ACO) [46], are used to adaptively control traffic lights based on real-time conditions, improving traffic flow and reducing congestion.

4.1 Concepts of Swarm Optimization

Swarm optimization is a nature-based optimization model that simulates the natural features and collective behavior of different animals such as birds, ants or fish. To detail the concepts of swarm intelligence, let's explore some key concepts [47, 48]:

- **Swarm Intelligence:** Is based on the collective behavior of decentralized, self-organizing systems where individual agents (or particles) interact locally with their environment and other agents to achieve global goals. The key idea behind swarm intelligence is that simple agents adhering simple rules can lead to sophisticated, intelligent global behaviors without central oversight. Examples include bird flocks, fish schools and foraging ants.

– **Population-Based Search:** Swarm optimization algorithms use a population of agents (solutions) to simultaneously explore the solution space. Each agent represents a potential solution to the problem, and the collective aim to improve these solutions over time. This approach helps prevent premature convergence towards local optima, allowing different agents to explore the search space concurrently.
– **Exploration and Exploitation Balance:** Exploration is the process of searching different parts of the search space to discover new solutions. It prevents agents from getting stuck in sub-optimal locations. Exploitation refines solutions for further optimization by improving the best solutions found so far. Swarm algorithms balance exploration and exploitation to achieve global optimization, ensuring don't settle agents in local optima while prioritizing the improvement of promising solutions.
– **Decentralized Decision-Making:** Swarm intelligence enables collective decision making by pooling the knowledge and preferences of individual agents. By leveraging the wisdom of the crowd, swarm systems can often achieve better outcomes than relying on a single decision making. For example, an agent discovers a new solution based on local information and simple rules, then communicate information about potential solution, and when a certain number of agents agree on a solution, the swarm collectively decides to consider that solution. Thus, the absence of central control enables the creation of flexible, scalable systems. This decentralized decision making allows the swarm to adapt to changing conditions and make optimal decisions.
– **Fitness Function and Objective Optimization:** The fitness function (or objective) evaluates the quality of a solution. Each agent in a swarm optimization algorithm uses this function to evaluate its position or solution. The aim is to either minimize or maximize the fitness function, guiding the agents towards the optimal solution iteratively.
– **Collective Behavior:** Agents work together by sharing information or exerting influence (e.g., the position or solution of a neighboring agent) to achieve a common goal. No single agent has complete knowledge or control over the entire system. Instead, collective behavior arises from simple interactions of agent with their

neighbors that enable them to converge on a high-quality solution. An excellent example is how a group of ants can find the best solutions and exhibit coordinated movements without any central leader.

- **Emergent Behavior:** When groups of individuals interact with each other, they can exhibit behaviors not found in any single individual. This emergent behavior arises from complex interactions and coordination among agents, leading to the emergence of new patterns or behaviors at a higher level of organization. For example, in ACO, each ant selects paths based on pheromone concentration which ultimately leads the colony to collectively discovers the shortest path to a food source.
- **Stochastic and Probabilistic Nature:** Swarm optimization incorporates random elements into agent behavior to ensure comprehensive exploration of the search space. This reduces the risk of agents getting stuck in sub-optimal solutions. In some algorithms, agents make decisions based on probabilities (for example, the probability of selecting a particular path based on pheromone levels in ACO or the relative fitness of solutions in PSO).
- To summarize, swarm optimization is an adaptive, decentralized, population-based approach to solving complex optimization problems. It balances exploration and exploitation, enabling individual agents to follow straightforward rules and make decentralized decisions. Thanks to collective and emergent behavior, the swarm probabilistically converges towards optimal solutions. Figure 1.8 shows the concepts of swarm optimization.

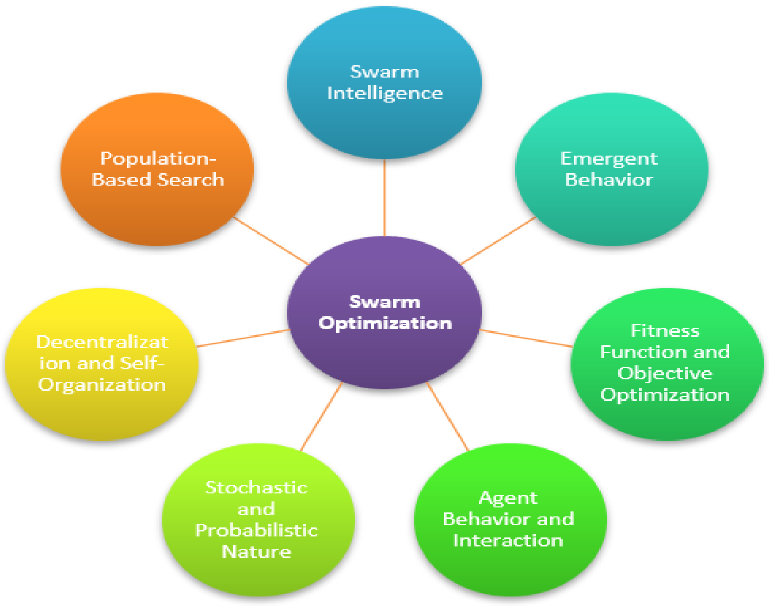

Swarm Intelligence
Population-Based Search
Emergent Behavior
Swarm Optimization
Decentralization and Self-Organization
Fitness Function and Objective Optimization
Stochastic and Probabilistic Nature
Agent Behavior and Interaction

Figure 1.8 Concepts of swarm optimization.

4.2 Standard Swarm Optimization Algorithms and Applications

Swarm optimization algorithms represent a class of bio-inspired computational techniques aimed at solving complex optimization issues by simulating the collective behavior of decentralized systems. They are defined by the presence of simple agents (or “particles”) that collaborate to find optimal solutions without any central control. Below are some of the most common swarm optimization algorithms and their applications.

4.2.1 Particle Swarm Optimization (PSO) Algorithm

PSO is a population-based optimization algorithm inspired by the social behavior of birds in flight or fish in schools. The operational principals of PSO are based on simulating the movement and interaction among particles in a search space to find the optimal solution. PSO was proposed by J. Kennedy and R. Eberhart [49] in 1995. In PSO, particles move in a multi-dimensional search space, where each particle's position represents a candidate solution to the optimization problem. Each particle maintains its current position and velocity, and continuously update them to find out the optimal solution. The interactions between particles for the PSO enables the swarm to solve complex optimization problems while maintaining a balance between exploration and exploitation. The pseudo code of PSO is shown in Algorithm 1.1.

Algorithm 1.1 Particle swarm optimization (PSO).

```
1. Initialize velocity $v_i$ and position $p_i$ of each particle $i$.
2. Evaluate the objective function and updates local best position $pBest$ .
3. Update the position and velocity of each particle $i$ according to its current velocity
4. Update the global best position $gBest$ according to local best positions.
5. Repeat Steps 2 to 4 until particles drives towards the same best global position.
```

4.2.2 Chicken Swarm Optimization (CSO) Algorithm

CSO is a nature-inspired metaheuristic algorithm. It is based on the social behavior of chickens and hierarchical structure and dynamics within a swarm, particularly within a flock. CSO was proposed by X. Meng *et al.* [50] in 2014. CSO simulates a population of chickens that play various roles (roosters, hens and chicks), which interact based on their social status to find optimal solutions in a given problem space. Roosters is the best individuals in the population and shares their knowledge with hens, but compete with other roosters for food (optimal solutions). Hens is the intermediate individuals, who follow roosters to gain knowledge but may also attempt to steal food (solutions) and chicks is the least competent individuals in the population, relying on their mother hens (parents) for guidance. Each of these components is associated with specific movement and learning strategies. The interaction between these roles reflects competition, cooperation and social hierarchy, facilitating the discovery of optimal solutions through collective search behavior. The pseudo code of CSO is shown in Algorithm 1.2.

Algorithm 1.2 Chicken swarm optimization (CSO).

```
1.  Initialize chicken population.
2.  Evaluate the fitness of all chickens
3.  While (max iterations is not met)
4.    Sort chickens by fitness and establish the hierarchical order in the swarm
5.    Divid the swarm into different groups: chicks, mothers and hens
6.    Update position of rooster
7.    Update position of hen
8.    Update position of chick
9.    Evaluate the new solution
10.   Update current best solution if the new solution is better.
11. End While
12. return best solution.
```

4.2.3 Ant Colony Optimization (ACO) Algorithm

ACO is a population-based metaheuristic algorithm inspired by the foraging behavior of ants. In 1990, M. Dorigo *et al.* [51] studied the mechanism of ant colony to find the shortest path from their colony to food sources. In this study, some ants deposit a pheromone on paths they traverse in each iteration. Other ants are attracted to the paths with the highest pheromone and, over time, paths that contain more pheromones are reinforced, facilitating the discovery of optimal or nearly optimal solutions. ACO has been widely applied to combinatorial optimization problems, particularly those related to pathfinding and routing, where finding optimal solutions can be computationally difficult. The pseudo code of ACO is shown in Algorithm 1.3.

Algorithm 1.3 Ant colony optimization (ACO).

```
1.  Initialize ant colony.
2.  Initialize the pheromone levels.
3.  While (termination criteria is not met)
4.      For each ant
5.          Built a path based on the pheromone levels and a local heuristic.
6.          Update the pheromone levels
7.      End For
8.  End While
9.  return Path with stronger pheromone trails.
```

4.2.4 Artificial Bee Colony (ABC) Algorithm

ABC is a swarm intelligence-based optimization algorithm inspired by the foraging behavior of honeybees. It is recently proposed by D. Karaboga [52] in 2005 to simulate how bees search for nectar sources (solutions) and share information with their hive to locate the most beneficial food sources (optimal solutions). This algorithm is particularly effective for complex, non-linear and multidimensional optimization problems. The algorithm divides the bees in a colony into three distinct categories, each performing specific tasks:

1. **Employed bees:** These bees exploit the food sources (solutions) assigned to them, and convey the quality of these sources to other bees. Each employed bee represents a particular solution in the solution space,
2. **Onlooker bees:** Onlooker bees evaluate the information shared by employed bees and select food sources based on the quality of the solutions. The probability of selecting a food source being proportional to its quality and
3. **Scout bees:** When a food source is exhausted (i.e. a solution is no longer improving), a scout bee is designated to randomly search for new sources (solutions), enabling the algorithm to explore new areas of the solution space.

The pseudo code of ABC is shown in Algorithm 1.4.

Algorithm 1.4 Artificial bee colony (ABC).

```
1. Initialize bees.
2. While (Termination is not met) /* Termination is meet when optimal solution is found*/
3.   Employed bees search for new solution
4.   Onlooker bees probabilistically choose one solution using objective function
5.   Scout bees' phase
6.   Update Best solution by greedy algorithm
7. End While
8. Return Best solution.
```

4.2.5 Firefly Algorithm (FA)

FA is an optimization technique inspired by nature and based on the bioluminescent behavior of fireflies. It is developed by X. S. Yang [53], in 2009 to simulate how fireflies attracted to each other according to the intensity of their light, which reflects the solutions quality in the algorithm. The fundamental concepts include: 1) - *attractiveness***:** fireflies are attracted to each other according to their brightness, which represents the solution quality, 2) - *attraction based on distance* between two fireflies decreases as their distance increases and 3) - *movement***:** a firefly moves towards a brighter one, enabling the algorithm to efficiently explore the solution space. The purpose of FA is to find optimal or near-optimal solutions by balancing exploration (global search) and exploitation (local search) using these natural behaviors. The pseudo code of FA is shown in Algorithm 1.5.

Algorithm 1.5 Firefly algorithm (FA).

```
1. Initialize firefly population.
2. While (max iteration is not met)
3.   Evaluate objective function
4.   Rank firefly
5.   Find the best solution
6.   Move all firefly to their best solutions
7. End While
8. return best solution.
```

4.2.6 Bat Algorithm (BA)

BA is an optimization algorithm inspired by echolocation behavior of bats, developed by X.S. Yang [54] in 2010. Particularly how microbats use sonar to detect prey and avoid obstacles. Bats emit high-frequency sound waves, which they use to listen to the echoes and adjust their flight path accordingly. This natural behavior is mathematically modeled in the bat algorithm to efficiently explore and exploit the solution space. The Bat algorithm is both flexible and easy to implement, making it suitable for solving a wide range of optimization problems, particularly in complex, non-linear and multimodal search spaces. Each bat emits a unique frequency and pulse rate that balances the exploration and exploitation phases. A higher frequency with a lower pulse frequency correspond to a more global exploration, while a lower frequency and a higher pulse frequency led to a local search around the right solutions. The pseudo code of BA is shown in Algorithm 1.6.

Algorithm 1.6 Bat algorithm (BA)

```
1.  Initialize population of bats.
2.  Initialize loudness and pulse rate
3.  While (max generation is not met)
4.      Produce new solution by setting frequency
5.      Update velocities and solutions
6.      If rand> pulse rate, Then
7.          Choose solution from the best solution
8.          Produce a solution around the chosen local solution
9.      End If
10.     By flying randomly, generate new solution
11.     If rand > pulse rate and $f_i > f$ Then
12.         Accept new solution
13.         Reduce loudness and increase pulse rate
14.     End If
15.     Rank the bats
16. End While
17. return best solution.
```

4.2.7 Cuckoo Search (CS) Algorithm

CS is a meta-heuristic algorithm inspired by some cuckoo species laying in the nest of other species birds, developed by X. S. Yang and S. Deb [55] in 2009. In CS, we have two bird's species: cuckoos and host birds. Bird, they do not have any nest they lay their eggs in the nests of other species, we have a nest for the host bird cuckoo bird will lay the egg in the nest of the host bird. If the host bird detects the foreign egg, it can either abandon the nest and build completely new egg. The algorithm is also inspired by Lévy flight, a random walking process used by animals to explore their environment in search of food. CS leverages these natural strategies to perform a global optimization, using a population of candidate solutions (cuckoos) and replacing the weakest solutions (the host eggs) with the best. The search process is driven by two main mechanisms: (1) **Laying eggs in random nests:** A set of candidate solutions is maintained, and new solutions are created by modifying existing ones. Poor solutions are replaced by more promising ones, imitating the cuckoo's strategy of laying eggs in host nests and (2) **Lévy flight:** New solutions are generated using Levy flight enabling the algorithm to explore the solution space in large steps, improving global exploration and helping the algorithm escape local optima. Cuckoo Search is known for its simplicity, efficiency, and ability to handle a wide range of optimization problems, particularly those with complex, multimodal landscapes. The pseudo code of BA is shown in Algorithm 1.7.

Algorithm 1.7 Cuckoo search (CS).

```
1.  Initialize CS parameters.
2.  Initial generation of host nest population
3.  **While** (max iteration is not met)
4.    Get a cuckoo randomly by Levy flight and evaluate its fitness $f_i$
5.    Choose a nest randomly among $n$ (set $j$)
6.    **If** ($f_i > f_j$) **Then**
7.      Replace j with the solution
8.      Abdnndon worst Egg
9.    **End If**
10.   Rank solution and find current best solution
11. **End While**
12. **return** best solution.
```

4.2.8 Grey Wolf Optimizer (GWO) Algorithm

The bio-inspired Grey Wolf Optimization (GWO) algorithm is a meta-heuristic approach developed by Mirjalili and colleagues [56] in 2014. GWO is inspired by their hunting and prey-seeking behavior in nature. In GWO, the population is divided into four categories: alpha, beta, delta, and omega. The alpha wolf is the pack leader responsible for decision-making. The beta wolf is the second leader, assisting the alpha in decisions and other activities. The delta wolf is the third leader of the group, dominating the omega wolves. Mathematically, the three fittest solutions in GWO are denoted by the letter's alpha (α), beta (β), and delta (δ) respectively. The other individuals are considered omegas (ω). In GWO, the hunting process is guided by (α), (β), and (δ), while (ω) follows these three leaders.

The hunting process of grey wolves consists of three main phases, which are reflected in the GWO algorithm: 1) - **encircling the prey:** Wolves surround the prey and move towards it by adjusting their positions relative to the alpha, beta, and delta wolves. In GWO, candidate solutions are updated based on their distance to these three best solutions, 2) - **hunting:** The wolves attack the prey cooperatively by updating their positions to converge toward the best solution and 3) **attacking or searching for Prey:** Wolves either attack the prey if it is close or search for another prey if the current prey escapes. The GWO algorithm is recognized for its simplicity, efficient, and suitability for solving continuous optimization problems, especially those with complex, multimodal landscapes. The pseudo code of GWO is shown in Algorithm 1.8.

Algorithm 1.8 Grey wolf optimizer (GWO).

```
1.  Initialize Grey wolf population.
2.  Initialize wolf parameters
3.  Calculate the fitness of each search agent
4.  alpha←first best solution; beta← second best solution; delta ←third best solutions
5.  While (max iteration is not met)
6.      Update position of wolf based on alpha, beta, and delta wolves
7.      Update wolf parameters
8.      Calculate the fitness of all search agents
9.      Update position of alpha, beta, and delta wolfs
10. End While
11. return alpha.
```

4.2.9 Whale Optimization Algorithm (WOA)

WOA is a metaheuristic algorithm inspired by the social behaviors and hunting strategies of humpback whales, especially the bubble-net feeding method, proposed by S. Mirjalili and A. Lewis [57] in 2016. Humpback whales form spiral-shaped bubbles around their prey to trap them. This hunting technique is mathematically modeled in WOA to address optimization problems. WOA emulates several key behaviors:

– **Encircling prey:** The whales identify the current best solution (prey) and surround it. This is mathematically modeled by updating the positions of potential solutions towards the best solution found so far.

- **Bubble-net feeding:** This technique is modeled by two main strategies:
  - **Shrinking encircling mechanism:** Whales get closer to the prey by gradually decreasing the distance between their position and best solution.
  - **Spiral updating position:** This imitates the bubble-net feeding behavior, where whales move in a spiral around the prey, balancing exploration and exploitation.
- **Search for Prey:** Whales also conduct a random search for prey by updating their positions in the search space. This ensures a comprehensive exploration of the search space, preventing that the algorithm does not get stuck in local optima too early.

WOA alternates between these behaviors based on a probability factor, allowing it to balance exploration and exploitation effectively. It is designed to solve continuous optimization problems but can also be adapted for discrete problems. The pseudo code of WOA is shown in Algorithm 1.9.

Algorithm 1.9 Whale optimization algorithm (WOA).

```
1. Initialize the whale population.
2. Initial generation of host nest population
3. While (max iteration is not met)
4.     Calculate the fitness of each agent
5.     Select the best position of the herd
6.     Update the individual position
7. End While
8. return best position.
```

4.2.10 Genetic Swarm Optimization (GSO) Algorithm

GSO is a hybrid optimization algorithm that combines the core features of Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) to create a more efficient and robust optimization method, developed by Grimaldi and colleagues [58] in 2005. The algorithm integrates the exploration capability of GA with the exploitation capability of PSO, aiming to improve search performance, solution diversity, and convergence speed. The hybrid mechanism in GSO:

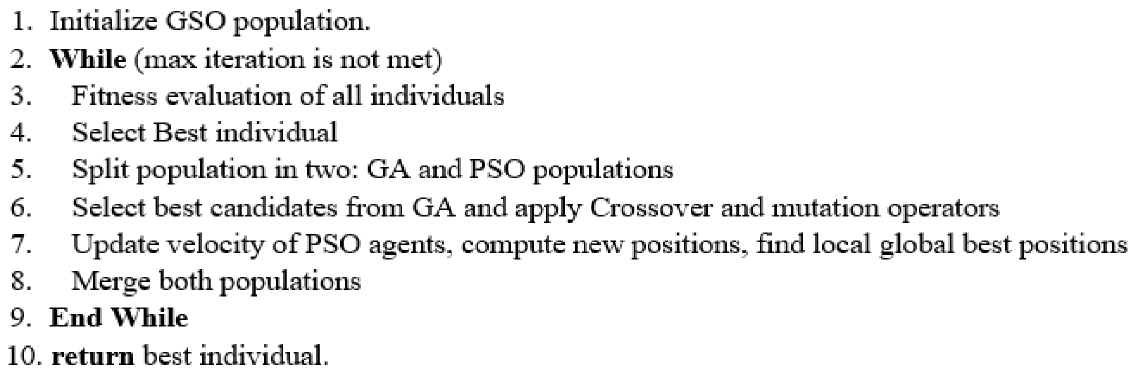
- **Selection:** GSO applies selection mechanisms from GA to choose the fittest particles from the swarm. These particles then proceed to the next generation, ensuring that only the best solutions survive.
- **Crossover:** A crossover operator is applied to particles, where portions of their "genetic material" (solution parameters) are combined, creating new particles with

traits from two or more parents. This operation helps create diversity in the search space.

- **Mutation:** To avoid premature convergence and ensure exploration of the search space, GSO applies mutation operators that randomly alter particle positions or velocities.
- **PSO Dynamics:** GSO retains the velocity and position update rules of PSO, allowing particles to move through the search space by considering both their personal and the global best positions.

By combining PSO's swarm intelligence and GA's evolutionary operators, GSO seeks to find an optimal balance between exploration (searching new areas) and exploitation (refining known solutions). The evolutionary operators ensure that the algorithm maintains diversity in the population and avoids stagnation in local optima, while PSO's swarm behavior enhances convergence toward the best solutions. The pseudo code of GSO is shown in Algorithm 1.10.

Algorithm 1.10 Genetic swarm optimization (GSO).



```
1. Initialize GSO population.
2. While (max iteration is not met)
3.    Fitness evaluation of all individuals
4.    Select Best individual
5.    Split population in two: GA and PSO populations
6.    Select best candidates from GA and apply Crossover and mutation operators
7.    Update velocity of PSO agents, compute new positions, find local global best positions
8.    Merge both populations
9. End While
10. return best individual.
```

4.2.11 Swarm Optimization Applications

Swarm optimization algorithms have been applied in various fields [59]:

- **Networks:** Traffic flow management, load balancing and network design.
- **Artificial intelligence and machine learning:** Neural network training, feature selection, hyperparameter tuning.
- **Engineering design:** Structural optimization, control system tuning, electrical circuit design.
- **Robotics:** Trajectory planning, multi-robot coordination, obstacle avoidance.
- **Finance and economics:** Portfolio optimization, algorithmic trading, risk management.
- **Bioinformatics:** Gene selection, protein structure prediction, sequence alignment.

These algorithms are highly versatile, making them applicable to numerous real-world optimization challenges. Table 1.1 illustrates different application domains of swarm optimization algorithms.

Table 1.1 Swarm optimization algorithm applications

Application Domains	Swarm Optimization Algorithm									
	PSO	*CSO*	*ACO*	*ABC*	*FA*	*BA*	*CS*	*GWO*	*WOA*	*GSO*
Bioinformatics	✓			✓		✓				
Biomedical Engineering		✓					✓			
Clustering				✓		✓	✓	✓	✓	✓
Communication Systems			✓		✓					
Dynamic Optimization	✓		✓	✓		✓	✓	✓	✓	✓
Engineering Design	✓	✓		✓	✓	✓	✓	✓	✓	✓
Energy Systems		✓								
Economic Modeling	✓					✓				
Image Processing	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Job Scheduling	✓	✓	✓	✓		✓	✓	✓	✓	✓
Power System	✓			✓	✓	✓		✓	✓	✓
Robotics and Path Planning	✓			✓		✓		✓	✓	
Supply Chain Management		✓	✓	✓						✓
Traffic Signal Optimization	✓	✓								
Traveling Salesman Problem			✓		✓					
Wireless Sensor Networks		✓		✓		✓	✓	✓	✓	

✓ : Considered

4.2.12 Main Advantage and Disadvantage of Swarm Optimization Algorithms

Table 1.2 presents a comparison of swarm optimization algorithms mentioned below. While some of these algorithms have applied in various applications, they also come with both advantages and disadvantages. In this research study, the CSO are used and improved to determine the optimal weights for traffic flow.

Table 1.2 Majors' advantages and disadvantages of swarm optimization algorithms

Algorithms	Majors' Advantages	Majors' Disadvantages
PSO	Easy to implement	Risk of premature convergence
CSO	Balanced exploration and exploitation	Sensitivity to parameter
ACO	Effective for combinatorial problems	High computational cost
ABC	Well-balanced exploration and exploitation	Limited search space exploration
FA	Effective for multimodal problems	Sensitivity to light absorption coefficients
BA	Fast convergence for continuous and multimodal optimization	Need for parameter tuning
CS	Fast and efficient for real-time and high-dimensional optimization	Scope of application is limited
GWO	Strong capabilities of exploration	Risk of early convergence
WOA	Flexible exploration techniques	Limited parameter control
GSO	Combine advantages of GA and PSO	Difficult to implement

4.3 Benefits of Swarm Optimization in Smart Traffic Light Systems

Swarm optimization algorithms including Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Artificial Bee Colony (ABC) and others [46, 60-62], have demonstrated significant potential in optimizing STLS. These algorithms leverage collective intelligence and decentralized decision-making to solve intricate challenges such as traffic light control. Below are some prominent benefits of using swarm optimization in STLS:

- **Real-time adaptation:** Swarm optimization algorithms are dynamic and adaptive by nature, enabling smart traffic lights to react in real time to adjust quickly to evolving conditions. They adjust traffic light schedules according to real-time traffic data, reducing delays and congestion at peak times or during unexpected traffic incidents.
- **Decentralized control:** Swarm-based approaches can be used in a decentralized manner, enabling each traffic light to act as an agent and optimizes its local traffic flow while cooperating with neighboring lights. This decentralized control minimizes the need for a central management system and makes the traffic network more resilient against failures or disruptions at individual intersections.
- **Scalability:** Swarm algorithms efficiently handle the complexity of urban traffic networks, making them suitable for large cities. Whether a small network of ~~a~~ few intersections or a large metropolitan area, swarm optimization can be adapted to control multiple traffic lights simultaneously without significant performance loss.
- **Reducing traffic congestion:** Swarm algorithms are particularly proficient at optimizing multiple objectives at the same time. In intelligent traffic systems, this means they can minimize overall vehicle delays, waiting times and queue lengths at intersections.
- **Handling complex traffic patterns:** Swarm optimization techniques can effectively handle complex and dynamic traffic situations, including those triggered by special events, weather changes or accidents.
- **Collaboration and coordination between intersections:** Swarm optimization enables traffic lights at different intersections to work together in a coordinated fashion, optimizing traffic flow across multiple intersections. This creates “green waves”, where several consecutive traffic lights turn green at the right time, enabling vehicles to pass through several intersections without stopping, thereby improving overall traffic flow efficiency.

- **Robustness and fault tolerance:** Swarm algorithms promote the robustness and fault tolerance of intelligent traffic systems. Due to their decentralized nature, these algorithms are less sensitive to failures in the overall system. If an intersection or sensor fails, the rest of the system can continue to operate, adjusting traffic lights using available data.

5. Artificial Intelligence in Smart Traffic Light System

5.1 Introduction to Artificial Intelligence (AI)

Artificial Intelligence (AI) simulates human intelligence and understands its nature by creating computer programs that simulate human behavior and intelligence. AI is no longer a marginal part of the era of the cognitive and technological revolution. This is because AI has become the most important outcome of the Fourth Industrial Revolution owing to its many uses in smart cities, AI is seen in self-driving cars, drones, real traffic flow management and other real live applications. It is noteworthy that AI can contribute to a sustainable environment and quality of life for humans in Internet age. AI continues to develop rapidly, is an essential part of our daily lives and has huge potential to achieve effective traffic management.

5.2 The Role of AI in Smart Traffic Light Systems

Traffic management can benefit greatly from AI, which continues to develop rapidly in our daily lives. To achieve durable and effective traffic flow management goals, AI can be properly exploited. Effective traffic flow management optimizes and integrates intelligent processes to ensure all aspects of traffic flow control operations. AI seamlessly with the ultimate goal of reducing vehicle waiting time, vehicle occupation rate, and lowering costs. AI is typically invisible to the end-drivers but it plays a vital role in helping drivers fulfil their tasks quickly and accurately to provide great driver's satisfaction. However, the increasing number of emergency events and congestions brings with it significant and complex challenges, ranging from issues of reducing costs and passing through security risks of traffic data, to exacerbating the problems of traffic flow optimization. Addressing these challenges may be as difficult as the solutions to achieve the goals of achieving efficient traffic flow management.

5.3 Types of AI-Powered Traffic Control Management Systems

There are three main types of traffic control management systems, namely centralized, decentralized and distributed. Transport organizations can choose, according to the movement of vehicles and people, between centralized, decentralized and distributed AI-powered traffic control management systems. Each approach has its characteristics and advantages as well as disadvantages. Figure 1.9 presents the different traffic control management systems in general.

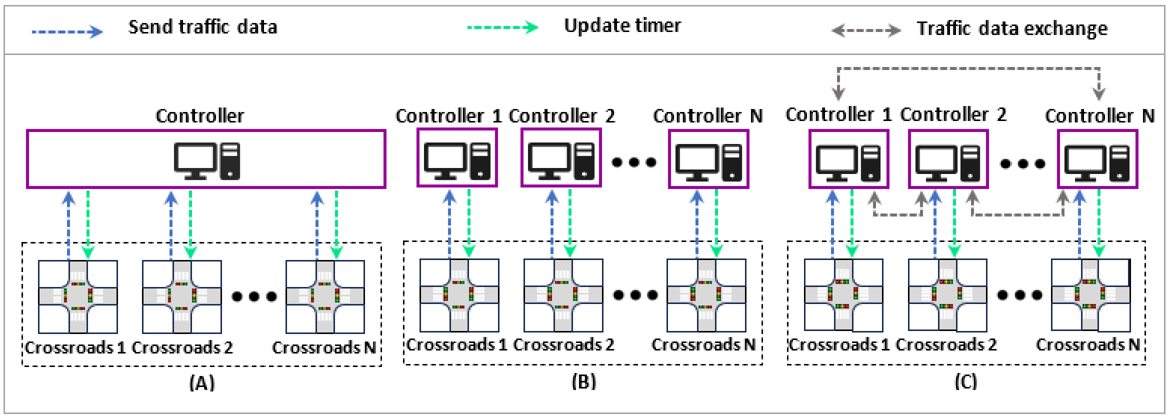


Figure 1.9 Evolution of different traffic control management systems, (A) Centralized, (B) Decentralized and (C) Distributed.

5.3.1 AI-Powered Centralized Traffic Control Management System

A centralized traffic control management system is a single controller that receives traffic data from sensors, cameras, and other sources and issues traffic decisions to the whole traffic network [63]. A centralized traffic control management system has potential advantages but also presents some disadvantages.

- **Advantages**
  - It provides unified control serving all components of the network.
  - It relies on global traffic optimization to reduce congestion, waiting, and/or travel times.
  - It allocates resources, such as traffic lights or lanes, more efficiently based on real-time data, thus improving traffic flow.
  - Suitable for changing traffic conditions, adapts and updates its decision-making policies dynamically.
  - Traffic control management costs are low.

- **Disadvantages**
  - It suffers from a single point of failure. Centralized traffic control systems are vulnerable to system failures or cyber-attacks, which could disrupt traffic management tasks for the whole network.
  - Less scalability when traffic network sizes are high.
  - Lack of privacy concerns raises privacy issues regarding sensitive information about individual drivers or vehicles.

5.3.2 AI-Powered Decentralized Traffic Control Management System

A decentralized traffic control management system is a type of system that distributes controls to several nodes or agents and not to the whole network [64]. Decentralized traffic control management may be in many numbers in the transport network, as each interaction has its own AI powers.

- **Advantages**
  - There is no main traffic control management system, decisions are distributed directly to multiple agents or nodes.
  - Suitable for intersections that are producing large volumes of data.
  - Controlled by an intersection traffic flow agent.
  - Fault tolerance and more robust than a centralized traffic control management system, reducing the risk of system-wide disruptions.
  - More scalable than centralized AI systems, this is more efficient for large and big traffic networks that distribute computations and decisions among multiple agents.
  - More privacy-preservation than centralized traffic control systems. It limits the sensitive data exchanges between agents while ensuring efficient traffic control management.
- **Disadvantages**
  - Traffic control management costs are very high.
  - It requires effective coordination between individual agents to achieve overall traffic control management objectives, which can be difficult to implement.
  - It is cost-effective resource utilization. costs are optimal when individual agents prioritize local objectives over global optimization.

5.3.3 AI-Powered Distributed Traffic Control Management System

A distributed traffic control management system is a sophisticated network of interconnected devices controller that work together to optimize traffic flow and manage congestion across a transportation network. Unlike traditional centralized traffic control systems, which rely on a single central node to manage and optimize all traffic flow, distributed systems rely on decentralized nodes to communicate and coordinate over a common traffic optimization decision [65].

- **Advantages**
  - It reacts quickly to changes in road and traffic conditions.
  - It reduces congestion and optimizes traffic flow based on real time conditions.
  - It reduces risk of single points of failure. Even if one network node fails, other nodes can continue to operate autonomously, reducing the risk of widespread disruption.
  - It adapts as traffic volume and complexity increase which enabling new nodes to be seamlessly added to the network.
  - Network nodes can adjust, according to local traffic conditions, their independent control strategies to ensure more flexibility and efficiency.
- **Disadvantages**
  - It requires careful planning and management when coordinating communication and decision-making between several controllers.
  - More costly than traditional centralized systems. However, additional infrastructure, communication networks and software for development are needed which can increase overall costs.
  - Necessity of having secure data transmission and steady storage spaces to protect traffic data against potential cyber threats and unauthorized access.
  - Lack of specialized training and practical experiences which are essential to ensure performance. It requires maintenance and updates of software bugs, hardware failures and changing traffic management needs.

5.3.4 Centralized, Decentralized and Distributed Traffic Control Management Systems

Table 1.3 presents a comparison between centralized, decentralized and distributed traffic control management systems in terms of (1) – decision-making, (2) adaptability, (3) traffic costs, (4) suitability, and (5) fault tolerance. Traffic control management systems must be smart, fast, and robust against different scenarios of equipment failures. In traffic control management systems, it is recommended to use decentralized because it is more efficient than centralized systems due to its poor scalability and poor fault tolerance. The integration of AI algorithms is ensured by efficient traffic data classification.

Table 1.3 Centralized, decentralized and distributed traffic control management systems

Centralized Control Management	Decentralized Control Management	Distributed Control Management
There is one big traffic control management system serving all components of the network	There is no main traffic control management system, decisions are distributed directly to multiple agents or nodes	There is collaborative decision-making among individual traffic signals controllers or nodes
Suitable for changing traffic conditions, adapts and updates its decision-making policies dynamically	Suitable for intersections that are producing large volumes of data.	Suitable for real-time adaptation through coordinated traffic information sharing
It is highly cost-effective (costs are low)	Traffic management costs are very high	Traffic management costs are reduced through decentralized control
It provides unified control over control operations	Controlled by an intersection traffic flow agent.	Controlled by agents that coordinate based on local traffic
It is vulnerable to system failures or cyber-attacks.	Fault tolerance and more robustness	Redundant and cooperative control prevents system failures

5.4 Coordination and Cooperation of Traffic Control Mechanisms

The terms "coordination" and "cooperation" are frequently used interchangeably in the context of traffic control mechanisms, and their meanings might vary according to particular application and environment context. However, in the field of AI and transportation systems, they often refer to various approaches for optimizing traffic flows [66]:

5.4.1 Coordination of Traffic Control Mechanisms

Coordination involves organizing and synchronizing individual efforts of controllers towards the ultimate goal of optimizing traffic flow throughout the whole transport network [66]. Coordination of traffic control mechanisms focuses on harmonizing the collection of data from several sources (such as sensors, cameras and GPS devices), analyses and evaluates them, and then optimizes traffic flows by adjusting signals, routes, or other infrastructure elements.

5.4.2 Cooperation of Traffic Control Mechanisms

The goal of cooperation between traffic control mechanisms is to improve collaboration and coordination among many entities of the transportation system such as vehicles, infrastructure and authorities. To maximize traffic flow, improve safety, and reduce congestion, vehicles interact with each other and with the infrastructure (V2V and V2I communication) [67].

5.4.3 Coordinative vs. Cooperative Traffic Control Mechanisms

Table 1.4 presents a comparison between cooperative and coordinative traffic control management mechanisms in terms of (1) – context meaning, (2) formal and informal relationships, (3) interchangeably, (4) objectives and (5) traffic flow optimization strategy. Both coordination and cooperation are essential for the success of traffic control mechanisms, and combining both strategies enables controllers to produce more benefits and ensure effective traffic light management.

Table 1.4 Coordinative vs. cooperative traffic control management

Coordinative Control Management	Cooperative Control Management
It relies on the collective reasoning of agents to optimize traffic flow.	Each agent contributes to traffic flow global optimization by helping each other.
It relies on informal collaboration.	It relies on formal relationships.
It requires collaboration with different agents.	It requires coordinating with different agents or groups of agents.
It relies on flexibility and adaptability.	It relies on efficiency and productivity.
Traffic flow optimization is achieved via hierarchical structures and agents	Traffic flow optimization is achieved via mutual agreement and trust.

5.5 Existing AI Approaches in Smart Traffic Light System

Smart traffic light systems use a variety of artificial intelligence approaches to optimize traffic flow, reduce congestion, and improve overall efficiency. Here are the main AI approaches commonly used in STLS.

5.5.1 Multi-Agent Based Traffic Flow System

Multi-Agent System (MAS) enables decentralized control by assigning intelligent agents to individual intersections, allowing them to make local decisions and coordinate with neighboring agents. Recent research has demonstrated the efficacy of MAS in managing traffic flow, particularly through the application of Multi-Agent Reinforcement Learning (MARL) [68].

5.5.2 Machine Learning Approaches and Optimization Algorithms

Machine learning techniques, such as Support Vector Machine (SVM) and neural Networks (NN), analyze historical traffic data to predict traffic patterns and optimize signal schedules, while optimization algorithms such as Particle Swarm Intelligence (PSO) and Genetic Algorithms (GA) fine-tune these timings to reduce congestion [69].

5.5.3 Deep Learning Approaches

Deep learning models, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), analyze traffic data to detect vehicles and pedestrians, enabling precise control of traffic light timing [70].

5.5.4 Hybrid Approaches

Hybrid approaches combine several artificial intelligence techniques, such as integrating sensor data with machine learning algorithms or combining reinforcement learning with optimization methods. This enables STLS to provide more robust control of traffic lights, adapt dynamically to changing traffic conditions, improve safety, reduce travel times and minimize environmental impact [71].

Table 1.5 summarizes the comparison of current smart traffic management approaches in terms of principles, security, flexibility, and reliability.

Table 1.5 Comparison of AI approaches in STLS.

Approaches	Principals	Security	Flexibility	Reliability
MA-Based	Decentralized, adaptive and scalable traffic management.	Protects the system from cyberattacks.	Adapts to different traffic conditions.	smooth traffic flow in failures
ML-Based	Historical and real-time data analysis for traffic patterns recognition.	Data poisoning and hostile attacks.	Adaptable to changing traffic conditions.	Performance depends heavily on data quality.
DL-Based	Neural networks for traffic prediction and optimization.	Lack of robust data protection mechanisms.	Learn complex patterns.	Data intensive computing.
Hybrid	Combining multiple techniques to enhance performance.	Complex safety management	Balancing control and adaptability	Depending on individual components

6. Combine Swarm Optimization with AI in STLS

The combination of swarm optimization and AI in STLS represents a significant advance in the optimization of traffic flow management. Both techniques provide effective approaches for managing complex, dynamic, multi-agent systems such as urban traffic networks. This integration is continually advancing, leading to smarter, more adaptive and efficient traffic systems for smart cities.

6.1 Principles of Combining of Swarm Optimization and AI

The combination of swarm optimization and artificial intelligence in STLS revolves around several key principles. These principles focus on data collection, real-time processing, optimization algorithms and intelligent decision-making to create a responsive and efficient traffic control environment.

6.1.1 Integration of Sensors and Cameras for Data Collection

Sensors (such as inductive loop detectors, infrared sensors and vehicle GPS data) and cameras are vital components of STLS system, enabling the collection of real-time traffic data such as vehicle numbers, speed, density and incidents [72]. The main aim of sensors is to collect continuous, high-quality data for well-informed decision-making. However, AI techniques such as computer vision can analyze video streams to detect traffic conditions, while swarm optimization leverages this data to make real-time adjustments to traffic lights.

6.1.2 Integration of Edge Computing for Data Processing

Edge computing enables data to be processed instantly as its source, such as traffic light control systems, which reduces latency compared to cloud-based solutions. With processing power at the edge, traffic conditions can be analyzed quickly, leading to necessary optimizations [73]. The objective is to minimize decision-making latency and reduce dependence on central systems. Sensing devices with edge computing can deploy lightweight artificial intelligence while swarm optimization algorithms enable real-time adjustments based on changing traffic data.

6.1.3 Swarm Optimization for Signal Timing

Swarm optimization algorithms such as Particle Swarm Optimization (PSO) or Ant Colony Optimization (ACO) are used to determine optimal traffic light schedules. These algorithms simulate the behavior of natural swarms to explore the search space and find optimal or near-optimal timing configurations based on traffic data [49, 50]. The objective is to reduce traffic congestion by optimizing traffic light timings for a set of interconnected intersections. Since Swarm optimization performs as a decentralized method for adapting traffic lights in real-time to increased traffic flows, thereby minimizing waiting times.

6.1.4 Machine Learning and Deep Learning for Traffic Flow Prediction

Machine learning (ML) and Deep Learning (DL) models can forecast traffic flow, detect congestion hotspots and assess the impact of specific traffic light schedules by analyzing historical and real-time data. These models are capable of predicting future traffic conditions at intersections and along specific routes [69, 70]. The purpose of ML and DL models is to predict traffic congestion and adapt traffic light schedules accordingly. However, AI algorithms trained on large datasets can predict the number of vehicles approaching intersections, enabling proactive adjustments to traffic light timing to avoid bottlenecks.

6.1.5 Swarm Algorithms for Traffic Flow Optimization

Swarm algorithms are not only work on signal synchronization, but also in optimizing traffic flow across a network of intersections. By viewing each traffic light as an agent in a swarm, the system finds collective solutions to balance traffic flow, reduce congestion and improve vehicle throughput [74]. The goal is to manage traffic flows across multiple intersections, coordinating green lights to minimize delays. However, these algorithms dynamically adjust the phases and durations of traffic lights, facilitating a more even distribution of traffic across available routes.

6.1.6 Intelligent Agents for Dynamic Routing and Adaptation

AI-powered intelligent agents operate within the traffic network to dynamically reroute vehicles and adapt signal schedules according to current traffic conditions. This multi-agent system ensures that each intersection reacts both locally and in coordination with the entire network to optimize traffic flow [75]. The goal is to ensure adaptive control of traffic lights, ensuring that intersections work together to alleviate congestion. However, intelligent agents can learn and adapt autonomously to changing traffic conditions, enabling more efficient utilization of road capacity.

6.1.7 Emergency and Incident Management

Traffic systems must be able to handle unexpected events like accidents, road closures, or emergency vehicle priority. AI can predict the impact of such incidents, and swarm optimization can adjust the traffic signal timings in response. Additionally, systems can prioritize emergency vehicles, ensuring they can navigate traffic smoothly during critical situations [76]. The goal is to ensure fast and efficient routing for emergency vehicles while minimizing the impact on normal traffic. However, AI algorithms detect emergencies or incidents in real-time, and swarm optimization algorithms reroute traffic or adjust lights to accommodate these events.

6.2 Benefits of Combining of Swarm Optimization with AI

By combining swarm optimization and AI, STLS can achieve reduced congestion, heightened efficiency, real-time adaptability and scalable solutions for future needs, making it well-suited for modern, fast-moving urban cities. The key benefits include:

- **Reduced congestion and improved efficiency:** The combination of Swarm optimization and AI facilitates smoother traffic flow management by optimizing signal timings at intersections. Swarm algorithms adjust signals according to traffic patterns, while AI predicts upcoming congestion, enabling timely adjustments. This leads to less vehicle idling, fewer stops and starts, and shortened journey times.
- **Adaptive and real-time response:** AI-powered traffic systems continuously analyze real-time traffic data and adapt to fluctuating conditions such as accidents, traffic peaks or road closures. Swarm optimization ensures that each traffic light to make rapid, localized decisions, enabling immediate reactions to changing conditions without relying on a central controller.
- **Scalability and Flexibility:** The combination of swarm optimization and AI offer great scalability, allowing the system to grow alongside cities expand. Swarm algorithms enable decentralized control, meaning that integrating new intersections or signals can occur seamlessly without overhauling the entire system. AI models continue to learn and adapt to the growing complexity of traffic, ensuring long-term efficiency.

7. Conclusion

This chapter details the fundamental components of smart traffic light systems, which collect, process traffic data, and ensure accurate decision-making. The chapter also emphasized the significance of artificial intelligence, swarm optimization and computer vision techniques with a focus on four main approaches, which give a solution to properly manage huge traffic data and detect congestions. Each approach has its advantages and drawbacks.

In the next chapter, we will review existing traffic flow management techniques, mitigating waiting time and occupation rates concerns, and compare existing techniques under well-defined criteria and fault tolerance detection.

Chapter 2: A Survey on Smart Traffic Light Systems: Models and Approaches

Content

1. Introduction 45
2. Main Works in the STLS Area 46
3. STLS Approaches for a Single Intersection Model 50
   3.1 Heuristics based Approaches 50
   3.2 Sensor-based Approaches 56
   3.3 Hybrid Approaches 59
   3.4 Synthesis and Comparison 62
4. STLS Approaches for a Set of Intersections 63
   4.1 Heuristics based Approaches 63
   4.2 Sensor-based Approaches 70
   4.3 Hybrid Approaches 71
   4.4 Synthesis and Comparisons 73
5. Global Smart Traffic Light System and Internet of Things 73
6. STLS Simulation Methods 75
7. Conclusion 75

1. Introduction

Nowadays, we live in an era of smart cities that assist residents with intelligent transport and mobility solutions. However, managing high traffic volumes has been challenging, though progress in this area is inevitable. Vehicles are becoming safer, but the road environment is becoming more complex, mainly due to the rapid increase in the number of vehicles and their consequences. As a result, road traffic conditions have become complicated and chaotic. An adequate traffic network can offer a large variety of benefits, especially in the development of the socio-economic sector, including better traffic flow (With all possible resulting advantages) and a more agreeable environment, which in turn can strengthen economic performance.

An evaluative, adaptive and continuous traffic management system is needed to deal with this unstable problem and to meet the growing demand for vehicle flows. It is therefore a question of how to design a Smart Traffic Light System (STLS) meeting requirements of emerging smart cities concept or alleviating the heavy load of traditional large city crossroads. All this to take best advantage of the traffic circulation at intersections; knowing that many important cities are facing socio-economic problems caused by traffic congestion. The most striking case is the capital Algiers where traffic congestion causes citizens to lose valuable time in getting to their jobs.

Research into effective traffic management and situation emergencies in urban roads have been widely exhibited. This chapter classifies the STLS approaches according to the number of intersections in two models, with a set of intersections or a single intersection, with details of the control parameters for an intersection and the different solution architectures for a set of intersections. Figure 2.1 classifies different approaches of STLSs. Three models of approaches have been distinguished: optimization approach [11, 14, 16, 17] or data collected by the sensors [7, 20] while others use hybrid approach combining sensors and optimization techniques [8, 9, 23]. Several criteria are considered to summarize the principle of each approach, namely type of traffic management system, type of traffic parameters, priority, coordination and fault-tolerance. It is our aim to design a dynamic and adaptive traffic light system that can reduce congestion and manage emergency situations based on priority across multiple vehicles in smart cities and overcome the limitations of existing approaches.

The remainder of this chapter is structured as follows: Section 2 presents related work in the STLS domain while Section 3 details the different STLS approaches for a single intersection model. STLS approaches for a set of intersections are presented in Section 4. Section 5 describes the use of the Internet of Things particularly in the STLS context. Section 6 gives an overview of the simulation tools used. Finally, Section 7 concludes the chapter.

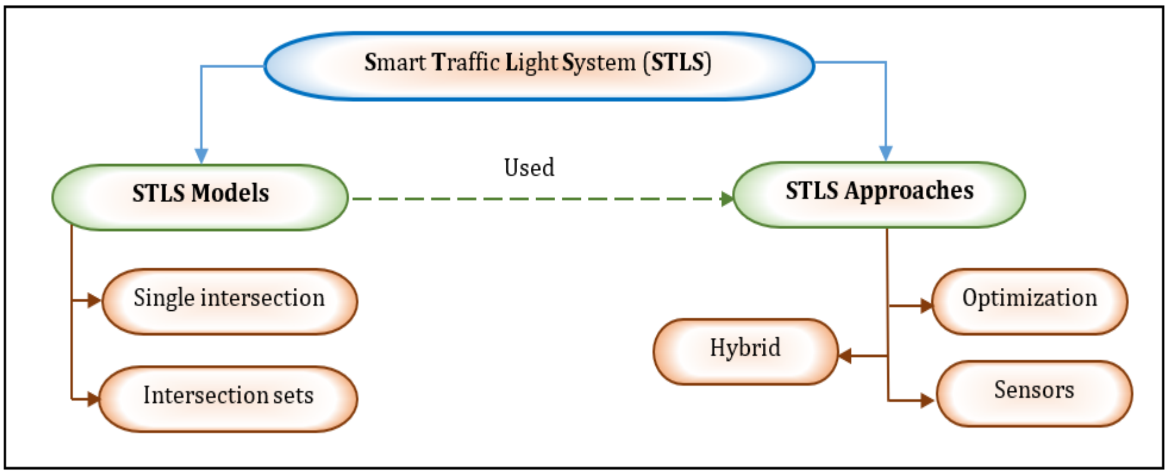


Figure 2.1 STLS models and approaches.

2. Main Works in the STLS Area

Much work has been done over the last decade to elaborate plausible approaches to the difficult issue of vehicle traffic at intersections. This problem has further attracted the attention of researchers with respect to the emergence of the intelligent city model. Although the design of STLSs is based on different approaches such as the use of sensors [7, 20] optimization methods [11, 14, 16, 17] and the combination of the two previous ones [8, 9, 23], these STLSs are still subject to improvements to comply with the requirements of smart cities.

The result of some important research studies that have addressed the problem of urban traffic at intersections, with respect to their objectives, advantages and disadvantages, are presented below.

Joo et *al.* proposed in [14] a system for coordinating traffic lights at an isolated intersection aiming to maximize the amount of vehicular traffic and minimize the standard deviation of queue size. Their system uses the Q-Learning (QL) method to ensure the balancing of signals between roads. The main strength of this system is its flexible structure and the adaptation of changes in the existing junction network. Their motivation for using one of the artificial intelligence techniques, which is reinforcement learning, comes down to the high complicity of the mathematical models [77, 78] due to the large amounts of data manipulated, and that the traffic environment has stochastic problems.

Lamghari et *al.* [16] developed a modular design approach for an urban signal intersection. They used Temporal Synchronized Petri Networks (TSPN) that allow for adaptive traffic signal management. This decreases the modelling complexity and can be easily adapted to a group of intersections. In addition, the diversity of phases in the intersection controller allows the implementation of several traffic signal control strategies.

Rasheed et *al.* [17] proposed the advanced technique called Deep Q-network (DQN) [79], based on deep learning and reinforcement learning (RL or Q-Learning) [80]. The proposed technique controls traffic signals to solve the traffic congestion problem. The authors extended traditional single-agent DQN to a multi-agent DQN (MADQN). They benefited from the advantages of traditional Q-Learning namely the reduction of the representation dimensions of the operating environment [81] with a lower traffic volume. Traffic disruption scenarios are used to increase dimensionality. Their study is focused on an urban zone in the city of Sunway in Malaysia.

Omar et *al.* [8] focused their work on two main aspects: the first one is the traffic flow theory, using the flow of arriving vehicles to minimize the waiting time of vehicles between traffic light cycles. In the second one, they try to predict the vehicular traffic flow level by developing a neural network using traffic information and machine learning techniques. Their aim is to design a reduced-scale, least-cost solution for studying, controlling and anticipating traffic flows. The combination of unifying parameters, such as the number of vehicles in a particular space and the speed or intensity of the cars, which are not indicative of real traffic conditions, allows for the correct interpretation of traffic flow. The use of modern sensors leads to huge quantities of information and with the help of machine learning; the system can create models to predict future traffic.

Segredo et *al.* [11] proposed formal traffic signal planning to solve the traffic congestion problem of a large urban area. They modeled more realistic and less difficult scenarios and suggested advanced and more efficacious methods than those employed in previous studies [28, 82]. This study theme concentrates on the problem of establishing a traffic signal plan for an urban area by setting the duration of every phase and the phasing sequence of all junctions in that area. This formulation is based on the one described in [74, 83], but it also takes into account the time difference between the intersections. The authors use a multi objective evolutionary algorithm (MOEA) built on diversity, which has demonstrated promising results for solving single-objective problems [84, 85], taking advantage of the main benefits of MOEA, namely the good diversity of individuals and the premature convergence [86].

Kumar et *al.* [7] proposed a dynamic algorithm for traffic signal controller synchronization in a simple four-road intersection, takes as input parameters queue length, entry rate and exit rate, to estimate the green time allocated to each road. The motivations of this work are to ensure efficient distribution of green time based on congestion for traffic signals independently of their traffic status. They also proposed experimental comparisons between dynamic and static methods based on some efficiency parameters namely, traffic flow at the intersection, vehicle waiting time and queue length on roads.

Frank et *al.* [23] developed an IoT-based traffic signal control system capable of measuring the actual volume of traffic on all lanes of the junction by using real-time image and video handling methods. Their system provides the option of user control of the traffic light via a software application. Their motivation is to resolve the situation of road congestion, which is the primary reason for slow moving vehicles, increased waiting times and collisions.

Hosur et *al.* [20] proposed a sensor-based system with threshold distance definition and using IoT technology. Therefore, when the sensor detects a car at a certain distance and other ways are vacant, it turns on the green signal or else turns on the red signal. Their objective is to overcome the problem of displaying the green signal even if there are no cars on the lane, in order to reduce the waste of energy consumption, especially during off-peak hours.

Lin et *al.* [9] presented a vehicle registration plate recognition model based on flows of streaming data captured from various sensors in real time. For this model, an adaptation mechanism based on Gaussian combination is applied on the collected information to generate traffic signal plans, rendering the traffic management procedure of Tainan City in Taiwan flexible and fast.

Raj et *al.* [24] used a completely different approach by aiming to manage priority vehicles from video and audio processing. They converted the audio to spectrograms and split videos into frames that they processed using multiple neural networks with different depths, which are subsequently used for detecting priority vehicles. Thus, they obtained a maximum accuracy of 74.6%, which is still much lower than the results obtained using computer vision.

In [25], the authors introduced a new approach for emergency vehicles priority. The study was intended to give green time for regular vehicles and clear emergency vehicles. The proposed model has reduced the average waiting time of vehicles by 73.23% % surpassing many well-known existing systems.

In order to control emergency vehicles in the road, the authors in [26] suggested both on-demand signal timing and linear programming. Linear programming is used to find the shortest time in each phase after passing emergency vehicles. The study has led to an optimization of travel time by 62.85% outperforming the fixed-time control method.

Oliveira et *al.* [87] developed a centralized traffic signal control system based on a wireless communication network. The system implements direct control routines over network traffic lights to control abnormal and emergency events, such as closing roads due to accidents or public events. Also, the system implements safety routines to report the operating lamps' status of the traffic signal to the central management system.

Choudhary et *al.* [88] introduced a novel thread-based virtual traffic light system for managing traffic flow. The thread concept is a secure, scalable, and low-power IPv6-based network mesh platform for IoT devices. However, this work required vehicles with further hardware to build a thread-based mesh network (VANET).

Younes et *al.* [89] proposed an efficient traffic light scheduling for a road intersection based on traffic and safety parameters. The proposed approach focuses on the priorities and time of scheduling by considering more traffic information which significantly increases the throughput of the transportation network.

Zhu et *al.* [90] proposed an approach for Adaptive Traffic Light Control (ATLC) based on Broad Reinforcement Learning (BRL). The main goal of the proposed approach is to select relevant state information that will be considered for decision-making by keeping them using the Long Short-Term Memory (LSTM) network.

A federated learning for smart traffic flow management is proposed in [91]. The approach is very interesting and consists of achieving minimal performance with improvement in terms of communication cost. A federated learning approach was implemented to detect lamppost fault detection in smart city.

Authors in [92] suggest a novel reinforcement Q-learning-based deep learning to optimize traffic signal control at an isolated intersection. A new reward function was applied to control traffic signals. The use of deep reinforcement Q-learning with reward function achieves better performance.

In [93], an ontology approach has been proposed for traffic light optimization. An ontology-based driving simulation approach was detailed combining evolutionary-based optimization algorithms with ontology-based driving behavior simulation. The obtained results are significant only on some limited simulation scenarios.

Some other works [94, 95] have also applied well-known deep-learning models to detect emergency vehicles. For instance, popular frameworks such as CNN was applied in [94] giving a precision rate of 91.3 % for the best overall approach which was characterized by a YOLOV3 architecture. A similar approach was applied in [95] since context-aware reinforcement learning was applied giving a recovering rate of 80%.

3. STLS Approaches for a Single Intersection Model

Various approaches have been proposed for STLS. They are generally classified into two principal categories: models for a unique intersection and models for a group of intersections. For both models, the approaches used were: heuristic approaches [14, 16], sensor-based approaches [7, 20- 22] and hybrid approaches [8, 23] that combine the first two ones.

3.1 Heuristics based Approaches

Joo et *al.* [14] proposed the use a new QL reinforcement-learning approach to optimize traffic light signals, in which the main parameters are the flow rate and the typical difference between the waiting line sizes. This algorithm determines the action in driving directions to increase the quantity of cars travelling through a junction in a time period and to maintain a balance among the road ways and minimize the time delay of the traffic. The parameters of this algorithm, such as the length of the green time, are fixed, but the direction with the most vehicles will be selected along with another direction to provide maximum vehicle traffic flow. Thus, this algorithm can dynamically control eight different phases for a four-way intersection.

The reinforcement learning (Q-Learning) algorithm has been improved with temporal learning processes [96], and employs a process of test and failure to investigate the stochastic and complex environment and choose the best behavior depending on its experimentation [97]. Q-Learning has the principle of states (*situation of the environment*), action (*behavior*), and recompense (*experience*), as shown in equation (2.1).

$$S_t \overset{a_t}{\to} S_{t+1} \tag{2.1}$$

An action (a_t) is carried out in a state (S_t), it enables to advance to the next state (S_{t+1}). Q-Learning is a method that uses a table named « Q-table ». The lines are the states S_t and the columns are the actions a_t, their elements initialized by 0 will be refreshed with the precedent values of $(Q\ (s_t, a_t))$ for the actual state (S_t), the action (a_t), the recompense (r_{t+1}) and the highest values $(max_a\ Q\ (s_{t+1},\ a_{t+1}))$ in the new state (S_{t+1}), also utilizing the learning ratio (η) and the actualization factor (γ), as shown in equation 2.2.

$$Q\ (s_t, a_t) \leftarrow Q\ (s_t, a_t) + \eta \cdot [r_{t+1} + \gamma \cdot max_a\ Q\ (s_{t+1},\ a_{t+1}) - Q\ (s_t, a_t)] \quad (2.2)$$

The proposed approach uses a set of three actions to manage eight phases of a four-way intersection. The direction with the greater number of cars represents the actual state, and the action that provides the highest flow of cars will be will be selected from the sets of actions that include the direction of the current state.

The unit of throughput (tp) to be maximized is determined as the number of cars traversing a junction each hour. For queue length standard deviation (d_{ql}), it must be lower or similar to the threshold specified (φ). The green light duration (l_{signal}) is constant (c), the time duration from the termination of the green light in one way to the starting of the next green light (t_{inter}) in the identical way is lower than the predefined threshold (φ').

To calculate the recompense (r_t) and minimize the waiting time at the intersection, the recompense function as shown in equations (2.3) and (2.4), is established using two metrics, the typical deviation (d_{ql}) of waiting line lengths and the throughput (tp).

$$f(t) = \alpha \cdot (d_{ql}) + (1 - \alpha) \cdot (\tau^{tp}) \quad (2.3)$$

$$r_t = log_\delta\big(f(t)\big) \quad (2.4)$$

Such as:

– τ^{tp} is a simple function exponential form, note that the larger the throughput value (tp) the smaller the value τ^{tp}.

– α is the weight coefficient that relies on the traffic arrived in each hour, $0 \leq \alpha \leq 1$ takes a sigmoid function form, the closer it is to 1, the more cars coming in.

– The minimum value (t) leads to a maximum recompense (r_t).

– δ is the base of the log formula and their value is comprised from 0 to 1, for this work δ set to 0.5.

This approach uses different agents to control traffic flows. The environmental agent "*Intersection*" collects information about the environment and sends it to the "*Controller*" agent to take the best control action depending on the available traffic data. The interaction of the "*Intersection*" environment with the "*Controller*" agent is as follows:

The intersection agent:

- Sends the lengths of waiting lines (ql) and the flow rate (tp) of all way in a traffic junction to the controller agent.

The controller agent:

- Calculates the recompense (r_t), from equation (2.4) when passing from state (S_{t-1}) to state (S_t).
- Update the table Q.
- Defines the action (a_t) with the highest recompense.
- Transmits to the intersection agent, the lane (dc) indicated in this action.

To validate this approach in regards of typical deviation and waiting line length, they compared the proposed model against two additional QL approaches. The first approach is founded on the green light sequence in the way [98], a fixed time enhanced traffic signal, called extension traffic signal (E-TS). The other approach controls traffic lights via a clustered QL scheme, called clustered traffic signal (C-TS) [99]. Their experimental results revealed that the adopted solution shows improvements in relation to the waiting line length standard deviation and the reduction of queuing line length and delay time.

In [14], the objective or recompense function, allows signals to be placed only on high traffic sides, which can produce poor performance, as it did, it does not take into account the waiting times for car users on the opposite road side. For a possible improvement, the waiting time can be integrated into the objective function.

Lamghari et *al.* [16] suggested a timed and synchronized Petri net-based approach to adaptive traffic light management and the command scheme is divided among two interacting system elements. The master (*controller*) determines and chooses the following phase and the duration of their green signal while the slaves (*TSPN sub-models*) monitor signal lights display, transition phases and flowing traffic. To treat and control traffic signals, two methods have been used [78]. The first one enables the execution of optimal signal plans to improve the functionality of the system related to traffic flow and vehicle delays at the intersection [100]. A second method is also used to define the implementation of the signal control logic [101]. For this solution, the authors adapt both functionalities to lead to adaptive traffic signal timing plans.

This traffic light system consists of a standardized junction with four lanes (North (N), South (S), East (E) and West (W)) and two transitions for each lane; so, eight transitions are envisaged (Ns, Nl, Ss, Sl, Es, El, Ws and Wl.) and each transition attached to three phases, in total there are transitions in twelve phases.

Equation (2.5) below expresses the method of calculating the green light time required for all vehicles allowed leaving the intersection. This is related to the number of cars to be exited during the decision making for selecting the next phase, assuming that the duration of the orange signal is included in the duration of the red signal.

$$G = (N - 1) \times h + t_a \tag{2.5}$$

Such as:

- N is the cars number,
- h is the time taken to move a one vehicle,
- t_a is the duration taken by the first vehicle to exit the line stop.

A class of the mathematical Petri net model, called Timed Synchronized Petri Net (TSPN) and represented by a bipartite graph, is used to implement the vehicle flows and traffic signals.

The timed and synchronized Petri net module consists of eight TSPN sub-models, one for each movement. The TSPN Sub-model is shown in Figure 2.2, representing the traffic flow and control model for a movement $\langle ij \rangle, (i \in \{N, S, E, O\}\ et\ j \in \{s,\ l\})$. Each movement consists of five places and five transitions (see Table 2.1.a), $\langle ij \rangle$ denotes the eight movements and the outer events specified as the master's decision are shown in Table 2.1.b.

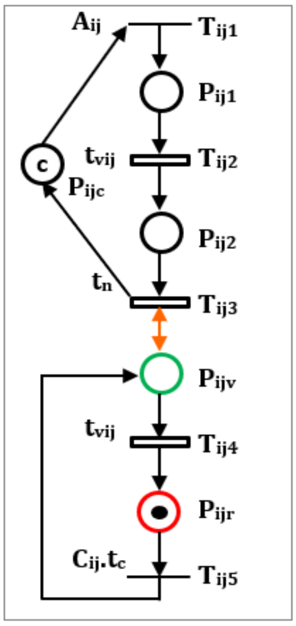


Figure 2.2 TSPN sub-model for movement (***ij***) [16].

Table 2.1 (a) Component definition of the TSPN Sub-model in Figure 2.2.
(b) Activation conditions for the decision events in Figure 2.2. [16].

a)	
Places	**Signification**
P_{ij1}	Entry of the vehicle into the way ‹ ij ›.
P_{ij2}	Cars to be moved out of the way ‹ ij ›.
P_{ijv}	Green light.
P_{ijr}	Red light.
P_{ijc}	Way capacity in movement ‹ ij ›.
Transitions	**Signification**
T_{ij1}	Sensing a car in way ‹ ij ›.
T_{ij2}	Next decision (the previous phase's green hour has elapsed).
T_{ij3}	The leaving of a car from way ‹ ij ›.
T_{ij4}	Change of light green to red.
T_{ij5}	Change of light red to green.
External events	**Significant**
A_{ij}	Sensor detection of car in the direction ‹ ij ›.
E_{ij}	The decision of the controller for the movement ‹ ij ›.
Time delays	**Significant**
t_n	Time required a car to depart from the stop line.
t_{vij}	Duration of the green light for the way ‹ ij ›.
t_c	Time for the last vehicle to cross the junction (red security).
b)	
Events	**Activation requirements**
C_{Ns}	$E_1 \cup E_5 \cup E_{11}$
C_{Nl}	$E_4 \cup E_6 \cup E_{11}$
C_{Ss}	$E_1 \cup E_7 \cup E_{12}$
C_{Sl}	$E_3 \cup E_7 \cup E_{10}$
C_{Es}	$E_2 \cup E_6 \cup E_{10}$
C_{El}	$E_4 \cup E_6 \cup E_{11}$
C_{Ws}	$E_2 \cup E_8 \cup E_9$
C_{Wl}	$E_4 \cup E_8 \cup E_{12}$

The movement way ‹ ij › is the space between the sensor (transition T_{ij1}) and the stop line (transition T_{ij3}), with a capacity c (number of tokens at P_{ijc}). The sum of the tokens in the two locations P_{ij1} and P_{ij2} represents the traffic flow. The two locations P_{ijv} (green light) and P_{ijr} (red light) represent traffic lights.

Two functions for controlling movement ‹ ij › are used:

First:

– If the sensor A_{ij} recognises the entry of a car, then activates the transition T_{ij1} and a token is moved from P_{ijc} to P_{ij1}.

Second:

- If a decision (E_{ij}) is given to movement ‹ ij › then
  - Activate the timed transition T_{ij2} that authorizes vehicles to be evacuated, with green temp t_{vij} and the tokens from P_{ij1} are moved to P_{ij2} .
  - Start the timed transition T_{ij3} which allows a vehicle to depart with a time t_n and the tokens from P_{ij2} are moved to P_{ijv}.
  - Set the value *(1)* to the external event C_{ij}.
  - Trigger the timed transition T_{ij5} and put a token at P_{ijv}, the green time t_{vij} is the duration needed for the vehicles to exit the line stop (release the place P_{ij2}), a delay t_c is added to guarantee that the latter car departs the area of conflict prior to the new phase.
  - Launch the transition T_{ij4} and the traffic lights switch to red. The master restarts to decide which movements will take next green light.

Two types of communication between the master and a TSPN sub model may be observed: The first one, depending on the decision and triggering times calculated by the master, which transmits a message to the TSPN indicating that a synchronized and/or timed transition must be triggered. The second communication function is from the TSPN to the master to update its database. This message can contain information from the sensors, the arrival time and number of cars at the junction, the triggering of transitions, etc.

Depending on TSPN state, the master component makes the decision according to parameters to be optimized in two steps: firstly, a movement ‹ ij › is chosen among the eight movements, secondly the master chooses the phase between three phases which does not conflict with the movement ‹ ij › . A two-strategy adaptive control algorithm is proposed for system performance optimization and green light assignment. The first strategy is based on the time of arrival and the number of waiting cars at the line of stop, and the second strategy uses only the arrival time of the vehicles. Experiments with the two adaptive control strategies showed that the arrival hour and number of cars in queue strategy provides relevant and efficient results.

In this paper [16], the objective function uses the parameters: arrival hour or/and the number of waiting cars at the stop line, for the choice of the movement to which the green light is not exactly defined.

3.2 Sensor-based Approaches

Kumar et *al.* [7] modeled a simple four-road traffic intersection: $\{R1, R2, R3, R4\}$. Each road R_i has a traffic light T_i and $2 \times k$ lanes: k lanes forward and k lanes backward (with $k = 3$). Between the switching of traffic lights from T_i to T_j, there will be a constant short time interval called pause time p. The time t_i allocated to a road R_i, when it is selected to be green. The goal of this work is to design an optimization approach that allocates the green time t_i to T_i in order to maximize the traffic flow at the junction and reduce the waiting time of vehicles and the length of road queues. In this work, the authors use green time management measures allocated to the road, which will be specified by the interval $[t_{min}, t_{max}]$, in order to guarantee the absence of cars in the queue. The road has a minimum green time and if there is a long queue (congestion case), the green time allocated does not exceed a certain value. Only one cycle model is used, each cycle contains four phases, and the phases will be selected alternately. A route is selected at each phase without repetition; the order of the phases at each cycle can be changed to be dynamic.

The traffic signal algorithm proposed in this work consists of three phases which are: (1) - The collection of real data from the intersection area. This implies the precise tracking of vehicles in each lane. It is realized for a particular range or distance from the intersection. (2) - Data collected from all lanes in the first phase will be processed in this phase. This processing consists of simplifying and adapting to the various input and output parameters needed to make the best use of the dynamic traffic signal algorithm. These parameters include queue length, entry rate and exit rate. (3) - A decision-making process consisting of selecting which traffic light will be green and how much time will be allocated to it. The decision-making system also keeps track of the system performance metrics in order to improve its efficiency.

In order to approve this approach, the results of the dynamic and static approach were compared with respect to traffic constraints. For this comparison, the green time is fixed and the order of traffic light selection is also fixed. The experimental results show the similarity of the efficiency of the dynamic approach and the static one when the traffic is uniformly distributed with a small improvement of the dynamic algorithm. The authors find that the efficiency of the dynamic approach is greater than the static one when the traffic is unequal. In addition, the dynamic algorithm is more flexible to be adapted to changes in traffic influx.

The delimitation of the green time allocated to a way by t_{min} and t_{max} can lead to a loss of time for a way with zero vehicles. Conversely, it can also lead to insufficient time allocated to a multi-vehicle lane. To solve this delimitation problem, one can set the tmin value to zero for an empty lane, and change the tmax value depending on the queue size of all lanes. Another disadvantage of this method is that the selection of the green light is based on the waiting time, which reflects the interval of the red time and does not take into the waiting time of each vehicle in the road. Thus, only one cycle model is used and the selection of phases is done per cycle, so it is better to use several cycle models and test the selection after each phase.

Rida et *al.* [21] proposed an adaptive approach to regulate traffic signals for an individual junction that takes into account a variety of circulation factors including volume of traffic and duration of delay, giving priority to the shortest queue. Magnetic sensors are used to collect the traffic data set and transmit it to the controller responsible for traffic management. The controller calculates the size of the lanes in every direction and their average waiting time in order to manage the flow of traffic.

The isolated intersection consists of four ways (*N, S, E, W*), each of them containing two directions (turn left and straight ahead). All cars are permitted right turns at all times with no traffic restrictions. Two magnetic captors are installed in each lane: one is situated behind the signal light to track vehicle departures and the other is placed at a variable range, which is related to the maximum green light time, from the first sensor to check the arrival of cars. Each traffic light manager establishes a cycle time which is a series of phases with a required green light time for two simultaneous movements. The adaptive control algorithm uses information obtained from the various sensors, and determines which phase will be executed with the calculation of their green time.

The authors in [21] introduced an adaptive traffic control system that determines the phase sequence and length of the green signal based on delay duration and the size of the queue for each road. The disadvantage is the increase in congestion and waiting time in the longest queues, which can block the traffic flow as priority is always given to the shortest queue. The proposed method does not detail the emergency mode, i.e., the passage of emergency vehicles.

Hosur et *al.* [20] developed a sensor-based solution from a distance threshold and the use of IoT technology to manage the display of green lights and decrease energy consumption (e.g., during off-peak hours) if they are higher than what is actually needed. Therefore, the system turns on the green light when a sensor detects a car at a certain distance and not all other lanes are occupied, else it turns on the red light. The authors use Raspberry Pi computing components evaluating traffic density with infrared ultrasonic sensors: these sensors measure distance using ultrasonic waves. Their approach dynamically manages traffic at junctions based on vehicle density; the authors used IoT technology where objects refer to embedded devices and sensors. The data collected can be treated and evaluated by means of internet sharing.

The traffic control system considers the traffic lights and sensors as physical layers related to the Raspberry Pi in the layer data link that acts as a manager device. Internet is in the layer network and in the layer application will have a portal web by means which we can monitor the IoT devices at distance. The details and operation of the application and simulation tests were not reported by the authors, which may induce a weakness in the applicability of the proposed system.

Firdous, et *al.* [22] proposed a prototype of a functional model for an intelligent traffic signal that adjusts its synchrony automatically according to the traffic flow by exploiting solar energy from a solar panel. The system manages a four-way intersection using four infrared digital captors, one for each way, and eight LEDs to serve as signal lights for each way. All the infrared sensors are linked to an Arduino Uno controller that reads information from the IR sensors. The system traffic lights are engineered with LEDs and every light consists of two LEDs (red, green) for every way. There are two modes of system operation, normal and high traffic as follows:

- Normal traffic means when all lanes have almost the same traffic density; that is, four ways named 1, 2, 3 and 4 have to be managed. The control system begins to vacate all roadways, starting with way 1, so the green light is on for way 1, and a red light on for the other roads. At the end of 2 seconds, the green LED is lighted for way 2, to release the flow on way 2, then way 3 and 4. This cycle runs continually until one of the roads has significant traffic.
- The traffic is high if one lane is denser as compared to the others. Assuming that route 1 has important traffic, this system tries to release first traffic on Highway 1. In effect, the green LED of route 1 set to on, and red LED set to off. Other routes, the green LED set to off and the red LED set to on. This mode continues unless the flow on route 1 has changed from higher to regular flow.

In [22], if hard objects, e.g., pedestrian crosses over the sensor block, this would affect the IR sensors which detects the event and sends erroneous data. In addition, Arduino Uno works as a central console (if fails the whole system crashes).

3.3 Hybrid Approaches

Omar et *al.* [8] provided a hybrid approach using two methods; the first one is related to traffic flow theory and the second one is dedicated to traffic flow prediction using a deep neural network based on traffic data harvested and learning machines method, to perform traffic signal optimization and monitoring.

In addition, the entry and exit rates of vehicles in each lane of an intersection were used to calculate the green time for four phases (North, East, South and West). The remaining vehicle exit rate is set to 2.04 vehicles per second for all lanes and measured by manually counting the number of vehicles exiting a lane. The input rate was considered an important factor calculated by aggregating data such as car density and vehicle frequency, which are updated at each light cycle.

The authors develop a traffic management system with USB cameras to track vehicles and collect traffic data (*number of cars, frequency and direction*), and send it to the Raspberry Pi 3 microcontroller [102] which in turn transmits it to the ThingSpeak online cloud service for further processing.

The traffic management system includes three main components:

– The collecting of traffic data, by a microcontroller, is made according to a three-state algorithm like wait, track and save. The traffic flow is calculated and generated data is uploaded to an open-source Thing Speak cloud solution, for future use.
– The approach designed to optimize traffic phases by exploiting instantaneous traffic information adopting two parameters for traffic flow: The frequency, considered as the reverse of the elapsed time from last crossing car, which is measured in cars per time unit. Traffic volume, measured as the total of cars occupying a per-unit road length. This work uses the Least Estimate Square (LES) algorithm, based on current data, and determines the green times and phases of traffic cycles by minimizing the waiting duration of cars at a junction.
– In the last step, the approach is using a network of neurons. At peak traffic hours, when the flow of traffic is particularly heavy, the data accumulated in memory is utilized to anticipate the flow of traffic.

The number of vehicles was used to calculate the traffic flow. Thus, when vehicles are travelling, the proposed solution uses their mobility in area of interest to detect the vehicle’s chassis by filtering image algorithms. This algorithm filters the flow of video acquired from the camera and compares the current images with the previous ones. After detecting a change, the micro controller tracks the limits of the object in movement and records the number of objects and their direction.

The traffic flow was considered a key factor in calculating the learning models. For this purpose, they used a simple traffic representation that depends on the flow intensity and the frequency of cars.

Initially, at t_0, number of cars in sections W, E, N and S is W_0, E_0, N_0 and S_0, respectively and the traffic light is on the point of changing between green to red for the Northern and Southern routes and red to green for the Western and Eastern routes. For the Western and Eastern directions, the green remains on up to t_1. For time duration $\Delta t = t_1 - t_0$ in each direction number of automobiles at t_1 is yielded as below:

$$\begin{aligned} W_1 &= W_0 + (in_W - out_W)\,\Delta t \\ E_1 &= E_0 + (in_E - out_E)\,\Delta t \\ N_1 &= N_0 + (in_N - 0)\,\Delta t \\ S_1 &= S_0 + (in_S - 0)\,\Delta t \end{aligned} \tag{2.6}$$

The four traffic directions are represented in matrix form as follows:

$$\begin{pmatrix} W_1 \\ E_1 \\ N_1 \\ S_1 \end{pmatrix} = \begin{pmatrix} in_W - out_W \\ in_E - out_E \\ in_N \\ in_S \end{pmatrix} \Delta t + \begin{pmatrix} W_0 \\ E_0 \\ N_0 \\ S_0 \end{pmatrix} \tag{2.7}$$

The problem can be modeled by the following equation matrix:

$$\mathrm{y} = \mathrm{Ax} + \mathrm{b} \tag{2.8}$$

The application of Least Squares Estimation (LSE) for the above system to minimizing $\| Ax - b \|^2$ is attained by the equation below:

$$x_{LSE} = (A^T A)^{-1} A^T b \tag{2.9}$$

Thus, they defined the equation for the simple traffic cycle duration matrix as follows:

$$\Delta t = \left[\begin{pmatrix} in_W - out_W \\ in_E - out_E \\ in_N \\ in_S \end{pmatrix}^T \begin{pmatrix} in_W - out_W \\ in_E - out_E \\ in_N \\ in_S \end{pmatrix} \right]^{-1} \begin{pmatrix} in_W - out_W \\ in_E - out_E \\ in_N \\ in_S \end{pmatrix}^T \begin{pmatrix} W_0 \\ E_0 \\ N_0 \\ S_0 \end{pmatrix} \tag{2.10}$$

Simulation results showed that for low-density traffic, where the traffic input volumes are less than the output volumes, the algorithm tries to maintain the traffic levels at end of all cycles. Moreover, for high traffic density, where there are more cars entering than exiting, the traffic conditions are rarely improved. So, the algorithm was able to maintain the lowest possible traffic conditions following several cycles, but was not efficient. On the contrary, the simulation results obtained with machine learning show that the neural network is able not only to learn traffic patterns, but also to predict with a certain degree of precision the traffic flow.

Frank et *al.* [23] elaborated a control system for traffic lights at an intersection using IoT environment and traffic density. They used image processing, as shown in Figure 2.3, in which images captured via cameras are saved in a computer server for real time comparison with the reference image to identify the traffic density, using Raspberry Pi.

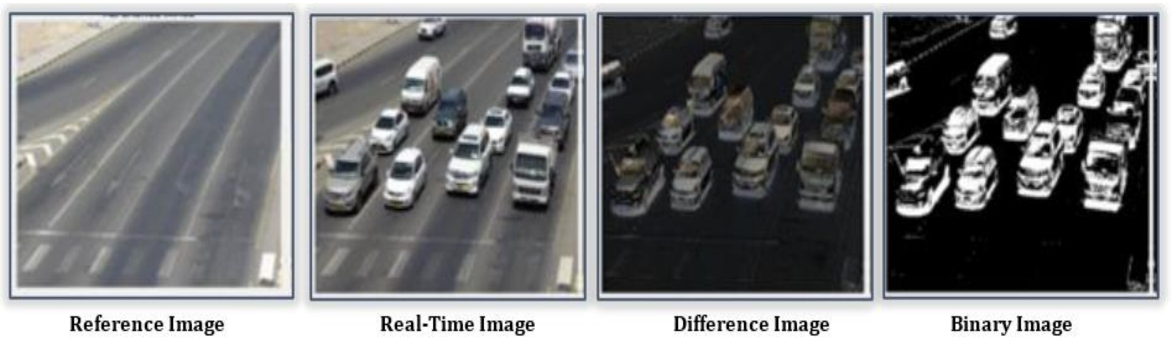


Figure 2.3 Image processing process [23].

To supervise and manage the traffic signal with the higher speed connectivity of the internet, the authors in [23] are based on IoT and density utilizing the processing of images, in which the images captured by the cameras are stored in a cloud server. These images will be compared in real-time to the reference images in order to identify the traffic density. For this purpose, a camera captures the image when the associated intersection is empty (zero traffic flow) which constitutes the image reference. Following the continuous capture of the real-time image of the junction by a USB camera, the Raspberry Pi uses a comparison of the reference image with the real-time image to calculate a differential image, then converting the difference frame to a grey scale image which in turn is converted to a binary black and white image (Figure 2.3). The resulting difference is zero if the two compared images are identical. However, if the camera's field of view includes vehicles, the difference image will be translated into white and black; the white pixels indicate different parts of the two images and black pixels correspond to similar elements. A comparison of the white and black pixels is used to determine the traffic density. Afterward, the intensity data is transferred to the M2x IoT platform cloud (the main server), which is accessed by a Java desktop application created to control and supervises road signals. The embedded system retrieves the control instructions, which are transferred to the platform cloud via an application, and links them via a Raspberry Pi to the traffic light unit control.

3.4 Synthesis and Comparison

Existing solutions in the STLS domain focus especially on solving the congestion problem in terms of queue size, waiting time. Three types of STLS approaches can be distinguished: heuristic-based approaches, sensor-based approaches and hybrid approaches. Table 2.2 shows a comparative study between these approaches for a single intersection based on the criteria: 1- hybrid cycle, 2- dynamic phase, 3- control metrics and 4- the consideration or not of priority vehicles. These criteria are chosen for several reasons: the control metrics refer to how the existing technique optimizes these metrics, the choice of the selection function that determines the next phase and improves the solution quality, the calculation of the green time and the efficient mechanisms for managing priority vehicles.

Most of the existing work neglected the automatic and intelligent management and control of priority vehicles at the intersection when scheduling traffic signal phases. For the cycle model, existing works [7, 8, 14, 16, 20–23] use either a single cycle model, or combine several models in the same solution. The choice of the next phase to be executed is done dynamically, i.e., the phase that gives a better reward is chosen first, or the phases are executed one after the other according to the order defined in the cycle model.

It is noted that the use of the control metrics "Waiting time, queue size, input flow and output flow" is varied from one solution to another. The sensor-based work [7] uses all four metrics; in the hybrid approach [8] three metrics are used. Two metrics have used in the work [7, 14, 16, 21] and [20, 22] used a unique metric.

Table 2.2 Comparison of different STLS approaches for a single intersection.

Approach		**Hybrid Cycle**	**Dynamic Phases**	**Control Metrics**				**Vehicle Priority**
				Waiting Time	**Queue Size**	**Input Flow**	**Output Flow**	
Heuristics	[14]	✓	✓	✗	✓	✗	✓	✗
	[16]	✓	✓	✓	✓	✗	✗	✗
Sensors	[7]	✗	✓	✓	✓	✓	✓	✗
	[20]	✓	✓	✓	✓	✗	✗	✗
	[21]	✗	✓	✗	✓	✗	✗	✗
	[22]	✗	✓	✗	✓	✗	✗	✗
Hybrids	[8]	✗	✓	✓	✗	✓	✓	✗
	[23]	✗	✓	✗	✓	✓	✗	✗

✓: Considered, ✗: Not considered

Moreover, the integration of fault tolerance mechanism and recovery (How to maintain the system operating in spite of failed component) is very little considered in most traffic light management approaches [7, 8, 14, 16, 20–23]. In addition, security aspect considerations, such as external attacks, indisputable and necessary, especially for works that use the Internet of Things or Cloud were not taken into account in design of STLSs.

4. STLS Approaches for a Set of Intersections

Intersections in cities come in a variety of configurations; however, we will examine the most common and general ones that may result in the implementation of an appropriate Smart Traffic Light System. Some of these configurations have been managed in STLS reported in previous works as heuristic-based approaches [11, 17, 103, 104], sensor-based approaches [105] and hybrid approaches [9], a solution that combines the first two approaches.

4.1 Heuristics based Approaches

Segredo et *al.* [11] proposed a vector formulation for traffic signal synchronization to solve the urban congestion problem. The solution is encoded by an array of integers as an individual coding, where every item is the duration of the phase of a given cycle of the signal lights implied at a particular junction. The phases of the various junctions are placed in the vector solution consecutively, so that the whole plan of traffic signals is modeled as a single vector of integers that represents the phase duration. This coding takes into account the time offset of each intersection, which increases the realism of the instances. Figure 2.4 shows an example of coding two consecutive intersections in the city by phase duration. The solution has been validated in a centralized environment, where a simple vector of integers represents the phase times of a particular cycle of traffic signals included in the crossroads of a metropolitan area.

The authors developed an extended algorithm based on NSGAII (Non-Dominant Sorting Genetic Algorithm II) [106] for the optimization of different traffic parameters. NSGA II represents one of the more approved multi-objective evolutionary algorithms (MOEA), based on solution diversity. The work benefits from the quick sorting function of non-dominations with decreased complexity of calculation. In order to ensure elitism, it employs a selection operator that couples the precedent populations with the new ones generated.

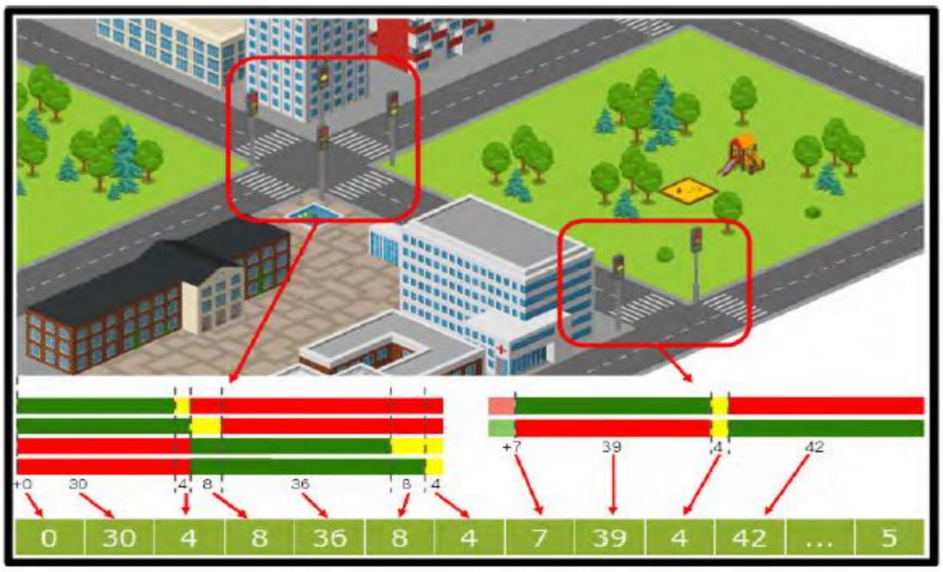


Figure 2.4 Solution coding, phase duration and traffic light offsets for two intersections [11].

The algorithm attempts to optimize the four objectives:

- Maximizing the sum of cars that attain their arrival point V_R, or similarly minimizing the sum of cars that do not attaining their arrival point V_{NR}, over a simulation period T_{sim}.
- Minimizing the total travel time T_{trip} from a starting point to the arrival point in the study field window, which is equal to the total travelling periods of all cars. The period travelled by a car is equal to the period it took to reach the arrival point. Cars that do not reach their arrival point consume all of the simulation period.
- Minimizing the sum of the stopping and waiting times of all vehicles T_{sw}, which is the global time consumed by each vehicle at intersections where the light is red.
- Maximizing the green and red colors ratio P in every state phase of all crossings, is given as below:

$$P = \sum_{i=0}^{intr} . \sum_{j=0}^{ph} d_{i,j} \frac{g_{i,j}}{r_{i,j}} \tag{2.11}$$

Such that:

- $intr$ is the number of all crossings.
- ph is the number of all phases.
- $g_{i,j}$,$r_{i,j}$ indicate the number of green and red traffic light colors at crossroad i and phase state j, respectively.

Note that the minimum value of $r_{i,j}$ is 1 and so as not to have it divided by zero.

- $d_{i,j}$ is the duration of the green time at crossroad i and phase state j.

Equation (2.11) favors green traffic lights at congested intersections and red lights at other low traffic intersections.

The objective function shown in Equation (2.12) has been proposed and discussed in the work reported in [74, 107]:

$$f = \frac{\mathrm{T_{trip}} + \mathrm{T_{sw}} + \mathrm{V_{NR}}\mathrm{T_{sim}}}{\mathrm{V_R^2} + \mathrm{P}} \qquad (2.12)$$

Note that the V_R criterion is squared to give it priority above the rest of criteria, as it constitutes the most important objective. Furthermore, the non-arriving cars V_{NR} is multiplied by the duration of simulation T_{sim} to impose a penalty to this undesirable situation.

Multi-objective optimization is performed by adapting the principle of individual diversity (ADI, DBI or DCN). Their objective functions, based on diversity maximization, have been used since they were applied successfully to solve some other problems in real world [108], such as:

– ADI (Average Distance for every Individuals) [86] is the average distance in Euclidean space of the genotype from other population individuals.

– DBI (Distance from Best Individual) [109] is the distance Euclidean in the space for genotypes of the population's individual best. The individual best is identified by its objective initial value.

– DCN (Distance to our Closest Neighbor) [109] is the distance Euclidean in genotype space to the nearest neighbor in the population

The variants of ADI NSGAII, DBI NSGAII and DCN NSGAII are compared to three optimization methods: two single-objective optimization methods by the GA (Genetic Algorithms) and PSO (Particle Swarm Optimization) meta-heuristics and one Variable Neighborhood Search (VNS) method on four different benchmarks: Berlin, Paris, Stockholm and Malaga. Two experiments were carried out to investigate the capabilities of the NSGAII variants. The first one consists in comparing the variants of ADI NSGAII, DBI NSGAII and DCN NSGAII. The results of the first experiment showed the effectiveness of DCN NSGAII. The second experiment compares DCN NSGAII variant with the GA, VNS and PSO meta-heuristics. The DCN NSGAII variant has performed well for small locations (Stockholm, Berlin and Paris). On other hand, in large locations (Malaga), the GA single-objective optimizer provided the best performance.

In [11], the authors proposed the NSGAII algorithm to improve the time offset parameter between intersections, and the NSGAII is one of the most used MOEAs nowadays. But it is noted that the diversity principle of the NSGAII algorithm fails when analyzing the largest instance. For example, the authors considered a central solution that uses a single vector representing the solution of all intersections. So, instead of using a single vector for the solution to the large instances, it is preferable to consider a distributed system with a single integer vector representing the solution of each intersection.

Rasheed et *al.* [17] developed an advanced technique named MADQN (Multi-Agent Deep Q-Network) founded on multi-agent aspects to manage and control high volume traffic. They also included scenarios of weather disturbances that can aggravate the traffic situation. The MADQN technique was adopted for an area located in the city of Sunway, Malaysia to demonstrate its effectiveness. Their simulations were realized on a 3×3 traffic network with nine intersections, each intersection with four lanes. The proposed technique provides stable and optimal solutions, by having one agent collaborate to affect the operating environment of neighboring agents as cars move from one junction to next. In addition, the agents collaboratively coordinate their actions with the actions of neighboring agents.

Moreover, the global optimal value Q_value summarises all local values Q_value_i found by all agent i and corresponds the overall objective function that leads to an optimum balancing. The local view availability of neighbouring agents for an agent is attributed to the convergence, updating the Q_i values of an agent i using data from neighbouring agents j (e.g., Q_j values), and the best response to neighbouring agents j is the action of an agent i.

In DQN system, the neural artificial network contains three convolutional levels. During learning phase, data is routed from the entry level to the cache level, and lastly to the exit level, which represents $Q_{value}, Qt\ (s_t,\ a_t)$ of the possible actions a_t. The DQN system has two main characteristics compared to the traditional Q-Learning approach. The first characteristic is experience replay, where an agent stores an experience $e_t\ (s_t,\ a_t,\ r_{t+1}, s_{t+1})$, in a replay memory $D_t =\ (e_1,\ e_2, \dots,\ e_t)$, and then trains using the randomly chosen experiences from its replay memory D_t. The second characteristic is using the weight θ_k to approximate the $Q\_value\ Qt\ (s,\ a\ ; \theta_k)$ at the k^th^ iteration. In their study, they used the non-exponential Burr XII type distribution [110], to represent the vehicles' inter-arrival time, which generalizes the Poisson process in which the vehicles' inter-arrival times are distributed exponentially [111].

To model traffic signal controllers, the authors envisage a set of junctions I in a network traffic where each junction $i \in I$ has a set of arrival ways K^i, and a set of neighbouring junctions J^i. The crossroads turns on the phases of traffic in a circular manner and regulates the duration of the phases in traffic using DQN.

The architecture of the standard DQN and applied DQN multi-agent approach is presented in Figure 2.5, which shows that the DQN is integrated with the traffic signal control at the intersection i. The entry level contains five neural cells, each one corresponding to a state, five hidden levels fully connected to 400 neural cells each, and the exit level has five neural cells, each r one corresponding to an action possible. A weight is assigned to each connection. Each node has a function of linear rectified activation that conducts a descent gradient.

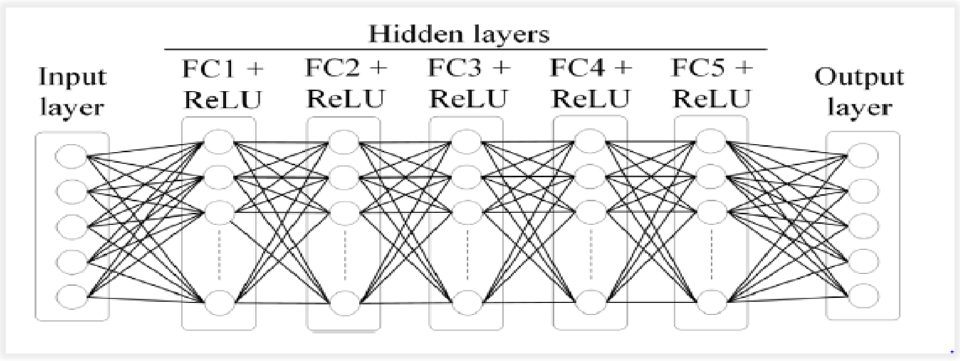


Figure 2.5 DQN architecture [17].

During learning, the 5 sub-states of the state $s_t^i = (s_{1,t}^i, s_{2,k,t}^i, s_{3,k,t}^i, s_{4,t}^i, s_{5,t}^i)$ are introduced to the neural cells of the entry level. The flows data are subsequently transmitted to the hidden levels, and ultimately to the exit level which delivers the Q_value of $Q_t^i = (s_t^i, a_t^i)$ of its probable actions $a^i = (0, 1, 2, 3, 4)$, where $a_{max} = 4$ at junction i.

The delayed reward ${r_t}^i\,({s_t}^i)$ is calculated based of the time waiting total. This reward is equal to the difference in time waiting total of all cars at a junction i at time t « ${W_t}^i$» and time $t + 1$ « ${W_{t+1}}^i$», (I.e., prior to and post performing the action) is defined by the formula in Equation (2.13):

$$ {r_t}^i\,({s_t}^i) = {W_t}^i - {W_{t+1}}^i \tag{2.13} $$

The agent receives a positive deferred reward when ${W_t}^i > {W_{t+1}}^i$, a negative deferred reward when ${W_t}^i < {W_{t+1}}^i$ and a null deferred reward when ${W_t}^i = {W_{t+1}}^i$.

The authors take four approaches for their simulation namely, deterministic (fixed phase and period), RL, MARL and MADQN. The three parameters measured are: Queue length, waiting time and flow rate which measures number of cars passing through a junction during a unique phase of traffic. The simulation results show that MADQN outperforms the other approaches by increasing flow rate by 70%, reducing length of queue by 75% and the waiting time by 7%.

Rasheed et *al.* [17] presented an optimization method based on delayed reward, but it is calculated only by the total waiting time difference of all cars at a given junction. This method ignores the flow parameters and queue length for the calculation of the reward, which are proven in their simulation results.

Nesmachnow et *al.* [103] proposed an evolutionary parallel algorithm to synchronize and adapt traffic signals to optimize public transport, in particular rapid bus system. This algorithm maximizes, for buses and other cars, the average speed. This work applies the proposed algorithm to the case study of Garzón Avenue (Montevideo, Uruguay), and compares the results obtained from the parallel evolutionary algorithm with the modeling of the real non-optimized scenario. Their motivation is to utilize a parallel evolutionary genetic algorithm to minimize the execution time compared to a simple sequential model such as traditional genetic algorithms. They apply a parallel master-slave model and use a multithreaded system, adapted to run on modern multicore computers. The objective function used is the following:

$$f = W_B \times \overline{S_B} + W_O \times \overline{S_O} \tag{2.14}$$

Such as:

- $0 \leq (W_B, W_O) \leq 1$ and $W_B + W_O = 1$,
- S_B is the medium bus speed,
- S_O is the medium speed of other cars,
- W_B denotes the weight of the medium bus speed parameter,
- W_O is the weight of the medium speed of other cars.

The optimization of solution allows simultaneous maximization of both speeds S_B and S_O. The objective function used to evaluate solutions of the evolutionary algorithm, assigns to public transport a high priority by attributing adequate values to the weight W_B and W_O.

The model of the proposed solution, illustrated in Figure 2.6, shows that their system is composed of two components, the first one is the optimization algorithm and the second is the procedure using simulations to evaluate solutions. The master-slave parallel evolutionary optimization algorithm aims to find the optimal configuration (solutions) of traffic lights at each slave thread, taking into account the objective function. In turn, the simulation procedure evaluated the solutions obtained by the parallel algorithm. This approach uses two categories of processes: the master and slave processes. A master process assigns to the slave processes a solution set to be examined within the threads. The slave processes have the objective to assess every traffic light combination and send the evaluation findings back to the master process in order to select the optimal configuration.

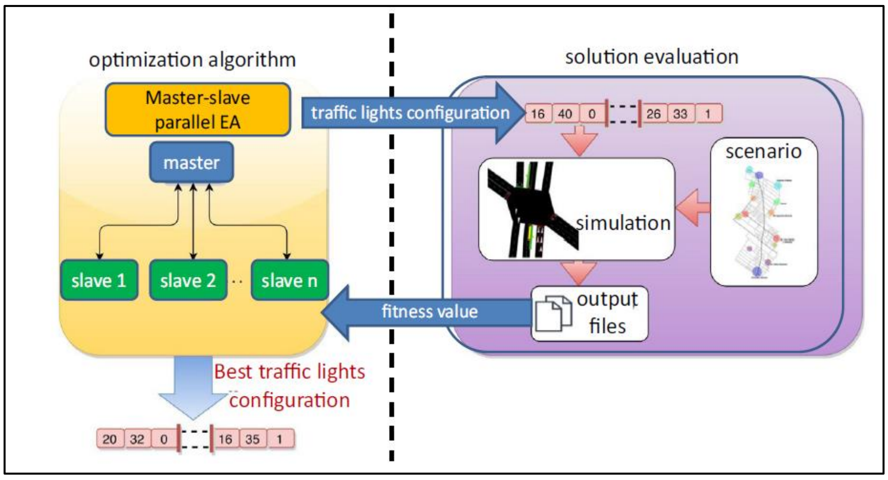


Figure 2.6 Optimization model [103].

For the synchronization of the traffic lights, the solution was defined as an integer vector. The vector indicates the time of each phase for each intersection, as well as the start time of the traffic light cycle. The values are expressed in seconds. The phase values are limited between 16 and 120 seconds, the cycle time determines the number of light phases in each traffic junction. The authors have then presented an interesting parallel evolutionary algorithm to resolve the synchronization problem for a set of intersections. However, this solution approach may suffer from the fact that it is based on a centralized system since the solution is encoded on a single integer vector. The proposed solution, coming from a centralized decision system in a distributed environment, risks not adapting to a distributed decision evolution involving IoT services for the benefit of junction users.

Sofronova et *al.* [104] presented a mathematical method to simulate the flow control problem for a set of intersections in an urban area. This work uses an evolutionary genetic algorithm for optimal control of the solution. The mathematical system founded on the theory of controlled networks describes the control system by differential equations. In addition, one arrives at a finite difference recurrent equation system to describe the traffic flow at each intersection and at different time intervals. The number of vehicles at every crossroad in the traffic network is defined as a state vector and the number of crossroads describes the size of the state space. The model takes into account the traffic evolution according to the traffic light signals. The movement of cars between intersections causes the state vector to change, so that the control determines the timing of the traffic light phases at the intersections, allowing or disallowing certain movements between intersections.

The connection information between the intersections and the number of directions at each intersection are needed to define the mathematical model of control. The length of all phases is an integer and the traffic signals are synchronized, so that the count of time is done simultaneously at all network traffic signals. The permutation order of the traffic signal phases remains the same; the controller defines the permutation times of the phases at an intersection. Either this approach keeps the phase unchanged or a switch to the next phase is provided. Thus, a single fixed cycle and phase model is applied.

The authors have effectively introduced a mathematical model with the help of genetic algorithms to determine the optimal solution; however, the model lacks of robustness since the system is centralized and uses a fixed cycle model.

4.2 Sensor-based Approaches

Lah et *al.* [105] suggested smart traffic control and monitoring architecture for a set of intersections. The system includes magnetic detectors using magneto-resistors that are highly sensitive to anomalies in the magnetic field of terrestrial environment due to the presence of a vehicle. A wide area network with low power ensures that the captured information is transmitted to a specific control station. The central server manages the operation of the traffic lights and will have complete control, incorporating all the traffic signals in the region.

Each signaled intersection is covered by a LoRA transmitter and receiver, which sends and receives instructions to or from the main host. Additionally, a LoRA station at the main host sends commands to the traffic lights using local stations at each intersection. For measuring traffic flow, sensors are placed strategically along the road. The data collected will then be sent to the Control Centre. This will include a server that guides the programming of traffic lights at junctions.

The monitoring and control operation is as follows:

- At the intersection:
    - Receiving the signal from the magnetic sensor by LORA.
    - Group the signal according to the traffic line from which it comes.

  - Translate the signal into traffic volume.
  - Transmit the data to the central server.
- At the central server:
  - Analyze the traffic configuration.
  - Identify the line where the traffic is intense.
  - Refer to the database for traffic lights that relate to specific routes.
  - Send a command to the traffic lights to extend the green light or change to the red light.

The authors used magnetic sensors to monitor traffic flow at a regular time and insensitivity to weather conditions. Nevertheless, the proposed system from high installation and maintenance cost, due to the difficulty of setting up this type of sensor.

4.3 Hybrid Approaches

Lin et *al.* [9] proposed a centralized system for recognizing vehicle license plates in real time to manage traffic in Tainan City in Taiwan, based on traffic flow data from sensors. They also proposed a mathematical model for combining Gaussians. The system analyses information from a network of traffic lights in the central of Tainan City, including 124 traffic signals with 53 sensors for vehicle license plate identification.

The primary goal of this work is to generate in a timely way the traffic signal schedule for the next 10 minutes. For this purpose, the authors adopt real-time vehicle number plate recognition-based flow prediction, with the adaptation of a mathematical model. Their solution, illustrated in Figure 2.7, consists of two parts: the prediction of the traffic flow transition and the generation of traffic signal schedules.

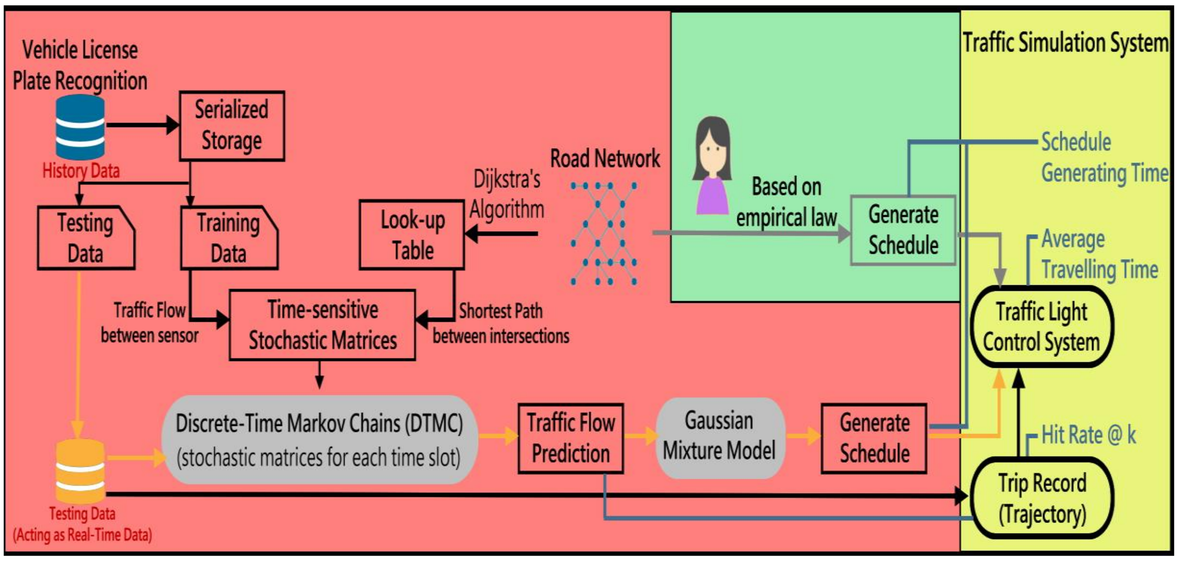

Vehicle License Plate Recognition
History Data
Serialized Storage
Testing Data
Training Data
Look-up Table
Dijkstra's Algorithm
Road Network
Based on empirical law
Generate Schedule
Traffic Simulation System
Schedule Generating Time
Average Travelling Time
Traffic Flow between sensor
Time-sensitive Stochastic Matrices
Shortest Path between intersections
Traffic Light Control System
Discrete-Time Markov Chains (DTMC) (stochastic matrices for each time slot)
Traffic Flow Prediction
Gaussian Mixture Model
Generate Schedule
Hit Rate @ k
Testing Data (Acting as Real-Time Data)
Trip Record (Trajectory)

Figure 2.7 Traffic light control new proposal [9].

To predict traffic flow transition, Dijkstra algorithm was initially used to construct a search table that lists the shortest paths between intersections. Next, they adopted Markov chain discrete time to describe stochastic time matrices that register the transitional movement at various monuments among peaks using the license plate recognition historical data. The next step, based on the DTMC, is the prediction of the traffic flow.

For example, in Figure 2.8, given the rates of transition for $\overrightarrow{AF}$ and $\overrightarrow{BD}$ and the traffic flows in real time for *A* and *B*, the ratio of basic time for junction *C* is defined as:

$$T_{ACF} : T_{BCD} = (F_A \times M_{AC} + F_F \times M_{FC} + \mu) : (F_B \times M_{BC} + F_D \times M_{DC} + \mu) \qquad (2.15)$$

Such as:

- T_{ACF} is the green light time.
- F_A is the traffic flow in real time registered at junction *A*.
- M_{AC} is the transition predicted from junction *A* to *C* from the DTMC technique in the given interval time.
- μ is a fixed value to prevent a lane that contains almost no flow of traffic from receiving precisely 0 seconds of green light (μ is fixed to 0.002 in the study).

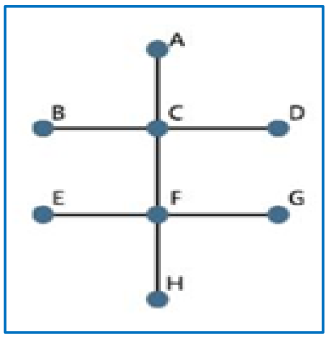


Figure 2.8 Example of a solution for eight junctions [9].

Therefore, the Mixture Gaussian Model (MGM) is adapted to adjust the green light time difference for every road in the identical direction, so the modified time ratio for junction C is determined as follows:

$$T'_{AC\,F} : T'_{BCD} = T_{AC\,F} + G^F\,(C) \times T_{CFH} : (T_{BCD}) \qquad (2.16)$$

Where:

- T'_{ACF}, Related to the modified green light period.
- G^F (C) Denotes, at the junction C, the Gaussian distribution with its centre fixed at junction F (σ is set to 1).

Their preliminary evaluations proved that basic methods, such as minimum distance and route-based transitions, correctly inferred at least half of the transition destinations.

4.4 Synthesis and Comparisons

We analyzed and compared existing STLS approaches for a set of intersections according to three criteria: 1- type of system, centralized/distributed, 2-control parameters (waiting time, queue size, vehicle flow, others) and 3- coordination between intersections. The comparative study of STLS approaches for a set of intersections is summarized in Table 2.3 We have identified common limitations, in particular for coordinating multiple intersections with centralized approaches. Most existing works [9, 11, 103- 105] do not consider coordination between intersections. They do not provide efficient mechanisms to optimize all control parameters (waiting time, queue size, vehicle flow).

Table 2.3 Comparative study of STLS approaches for an intersection set.

Related works		Type of system	Control parameters				Coordination between intersections
			Waiting Time	**Queue Size**	**Vehicle Flow**	**Others**	
Heuristics	[11]	Centralized	✓	×	×	Vehicles arriving at destination	×
	[17]	Distributed	✓	×	×	×	✓
	[104]	Centralized	×	✓	×	×	×
	[103]	Centralized	×	×	×	Vehicle speed	×
Sensors	[105]	Centralized	×	✓	×	×	×
Hybrids	[9]	Centralized	×	×	✓	×	×

✓: Considered, ×: Not considered

5. Global Smart Traffic Light System and Internet of Things

We are currently witnessing an important shift from the traditional internet to the Internet of Things (IoT) [112] or internet of everything. Therefore, all domains are being driven by IoT and more and more innovative X-IoT applications are emerging such as: Agriculture IoT [113], Smart-cities IoT [114, 115], Vehicles-IoT [116], Healthcare IoT [117, 118], Smart Home automation or Domotics IoT [119, 120], Industries IoT [121 -123], Intelligent Transportation System IoT [124], Environment IoT [125], Nanonetworks IoT [126], etc. as illustrated in Figure 2.9. In our context, any vehicle can be an object connected to this IoT, as long as it is equipped with the appropriate equipment. As vehicles are increasingly equipped with powerful platforms, which are convenient for a smart city environment, traffic management at intersections would be of great benefit. Therefore, the cooperation and coordination of all STLSs of a smart city should result in a Global STLS enabling to provide instantly all vehicles with interesting information about traffic status. This valuable service can be provided by a robust dedicated server via IoT infrastructure, additionally to other services considering a vehicle as part of a more important global domain likely Intelligent Transport System which include road security issue, infotainment, etc.

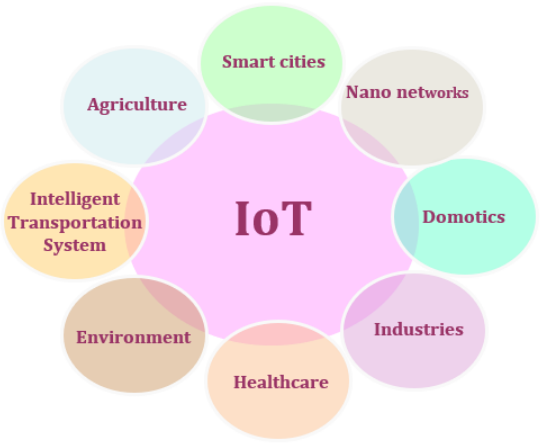


Figure 2.9 Example of domains influenced by IoT.

From the perspective of smart city management, citizens must be able to access useful information on the status of urban traffic at any time. This can be done via display panels, smart phone or directly on vehicle boards. For this purpose, an urban traffic information management center must be available to retrieve the status of all intersections in the city. Thus, the user will be provided with the least congested route (or guide him by voice according to the selected route) allowing him to reach his destination in a smooth way. Obviously, this operation and many others (such as the physical status of the roads, etc.) can only be efficiently accomplished in Internet of Vehicles supported by Internet of Things environment. Figure 2.10 shows an example of IoV sustained by IoT.

Figure 2.10 IoV sustained by IoT.

6. STLS Simulation Methods

Different methods are used to simulate the performance of current STLS field approaches, some researchers using already existing simulators, such as: SUMO [127], VISUM [128], EMME [129], etc. Other authors developed their own simulators to evaluate their approaches.

The authors in [11, 14, 16, 103] utilized several scenarios in Urban MObility (SUMO) software [127], to evaluate the solutions generated by their approaches. In order to evaluate performance of their solution, Kumar et al. [7] developed their own simulator using Java fx technology. Simulation tests in [17] were performed using combination of SUMO traffic simulator and MATLAB. To simulate traffic conditions, the authors in [8] used a PYTHON program that was written for this end. In [23], the authors developed a user interface application in Java to monitor data in a cloud server, and to check the traffic density, an image processing is simulated with MATLAB. For the demonstration of their work, the authors in [9] provided an online web page [9].

7. Conclusion

In this chapter, we have presented the state of the art of the various STLS approaches, describing their models and algorithms. STLS models are classified into two types: models for a set of intersections and models for a single intersection. We have detailed the different parameters of STLS models for a particular intersection and the different architectures of STLS models for a set of intersections. In particular, we have classified STLS approaches, depending on the algorithm applied, into three categories: heuristic-based optimization approaches, sensor-based approaches and hybrid approaches that combine the last two approaches. In addition, we presented performance evaluation methods for existing STLS approaches and discussed their simulation results. Nevertheless, to this day and from all the work reviewed dealing with STLS, it seems that the latter have not yet been able to really satisfy what is expected from smart city management.

The following chapter focuses on the contributions we have made, presenting our Adaptive and Dynamic Smart Traffic Light System (ADSTLS) for single and multiple intersections.

Part 2: Contributions

Chapter 3: ADSTLS: Adaptive and Dynamic Smart Traffic Light System for Single and Multiple Intersections

Content

1. Introduction 78
2. System Architecture..78
   2.1 Single Intersection 79
   2.2 Multiple Intersections 80
3. Assumptions 81
4. System Models and Traffic Metrics..81
   4.1 Intersection Model...81
   4.2 Cycle and Phase Models 82
   4.3 ADSTLS's Traffic Metrics..83
5. ADSTLS Control System for Single Intersection 86
   5.1 Fault Tolerance 87
   5.2 Traffic Data Collection and Preprocessing 88
   5.3 Traffic Data Privacy and Security 89
   5.4 Computation of the System Metrics 91
   5.5 Decision Making in ADSTLS Modes..92
   5.6 Traffic Light Control Execution 107
6. Chronological Coordination of ADSTLS at Multiple Intersections..........107
   6.1 WCSO-Based Agent Intersection Controller 107
   6.2 Chronological Coordination of WCSO Agent Controllers 108
7. Conclusion..112

1. Introduction

Currently, traffic flow using different lanes is steadily increasing. Traffic congestion is becoming more difficult in different situations due to unpredicted events and accidents, requiring effective traffic light management techniques to meet these evolving changes. On the other hand, managing traffic based on conflicting traffic metrics like input traffic flows and queue occupancies is challenging. This complexity is evident and handled by one or more intersections. It is crucial to select the most suitable lane, considering traffic flow metrics and performance. Smart Traffic Light System (STLS) plays a significant role in efficiently managing traffic flows. However, these systems lack enough flexibility and adaptability to effectively handle both regular and urgent vehicles at the city's intersections.

This chapter introduces an Adaptive and Dynamic Smart Traffic Light System (ADSTLS) designed to coordinate traffic signals at a single intersection, manage emergency traffic intelligently and effectively, and coordinate signals at multiple intersections to alleviate congestion and enhance traffic flow during emergencies and changing weather conditions. The proposed system comprises six stages: ensuring fault tolerance in ADSTLS components, collecting and preprocessing traffic data, ensuring privacy and security of traffic data, calculating traffic metrics, making decisions in dynamic and adaptive mode, and executing traffic light control.

Two control modes are proposed: Dynamic Intelligent Mode (DIM) and Adaptive Emergency Mode (AEM), while adapting the improved optimization method based on WCSO (Weight Chicken Swarm Optimization). To keep dynamicity, we have chosen a flexible approach that effectively utilizes traffic information and reacts rapidly using three traffic light models (cycle, phase, and hybrid) in DIM mode. Additionally, the AEM mode, effectively, manage emergency vehicles with varying priority levels to synchronize traffic signals in real time. An extension known as CCADSTLS (Chronological Coordination of ADSTLS) is introduced for multiple intersections. This extension aims to minimize congestion and traffic delays during disasters by facilitating timely coordination among ADSTLS agents.

2. System Architecture

This section describes the general architecture of the ADSTLS system, presenting the composition of the different units of the system for an intersection and the multi-agent model for a set of intersections.

2.1 Single Intersection

A sensor video and three signal lights were included in each of the four lanes. Each sensor video (i.e., camera) monitored the traffic flow videos in real time and transmitted this information to a video processing unit. At this level, the videos were analyzed to extract traffic data for each direction and lane. The traffic signal controller processes traffic data and strategically manages traffic lights. The architecture is built around two principal units, as shown Figure 3.1.

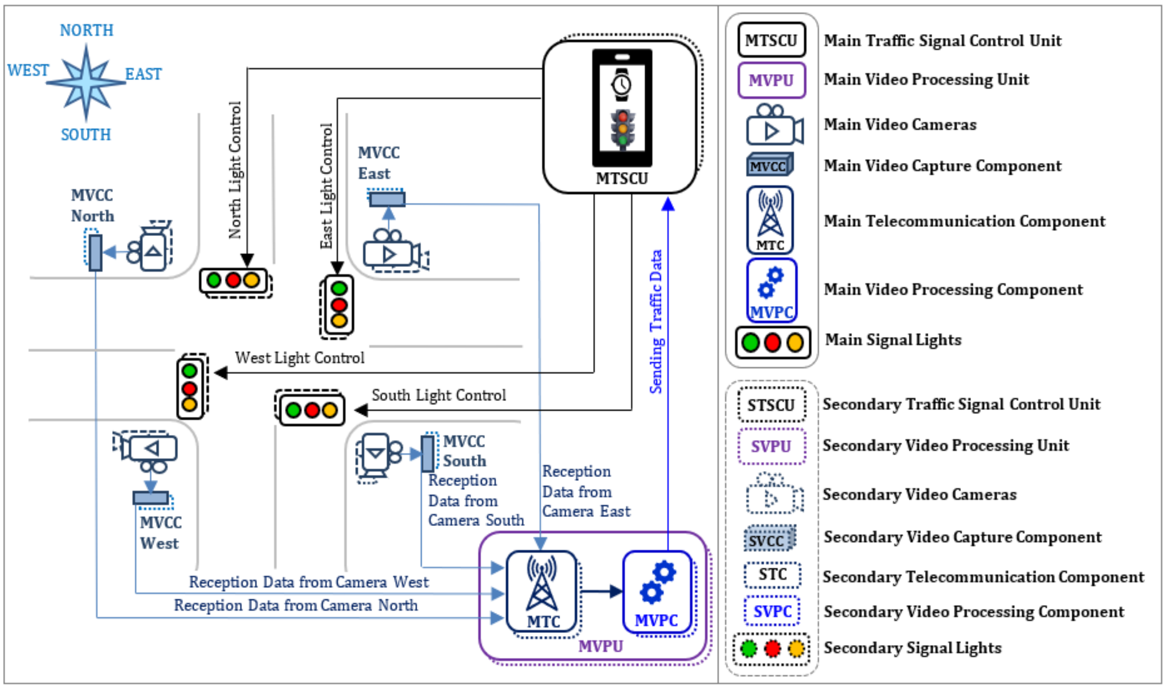

NORTH
WEST
EAST
SOUTH
MVCC North
North Light Control
East Light Control
MVCC East
MTSCU
Sending Traffic Data
West Light Control
South Light Control
MVCC South
Reception Data from Camera South
Reception Data from Camera East
MVCC West
Reception Data from Camera West
Reception Data from Camera North
MTC
MVPC
MVPU
MTSCU Main Traffic Signal Control Unit
MVPU Main Video Processing Unit
Main Video Cameras
MVCC Main Video Capture Component
MTC Main Telecommunication Component
MVPC Main Video Processing Component
Main Signal Lights
STSCU Secondary Traffic Signal Control Unit
SVPU Secondary Video Processing Unit
Secondary Video Cameras
SVCC Secondary Video Capture Component
STC Secondary Telecommunication Component
SVPC Secondary Video Processing Component
Secondary Signal Lights

Figure 3.1 General architecture of ADSTLS.

2.1.1 Main Video Processing Unit (MVPU)

It automatically analyzes the collected videos to extract the traffic information and transmits it to the traffic signal controller. This unit comprises three components:

- The Main Video Capture Component (MVCC) consists of a video camera road that monitors and captures traffic videos in real time.
- The Main Telecommunication Component (MTC) transmits videos to the video processing component,
- The Main Video Processing Component (MVPC) extracts the traffic data and transmits it to the traffic controller.

2.1.2 Main Traffic Signal Control Unit (MTSCU)

It receives and processes traffic data sent by video processing unit as follows:

- Calculate traffic metrics used by ADSTLS system,
- Control traffic lights in each lane,
- Switch between intelligent mode and emergency mode depending on the situation and,
- Communicate with the second traffic signal control unit of the same intersection.

To ensure fault tolerance, all ADSTLS components were duplicated, as shown in Figure 3.1. Our goal was to enable the system to continue operating efficiently when one or more of its components fail.

2.2 Multiple Intersections

We assume a traffic network consisting of a set of connected intersections where vehicles travel from one intersection to another. Each intersection I_{id} is assigned an identifier (id). An intersection has a maximum of four neighboring intersections on each side of the roadway (west, north, east, south), which are recognized by this intersection based on their identifiers $(WI_{id}, NI_{id}, EI_{id}, SI_{id})$.

Our system consists of agent controllers A_{id} and their managed intersections (see Figure 3.2). The agent controller A_{id} which is deployed in a base station, is responsible for controlling all traffic lights and a network of video camera sensors. It monitors the arrival and departure of vehicles in the individual lanes in order to record the traffic flow at an intersection. The Agent Controller processes this information, and ensure optimal control of lights at the intersection. This optimization is performed for all intersections a road network. Each agent cooperates with its neighboring agent controllers $(WA_{id}, NA_{id}, EA_{id}, SA_{id})$.

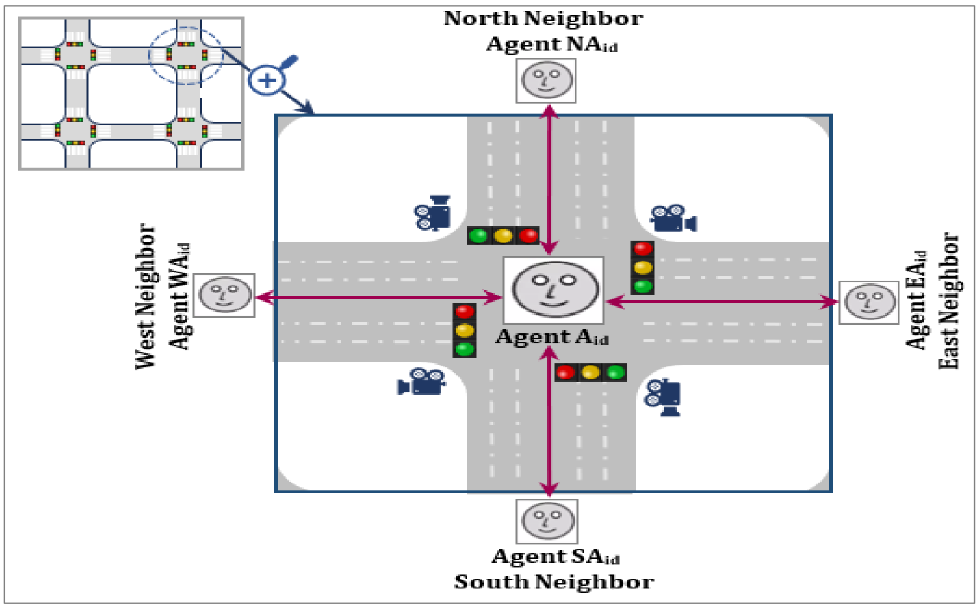


Figure 3.2 The multi-agent architecture for traffic-light control.

3. Assumptions

We propose a novel sensor-based system that addresses the following assumptions for synchronizing and controlling traffic lights at isolated intersection, and enables coordination between the set of intersections.

- All system components (sensors, traffic lights, control units, and video processing units) were supplied directly by the power supply. Each part is equipped with an emergency battery to ensure functionality when power is unavailable or exhausted. In such cases, each component informs the coordinator of the current energy state.
- We assume that each emergency vehicle and its priority level are identified by its lights.
- Each Main Traffic Signal Control unit (MTSCU) is monitored by another Secondary Traffic Signal Control Unit (STSCU), which allows the system to handle its failure appropriately. Failure is detected by a heartbeat mechanism that permits the replacement of the MTSCU by the STSCU. Therefore, the MTSCU becomes secondary after repair.
- The Main Video Processing Unit (MVPU) is monitored by another secondary unit (SVPU). In addition to MVPU components (MVCC, MTC, and MVPC) are monitored by other secondary components (SVCC, STC, and SVPC).
- The system is safe and reliable for traffic-light control and management. Each traffic light and video sensor were replicated to ensure resilience.

4. System Models and Traffic Metrics

In this section, we detail the intersection model, the cycle model, the phase model, and different metrics applied by the ADSTLS.

4.1 Intersection Model

ADSTLS is a system based on a standard model of a single intersection that consists of four vehicle lanes (north (N), south (S), east (E), and west (W)). Two movements per lane, going straight or turning left, from one direction to another. We assume that the right-turn movement does not interfere with the other road traffic movements. Therefore, we have eight moves, $wm = \{We, Wn, Ns, Ne, Ew, Es, Sn, Sw\}$, as shown in Figure 3.3. Each lane had a video sensor and three signal lights (one light per movement). The ADSTLS control system is deployed at a base station at an intersection and is connected to all system components.

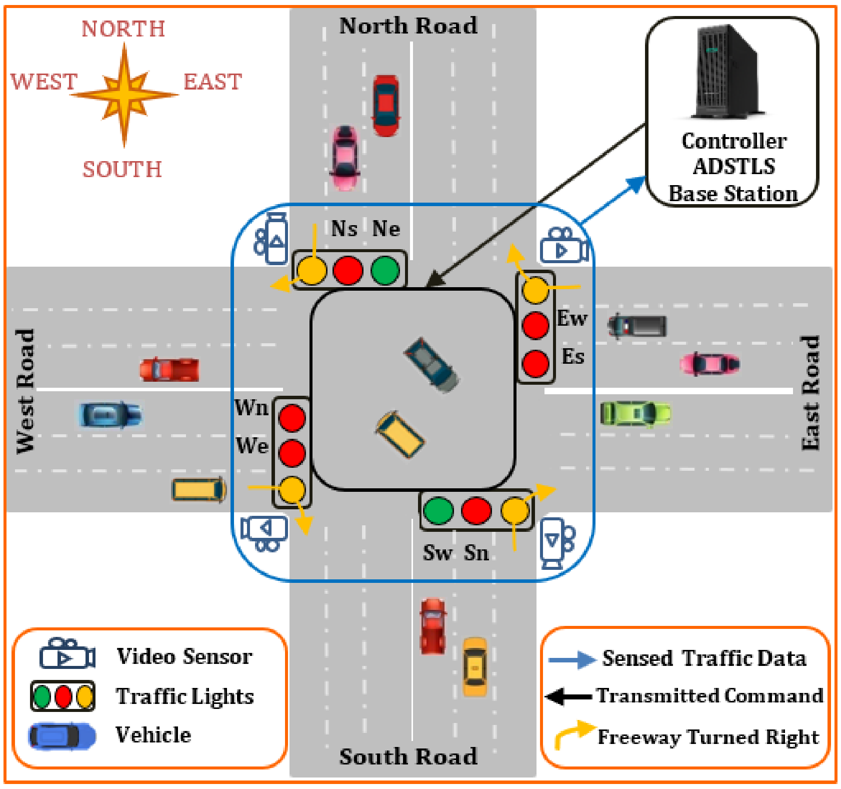


Figure 3.3 Intersection model.

4.2 Cycle and Phase Models

Our intersection model uses eight vehicle movements, *wm = {We, Wn, Ns, Ne, Ew, Es, Sn, Sw}* each of which competes with other movements to allow vehicles to travel through an intersection from one direction to another without conflict between them.

In traffic signal control systems, a cycle refers to a complete sequence of signal indications for all movements at an intersection or along a specific road section. It includes green, orange, and red intervals for each traffic direction as well as pedestrian intervals where appropriate. The cycle comprises the entire set of signal phases that are regularly used at an intersection to control traffic and pedestrian flow. A phase in the traffic light cycle denotes a specific period of time during which certain movements are allowed to proceed when the signal is green [130].

Based on the above standard definitions, ADSTLS uses a set of eight phases, *ph = {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw), (We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}*, where we define two split-phase cycle models that allow all intersection movements to pass. *Cycle 1 = {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw)}* and *Cycle 2 = {(We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}*.

It can be observed from the phase set that each direction belongs to two phases, which implies that each direction is coherent with only two other directions. Figure 3.4 shows the phase and cycle models used in the ADSTLS system.

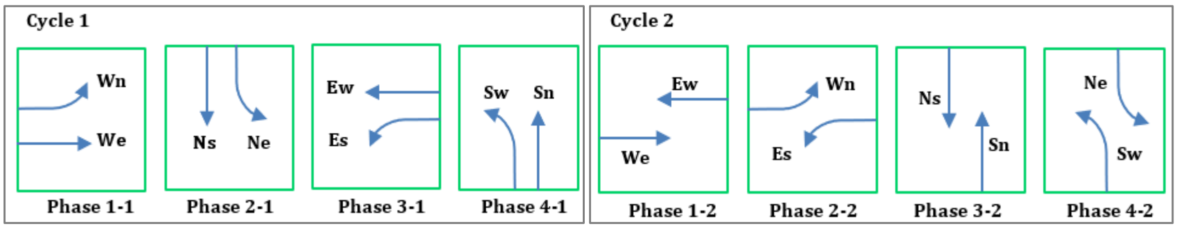


Figure 3.4 Cycle and phase models.

4.3 ADSTLS's Traffic Metrics

We define three types of traffic metrics: 1) vehicle, 2) queue, and 3) intersection.

4.3.1 Vehicle Metrics

Vehicle metrics can provide meaningful traffic information when evaluating system performance. These are applied to each vehicle v_i in queue k ($k \in wm$). First, we define the current system time at an intersection (t_c), which can be used for all vehicle metrics. These metrics are defined as follows:

- $d_{v_i}^{k}$: Distance of vehicle v_i from the traffic light.
- $in\_t_{v_i}^{k}$: Entering time of vehicle v_i through particular queue k as defined in (3.1).

$$in\_t_{v_i}^{k} = t_c \tag{3.1}$$

- $out\_t_{v_i}^{k}$: Departure time of vehicle v_i from a queue k defined using equation (3.2).

$$out\_t_{v_i}^{k} = t_c \tag{3.2}$$

- $pl_{v_i}^{k}$: Priority level of vehicle v_i. There are four priority levels (3, high; 2, medium; 1, low; and 0, no priority) as defined in equation (3.3).

$$pl_{v_i}^{k} = \{0, 1, 2, 3\} \tag{3.3}$$

- $c\_wt_{v_i}^{k}$: Current waiting time of vehicle v_i in a particular queue k, calculated using equation (3.4).

$$c\_wt_{v_i}^{k} = t_c - in\_t_{v_i}^{k} \tag{3.4}$$

- $wt_{v_i}^{k}$: Waiting time of vehicle v_i spent in a particular queue k, as defined in equation (3.5).

$$wt_{v_i}^{k} = out\_t_{v_i}^{k} - in\_t_{v_i}^{k} \tag{3.5}$$

4.3.2 Queue Metrics

These metrics are used to measure queue length based on the number of stopped vehicles in a single lane, vehicle waiting time, and traffic congestion, which are defined for each queue k with $k \in wm$ as follows:

- $q\_size_{max_k}$: Maximum queue size, is the maximum number of vehicles stopped in the lane within one cycle or phase light.
- d_{lv_k}: Distance from the latest vehicle stopped in the queue to traffic lights within one cycle or phase.
- in_k: Input queue flow represents the number of vehicles entering a particular queue k. It is initialized as zero and increases by one at each arrival of a vehicle in the queue, as calculated using equation (3.6).

$$in_k = in_k + 1 \tag{3.6}$$

- out_k: Output queue flow, where is the number of vehicles departing from queue k. It is initialized as zero and increases by one at each departure of the vehicle from the queue, as calculated in equation (3.7).

$$out_k = out_k + 1 \tag{3.7}$$

- q_size_k: Queue size in number of vehicles versus time, where $0 \leq q\_size_k \leq q\_size_{max_k}$ as defined using equation (3.8).

$$q\_size_k = in_k - out_k \tag{3.8}$$

- $occup_{r_k}$: Queue occupancy rate where $0 \leq occup_{r_k} \leq 1$. An occupancy rate close to one indicates that the queue has reached its maximum size. This metric was calculated using equation (3.9).

$$occup_{r_k} = \frac{q\_size_k}{q\_size_{max_k}} \tag{3.9}$$

- $aqwt_k$: Average queue waiting time of all waiting vehicles in queue k, calculated using equation (3.10).

$$aqwt_k = \frac{\sum_{i=1}^{q\_size_k} c\_wt_{v_i}^{k}}{q\_size_k} \tag{3.10}$$

- awt_k: Average waiting time of all outgoing and waiting vehicles in queue k, calculated using equation (3.11).

$$awt_k = \frac{\sum_{i=1}^{q\_size_k} c\_wt_{v_i}^{k} + \sum_{i=1}^{out_k} wt_{v_i}^{k}}{q\_size_k + out_k} \tag{3.11}$$

- ${n_{vp}}_k^{pl_v}$: Number of emergency vehicles with priority level pl_v in queue k. This metric is initialized to zero and increases by one at each arrival of an emergency vehicle in the queue, as shown in equation (3.12).

$$ {n_{vp}}_k^{pl_v} = {n_{vp}}_k^{pl_v} + 1 \qquad (3.12)$$

  However, when an emergency vehicle leaves, the number of emergency vehicles is reduced by one, as shown in equation (3.13).

$$ {n_{vp}}_k^{pl_v} = {n_{vp}}_k^{pl_v} - 1 \qquad (3.13)$$

- ${n_{vp}}_k$: Number of emergency vehicles in queue k calculated using equation (3.14).

$$ {n_{vp}}_k = \sum_{pl_v=1}^{3} {n_{vp}}_k^{pl_v} \qquad (3.14)$$

4.3.3 Intersection Metrics

Intersection metrics were used to evaluate the performance of the intersection. An identifier (id) is assigned to each intersection I_{id}, which contains k queues ($k \in wm$) defined by the following metrics:

- ${n_{vp}}_{id}$: Number of all emergency vehicles currently at the intersection I_{id}, calculated using equation (3.15).

$$ {n_{vp}}_{id} = \sum_{i=1}^{k} {n_{vp}}_i \qquad (3.15)$$

- aor_{id}: Average occupancy rate of all queues recorded at intersection I_{id}, defined in equation (3.16).

$$ aor_{id} = \frac{\sum_{i=1}^{k} {occup_r}_i}{k} \quad and \;\; 0 \leq aor \leq 1 \qquad (3.16)$$

- ofr_{id}: Output traffic flow rate in the intersection I_{id}, calculated using (3.17).

$$ ofr_{id} = \frac{\sum_{i=1}^{k} out_i}{\sum_{i=1}^{k} in_i} \quad and \quad 0 \leq ofr \leq 1 \qquad (3.17)$$

- awt_{id}: Average waiting time for all outgoing and waiting vehicles at the intersection I_{id}, defined using equation (3.18).

$$ awt_{id} = \frac{\sum_{i=1}^{K} \; awt_i}{k} \qquad (3.18)$$

4.3.4 Other Metrics

Other metrics express the time scale, vehicle speed, hybrid rate and coordination rate of ADSTLS, as defined by the following equations:

- t_s: The time scale normalizes the average waiting time (awt_{id}) of an intersection and average waiting time in each queue k ($aqwt_k$). These metrics are normalized using equations (3.19) and (3.20).

$$\overline{awt_{id}} = \frac{awt_{id}}{t_s} \quad with \quad 0 \leq \overline{awt_{id}} \leq 1 \quad (3.19)$$

$$\overline{aqwt}_k = \frac{aqwt_k}{t_s} \quad with \quad 0 \leq \overline{aqwt}_k \leq 1 \quad (3.20)$$

- s_v: Speed of vehicles during the simulation depending on the simulator configuration.
- h_r: Hybrid rate which is used in the dynamic hybrid model.
- c_r: Coordination rate is the parameter that triggers and control communication between intersection agents.

5. ADSTLS Control System for Single Intersection

STLS is a system that enables smarter urban traffic management. We anticipate that STLS will be affected by both traffic congestion and unpredictable traffic events that may disrupt traffic flow. However, guiding vehicles in the correct direction can be achieved using a combination of intelligent and flexible strategies. By selecting the optimal directions of the cycle, STLS can achieve efficient traffic management. We propose a weight-based intelligent traffic flow optimization system based on traffic intersection parameters to mitigate congestion and decrease vehicle waiting time. This optimization is based on the WCSO (Weight Chicken Swarm Optimization) algorithm to reduce both waiting time and occupancy rate and maximize the gain of emergency services, resulting in improved traffic management and efficiency.

To overcome the issues of effective traffic light management and improve quality of life, we define a 6-stages process to reduce traffic congestion and waiting times. The flowchart in Figure 3.5 shows the stages of the dynamic and adaptive smart traffic light control process. The flowchart begins with stage one (section 5.1), the early detection of failure components based on the heartbeat mechanism. Stage two (section 5.2) collects traffic data using camera sensors for pre-processing and interpretation. Stage three (section 5.3) ensures the security and privacy of data transmitted from the MVPU to the MTSCU based on asymmetric encryption protocol. In stage four (section 5.4), the decrypted traffic data is used to compute traffic metrics. In stage five (section 5.5), the traffic metrics from stage four are used to select the optimal phase based on WCSO and to determine the green time duration based on the presence or absence of emergency vehicles in adaptive or dynamic mode. In the sixth stage (section 5.6), the process ends with executing the traffic light control.

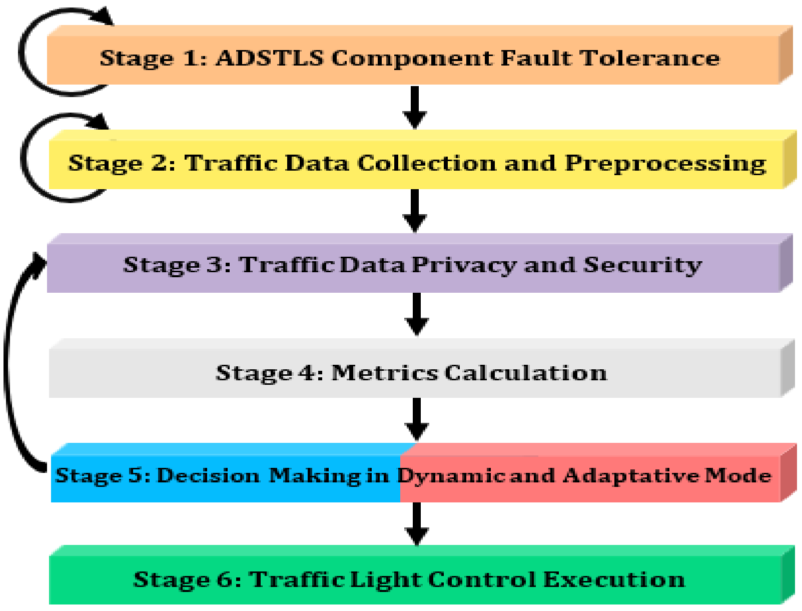

Stage 1: ADSTLS Component Fault Tolerance
Stage 2: Traffic Data Collection and Preprocessing
Stage 3: Traffic Data Privacy and Security
Stage 4: Metrics Calculation
Stage 5: Decision Making in Dynamic and Adaptative Mode
Stage 6: Traffic Light Control Execution

Figure 3.5 Functional model of ADSTLS.

5.1 Fault Tolerance

A heartbeat mechanism was used to monitor the ADSTLS primary components. Figure 3.6 shows the flowchart of this stage.

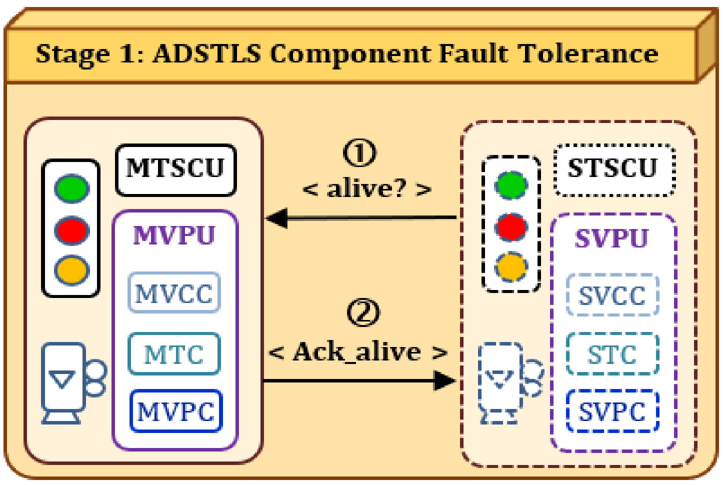


Figure 3.6 ADSTLS fault tolerance process.

First, the secondary component sends an (alive?) message to its corresponding primary component, and waits for acknowledgment. If, after Δt milliseconds, the secondary component does not receive a response from the primary component, the last component is declared to fail and is replaced. All secondary components in ADSTLS run the same algorithm. The algorithm takes the state of the primary component as input and returns a repair message to the city's global decision system when the primary component fails.

5.2 Traffic Data Collection and Preprocessing

This stage extracts and updates information on different queues and vehicles based on the traffic data extracted from the video processing component of the video processing unit. This information is sent to the traffic-light controller to make decisions and manage traffic flows. The pre-processing and interpretation of intersection signals consist of the following two steps.

- **Pre-processing**: The traffic input signals are initially subjected to pre-processing, where signal filtering occurs.
- **Interpretation**: The obtained pre-processed signals are given to extract traffic information, including 1) queue inflow, 2) queue outflow, 3) vehicle entry time in each queue, 4) vehicle exit time from each queue, 5) vehicle waiting time in queues, 6) priority level of each vehicle, 7) number of emergency vehicles with different priority levels in each queue, and 8) distance of the vehicle from the traffic light.

Figure 3.7 shows the collection and preprocessing of traffic data.

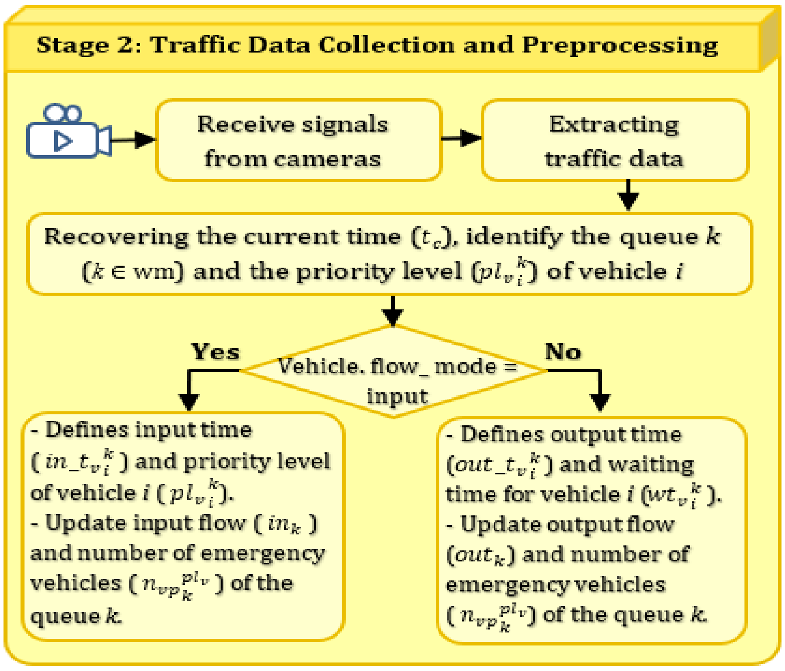

Stage 2: Traffic Data Collection and Preprocessing
Receive signals from cameras
Extracting traffic data
Recovering the current time (t_c), identify the queue k ($k \in$ wm) and the priority level ($pl_{v_i}^k$) of vehicle i
Vehicle. flow_ mode = input
Yes
No
- Defines input time ($in\_t_{v_i}^k$) and priority level of vehicle i ($pl_{v_i}^k$).
- Update input flow (in_k) and number of emergency vehicles ($n_{vp_k}^{pl_v}$) of the queue k.
- Defines output time ($out\_t_{v_i}^k$) and waiting time for vehicle i ($wt_{v_i}^k$).
- Update output flow (out_k) and number of emergency vehicles ($n_{vp_k}^{pl_v}$) of the queue k.

Figure 3.7 Traffic data collection and preprocessing process.

The ADSTLS system relies on an image processing model implemented using OpenCV and Java libraries to detect real-time video streams of a vehicle tracked by a Sahar camera plugged in the Raspberry PI. The vehicle detector is applied to every frame of a video stream of consecutive green time.

First, the cameras read all frames between consecutive green times. Then, the system converts every pair of RGB frames to grayscale and computes the intensity difference for every pair of frames. This sets the threshold for the difference in the output image. Then, the system performs a dilation on the thresholding image to detect “Region of Interest” and find contours in the output image of dilation. Next, the system detects the vehicle and draws contours around it in the detected vehicle in the original frame. After detecting the vehicle, the system classifies it as an "emergency vehicle" or "normal vehicle" by matching the shape of the detected vehicle with the shape class vehicle and extracting the vehicle number of every detected vehicle using Automatic Number Plate Recognition (ANPR) system [131]. The system computes the traffic information based on the extracted features of each vehicle in all frames and adds contours to all moving vehicles in all frames. Finally, the system saves the updated frames along with the computed traffic information and sends them to the controller.

To better control traffic lights during the passage of emergency vehicles, the emergency event broadcasting technique [132] is adopted. Based on a Vehicle Message Scheduling (VMS) scheme, no two vehicles or more have the same trust value. Thus, the selected vehicle that broadcasts emergency events is the vehicle that has the maximum trust value, minimum number of interferences, and its location close to the Road Side Unit (RSU). In the proposed system, the top selected vehicle on the road collects and broadcasts accident/emergency events in real-time.

5.3 Traffic Data Privacy and Security

In this section, we develop a robust protocol that ensures secure communication between the Main Video Processing Unit (MVPU) and Main Traffic Signal Control Unit (MTSCU). It is based on a shared generated key, to secure and protect traffic data, particularly video data from cameras to avoid persistent malicious attacks.

5.3.1 The Proposed Protocol

In our protocol, the MVPU and MTSCU components generate a shared secure key for encrypting traffic data and exchange them implicitly at each iteration to avoid the risk of attacks. To address the vulnerability of video capture and transmission, we exploited the GDPR policy with personal information hiding. However, we first detail the key exchange and image frame encryption, particularly the video data from cameras between the MVPU and the MTSCU, to better understand how the traffic data is encrypted/decrypted by both components with low computational complexity. This protocol consists of two verification steps: 1) MVPU checks whether the random value generated by the MTSCU is altered, and 2) MTSCU checks whether its value computed by the MVPU has been altered. The proposed protocol is described as follows.

- The MVPU generates a random value α and sends it to the MTSCU.
- The MTSCU generates a random key β, which is the encryption key. β is generated at each iteration.
- After receiving α, the MVPU calculates $(\beta^{\alpha}, \beta^{\beta})$.
- The MVPU computes β using α' , where $\alpha' = \big(p + i * (p-1)\big)/\alpha$ and β^{β} are used to check the value of β where p is close to 1.
- The MVPU then sends the computed β_1 using β to the MTSCU in the form of ${\beta_1}^{\alpha}$
- Finally, the MTSCU checks if ${\beta_1}^{\alpha}$!= β^{α} or not.

In our protocol, the MVPU wants to send secure traffic data from video to the MTSCU. Equation (3.21) shows how to encrypt image frame m from any video content using a secret key β^{α}:

$$img_e = img \cdot \beta^{\alpha} \tag{3.21}$$

During decryption, the MTSCU receives the encrypted image frame img_e and decrypts it using the secret key $\beta^{-\alpha}$ as given in equation (3.22). The decryption process must take place in reverse order to the encryption process.

$$img = img_e \cdot \beta^{-\alpha} \tag{3.22}$$

5.3.2 GDPR Policies and Regulations

We ensure the privacy and security of traffic data based on GDPR policy to protect video content. The GDPR has a unique feature related to the compliance check of sensitive and personal information vulnerable to illegal users, who may ensure their privacy and copyright.

The proposed system ensures compliance with the GDPR by providing policy reinforcement using Privacy Policy Based Access Control (P2BAC). This reinforcement is used by the MVPU and the MTSCU. They use computer-interpretable privacy policies and check them for compliance with the GDPR. In addition, privacy policies are used to determine which video content is considered in the authenticity and protection of personal data. However, policies can have an underlying policy that focuses on hiding personal information to minimize the issues of protection rights, integrity, and authenticity. It helps to inform end users or traffic data subjects about the collection and processing of traffic data transparently.

5.4 Computation of the System Metrics

In this stage, we compute all traffic metrics used in the decision-making process, either by vehicle, queue, intersection, or by other metrics. As detailed in Algorithm 3.1, the system applies different metrics (Equation (3.1)–Equation (3.20)) to provide the traffic information.

Algorithm 3.1 Computation of the system metrics.

```
Input:  $in_k$                : Input flow of the queue $k$ .
        $out_k$               : Output flow of the queue $k$.
        $in\_t_{v_i}^{k}$     : Input time of vehicle $v_i$ in queue $k$.
        $n_{vp_k}^{pl_v}$     : Number of emergency vehicles of priority level $pl_v$ in queue $k$.
        $t_c$                 : Current time of system.
        $q\_size_{max_k}$     : Maximum size of the queue $k$, $k \in wm$ and $wm = \{We, Wn, Ns, Ne, Ew, Es, Sn, Sw\}$.
        $t_s$                 : Time scale.
Output: $occup_{r_k}$         : Occupancy rate of the queue $k$. $k \in wm$ and $wm = \{We, Wn, Ns, Ne, Ew, Es, Sn, Sw\}$.
        $\overline{aqwt}_k$   : Normalized average queue waiting time of all vehicles currently in the queue $k$.
        $n_{vp_{id}}$         : Number of vehicles priority in the intersection $I_{id}$.
        $aor_{id}$            : Average occupancy rate for all queues in the intersection $I_{id}$.
Begin
 (1)  : Recover the current system time $t_c$
 (2)  : for each queue $k$ do
 (3)  :     Compute the current queue size ($q\_size_k$), using (3.8)
 (4)  :     Compute the queue occupancy rate ($occup_{r_k}$), according to (3.9)
 (5)  :     for each vehicle $v_i$ in queue $k$ do
 (6)  :         Calculate the current waiting time of vehicle $v_i$ in queue $k$( $c\_wt_{v_i}^{k}$), based on (3.4)
 (7)  :     end for
 (8)  :     Compute the average waiting time of all waited vehicles in queue $k$ ($aqwt_k$), with (3.10)
 (9)  :     Compute the normalized average queue waiting time of all waited vehicles in queue $k$ ($\overline{aqwt}_k$), using (3.20)
 (10) :     Compute the number of all emergency vehicles in lane $k$ ($n_{vp_k}$), according to (3.14)
 (11) : end for
 (12) : Compute the number of priority or emergency vehicles in the intersection $I_{id}$ ($n_{vp_{id}}$), following (3.15)
 (13) : Compute the average occupancy rate of all queues in the intersection $I_{id}$ ($aor_{id}$), according to (3.16)
End
```

Algorithm 3.1 clearly demonstrates the computation of most of the traffic metrics. It takes the maximum size of each queue, input flow of each queue, output flow of each queue, time scale, system time, vehicle entry time in each queue, and number of emergency vehicles per priority in each queue as inputs. It returns the occupancy rate of each queue, normalized average waiting time of all vehicles waiting in each queue, average occupancy rate of all queues in the intersection, and number of vehicles in the intersection as outputs. **Algorithm 3.1** begins by computing the occupancy rate of each queue ($occup_{r_k}$) and the normalized average waiting time of all the vehicles waiting in each queue ($\overline{aqwt_k}$). Subsequently, it computes the number of emergency vehicles at the intersection ($n_{vp_{id}}$) and the average occupancy rate of all the intersection lanes (aor_{id}).

5.5 Decision Making in ADSTLS Modes

The system passes the optimized phase and weights, and calculates the traffic metrics for the decision-making process. The process is performed in two modes: dynamic intelligence (Stage 5.a) and emergency adaptation (Stage 5.b). Upon congestion and emergency detection, the phase or cycle models are dynamically managed. However, the WCSO algorithm is employed to dynamically adjust the optimal weights, select the phases of each round, and send it to the controller.

5.5.1 Phase Selection and Weight Optimization

After calculating all traffic metrics, the ADSTLS system applies two new weight-based heuristics to select the best phase and queue to optimize traffic parameters (i.e., output flow rate of vehicles, average waiting time, and average occupancy) in dynamic mode, and the highest priority queue is used in emergency mode to determine priority and coherent directions. The optimization of the weights was based on an enhanced WCSO algorithm. It is a metaheuristic bio-inspired algorithm that imitates a chicken swarm hierarchy and individual chicken behavior. Our main goal is to select an efficient traffic model, smart up decision-making, and decrease the vehicle waiting time for high-traffic data.

a. Queue and Phase Selection Functions

Heuristic-based approaches are generally a great way to manage and optimize traffic flow [8, 14, 16, 18, 19, 23]. We propose two new selection functions as heuristics for determining the best phase in a traffic flow model that optimizes traffic parameters. These parameters are the Average Waiting Time (awt) of vehicles, Average Occupancy Rate (aor) of all intersection queues, and vehicle Output Flow Rate (ofr). The controller selects the optimal queue using a queue selection function and selects the best phase using a phase selection function, which is described as follows:

– **Queue selection function**: the queue selection function is defined as a linear combination of the queue occupancy rate ($occup_{r_k}$) and average vehicle waiting time ($aqwt_k$) of queue k. Furthermore, the queue with the highest queue waiting time and occupancy rate should be given a maximum score to be selected. The queue-selection function f_s_queue of queue k is calculated using equation (3.23).

$$f_s\_queue_k = w_q * occup_{r_k} + w_t * \overline{aqwt}_k \qquad (3.23)$$

Where:

- $k \in wm, wm = \{We, Wn, Ns, Ne, Ew, Es, Sn, Sw\}$.
- w_q : is weight value of occupation rate of queue k. $0 \leq w_q \leq 1$
- w_t : is weight value of average vehicles waiting time queue k. $0 \leq w_t \leq 1$
- $w_q + w_t = 1$.

– **Phase selection function**: the queue selection function is used to choose the phase that leads to the optimal management metric values awt, aor and ofr. The normalized phase selection function f_s_phase of phase (l, n) is calculated using equation (3.24).

$$f_s\_phase_{(l,n)} = (f_s\_queue_l + f_s\_queue_n)/2 \qquad (3.24)$$

Where:

- l, n: queue pair of phases ph with ph= {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw), (We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}.
- $f_s\_queue_l$: queue selection function for queue l,
- $f_s\_queue_n$: queue selection function of queue n.

b: Weights Optimization using WCSO Algorithm

This work aims to optimize the phase weights which are average waiting time and occupation rate using WCSO algorithm where CSO (Chicken Swarm Optimization) is the base algorithm. The reason for choosing CSO is its speed and performance. It is a bio-inspired metaheuristic algorithm that simulates a chicken swarm's hierarchy and individual chicken behavior. Their hierarchical order plays a critical role in the social lives of chickens. Several groups exist in the swarm hierarchy, each with a rooster, several hens, and chicks.

A set of chickens has its own movement pattern, with stronger chickens dominating weaker ones. Generally, dominant hens stay closest to head roosters, whereas submissive hens and roosters form the periphery of the group [133].

CSO mathematical formulations were defined based on the hen behavior using the following rules:

- The swarm is composed of several small groups controlled by a dominant rooster, followed by hens and chicks.
- The swarm hierarchy is based on the fitness values of the chickens. Chickens with the highest fitness are roosters, whereas those with the lowest fitness are chicks. The remaining chickens were hens.
- The hierarchy of swarm and mother-child relationships remains unchanged, whereas the status of the chickens in the swarm is updated at each time step.
- A chicken swarm composed of N virtual chickens was divided into roosters (RN), hens (HN), chicks (CN), and mothers (MN). Each individual represents a position in a D- dimensional space $x_{i,j}$ ($i \in [1..N\ ], j\ \in [1..D]$).

This stage provides the optimized phase selection weights defined in equation (3.24) using the bio-inspired optimization WCSO algorithm with the new objective function defined as follows:

$$f = max\ \left(f_{s\_}phase_{(l,\ n)}\right) \tag{3.25}$$

where $(l, n)\ \in$ ph, $w_q = w_{q_{optimal}}$ and $w_t\ =\ w_{t_{optimal}}$.

The bio-inspired optimization WCSO algorithm determines the optimal weight values for the next phase.

c: Movements of Chickens

Initially, position $\left(x_{i,j}^a\right)$ of chicken $c_{i,j}$ at time a is replaced by the positions of the weights:

$$x_{i,j}^a\ =\ W_{i,j}^a = ({w_q}_{i,j}^a, {w_t}_{i,j}^a) \tag{3.26}$$

Where:

- $x_{i,j}$: designates chicken $c_{i,j}$ with index i and group j,
- $x_{i,j}^a$: position of chicken at iteration a,
- a : iteration number.

The chicken movements are defined as follows:

- **Roosters:** Powerful roosters with higher fitness values can search for optimized weights in the D-dimensional space, as described in equations (3.27) and (3.28):

$$x_{i,j}^{a+1} = x_{i,j}^{a} * \left(1 + randn(0, \sigma^2)\right) \tag{3.27}$$

$$\sigma^2 = \begin{cases} 1, & if\ f_i \leq f_k, k \in [1..N], k \neq i \\ exp\left(\frac{f_k - f_i}{|f_i| + \varepsilon}\right), & otherwise, \end{cases} \tag{3.28}$$

Where:

- $x_{i,j}$: denotes the rooster $c_{i,j}$, $c_{i,j} \in (RN)$,
- $randn(0, \sigma^2)$: denotes the Gaussian distribution with mean 0 and standard deviation σ^2.
- ε : denotes the minimum constant used to overcome division-by-zero problems.
- k: is a selected index of a rooster and is randomly selected from the group of roosters (RN) and $k \neq i$,
- f_i: fitness value of the rooster.

– **Hens:** The roosters are followed by the hens to search for optimized weights. Hens also include a random movement towards the other hens' food (optimized weights), even though they would be recalculated by other hens. Dominant hens are more advantageous than submissive hens in competition for food, which can be mathematically explained by equations (3.29), (3.30), and (3.31):

$$x_{i,j}^{a+1} = x_{i,j}^{a} + S_1 * Rand * \left(x_{r_1,j}^{a} - x_{i,j}^{a}\right) + S_2 * Rand * \left(x_{r_2,j}^{a} - x_{i,j}^{a}\right) \tag{3.29}$$

$$S_1 = exp\left(\frac{f_i - f_{r_1}}{|f_i| + \varepsilon}\right) \tag{3.30}$$

$$S_2 = exp\left(f_{r_2} - f_i\right) \tag{3.31}$$

Where:

- $x_{i,j}$: Denotes the hen $c_{i,j}$, $c_{i,j} \in (HN)$,
- $Rand$: random numbers between 0 and 1,
- r_1: Rooster hen index, $r_1 \in (RN)$
- r_2: Index of the chicken (rooster or hen) chosen at random from the swarm, $r_2 \in [(RN) \cup (HN)]$
- ε: Minimum constant used to overcome division-by-zero problems.
- f_i: Fitness value of the hens.

– **Chicks:** The movement of chicks is limited to the following mother hens in their search for food, as represented by equation (3.32)

$$x_{i,j}^{a+1} = x_{i,j}^{a} + L * \left(x_{m,j}^{a} - x_{i,j}^{a}\right) \tag{3.32}$$

Where:

- $x_{m,j}^{a}$: Position of the chick's mother and m ∈ (MN),
- L : Chick speed parameter to track mothers, and the differences between chicks were calculated by choosing L randomly in the interval [0,2].

d: WCSO based Phase Selection Algorithm

Although the original CSO algorithms are the best for global solutions, they consume considerable time in some cases, thereby affecting the convergence performance. This problem is solved by adding a new parameter, inertia weight (W), when updating the positions of roosters, hens, and chicks. A new velocity update equation was iteratively used by the swarm optimization algorithm. This study introduced an inertia weight parameter in the updated equations (3.29) by (3.33) and (3.32) by (3.34), as shown below:

$$x_{i,j}^{a+1} = x_{i,j}^{a} + W(S_1 * Rand * (x_{r_1,j}^{a} - x_{i,j}^{a}) + S_2 * Rand * (x_{r_2,j}^{a} - x_{i,j}^{a})) \quad (3.33)$$

$$x_{i,j}^{a+1} = x_{i,j}^{a} + W * L * (x_{m,j}^{a} - x_{i,j}^{a}) \quad (3.34)$$

And:

$$W = W_{i,j}^{0}\, e^{-\beta a} \quad (3.35)$$

Where:

- $W_{i,j}^{0}$: initial position of the chicken $c_{i,j}$,
- β: Positive constant,
- a : Iteration number.

Figure 3.8 clearly shows the phase selection and optimization of weights based on the WCSO. It takes the occupancy rate of each queue, the normalized average waiting time of all vehicles currently in each queue, the set of N chicken, and the fixed values of the algorithm parameters, and returns the selected phase and associated weights. First, the system initializes N detailed weights of the phase selection function and defines the related parameters. The system then evaluates the weights of N chickens and the iteration equal zero. Furthermore, we determined the best global chickens with optimal weights. In this round, the system evaluates N weights and updates the best solution for the chicken and best global solution for the swarm. Subsequently, the weight fitness values are ranked and established in hierarchical order in the swarm.

The chickens were then divided into different groups and the relationship between the chicks and mother hens in each group was determined. The positions of all chickens are dynamically updated in the current iteration if the new position is better than the previous iteration. Finally, the selected phase and the associated optimal weights are returned.

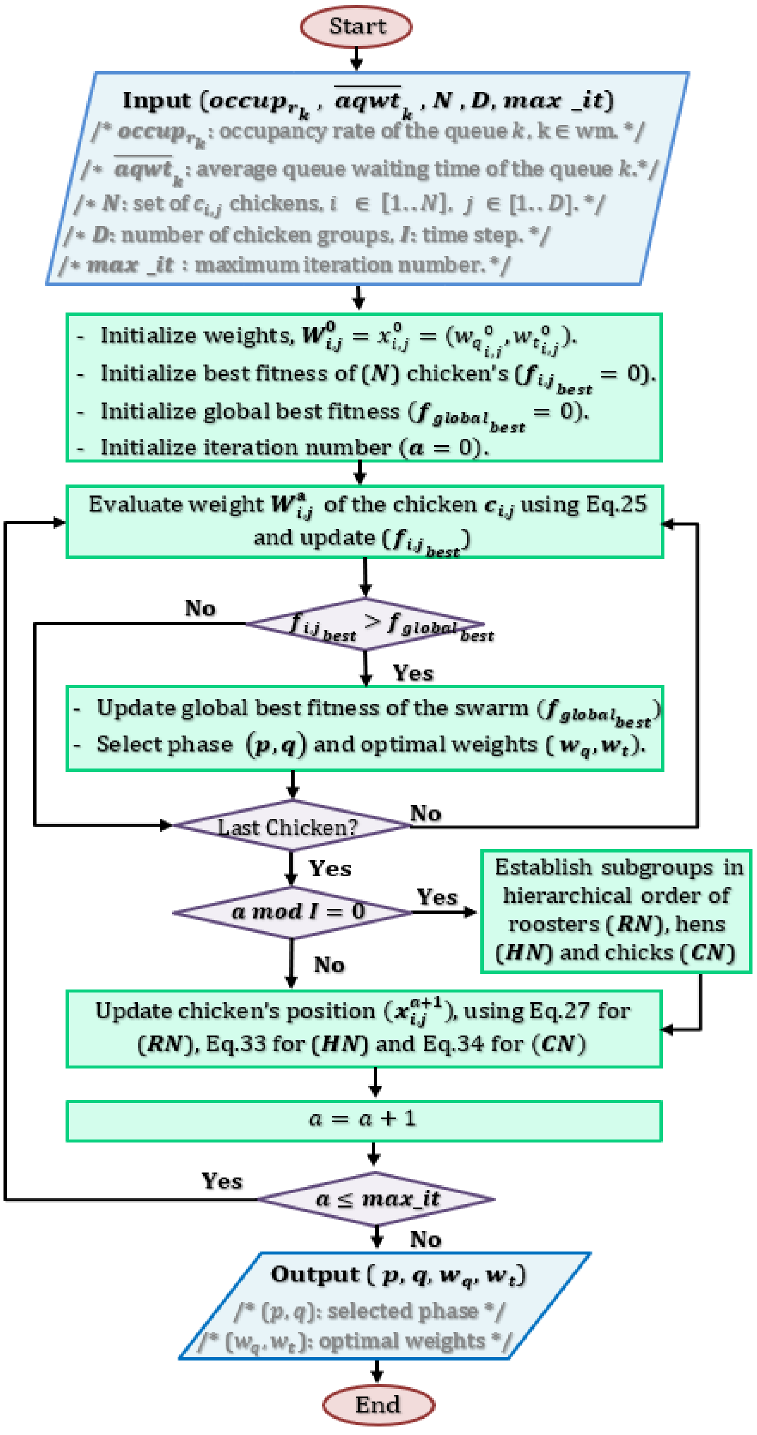

Start
Input $(occup_{r_k}, \overline{aqwt}_k, N, D, max\_it)$
/* $occup_{r_k}$: occupancy rate of the queue k, $k \in wm$. */
/* $\overline{aqwt}_k$: average queue waiting time of the queue k.*/
/* N: set of $c_{i,j}$ chickens, $i \in [1..N]$, $j \in [1..D]$. */
/* D: number of chicken groups, I: time step. */
/* max_it : maximum iteration number. */
- Initialize weights, $W^0_{i,j} = x^0_{i,j} = (w_{q^0_{i,j}}, w_{t^0_{i,j}})$.
- Initialize best fitness of (N) chicken's ($f_{i,j_{best}} = 0$).
- Initialize global best fitness ($f_{global_{best}} = 0$).
- Initialize iteration number ($a = 0$).
Evaluate weight $W^a_{i,j}$ of the chicken $c_{i,j}$ using Eq.25 and update ($f_{i,j_{best}}$)
$f_{i,j_{best}} > f_{global_{best}}$
No
Yes
- Update global best fitness of the swarm ($f_{global_{best}}$)
- Select phase (p, q) and optimal weights (w_q, w_t).
Last Chicken?
No
Yes
$a \bmod I = 0$
Yes
Establish subgroups in hierarchical order of roosters (RN), hens (HN) and chicks (CN)
No
Update chicken's position ($x^{a+1}_{i,j}$), using Eq.27 for (RN), Eq.33 for (HN) and Eq.34 for (CN)
$a = a + 1$
Yes
$a \leq max\_it$
No
Output (p, q, w_q, w_t)
/* (p, q): selected phase */
/* (w_q, w_t): optimal weights */
End

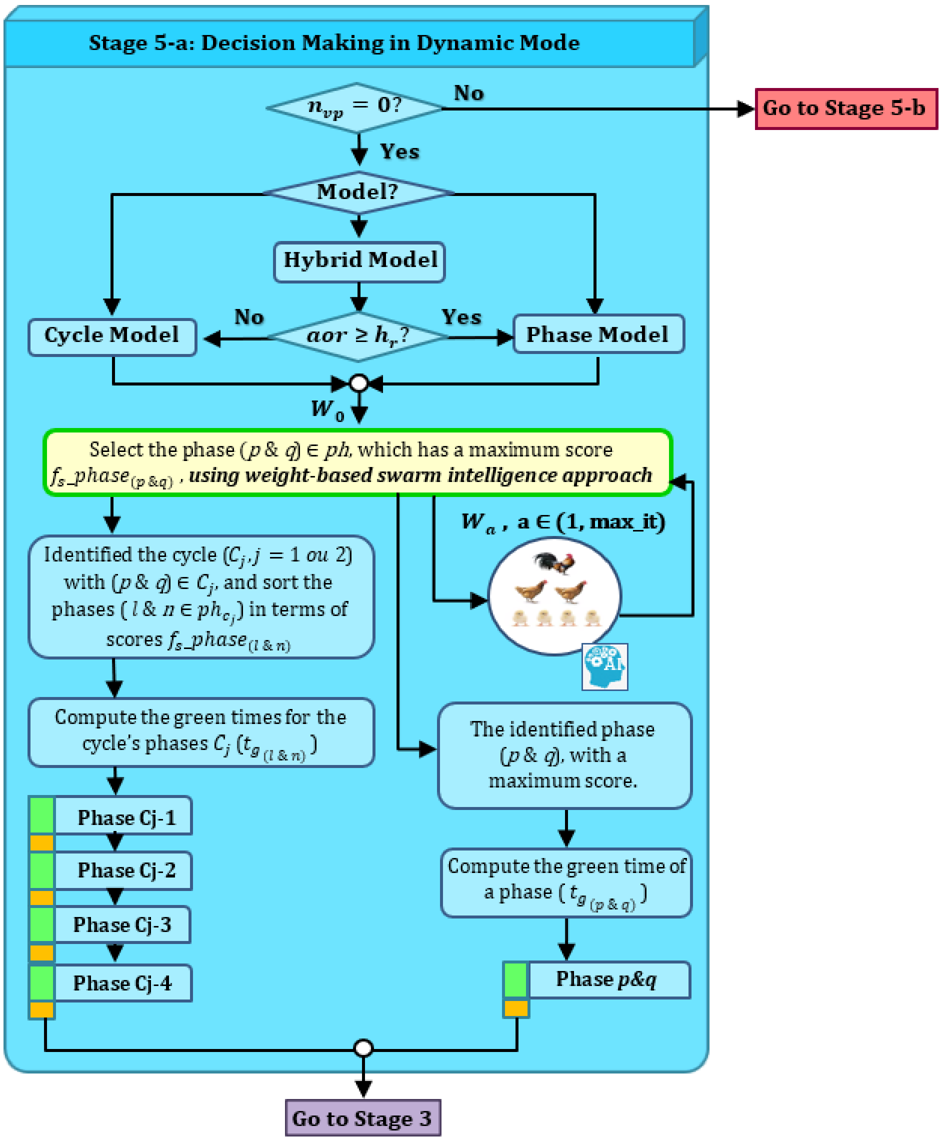
Figure 3.8 Flowchart of WCSO-based phase selection.

The proposed WCSO algorithm attempts to find the phase with the highest possible average waiting time of the vehicles and average occupancy rate of queues, which leads to finding the maximum value of the phase selection function. The optimal weights of the queue selection function w_t and w_q are also determined, so that we have an optimization problem with three variables (dimensions); that is, we work in a three-dimensional space. We further demonstrated the importance of optimal phase selection in stage 5.a (section 5.5.2) and stage 5.b (section 5.5.3).

5.5.2 WCSO based Dynamic and Intelligent Mode

The decision-making process is based on heuristics that consider traffic information (e.g., number of vehicles in each direction, their entry and exit times, and their coordinates) and select the phase and cycle in optimal and dynamic ways. The system estimates the traffic density and controls the green light based on traffic density. Thus, we suggest using the highest occupancy rate and the longest waiting time.

Figure 3.9 shows the ADSTLS decision making in dynamic and intelligent mode.



Stage 5-a: Decision Making in Dynamic Mode
$n_{vp} = 0$?
No
Go to Stage 5-b
Yes
Model?
Hybrid Model
Cycle Model
No
$aor \geq h_r$?
Yes
Phase Model
W_0
Select the phase (p & q) ∈ ph, which has a maximum score $f_s\_phase_{(p\,\&\,q)}$, using weight-based swarm intelligence approach
W_a, a ∈ (1, max_it)
Identified the cycle (C_j, j = 1 ou 2) with (p & q) ∈ C_j, and sort the phases (l & n ∈ ph_{c_j}) in terms of scores $f_s\_phase_{(l\,\&\,n)}$
Compute the green times for the cycle's phases C_j ($t_{g\,(l\,\&\,n)}$)
The identified phase (p & q), with a maximum score.
Compute the green time of a phase ($t_{g\,(p\,\&\,q)}$)
Phase Cj-1
Phase Cj-2
Phase Cj-3
Phase Cj-4
Phase p&q
Go to Stage 3

Figure 3.9 ADSTLS decision making in dynamic mode process.

To ensure greater dynamicity, we defined three decision-making models in the WCSO Dynamic and Intelligent Mode.

a. WCSO Dynamic and Intelligent Phase Model

This model aims to select the phase that will be executed next time based on two traffic metrics of each queue: waiting time and occupancy rate. The phase (p,q) is selected from eight existing models ph ($(p,q) \in ph$), based on the phase selection function defined by (3.24) and the objective function defined in (3.25). The duration of the green time of the traffic light in the selected phase was determined based on the speed of the vehicle and the distance of the last vehicle in this phase. After the green time, the positive constant duration of the orange times lights up at both traffic lights in queues p and q. The duration of the traffic light red time for the other queues $k \in \{wm - (p,q)\}$ was also calculated.

The distance ($d_{lv_{(p,q)}}$) is calculated as follows:

$$d_{lv_{(p,q)}} = max\left(d_{v_i}^{p}, d_{v_j}^{q}\right) \ for\ all\ \ v_i \in p\ and\ \ v_j \in q \quad (3.36)$$

where d_{lv_p} and d_{lv_q} are the distances of the last vehicle lv in queues p and q, respectively.

The green time of the traffic lights of queues p and q ($t_{g_{(p,q)}}$) is given by equation (3.37).

$$t_{g_{(p,q)}} = \frac{d_{lv_{(p,q)}}}{s_v} + c_g \quad (3.37)$$

Where $d_{lv_{(p,q)}}\ and\ \ s_v$ are the distances of the last vehicle lv in the selected phase (p,q) and the speed of vehicle v, respectively, and c_g is a positive constant that refers to the start-up time of the first vehicle in the queue.

The orange time for both traffic lights of queues p and q ($t_{o_{(p,q)}}$) is defined by equation (3.38).

$$t_{o_{(p,q)}} = c_o \quad (3.38)$$

Where c_o is a positive constant.

The duration of the traffic light red time (t_{r_k}) for the other queues $k \in \{wm - (p,q)\}$ is calculated using equation (3.39).

$$t_{r_k} = t_{g_{(p,q)}} + t_{o_{(p,q)}} \quad (3.39)$$

In this model, signal planning is organized into phases, that is, after the green time and during the orange time, and the next phase is determined and activated directly after the orange time. Figure 3.10 shows an example of signal planning in the WCSO's dynamic and intelligent phase model, which also illustrates that the green time depends on the distance to the last vehicle in the selected phase and that there are no overlaps between the phases.

A maximum value of green time means that one or both of the selected queues have a maximum size, and in pure cases where all queues are empty, we assign a constant positive value c_g as the green time in the last selected phase.

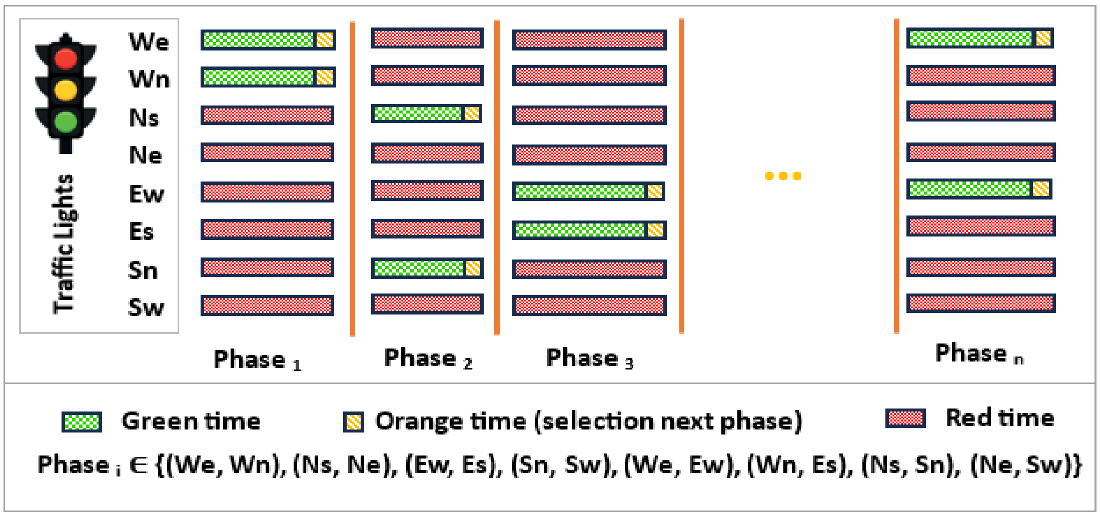

Traffic Lights
We
Wn
Ns
Ne
Ew
Es
Sn
Sw
Phase $_1$
Phase $_2$
Phase $_3$
...
Phase $_n$
Green time
Orange time (selection next phase)
Red time
Phase $_i$ ∈ {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw), (We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}

Figure 3.10 Signal planning in WCSO dynamic intelligent phase model.

b. WCSO Dynamic and Intelligent Cycle Model

This model aims to select during the orange time (t_o) of the fourth phase in the current cycle by calculating the duration of the cycle time for each queue, which includes a first red time, then a green time, followed by an orange time and ending with a second red time, after which the phases are triggered. This is based on each direction of the average waiting time of the vehicle and the queue occupancy rate. The green time duration for each phase is estimated based on the ratio between the distance of the last vehicle and the average speed of the vehicle at the intersection. The orange time is a positive constant value, and the first and second red times of a phase are calculated according to the execution order and green time of the other phases.

The WCSO selection function returns the selected phase (p, q) (called (p_1, q_1)) and the optimal values w_q, w_t. We determined the cycle model "$cycle_j$" (cycle$_1$ or cycle$_2$). If $\left(p_1, q_1\right) \in$ cycle$_1$ then cycle$_1$ is determined; otherwise, cycle$_2$ is determined as the execution cycle. Next, we calculate the value of the selection function of the other phases according to equation (3.24) and arrange the execution order of the phases in descending order according to the value of the selection function, starting with phase $(p_{1,}\ q_1)$, then $(p_{2,}\ q_2)$ to $(p_{3,}\ q_3)$ and ending with $(p_{4,}\ q_4)$.

The green time of the traffic lights of the two queues $p_{\mathrm{i},}$ and q_{i} $(t_{g\,(p_{i,}\ q_i)})$ is calculated according to equation (3.37).

The orange time for each of the two traffic lights of queues $p_{\mathrm{i},}$ and q_{i} ($t_{o\,(p_{i,}\ q_i)}$) is equal to a positive constant c_o, defined by equation (3.38).

The cycle time ($t_{cycle_j\,(p_{i,}\ q_i)}$) for a phase $(p_{i,}\ q_i)$ of the $cycle_j$ is calculated according to equation (3.40).

$$t_{cycle_j\,(p_{i,}\ q_i)} = t_{r1(p_{i,}\ q_i)} + t_{g\,(p_{i,}\ q_i)} + t_{o\,(p_{i,}\ q_i)} + t_{r2(p_{i,}\ q_i)} \tag{3.40}$$

Where $t_{r1(p_{i,}\ q_i)}$ is the first red time and $t_{r2(p_{i,}\ q_i)}$ is the second red time for each of the two queues p_i and q_i. With the signal planning of a $cycle_j$ we define these times as follows (3.41):

$$\left(\begin{array}{l}
\left\{t_{r1(p_{1,}\ q_1)} = 0 \,\middle|\, t_{r2(p_{1,}\ q_1)} = t_{g\,(p_{2,}\ q_2)} + t_{g\,(p_{3,}\ q_3)} + t_{g\,(p_{4,}\ q_4)} + c_o * 3\right\} \\
\left\{t_{r1(p_{2,}\ q_2)} = t_{g\,(p_{1,}\ q_1)} + c_o \,\middle|\, t_{r2(p_{2,}\ q_2)} = t_{g\,(p_{3,}\ q_3)} + t_{g\,(p_{4,}\ q_4)} + c_o * 2\right\} \\
\left\{t_{r1(p_{3,}\ q_3)} = t_{g\,(p_{1,}\ q_1)} + t_{g\,(p_{2,}\ q_2)} + c_o * 2 \,\middle|\, t_{r2(p_{3,}\ q_3)} = t_{g\,(p_{4,}\ q_4)} + c_o\right\} \\
\left\{t_{r1(p_{4,}\ q_4)} = t_{g\,(p_{1,}\ q_1)} + t_{g\,(p_{2,}\ q_2)} + t_{g\,(p_{3,}\ q_3)} + c_o * 3 \,\middle|\, t_{r2(p_{4,}\ q_4)} = 0\right\}
\end{array}\right) \tag{3.41}$$

In this model, signal planning is organized in cycles. After each green phase, a fixed orange time period begins. During the green and orange periods of the current phase, the other phases are highlighted in red. Figure 3.11 shows examples of several successive cycles.

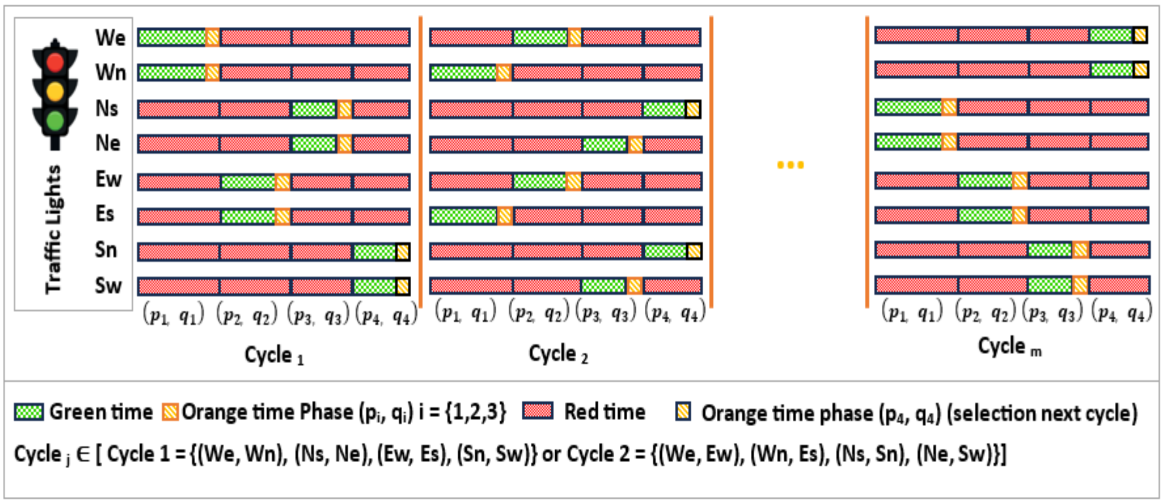

Traffic Lights
We
Wn
Ns
Ne
Ew
Es
Sn
Sw
(p1, q1) (p2, q2) (p3, q3) (p4, q4)
Cycle 1
Cycle 2
Cycle m
...
Green time
Orange time Phase (pi, qi) i = {1,2,3}
Red time
Orange time phase (p4, q4) (selection next cycle)
Cycle j ∈ [Cycle 1 = {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw)} or Cycle 2 = {(We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}]

Figure 3.11 Signal planning in WCSO dynamic intelligent cycle model.

c. WCSO Dynamic and Intelligent Hybrid Model

This model combines two previous models: the WCSO Cycle model and WCSO Phase model. It aims to improve system performance and reduce computation costs when traffic is fluid. Thus, we included the average queue occupancy rate in the decision-making process (aor). This will help the system to measure intersection traffic density. Hence, in the decision-making process, if the average queue occupancy rate is higher than the hybrid rate h_r (i.e., dense traffic), the WCSO Phase model is applied. Otherwise, the WCSO Cycle model is used, as shown in Figure 3.9.

In this model, signal planning is hybrid, it is organized in phases when traffic is dense ($aor_{id} \geq h_r$), otherwise in cycles ($aor_{id} < h_r$). Figure 3.12 shows examples of the successive phases and cycles.

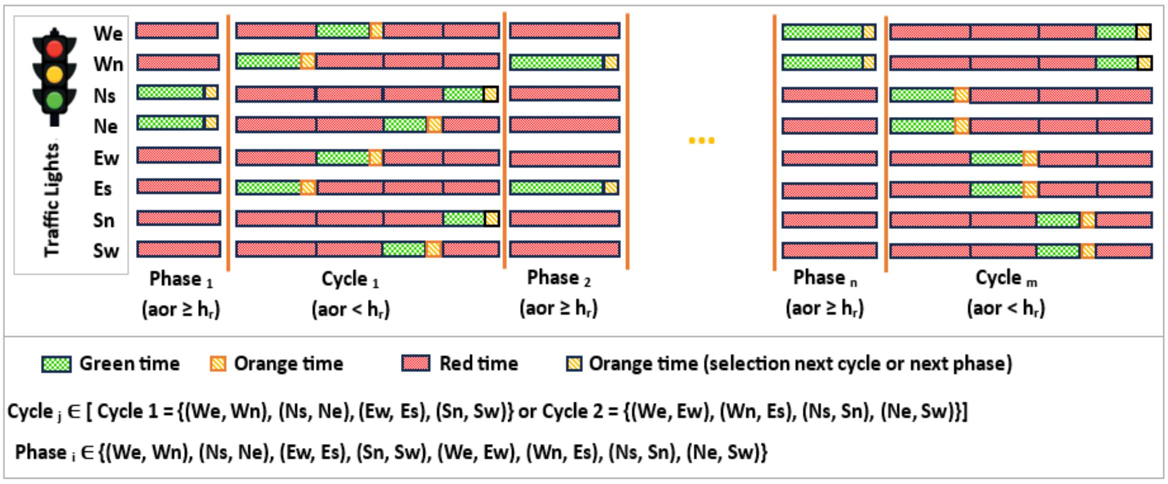

Traffic Lights
We
Wn
Ns
Ne
Ew
Es
Sn
Sw
Phase 1 (aor ≥ hr)
Cycle 1 (aor < hr)
Phase 2 (aor ≥ hr)
...
Phase n (aor ≥ hr)
Cycle m (aor < hr)
Green time
Orange time
Red time
Orange time (selection next cycle or next phase)
Cycle j ∈ [Cycle 1 = {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw)} or Cycle 2 = {(We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}]
Phase i ∈ {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw), (We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}

Figure 3.12 Signal planning in WCSO dynamic intelligent hybrid model.

Algorithm 3.2 clearly demonstrates how the proposed ADSTLS controller works in both the dynamic and intelligent modes. After receiving the optimal weights $(w_q, w_t)_{optimal}$ and the selected phase (p, q) returned by WCSO phase selection function, the controller calculates the green and orange times ($t_{g_{(p,q)}}$, $t_{o_{(p,q)}}$) of the selected phase (p, q), and the red time of the other queues k ($k \in \{wm - (p, q)\}$), for two models: phase or hybrid when $aor_{id} \geq h_r$. For other models, cycle or hybrid, when $aor_{id} < h_r$, the controller first identifies and aggregates the phases in $Cycle_j$ that are included in the selected phase (p, q). Furthermore, the cycle phases (p_i, q_i) with the highest score were chosen by the controller. In this round, the controller sorts the phases of each cycle in the ascending order using a selection function. Finally, the cycle time is computed to schedule the next cycle.

Algorithm 3.2 Decision-making in dynamic intelligent mode based WCSO.

```
Input:  $model$ : Dynamic and intelligent mode model, model = {cycle, phase, hybrid}.
        $(p, q)$ : Selected phase returned by the WCSO Phase Selection Function.
        $w_{q_{optimal}}$ : Optimal weight of queue occupation rate for selected phase.
        $w_{t_{optimal}}$ : Optimal weight of waiting time for selected phase.
        $d_{v_i}^{k}$ : Distance of vehicle $v_i$ in queue $k$. $k \in$ wm and wm $=$ $\{We, Wn, Ns, Ne, Ew, Es, Sn, Sw\}$.
        $aor_{id}$ : Average occupancy rate of all queues in the intersection $I_{id}$.
        $s_v$ : Speed of the vehicles.
        $h_r$ : Hybrid rate.
        $c_g, c_o$ :Two different positive constants.
Output: 
        | $(p, q)$ : Next phase to be executed.
        | $t_{g_{(p,q)}}$ : Green time of the selected phase.
        | $t_{o_{(p,q)}}$ : Orange time of the selected phase.
        | $t_{r_k}$ : Red time for the other queues $k \in \{wm - (p,q)\}$.
        Or
        | $cycle_j$ : Next cycle to be executed contains the phases $ph_{cycle_j}$ ,
        |           Such as: $(p, q) \in Cycle_j$ and $j = 1$ ou 2.
        | $t_{cycle_j\,(p_i,\, q_i)}$ : Cycle time for all phases $(p_i,\ q_i)$ of the selected cycle $C_j$.
Begin
(1)  : Call WCSO Phase Selection Function.
(2)  : if ( $model = phase$ ) or ( ( $model = hybrid$ ) and ( $aor_{id} \geq h_r$ ) ) then
(3)  :     Finds the distance ( $d_{lv(p,q)}$ ) of the last vehicle in two queues p and q using equation (3.36).
(4)  :     Calculates the green time ( $t_{g_{(p,q)}}$ ) of the phase $(p, q)$ based on equation (3.37), and the orange time
           ( $t_{o_{(p,q)}}$ ) using equation (3.38).
(5)  :     Calculates the red time for the other queues $k \in \{wm - (p,q)\}$, according to equation (3.39).
(6)  : endif
(7)  : if [ $model = cycle$ ] or [ ( $model = hybrid$ ) and ( $aor_{id} < h_r$ ) ] then
(8)  :     if $(p \,\&\, q) \in ph_{cycle_1}$ then // cycle model identification
(9)  :         $Cycle_1$: $ph_{cycle_1}$ = {(We & Wn), (Ns & Ne), (Ew & Es), (Sn & Sw)}   // cycle 1 model
(10) :     else
(11) :         $Cycle_2$: $ph_{cycle_2}$ = {(We & Ew), (Wn & Es), (Ns & Sn), (Ne & Sw)}   // cycle 2 model
(12) :     endif
(13) :     Computes $f_s\_phase_{(p_i,\, q_i)}$ where $(p_i,\ q_i) \in ph_{cycle_j}$ using equation (3.24) and optimized weights
           $(w_q, w_t)_{optimal}$;
(14) :     Sorts the phases $(p_i,\ q_i) \in ph_{cycle_j}$ of the $Cycle_j$ in descending order by $f_s\_phase_{(p_i,\, q_i)}$ ;
(15) :     for each phase $(p_i,\ q_i) \in ph_{cycle_j}$ of the $Cycle_j$ do
(16) :         Finds the distance $d_{lv(p_i,\, q_i)}$ of the last vehicle in the two queues $p_i$ and $q_i$ according to (3.36);
(17) :         Calculates the cycle time $t_{cycle_j\,(p_i,\, q_i)}$ of the phase, based on equations (3.40) and (3.41);
(18) :     end for
(19) : endif
End
```

5.5.3 WCSO based Adaptive Emergency Mode

For managing emergency vehicles, we extend the WCSO DIM Phase model with different values of priority levels: '3': High, '2': Medium and '1': Low. When emergency vehicles (e.g., ambulances, firefighters, and police) automatically enter an intersection, the system switches to adaptive emergency mode. If the green light is already in a certain direction, the signal remains until the emergency vehicle passes. Otherwise, the green light switches to emergency vehicle direction.

Figure 3.13 shows the ADSTLS decision making in adaptive emergency mode.

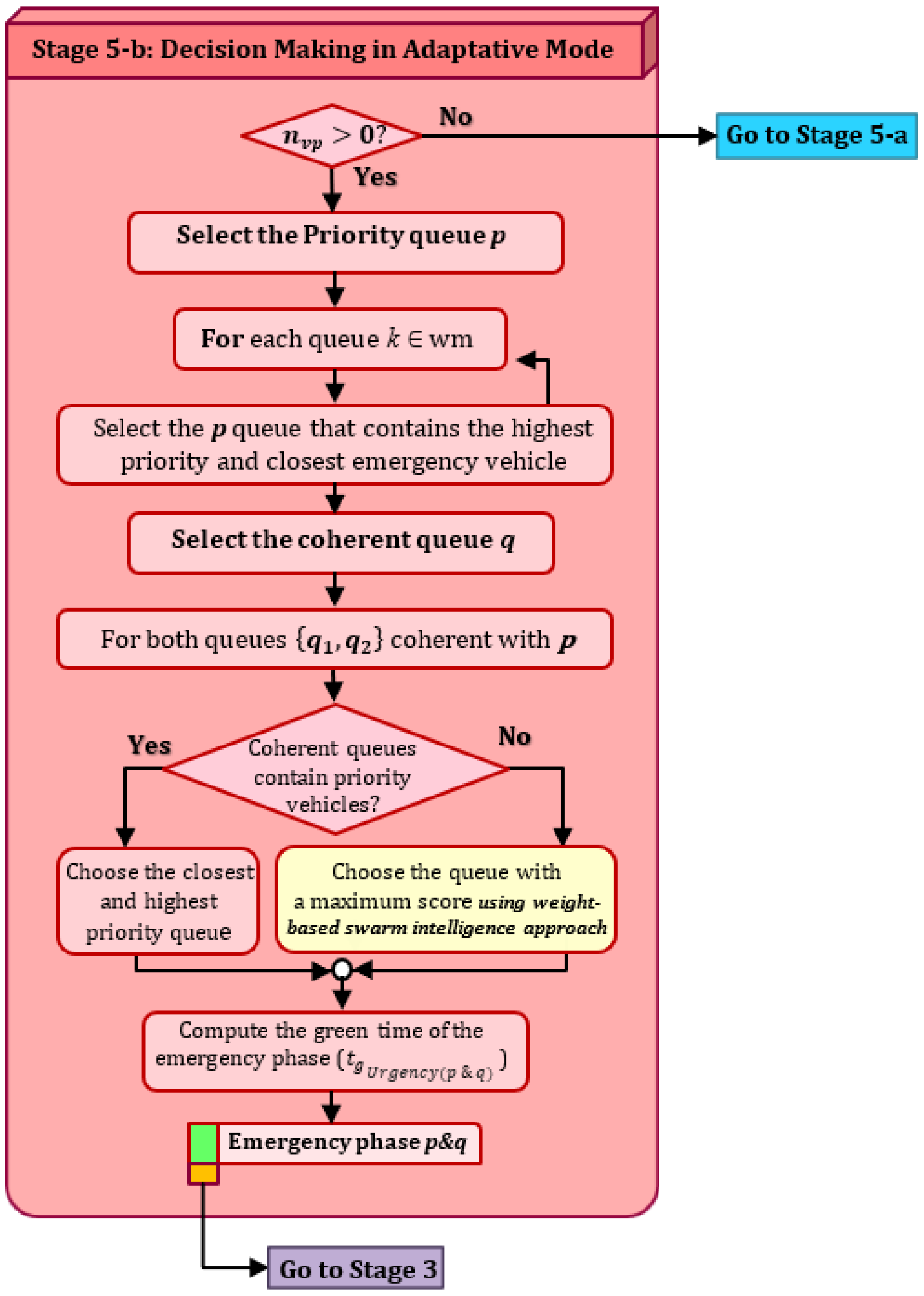

Stage 5-b: Decision Making in Adaptative Mode
$n_{vp} > 0$?
No
Go to Stage 5-a
Yes
Select the Priority queue p
For each queue $k \in$ wm
Select the p queue that contains the highest priority and closest emergency vehicle
Select the coherent queue q
For both queues $\{q_1, q_2\}$ coherent with p
Yes
Coherent queues contain priority vehicles?
No
Choose the closest and highest priority queue
Choose the queue with a maximum score using weight-based swarm intelligence approach
Compute the green time of the emergency phase ($t_{g_{Urgency(p\,\&\,q)}}$)
Emergency phase p&q
Go to Stage 3

Figure 3.13 ADSTLS decision making in adaptive mode process.

Our intelligent adaptive strategy consists of selecting two coherent directions, as described below.

a. Priority Direction Selection

The system selects the priority direction based on emergency vehicle distance and priority levels. When there are many emergency vehicles in the same direction with the same priority level, the system selects the direction with the closest emergency vehicles. Otherwise, it selects the vehicle with the closest and higher-level vehicles.

b. Coherent Direction Selection

The system selects the second consistent direction as the first among two possible directions. If one or both directions contain emergency vehicles, the system determines the direction that includes the closer and higher-level vehicle. Otherwise, the system selects the direction that maximizes the selection function.

c. Green Urgency Time

To ensure a prompt response before the arrival of another priority vehicle at the intersection, the green urgency time ($t_{g_{urgency\,(p,q)}}$) was computed based on the distance of the emergency vehicle ($d_{v_{urgency}}$) from the priority direction to the traffic light. After emergency vehicles have passed, the dynamic intelligent mode is automatically switched.

Algorithm 3.3 clearly shows the ADSTLS controller algorithm designed in adaptive emergency mode. In this algorithm, the controller selects the optimal emergency phase and its green time based on the WCSO algorithm. First, the controller selects the highest priority direction p that is closest to the urgent vehicle. In addition, it selects direction q with the nearest higher-priority vehicle from two coherent directions q_1, q_2 with p if one of them contains an emergency vehicle. Otherwise, it calls WCSO Phase Selection Function to return optimal weights $(w_q, w_t)_{optimal}$. After receiving the optimal weights, the controller selects direction q from the two coherent directions q_1, q_2 with p which maximizes the score based on the phase selection function $f_{s_{phase}}$. Finally, the green emergence and orange times of the nearest higher-priority vehicle in the priority direction (p, q) are computed and assigned to the next round, as is the red time of the other queues k ($k \in \{wm - (p, q)\}$).

Algorithm 3.3 Decision-making in adaptive emergency mode based WCSO.

```
Input:  $d_{v_i}^{k}$            : Distance of vehicle $v_i$ in queue $k$. $k \in$ wm .
        $pl_{v_i}^{k}$           : Priority level of vehicle$v_i$ in queue $k$. $pl_v = \{0, 1, 2, ou\ 3\}$.
        $n_{vp_k}^{pl_v}$        : Number of emergency vehicles with priority level $pl_v$ in queue $k$. $pl_v = \{1, 2, et\ 3\}$.
        $w_{q_{optimal}}$        : Optimal weight of queue occupation rate for selected phase.
        $w_{t_{optimal}}$        : Optimal weight of waiting time for selected phase.
        $s_v$                    : Speed of the vehicles.
        $c_g, c_o$               : Two different positive constants.
Output: urgency (p & q) : The priority phase selected by the controller.  (p & q) $\in ph$
        $t_{g_{urgency\,(p\,\&\,q)}}, t_{o\,(p\,\&\,q)}$ : Green and orange time of the priority phase selected.
        $t_{r_k}$                : Red time for the other queues $k \in \{wm - (p, q)\}$.
Begin
 (1)  : for each queue $k \in wm$ do      // Select priority direction p
 (2)  :     Computes number of all emergency vehicles $n_{vp_k}$ in the queue $k$ using equation (3.14);
 (3)  :     if $(n_{vp_k} > 0)$ and ( $k$ contains the highest priority $(pl_{v_i}^{k})$ and closest vehicle $(d_{v_i}^{k})$) then
 (4)  :         $p = k$;
 (5)  :         $d_{v_{urgency}} = d_{v_i}^{k}$;
 (6)  :     endif
 (7)  : end for
 (8)  : for tow coherent queues n = $\{q_1, q_2\}$ do   // Select coherent direction q, if it contains a priority vehicle.
 (9)  :     Computes number of all emergency vehicles $n_{vp_n}$ in the queue $n$ equation (3.14);
 (10) :     if $(n_{vp_n} > 0)$ and ($n$ contains the highest priority $(pl_{v_i}^{n})$ and closest vehicle $(d_{v_i}^{n})$) then
 (11) :         $q = n$
 (12) :     endif
 (13) : end for
 (14) : if $(n_{vp_{q_1}} = 0)$ and $(n_{vp_{q_2}} = 0)$ then // Select coherent direction q, if it does not contain a priority vehicle.
 (15) :     Call WCSO Phase Selection Function.
 (16) :     Computes the tow selection functions $f_{s\_}phase_{(p\,\&\,q_1)}$ , $f_{s\_}phase_{(p\,\&\,q_2)}$,
            using Eq. (3.24) and $w_{t_{optimal}}, w_{q_{optimal}}$
 (17) :     if $(f_{s\_}phase_{(p\,\&\,q_1)} > f_{s\_}phase_{(p\,\&\,q_2)})$ then
 (18) :         $q = q_1$
 (19) :     else
 (20) :         $q = q_2$
 (21) :     end if
 (22) : end if
 (23) : Computes green time $t_{g_{urgency\,(p\,\&\,q)}}$ of the urgency phase (p & q), based on equation (3.37) ; and the
        orange time $(t_{o_{(p,q)}})$ using equation (3.38).
 (24) : Computes the red time for the other queues $k \in \{wm - (p, q)\}$, according to equation (3.39).
End
```

5.6 Traffic Light Control Execution

By defining the phase or cycle selected in the previous section with green, orange and red times, the traffic signal lights can be switched from one state to another in a specific sequence. Each state is designed to control traffic flow safely and efficiently. Below, we will find a detailed explanation of the transition process from one state to another:

- **Green traffic light status**: Allows vehicles to pass through the intersection. It is triggered at the end of the green light phase.
- **Orange traffic light status**: Warns drivers that the traffic light will soon turn red. Vehicles should prepare to stop. It is triggered at the end of the green light phase.
- **Red traffic light status**: All vehicles stop in the corresponding direction so that cross traffic can proceed. It is triggered at the end of the orange light phase.

6. Chronological Coordination of ADSTLS at Multiple Intersections

In this section, we propose an extension of our control system called CCADSTLS (chronological coordination of ADSTLS) for reducing congestions at multiple intersections. It involves the ability of traffic control agents to autonomously identify the source of congestion early so that optimal, cost-effective decisions can be made to keep traffic flow smoothly. The system ensures the safety and efficiency of the intersections under severe weather fluctuations.

6.1 WCSO-Based Agent Intersection Controller

Each traffic-light controller is an agent consisting of Modules: Traffic Data Collector (TDC), Local Database (LDB), Decision Making (DM), and Traffic Communication (TCM), as shown in Figure 3.14. The TDC module processes the traffic data collected by cameras and sensors and stores it in a local database. The DM module relies on the collected data and operates in four modes: regular, emergency, congested, and weather disaster. The DM module in regular mode optimizes traffic flow in the intersection based on two main metrics, the queue occupancy rate ($occup_{r_k}$) and average vehicle waiting time ($aqwt_k$) of queue k. The DM uses a phase model to make decisions after each executed phase using real-time traffic data provided by video cameras. The traffic measurements are estimated and the signal is set to green based on the bio-inspired optimization WCSO algorithm with the objective function f, defined by equation (3.25).

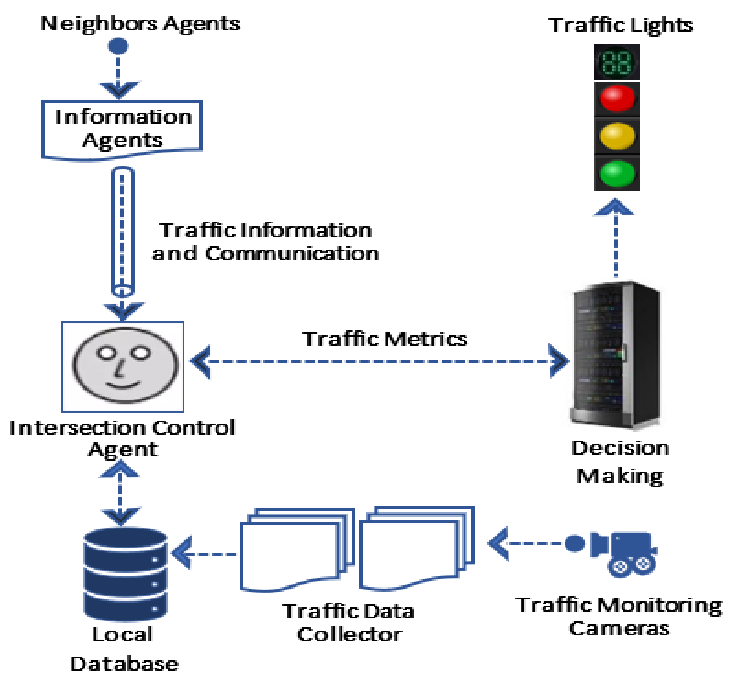


Figure 3.14 Modules of WCSO based agent intersection controller.

6.2 Chronological Coordination of WCSO Agent Controllers

The traffic consists of some control parameters being passed to the traffic simulator to control some traffic lights, or the agent communicates some traffic information to another agent for coordination control using the TCM module. These control agents operate in a traffic flow system that involves continuous interactions between them via request and response messages. The TCM module of the agent controller A_{id} uses six types of messages to manage congestion and weather disasters ($coordination$, $congestion\ search$, $congestion\ resolving$, $blockage\ search$, $out\ of\ service$ and $return\ to\ service$).

In general, agent controllers are mainly designed to optimize average waiting time (awt) and average occupancy rate (aor) during congestion and weather disaster modes taking into account the potential safety risks. The coordination of agent controllers consists of two main processes:

6.2.1 Coordination Process in Congestion Mode

Figure 3.15 shows the flowchart of chronological coordination process in congestion mode, which runs before the selected phase is switched to green by an agent A_{id} using the following steps:

a. Call WCSO selection function which returns the selected phase $(p, q) \in ph_{Cycle\ 1}$ that should be green.

b. Compute average occupancy rate ($aor_{I_{id}}$) of all roadways recorded at the intersection I_{id}, using equation (3.16).

c. The agent A_{id} broadcasts a coordination message (msg_{Coord}) with the current average occupancy rate ($aor_{I_{id}}$) at time $t_{e_c}$to all neighboring intersections. It then receives coordination messages (msg_{Coord}) at t_{r_c} from its neighbors containing occupation rates ($aor_{JI_{id}}$) and updates this information in its local database.

d. Depending on the average occupancy rate $aor_{I_{id}}$ of intersection I and the reception of congested messages from its neighboring intersections, we study the following two cases:

   - Case 1: ($aor_{I_{id}} > c_r$ and $aor_{I_{id}} >$ max ($aor_{JI_{id}}$) and mode = $regular$) when the occupation rate is greater ($aor_{I_{id}}$) than coordinate rate (c_r) and the intersection has a higher occupation rate to manage more traffic flow, the incoming traffic flow would be controlled carefully.
   - Case 2: the agent A_{id} receives a congested message from its neighbors (msg_{Cong} = $CP_{JI_{id}}$, $t_{r_{sc}}$)

   In both cases, the intersection control agent switches to congestion mode, updates the congested path (CP), sets the selected phase to green, and sends a congestion message (msg_{Cong} = $CP_{I_{id}}$, $t_{e_{sc}}$) to neighbors on the side of optimal phase of intersection control agent (N_{id}).

e. Forward iteratively congested path to next intersection control agent N_{id}. Repeat steps 'd' and 'e' until a city entrance/exit is reached. Otherwise, Goto Step 'f'.

f. Penalize and set the selected phase of exit/entry intersection to red and select the second phase as green.

g. Release the intersections from the last and start nodes in the opposite directions of the congested path. It sends congestion-solving messages (msg_{solve} = $CP_{I_{id}}$, $t_{e_{cr}}$), to its predecessor agent which is specified in the congested path and returns to regular mode.

To avoid blocking the CCADSTLS system, it does not consider crossed congested paths, i.e. a crossing is displayed when a single path is resolved, and if a second message is received to search for another congested path, it is answered by a blocking search message (msg_{block} = $CP_{JI_{id}}$,$t_{e_{bc}}$).

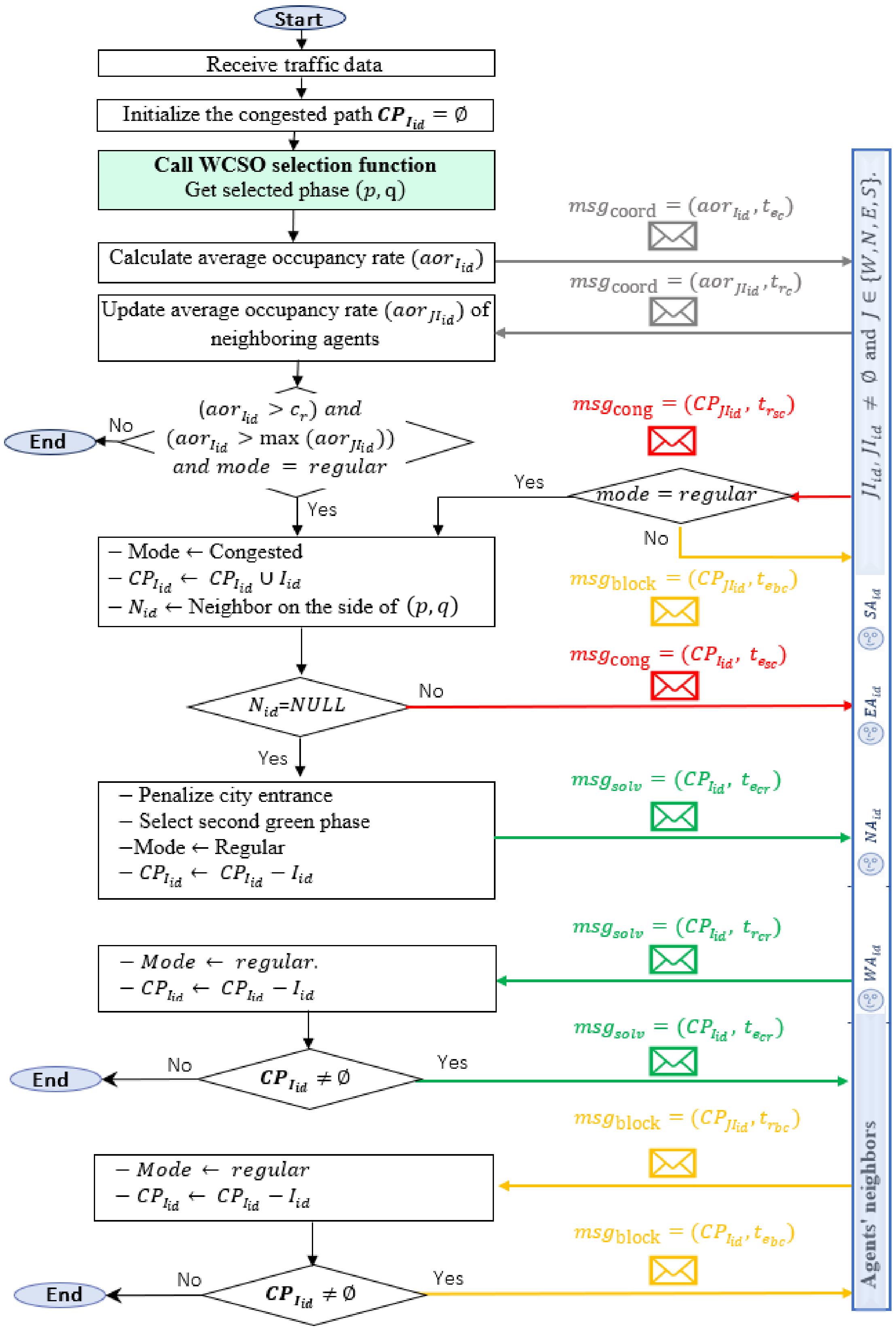

Start
Receive traffic data
Initialize the congested path $CP_{I_{id}} = \emptyset$
Call WCSO selection function
Get selected phase (p, q)
Calculate average occupancy rate $(aor_{I_{id}})$
Update average occupancy rate $(aor_{JI_{id}})$ of neighboring agents
$(aor_{I_{id}} > c_r)$ and $(aor_{I_{id}} > \max(aor_{JI_{id}}))$ and mode = regular
No
End
Yes
mode = regular
Yes
No
− Mode ← Congested
− $CP_{I_{id}} \leftarrow CP_{I_{id}} \cup I_{id}$
− N_{id} ← Neighbor on the side of (p, q)
N_{id}=NULL
No
Yes
− Penalize city entrance
− Select second green phase
−Mode ← Regular
− $CP_{I_{id}} \leftarrow CP_{I_{id}} - I_{id}$
− Mode ← regular.
− $CP_{I_{id}} \leftarrow CP_{I_{id}} - I_{id}$
$CP_{I_{id}} \neq \emptyset$
No
End
Yes
− Mode ← regular
− $CP_{I_{id}} \leftarrow CP_{I_{id}} - I_{id}$
$CP_{I_{id}} \neq \emptyset$
No
End
Yes
$msg_{coord} = (aor_{I_{id}}, t_{e_c})$
$msg_{coord} = (aor_{JI_{id}}, t_{r_c})$
$msg_{cong} = (CP_{JI_{id}}, t_{r_{sc}})$
$msg_{block} = (CP_{JI_{id}}, t_{e_{bc}})$
$msg_{cong} = (CP_{I_{id}}, t_{e_{sc}})$
$msg_{solv} = (CP_{I_{id}}, t_{e_{cr}})$
$msg_{solv} = (CP_{I_{id}}, t_{r_{cr}})$
$msg_{solv} = (CP_{I_{id}}, t_{e_{cr}})$
$msg_{block} = (CP_{JI_{id}}, t_{r_{bc}})$
$msg_{block} = (CP_{I_{id}}, t_{e_{bc}})$
$JI_{id}, JI_{id} \neq \emptyset$ and $J \in \{W, N, E, S\}$.
SA_{id}
EA_{id}
NA_{id}
WA_{id}
Agents' neighbors

Figure 3.15 Chronological coordination process in congestion mode.

6.2.2 Coordination Process in Weather Disasters

The pseudo-code of the chronological coordination process in weather disasters is shown in Figure 3.16, which runs before the detection of weather disasters by an agent A_{id} and before the initiation of the coordination process for the detection of congested paths, is described in the following steps:

a. The agent A_{id} processes data about weather situations.

b. When weather disaster is detected, agent A_{id} sends an out-of-service message ($msg_{\text{out}} = I_{id}\ , t_{e_{os}}$) to neighboring intersections to inform them of the out-of-service status.

c. The neighboring intersections take measures to block the roads leading to them.

d. When weather conditions improve, the traffic flow service is resumed at the intersection I_{id}, so that another return service message ($msg_{\text{return}} = I_{id}\ , t_{e_{rs}}$) is sent to the neighbors.

e. The roads are unblocked by the neighbors.

f. Call the above chronological coordination process in congestion mode.

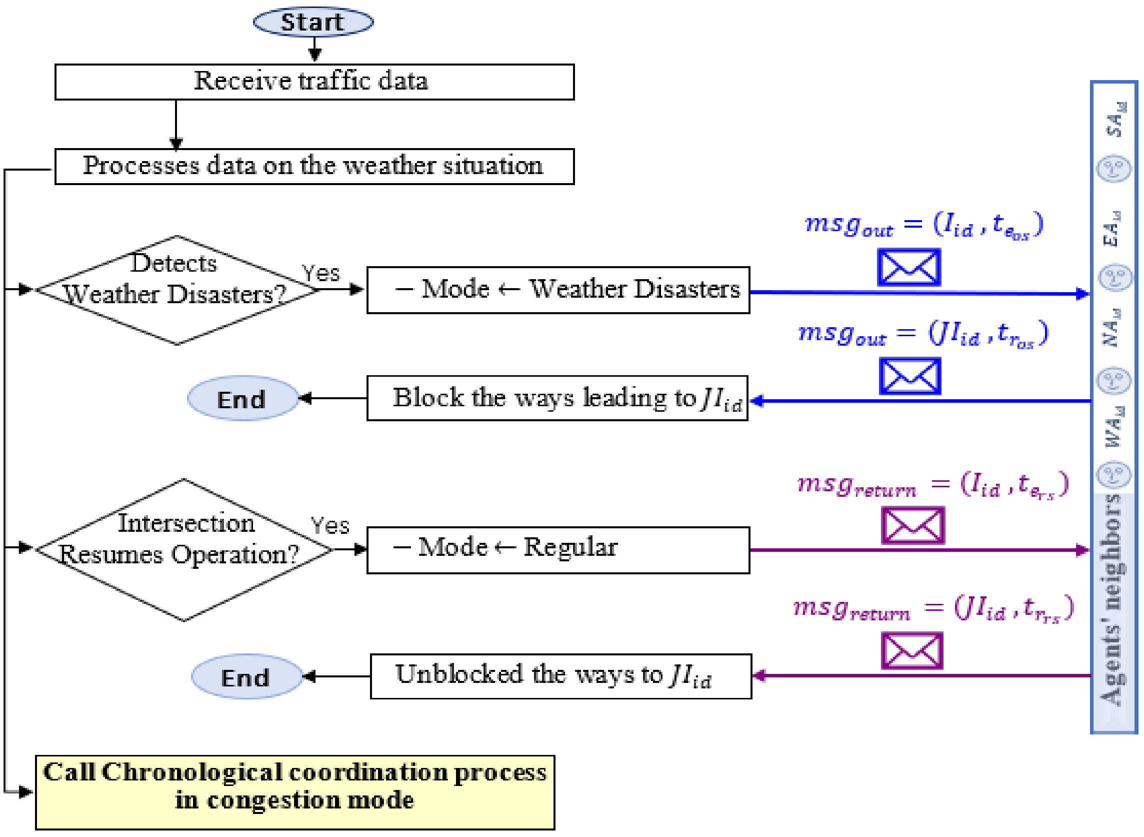

Start
Receive traffic data
Processes data on the weather situation
Detects Weather Disasters?
Yes
$-$ Mode $\leftarrow$ Weather Disasters
$msg_{out} = (I_{id}\ , t_{e_{os}})$
$msg_{out} = (JI_{id}\ , t_{r_{os}})$
End
Block the ways leading to JI_{id}
Intersection Resumes Operation?
Yes
$-$ Mode $\leftarrow$ Regular
$msg_{return} = (I_{id}\ , t_{e_{rs}})$
$msg_{return} = (JI_{id}\ , t_{r_{rs}})$
End
Unblocked the ways to JI_{id}
Call Chronological coordination process in congestion mode
SA_{id}
EA_{id}
NA_{id}
WA_{id}
Agents' neighbors

Figure 3.16 Chronological coordination process in weather disasters mode.

7. Conclusion

In this chapter, we introduced a new system known as ADSTLS (Adaptive and Dynamic Smart Traffic Light System). It is based on smart traffic data collection and collective intelligence algorithms. After presenting the general architecture and functional model of our system, we detailed the steps for managing the traffic lights. ADSTLS aims to efficiently manage traffic flow at a single intersection within specific assumptions using three traffic light models: phase; cycle, and hybrid. Indeed, the system provides an automatic decision-making process for regular and emergency scenarios based on two control modes: Dynamic Intelligent Mode (DIM) and Adaptive Emergency Mode (AEM). In the first mode, an optimal phase is selected that meets the criteria for the average waiting time and lane occupancy constraints based on the WCSO algorithm. This algorithm enables the optimization of weights of average waiting time and lane occupancy. A second mode selects an optimal phase in the best mode obtained from the first mode by integrating the priority of urgent vehicles.

Furthermore, we proposed an extension of ADSTLS, called CCADSTLS which is based on a multi-agent approach. This extension enables the effective management of multiple intersections through different ADSTLS autonomous agent controllers. These agents are coordinated chronologically to alleviate traffic congestion and manage weather fluctuations at several intersections. CCADSTLS involves the strategic collection of real-time traffic data from various intersections from multiple cameras and sensors, the integration of various exchanged messages among autonomous controllers, the identification of optimal congestion paths using traffic flow statistics and incremental forward control, and the development of robust traffic flow management models.

In the next chapter, we will present a Java simulator that illustrates our approach in practice. We will also present some evaluations and comparisons with other existing methods in the literature.

Chapter 4: Smart Traffic Flow Simulation System and Evaluation

Content

1. Introduction 114
2. Simulator Presentation 114
3. Simulator Demonstration 116
   3.1 Presenting the Case Study 116
   3.2 Case Study Simulation 117
   3.3 Implementation and Deployment Challenge 118
4. ADSTLS Performance Evaluation for a Single Intersection 119
   4.1 Evaluation Metrics 120
   4.2 Optimization Performance Comparison 123
   4.3 Performance Comparison of ADSTLS Models in Dynamic Mode 126
   4.4 Performance Comparison of ADSTLS Models in Adaptive Mode 129
   4.5 Comparison of Traffic Signal Plan and Performance Optimization 131
   4.6 Results Analysis and Key Observations 135
5. CCADSTLS Performance Evaluation for a Set of Intersections 137
   5.1 Traffic Congestion Scenario 139
   5.2 Weather Disaster Scenario 139
   5.3 Performance Evaluation Metrics 140
   5.4 Simulation Results and Discussion 141
6. Conclusion 142

1. Introduction

Our challenge is to propose a system that can be deployed as a smart traffic light system that can manage regular traffic flows and congestions, and also a sensors-based environment to improve traffic flow management. Our approach considers priority vehicles and environmental conditions to provide drivers with traffic paths suited to various situations.

In this chapter, we present a simulator designed to facilitate dynamic and adaptive management of regular traffic flow and traffic congestion. First, we will give an overview of this simulator and showcase its application at the El-Hidhab Setif city intersection. Next, we present the evaluation criteria and metrics to validate our proposals. Finally, we will conduct a comparative analysis of the proposed methods and the existing methods available in the literature.

2. Simulator Presentation

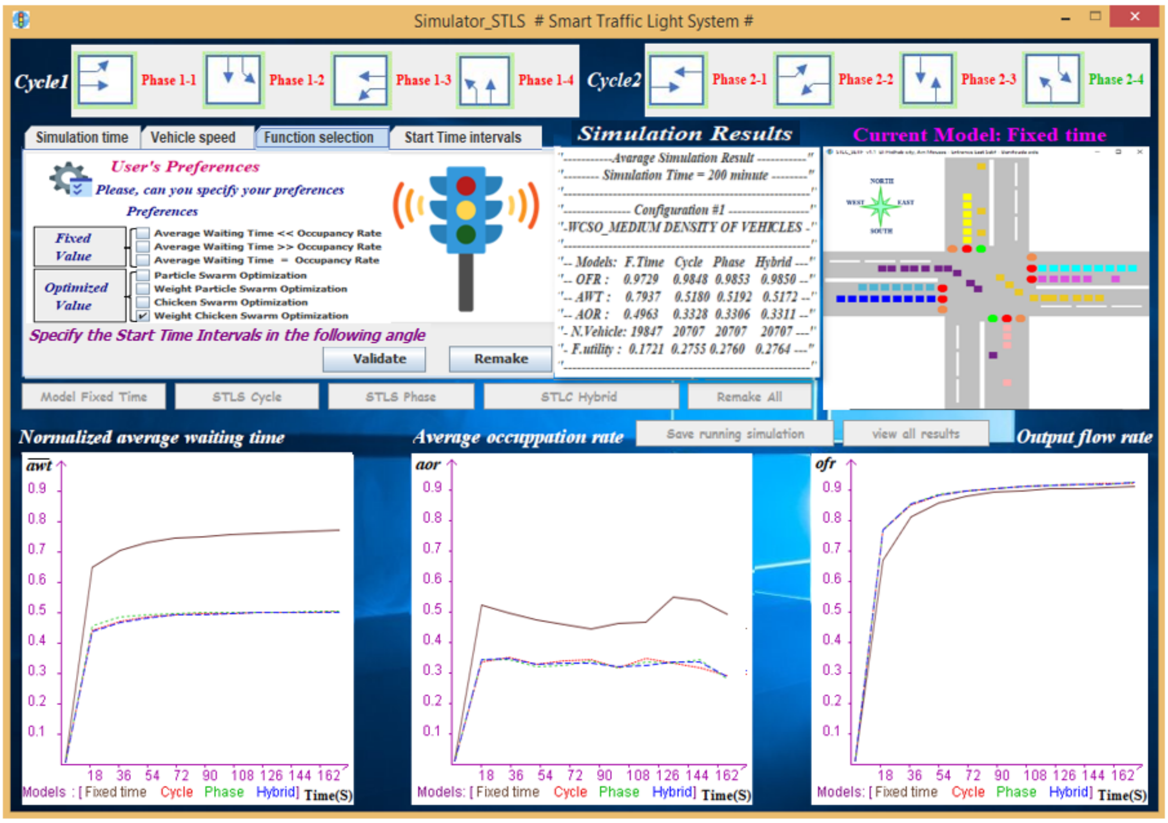
Our simulator STLS (Smart Traffic Light System) Our simulator, STLS, was implemented on Java using Eclipse. The STLS simulator includes dynamic and adaptive control models of an isolated intersection with four directions (north (N), south (S), east (E), and west (W)). Each of these consists of two movements or queues. Thus, there are eight movements $(We, Wn, Ns, Ne, Ew, Es, Sn, Sw)$. Two cycle-based models were used, as shown in Figure 3.3.

Each vehicle with a driver was modeled as an agent interacting with another agent in synchronous mode using ACL messages, and a linear dynamic model was used for vehicle-following behavior based on speed and acceleration parameters.

The STLS is sufficiently smart to efficiently manage traffic flow and emergency vehicles using four different control models with the same configuration. Thus, it allows viewing and comparison of the simulation results.

For multiple intersections, the JADE multi-agent framework is used to develop a CCADSTLS-based traffic simulator. Each controller manages its intersection as an autonomous agent and coordinates with other controllers. First, we can use the simulator to simulate different traffic scenarios to test our agent approach. Second, the simulator provides the necessary traffic data in real-time so that for agents can analyze traffic metrics relating to congestion and weather disasters. This traffic data is continuously monitored by the agents. In addition, the agents control the traffic lights by sending commands to the simulator in the form of control parameters to set the duration of green or red.

Figure 4.1 shows the main user interface of the STLS simulator.


Simulator_STLS # Smart Traffic Light System #
Cycle1
Phase 1-1
Phase 1-2
Phase 1-3
Phase 1-4
Cycle2
Phase 2-1
Phase 2-2
Phase 2-3
Phase 2-4
Simulation time
Vehicle speed
Function selection
Start Time intervals
User's Preferences
Please, can you specify your preferences
Preferences
Fixed Value
Optimized Value
Average Waiting Time << Occupancy Rate
Average Waiting Time >> Occupancy Rate
Average Waiting Time = Occupancy Rate
Particle Swarm Optimization
Weight Particle Swarm Optimization
Chicken Swarm Optimization
Weight Chicken Swarm Optimization
Specify the Start Time Intervals in the following angle
Validate
Remake
Simulation Results
Current Model: Fixed time
NORTH
WEST
EAST
SOUTH
Model Fixed Time
STLS Cycle
STLS Phase
STLC Hybrid
Remake All
Save running simulation
view all results
Normalized average waiting time
Average occuppation rate
Output flow rate
awt
aor
ofr
Time(S)
Models : [Fixed time Cycle Phase Hybrid]

```
"'-----------Avarage Simulation Result -----------"
"'-------- Simulation Time = 200 minute --------"
"'-----------------------------------------------------"
"'--------------- Configuration #1 ----------------"
"'-WCSO_MEDIUM DENSITY OF VEHICLES -"
"'-----------------------------------------------------"
"'-- Models:  F.Time   Cycle   Phase   Hybrid ---"
"'-- OFR :    0.9729   0.9848  0.9853  0.9850 --"
"'-- AWT :    0.7937   0.5180  0.5192  0.5172 --"
"'-- AOR :    0.4963   0.3328  0.3306  0.3311 --"
"'- N.Vehicle: 19847   20707   20707   20707 ---"
"'- F.utility :  0.1721  0.2755 0.2760  0.2764 ---"
"'-----------------------------------------------------"
```

Figure 4.1 Main user interface of STLS simulator.

The main functionalities of the STLS are as follows:

- Selecting configuration to be simulated by the user.
- Launching successively and independently simulations of four control models.
- Displaying phases during green, orange and red times.
- Visualizing partial simulation results by model and final evaluation results.
- Display evaluation reports per model in terms of the vehicle waiting time, queue occupancy rate, and output flow.

The simulation settings of an intersection are:

- There are nine vehicles per queue.
- The safety distance and typical vehicle length are 26 simulated pixels (i.e., 18 pixels of vehicle length plus 8 pixels of safety distance), equivalent to an average real length of 4.5 meters (4 m vehicle and safety distance of 0.5 m).
- 234 simulated pixels, or 40.5 m, constitute the queue length.
- The user specifies the simulation time in minutes and vehicle speed. The speed of the vehicle s_v during the simulation is one of three modes: 1) - *Fast* corresponds to a value of 143.55 pixel/s, 2) *Medium* corresponds to a value of 71.78 pixel/s and 3) -

Long corresponds to a value of 47.85 pixel/s. We assumed that these values are equivalent to 15 km/h in the real average speed (e.g., adapted to real tests) equal to 4.16 m/s. The simulation time needed to traverse a queue is 1.63s, 3.26s, and 4.89s in fast, medium and long modes respectively, equivalent to 9.74s in real-time. This means that the margin of the simulation time to real-time s_r is defined as follows: one second simulated equals 6 s (*fast mode*), 3 s (*medium*), and 2 s (*long*) in real time.

– Furthermore, a user specifies their preferences to select the next green phase or cycle. The user preferences reflect the weights of the phase or cycle selection function parameters for normalized average queue waiting time of all vehicles currently in the queue ($\overline{aqwt_k}$), and queue occupancy rate $\left(occup_{r_k}\right)$.
– Finally, a user specifies the start time interval ($t_{sti} = [begin\ time, end\ time]$) is the delay between the arrival of two successive vehicles in each queue. The vehicles were introduced randomly, and the time interval was configured with a random simulation step x of 100 ms. If $t_{sti} = [1000, 4000]$, the delay between the arrival of two successive vehicles is $1000 + x * 100$ and $0 \leq x \leq 30$, that is, the possible values of x are 31.

3. Simulator Demonstration

The STLS simulator is mainly demonstrated through a real traffic case study at the El-Hidhab Setif city intersection. Therefore, in this section, we present the selected case study and illustrate the relevant fundamental services.

3.1 Presenting the Case Study

The proposed system was evaluated at El-Hidhab Setif city intersection using three control models (cycle, phase, and hybrid) as well as available smart sensors to improve the performance of our system. Our goal is to collect traffic data using multimedia sensors such as cameras so that vehicles and intersections can be efficiently controlled.

In this case study, the cycle 2 model was used to synchronize the traffic signals, as shown in Figure 4.2, with a fixed green light duration for all phases, as follows: Phases 1-2 and 3-2 30 seconds, phases 2-2 and 4-2 24 seconds. The duration of the orange light was 5 seconds. These durations were converted to simulated values using the simulation time rate s_r such that the green light duration was 30 s, which is equivalent to 10 s simulated for the medium speed mode.

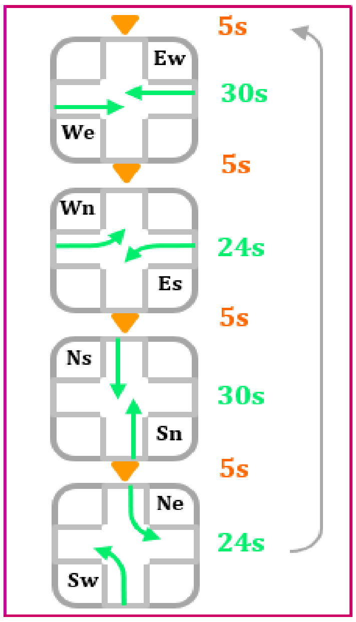
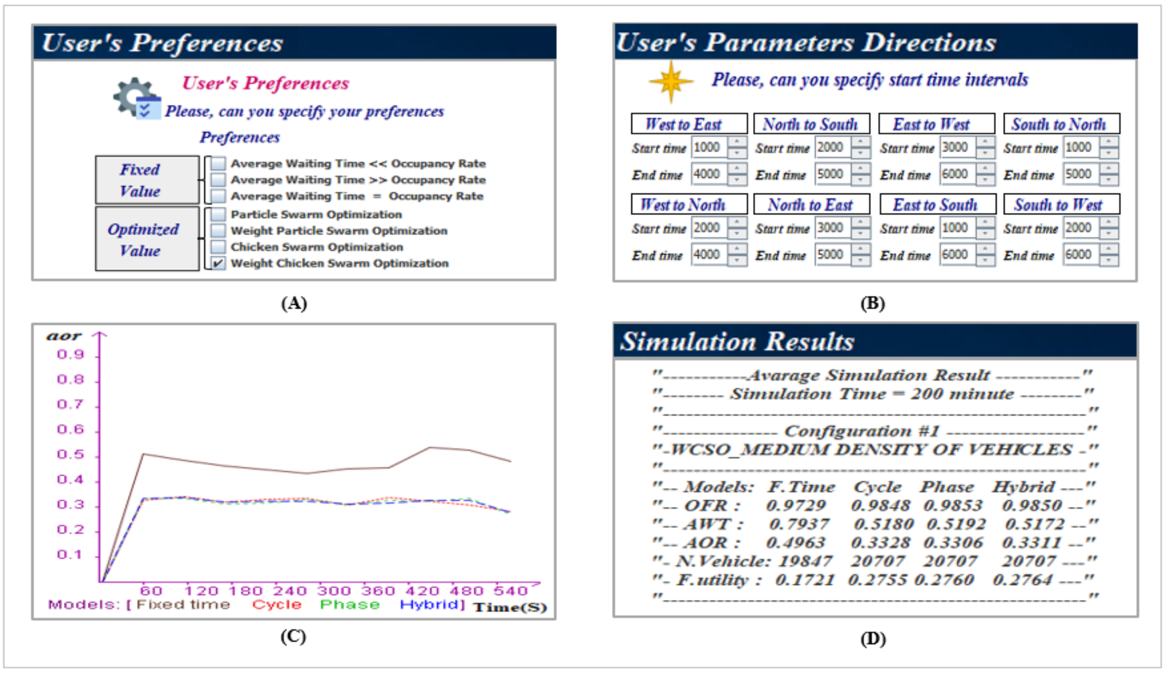


Figure 4.2 Description of cycle, green and orange time.

3.2 Case Study Simulation

During the phase and cycle selection, the user selects their preferences, such as the average vehicle wait time and queue occupancy rate. The weights of the selection function parameters are either fixed by the user or automatically optimized using the proposed swarm intelligence dynamic approach, as shown in Figure 4.3.A.



User's Preferences

User's Preferences
Please, can you specify your preferences
Preferences

Fixed Value
- ☐ Average Waiting Time << Occupancy Rate
- ☐ Average Waiting Time >> Occupancy Rate
- ☐ Average Waiting Time = Occupancy Rate

Optimized Value
- ☐ Particle Swarm Optimization
- ☐ Weight Particle Swarm Optimization
- ☐ Chicken Swarm Optimization
- ☑ Weight Chicken Swarm Optimization

(A)

User's Parameters Directions

Please, can you specify start time intervals

	West to East	North to South	East to West	South to North
Start time	1000	2000	3000	1000
End time	4000	5000	6000	5000

	West to North	North to East	East to South	South to West
Start time	2000	3000	1000	2000
End time	4000	5000	6000	6000

(B)

Simulation Results

```
"-----------Avarage Simulation Result -----------"
"-------- Simulation Time = 200 minute --------"
"------------------------------------------------"
"-------------- Configuration #1 ----------------"
"-WCSO_MEDIUM DENSITY OF VEHICLES -"
"------------------------------------------------"
"-- Models:  F.Time   Cycle   Phase   Hybrid ---"
"-- OFR :    0.9729   0.9848  0.9853  0.9850 --"
"-- AWT :    0.7937   0.5180  0.5192  0.5172 --"
"-- AOR :    0.4963   0.3328  0.3306  0.3311 --"
"- N.Vehicle: 19847   20707   20707   20707 ---"
"- F.utility :  0.1721  0.2755 0.2760  0.2764 ---"
"------------------------------------------------"
```

(D)

Figure 4.3 Model configuration parameters and simulation results, (A) Specification of user's preference, (B) Specification of start time intervals, (C) Graph of average queue occupancy rate, (D) Displaying evaluation results.

The user specifies the start time intervals between the two vehicles in each queue. This is illustrated in Figure 4.3.B. The user determines two interval bounds, 1 and 4 seconds, for the west-to-east queue. The performance results of the four models (cycle, phase, hybrid, and fixed light green time) are visualized graphically, as shown in Figure 4.3.C Average queue occupancy rate graph for a simulation time of 10 minutes. Detailed evaluation results of the configuration for several simulations are shown in Figure 4.3.D.

3.3 Implementation and Deployment Challenge

ADSTLS is implemented using different manufacturers' technologies (IoT sensors, cameras, and communication protocols) to ensure real-time traffic data collection, reduce congestion, and improve emergency response time. Despite the benefits, the diverse technologies present significant challenges, including incompatibility, scalability, regulatory and policy compliance, and traffic data security concerns. Incompatibility issues require standards and protocols for interoperability, necessitating continuous testing of video content delivery and metric calculations before deployment. Scalability, driven by increased traffic volume, also requires strong data privacy measures to protect traffic video content from potential attacks. However, a cost-benefit analysis is essential to assess the economic feasibility and justify implementing ADSTLS systems, which contribute positively to urban traffic management and quality of life. The economic analysis of ADSTLS deployment is defined as follows:

- **Cost:** The cost includes infrastructure, labor, and maintenance costs. Infrastructure costs encompass the installation of cameras, IoT devices, software, and communication networks. Labor costs involve various uses of the company's personal time and training processes. Maintenance costs include the continuous provision of monitoring systems, communications system maintenance, and software updates. The total infrastructure costs are amortized over ten years, ensuring the sustainability of this investment.
- **Benefit:** The benefits include public and organizational advantages. Public benefits encompass the optimization of travel time (i.e., waiting time and occupancy rate) and a reduced number of congestions (i.e., fuel saving, which will significantly impact daily operations. Organizational benefits include the avoidance of manual data collection and unnecessary maintenance activities.

We can define the cost-benefit (B_C) function as the ratio of all benefits (B) to that of costs (C) to deploy ADSTLS. The total cost includes infrastructure, labor, and maintenance costs defined by C_1, C_2 and C_3 respectively. The total benefits include reduced travel time, improved safety, and fuel saving benefits defined by B_1 , B_2 and B_3 respectively. The cost-benefit is formulated as follows equation (4.1):

$$B\_C = \frac{\sum_{i=1}^{3} B_i}{\sum_{i=1}^{3} C_i} \tag{4.1}$$

Table 4.1 provides assessment of economic costs-benefits of implementing ADSTLS system. The value of B_C is 1.02, so that the total annual benifice is greater than the total costs.

Table 4.1 Economic cost-benefit analysis for ADSTLS system implementation.

Category	Description Name	Calculation/Estimation ($)
Cost		
Infrastructure installation	Installation of cameras, IoT devices, software and communication networks	5500 $
Labor	Personal time and training effort	4000 $
Maintenance	Monitoring, communications maintenance and software updates	3500 $
Benefit		
Reduced travel time	Reduced travel time including occupation rate and waiting times.	5000 $
Improved safety	Improved safety by reducing congestions and accidents.	3800 $
Fuel saving	Fuel saving by optimizing	4500 $
Cost-Benefit		1.02

4. ADSTLS Performance Evaluation for a Single Intersection

We conducted several experiments on various traffic scenarios at intersection lanes to evaluate the effectiveness of ADSTLS using the proposed traffic flow models (Fixed-time, Phase, Cycle, Hybrid) and selection functions. Our goal is to evaluate the efficiency of ADSTLS in dynamic mode and efficiently manage prioritized vehicles in adaptive mode. We carried out two types of experiments: an optimization performance comparison and a traffic model performance comparison.

The optimization performance comparison involves swarm intelligence algorithm (WCSO, CSO, WPSO, PSO) testing to determine which algorithms quickly converge to the optimal solution for selection function weights. These results were compared with those of the fixed weights approach.

The performance traffic model comparison consisted of two modes: dynamic and emergency. In dynamic mode, we performed experiments to evaluate the effectiveness of our system on three signal flow management models: phase, cycle, and hybrid, in terms of output flow rate, average waiting time, and average occupancy rate. In the intelligent emergency mode, we conducted experiments on various emergencies to observe the responsiveness and effectiveness of the proposed situation model in terms of waiting time for emergency vehicles.

All experiments were performed using an HP ProBook PC with an Intel(R) Core (TM) i5-6200U CPU @ 2.30GHz 2.40GHz, and 8GB RAM. The vehicle speed (s_v) during simulation on medium mode is 71.78 pixels/s, the value of the constant c_g is 1000 ms and the simulated orange time (t_o) is 1700 ms.

4.1 Evaluation Metrics

To evaluate the performance and effectiveness of the proposed ADSTLS on the simulation dataset under different vehicle densities, several criteria and metrics were suggested and compared with fixed weight-based approaches commonly used as baseline models in traffic light control systems. Three different traffic metrics, including average waiting time, average occupancy rate, and output flow rate for each simulated model, were used to evaluate ADSTLS. Furthermore, we defined two other metrics during the traffic control process, namely best fitness and convergence speed, to evaluate the weight-based swarm intelligence approach.

- **Best Fitness** ($\boldsymbol{bf^m}$): this is the average fitness value, computed for each model m after N_s simulations, either by a fixed-weight model or weight-based swarm intelligence optimization model, as defined by equation (4.2).

$$bf^m = \frac{\sum_{j=1}^{N_s} f_j^m}{N_s} \tag{4.2}$$

  where f_j^m is the average fitness value of model m for a given simulation j after N_c calls of the selection function, calculated using equation (4.3).

$$f_j^m = \frac{\sum_{i=1}^{N_c} f_i^m}{N_c} \tag{4.3}$$

  f_i is the fitness value f returned for each call i to the phase selection function defined using equation (3.25).

- **Convergence Speed** ($\boldsymbol{cs^m}$): is the average number of iterations needed to achieve optimal weights computed for each model m after N_s simulations using equation (4.4):

$$cs^m = \frac{\sum_{j=1}^{N_s} cs_j^m}{N_s} \tag{4.4}$$

Where cs_j^m is the average iteration number of model m for a given simulation j after N_c calls of the phase selection function, calculated using equation (4.5).

$$cs_j^m = \frac{\sum_{i=1}^{N_c} \ cs_i^m}{N_c} \tag{4.5}$$

cs_i^m is the average iteration number for each call i in the phase selection function.

– **Accuracy ($\boldsymbol{acc^m}$)**: is the ratio between the obtained results and the number of calls (N_c) to the selection function, as computed by equation (4.6).

$$acc^m = \frac{acc_j^m}{N_c} \tag{4.6}$$

Where acc_j^m is a number initialized by zero and increased by one if the phase selected by the fixed-weight type is the same as that found by the swarm intelligence algorithms, calculated using equation (4.7).

$$acc_j^m = acc_j^m + 1 \tag{4.7}$$

On the other hand, we compared three proposed traffic control models (phase, cycle, and hybrid) and fixed-time models that represent our case study. Therefore, five metrics were used to evaluate and compare the performance of these models: total number of simulated vehicles, average waiting time, output flow rate, average occupancy rate, and model utility.

– **Number of Simulated Vehicles ($\boldsymbol{n_v{}^m}$)**: is the number of simulated vehicles for model m after N_s simulations, computed by equation (4.8):

$$n_v{}^m = \sum_{j=1}^{N_s} n_{v_j}^m \tag{4.8}$$

Where $n_{v_j}^m$ is the number of simulated vehicles in all queues for a given simulation j, defined using equation (4.9).

$$n_{v_j}^m = \sum_{i=1}^{k} in_i^m \text{, k} \in wm \tag{4.9}$$

and in_i^m is the input flow for queue k calculated using equation (3.6), with $wm = \{We, Wn, Ns, Ne, Ew, Es, Sn, Sw\}$.

– **Average Waiting Time ($\boldsymbol{\overline{awt}^m}$)**: is the value of the normalized average waiting time for model m after N_s simulations, we use equation (4.10):

$$\overline{awt}^m = \frac{\sum_{j=1}^{N_s} \overline{awt}_j^m}{N_s} \tag{4.10}$$

where $\overline{awt}_j^m$ is the normalized average waiting time of all outgoing and waiting vehicles at the intersection of model m for a given simulation j, calculated using equation (3.19).

- **Output Flow Rate** ($\boldsymbol{ofr^m}$): is the value of the output flow rate for model m after N_s simulations, computed by equation (4.11):

$$ofr^m = \frac{\sum_{j=1}^{N_s} ofr_j^m}{N_s} \tag{4.11}$$

Where ofr_j^m is the output flow rate of model m for a given simulation j, as defined in equation (3.17).

- **Average Occupancy Rate** ($\boldsymbol{aor^m}$): is the average occupancy rate of the queues for model m after N_s simulations, calculated using equation (4.12):

$$aor^m = \frac{\sum_{j=1}^{N_s} aor_j^m}{N_s} \tag{4.12}$$

Where aor_j^m is the average occupancy rate in all queues for a given simulation j, defined using equation (4.13). We divided the simulation time into p intervals, as follows:

$$aor_j^m = \frac{\sum_{i=1}^{p} aor_i^m}{p} \quad and \quad p = 10 \tag{4.13}$$

and aor_i^m is defined in equation (3.16).

- **Model's Utility** ($\boldsymbol{F_u{}^m}$): is the ratio of the utility value of a model m to the total utility of all models. The best model had the highest utility rate. This is calculated using equation (4.14).

$$F_u{}^m = \frac{\overline{U}^m}{\sum_{i=1}^{m} \overline{U}^m} \tag{4.14}$$

where $\overline{U}_m$ is the utility weighting of the model m computed by equation (4.15):

$$\overline{U}^m = U^m * R^m \tag{4.15}$$

U_m is the utility of the model m calculated by equation (4.16):

$$U^m = \frac{ofr^m}{\overline{awt}^m + aor^m} \tag{4.16}$$

and R^m is the weight function of the model m computed by equation (4.17):

$$R^m = \frac{n_v{}^m}{max\,(n_v{}^{Fix-time}, n_v{}^{Phase}, n_v{}^{Cycle}, n_v{}^{Hybride})} \tag{4.17}$$

R^m is used to determine the best model with more simulated vehicles. The best value is equal to 1, where $0 < \mathrm{R}^m \leq 1$.

4.2 Optimization Performance Comparison

We performed optimization comparisons of four swarm intelligence optimization algorithms, including WCSO, CSO, WPSO, and PSO, which converged to an optimal solution with different numbers of iterations and different accuracy results. We attempt to determine the best algorithm that converges quickly to the optimal weights of the selection function. Furthermore, three fixed-weight selection function algorithms with different preferred criteria were evaluated with the best weight-based swarm intelligence to observe their fitness optimality and result accuracy in the selection process. Fixed-1 is a fixed-weight algorithm that prefers waiting time weight to queuing time weight ($w_q \gg w_t$). Fixed-2 is a fixed-weight algorithm with any preference between waiting time weight and waiting time weight ($w_q = w_t$). Fixed-3 is a fixed-weight algorithm that prefers a queuing time weight versus waiting time weight ($w_q \ll w_t$). Their fixed weights were (0.7, 0.3), (0.5, 0.5), and (0.3, 0.7), respectively.

Convergence speed, best fitness, and result accuracy are evaluation metrics that determine the best algorithm. We observed the performance in the optimization of the four algorithms with different specifications for start time interval (t_{sti}), which defines the time interval between the arrival of two successive vehicles: 1) Large Start Time Interval (LSTI) with t_{sti} = [1000ms, 5000ms], 2) Medium Start Time Interval (MSTI) with t_{sti} = [1000ms, 4000ms], and 3) Small Start Time Interval (SSTI) with t_{sti} = [1000ms, 3000ms]. We observed the optimization of the four swarm intelligence algorithms with varied numbers of particles in the case of WPSO and PSO or varied numbers of chickens in the case of WCSO and CSO: 30, 50, 70, and 90. More simulation parameters are provided in Table 4.2, with a total simulation time of 1710 minutes.

As shown in Table 4.3, the WCSO convergence speed continues to increase as the number of vehicle arrivals increases. The other three algorithms, CSO, WPSO, and PSO, are all stabilized in their convergence speed; thus, the convergence speed value of WCSO is 8 iterations, outperforming CSO by 19 iterations, WPSO by 29 iterations, and PSO by 46 iterations. We also note that WPSO recorded the best fitness value, surpassing CSO by 0.03%, PSO by 0.15%, and WCSO by 0.40%. The best optimization algorithms are those with the lowest iteration values and highest fitness values.

Table 4.2 Simulation parameters in optimization performance.

Parameter	**Value**
Duration of simulations	10 minutes
Number of simulations for each traffic situation, N_s	3
Maximum iteration, max_{it}	100
Number of particles, P	30-50-70-90
Number of chickens, N	30-50-70-90
Value of constant β in WCSO and WPSO	0.5
Value of constant $c_1$in WPSO and PSO	0.3
Value of constant c_2 in WPSO and PSO	0.7
Value of constant ω in PSO algorithm	0.5
Minimum constant ε in WCSO and CSO	10^{-16}

Table 4.3 Comparison of best fitness values and convergence speed.

Methods		**WCSO**		**CSO**		**WPSO**		**PSO**	
	#Particles	cs(U)	bf(%)	cs(U)	bf(%)	cs(U)	bf(%)	cs(U)	bf(%)
LSTI	**90**	5	0,9170	26	0,9225	36	0,9229	55	0,9208
	50	5	0,9170	26	0,9225	36	0,9229	55	0,9208
	70	5	0,9169	26	0,9224	36	0,9228	55	0,9207
	30	5	0,9168	26	0,9224	36	0,9228	55	0,9207
MSTI	**90**	7	0,9308	27	0,9340	37	0,9342	54	0,9329
	50	7	0,9307	27	0,9339	37	0,9341	54	0,9328
	70	7	0,9305	27	0,9338	37	0,9340	54	0,9327
	30	7	0,9304	27	0,9336	37	0,9339	54	0,9326
SSTI	**90**	12	0,9739	27	0,9757	38	0,9759	55	0,9751
	50	12	0,9738	27	0,9756	38	0,9758	55	0,9750
	70	12	0,9737	27	0,9755	38	0,9757	55	0,9749
	30	12	0,9736	27	0,9754	38	0,9756	55	0,9748
Avg Results	**90**	**8**	0,9406	27	0,9441	37	**0,9443**	54	0,9429
	50	**8**	0,9405	27	0,9440	37	0,9442	54	0,9428
	70	**8**	0,9404	27	0,9439	37	0,9442	54	0,9428
	30	**8**	0,9403	27	0,9438	37	0,9441	54	0,9427

WCSO demonstrated a small number of iterations and optimal weight values. Thus, we consider WCSO for decision making during the selection process.

We also compared the performance of the WCSO algorithm with three fixed-weight algorithms (Fixed-1, Fixed-2, and Fixed-3) using different vehicle arrival times (small, medium, and large) in terms of accuracy, best fitness, number of simulated vehicles, and number of calls to the selection function. As shown in Table 4.4, the accuracy score for the fixed-weight approach increases with an increase in the number of vehicles arriving at the intersection. However, the performance of the WCSO algorithm showed a significant improvement of 3.84% compared with the Fixed-3 algorithm, 5.46% compared with the Fixed-2 algorithm, and 8.95% compared with the Fixed-1 algorithm. Similar to the best fitness, the WCSO algorithm promptly showed a significant improvement of 7.41% over the Fixed-3 algorithm, 11.64% over the Fixed-2 algorithm, and 15.97% over the Fixed-1 algorithm.

Finally, the WCSO has a better and more stable optimization performance when increasing the number of arriving vehicles. With an increasing number of vehicles, we also obtained an optimized performance of the selection function within a certain boost of weights, which is more conducive to system quality and stability.

Table 4.4 Accuracy comparison of fixed weight vs optimized weight approaches using WCSO.

STI			**WCSO Approach**	**Fixed Weight Approach**					
				Fixed-1 $\boldsymbol{w_q \gg w_t}$		**Fixed-2** $\boldsymbol{w_q = w_t}$		**Fixed-3** $\boldsymbol{w_q \ll w_t}$	
	n_v (u)	N_c(u)	bf(%)	bf(%)	acc(%)	bf(%)	acc(%)	bf(%)	acc(%)
LSTI	4749	408	0.9170	0.7043	0.8725	0.7647	0.9216	0.8269	0.9510
MSTI	5623	391	0.9308	0.8094	0.9105	0.8411	0.9540	0.8747	0.9719
SSTI	6342	374	0.9739	0.9194	0.9519	0.9218	0.9626	0.9255	0.9626
Final Result	16714	1173	0.9406	0.8110	0.9105	0.8425	0.9454	0.8757	0.9616

4.3 Performance Comparison of ADSTLS Models in Dynamic Mode

We employed the evaluation metrics described in Section 4.1 to compare the proposed WCSO-based dynamic intelligent models (phase, cycle, and hybrid) with a fixed-time model. Three classes of experiments were conducted to ensure a comprehensive assessment.

- **Normal Density of Vehicles (NDV)**: consists of three start time intervals: 1) Large Start Time Interval (NDV-LSTI) with t_{sti} = [3000ms, 6000ms], 2) Medium Start Time Interval (NDV-MSTI) with t_{sti} = [3000ms, 5000ms] and 3) Small Start Time Interval (NDV-SSTI) with t_{sti} = [3000ms, 4000ms].
- **Medium Density of Vehicles (MDV)**: consists of three start time intervals: 1) Large Start Time Interval (MDV-LSTI) with t_{sti} = [2000ms, 6000ms], 2) Medium Start Time Interval (MDV-MSTI) with t_{sti} = [2000ms, 5000ms] and 3) Small Start Time Interval (MDV-SSTI) with t_{sti} = [2000ms, 4000ms].
- **High Density of Vehicles (HDV)**: consists of three start time intervals: 1) Large Start Time Interval (HDV-LSTI) with t_{sti} = [1000ms, 6000ms], 2) Medium Start Time Interval (HDV-MSTI) with t_{sti} = [1000ms, 5000ms] and 3) Small Start Time Interval (HDV-SSTI) with t_{sti} = [1000ms, 4000ms].

For the WCSO-based dynamic intelligent and fixed-weight models, the total number of simulated vehicles, average waiting time, average output flow rate, average occupancy rate, and utility of the model over 10 independent simulations of 180 minutes are reported. Each traffic model was run with three vehicle densities (NDV, MDV, and HDV). We used the final results as the output flow rate of vehicles (ofr), average waiting time (awt), and average occupancy rate (aor) obtained from each category of the experiment to compute the utility function F_u for the four models according to the time between the arrival of two successive vehicles in each queue. The values of the simulation parameters are listed in Table 4.5.

Table 4.5 Simulation parameters in dynamic mode.

Parameter	Value
Duration of simulations	10 minutes
Number of simulations for each traffic situation, N_s	5
Value of constant β in WCSO	0.5
Minimum constant ε in WCSO	10^{-16}
Maximum iteration, max_{it}	50
Number of chickens, N	30
Hybrid rate, h_r	0.5
Time scale, t_s	20000 ms

Table 4.6 shows the performance of the four control models (fixed-time, WCSO phase, WCSO cycle, and WCSO hybrid) at different start time intervals when the system generates normal, medium and high vehicles density. When the vehicle density is normal, a vehicle reaches each queue at the intersection every 3.98 seconds in the simulation, which corresponds to 11.94 seconds in reality.

With a medium density of vehicles, one vehicle reaches each queue at the intersection every 3.48 seconds in the simulation, while it takes 10.43 seconds in reality. In the case of the high density of vehicles, one vehicle reaches each queue at the intersection every 2.99 seconds in the simulation, while it takes 8.99 seconds in reality.

We further demonstrate the performance of the dynamic ADSTLS models in the SETIF city case study. When a normal, medium and high vehicles density of vehicles is generated, the dynamic ADSTLS models perform better than the fixed-time model because the WCSO algorithm is considered. However, the performances of the different models (WCSO cycle, WCSO phase, and WCSO hybrid) exhibited improvements of 56.17%, 56.75% and 56.82%, respectively in normal vehicles density and improvements of 60.13%, 60.40% and 60.63% in medium vehicles density, respectively. At high vehicle density, the different models (WCSO cycle, WCSO hybrid, and WCSO phase) exhibited improvements of 61.00%, 61.99%, and 68.26%, respectively.

When comparing the different dynamic models under normal and medium vehicle density, we found that the WCSO hybrid model achieves significantly higher utility and provides better results for the average waiting time in the system.

At high vehicle density, the simulation results show the predominance of the WCSO phase model, which performs the best in terms of vehicle output flow, average waiting time, and average occupancy rate in the LSTI, MSTI and SSTI start time intervals.

Figure 4.4 shows the output flow rate of vehicles (ofr), the average waiting time for vehicles (awt), the average occupancy rate of queues (aor) and the utility model (F_u) for configurations with normal, medium and high vehicle density. It is important to note that different configurations lead to different simulation results. When we look at the utility, we find that simulation results show the dominance of the hybrid model, which outperforms the WCSO phase and cycle models by 0.04% and 0.42%, respectively, at normal vehicle density, and by 0.15% and 0.32% respectively at medium vehicle density. At high vehicle density, the WCSO phase model outperforms the WCSO hybrid models by 3.87% and the WCSO cycle model by 4.51%.

Table 4.6 Comparison of model's utility under different vehicles started time in dynamic mode.

Evaluation Metrics	Start Time Intervals (STI)											
	Normal density of vehicles (NDV)				Medium density of vehicles (MDV)				High density of vehicles (HDV)			
	LSTI	MSTI	SSTI	Average Result	LSTI	MSTI	SSTI	Average Result	LSTI	MSTI	SSTI	Average Result
Fixed-Time Model												
ofr (%)	0.9747	0.9736	0.9739	0.9741	0.9722	0.9742	0.9722	0.9729	0.9729	0.9728	0.9701	0.9720
$\overline{awt}$ (%)	0.7559	0.7651	0.7809	0.7673	0.7860	0.7735	0.8216	0.7937	0.7830	0.8164	0.8813	0.8269
aor (%)	0.3867	0.4336	0.5008	0.4404	0.4319	0.4869	0.5700	0.4963	0.4889	0.5539	0.6314	0.5581
n_v (unit)	**5329**	5954	6789	18072	5842	6678	7327	19847	6569	7214	7635	21418
F_u	0.1731	0.1764	0.1768	0.1755	0.1703	0.1777	0.1689	0.1721	0.1758	0.1701	0.1632	0.1691
WCSO Cycle Model												
ofr (%)	**0.9865**	0.9849	0.9855	**0.9856**	0.9841	**0.9858**	0.9844	0.9848	**0.9851**	0.9830	0.9825	0.9835
$\overline{awt}$ (%)	0.4866	0.5098	**0.5092**	0.5019	0.5031	**0.5116**	0.5394	0.5180	0.5243	0.5413	0.6048	0.5568
aor (%)	0.2394	0.2872	**0.3194**	0.2820	0.2786	**0.3189**	0.4008	0.3328	**0.3250**	0.3950	0.5394	0.4198
n_v (unit)	**5329**	**5955**	**6821**	**18105**	**5965**	**6778**	**7964**	**20707**	**6763**	7862	9406	24031
F_u	0.2757	0.2684	**0.2781**	0.2741	0.2743	**0.2770**	0.2751	0.2755	0.2744	0.2741	0.2692	0.2723
WCSO Phase Model												
ofr (%)	0.9856	**0.9859**	0.9848	0.9854	**0.9861**	0.9854	0.9846	**0.9853**	0.9842	**0.9843**	**0.9833**	**0.9840**
$\overline{awt}$ (%)	**0.4836**	0.4990	0.5184	0.5003	**0.5017**	0.5161	0.5398	0.5192	**0.5108**	**0.5298**	**0.5639**	**0.5348**
aor (%)	**0.2386**	**0.2742**	0.3286	**0.2805**	**0.2692**	0.3286	0.3942	**0.3306**	0.3253	**0.3853**	**0.5000**	**0.4035**
n_v (unit)	**5329**	**5955**	**6821**	**18105**	**5965**	**6778**	**7964**	**20707**	**6763**	**7864**	**9490**	**24117**
F_u	**0.2769**	0.2769	0.2719	0.2751	**0.2787**	0.2723	0.2770	0.2760	**0.2785**	**0.2809**	**0.2924**	**0.2846**
WCSO Hybrid Model												
ofr (%)	0.9854	0.9847	**0.9856**	0.9852	0.9853	0.9850	**0.9848**	0.9850	0.9837	0.9833	0.9820	0.9830
$\overline{awt}$ (%)	0.4869	**0.4940**	0.5137	**0.4982**	0.5022	0.5128	**0.5364**	**0.5172**	0.5211	0.5423	0.5926	0.5520
aor (%)	0.2419	0.2744	0.3300	0.2821	0.2733	0.3294	**0.3906**	0.3311	0.3367	0.3917	0.5308	0.4197
n_v (unit)	**5329**	**5955**	**6821**	**18105**	**5965**	**6778**	**7964**	**20707**	6762	7863	9444	24069
F_u	0.2743	**0.2783**	0.2732	**0.2753**	0.2768	0.2730	**0.2791**	**0.2764**	0.2713	0.2749	0.2752	0.2740

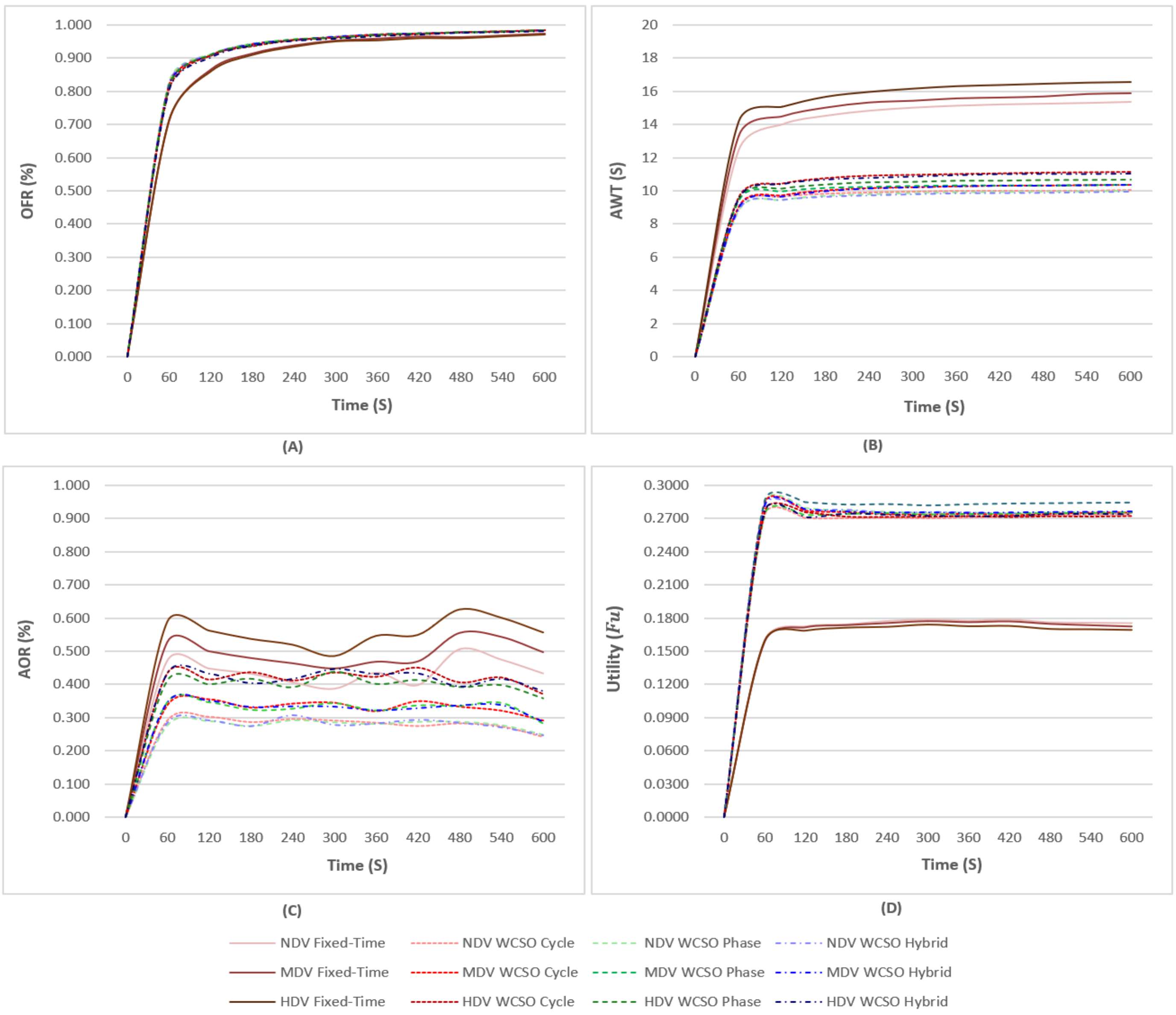


Figure 4.4 Comparison of model's performance under different vehicles started time in dynamic mode (A) Output flow rate, (B) Average waiting time, (C) Average occupancy rate and (D) Model utility.

4.4 Performance Comparison of ADSTLS Models in Adaptive Mode

The above results indicate that the dynamic WCSO phase model performs better in high density of vehicles than the dynamic WCSO cycle and dynamic WCSO hybrid models. Therefore, we consider this model in the adaptive emergency mode of ADSTLS to optimize the waiting time of both the emergency and regular modes. However, we used the WCSO algorithm to select a coherent queue that does not contain vehicle emergencies. Simulation parameter values are provided in Table 4.7.

Table 4.7 Simulation parameters in adaptive mode.

Parameter	**Value**
Duration of simulations	10 minutes
Number of simulations for each traffic situation, N_s	5
Value of constant β in WCSO	0.5
Minimum constant ε in WCSO	10^{-16}
Maximum iteration, max_{it}	50
Number of chickens, N	30

We conducted a simulation of five carefully selected situations to perform a comprehensive evaluation using automatic metrics. In this assessment, four priority values (0-No priority, 1-Low priority, 2-Medium priority, 3-High priority) and three start time intervals (Large Start Time Interval (LSTI) with $\mathrm{t_{sti}}$ = [1000ms, 6000ms], Medium Start Time Interval (MSTI) with $\mathrm{t_{sti}}$ = [1000ms, 5000ms] and Small Start Time Interval (SSTI) with $\mathrm{t_{sti}}$ = [1000ms, 4000ms]) were employed to evaluate the dynamic WCSO phase model in emergency mode. The WCSO phase model is assessed in emergency mode based on the following two metrics: the number of emergencies and regular vehicles (n_v) and the average waiting time of both emergency and regular vehicles (awt). The results obtained at different start time intervals for high-density vehicles with different priority level values in adaptive mode are presented in Table 4.8. Similar to automatic evaluation of ADSTLS models in dynamic mode, the WCSO phase model that manages regular and emergency vehicles provides acceptable average waiting time for high-priority, medium-priority, and low-priority emergency vehicles are 5.037, 6.423, and 8.017 seconds respectively in simulated time.

Table 4.8 Simulation of different start time intervals for emergency vehicles.

Start Time Intervals	**Emergency Vehicles**						**Regular Vehicles ($pl_v = 0$)**	
	High priority ($pl_v = 3$)		Medium priority ($pl_v = 2$)		Low priority ($pl_v = 1$)			
	$\mathrm{n_v}$(u)	awt (s)	$\mathrm{n_v}$(u)	awt (s)	$\mathrm{n_v}$(u)	awt (s)	$\mathrm{n_v}$ (u)	awt (s)
HDV-LSTI	306	4806.93	160	5693.93	106	6619.03	6162	12871.80
HDV-MSTI	348	5007.20	182	6269.26	120	7885.55	6957	14373.68
HDV-SSTI	382	5295.46	199	7304.99	132	9544.97	7618	16187.14
Avg Results	1036	**5036.53**	541	**6422.73**	358	**8016.52**	20737	**14477.54**

4.5 Comparison of Traffic Signal Plan and Performance Optimization

In this section, the three proposed WCSO models (cycle, phase, and hybrid) in the dynamic intelligent mode are quantitatively compared with the fixed-time model, which is our case study, and with two other related studies [14, 21]. The comparison was based on the optimal traffic signal plan of traffic control models. The green and orange signal sequences are extracted from these models for the selected phases or cycles with the duration of the green time of each phase to show the difference between them. For the optimization comparison, three metrics were selected based on the experimental findings of the optimized traffic densities. The values of the vehicle output flow rate, average waiting time, and queue occupancy rate between these models were compared to determine the model with the best optimized performance.

For this purpose, we used the cycle and phase models already defined in Figure 3. 4, such as the two cycles: Cycle 1 = {(We, Wn), (Ns, Ne), (Ew, Es), (Sn, Sw)} and Cycle 2 = {(We, Ew), (Wn, Es), (Ns, Sn), (Ne, Sw)}, and the eight phases: ph1 = (We, Wn), ph2 = (Ns, Ne), ph3 = (Ew, Es), ph4 = (Sn, Sw), ph5 = (We, Ew), ph6 = (Wn, Es), ph7 = (Ns, Sn) and ph8 = (Ne, Sw).

Three categories of experiments are also used for these comparisons to ensure a complete evaluation: normal density of vehicles, medium density of vehicles, and high density of vehicles. The simulation parameters are described in Table 4.3 in Section 4.3

4.5.1 Comparison of Traffic Signal Plans

Signal planning is the organization and coordination of traffic signals to efficiently, safely, and smoothly control vehicular and pedestrian traffic. Recall that the fixed-time model uses a cycle 2 signal plan with fixed values for the green, orange, and red times; the WCSO cycle model uses a cycle 1 or cycle 2 plan; the WCSO phase model uses a phase plan; and the WCSO hybrid model uses a mixture of the aforementioned two WCSO cycle and phase models. Rida et al. [21] used a cycle signal plan to optimize vehicle waiting times and queue sizes, favoring the phase with the lowest number of vehicles at the beginning of the cycle. Joo et al. [14] used a phase signal plan to optimize queue size and vehicle output, selecting the optimal phase with a fixed green time to cover the entire queue. For all models, it should be noted that when the two traffic lights of the selected phase are green or orange, the other six are red. Figure 4. 5 show extracts from real 3-minute simulations (180 seconds) in the three test categories:

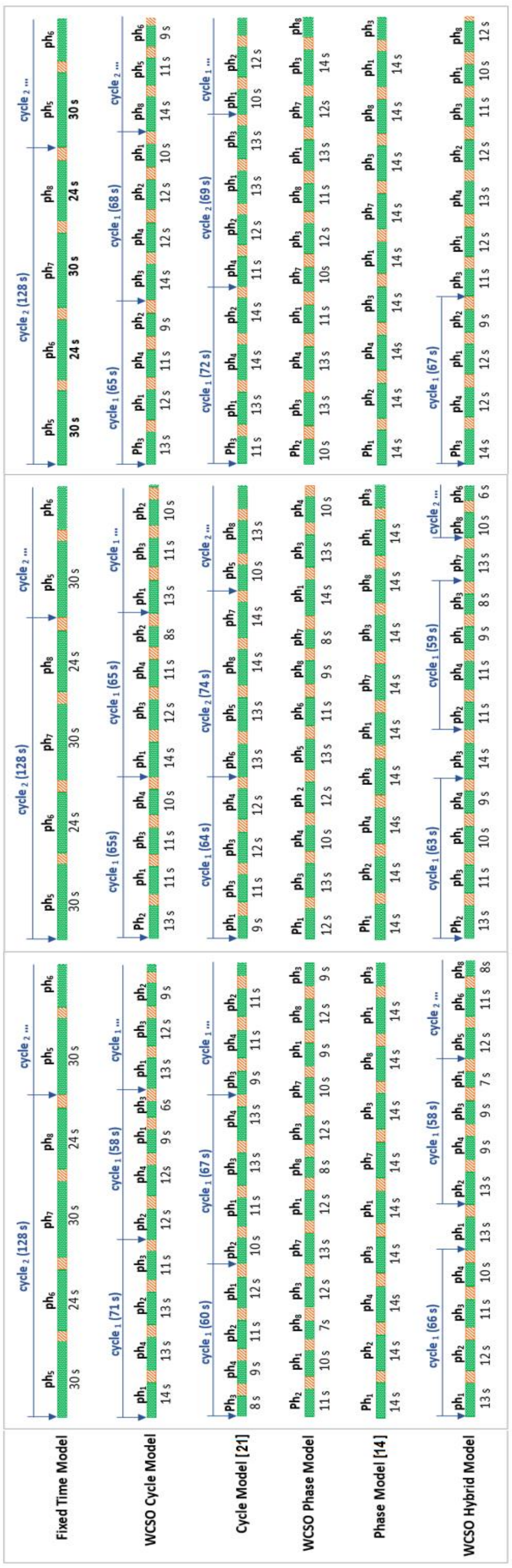


Figure 4.5 Comparison of traffic signal plans (A) Normal density of vehicles, (B) Medium density of vehicles and (C) High density of vehicles.

- **Traffic signal plan for normal density of vehicles**: Figure 4.5. (A) shows the average green times for the six models as follow: 27.00 seconds for the fixed-time model, 11.09 seconds for the WCSO cycle model, 11.18 seconds for the WCSO phase model, 10.73 seconds for the WCSO hybrid model, 11.80 seconds for the cycle model [21], and 14.00 seconds for the phase model [14].
- **Traffic signal plan for medium density of vehicles**: Figure 4.5. (B) shows the average green times for the six models as follow: 27.00 seconds for the fixed-time model, 11.27 seconds for the WCSO cycle model, 11.36 seconds for the WCSO phase model, 10.82 seconds for the hybrid-WCSO model, 12.10 seconds for the cycle model, and 14.00 seconds for the phase model.
- **Traffic signal plan for high density of vehicles**: Figure 4.5. (C) shows the average green times for the six models as follow: 27.00 seconds for the fixed-time model, 11.55 seconds for the WCSO cycle model, 11.90 seconds for the WCSO phase model, 11.64 seconds for the WCSO hybrid model, 12.33 seconds for the cycle model, and 14.00 seconds for the phase model.

According to the fixed-time model and phase model [14], a fixed green time is recorded in relation to their signal plan. For the signal plans of the WCSO cycle, phase, hybrid, and cycle models [21], we found that the average green time of the phase increased when the density of arriving vehicles also increased. However, the WCSO hybrid model always has a minimum average green time, which is due to the flexibility of switching between the phase and cycle models depending on the instantaneous traffic density.

4.5.2 Comparison of Performance Optimization with Relative Models

The aim of determining the best signal plans for the intelligent traffic signal control model is to maximize vehicle output rate at the intersection and minimize queue size and vehicle waiting time. In Section 4.3, we compare the three proposed WCSO DIM models (cycle, phase, and hybrid) and the fixed-time model, and we extend our comparison with two more models from the literature [14, 21]. The cycle model [14] is compared with the proposed WCSO cycle model, and the phase model [21] with the proposed WCSO phase model. These comparisons are based on the strategy of prioritizing the shortest queue in the cycle model [21], and in the phase model [14], the choice of a fixed green time that primarily influences traffic optimization. Figure 4.6 displays an optimization comparison of the six traffic models in terms of the Output Flow Rate (ofr), Average Waiting Time (awt), and Average Occupancy Rate (aor).

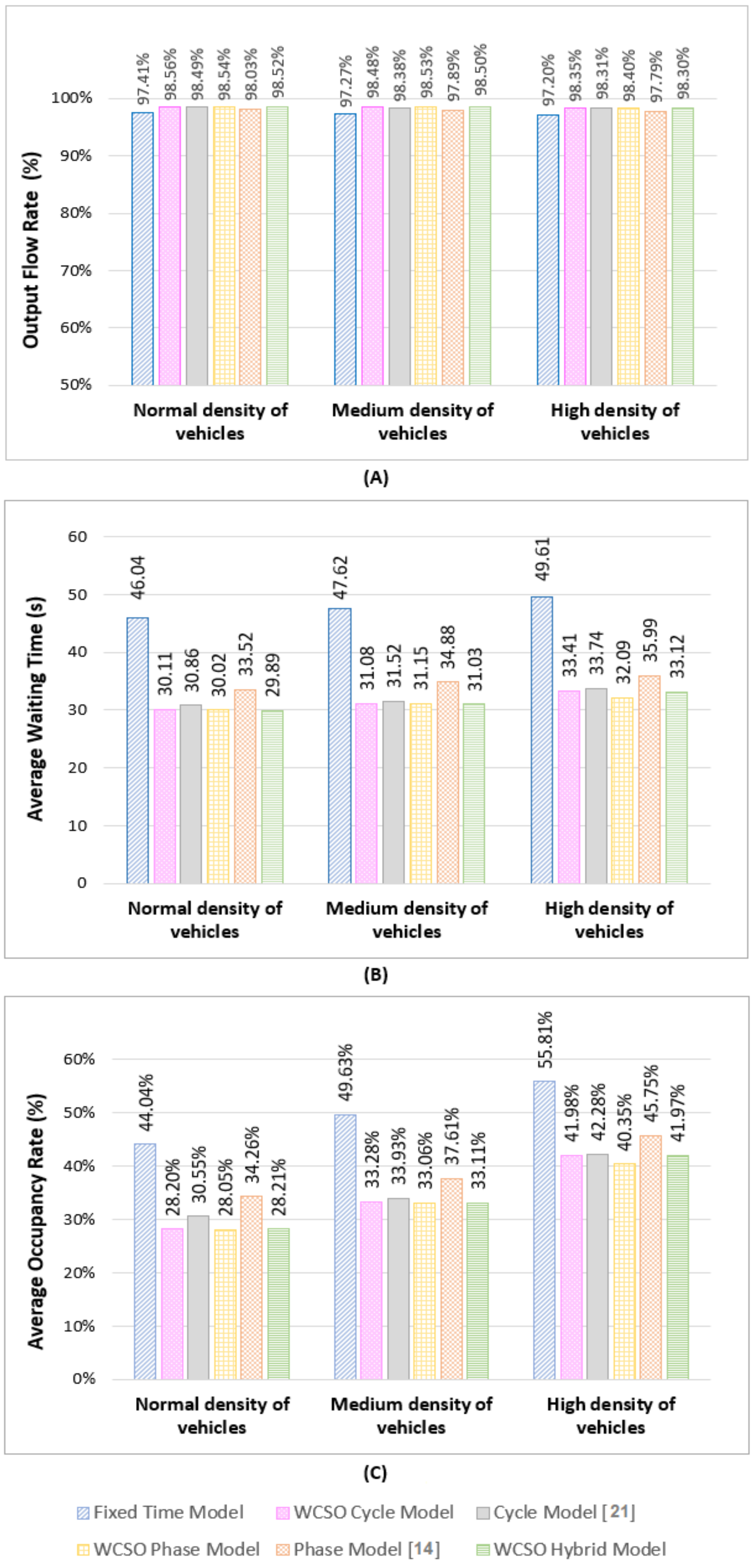


Figure 4.6 Comparison of performance optimization (A) Output flow rate, (B) Average waiting time and (C) Average occupancy rate.

Figure 4.6 (A) shows that the WCSO cycle model provides the best optimal ofr scores in NDV, whereas in MDV and HDV, the WCSO phase model provides the best optimal ofr scores. The WCSO cycle model achieved an average improvement of 0.07% over the cycle model [21] and the WCSO phase model achieved an average improvement of 0.59% over the phase model [14]. Figure 4.6 (B) illustrates that the WCSO hybrid model achieves the best awt values in the NDV and MDV, whereas the WCSO phase model achieves the best awt value in the HDV. The WCSO cycle model gives an average improvement of 0.51 seconds over the cycle model [21], and the WCSO phase model gives an average improvement of 3.71 seconds over the phase model [14]. Figure 4.6. (C) shows that the WCSO phase model achieves the best optimal aor values for the NDV, MDV, and HDV. The WCSO cycle model showed an average improvement of 1.10% compared to the cycle model [21], and the WCSO phase model showed an average improvement of 5.39% compared to the phase model [14].

From the above experimental results, we can conclude that our proposed WCSO DIM models (cycle, phase, and hybrid) yield an optimal average performance of 98.50% in terms of ofr, 31 seconds in terms of awt, and 33.82% in terms of aor for the three categories of vehicle arrival density. Moreover, the traffic signal plan of the proposed model is optimal.

4.6 Results Analysis and Key Observations

This section discusses and analyzes the impact of weight-based swarm intelligence algorithms in terms of the optimization and effectiveness of different dynamic traffic management and control models.

4.6.1 Optimization Performance Comparison of Different Swarm Intelligence Algorithms

The results obtained (Table 4.3) by WCSO outperformed all evaluated optimization algorithms in terms of convergence speed (e.g., eight iterations). In addition, WPSO achieves a better fitness function value (e.g., 0.9443) but requires more iterations than WCSO. This is because of the crucial role of particles or chickens in quickly converging optimal weights based on the new inertia weight formula.

4.6.2 Fixed-weights Approach vs Weight-based Chicken Swarm Optimization Approach

The above results from Table 4.4 show that dynamic traffic management models designed by WCSO swarm intelligence algorithm effectively improve the phase selection with an enhancement of 8.95% over fixed-weight approach because the weights of the selection function are calculated in each iteration until the performance of the traffic model converges to optimal values.

4.6.3 Dynamic intelligent vs fixed-time traffic control models

We also analyzed the model performance using three ADSTLS dynamic intelligent models and the fixed-time traffic control model in a case study of El-Hidhab Setif city intersection in Figure 4.4 and Table 4.6 to determine the best light green times. The obtained results show the superiority of the ADSTLS dynamic models compared with the fixed-time traffic light control model in terms of utility, which achieved a value of 0.1755. We also observed that the performance results of all dynamic intelligent models were promising. Therefore, this analysis demonstrates that making adequate intelligent decisions greatly improves traffic flow control results.

4.6.4 DIM Models Performance Comparison: WCSO Phase vs WCSO Cycle vs WCSO Hybrid

The time flow utility obtained by the ADSTLS models in dynamic intelligent mode is satisfactory and demonstrates the effectiveness of the proposed system (Table 4.6). Nonetheless, we classify the proposed dynamic traffic models according to vehicle density into three cases. In the normal or medium density of vehicles, the WCSO hybrid model was set first, followed by the WCSO phase model, and then the WCSO cycle model. For high density of vehicles, the WCSO phase model was set first, followed by the WCSO hybrid model, and then the WCSO cycle model.

Based on the experiments, we report that the WCSO hybrid and phase models yield favorable results in terms of system effectiveness and optimization performance. The phase model outperformed the hybrid and cycle models by reducing the queue average occupancy rate by 33.82% in all simulated traffic scenarios. However, the hybrid model is the superior traffic model as it reduces the average waiting time to 29.89 seconds and 31.03 seconds for normal and medium density vehicles respectively. Similarly, the phase model outperforms the other models for high-density vehicles, achieving an average waiting time of 32.09 seconds.

The utilities obtained by the El-Hidhab Setif city intersection case study are shown in Figure 4.7, which illustrates that the proposed hybrid model with WCSO manages control lights with the highest utility (e.g., 0.2764), followed by the WCSO phase and WCSO cycle models. We also observed that in normal- and medium-density vehicles, the proposed WCSO hybrid model reacts quickly to queue occupancy decisions because it uses a cycle or phase model. In cases where vehicle density is high, the WCSO phase model has superior utility (e.g., 0.2846) and greater reactiveness to green light when the next phase turns on. These results are interesting and reflect the model's performance during traffic congestion, greatly reducing the detected incident issues and driver claims.

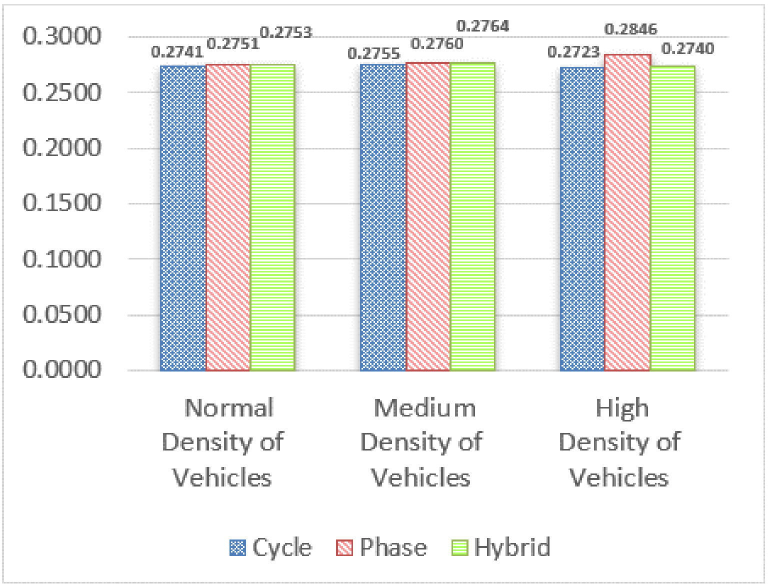


Figure 4.7 Comparison of utilities between WCSO dynamic models.

4.6.5 Comparison of Average Waiting Time of Emergency Vehicles vs Regular Vehicles in Adaptive Phase Model

From Table 4.8, we can see that the average waiting time of emergency vehicles with all priority levels is better than that of regular vehicles. This is due to the suitable pre-decision awareness that considers vehicles' priority levels.

5. CCADSTLS Performance Evaluation for a Set of Intersections

This section demonstrates the application and effectiveness of the proposed CCADSTLS. A 4×4 traffic network with sixteen intersections (see Figure 4.8) is used to evaluate the two scenarios, the first being the management of the most congested uncrossed paths, and the second also considering weather fluctuations.

All experiments were performed in our LRSD laboratory using a workstation equipped with a 13th generation Intel(R) Core (TM) i9 -13900K processor with 3.00GHz and 128 GB RAM.

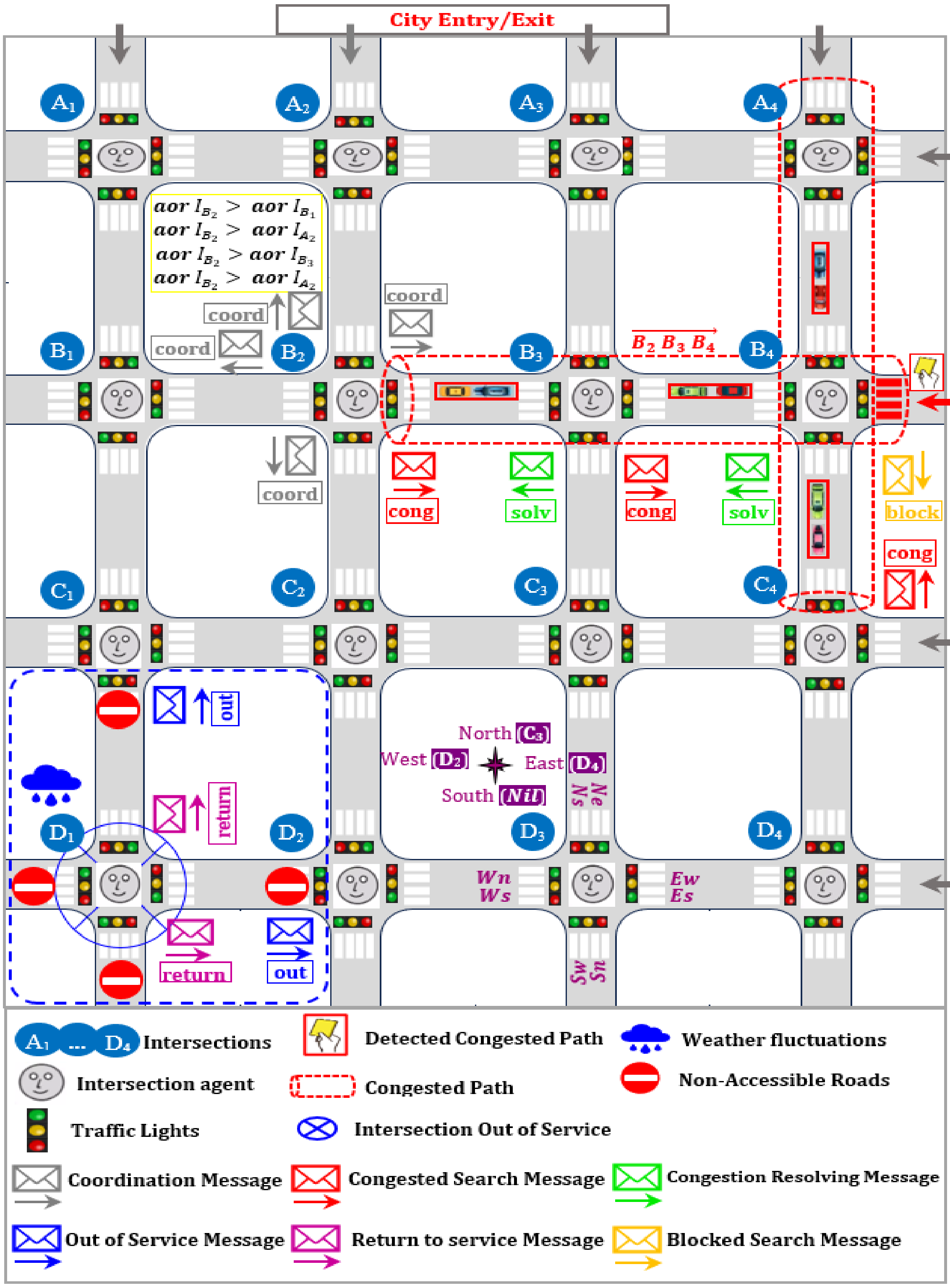


Figure 4.8 CCADSTLS based 4×4 traffic network with sixteen intersections.

5.1 Traffic Congestion Scenario

CCADSTLS enables the smart system to manage traffic congestion in real time, illustrated in Figure 4.8, as follows:

- The initial congested path is empty ($\overrightarrow{CP_{I_{B_2}}} = \emptyset$), a comparison between aor values ($aor_{I_{B_2}} > aor_{WI_{B_2}}, aor_{I_{B_2}} > aor_{NI_{B_2}}$, $aor_{I_{B_2}} > aor_{EI_{B_2}}$, and $aor_{I_{B_2}} > aor_{SI_{B_2}}$) determines.
- The intersection I_{B_2}has a high average occupancy rate threshold ($aor_{I_{B_2}} \geq c_r$), a coordination message ($msg_{\text{coord}} = aor_{I_{B_2}}$) is sent to neighbors ($WI_{B_2} = B_1$, $NI_{B_2} = A_2$, $EI_{B_2} = B_3$, $SI_{B_2} = C_2$), which can also receive the same type of message from these neighbors ($aor_{I_{B_1}}, aor_{I_{A_2}}, aor_{I_{B_3}}$, and $aor_{I_{C_2}}$). that the intersection I_{B_2} triggers the search for a more congested path ($\overrightarrow{CP_{I_{B_2}}} = \overrightarrow{B_2}$ and $state_{I_{B_2}} = Congested$).
- The congested path is determined ($\overrightarrow{CP_{I_{B_2}}} = \overrightarrow{B_2}$after $\overrightarrow{CP_{I_{B_2}}} = \overrightarrow{B_2\,B_3}$ and at the end $\overrightarrow{CP_{I_{B_2}}} = \overrightarrow{B_2\,B_3\,B_4}$), ($state_{I_{B_2}} = state_{I_{B_3}} = state_{I_{B_4}} = Congested$) and the selected phase $(Ew, Es)_{I_{B_4}}$ lights up red.
- The value of the congested path increases against $\overleftarrow{CP_{I_{B_2}}} = \overleftarrow{B_2\,B_3\,B_4}$, then $\overleftarrow{CP_{I_{B_2}}} = \overleftarrow{B_2\,B_3}$, after $\overleftarrow{CP_{I_{B_2}}} = \overleftarrow{B_2}$ and finally $\overleftarrow{CP_{I_{B_2}}} = \emptyset$, Thus, $state_{I_{B_2}} = state_{I_{B_3}} = state_{I_{B_4}} = regular$.
- Intersection I_{B_4} received a second call to participate in the search of congested path by intersection I_{C_4} ($msg_{\text{cong}} = \overrightarrow{CP_{I_{C_4}}} = \overrightarrow{C_4}$), I_{B_4} responded with a blocked search message ($msg_{\text{block}} = \overleftarrow{CP_{I_{C_4}}} = \overleftarrow{C_4}$).

5.2 Weather Disaster Scenario

Weather fluctuations can significantly affect road conditions and traffic behavior. The proposed system integrates weather conditions. As shown in Figure 4.8, the intersection I_{D_1} sends an out-of-service message (msg_{out}) to the north neighbor ($NI_{D_1} = C_1$) and the eastern neighbor ($EI_{D_1} = D_2$) when the weather fluctuates, it, ($state_{I_{D_1}} = Weather\ disasters$), I_{D_1} also blocks the city entrances on the west and south sides. The reception of messages I_{C_1} and I_{D_2} leads the blocked paths to I_{D_1} . After I_{D_1} is operational again, a second reopening message (msg_{return}) is sent to the neighbor, and all blocked roads are reopened.

5.3 Performance Evaluation Metrics

Four metrics were used to evaluate and compare the performance of CCADSTLS: global number of simulated vehicles, global average waiting time, global average occupancy rate, and global utility.

– **Global Number of Simulated Vehicles** ($\boldsymbol{Gn_v}^{\boldsymbol{c_r}}$): is the global average number of simulated vehicles computed by equation (4.18) for all intersections set I of a test value c_r,

$$Gn_v{}^{c_r} = \frac{\sum_{id=1}^{N_I} n_{v_{id}}{}^{c_r}}{N_I} \tag{4.18}$$

Where $n_{v_{id}}{}^{c_r}$ is the number of simulated vehicles in all queues for an intersection id after N_s simulations defined by equation (4.8).

– **Global Average Waiting Time** ($\boldsymbol{G\overline{awt}^{c_r}}$): is the global average value of the normalized average waiting time for all intersections set I, of a test value c_r, we use equation (4.19).

$$G\overline{awt}^{c_r} = \frac{\sum_{id=1}^{N_I} \overline{awt}_{id}{}^{c_r}}{N_I} \tag{4.19}$$

Where $\overline{awt}_{id}{}^{c_r}$ is the normalized average waiting time for intersection $id,$ after N_s simulations, calculated by equation (4.10).

– **Global Average Occupancy rate** ($\boldsymbol{Gaor^{c_r}}$): is the global average value of average occupancy rate of all intersections set I, of a test value c_r, calculated by equation (4.20):

$$Gaor^{c_r} = \frac{\sum_{id=1}^{N_I} aor_{id}{}^{c_r}}{N_I} \tag{4.20}$$

Where $aor_{id}{}^{c_r}$ is the average occupancy rate for a given intersection $id,$ after N_s simulations, defined using equation (4.12).

– **Global Utility** ($\boldsymbol{GF_u}^{\boldsymbol{c_r}}$): is a ratio of utility value of a test value c_r to total utility of all test's values. The best test value has the highest utility rate. It is calculated by equation (4.21):

$$GF_u{}^{c_r} = \frac{G\overline{U}^{c_r}}{\sum_{i=1}^{c_r} G\overline{U}^{c_r}} \tag{4.21}$$

Where $G\overline{U}^{c_r}$ is the utility weighting of the test value c_r computed by equation (4.22):

$$G\overline{U}^{c_r} = GU^{c_r} * GN^{c_r} \tag{4.22}$$

GU^{c_r} is the utility of test value c_r calculated by equation (4.23):

$$GU^{c_r} = \frac{1}{\overline{Gawt}^{c_r} + Gaor^{c_r}} \tag{4.23}$$

GN^{c_r} is the weight function of c_r computed by equation (4.24):

$$GN^{c_r} = \frac{{Gn_v}^{c_r}}{max\,({Gn_v}^{{c_r}^1}, {Gn_v}^{{c_r}^2})} \tag{4.24}$$

5.4 Simulation Results and Discussion

We test the multi-agent CCADSTLS WCSO with two coordinate rates c_r (${c_r}^1 = 0.50$, ${c_r}^2 = 0.65$), considering the High Vehicle Density (HVD) arrival scenario with three random time intervals with a time step of 500 ms: 1) Large Time Interval (HVD-LTI) = [1000ms, 9000ms], 2) Medium Time Interval (HVD-MTI) = [1000ms, 8000ms] and 3) Small Time Interval (HVD-STI) = [1000ms, 7000ms].

With a time scale (t_s = 40s) and a total simulation time of 24h. Figure 4.9 shows that Weather Disaster Scenario (WDS) records more congested paths than the Congestion Scenario (CS), which is due to the random out-of-service of two intersections in each simulation configuration. Moreover, Table 4.9 shows that WDS-${c_r}^2$ achieves the best utility value and outperforms WDS -${c_r}^1$ by 0.46%, CS-${c_r}^2$ by 7.29% and CS-${c_r}^1$ by 8.22%, so better coordination in case of a weather disaster doesn't affect system performance but improves it in contrast to the congestion scenario. Table 4.9 also shows that WDS-${c_r}^2$ has the best waiting time and CS-${c_r}^1$ has the best average occupancy.

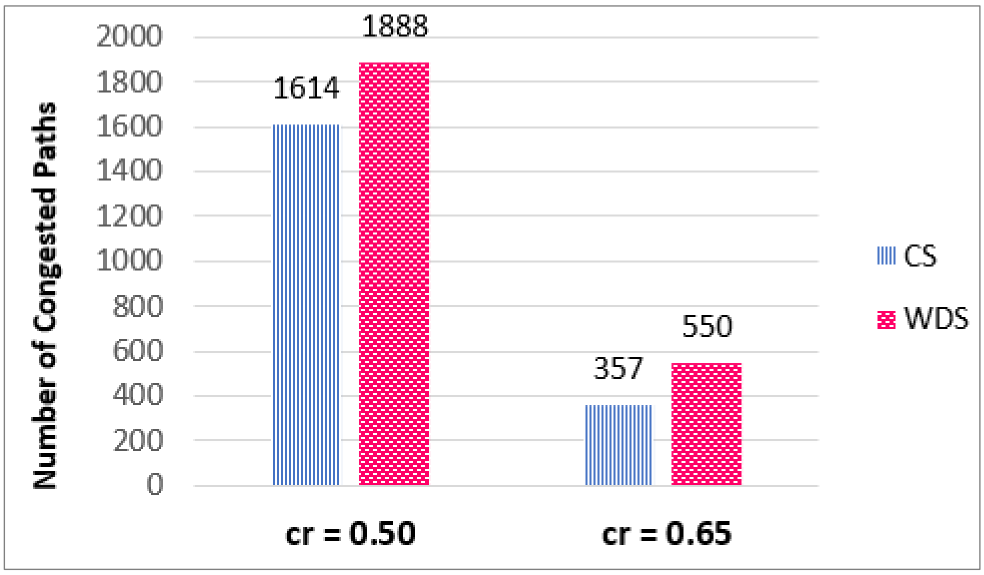


Figure 4.9 Number of congested paths detected and resolved using CCADSTLS.

Table 4.9 Comparison of CCADSTLS utility under different vehicles arrival time and coordinate rate value in high vehicles density.

Start Time Interval	**Evaluation Metrics**	**Congestion Scenario**		**Weather Disaster Scenario**	
		c_r^{1}	c_r^{2}	c_r^{1}	c_r^{2}
HVD- LTI	$G\overline{awt}$ (%)	0.6137	0.6171	0.6155	**0.6116**
	$Gaor$ (%)	**0.3868**	0.3972	0.4293	0.4238
	Gn_v (unit)	8083	8119	8835	**8830**
	GF_u	0.2443	0.2421	0.2557	**0.2579**
HVD -MTI	$G\overline{awt}$ (%)	0.7820	0.7727	0.7499	**0.7445**
	$Gaor$ (%)	0.5519	**0.5316**	0.5656	0.5684
	Gn_v (unit)	9026	9053	9793	**9800**
	GF_u	0.2365	0.2426	0.2601	**0.2608**
HVD - STI	$G\overline{awt}$ (%)	0.7700	0.7677	0.7394	**0.7363**
	$Gaor$ (%)	**0.5358**	0.5462	0.5491	0.5542
	Gn_v (unit)	9062	9202	9823	**9868**
	GF_u	0.2376	0.2397	0.2610	**0.2617**
Average Results	$\boldsymbol{G\overline{awt}}$ **(%)**	0.7219	0.7192	0.7016	**0.6975**
	$\boldsymbol{Gaor}$ **(%)**	**0.4915**	0.4917	0.5147	0.5155
	$\boldsymbol{Gn_v}$ **(unit)**	8724	8791	9484	**9499**
	$\boldsymbol{GF_u}$	0.2390	0.2414	0.2592	**0.2604**

6. Conclusion

In this chapter, we have presented a Java-based simulator for dynamic and adaptive management of smart traffic light systems, with their various GUIs, and the obtained results achieved through its application at the El-Hidhab Setif city intersection as a case study.

Furthermore, we situated our proposals to existing works by comparing the obtained results with a baseline fixed-time model and other recent methods using specific evaluation criteria and metrics. Indeed, we evaluated and compared the effectiveness of the proposed ADSTLS-WCSO algorithm of a single intersection on normal, medium, and high-density vehicles. Then, we compared the results obtained from the optimal phase of ADSTLS-WCSO which allows to reduce the vehicle waiting time and occupation rates in emergencies. The effectiveness and optimality of the ADSTLS-WCSO algorithm for a single intersection have been experimentally proven to be superior to other algorithms, particularly in minimizing waiting time. We have also demonstrated through experiments that the multi-agent extension of ADSTLS, demonstrated outstanding performance in managing multiple intersections with high vehicle density, resulting in an average vehicle waiting time of 28.4 seconds and an average lane occupancy rate of 50%. By integrating the chronological coordination of ADSTLS agents, we were able to overcome the main drawbacks of traditional agent methods and previous studies on congested paths, leading to enhanced and valuable traffic performance improvements.

General Conclusion

The global objective of this work is to develop a new STLS system, called ADSTLS (Adaptive and Dynamic Smart Traffic Light System), that enables robust, intelligent, dynamic, and adaptive management of regular and emergency vehicles to minimize the total travel time at an intersection for all drivers. As a result, the system must adapt to different contextual changes. For this reason, we need to control the traffic flow according to traffic congestion, road accidents, and weather conditions. Our objective is to efficiently manage a set of intersections to enhance flexibility and intelligent traffic light control in an optimal way, while also being able to respond to emergencies, reduce waiting times, and maximize the effectiveness of emergency services.

First, we have proposed a new hybrid traffic flow model that combines a cycle model and a phase model for optimizing traffic light planning based on traffic density. The aim is to minimize vehicle waiting time and queue occupancy at the city's intersections. This model offers adaptability and flexibility in managing traffic flow. It also enables the automatic switch from a cycle model to a phase model when the average queue occupancy rate is higher and vice versa.

Our research proposed a reliable ADSTLS system based on a heartbeat mechanism to manage fault tolerance in the real world. Furthermore, ADSTLS performs in two modes: dynamic intelligence and emergency adaptation. When congestion or emergency is detected, the phase or cycle models are dynamically managed. A Weighted Chicken Optimization Algorithm (WCSO) in the dynamic intelligence mode is used to find the best traffic flow model (i.e. the phase or cycle model) that can optimize traffic parameters by determining the best weights to optimize traffic light planning. The mono-objective algorithm for bio-inspired optimization is used to determine the most suitable phase based on the optimal weights of a single intersection using WCSO and two new objective functions as heuristics. These heuristics are based on a novel weighted function in terms of the average waiting time and occupation rate of all intersection queues to select the best phase while adhering to the constraints of traffic parameter optimization. For adaptive mode vehicles, the WCSO dynamic phase model is extended to include various priority levels for emergency vehicles. A new technique based on the distance of emergency vehicles and their priority levels is proposed for determining the priority direction

A new system, CCADSTLS, is proposed as an extension of ADSTLS. It is based on autonomous agents collaborating and coordinating together to enhance system performance by reducing execution time and identifying safe routes to address congestion at intersections.

To validate our method, we developed a JAVA simulator. This simulator was applied to a real case study of the EL-Hidhab Setif city intersection to effectively manage a single intersection, adapting to various changes such as traffic congestion through dynamic and adaptive traffic parameters management using three traffic lights models (cycle, phase, and hybrid). Furthermore, different vehicle densities (i.e. normal, medium, and high) were generated to extension the efficiency of the proposed method in terms of average waiting time, occupancy rate, and model utility. Indeed, the evaluation of this approach provides improved performance and an optimal choice of the best phase, considering the traffic flow parameters such as average waiting time, occupancy rate, the number of emergency vehicles at the intersection, and the model utility.

Our comparative study of the proposed approach to other current traffic flow management models showed the effectiveness of the ADSTLS system. Additionally, it indicated that considering the distance of emergency and their priority levels yields enhanced flexibility and effective results that resolve the congestion problems.

In order to enhance the performance of the proposed system, a coordinative system is utilized for managing multiple intersections using a multi-agent approach. A comparison of the performance results of CCADSTLS with agents has been carried out in two different scenarios, congestion and weather disaster. Experimental results revealed that intelligent coordination of agents combined with a sophisticated WCSO can achieve good results in multiple intersections even in weather environmental conditions.

Finally, we recognize that various aspects within the realm of traffic management systems could enhance our work. The simulator we developed holds promise for further improvement and real-world deployment. To achieve this, we need to consider several perspectives that we consider crucial for both short and long-term, and for the comprehensive development of the ADSTLS system.

- **Integration of non-functional requirements**: This study focuses only on functional metrics, such as the output flow rate of vehicles, average waiting time, and average occupancy rate. In certain cases, the number of requests made to the server is relevant to selecting the next phase or cycle. This metric will be considered in future research. We need to improve our system so that it can predict future traffic flow along with identifying the current traffic flow and the user requests.

- **Advanced Security of traffic data flow**: The proposed system must prioritize the security and privacy of traffic data by implementing new secure and lightweight protocols to control data exchange messages. Advanced security techniques need to be integrated to guarantee the data's privacy and confidentiality.
- **Integration of Deep learning with social media networks**: It is important that an intelligent traffic light model combined with a sophisticated deep learning approach associated with social media integration can improve decision-making based on large traffic datasets.

References

[1] N. A. Al-Madi, and A. A. Hnaif, “Optimizing Traffic Signals in Smart Cities Based on Genetic Algorithm,” *Computer Systems Science & Engineering*, vol. 40, no. 1, pp. 65-74, 2022.

[2] D. Desmira, M. Abi Hamid, N. A. Bakar, M. Nurtanto, and S. Sunardi, “A smart traffic light using a microcontroller based on the fuzzy logic,” *IAES International Journal of Artificial Intelligence*, vol. 11, no.3, pp. 809, 2022.

[3] F. Rasheed, K. L. A. Yau, R. M. Noor, C. Wu, and Y. C. Low, “Deep reinforcement learning for traffic signal control: A review,” *IEEE Access*, vol. 8, pp. 208016-208044, 2020.

[4] R. Sathiyaraj, and A. Bharathi, “An efficient intelligent traffic light control and deviation system for traffic congestion avoidance using multi-agent system,” *Transport*, vol. 35, no. 3, pp. 327-335, 2020.

[5] S. Smys, A. Basar, and H. Wang, “Artificial neural network based power management for smart street lighting systems,” *Journal of Artificial Intelligence*, vol. 2, no. 01, pp. 42-52, 2020.

[6] M. Papageorgiou, C. Diakaki, V. Dinopoulou, A. Kotsialos, and Y. Wang, “Review of road traffic control strategies,” *Proceedings of the IEEE*, vol. 91, no. 12, pp. 2043–2067, 2003.

[7] S. Kumar, A. Baliyan, A. Tiwari, A. K. Tripathi, and B. Jaiswal, “Intelligent traffic controller,” *International Journal of Information Technology*, pp. 1-13, 2019.

[8] T. Omar, D. Bovard, and H. Tran, “Smart Cities Traffic Congestion Monitoring and Control System,” *2020 ACM Southeast Conference (ACMSE 2020)*, pp. 115-121, 2020.

[9] F. Lin, S. Lin, J. Fang, and H. Hsieh, “Traffic Light Control with Real-Time Vehicle License Plate Recognition,” *Companion Proceedings of the Web Conference 2020*, pp. 100-102, 2020.

[10] A. Jovanović and D. Teodorović, “Pre-timed control for an undersaturated and over-saturated isolated intersection: a bee colony optimization approach,” *Transportation Planning and Technology*, vol. 40, no. 5, pp. 556–576, 2017.

[11] E. Segredo, G. Luque, C. Segura and E. Alba, “Optimising Real-World Traffic Cycle Programs by Using Evolutionary Computation,” *IEEE Access*, vol. 7, pp. 43915-43932, 2019.

[12] P. Balaji and D. Srinivasan, “Multi-agent system in urban traffic signal control,” *IEEE Computational Intelligence Magazine*, vol. 5, no. 4, pp. 43–51, 2010.

[13] R. Ranjan Rout, S. Vemireddy, S. R. Kumar, and D.V.L.N. Somayajulu, “Fuzzy logic-based emergency vehicle routing: An IoT system development for smart city applications,” *Computers & Electrical Engineering*, vol. 88, pp. 106839, 2020.

[14] H. Joo, S. H. Ahmed, Y. Lim, “Traffic signal control for smart cities using reinforcement learning,” *Computer Communications*, vol. 154, pp. 324-330, 2020.

[15] K.-H. N. Bui and J. J. Jung, “Cooperative game theoretic approach to traffic flow optimization for multiple intersections,” *Computers & Electrical Engineering*, vol. 71, pp. 1012–1024, 2018.

[16] E. H. Lamghari, A. Nait-Sidi-Moh, and A. Tajer, “Modular design for an urban signalized intersections network using synchronized timed Petri nets and responsive control,” *Procedia Computer Science*, vol. 170, pp. 458-465, 2020.

[17] F. Rasheed, K. A. Yau, and Y. Low, “Deep reinforcement learning for traffic signal control under disturbances: A case study on Sunway city, Malaysia,” *Future Generation Computer Systems*, vol. 109, pp. 431-445, 2020.

[18] X. C. Vuong, R. F. Mou, T. T. Vu, and H. Van Nguyen, “An Adaptive Method for an Isolated Intersection Under Mixed Traffic Conditions in Hanoi Based on ANFIS Using VISSIM-MATLAB,” *IEEE Access*, vol. 9, pp. 166328-166338, 2021.

[19] G. R Jagadeesh, J. Yang, K. M. Ng, K. L. Poh, Z. Yang, T. H. Teng, and Z. Zhuo, “A Computationally Efficient Formulation for Traffic Signal Timing Optimization of an Isolated Intersection,” *2022 8th International Conference on Control*, Decision and Information Technologies (CoDIT), vol. 1, pp. 821-826, 2022.

[20] J. Hosur, R. Rashmi, and M. Dakshayini, “Smart Traffic light control in the junction using Raspberry PI,” *3rd International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 153-156, 2019.

[21] N. Rida, M. Ouadoud, A. Hasbi, and S. Chebli, “Adaptive Traffic Light Control System Using Wireless Sensors Networks,” *IEEE 5th International Congress on Information Science and Technology (CiSt)*, pp. 552-556, 2018.

[22] A. Firdous, Indu, and V. Niranjan, “Smart Density Based Traffic Light System,” *8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 497-500, 2020.

[23] A. Frank, Y. S. Khamis Al Aamri and A. Zayegh, “IoT based Smart Traffic density Control using Image Processing,” *2019 4th MEC International Conference on Big Data and Smart City (ICBDSC)*, pp. 1-4, 2019.

[24] V. S. Raj, J. V. M. Sai, N. A. L. Yogesh, S. B. K. Preetha, and R. Lavanya, “Smart Traffic Control for Emergency Vehicles Prioritization using Video and Audio Processing,” *2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 1588-1593, 2022.

[25] P. Rosayyan, J. Paul, S. Subramaniam, and S. I. Ganesan, “An optimal control strategy for emergency vehicle priority system in smart cities using edge computing and IoT sensors,” *Measurement: Sensors*, vol. 26, pp. 100697, 2023.

[26] L. Zhong, and Y. Chen, “A Novel Real-Time Traffic Signal Control Strategy for Emergency Vehicles,” *IEEE Access*, vol. 10, pp. 19481-19492, 2022.

[27] L. Zhang, W. Zhao, and X. Zhang, “Traffic Signal Control Strategy under Adverse Weather Condition,” *18th COTA International Conference of Transportation Professionals*, pp. 1495-1504, 2018.

[28] R. Florin, and S. Olariu, “A survey of vehicular communications for traffic signal optimization,” *Vehicular Communications*, vol. 2, no. 2, pp. 70–79, 2015.

[29] Y. M. Jagadeesh, G. M. Suba, S. Karthik, and K. Yokesh, “Smart autonomous traffic light switching by traffic density measurement through sensors,” *2015 International Conference on Computers, Communications, and Systems (ICCCS)*, IEEE, pp. 123-126, 2015.

[30] W. Boubakri, W. Abdallah, and N. Boudriga, “A light-based communication architecture for smart city applications,” *2015 17th International Conference on Transparent Optical Networks (ICTON)*, pp. 1-6, IEEE, 2015.

[31] B. C. C. Meng, N. S. Damanhuri, and N. A. Othman, “Smart traffic light control system using image processing,” *IOP Conference Series: Materials Science and Engineering*. IOP Publishing, vol. 1088, no. 1, pp. 012021, 2021.

[32] A. Agrawal, and R. Paulus, “Intelligent traffic light design and control in smart cities: a survey on techniques and methodologies,” *International Journal of Vehicle Information and Communication Systems*, vol. 5, no. 4, pp. 436-481, 2020.

[33] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, and J. P. Hubaux, “Secure vehicular communication systems: design and architecture,” *IEEE Communications magazine*, vol. 46, no.11, pp. 100-109, 2008.

[34] A. M. Rook, and J. H. Hogema, “Effects of human–machine interface design for intelligent speed adaptation on driving behavior and acceptance,” *Transportation research record*, vol. 1937, no. 1, pp. 79-86, 2005.

[35] S. Mammar, “Systèmes de Transport Intelligents, modélisation, information et contrôle,” 2007.

[36] J. C.Chedjou, and K. Kyamakya, “A review of traffic light control systems and introduction of a control concept based on coupled nonlinear oscillators,” *Recent Advances in Nonlinear Dynamics and Synchronization: With Selected Applications in Electrical Engineering, Neurocomputing, and Transportation*, pp. 113-149, 2018.

[37] L. A. Klein, M. K. Mills, and D. R. Gibson, “Traffic detector handbook: Volume I,” (No. FHWA-HRT-06-108). *Turner-Fairbank Highway Research Center,*2006.

[38] L. E. Y. Mimbela, and L. A. Klein, “Summary of vehicle detection and surveillance technologies used in intelligent transportation systems”, 2007.

[39] V. Pandit, J. Doshi, D. Mehta, A. Mhatre, and A. Janardhan, “Smart traffic control system using image processing,” *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)*, vol. 3, no. 1, pp. 2278-6856, 2014.

[40] W. M. H. bin Wan Hussin, M. M. Rosli, and R. Nordin, “Review of traffic control techniques for emergency vehicles,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 13, no.3, pp. 1243-1251, 2019.

[41] R. Szeliski, “Computer vision: algorithms and applications,” *Springer Nature,* 2022.

[42] G. Wang, D. Xiao, and J. Gu, (2008, September) “Review on vehicle detection based on video for traffic surveillance,” *2008 IEEE international conference on automation and logistics*. IEEE, pp. 2961-2966.

[43] M. Sandim, R. J. Rossetti, D. C. Moura, Z. Kokkinogenis, and T. R. Rúbio, “Using GPS-based AVL data to calculate and predict traffic network performance metrics: A systematic review,” *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC),* IEEE, pp. 1692-1699.2016.

[44] N. Seenouvong, U. Watchareeruetai, C. Nuthong, K. Khongsomboon, and N. Ohnishi, “A computer vision based vehicle detection and counting system,” *2016 8th International conference on knowledge and smart technology (KST)*,” IEEE, pp. 224-227, 2016.

[45] B. Manandhar, and B. Joshi, “Adaptive traffic light control with statistical multiplexing technique and particle swarm optimization in smart cities,” *2018 IEEE*

3rd International Conference on Computing, Communication and Security (ICCCS), IEEE, pp. 210-217, 2018.

[46] J. He, and Z. Hou, “Ant colony algorithm for traffic signal timing optimization,” *Advances in Engineering Software*, vol. 43, no. 1, pp 14-18, 2012.

[47] F. Chan, and M. Tiwari, “Swarm Intelligence: focus on ant and particle swarm optimization,” *BoD–Books on Demand*, (Eds.), 2007.

[48] A. P. Engelbrecht, “Fundamentals of computational swarm intelligence,” *John Wiley & Sons, Inc.*, 2006.

[49] J. Kennedy, and R. Eberhart, “Particle swarm optimization,” *Proceedings of ICNN'95-international conference on neural networks*, IEEE, Vol. 4, pp. 1942-1948, 1995.

[50] X. Meng, Y. Liu, X. Gao, and H. Zhang, “A new bio-inspired algorithm: chicken swarm optimization,” *Advances in Swarm Intelligence: 5th International Conference, ICSI 2014, Hefei, China, Proceedings, Part I 5*, Springer International Publishing, pp. 86-94, 2014.

[51] M. Dorigo, and T. Stützle, “Ant colony optimization: overview and recent advances,” *Springer International Publishing*, pp. 311-351, 2019.

[52] D. Karaboga, “An idea based on honey bee swarm for numerical optimization,” *Technical report-tr06, Erciyes university, engineering faculty, computer engineering department*, Vol. 200, pp. 1-10, 2005.

[53] X. S. Yang, “Firefly algorithms for multimodal optimization,” *International symposium on stochastic algorithms*, Springer Berlin Heidelberg, pp. 169-178, 2009.

[54] X. S. Yang, “A new metaheuristic bat-inspired algorithm,” *Nature inspired cooperative strategies for optimization (NICSO 2010)*, Springer Berlin Heidelberg, pp. 65-74, 2010.

[55] X. S. Yang, and S. Deb, “Cuckoo search via Lévy flights,” *2009 World congress on nature & biologically inspired computing (NaBIC)*, IEEE, pp. 210-214, 2009.

[56] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey wolf optimizer,” *Advances in engineering software*, vol. 69, pp. 46-61, 2014.

[57] S. Mirjalili, and A. Lewis, “The whale optimization algorithm,” *Advances in engineering software*, vol.95, pp. 51-67, 2016.

[58] E. A. Grimaldi, F. Grimaccia, M. Mussetta, P. Pirinoli, and R. E. Zich, “Genetical swarm optimization: a new hybrid evolutionary algorithm for electromagnetic applications,” *2005 18th International Conference on Applied Electromagnetics and Communications*, IEEE, pp. 1-4, 2005.

[59] M. N. Ab Wahab, S. Nefti-Meziani, and A. Atyabi, “A comprehensive review of swarm optimization algorithms,” *PloS one*, vol. 10, no.5, pp. e0122827, 2015.

[60] K. Gao, Y. Zhang, A. Sadollah, and R. Su, “Improved artificial bee colony algorithm for solving urban traffic light scheduling problem,” *2017 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, pp. 395-402, 2017.

[61] F. Caselli, A. Bonfietti, and M. Milano, “Swarm-based controller for traffic lights management,” *AI* IA 2015 Advances in Artificial Intelligence: XIVth International Conference of the Italian Association for Artificial Intelligence, Proceedings*, Springer International Publishing, vol. 14, pp. 17-30, 2015.

[62] H. Jia, Y. Lin, Q. Luo, Y. Li, and H. Miao, “Multi-objective optimization of urban road intersection signal timing based on particle swarm optimization algorithm,” *Advances in Mechanical Engineering*, vol. 11, no. 4, pp. 1687814019842498.

[63] E. Franceries, and K. Liver, “Centralized traffic management system as response to the effective realization of urban traffic fluency,” *Archives of Transport System Telematics*, vol. 4, no. 4, pp 4-10, 2011.

[64] F. Kitahara, K. Kera, and K. Bekki, “Autonomous decentralized traffic management system,” *Proceedings 2000 International Workshop on Autonomous Decentralized System (Cat. No. 00EX449)*, IEEE, pp. 87-91, 2000.

[65] A. M. de Souza, and L. A. Villas, “A fully-distributed traffic management system to improve the overall traffic efficiency,” *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*, pp. 19-26, 2016.

[66] S. F. Cheng, M. A Epelman, and R. L. Smith, “CoSIGN: A parallel algorithm for coordinated traffic signal control,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 4, pp. 551-564, 2006.

[67] S. Mandžuka, E. Ivanjko, M. Vujić, P. Škorput, and M. Gregurić, “The use of cooperative ITS in urban traffic management,” *Intell. Transp. Syst. Technol. Appl*, pp. 272-288, 2015.

[68] L. Pan, N. Yang, L. Zhang, R. Zhang, B. Xie, and H.Yan, “Assessment of the Impact of Multi-Agent Model-Based Traffic Optimization Interventions on Urban Travel Behavior,” *Electronics*, vol. 14, no. 1, p. 13, 2024.

[69] A. Navarro-Espinoza, O. R. López-Bonilla, E. E. García-Guerrero, E. Tlelo-Cuautle, D. López-Mancilla, C. Hernández-Mejía, and E. Inzunza-González, “Traffic flow

prediction for smart traffic lights using machine learning algorithms," *Technologies*, vol. 10, no.1, pp. 5, 2022.

[70] K. Behrendt, L. Novak, and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp. 1370-1377, 2017.

[71] I. O. Olayode, L. K. Tartibu, M. O. Okwu, and A. Severino, "Comparative traffic flow prediction of a heuristic ANN model and a hybrid ANN-PSO model in the traffic flow modelling of vehicles at a four-way signalized road intersection," *Sustainability*, vol. 13, no. 19, pp. 10704, 2021.

[72] N. A. Khan, and R. Ansari, "Real-time traffic light detection from videos with inertial sensor fusion," *Proceedings of the 1st ACM SIGSPATIAL Workshop on Advances on Resilient and Intelligent Cities*, pp. 31-40, 2018.

[73] G. Liu, H. Shi, A. Kiani, A. Khreishah, J. Lee, N. Ansari, and M. M. Yousef, "Smart traffic monitoring system using computer vision and edge computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 12027-12038, 2021.

[74] J. Garcia-Nieto, E. Alba, and A. C. Olivera, "Swarm intelligence for traffic light scheduling: Application to real urban areas," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 2, pp. 274–283, 2012.

[75] Y. Liu, L. Liu, and W. P. Chen, "Intelligent traffic light control using distributed multi-agent Q learning," *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*, IEEE, pp. 1-8, 2017.

[76] A. Sachan, and N. Kumar, "Intelligent traffic control system for emergency vehicles," *IoT and Analytics for Sensor Networks: Proceedings of ICWSNUCA 2021*, Springer Singapore, pp. 151-160, 2022.

[77] P. Serafini, and W. Ukovich, "A mathematical model for the fixed-time traffic control problem," *European Journal of Operational Research*, vol. 42, no. 2, pp. 152–165, 1989.

[78] G. F. List, and M. Cetin, "Modeling traffic signal control using Petri nets," *IEEE Transactions on intelligent transportation systems*, vol. 5, no. 3, pp 177–187, 2004.

[79] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, and D. Hassabis, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp 529-533, 2015.

[80] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature*, vol. 521, no. 7553, pp 436-444. 2015.

[81] K. L. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, and P. Komisarczuk, “A survey on reinforcement learning models and algorithms for traffic signal control,” *ACM Computing Surveys (CSUR)*, vol. 50, no .3, pp. 1–38, 2017.

[82] M. Ferreira, and P. M. d'Orey, “On the impact of virtual traffic lights on carbon emissions mitigation,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no .1, pp. 284–295, 2011.

[83] J. Garcia-Nieto, A. C. Olivera, and E. Alba, “Optimal cycle program of traffic lights with particle swarm optimization,” *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 6, pp. 823–839, 2013.

[84] H. A. Abbass, and K. Deb, “Searching under multi-evolutionary pressures,” *International conference on evolutionary multi-criterion optimization. Springer*, pp. 391–404, 2003.

[85] C. Segura, C. A. Coello Coello, G. Miranda, and C. León, “Using multi-objective evolutionary algorithms for single-objective optimization,” *4OR*, vol. 11, no. 3, pp. 201–228, 2013.

[86] A. Toffolo, and E. Benini, “Genetic diversity as an objective in multi-objective evolutionary algorithms,” *Evolutionary computation*, vol. 11, no. 2, pp. 151–167, 2003.

[87] L. F. P. de Oliveira, L. T. Manera, and P. D. G. D. Luz, “Development of a Smart Traffic Light Control System with Real-Time Monitoring,” *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3384-3393, Mar. 2021.

[88] P. Choudhary, and R.K. Dwivedi, “A novel algorithm for traffic control using thread based virtual traffic light,” *International Journal of Information Technology*, vol. 14, no. 1, pp. 115–124, 2022.

[89] M. B. Younes, A. Boukerche, and F. De Rango, “SmartLight: A smart efficient traffic light scheduling algorithm for green road intersections,” *Ad Hoc Networks*, vol. 140, pp. 103061, 2023.

[90] R. Zhu, S. Wu, L. Li, P. Lv, and M. Xu, “Context-Aware Multiagent Broad Reinforcement Learning for Mixed Pedestrian-Vehicle Adaptive Traffic Light Control,” *IEEE Internet of Things Journal*, vol. 9, no. 20, pp. 19694-19705, Oct. 2022.

[91] D. Anand, I. Mavromatis, P. Carnelli, and A. Khan, “A federated learning-enabled smart street light monitoring application: benefits and future challenges,” *Proceedings*

of the 1st ACM Workshop on AI Empowered Mobile and Wireless Sensing, pp. 7-12, Oct.2022.

[92] R. Ducrocq, and N. Farhi, "Deep reinforcement Q-learning for intelligent traffic signal control with partial detection," *International Journal of Intelligent Transportation Systems Research*, vol. 21, no 1, pp.192-206, 2023.

[93] A. Zaji, Z. Liu, T. Bando, and L. Zhao, "Ontology-Based Driving Simulation for Traffic Lights Optimization," *ACM Transactions on Intelligent Systems and Technology*, vol. 14, no 3, p. 1-26, 2023.

[94] K. D. S. A. Munasinghe, T. D. Waththegedara, I. R. Wickramasinghe, H. M. O. K. Herath and V. Logeeshan, "Smart Traffic Light Control System Based on Traffic Density and Emergency Vehicle Detection," *2022 Moratuwa Engineering Research Conference (MERCon)*, pp. 1-6, 2022.

[95] F. Kiani, and Ö.F. Saraç, "A novel intelligent traffic recovery model for emergency vehicles based on context-aware reinforcement learning," *Information Sciences*, vol. 619, pp 288-309, 2023.

[96] G. Li, R. Gomez, K. Nakamura, and B. He, "Human-centered reinforcement learning: A survey," *IEEE Transactions on Human-Machine Systems*, vol. 49, no. 4, pp. 337–349, 2019.

[97] D. Pandey, and P. Pandey, "Approximate Q-learning: An introduction," *2010 second international conference on machine learning and computing*. IEEE, pp. 317–320, 2010.

[98] Y. K. Chin, N. Bolong, A. Kiring, S. S. Yang, and K. T. K. Teo, "Q-learning based traffic optimization in management of signal timing plan," *International Journal of Simulation, Systems, Science & Technology*, vol. 12, no. 3, pp. 29–35, 2011.

[99] W. Liu, G. Qin, Y. He, and F. Jiang, "Distributed cooperative reinforcement learning-based traffic signal control that integrates V2X networks' dynamic clustering," *IEEE transactions on vehicular technology*, vol. 66, no. 10, pp. 8667–8681, 2017.

[100] Y.-S. Huang, Y.-S. Weng, and M. Zhou, "Design of Regulatory Traffic Light Control Systems with Synchronized Timed Petri Nets," *Asian Journal of Control*, vol. 20, no. 1, pp. 174–185, 2018.

[101] G. F. List, and M. Mashayekhi, "A Modular Colored Stochastic Petri Net for Modeling and Analysis of Signalized Intersections," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 3, pp. 701–713, 2016.

[102] W. Zhao, Y. Xing, X. Ban, and C. Guo, "Probabilistic Forecasting of Traffic Flow Using Multikernel Based Extreme Learning Machine," *Scientific Programming*, vol. 2017, no 1, pp. 2073680, 2017.

[103] S. Nesmachnow, R. Massobrio, E. Arreche, C. Mumford, A. C. Olivera, P. J. Vidal, and A. Tchernykh, "Traffic lights synchronization for Bus Rapid Transit using a parallel evolutionary algorithm," *International Journal of Transportation Science and Technology*, vol. 8, no. 1, pp. 53–67, 2019.

[104] E.A. Sofronova, A.A. Belyakov, and D.B. Khamadiyarov, "Optimal Control for Traffic Flows in the Urban Road Networks and Its Solution by Variational Genetic Algorithm," *Procedia Computer Science*, vol. 150, pp. 302-308, 2019.

[105] A. A. A. Lah, L. A. Latiff, R. A. Dziyauddin, H. M. Kaidi, and N. Ahmad, "Smart traffic monitoring and control architecture and design," *2017 IEEE 15th Student Conference on Research and Development (SCOReD)*, IEEE, pp. 72–76, 2017.

[106] K. Deb, A. Pratap, S. Agarwal, and T. A. M. T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[107] J. Garcia-Nieto, J. Ferrer, and E. Alba, "Optimising traffic lights with metaheuristics: Reduction of car emissions and consumption," *2014 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 48–54, 2014.

[108] E. Segredo, C. Segura, and C. León, "Fuzzy logic-controlled diversity-based multiobjective memetic algorithm applied to a frequency assignment problem," *Engineering Applications of Artificial Intelligence*, vol. 30, pp. 199–212, 2014.

[109] L. T. Bui, H. A. Abbass, and J. Branke, "Multiobjective optimization for dynamic environments," *2005 IEEE Congress on Evolutionary Computation*, IEEE, vol. 3, pp. 2349– 2356, 2005.

[110] H. M. Alhassan, and J. Ben-Edigbe, "Effect of rain on probability distributions fitted to vehicle time headways," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 2, no. 2, pp. 31–37, 2012.

[111] W. F. Adams, "Road traffic considered as a random series," *Institution of Civil Engineers*,1937.

[112] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Internet of Things applications: A systematic review," *Computer Networks*, vol. 148, pp. 241–261, 2019.

[113] X. Li, L. Zhu, X. Chu, and H. Fu, “Edge computing-enabled wireless sensor networks for multiple data collection tasks in smart agriculture,” *Journal of Sensors*, vol. 2020, no. 1, 2020.

[114] A. S. Syed, D. Sierra-Sosa, A. Kumar, and A. Elmaghraby, “IoT in smart cities: a survey of technologies, practices and challenges,” *Smart Cities*, vol. 4, no. 2, pp. 429–475, 2021.

[115] G. Javadzadeh, and A. M. Rahmani, “Fog computing applications in smart cities: A systematic survey,” *Wireless Networks*, vol. 26, no. 2, pp. 1433–1457, 2020.

[116] B. Ji, X. Zhang, S. Mumtaz, C. Han, C. Li, H. Wen, and D. Wang, “Survey on the internet of vehicles: Network architectures and applications,” *IEEE Communications Standards Magazine*, vol. 4, no. 1, pp. 34–41, 2020.

[117] B. Pradhan, S. Bhattacharyya, and K. Pal, “IoT-based applications in healthcare devices,” *Journal of healthcare engineering*, vol. 2021, no. 1, pp. 6632599, 2021.

[118] S.Ketu, and Mishra, P. K. “Internet of Healthcare Things: A contemporary survey,” *Journal of Network and Computer Applications*, vol. 192, pp. 103179, 2021.

[119] H. Yar, A. S. Imran, Z. A. Khan, M. Sajjad, and Z. Kastrati, “Towards smart home automation using IoT-enabled edge-computing paradigm,” *Sensors*, vol. 21, no.14, pp. 4932, 2021.

[120] T. Malche, and P. Maheshwary, “Internet of Things (IoT) for building smart home system,” *2017 International conference on I-SMAC (IoT in social, mobile, analytics and cloud) (I-SMAC)*, IEEE, pp. 65–70, 2017.

[121] X. Liao, M. Faisal, Q. QingChang, and A. Ali, “Evaluating the Role of Big Data in IIOT-Industrial Internet of Things for Executing Ranks Using the Analytic Network Process Approach,” *Scientific Programming*, vol. 2020, no. 1, pp. 8859454, 2020.

[122] W. Z. Khan, M. H. Rehman, H. M. Zangoti, M. K. Afzal, N. Armi, and K. Salah, “Industrial internet of things: Recent advances, enabling technologies and open challenges,” *Computers & electrical engineering*, vol. 81, pp. 106522, 2020.

[123] L. Da Xu, W. He, and S. Li, “Internet of things in industries: A survey,” *IEEE Transactions on industrial informatics*, vol. 10, no. 4; pp 2233-2243, 2014.

[124] A. I. Levina, A. S. Dubgorn, and O. Y. Iliashenko, “Internet of things within the service architecture of intelligent transport systems,” *2017 European Conference on Electrical Engineering and Computer Science (EECS)*, IEEE. pp. 351–355, 2017.

[125] J. M. Talavera, L. E. Tobón, J. A. Gómez, M. A. Culman, J. M. Aranda, D. T. Parra, and L. E. Garreta, “Review of IoT applications in agro-industrial and environmental fields,” *Computers and Electronics in Agriculture*, vol. 142, pp. 283–297, 2017.

[126] E. Almazrouei, R. M. Shubair, and F. Saffre, “Internet of nanothings: Concepts and applications,” *arXiv preprint arXiv:1809.08914*, 2018.

[127] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. P. Flötteröd, R. Hilbrich, and E. Wießner, “Microscopic traffic simulation using sumo,” *2018 21st international conference on intelligent transportation systems (ITSC)*, IEEE, pp. 2575–2582, 2018.

[128] https://www.ptvgroup.com/fr/solutions/produits/ptv-visum/.

[129] https://www.inrosoftware.com/en/products/emme/.

[130] T. Urbanik, A. Tanaka, B. Lozner, E. Lindstrom, K. Lee, S. Quayle, and D. Bullock “Signal timing manual,” Vol. 1, Washington, *DC: Transportation Research Board,* 2015.

[131] P. Kulkarni, A. Khatri, P. Banga, and K. Shah, “Automatic Number Plate Recognition (ANPR) system for Indian conditions,” *2009 19th International Conference Radio elektronika*, pp. 111-114, 2009.

[132] Q. Hu, and W. Tu, “A vehicle message scheduling scheme for vehicle trust management,” *In 2021 IEEE 46th Conference on Local Computer Networks (LCN) IEEE*, pp. 407-410, 2021.

[133] S. Deb, X. Z. Gao, K. Tammi, K. Kalita, and P. Mahanta, “Recent studies on chicken swarm optimization algorithm: a review (2014–2018)”. *Artificial Intelligence Review,* vol. 53, pp. 1737-1765, 2020.

List of Publications

Refereed Journal:

R. Zerroug, Z. Aliouat, M. Aliouat, and A. ALTI, “Adaptive and dynamic smart traffic light system for efficient management of regular and emergency vehicles at city intersection,” *IET Smart Cities*, vol. 6, no. 4; pp. 387-421, 2024. https://doi.org/10.1049/smc2.12090

R. Zerroug, Z. Aliouat, M. Aliouat, and A. ALTI, “Smart Cities Trafic Light Management Systems Review: Models and Approaches,” *Preprint*, 2023. https://doi.org/10.21203/rs.3.rs-3257486/v1

Referred International Conference:

R. Zerroug, Z. Aliouat, and A. ALTI, “Managing Congestions at Multiple Intersections During Disasters with Chronological Coordination of Intelligent Traffic Light Management,” *In 2024 International Conference on Information and Communication Technologies for Disaster Management (ICT-DM), Setif, Algeria, IEEE*, pp. 1-7, 2024. https://doi.org/10.1109/ICT-DM62768.2024.10798932

Referred National Conference:

R. Zerroug, Z. Aliouat, M. Aliouat, and A. ALTI, “Smart Traffic Signal Management for a Unique Intersection using Wireless Sensors Network,” *Conference on Trends and Advances in discrete Mathematics, operations Research, scientific Information and Computer Science (TAMARICS’2022)*, 2022. http://tamarics.cerist.dz/

R. Zerroug, and M. Aliouat, “Traffic Lights Synchronization in Smart Cities,” *Doctorial Day*, Setif1 University, Faculty of Sciences, June 09, 2022.

R. Zerroug, and Z. Aliouat, “Traffic Lights Synchronization in Smart Cities,” *Doctorial Day*, Setif1 University, Faculty of Sciences, May 04, 2023.

Abstract . Nowadays, STLS (Smart Traffic Light System) is widely adopted by smart cities to control traffic lights. They often rely on camera and sensors to collect traffic data. However, STLS in urban areas is not flexible enough to efficiently manage traffic congestions and emergencies. Therefore, the increasing need of automatic synchronized STLS system of various traffic controllers at different intersections which performed with reduced occupancy and waiting time benefits becomes a major concern. This is crucial for meeting the optimized traffic parameters of multiple intersections and driver's needs. Our objective is to develop a new system, called ADSTLS (Adaptive and Dynamic Smart Traffic Light System) to address traffic management at an intersection, solving the challenging problem of traffic congestion while prioritizing emergency vehicles. Therefore, we have proposed a system with new hybrid traffic flow model that combines a cycle model and a phase model for optimizing and efficiently managing traffic light planning, along with a decision-making approach focused on reducing congestion and average vehicle waiting time. By collecting traffic data, the system automatically extracts useful traffic information using computer vision and computing standard traffic metrics. Moreover, we propose two-traffic modes for regular and emergency vehicles to achieve an optimal decision-making process. The first mode, the dynamic mode, select the best phase using the Weight Chicken Swarm Optimization (WCSO) algorithm to ensure an optimal vehicle waiting time and queue occupancy at the city's intersection. The second mode, the adaptive mode, determines the priority direction based on the distance of emergency vehicles and their priority levels. We have also proposed extending ADSTLS system using the multi-agent aspect to improve the system's performance in terms of execution time. To demonstrate our approach, we have presented a simulator applied to a real case study of EL-Hidhab Setif city intersection. The experimental results show a decrease in the average vehicle waiting time (31 s) and queue occupation rate (33.82%) across all simulated traffic scenarios. Furthermore, compared to other car types, emergency vehicles usually had much shorter wait times.

Résumé. De nos jours, les systèmes de feux de circulation intelligents (STLS) sont largement adoptés dans les villes intelligentes pour contrôler les feux de circulation. Ils s'appuient souvent sur des équipements dédiés tels que : les caméras, les capteurs, etc. pour collecter des données sur le trafic. Cependant, les STLSs dans les zones urbaines ne sont pas assez flexibles pour gérer efficacement les embouteillages et les véhicules d'urgences. Par conséquent, le besoin croissant d'un système STLS automatique et synchronisé de divers contrôleurs de trafic à différentes intersections, qui fonctionne avec une occupation réduite et des avantages en termes de temps d'attente, devient une préoccupation majeure. Ceci est crucial pour répondre aux paramètres de trafic optimisés de plusieurs intersections et aux besoins des conducteurs. Notre objectif est de développer un nouveau système, appelé ADSTLS (Système adaptatif et dynamique de feux de circulation intelligents) pour gérer le trafic à une intersection, en résolvant le problème difficile des embouteillages tout en donnant la priorité aux véhicules d'urgence. . Nous avons donc proposé un système avec un nouveau modèle hybride de flux de trafic qui combine, un modèle de cycle et un modèle de phases pour optimiser et gérer efficacement la planification des feux de circulation, ainsi qu'une approche décisionnelle axée sur la réduction des embouteillages et du temps d'attente moyen des véhicules. En collectant des données sur le trafic, le système extrait automatiquement des informations utiles sur le trafic en utilisant la vision par ordinateur et en calculant des métriques standard du trafic. En outre, nous proposons deux modes de circulation pour les véhicules ordinaires et les véhicules d'urgence afin de parvenir à un processus décisionnel optimal. Le premier mode : le mode dynamique, sélectionne la meilleure phase à l'aide de l'algorithme d'optimisation « Weight Chicken Swarm Optimization (WCSO) » afin de garantir un temps d'attente optimal pour les véhicules et l'occupation des files d'attente aux intersections de la ville. Le second mode, le mode adaptatif, détermine la direction prioritaire en fonction de la distance des véhicules d'urgence et de leurs niveaux de priorité. Nous avons également proposé d'étendre le système ADSTLS en utilisant le paradigme multi-agents pour améliorer la performance du système en termes de temps d'exécution. Pour montrer la faisabilité de notre approche, nous avons réalisé un simulateur que nous avons appliqué à une étude de cas réelle de l'intersection de la cité EL-Hidhab, ville de Sétif. Les résultats expérimentaux ont montré une diminution du temps d'attente moyen des véhicules (31 s) et un taux d'occupation des files d'attente (33,82 %) dans tous les scénarios de trafic simulés. En outre, par rapport aux autres types de véhicules, les véhicules d'urgence ont également des temps d'attente beaucoup plus courts.

ملخص. في الوقت الحاضر، يتم اعتماد نظام إشارات المرور الذكية (STLS) على نطاق واسع في المدن الذكية للتحكم فيها. وغالباً ما تعتمد على الكاميرات وأجهزة الاستشعار لجمع بيانات حركة المرور. ومع ذلك، لا يتسم نظام STLS في المناطق الحضرية بالمرونة الكافية لإدارة الازدحام المروري وحالات الطوارئ بكفاءة. حيث يؤدي إلى مصدر قلق كبير ولذلك، تصبح الحاجة المتزايدة إلى نظام STLS آلي ومتزامن لمختلف وحدات التحكم في حركة المرور في التقاطعات المختلفة التي تؤدي إلى خفض معدل امتلاء طوابير الوقوف وتقلل من وقت الانتظار. وهذا أمر بالغ الأهمية لتلبية معايير حركة المرور المثلى للتقاطعات المتعددة واحتياجات السائقين. ويتمثل هدفنا في تطوير نظام جديد يسمى ADSTLS(نظام إشارات المرور الضوئية الذكي المتكيف والديناميكي) لمعالجة إدارة حركة المرور عند التقاطع، وحل المشكلة الصعبة المتمثلة في الازدحام المروري مع إعطاء الأولوية لسيارات الطوارئ. لذلك، اقترحنا نظامًا بنموذج هجين جديد لتدفق حركة المرور يجمع بين نموذج الدورة ونموذج المرحلة لتحسين تخطيط إشارات المرور وإدارتها بكفاءة، إلى جانب نهج اتخاذ القرار الذي يركز على تقليل الازدحام ومتوسط وقت انتظار المركبات. من خلال جمع بيانات حركة المرور، يستخرج النظام تلقائيًا معلومات مفيدة عن حركة المرور باستخدام الرؤية الحاسوبية وحساب مقاييس حركة المرور. علاوة على ذلك، نقترح وضعين لحركة المرور للمركبات العادية ومركبات الطوارئ لتحقيق عملية اتخاذ القرار الأمثل. يحدد الوضع الأول، وهو الوضع الديناميكي، أفضل مرحلة باستخدام خوارزمية تحسين سرب الدجاجة الموزونة (WCSO) لضمان أفضل وقت انتظار للمركبات وإشغال طابور الانتظار في تقاطع المدينة. يحدد الوضع الثاني، الوضع التكيفي، اتجاه الأولوية بناءً على مسافة مركبات الطوارئ ومستويات أولويتها. اقترحنا أيضًا توسيع نظام ADSTLS باستخدام جانب تعدد العملاء لتحسين أداء النظام من حيث وقت التنفيذ. ولإثبات نهجنا، قدمنا محاكاة تم تطبيقها على دراسة حالة حقيقية لتقاطع بحي الهضاب مدينة سطيف. تُظهر النتائج التجريبية انخفاضًا في متوسط وقت انتظار المركبات (31 ثانية) ومعدل إشغال طابور الانتظار (33.82%) في جميع سيناريوهات حركة المرور التي تمت محاكاتها. علاوة على ذلك، بالمقارنة مع أنواع السيارات الأخرى، يكون لمركبات الطوارئ أوقات انتظار أقصر بكثير.