SELF SYNCHRONIZATION OF MOVING VEHICLES

A DISSERTATION IN
Computer Science and Networking

Presented to the Faculty of the University
of Missouri-Kansas City in partial fulfillment of
the requirements for the degree

DOCTOR OF PHILOSOPHY

by
AMOL SHASHIKANT KHEDKAR

M.C.A., Amaravati University Maharashtra (India), 2001

Kansas City, Missouri
2015

SELF-SYNCHRONIZATION OF MOVING VEHICLES

Amol S Khedkar, Candidate for the Doctor of Philosophy

University of Missouri, Kansas City, 2015

ABSTRACT

In this dissertation, we investigate and develop a novel scheduling scheme for conflict-free movement of vehicles at road intersections. We claim that our scheduling scheme not only guarantees conflict-free movement at any intersection, it also provides nonstop movement for the maximum possible number of vehicles at multiple intersections on its route. If it is not possible to provide a nonstop movement to a vehicle, the proposed scheme works to minimize the waiting time for each of the vehicles at an intersection.

At present, traffic signals manage (synchronize) the conflict-free movement of vehicles on road intersections (common resource). These signals enforce the traffic rules to manage conflict-free movement of vehicles. Each side of traffic is allotted a stipulated time slot for crossing the intersection. The existing traffic signal scheme works well; however, it has a number of issues. These include the effect of changing traffic volume on traffic flow and indecisiveness of human drivers, etc., which can be eliminated by using state of the art technology. Motivated by the need of improving conflict-free traffic flow at road intersections, a large number of commercial and academic institutions have been taking a serious interest in

solving some of these issues. One of the main approaches is to create a virtual environment so that information of traffic on an intersection can be transmitted to adjacent intersections in order to provide stoppage free movement of vehicles.

In this dissertation, we investigate the traffic regulation problem from the point of view of "scheduling vehicle movement at road intersections". We develop innovative scheduling schemes that require minimum human intervention in conflict-free traffic movement at intersections. This leads to the mechanism of self-synchronization of vehicles at these intersections in which conflicting vehicles mutually synchronize their movement using real-time contextual information. In self synchronization approach, vehicles that use the shared resources (intersections) communicate with each other and make a decision who will utilize the resource first based on a fair scheduling algorithm.

To investigate and develop our fair scheduling algorithm, contextual information related to each of the vehicles must be exchanged among the vehicles in real-time. Existing communication protocols that are based on collision avoidance (of data packets) or collision detection and resolution may not work satisfactorily. The self-synchronization scheme generates a very dynamic, rapidly changing network of vehicles that requires a unique protocol for reliable real time data communication. So we have developed a new protocol for exchanging contextual information among vehicles.

APPROVAL PAGE

The faculty listed below, appointed by the Dean of School of Computing and Engineering, have examined a dissertation titled "Self Synchronization of Moving Vehicles," presented by Amol S. Khedkar, candidate for the Doctor of Philosophy degree, and certify that in their opinion it is worthy of acceptance.

Supervisory Committee

Vijay Kumar, Ph.D., Committee Chair
Department of Computer Science and Electrical Engineering

Cory Beard, Ph.D.
Department of Computer Science and Electrical Engineering

Praveen Rao, Ph.D.
Department of Computer Science and Electrical Engineering

Xiaojun Shen, Ph.D.
Department of Computer Science and Electrical Engineering

Lein Harn, Ph.D.
Department of Computer Science and Electrical Engineering

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ACK : Acknowledgement

BLUE : (not and Acronym it is a Name)

CHOKE : CHOose and Keep for Responsive Flow,
CHOose and Kill for Unresponsive Flow

CPU : Central Processing Unit

CALM : Communication Access for Land Mobiles

CSMA/CA : Carrier Sense Multiple Access Collision Avoidance

DSRC : Dedicated Spectrum For Short Range Communication

EWMA : Exponentially-Weighted Moving Average

FCC : Federal Communications Commission

FCFS : First Come First Serve

FDMA: Frequency Division Multiple Access

FRED : Flow Random Early Drop

Ghz / Mhz : Giga Hertz / Mega Hertz

GPRS : General Packet Radio Service

GPS : Global Positioning System

HSA : Hybrid Simulated Annealing

IEEE : Institute of Electrical and Electronics Engineering

ISO : International Organization for Standardization

ITS: Intelligent Transportation System

IETF : Internet Engineering Task Force

JSSP : Job Shop Scheduling Problem

LR : Langrage Relaxation

MAC : Media Access Control

MATLAB : Matrix Laboratory

MNSSA : Multi-Neighborhood Search Simulated Annealing

MPH : Miles Per Hour

OFDM : Orthogonal Frequency-Division Multiplexing

PHY : Physical Layer of Network Protocol

PSID : Provider Service Identification

RED : Random Early Drop

RITA : Research and Innovative Technology Administration

SFB : Stochastic Fair BLUE

SPT : Shortest Processing Time First

TCP/IP : Transmission Control Protocol , Internet Protocol

TDMA : Time Division Multiple Access

UDP : User Datagram Protocol

UMTS: Universal Mobile Telecommunications System

V2V : Vehicle to Vehicle

V2I : Vehicle to Infrastructure

WAVE : Wireless Access in Vehicular Environment

WIFI : Trademark of the Wi-Fi Alliance

WLAN : Wireless Local Area Network

WSMP : Wave Short Message Protocol

USDOT : United States Department of Transportation

CHAPTER 1

INTRODUCTION

Vehicle Safety and improvement in traffic flow has been an important topic of research for the past several years. Every big city in the world is suffering from traffic congestion during peak hours. The government tries to improve the public transportation systems so that the number of vehicles on the roads can be reduced without affecting the traffic flow. New freeways and flyovers are built to reduce the traffic load from certain busy routes at intersections. It is a fact that traffic congestion on roads will continue to remain a serious problem because of a number of factors such as social human behavior and technology to name a few. Traffic congestion, especially at road intersections, is a unique problem that will continue to exist on city roads irrespective of increase or decrease of traffic volume. Our solution tries to manage this issue.

Auto makers are emphasizing on improving the safety and comfort of drivers. New technology is being developed that predicts hazards ahead of time. Automated systems are made to prevent collisions due to the negligence of the drivers. Vehicles can park themselves [14] and some vehicles can drive without the help of human drivers on certain routes [19]. All these advancements do provide comfort to travelers, and yet, the inconvenience and risk due to congestion still remains.

The existing methods of traffic control use traffic lights and traffic signals, such as stop and yield signs, etc. These methods work fine; however, they have a number of limitations that result in poor utilization of resources (time, road intersections, etc.) The traffic signals work on a time share basis where intersections can only be used by one side at a time

(group of nonconflicting directions such as South-North etc.). Each side of the traffic is allotted a stipulated time slot for crossing an intersection. When the time slot for one side expires then it opens the intersection for other side. This is a typical example of mutual exclusion in traffic systems. We identify the following limitations of conventional methods for achieving mutual exclusion:

a. In peak time, the time interval of signals for each side is higher, and when one side is not very crowded, then the other side may have to wait, even if the intersection is available.

b. A vehicle may have to wait on a red signal even though there is no other vehicle occupying the intersection.

c. It is very difficult to set an optimal time interval for a signal to turn red from green. It may vary from time to time.

d. Electricity is wasted to keep these traffic signals working even if there are no vehicles at the intersection.

e. When a signal is green or red, it is easy to decide whether to stop or to cross the intersection but when a signal is yellow, it is difficult to decide whether to stop or cross the intersection before the light turns red. This may cause a collision situation.

f. There rules or guidelines for yield on left turn with a solid green light are not specific and the decision to move is on driver's discretion.

g. At peak times, signals cause congestion on roads and at ramps adjacent to traffic signals.

A driver's judgment is a vital part of a traffic signal approach. This becomes necessary when traffic rules are fuzzy, such as at yield signs, and prepare to stop signs, etc. In these

situations, the drivers may fail to make a correct decision to avoid collisions and such decisions may not always be possible. Similar indecisiveness happens at traffic lights also. Consider an instance when the traffic light turns yellow. Some drivers speed up to cross the intersection before the light changes to red and some drivers slam on their breaks creating a hazard for the drivers behind them. Traffic signals at some locations are controlled using fuzzy logic (micro-controller based system) so that it open or closes a side based on number of vehicles present on that side. It works fine when an intersection is saturated, but in sparse traffic, it opens the intersection after a vehicle approaches the intersection and stops. This makes almost every vehicle stop at the intersection even though no other vehicle is present.

We propose shifting the process that enforces mutual exclusion from traffic lights to conflicting vehicles. Thus, instead of using a traffic light, conflicting vehicles will mutually decide who and how many vehicles will go through the intersection, and who will wait. Thus, we refer to the process of enforcing mutual exclusion through a vehicle's "state" exchange (handshake) as "self-synchronization." Note that we are not changing the traffic light logic for achieving mutual exclusion; rather, we are eliminating the role of the third party, i.e. the traffic light in enforcing mutual exclusion to achieve a conflict-free flow of vehicles. This is a unique case of real-time scheduling algorithmically that decides which vehicle will use the intersection that is clearly identical to a critical section. Under this situation therefore, we define the dissertation problem as complex real-time scheduling. It is relatively complex because there are a total of twelve freedom of movement vehicles that the intersection encounters, and in addition to this, there is the presence of human discretion. In order to develop an efficient and high-performance scheduling scheme, our approach requires that

every vehicle must collect and exchange contextual information in real-time to achieve self-synchronization.

We identify that existing communication protocols are unable to achieve a timely exchange of contextual information, and a unique communication protocol is needed that must satisfy the temporal and spatial requirements. To achieve this exchange of information among vehicles in the vicinity of an intersection, we developed a unique communication protocol that guarantees uninterrupted communication and availability of the contextual information to a "relevant" number of vehicles on time and at the right place. The information is captured by the intelligent system in the vehicle and shares it with other vehicles in the vicinity thus enforcing self-synchronization so that every vehicle can cross the intersection without any conflict and without the support of any external agent.

The self-synchronization scheme is based on the communication between every pair of conflicting vehicles. The scheme makes sure that the decision made between each conflicting pair is (a) acceptable to them, (b) each party honors and implements the decision, and (c) enters the intersection as agreed in the decision. Since each conflicting vehicle follows the three criteria in a trustworthy manner, the collision situation is eliminated. The issue of an ad-hoc situation (human discretion) also does not arise because the decision is binding.

1.1. Contributions of this dissertation

A significant amount of work on improving driving comfort, communication among vehicles, self-parking, driverless cars, etc. have been done mainly by auto companies. However, the core problem of achieving conflict-free movement of vehicles at road intersections with minimum or no human intervention has not been investigated, even though

this is one of the core issues of traffic management. We emphasize that the contributions of our work is quite significant and unique. The research work done in this dissertation provides solutions to various complicated problems in vehicular environments that are listed below.

a. This dissertation establishes a protocol for wireless and mobile communication in vehicular environments to enable reliable vehicle to vehicle (V2V) and vehicle to infrastructure (V2I) communication.

b. This dissertation provides a standard message format for transmitting vehicular contextual information in real time.

c. It provides a solution to solve traffic management issues at intersections without the need of outside traffic management authority. It provides a tool to establish communication among vehicles and a real time scheduling algorithm to synchronize their movements at an intersection.

d. It provides a real time scheduling algorithm to enable nonstop movement of vehicles for upto three consecutive intersections.

e. The solution provided in this dissertation makes sure that all vehicles that are entering an intersection from any of the directions should be treated fairly and does not have to wait longer than the vehicles moving from any other direction at that intersection.

In the second chapter, we provide a detailed review of papers that are closely related to our work. As mentioned earlier, the majority of work on vehicle technology has been conducted by auto manufacturers. There is some work by university researchers and we review them here too.

CHAPTER 2

REVIEW OF EXISTING WORKS

We categorize our review of earlier work into (a) work by auto manufacturers and Google, (b) work on real-time scheduling, and (c) work on network protocol for vehicular environments.

2.1. Works by Auto Manufacturers and Google

Earlier and ongoing work on vehicle management has concentrated mainly on improving (a) driving comfort, (b) personal and vehicle safety, (c) lane merging and parking, and (d) driver-less vehicle management. The below reviewed work either uses sensors and radar technology to sense the surrounding environment of vehicles for safety purposes or uses a vehicle to vehicle (V2V) communication technique to send short alert signals. Though this work does not directly impact the research work on self-synchronization, we claim that these already developed techniques can aid in the development of fully autonomous vehicular environments when combined with the novel self-synchronization approach that is described later in this dissertation. We provide a brief review of work (research and development) that is related to our work.

Safe Intelligent Mobility-Test Field Germany (SIM-TD) [38]. This is a German (Deutschland) project. Its objective is to provide a safe and intelligent transportation system that deals with car-to-infrastructure (V2I) [38] and car-to-car (V2V) [38] communications. According to their vision, car-to-x communication can leverage the transmission of required information between vehicles as well as traffic control centers. Using this information, the users who are going to use this road later can be informed of any possible hazardous situations,

so that they can appropriately react in time. Using the anonymous car-to-x information, the roadside infrastructure that regulates the traffic can be tuned according to traffic requirements. This information can also support the road users in selection of an appropriate route that can reduce their journey time and is comfortable and safe.

SIM-TDs objective has been the integration of vehicle, communication and traffic technologies into one system. Their research emphasizes that vehicles recognize information on driving conditions and risks in a standardized way and forward such information precisely so that they can be translated into traffic control measures such as variable traffic signs or vehicle-related systems as quickly as possible. In this context, the research focuses on the question of how specific information transferred to individual vehicles can optimize overall traffic efficiency and safety. Within the SIM-TD project, the synchronization between collective control and individual traffic guidance and its efficient implementation will be shaped and tested. Based on this observation, we can enlist the principle objectives of SIM-TD as follows:

a. Improving road safety and efficiency of the current traffic management system.

b. Establish rollout scenarios for functions and applications for scientific questions through field tests and practice oriented experiments.

c. Categorize the functionality of car-to-x communication functions in traffic efficiency, road safety and value added services.

SIM-TD Technology. It integrates the relevant vehicle systems such as data buses to an in-vehicle communication platform, which is known as “ITS Vehicle Station.” This station is responsible for transmitting relevant data to other road users and the traffic infrastructure.

“ITS Vehicle Stations” use wireless technology that is based on the WLAN standard and is specifically developed for automobile field communication. This ensures safe and reliable communication in all conditions including high traffic density. Information can either be transferred directly to other vehicles or to ITS Roadside Stations installed along the road. If the communication partner is not located in close vicinity to the sender, other vehicles can transmit or store and forward information. In addition, mobile wireless technologies such as UMTS and GPRS are integrated to bridge the WLAN connectivity gap (e.g., if roadside infrastructure is lacking). This addition also supports many value-added services.

SIM-TD Test Environment. SIM-TD test environment was divided into three categories: Motorways, Rural Roads, and Urban Roads. In order to develop a transferable scenario on Germany's motorways, average route profiles were selected for the region. These involve the A5 motorway between the Bad Nauheim intersection and Westkreuz Frankfurt as well as several motorways surrounding the city of Frankfurt. The region was divided into two parts with different densities of ITS Roadside Stations. The research focus for the motorway scenario – apart from monitoring the traffic situation and identifying traffic events – is on the traffic forecast. Research will also work on a construction site information system and navigation as well as on end of traffic jam alerts, traffic sign assistance and traffic information, and route deviation management. The heavily used federal rural roads B3 and B455, including main through-roads and their connections to the A5 motorway, are also part of the test region. It is planned to equip a high number of traffic lights with ITS Roadside Stations as well as some more densely equipped road sections. This allows special incidents to be addressed.

The city test field is comprised of an important section of the main roads through the city of Frankfurt. This includes all relevant traffic generators, such as the Frankfurt Trade Fair, the main station, the Commerzbank arena, the Bürostadt Niederrad, the train station, and Frankfurt Airport. The selected city roads are directly connected to the subordinate road network. The company plans are to place one ITS Roadside Station at each major signalized intersection, mostly at distances of less than 500m. The focus, in addition to the general traffic situation survey, is on traffic lights' network control, local traffic-actuated traffic lights' control, location information services, as well as testing of light phases and intersection /cross-traffic assistant systems.

Innovative Technology by Ford Motors [14]. Ford Motors has a research wing to develop some innovative technologies that are designated to reduce drivers' stress by providing assistance in avoiding accidents, parking vehicles, traffic flow information, etc. The main objective of their research is to facilitate the driver and improve driving comfort. We summarize their work below.

In Dec. 2013, the Ford Motor Co. demonstrated [14] a test car equipped with its obstacle avoidance technology that automatically steers and brakes to direct the vehicle away from traffic if the driver fails to steer or brake when the system issues warnings. The system uses three radars, ultrasonic sensors, and a camera to scan the road as far as 656 feet ahead. If the system detects a slow-moving or stationary object, it initially displays a visual warning then an audible warning. If the driver does not steer or brake, the system applies the brakes, scans for gaps on either side of the hazard, and takes control of the electronic power steering to avoid a collision. The technology has been tested at speeds greater than 38 mph.

Perpendicular parking. Ford has already deployed parallel park assist, a popular feature that allows drivers to parallel park the vehicle without touching the steering wheel. The company is doing further research to improve this feature to provide a perpendicular parking assist feature that will allow drivers to park the vehicle any way they require. This feature will use the existing ultrasonic sensors that were deployed for parallel park assist. These sensors will identify a suitable parking space widthwise rather than lengthwise and then use the Electric Power Assisted Steer (EPAS) technology to steer the vehicle in the gap.

Traffic Jam Assist. Ford is also developing an advanced intelligent driving technology: "Traffic Jam Assist." This feature will use the radar and camera technology to help a vehicle keep pace with other vehicles in traffic and provide automated steering control to stay in its current lane. This will reduce driver stress and will potentially improve vehicle flow. Traffic Jam Assist can allow a vehicle to follow the speed of the vehicle in front while being in the lane it is in without any manual intervention. This will help make traveling through congestion a more relaxing experience and, because all the vehicles can travel at the same pace, it will potentially help relieve congestion on the road.

The initial simulation study of Ford Motor Co. claims that if 25 percent of vehicles on a stretch of road are equipped to automatically follow the traffic ahead, journey times can be reduced by 37.5 percent and delays reduced by 20 percent, saving millions of gallons of fuel each year. The Traffic Jam Assist feature has some limitations. It requires an environment where there are no pedestrians, cyclists, or animals, and where lanes are clearly marked.

Innovative technology by Mercedes. The Mercedes Corporation is working on some innovative technologies for future cars. In CES, Las Vegas Mercedes presented an idea of an

autonomous luxury car “0F15” [32]. A research car Mercedes Benz S 500 [33] has completed a test of autonomous long drives of approximately 100 kilometers [33]. Mercedes claims that autonomous car technology is not far from what they already introduced in current Mercedes Benz S class cars. Mercedes is working on providing the following features in the very near future:

Parking Pilot [3]. Using this feature, drivers can get out of the car and send the car autonomously to the parking lot using a mobile app on their smartphones or watches. This feature require a parking lot that supports and is capable of communicating with the car’s 360 degree sensor system.

Highway Pilot [3]. This feature will take control of the vehicle on highways. Using the 360 degree sensor system, it will observe and sense the driving conditions, other users on the road and those surrounding the vehicle. Using the information gathered or based on instructions from the traffic control centers, the highway pilot can navigate through traffic and adjust the speed. This feature can support the drivers in stressful highway driving situations, especially in stop-and-go traffic.

Some of the models of Mercedes’ cars are already equipped with the following features:

Collision prevention system [3][31]. A radar based system that senses the driving conditions and informs the drivers if any risk of collision occurs. An advanced version of this system “Collision Prevention System Plus” is capable of applying breaks automatically if the driver fails to react to collision warnings.

Distronic [31]. This is a driver assist feature that helps drivers to drive the cars semi-automatically. With this feature, a car can maintain a safe distance from the vehicles ahead of it. This feature also provides steering assistance using the radar sensors and multipurpose stereo camera. By using this feature, the car can autonomously follow the traffic while maintaining its lane up to a speed of 200 km/h. This feature doesn't require clear lane marking if the speed is less than 130km/h. [31].

Parktronic [3]. This feature allows the vehicle to park autonomously in either parallel or end-on space situations.

The Safety pilot program of USDOT [39]. The safety pilot program project is initiated by USDOT [39] to demonstrate the benefits of V2V communication in terms of safety and other non-safety applications [41]. The University of Michigan Transportation Research institute had been awarded a contract to model the deployment of a safety pilot program [41]. The initial results for the safety pilot driver clinic are available on the USDOT RITA site [41]. It indicates that more than 90% of the participants like the V2V safety feature in their vehicles and the majority of them are willing to pay for it. The safety features that were tested were the emergency brake-light warning, forward-collision warning, intersection movement assist, blind-spot and lane-change warning, do-not-pass warning, and left-turn assist.

Autonomous Vehicle by Google [34]. Google Inc. is developing autonomous cars [34] that are capable of driving themselves in all traffic conditions including city roads and highways (Figure 1). Their experimental cars have driven 30,000 accident-free miles [19]. These cars are equipped with various sensors and cameras to provide a perception of the

surrounding environment [29]. California, Nevada, Texas, and Florida have issued licenses to driverless cars [19][34].

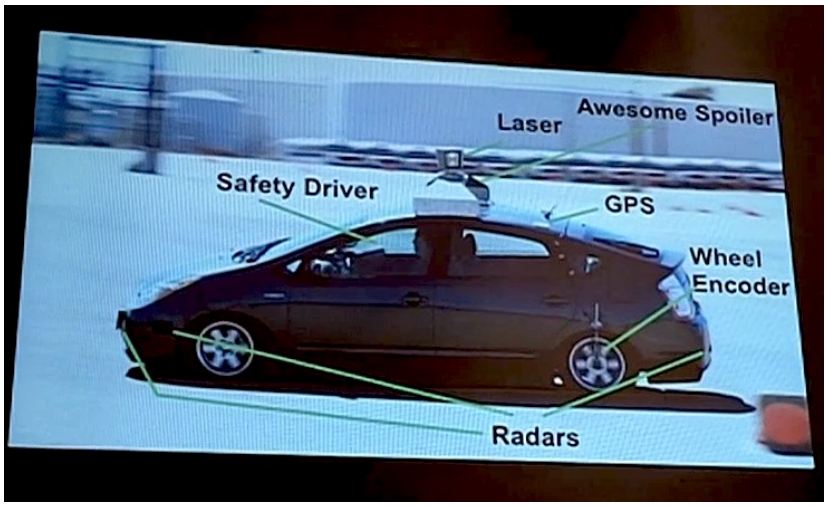


Figure 1 Google Car [20]

To drive on roads, the Google car relies on detailed maps of roads. Moreover, before testing the car on a particular route, a Google engineer drives along the route once or multiple times to gather data for test environments such as poles, trees, and other static structures etc., so that the vehicle can distinguish pedestrians from objects. As mentioned earlier, the overall objective of all this work is to improve the comfort level of drivers on the road. To the best of our knowledge, none of this research is working on V2V communications for synchronization at intersections.

The main part of this system is a laser range finder that is mounted on top of a car. The device, a Velodyne 64-beam laser, generates a detailed 3D map of the car's surroundings. These measurements are then combined with a high-resolution map of the world that produces

different data models, which helps the car to drive autonomously while following all the traffic rules and also avoiding the obstacles. There are some other sensors that are also installed on the car including the following: 1) four radars that are mounted on the front and rear bumpers that help the autonomous system of the car to scan ahead, enough to be able to deal with fast traffic on highways, 2) a camera, which is installed near the rear-view mirror that interprets traffic lights and signals, 3) a GPS, 4) an inertial measurement unit, and 5) a wheel encoder, that determines the location of the car and keeps track of its movements.

Innovative Technology by Other Car Manufacturers. The advancements in V2V communication and sensor technology have motivated every automobile manufacturer to include as many features as possible to improve the safety and comfort of drivers. Although the technology behind these features is different and certainly proprietary, the majority of features have the same template of "safety and comfort." We discuss some of the more important ones.

2.2. Works on Real-time Scheduling

The works and development that we have discussed focus mainly on comfort and safety issues. These issues are important, however, underlying these investigations and development, there lies a fundamental problem of synchronization. Every vehicle (manual or automatic) has to deal with mutual exclusion issues at a road intersection. We observed that autonomous vehicles would not be fully autonomous unless they resolve the issue of mutual exclusion at intersections without the assistance of a third party. In this dissertation, we have investigated this issue and developed an innovative scheme that achieves mutual exclusion at intersections simultaneously. This issue has not been addressed in the literature. There are

many works on V2V or V2I communications, autonomous driverless vehicles, driver's assistance technology, etc., but to the best of our knowledge, we did not find any work that deals with traffic management at intersections using V2V communications and synchronization, even though it is one of the core issues of traffic management.

We identify the self-synchronization problem as a real-time scheduling issue where vehicles represent processes that synchronize themselves using contextual information they continuously exchange. We review real-time scheduling works that relate to our work. We would like to emphasize that these works mainly deal with process and production job shop scheduling. We review a set of works related to the problem of this dissertation.

The work in [2] reviews a number of commonly used techniques in job shop scheduling. It provides insight into techniques in order to find near optimal solutions for different types of job shop scheduling problems. It discusses various models of job shop scheduling and steers us in the right direction for developing a solution for self-synchronization at multiple intersections. Some of the main techniques discussed in this paper are Mathematical programming, Dispatching rules, Artificial Intelligence Expert Systems, Neural networks, Neighborhood Searching and Genetic algorithms. Some of the techniques discussed in [2] are reviewed below.

Mathematical Programming. In this approach, the problem is formulated using integers, mixed integers, or dynamic programming techniques. Because the job shop scheduling problems are NP complete, these techniques have limited use and will remain so until further advancements in computing power and new research will decompose the problem into smaller sub-problems and new techniques are developed.

Several decomposition techniques were developed to divide the problem into multiple levels of scheduling or multiple layers of decision making [2]. The other popular technique in mathematical programming is branch and bound. It is indicated in [2] that "The basic idea of branching is to conceptualize the problem as a decision tree. Each decision choice point - a node - corresponds to a partial solution. From each node, there grows a number of new branches, one for each possible decision. This branching process continues until leaf nodes which cannot branch any further, are reached. These leaf nodes are solutions to the scheduling problem". Branch and bound is an effective technique, but it still requires intensive computations if the scheduling problem is large. A new technique, Langrage relaxation, is used to eliminate the problematic integer constraint by adding penalties as additional costs to the objective function in place of constraints. This technique is used for a while, but like branch and bound, this technique also requires extensive computation for large scheduling problems.

Dispatching rules. These are procedures used to dispatch jobs based on specific criteria. The dispatching rules are synonymously used with sequencing rules, scheduling rules or heuristics and are classified into several classes [2]. Class 1 of the scheduling rules depends on some basic information of the jobs such as processing time, arrival time, etc. These classes are named as Shortest Processing Time First [SPT], First Come First Serve [FCFS], etc. All these types of scheduling techniques are discussed in [17], which is reviewed later in this chapter. Class 2 of the dispatching rules is a combination of rules in class 1. Any particular rule can be applied based on the current situation in the job queue, e.g., if the waiting time exceeds the threshold use, FCFS otherwise SPT. Class 3 of the dispatching rules depends on

the multiple properties of the job. These class of rules are applied based on a cost or weight function that determines the dispatching rule.

Artificial Intelligence. Expert systems, knowledge based systems, and many search techniques are known as artificial intelligence systems. According to the review [2], these systems have four advantages. “First, and perhaps most important, they use both quantitative and qualitative knowledge in the decision-making process. Second, they are capable of generating heuristics that are significantly more complex than the simple dispatching rules described above. Third, the selection of the best heuristic can be based on information about the entire job shop including the current jobs, expected new jobs, and the current status of resources, material transporters, inventory, and personnel. Fourth, they capture complex relationships in elegant new data structures and contain special techniques for powerful manipulation of the information in these data structures.” Although these expert systems are very complex to design and develop and may require very high computational power, due to advancements in computational technology, these systems have become more feasible. The expert systems are very closely tied up with the specific environment or problem, and therefore, there is no generic expert system for all types of job shop problems.

Neural Networks. Neural network systems were designed to incorporate techniques that are similar to human abilities of learning and making predictions based on prior experiences. Neural networks are also known as distributed parallel processing systems. A neural network model consists of learning rules, network topology, and the characteristic of its nodes. In a neural network model, historical data are used for learning and are based upon the difference between desired output and actual output where a new solution is generated that

makes the new output more close to the desired output. This same process is applied iteratively until the difference between the desired output and actual output falls under a tolerance limit. Two of the main techniques used in neural network models are “Relaxation” and “Temporal Reinforcement Learning.”

Neighborhood Search. This is a very popular technique in finding solutions for job shop scheduling problems. Neighborhood search allows enhancing the solution found through any heuristic. The initially developed methods for neighborhood search used the iterative method to tweak (make small alterations to) the solution obtained through any heuristic. This iterative method continues to find the alternative solution and keeps evaluating the resulting schedule until there is no improvement in the value of the objective function. The major neighborhood search techniques are “Tabu Search” and “Simulated Annealing” [2]. The tabu search technique keeps a list of moves that are not feasible or forbidden while making the neighborhood search, known as the tabu list. The tabu list may expand based on previous experiences. On the other hand, the simulated annealing is based on the analogy of the physical characteristics of metal cooling and recrystallization. According to [2], “Using this analogy, the technique randomly generates new schedules by sampling the probability distribution of the system.”

Genetic Algorithm. This technique is based on Darwinian natural selection and mutations in biological reproduction. According to [2], a genetic algorithm performs a multi-process search through concept space. Each process tries to improve the solution. Then each solution and recombination of these concepts is evaluated with a function that determines whether the proposed solution can be considered for further improvement or should be

discarded. This is analogous to survival of the fittest. Following are the components of genetic algorithms [2]:

a. A method of solution representation (encoded in string format or some other simple format).

b. A function that evaluates and can rate the solutions.

c. A method to populate initial solutions.

d. Operators to relate parents that generate crossover solutions and other operators related to specific domains.

e. Setting of parameters, operators, and others.

According to [2], each of the techniques described above can be used to generate solutions in some specific environment. Each has some advantages and disadvantages. In genetic algorithms, if the initial solution is selected randomly, then it is not as effective as the annealing method, but if the initial solution is generated using heuristics, then it becomes more effective than any other method. Especially if the genetic algorithms are combined with other searching techniques, it provides better results.

The work reported in [6] is about a Multiprogramming Scheduling algorithm. The main objective of this paper is to study two different scheduling algorithms for a real time multiprogramming system where each program requires guaranteed services. It investigates (a) fixed priority system and (b) dynamic priority system where priorities are decided on the basis of the deadline of a job. Both of these systems are preemptive, i.e., processing of a job can be terminated if a higher priority job request comes in for processing. According to this

paper, the fixed priority system can achieve 70% of CPU capability, whereas the dynamic priority based system can achieve 100% of the CPU capability.

The work reported in [17] analyzes the performance of various active queue management system algorithms that are used in a TCP/IP networking environment by a router for its queue buffer management. The algorithms analyzed in this paper are FRED, BLUE, SFB and CHOKE. To develop a solution for the Self-Synchronization approach for multiple junctions, we must understand the queue management thoroughly. This work helps us understand the basics of queue management.

In a TCP/IP environment, the outgoing bandwidth, as well as the queue buffer space for incoming packets, is limited at the router. The active queue management algorithm is required when there are too many packets coming to the router and content for the limited queue buffer space and outgoing bandwidth. If the incoming packet rate is high, then a congestion situation may occur. In a congestion situation, packets in the network may be delayed or even dropped. Therefore, the congestion situation is the main cause behind the decreased efficiency of a network.

To avoid or control the congestion situation in a network, there are many congestion control methods used. These congestion control methods mainly use a feedback technique to recognize the congestion situation. There are two types of feedback systems: implicit and explicit. In explicit feedback systems, a special packet is sent to indicate congestion in a network; whereas in implicit feedback systems, it recognizes the congestion based on various factors of the network, such as delay, throughput, and packet loss. To detect and control congestion efficiently, researchers and IETF proposed an active queue management

mechanism. Further, they recommended to deploy the AQM at the router level to improve network performance.

The AQM runs on routers and detects or predicts a probable congestion by analyzing the current and average size for the queue. If the queue size exceeds a threshold value, it notifies the end system by either dropping some packets or by setting some special bits in packets. When the end system receives the packet with the special bit set, it sends an ACK that has a special bit set. When the sender receives this special ACK it reduces the packet rate to avoid congestion. We summarize various AQM algorithms analyzed in [17].

RED (Random Early Drop). The RED scheme was designed to improve network performance. The idea behind RED is to detect the congestion situation early and inform the end system before congestion occurs so that they can reduce the packet rate. To perform, this RED keeps track of the Exponentially-Weighted Moving Average (EWMA) for the queue length. When the EWMA crosses a minimum threshold (Min_{tr}), it drops random packets or marks them with an Explicit Congestion Notification bit. When the EWMA crosses a maximum threshold value (Max_{tr}), it drops or marks all the packets.

According to the paper, the RED has some limitations. It only keeps track of the average queue size that cannot predict the severity of congestion or the source of congestion. It does not keep track of packets from each flow so it requires the proper setting of many parameters and a very large queue buffer size for good performance.

FRED (Flow Random Early Drop). FRED is an extension to the RED technique. It keeps track of packets from each active flow. It makes a decision of dropping or marking packets based on the bandwidth to be used and the packet rate of each active flow. It introduces

two parameters, Max_q and Min_q that are, the maximum and minimum number of packets a flow can use in a buffer. FRED has the following features:

a. Penalizes non- adaptive flows when they exceed the maximum number of packets allowed to buffer.

b. Protects the fragile flow by deterministically accepting packets from low bandwidth connections.

c. Implements fairness in sharing of a buffer for large flows.

d. Improving on RED by keeping track of the average queue length at arrival as well as departure.

BLUE. This method uses the history of packet drops and link usage in place of queue size. It keeps a packet drop probability parameter Pm, that determines the packet drop (or marking) rate. It keeps resetting the Pm based on the history or packet drop due to the queue overflow or link usage. When Pm is high, it drops (or marks) packets more frequently. Because BLUE uses the active queue size as well as the current link usage parameters, it can efficiently learn the correct packet drop rate or marking rate.

SFB (Stochastic Fair BLUE). The SFB scheme is based on the BLUE technique. It protects the TCP flow against the non-responsive flows. The SFB maintains the accounting bins. It keeps the L levels with N bins at each level. It also keeps the L different hashing functions, each related to a level. The hash functions assign each flow to one bin in their respective levels. If the number of packets in a bin exceeds the threshold, the Pm for that bin is increased and if the number of packets drops to zero in a bin then the Pm for that bin decreases. A non-responsive flow will drive the Pm to 1 in each bin it has hashed into. The

responsive flow may share some of the bin with non-responsive flows, but if the number of non-responsive flows is not very high, then a responsive flow will hash into at least one bin that is not shared with non-responsive flows. The Pm for a flow is decided by the lowest Pm of all the bins that it is mapped into. This way, the algorithm protects the responsive flows and punishes the non-responsive flows.

CHOKE (CHOose and Keep for Responsive Flow, CHOose and Kill for Unresponsive Flow). The CHOKe algorithm also uses the average length for the FIFO buffer. It penalizes the non-responsive flows using buffer occupancy of each flow. It has two threshold values for the FIFO buffer: Max_{th} and Min_{th}. If the buffer length is less than Min_{th}, then all the arriving packets are buffered. When the length of the queue is greater than Max_{th}, then all the arriving packets are dropped. When the length of queue is in-between Min_{th} and Max_{th}, for every arriving packet it selects a random packet from the FIFO buffer called a drop candidate. If the flow id for a drop candidate is the same as the new packet, both the packets are dropped. If the flow ids are different for them, then the new packet is dropped based on Pm.

The work reported in [5] proposes a fuzzy logic based approach for traffic control to improve the capacity of signals and reduce the delay time. Our goal in the development of self-synchronization for multiple intersections is very similar to this work except that the self-synchronization approach eliminates the requirement of an outside agent for traffic management. We can understand the basic idea for throughput improvement on an intersection using this work. This fuzzy logic system involves two major components on each

intersection: a) vision sensors for image processing and b) intelligent fuzzy logic based MATLAB boxes.

In self synchronization we propose to control the traffic at intersections with a novel approach of synchronization among vehicles without the help of an outside controlling agent. But, in the current scenario, traffic at intersections are controlled using traffic signals. The traditional method of traffic control at a four way intersection using traffic lights is to use a preset timer for traffic signal lights from each direction. In this approach each side of signals turns green for a stipulated time period in round robin fashion.

The image processing logic recognizes the vehicle irrespective of its color and shape with the help of colored vision sensor. The result is then sent to the fuzzy logic based MATLAB box. Based on the result of intelligent logic, it controls the signals at an intersection. There are several rules defined for the duration of the green signal on each side. The real time image processing counts the vehicles on each side of the intersection and based on these numbers the controllers calculate the duration of the green signal for each direction. According to results presented in this paper, this fuzzy logic approach reduces the delay at intersections by around 27%.

The work reported in [43] provides a hybrid solution to non-preemptive open shops scheduling problem that have sequence-dependent setup costs involved. This paper introduces two new hybrid annealing techniques for solution generation: hybrid simulated annealing (HAS) and multi-neighborhood search simulated annealing (MNSSA).

The work reported in [8] discusses a practical approach for job shop scheduling problems where production jobs falls in low volume/ high verity and mid volume/high verity

categories. According to this work the scheduling of a job may have a span of days to years based on number of operations required and verity of machines needed for a job. Therefore, optimal scheduling methods become impractical due to the requirement of expensive and time consuming iterations. This work concentrates on providing near optimal solution within a reasonable time frame. It provides an innovative solution to classic job shop scheduling problems. The solution presented in this paper is based on Langrage relaxation (LR) and Augmented Langrage relaxation. In the LR approach, it relaxes the precedence constraints and capacity constraints using nonnegative langrage multipliers. In this work, the job shop problem is decomposed to job level minimization sub problems and then to operation level sub-problems. The operation level sub problem is then solved for every machine type iteratively. Then, beginning time and machine type related to lowest cost is used to update the langrage multipliers. In this approach because oscillation between small and large beginning time causes the oscillation in multipliers values, a quadratic penalty term has been added to regulate the solution.

The augmented LR approach used in this work adds a constraint related penalty term to LR objective function. The work in [8] claims that the approach of adding constraint related penalty terms to LR objective function has not been used in Job Shop Scheduling problems before. In this work it uses Gauss-Seidel iterative approach to overcome the interdependency of two successive operations. In this approach, if successive Gauss-Seidel iteration yields to the same solution then it is said to be converged. But if successive iteration are not converging to a solution after a stipulated number of iterations then the solution relative to lowest cost is used. In this work it further enhances the langrage multipliers for generating feasible solutions.

The work reported in [7] provides a genetic solution for open shop scheduling problems with separate setup, removal and processing time. The major difference in job shop scheduling problem and open shop scheduling problem is that there is no restriction on sequence of processing of operations for a job. Any operation can be performed in any order. Though this solution is not directly related to job shop scheduling, it provides a good insight into use of genetic algorithms that help us develop a solution for the self-synchronization approach.

The work in [7] uses a hybrid algorithm that involves Genetic Algorithms (GA), Simulated Annealing (SA) and Tabu Search (TS) techniques to find the optimal solution. The optimization function for this work is minimization of total tardiness. The process of finding a solution for the given open shop scheduling problems is done through four steps. In the first step it finds a random initial solution. In second step it finds the local optima using SA, TS, or uses GA to enhance solutions for the entire problem set. In the third step, it applies selection, mutation and crossover techniques to generate the next generation of solutions for the current set of jobs and operations. The step 2 and 3 are repeated for the stipulated maximum number of times and the lowest value of optimization function is selected as a solution.

The work in [7] states that it is very important to properly represent the candidates of the solution to the problem. It is basic for applying genetic algorithms to find solutions for the actual problem. In this work they provided an example of 3 machines, 3 job problems and represented each machine job combination with a unique number from 1 to 9. So a chromosome presented in [7] "[6-3-1-4-9-7-2-8-5]" can be decoded in a sequence of job

scheduling. The experiment performed in [7] shows that the Distributed Genetic Algorithms provides the best quality and robust solution in hybrid genetic based heuristics.

As mentioned in [2] there are hundreds of papers published on job shop scheduling problems (JSSP) and they are still being written. Each of the works focus on providing a near optimal solution to this NP-Hard issue. In this review section we have reviewed a few of these papers that helped us in understanding the JSSP and a few common approaches to obtain solutions for it. The self-synchronization of moving vehicles at multiple intersections is a variation of JSSP but it is not like a typical JSSP problem. It is a real-time job shop problem. In this problem though, there is no deadline or delivery date. Fairness has a high weight and long wait time for any vehicle at any intersection has a high penalty.

2.3. Works on Networking Protocol in Vehicular Environment

V2V communication basics. V2V and V2I networking is one of the fastest growing research fields. The main objective of these researches is to provide safety for vehicles and people travelling on road. According to the Association of Safe International Road Travel, more than 37,000 people die every year in road accidents only in America. An additional 2.3 million are injured each year. All the government agencies making efforts to provide safe road travel. To make road safety a major priority, the US government (FCC) has provided dedicated spectrum for Short Range Communication (DSRC), which is specifically to be used for road safety and other utility application that involves short range V2V or V2I communication. This spectrum has band width of 75 Mhz on band 5.9Ghz (5.850 Ghz – 5.925 Ghz). Other countries have also provided a dedicated spectrum for this purpose.

Table 1 displays the list of countries and allocated bandwidth for Vehicle oriented communication

Table 1

Country wise bandwidth listing for DSRC/WAVE protocol

Region	Bandwidth/ band
North America [13]	75MHz 5.850 – 5.925 GHz
Europe [12]	70MHz 5.855 – 5.925 GHz
Japan [24]	80MHz 5.770 – 5.850 GHZ
ITU ISM Band [23]	150MHz 5.725 – 5.875 GHZ

IEEE 802.11p DSRC [21] standards and IEEE 1609.4 WAVE [22] (Wireless Access in Vehicular Environment) standards are defined to provide common ground for ITS (Intelligent Transportation System) applications. A description of DSRC/Wave protocol is defined later in this paper.

The DSRC/Wave Standards are developed for short range communication. At the same time in any vehicular application, time is a vital component. Due to the high speed of moving vehicles, the application environment is very dynamic. If the information is not delivered in a specific time limit (Real Time), it becomes invalid and not useful. Due to the limitation of range and time constraints, the majority of the applications that are developed or are developing using the DSRC/WAVE protocol concentrate on spreading the alert in a very

short range, making the decision based on the present vehicular environment in close proximity, e.g. slow down if there is an accident ahead.

The research shows that the packet delivery ratio of DSRC/WAVE protocol drops drastically when the number of vehicles increases. This performance is not acceptable for the safety standard applications for vehicles [30].

Overview of DSRC/WAVE [44]. The DSRC 802.11p protocol standard is based on basic 802.11 WIFI protocol standards. The WIFI standard would not have been sufficient for the vehicular communication environment due to the following:

a. The vehicles move at a very high speed that causes the vehicle network topology to change rapidly.

b. The packet latency plays a strong role in the vehicle safety application. If a hazard alert message is delayed by a fraction of time, it may not be useful to vehicles moving in that direction.

c. The wireless environment is a very unreliable communication medium. The packet delivery ratio may drop significantly if traffic increases in the network, but for safety application reliable packet delivery it is most important. An application can't provide a safe passage if the communication network doesn't provide the appropriate packet delivery ratio.

Considering the above points, the DSRC/WAVE standards are optimized for the vehicular environment. DSRC 802.11p amendment provides guidelines for lower layer protocols (MAC and PHY), whereas the WAVE 1609.4 (Multi Channel Operation) standard provides guidelines for the WAVE protocol stack. Figure 2 shows the basic WAVE protocol stack

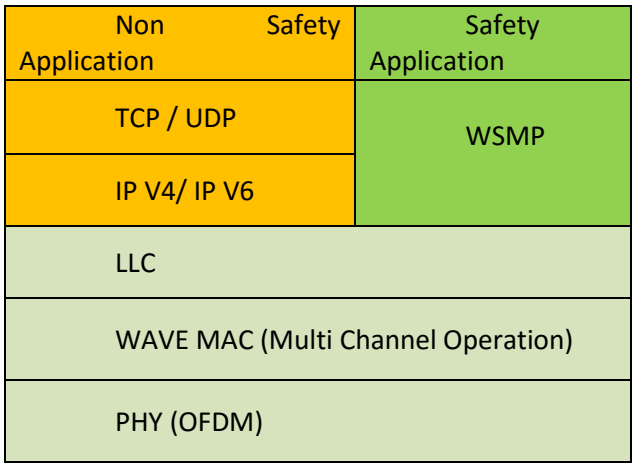


Figure 2 WAVE Protocol Stack

Due to latency and packet delivery ratio constraints in safety applications, it is required that the packet sizes are as small as possible. Therefore, the IPV4/IPV6 protocols are not very useful in these applications due to large header size (The normal IPV6/UDP header size is 52 Bytes). The new protocol developed to reduce this overhead is named Wave Short Message Protocol (WSMP) in IEEE 1609.3. Figure 3 displays the packet structure of WSMP. It only requires 11 bytes for the header. The WSMP protocol also allows controlling of lower layer parameters such as transmission channel, transmission power, bit rate, receiver MAC etc.

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	4 Bytes	2 Bytes	Variable
Version	Security Type	Transmission Channel	Transmission Data Rate	Transmission Power	PSID	Packet Length	Data

Figure 3 WAVE Protocol Header

Use of each field is as follows.

a. Version: To identify whether the protocol is supported on received device.

b. Security Type: to provide encryption information (signed, encrypted, not encrypted etc.)

c. Transmission Channel, Data Rate, Power: To control the radio channel.

d. PSID (Provider Service ID): To Identify the communication circle similar to port number on UDP/TCP.

e. Length: Describe the length of data carried in WSM packet.

The MAC layer in the WAVE protocol stack supports three types of communication: V2V (vehicle to vehicle), V2I (Vehicle to Infrastructure) and I2V (Infrastructure to Vehicle). In IEEE802.11 architecture two independent entities can communicate with each other only if they are members of same service set. Conversely, WAVE applications forming a service may not be feasible due to the time required to form a service set (Frequency & Time synchronization, association etc.). Therefore, in WAVE environment, a new mode is introduced i.e. "Outside the Context of BSS (OCB)." In OCB mode, two entities can directly communicate with each other if they are within the range of a radio link. However, in this mode there are no security services provided by MAC layer. The security services are moved to higher layers as defined in IEEE1609.2 standards.

The data transmission in MAC layer is controlled using IEEE802.11 CSMA/CA (Carrier Sense Multiple Access Collision Avoidance) mechanism. The 1609.4 Standard provides an extension to WAVE MAC layer that allows multi-Channel Operation. The multi-channel operation makes uses of FDMA/TDMA (Frequency Division Multiple Access / Time Division Multiple Access) technique. The 75Mhz band is divided into 7 FDMA channels as shown below in Table 2.

Table 2

Frequency Distribution for DSRC/WAVE Protocol

Channel	Usage	Frequency
Guard	Stop Interference	5.850 – 5.855 GHZ
172	SCH (Service Channel) For Critical Life Safety Messages	5.855 – 5.865 GHZ
174	SCH (Service Channel)	5.865 – 5.875 GHZ
176	SCH (Service Channel)	5.875 – 5.885 GHZ
178	CCH (Control Channel)	5.885 – 5.895 GHZ
180	SCH (Service Channel)	5.895 – 5.905 GHZ
182	SCH (Service Channel)	5.905 – 5.915 GHZ
184	SCH (Service Channel) for High Power Transmission Public Safety	5.915 – 5.925 GHZ

Channels 174 and 175 can be combined into a single 20 MHz channel 175. Similarly, channels 180 and 182 can be combined into channel 181. Each FDMA channel is then divided in slots of 100ms where 46ms slots are allotted to each of the CCH transmission and the SCH transmission. A 4ms interval works as a guard interval before each transmission. This arrangement is done to allow single channel radios to access both CCH and SCH Services. A single channel radio can either transmit or receive data on a 10MHz channel, but can't perform both simultaneously. Figure 4 A 100ms TDMA Slotshows the 100ms TDMA slot.

Figure 4 A 100ms TDMA Slot

The PHY layer of WAVE uses IEEE802.11 OFDM technique for transmission. There are some modifications in IEEE802.11P standard as below.

a. The subcarrier spacing is halved (0.15625 MHz).

b. The supported data rate is halved (3, 4, 5, 6, 9, 12, 18, 24 and 27 Mbit/S).

c. The Symbol Interval and Cycle Prefix interval is doubled (8□S and 1.6 □S).

In our research we have developed an application layer protocol which supports long range multi hop communication based on WAVE/DSRC standard. This protocol provides a base for long range transmission of the contextual information that will facilitate resolving complex traffic issues such as collision avoidance on intersections, traffic synchronization, etc. This protocol also ensures on time delivery of contextual information to desired entities. Later in this dissertation we present a unique solution to packet delivery ratio issues for a large number of vehicles.

Existing Work in V2V Communication. As mentioned earlier, Vehicle 2 Vehicle communication is an emerging research field. Much research has been done or is in progress in different parts of the world. ISO-CALM standards that are developed for Intelligent Transportation System (ITS) projects by the ISO (International Standard Organization), provide standards for network technology independent communication. It provides an abstracted layer between the application and communication network.

Safe Intelligent Mobility (SIM-TD) [38] described earlier, which is a German project, is still in progress having objective of traffic information gathering, traffic management (Congestion control by providing alternate route information to vehicles), providing hazard alerts (Accident ahead etc.), providing safety alerts (Signals, stop signs etc.), providing driving assistance, etc. It requires a roadside infrastructure for data gathering and providing information to vehicles.

CHAPTER 3

INTRODUCTION TO SELF-SYNCHRONIZATION

The current traffic management system at intersections normally uses traffic lights. The color of lights indicate to the upcoming traffic what action they should take. There are mainly three colors of traffic signals: Red, Green and Yellow. If the signal is green, traffic can access the intersection. If it is red, then the vehicle must stop for the other sides, but in case of a yellow signal, it is up to the drivers to decide whether to enter the intersection or stop. The decision for stop or go is decided based on the perception of individual drivers such as whether they can safely pass through the intersection before the red light turns on. Then they should access it. Otherwise they should stop. Because of this dilemma, it is very common that some drivers enter the intersection when the signal is yellow, but they couldn't pass the intersection completely before the red light turns on. Therefore, there must be a delay between signaling the red light to one side and opening the green signal to the other side. Our general observation of intersections showed that there is a 3 second delay for this transition at most of the signal managed intersections.

At many intersections, there is no clear green for left turns. The traffic rule says that the vehicles turning left at intersections must yield for traffic going straight in the opposite direction, if the signal is solid green. In this case, it is also up to perception of individual drivers to decide whether they can cross the intersection safely without disturbing traffic from the opposite direction. Vehicle accident statistics show that 22.2% of collisions happen while turning left at the intersection [40]. The statistic also shows that around 29.2% of the accidents

happen due to wrong decisions made by drivers. Moreover, around 96.1% of the accidents at intersections happen due to driver error.

We propose a new traffic management system using self-synchronization which can overcome most of the issues related to current traffic management systems. In our scheme, we propose to eliminate or minimize human discretion parameters. The self-synchronization system provides precise instruction to drivers whether to stop at an intersection or go irrespective of direction they intend to go. The instruction provided to drivers are calculated based on incoming traffic from all directions. The decisions made for every vehicle is passed to all of the concerned vehicles at that intersection so that proper instructions can be provided.

The proposed self-synchronization system doesn't require an outside administrative system. All the vehicles that are concerned at the intersection make the decision by themselves after communicating contextual information with each other. Therefore, it doesn't require any infrastructure support at intersections and can therefore save millions of dollars on energy consumption by traffic signals and other infrastructure equipment.

Using the proposed system, the decision for vehicles to stop and go can be calculated well before the vehicles reach the intersection. The movement of every vehicle will be notified of every other vehicle at that intersection electronically. Therefore, there is no need of any delay in execution of stop or go instructions. For example, if vehicle A has decided to stop for vehicle B, C and D, then it can start moving as soon as the last vehicle among B, C and D passes the intersection.

3.1. Self-Synchronization Basics

The dynamics of moving vehicles through an intersection are quite complex. Figure 5 Dynamics of moving vehicles through an intersection illustrates the movement of vehicles at an intersection with one-lane crossroads marked as road 1, 2, 3 and 4. All possible conflict-generating movement directions of vehicle A from road 1 to other roads are shown. This is replicated at road 2, road 3, and road 4 generating 12 possible conflict situations. The existence of "turn on red" policy and the priority of emergency vehicles such as ambulances, police cars, fire vehicles, and so on, further add to the complexity of conflict-free scheduling. Note that all these parameters are subject to "mobility constraints." That is, the scheduler must take into consideration the velocity of moving vehicles. The task of conflict-free scheduling is to:

a. manage these 12 conflict situations

b. generate conflict-free traffic flow with minimum delay at the entry to each intersection

c. repeat this flow as much as possible to subsequent intersections

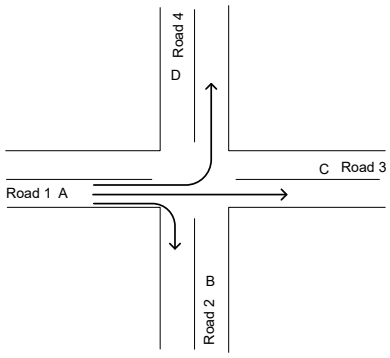


Figure 5 Dynamics of moving vehicles through an intersection

To achieve conflict-free movement of vehicles, it requires the synchronization among them. To achieve synchronization among vehicles, all the vehicles must be able to transmit and receive information from the other vehicles in the vicinity. It raises the requirement of a communication protocol to form a communication network among vehicles. The communication protocol for this scheme has the following properties or limitations:

a. Because there is no centralized controller in this scheme, it requires an ad-hoc communication network for data transmission.

b. Since the vehicles act as the moving nodes in this network that may join or leave the network at any point of time, the communication protocol must be dynamic and self-healing.

c. The communication protocol is responsible for initial handshake among vehicles and once the link is established, it must transmit contextual information to all other vehicles in the vicinity.

d. It should be able to accommodate large number of vehicles while maintaining reliable communication among them in real time.

e. It should be able to transmit the contextual information long range for multiple intersection traffic synchronization.

Existing network protocols such as token-passing, self-healing, etc. are not able to satisfy the above requirements of self-synchronization. It is, therefore, necessary to develop a new protocol that can handle all requirements.

When vehicles receive the contextual information related to all other vehicles in the vicinity, they have to generate a schedule such that they don’t cause any conflict while

vehicles are accessing the intersection. The schedule of vehicles to access the intersection must be generated in real time because any delay in decision making can cause hazardous situations or can create congestion in intersections. The solution must follow some sort of optimization policies so that the utilization of intersection can be maximized without causing long delays to any of the vehicles trying to access the intersection.

3.2. Objective

Our objective is to investigate and develop an efficient scheduling scheme which will make mutual decisions and enforce it to achieve conflict-free traffic flow at intersections. We will investigate the following issues in our investigation and for the development of our scheme referred to in this dissertation as "Efficient scheduling scheme for self-synchronization."

a. Design, develop and use new energy-efficient communication protocols for sharing contextual information in real-time for enforcing mutual exclusion.

b. Eliminate the limitations of existing traffic rules that are enforced by a third party.

c. Develop and incorporate more synchronization criteria to existing traffic light rules in our scheduling scheme.

d. Minimize or eliminate delay (waiting time) at each intersection.

e. Extend one intersection scheduling scheme for multi-intersection to achieve higher throughput.

f. Provide non-stop movement (green channel) of each vehicle through multiple intersections.

g. Maintain fairness while increasing throughput.

3.3. Providing Green Channels for Vehicles

The self-synchronization approach can ensure collision-free movement of vehicles at intersections. But ensuring collision free movement is not enough to resolve current traffic issues. If a vehicle has to stop at every intersection it passes to implement this system then this system is no more than a modified stop sign. In the self-synchronization approach, we also need to ensure the least amount of stop and wait time for each vehicle moving through a grid of intersections. The green channel refers to a stretch of road, which includes a number of intersections, where vehicles can achieve non-stop conflict free movement. To create green channels for every vehicle present in the system, we need to synchronize the pair of vehicles which may conflict not only at the next intersection on its route, but also at all the intersections that are present on its green channel.

3.4. Fairness

Simply stated, fairness to access the intersection means no one has to wait at the intersection for more than the stipulated time limit. We call it threshold time for wait at intersection. The fairness in terms of scheduling is far more complicated than the time limit. Below are the assumptions:

a. Each vehicle that requires passing through the intersection will be treated as a node in the request queue.

b. Each side of an intersection will be treated as a separate request queue. So at a 4 way intersection, there will be 12 request queues.

c. We will define the fairness with respect to requests per minute.

d. An intersection is approximately 25 meters (80 Ft.) long in each direction. Suppose average vehicle speed is 40MPH or 17.78 meters/sec. When a vehicle reaches the intersection and can pass without stopping, it requires approximate 1.5 seconds to pass through the intersection. If the vehicle has to stop at the intersection before accessing it, we can calculate the time required using the acceleration formula "$S=1/2at^2$." According to [18], a passenger vehicle can achieve 1.74 mps^2 of maximum acceleration from rest. Based on this information it requires approximate 5.36 seconds to pass through the intersection of 25 meters.

e. In low traffic scenario, the maximum time limit for continued access to the intersection for each passing group is 15 seconds, when the vehicles in conflicting directions are already waiting.

f. In high traffic scenario, the maximum time limit for continued access to the intersection for each passing group is 45 seconds, when the vehicles in conflicting directions are already waiting.

Below are the different scenarios for traffic flow at the intersections and a fairness definition for it:

a. Low traffic scenario: When traffic-flow at an intersection is very low, for example, when there are less than 10 vehicles joining each queue per minute. In this case, the fairness can be defined as: in the best case, none of the vehicle has to wait at the intersection and in worst case, the wait time for a vehicle at the intersection is not more than 1 minute. Because the traffic flow is very low, the scheduling can be done so that none has to stop at the intersection. There are 12 conflicting sides at a 4 way intersection which can be combined to 4 passing groups (group of non-conflicting directions see Table A1). After vehicles from

group 1 cross the intersection, one vehicle from every direction of the remaining three groups can cross the intersection without stop in 6 seconds. (Assumption 4, one group of non-conflicting directions requires only 1.5 seconds to pass one vehicle from each direction). Since vehicles are arriving at a rate of one vehicle every 6 seconds from one direction, the next vehicle from each of the directions of group 1 can pass the intersection without stopping. Now suppose that one of the vehicles has to stop for the other vehicles to pass from the remaining groups. Because we restrict every passing group to access the intersection continuously for no more than 15 seconds, the stopped vehicle has to wait for a maximum of 45 seconds. (This is because at a 4 way intersection a vehicle can have conflict with at most 6 other sides that form 3 passing groups. Therefore, the maximum time limit for all 6 conflict directions will be 45 second).

b. Medium to High Traffic scenario: When traffic is medium, for example, if there are 11 to 30 vehicles joining the request queues per minute. Now because vehicles are joining every 2 seconds from each side, it is impossible to have a nonstop flow of vehicles. In this scenario the fairness can be defined by average waiting time. In medium to high traffic scenario, there can be even traffic flow from all directions or there can be an uneven traffic flow where some directions are very crowded and others are sparse. If the traffic flow is even, then every side can be open for a stipulated time slot and can cycle through all the conflicting directions one by one. This way everyone has the fair share of the intersection. In the case of uneven traffic flow, there is a lot of traffic from one conflicting direction and only a few from the other. If we use the above technique, the throughput at the intersection will drop significantly and the vehicles from the high traffic flow direction will have to wait longer

unnecessarily. On the other hand if we keep the high traffic direction open for a longer duration, the throughput will increase at the intersection, but vehicles from low traffic directions have to wait for long durations. In this type of scenario the threshold time comes into effect. We can calculate the time for which a direction can remain open based on arrival rate from that direction, but can limit this based on threshold time so that the vehicles from low traffic directions don't have to wait for an unfair amount of time.

c. In the best case scenario, if no vehicle has to stop at an intersection, then the throughput can be 160 vehicles per minute.

It requires 1.5 second for one vehicle from one direction to cross the intersection.

We can group together 4 directions at each of the passing groups.

There are 4 passing groups at a 4 way intersection.

One cycle of accessing the intersection by one vehicle from each direction of each passing group finishes in 6 seconds. Hence 16 vehicles can access the intersection in 6 seconds. Hence in the best case scenario, 160 vehicles can access the intersection per minute.

Our goal is to achieve this throughput rate at an intersection while keeping the average wait time low as well as individual wait time below the threshold time.

3.5. Input and Output of Self-Synchronization

The self-synchronization requires synchronization between every pair of vehicles that are in conflict. For example, consider four direction and from each direction two vehicles trying to cross the junction A1, A2, B1, B2, C1, C2 and D1, D2. Therefore, we need to synchronize the following pairs (Table A1)

(A1, B1), (A1, B2), (A1, C1), (A1, C2), (A1, D1), (A1, D2), (A2, B1), (A2, B2), (A2, C1), (A2, C2), (A2, D1), (A2, D2), (B1, C1), (B1, C2), (B1, D1), (B1, D2), (B2, C1), (B2, C2), (B2, D1), (B2, D2), (C1, D1), (C1, D2), (C2, D1), (C2, D2).

So if there are N vehicles from each direction in conflict at an intersection, we need to $6N^2$ pairs of vehicles. We can derive this count as seen below.

a. N vehicles from direction 1 are in conflict with N vehicles from each of the other three directions which is a total of $3N^2$ number of conflicting pairs of vehicles.

b. N vehicles from direction 2 are in conflict with N vehicles from each of the remaining two directions which is a total of $2N^2$ number of conflicting pairs of vehicles.

c. N vehicles from direction 3 are in conflict with N vehicles from direction 4 which is a total of N^2 number of conflicting pairs of vehicles.

The total of A, B, and C is $6N^2$ number of conflicting pairs of vehicles. The conflict between each pair should be resolved in such a way that it minimizes the wait time of vehicles and maximizes the throughput while satisfying fairness.

Input Parameters. The input parameters to the self-synchronization algorithm are:

a. A predefined Map module which provides longitudes, latitudes of intersections, roads points and other geographical locations. The map module should also provide the max speed limit at any given location. Using this information, a vehicle can identify its current location and can calculate the estimated time it requires to travel a certain distance.

b. Vector of intersection which is predefined in map module.

$(I_1, I_2, I_3, I_4 \ldots\ldots..)$

c. Vector of min time required to travel from one intersection to intersections adjacent to it based on speed limit and distance between two intersections.

(W_{12}, W_{13}, W_{34}…….) where W_{IJ} is min time required to travel from intersection I to intersection J.

d. Predefined conflicting direction matrix for each intersection (Table A1).

e. Vector of vehicles in the vicinity

$$(V_1, V_2, V_3, V_4 \ldots)$$

f. Route of each vehicle or vector of requests for intersection for each vehicle in the order of intersection access along with (a) estimated time interval in which a vehicle is intended to access the intersection, (b) direction it is coming from and (c) direction it is going to. For example, if vehicle V_1 plans to pass through the intersection I_1 (Figure 3) at time T_1 from direction 1 to 2 and then intersection I_2 at time T_2 from direction 1 to 2 and so on the request vector for V_1 would be as below

$$RV_1 = \{(I_1, T_1, 1, 2\}, (I_2, T_2, 1, 2), (I_3, T_3, 1, 2)\}$$

Similarly for V_2

$$RV_2 = \{(I_1, T_1, 4, 2\}, (I_4, T_4, 4, 2), (I_5, T_5, 4, 1)\}$$

Here vehicle V_1 and V_2 both request intersection I_1 at the time T_1 and move in conflicting directions so this may cause a conflicting scenario that require scheduling between vehicle pairs (V_1, V_2).

g. Decision vector (described below) if exists from any of the conflicting vehicles. This will help reducing the decision calculation time.

The output of Self-Synchronization. After gathering the request vector from each of the vehicles in the vicinity and performing the scheduling, the algorithm will provide the following decision vector. This decision vector contains a straight forward command of either WAIT for conflicting vehicles to pass through the requested intersection or PASS through intersection before the conflicting vehicle passes through it. For example, the decision vector for vehicle V_1 is:

$DV_1 = \{(I_1, V_2, STOP, T_1), (I_1, V_3, STOP, T_x), (I_1, V_4, STOP, T_y), (I_1, V_5, STOP, T_z), (I_1, V_6, GO, T_2), (I_1, V_7, GO, T_2), (I_1, V_8, GO, T_2), (I_2, V_9, GO, T_3), (I_2, V_{10}, GO, T_3) \ldots\}$

At the same time, vehicle V2 will have the following node in its decision vector:

$$DV_2= \{(I_1, V_1, GO, T_1), \ldots \}$$

Here each four tuple of decision vectors provides the following information:

a. The intersection for which the decision is made.

b. Conflicting vehicle id.

c. Decision whether to wait (STOP) for other vehicle to cross the intersection or GO before it.

d. Time at which conflicting vehicle will pass the intersection if decision is to STOP for it or time at which the vehicle should pass the intersection if the decision is to GO.

In the above example, vehicle V_1 will wait for vehicle V_2, V_3, V_4 and V_5 to pass the intersection I_1. When all these vehicles pass through intersection I_1, V_1 will pass the intersection at the approximate time T_2 before vehicles V_6, V_7 and V_8. Based on the above decision matrix, each vehicle can generate the request vector for the next processing cycle. Input Parameters that will be used for decision making:

To simplify the representation of the solution and to reduce the amount of data to be transmitted on the communication network we can represent the output of the scheduling algorithm using Scheduling vector SV for each intersection I as shown below:

$$SV_I = \{V_1, V_2 \ldots\ldots\ldots\ldots V_n\} \qquad \text{(Solution 1)}$$

The above scheduling vector is arranged in order of the vehicle accessing the intersection I. Therefore, in the above example, vehicle V_1 will access the intersection first, then V_2, V_3, and so on. This scheduling vector is generated for each of the intersections and will be transmitted to all the concerned vehicles.

Vehicles can populate the decision vector DV based on this schedule using contextual information of other vehicles such as their length, speed limits, and their current location.

3.6. Problem Definition in terms of research issues

3.6.1. Communication Protocol

The first research issue is to develop a mobile communication protocol which will provide uninterrupted and timely exchange of contextual information among vehicles for making a decision. Also, once the decision is calculated, the self-synchronization scheduling algorithm requires a communication protocol for transmission of the final decision to the vehicles. Various government agencies have defined DSRC protocol for communication among moving vehicles. This protocol is a short range communication protocol and has several limitations. In our scenario, the communication range is larger than DSRC protocol can support. Our setup, therefore, requires a protocol which will transmit data beyond the limit of DSRC protocol [30]. We address the following issues in developing a new communication protocol.

a. *Cell Definition*. Define cell boundaries based on hardware capabilities.

b. *Real Time delivery*. The communication protocol must be able to transmit the required information in real time. Therefore, we need to define a threshold time T in which a cycle of communication from all the vehicles in a cell must be completed.

c. *Cell Master*. The moving vehicles form a dynamic network in each cell. Since there is no external static agent that can handle the responsibility of the server, a cell master (leader) must be selected among the existing vehicles in the cell. The cell master (Leader) selection is also a research issue.

d. *Long Range Communication*. Once all the vehicles in one cell finish a cycle of data transmission, the gathered data must be transmitted across the cell boundary. Therefore, an intra-cell communication protocol must be defined.

e. *Message format*. The contextual information [27] that is shared must be small so that all the vehicles can transmit their information in real time. The message should be large enough to include required information for scheduling algorithm.

3.6.2. Self-Synchronization of moving vehicle at one intersection

The second issue of our research is to find a conflict free schedule for vehicles at one intersection. Below are the properties of single intersection issues:

a. An intersection can have n number of roads joining. If n roads are joining at an intersection, it is called as n-way intersection. The most common intersections are 4-way intersections.

b. If we consider a 4-way intersection, it has 12 freedom of movements i.e. 12 directions available based on the incoming road and the outgoing road. From these 12

directions, some can access the intersection together and some cannot based on the conflict matrix of that intersection. One example is shown in Table A1. We can form four direction-groups where vehicles moving in any direction of one direction-group can access the intersection together, but different direction-groups cause conflict. Again, to ease the problem, we assume that at a 4-way intersection there are 4 conflicting directions in place of the 4 conflicting direction-groups.

c. If we consider U-turns as another legal direction of movement, it adds 4 more freedom of movements at the intersection. In this dissertation we do not consider U-Turns as a legal freedom of movement, hence we ignore it.

d. Vehicles moving in one of the direction-groups conflicts with all the vehicles moving in any other direction-group.

e. At a given point of time, the number of vehicles at the intersection are fixed with no additional vehicles joining or leaving. In reality, vehicles join and leave the intersection very frequently. We have to select a recycle time period (RT) for which we assume that there is no change of vehicle inventory at the intersection. The recycle time should be calculated based on the maximum allowed speed in the vicinity of the intersection. The scheduling algorithm must provide a feasible solution within the RT. Otherwise the solution will not be valid for the present scenario at that intersection.

f. Vehicles may or may not be distributed evenly in all the conflicting directions.

g. The sequence of the arrival of the vehicle is important in each individual direction. Vehicles which arrive early should leave before vehicles arrive later from the same direction.

The precedence rule is only applicable for vehicles moving in the same direction, but it is not applicable for vehicles moving in different directions.

h. The length of vehicles may differ. Some vehicles are large like transport trucks. Some have medium length like vans and buses, and some are small like passenger cars. Based on the capability of the vehicle and its length, every vehicle takes a different amount of time to cross the intersection.

i. It requires more time to cross the intersection if the vehicle is in the stopped position just before entering the intersection. The difference is discussed in section 3.4. If vehicles from a direction are waiting to access the intersection, and if the solution allows more than one vehicle to access the intersection, then only the vehicle in the front of the queue will require this extra time. The vehicles behind the first vehicle will already be in motion when they reach the intersection, hence they do not require additional time.

j. Though the general passenger vehicles and transport vehicles do not have any special priority, the vehicles like fire trucks, emergency vehicles, and police vehicles have priority over general traffic.

Formulation of one intersection problem. Based on the above assumption and input/output definition in section 3.5, we define the problem of synchronization of moving vehicles at one intersection as a scheduling problem. The theoretical notification for scheduling problems was introduced in [37] and is denoted by $\alpha \mid \beta \mid \gamma$. Where α represents the machine environment, β represents the job properties and γ represents the optimality criteria. Below are the properties for self-synchronization at one intersection problem:

Machine Environment α. There is one intersection (I1) which can process one vehicle at a time from each of the direction grouped as non-conflicting directions, but vehicles from conflicting directions can't be processed through the intersection together. Therefore $\alpha = 1$.

Job properties β.

a. Each vehicle moving towards the intersection can be represented as an individual Job that needs to be processed through the intersection.

b. Vehicles requires different amounts of time to pass through the intersection, they have arbitrary processing times. We assume that a general passenger car (sedan) takes two units of time to cross the intersection $P_i = 2$ for sedans and the long 18-wheeler truck may take 6 to 8 units of time based on its direction of movement at the intersection. Other vehicles fall in between these two measures. Therefore $2 \leq p_i \leq 8$.

c. Some special vehicles have priority over ordinary vehicles. This can be satisfied by assigning weightage to vehicles. We assume ordinary vehicles have weight 1, whereas special vehicles can be assigned weight > 1. Therefore $w_i \geq 1$. Also, the special vehicle doesn't have to follow the precedence rule, therefore, we assume that these vehicles have special paths to access the intersection before vehicles already arrive at the intersection.

d. Vehicles moving towards the intersection are not always ready to access the intersection when we take the snapshot of the problem instance. The vehicles that are behind in the queue can't be ready to access the intersection unless the vehicle ahead of it has passed through. Some vehicles are far away from the intersection and may require some time before they reach the intersection even though there are no vehicles in front of them. In other words we can say that not all vehicles are ready to process at the time when the snapshot of the

problem is taken. Therefore each vehicle has a release time $r_i \geq 0$ related to it. The vehicles that are first from each direction at the intersection have release time 0, all other vehicles have $r_i > 0$, below is the method to calculate the release time for each vehicle:

$$r_i = \text{Max} \begin{Bmatrix} \sum p_i \forall v_i ahead\ in\ same\ direction, \\ Time\ required\ to\ reach\ at\ intersection \end{Bmatrix}$$

e. Though it is not required for a vehicle to pass through the intersection at a predefined time or before a specified deadline, to measure the optimization statistics we define the due date of vehicle as $d_i = r_i + p_i$ (release time + processing time). Our goal is to provide nonstop movement for all the vehicles, therefore, this deadline fulfills our requirement. It is notable that the due date strongly agrees with the release time.

f. Vehicles moving in conflicting directions toward the intersection that require access to the intersections, one by one, can be represented by conflicting families of jobs that have precedence rules that form a chain. There is no precedence rule across the families of vehicles (vehicles moving in different directions). Therefore, the precedence rule in this problem is a set of *chains* and it forms Conflicting families of jobs.

g. Because the vehicle entering the intersection can't be stopped or taken back half way, it is therefore a non-preemptive scheduling problem.

h. If a vehicle has to stop before entering the intersection, it requires more time than if it accesses it without stopping. This setup time is applicable only to the first vehicle in sequence from that direction. The vehicles that follow the first vehicle, access the intersection while in motion therefore they do not require additional time. This property can be satisfied by introducing batch dependent setup time. We assume that setup time for each of the families

is constant and can be assigned either 2 or 3 units. Notation for sequence dependent setup time for families or batches is defined in [1]. For our problem it can be shown as $ST_{sb,\,d} = 2 \mid 3$.

Optimization Criteria γ. The objective of this problem is to process the maximum number of vehicles in a given time slot. This can be achieved by minimizing the average weighted tardiness, which is defined as below:

$$\overline{WT} = \frac{1}{n}(\sum w_i * \max(d_i - c_i, 0))$$

To maintain the fairness means a vehicle from one direction shouldn't be waiting for too long at the intersection if the other directions are busy, which can be achieved by minimizing the maximum wait time for batches. The wait time for a batch is the delay between the processing of two consecutive vehicles from the same direction if the vehicles are ready to process. If v_i and v_j are two consecutive vehicles from a direction then the wait time is defined as:

$$\text{Wait}_j = \begin{cases} C_j - (C_i + p_j) & if r_j \leq C_i \\ C_j - (r_j + p_j) & Otherwise \end{cases}$$

If we combine the two minimization objectives stated above, we get the following objective function for this problem:

$$\gamma = \min(\overline{WT}, Wait_{max}) \quad (1)$$

Based on the above properties, the self-synchronization of the moving vehicle at one intersection problem is:

$$1 \mid \text{chains}, (r_i\,,\, d_i), \text{conflicting families}, ST_{sd,b} \mid \overline{WT}, Wait_{max}$$

Complexity of the problem. If we assume that $r_i = 0$, $p_i = 2$, $w_i = 1$ and $ST_{ij} = 0$ for all i,j then the problem simplifies as:

$$1|\ \text{chains}, p_i=2\ |\ \overline{T}, Wait_{max} \tag{2}$$

We know that the vehicles that are behind in the queue to access the intersection have larger due dates than the vehicles ahead. That means if $v_i > v_j$ then $d_i < d_j$ which means the due date agrees with the chain precedence rule. According to Lawler [10] if $p_i <= p_j$ and $d_i < d_j$ then $j_i > j_j$ in the optimal solution sequence for the problem. Since we assume $p_i = 2$ for all v_i, the EDD sequence will provide optimal sequencing for both $\overline{T}$ as well as $Wait_{max}$. Also, EDD will take care of the chain precedence as well. Hence the problem defined in (2) has a polynomial solution.

Even if we remove the assumption $r_i = 0$, the EDD will still be the optimal solution for the problem because r_i and d_i strongly agree. Lenstra and Rinnooy Kan [26] have shown that "1| prec, p=1|$\sum T_i$" is strong NP-hard, Leung and Young [25] have shown that even "1| chain, p=1|$\sum T_i$" is strong NP-hard with arbitrary due dates. Since precedence rule and due dates agrees in our problem, it has a polynomial solution with the assumption p=2. But this is as far we can find a polynomial solution. As soon as we remove the assumption p=2 and take arbitrary processing time for vehicles, the problem is defined as 1| chain |$\sum T_i$ which is strong NP-Hard. If we consider that the chain precedence rule is nullified due to its agreeance with the due date, still, Lenstra and Rinnooy Kan [26] have shown that 1| |$\sum T_i$ is ordinary NP-hard. Lawler [10] has developed a pseudo polynomial algorithm of $O(n^4 \sum p_i)$ complexity for it. [26] has shown that 1| rj |$\sum T_i$ is obviously strong NP-Hard.

Pinedo [35] has provided a complexity hierarchy of the deterministic scheduling problem. Using this complexity hierarchy, we can determine if one scheduling problem is reducible to the other. If problem A is reducible to problem B (A ∞ B), then the solution technique used for problem B can also be used to find the solution for problem A. Using the complexity hierarchy, we can derive the following relation:

$1|\, r_j \,|\sum T_i \quad \infty \quad 1|\, (r_j, d_i)\,|\sum T_i \quad \infty \quad 1|\, \text{chains}, (r_j, d_i)\,|\sum T_i \quad \infty \quad 1|\, \text{chains}, (r_j, d_i)\,|\sum w_i T_i$

$1|\, \text{chains}, (r_j, d_i)\,|\sum w_i T_i \quad \infty \quad 1|\, \text{chains}, (r_j, d_i), \text{conflicting families}, ST_{sd,b}\,|\sum w_i T_i \quad (3)$

By relationship (3) it is clear that problem "*1| chains, (r_j, d_i), conflicting families, $ST_{sd,b}$ | $\sum w_i T_i$*" is strong NP-hard. Even though we are adding additional optimization criteria to this problem, it doesn't make the problem any easier. Therefore, the problem of the self-synchronization of moving vehicles at one intersection is strong NP-Hard.

3.6.3. Self-Synchronization Of Moving Vehicles Through Multiple Intersections (Green Channel)

The third research issue in this dissertation is to develop a schedule for vehicles that ensures a wait-free passage to each vehicle at least for 3 consecutive intersections on its route. Below are the properties of this problem.

a. This problem involves more than one intersection, but every intersection inherits all the properties that are defined for an intersection as in the second issue earlier in this section. Though all the intersections have similar properties, every intersection may vary in terms of its property values such as inner length of the intersection, number of roads joining at the intersection, speed limit at the intersection, etc.

b. The rules for vehicle movement at each intersection remain the same for this problem.

c. We consider that vehicles must have not-stop movement for at least three consecutive intersections on their route. Therefore, we consider a grid of 3 X 3 intersections as the problem base in this issue.

d. Vehicles can travel from one intersection to the other using connecting roads. Two intersections are adjacent if they are directly connected through a connecting road. If there is no direct connecting road between two intersections X and Y, then to reach Y from X, vehicles must go through at least one additional intersections between X and Y. An n-way intersection has n adjacent intersections.

e. The route for every vehicle is predefined and fixed before entering the intersection grid area. The route of a vehicle defines the order in which it will access the intersections. Two consecutive intersections in the route of a vehicle must be adjacent intersections.

f. The time required to cross the intersection (processing time) for a vehicle may be different for different intersections on its route, which is based on the speed limit and the length of the particular intersection. The processing time can be calculated while constructing the route for a vehicle. Therefore, it is predefined and fixed.

g. For this problem we need to make sure that the maximum number of vehicles accessing the three consecutive intersections are without any wait. At the same time, we need to make sure that the average wait time at the intersections by the vehicles is minimized.

Formulation for the self-synchronization of vehicles for 3 intersection problem.

Machine Environment α. We mentioned that we are considering a 3 X 3 grid of intersections, therefore there are 9 intersections involved, though all intersections have the same properties, but every intersection can be different in terms of processing time due to its inner length. Therefore, there are 9 different processing units (machines) involved in this problem. The machines are arranged neither parallel nor in series. Any vehicle can select its route independently, therefore the machine setup is a Job Shop setup. Hence $\alpha = Jm$ where $m=9$ so $\alpha = J9$.

Also, each machine can simultaneously process the jobs from non-conflicting families and processing time for a batch is dependent on the cumulative processing time of jobs in it. Therefore, it is an S-batch processing job shop system.

Job properties β.

a. Jobs: There can be n vehicles involved at a given point in time. We represent each vehicle represent a job j.

b. Processing time: processing time for job j at an intersection i will be represented as p_{ij} and is arbitrary. It is calculated based on the vehicle length, vehicle capability, inner length of the intersection and the speed limit at the intersection.

c. The sequence of operations: each intersection a vehicle has to access is an operation for the vehicle. A vehicle may request to access any number of intersections in any given arbitrary sequence. The route for vehicle j will be represented by R_j, and is a set of the intersection, arranged in the order of the vehicle movement. The two consecutive intersections in route R_j must be adjacent intersections.

d. Family of jobs: Every direction of movement at an intersection forms a family of jobs. The conflicting direction chart (Table A1) defines compatibility of job families at intersection. An intersection can simultaneously process jobs from compatible families.

e. The job precedence is defined: Vehicles moving in the same direction (jobs in the same job family) have a precedence of first come first serve (FCFS) but vehicles moving in conflicting directions (Jobs from incompatible Job families) have no precedence.

f. Transfer delay: every vehicle needs some time to travel to the next intersection on its route after it accesses one of the intersections. This transfer time can be predefined based on the distance between intersections and the maximum speed limit.

g. Sequence setup time: if vehicles access the intersection with zero wait time (no stop), then the setup cost is zero. Conversely, if a vehicle accesses the intersection with a positive wait time (has to stop at intersection), then the setup cost is the time it takes to accelerate from a stopped position. This setup cost is applicable only to the first vehicle in the queue. Therefore the setup cost is sequence dependent family based.

h. Job priority: as defined in issue two, the job priority for emergency vehicles is higher than other vehicles and will be defined as w_j.

i. Due date: There are no such due dates defined in traffic systems, but for the optimization purpose we calculate the due date of vehicles based on the time required by the vehicle to reach its destination without any stop and is denoted by d_j.

$$\boldsymbol{\beta} = \text{chains}, m_j, r_i, d_j, t_j, \text{ fmly}, st_{sd,\,b}$$

Optimization Criteria γ. In this problem we need to maximize the number of vehicles that reach their destination non-stop. So we need to minimize the number of late vehicles.

In order to keep the fairness we need to make sure that average tardiness of vehicles is minimum. Importance should be given to minimizing the average tardiness.

$$\gamma = \min\left(\overline{WT^2} * \sum U_i\right)$$

Based upon above properties the problem of self-synchronization can be defined in the below "*job shop scheduling*" problem.

$$J9 \mid chains, m_j, r_i, d_j, t_i, fmly, st_{sd,b} \mid \min\left(\overline{WT^2} * \sum U_i\right) \quad (4)$$

The problem defined in (4) is obviously Strong NP-Hard.

CHAPTER 4

SELF-SYNCHRONIZATION SOLUTION

This dissertation work focusses on the conflict-free sharing of road intersections by vehicles moving in different directions. We propose a new approach which is referred to as self-synchronization of moving vehicles. In our approach, we aim to install a traffic management logic with some intelligence that gives decision making capabilities to vehicles intended to share road intersections. Under our scheme, the intelligent system in vehicles will communicate with other vehicles in the vicinity and synchronize with them such that every vehicle can move through the intersection without the need of traffic signals or any other external regulating agent.

4.1. Equipment

A special GPS device which has a transceiver connected to it is installed in each vehicle. The transceiver is capable of sending and receiving information using DSRC protocol. To provide some intelligence to vehicles, the GPS device is also enabled with a special algorithm which will analyze the data received from other vehicles and will be responsible for making decisions. The GPS issues voice commands corresponding to decisions made.

4.2. Setup and Approach

Figure 6 shows setup for self-synchronization at one intersection. We define two types of spaces (a) a Synchronization Space and (b) a Synchronized Space around an intersection. Figure 6 shows an intersection of roads 1, 2, 3 and 4. The outer circle represents synchronization space. Its size is based on the capacity of the transceiver, speed limit at the

intersection, length of conflict-free distance (this may include more than one intersection) and, number of vehicles in the vicinity. Area of a synchronization space may contain more than one intersection to facilitate nonstop synchronized movement of vehicles across multiple intersections. As soon as vehicles enter in synchronization space, they start communicating with other vehicles that are already inside the synchronization space. The smaller circle represents synchronized space. All vehicles must make their decisions before entering the synchronized space. The size of synchronized space is decided based on the speed limit in the synchronization area, number of vehicles present in the vicinity, and the area of the synchronization space.

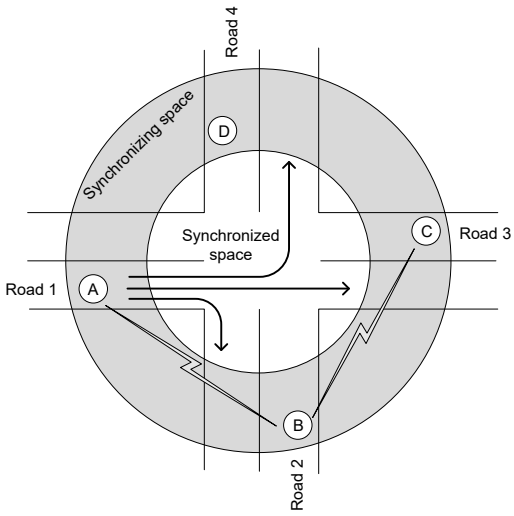


Figure 6 Scheduling moving vehicles

When the vehicles enter the synchronization space, they must establish a communication link with all the vehicles that are already present in the synchronization space of that intersection and then transmit and receive contextual information [27] in real-time.

We define the term “conflicting vehicles” as a pair of vehicles in synchronization space that are traveling in conflicting directions (Table A1). Vehicles enter synchronized space only after a mutual decision is made. For example in Figure 6, vehicle B is synchronizing with vehicles A and C. This way, each pair of conflicting vehicles from different directions say (A, B) know whether A has to cross the intersection before B or otherwise. Our algorithm also takes care of starvation. How long a vehicle should waits for other vehicles can be based on waiting time, and the number of vehicles passing through from other directions. Consequently it passes through the intersection at some point.

In our approach, each synchronization covers three consecutive intersections for each moving vehicle. This will eliminate stop and go situations at each intersection and minimize waiting time. Such situations happens quite frequently in current traffic management system because the traffic signals are not synchronized. Figure 7 displays a grid of 3 X 3 intersections where vehicles moving from left to right and from top to bottom.

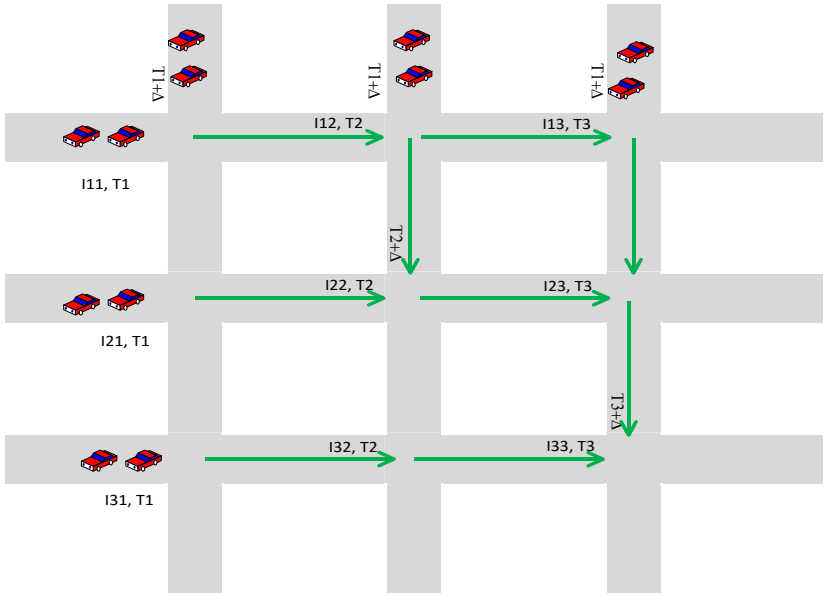


Figure 7 One Way Traffic Example on Grid of 3 X 3 intersection

In the example above, we have shown only one directions flow of traffic. The vehicles at intersection I_{11}, I_{21} and I_{31} start moving from left to right at time T_1. If these vehicle move without stopping, they will reach the first intersection at time T_2 and the next intersection at time T_3. To facilitate the non-stop movement for vehicles moving from left to right, the vehicles that are moving from top to bottom must start at $T_1+\Delta$ (Δ is a small time interval that is required by the group of vehicles to cross an intersection), so that they will reach the next intersection after the vehicles from the other direction already have crossed the corresponding intersection. This way, vehicles from both directions can cross all the intersections without stopping at any of these intersections.

This example illustrates that if we intend to achieve the non-stop movement of vehicles through multiple intersections, it requires the synchronization of their movement across

multiple intersections on their route. In the above example, the vehicles at intersection I_{31} must synchronize with the vehicles at I_{13}, and the vehicles at I_{21} must synchronize with the vehicles at I_{12}. The vehicles starting at I_{31} do not have to synchronize with the vehicles at I_{12}. Similarly, the vehicles starting at I_{21} don't have to synchronize with the vehicles at I_{13} because the timing of these vehicles don't create any conflicting scenarios. In this dissertation we focus on synchronization of vehicles at up to three consecutive intersections.

4.3. Solution For Communication Protocol

4.3.1. Overview

We envision that it will require transmitting contextual information [27] related to a moving vehicle to long range. This will help the synchronization in movements of vehicles in a very large area. This will help in regulating the traffic and better control over-congestion scenarios. This protocol will make use of a WSMP "radio channel control feature" in the application layer to provide high packet delivery ratio in all conditions. The analysis of WAVE/DSRC shows that packet delivery ratio decreases when the number of vehicles increases in one cell [30]. This decrease in ratio is due to CSMA/CA technique used while acquiring the transmission channel. Therefore, to reduce the number of random data collisions, we propose an application layer based solution where each vehicle will be provided with a unique virtual id in a cell. All vehicles transmit their data on a specific time slot based on virtual id and the number of vehicles in the cell at that time. The details of this technique are explained later in the paper. This approach reduces data collision in the communication channel because at a given point in time only one vehicle will be trying to transmit a packet. Since data collision will be negligible, the packet delivery ratio will improve.

4.3.2. Why this protocol

There are two types of basic networking protocols available: 1) protocol with collision avoidance 2) protocol with collision recovery.

The protocols which work as collision recovery protocol are not usable in this scenario because it is a real time application where synchronization between cars must be done within a stipulated time period. Similarly there may be huge numbers of cars trying to transmit at the same time so if we use collision recovery protocol, probability of getting data collision is very high. Therefore, it is not suitable for this application.

The existing networking protocols which use collision avoidance may suffer with the starvation issue. These protocols don't guarantee that all the nodes in the network, which are trying to transmit data, will get fair time for their transmission. These protocols are based on the random selection of the transmission node, not the round robin.

The other issue with the existing networking protocol is that it doesn't handle the scenario of frequently joining and leaving nodes from the network.

4.3.3. Definitions used for self-Synchronization communication protocol

a. Contextual information [27]: Information about vehicles (Location, Identity, Speed, Direction, Driving Predictions etc.) that is useful in various applications.

b. Cell: A virtual area on the map that defines boundaries for data transmission for the group of vehicles in that area. Typically, if the transmission range of communication hardware is 1000 m then the radius of the Cell should be 500 m. Each Cell can be identified by its unique ID (Cell ID)

c. Cell Master: one of the vehicles that is inside the virtual boundary of the Cell and is responsible for communicating contextual information [27] of all the vehicles in this cell to all adjacent cell masters.

d. Intersection: A place in the cell where two or more roads cross each other.

e. Non-Intersection: A place which is neither an intersection nor within the range of communication of an intersection that resides in the same cell.

f. Synchronization Space: A circular area around the intersection that establishes where a vehicle must be able transmit and receive contextual information [27] to synchronize its movement relative to other vehicles in the synchronization space of that intersection.

g. Synchronized Space: a smaller circular area around the intersection. A Vehicle must have finished its synchronization with all other vehicles that are in the synchronization space of that intersection before entering the synchronized space.

4.3.4. Packet Format

The protocol is a bit oriented protocol. Figure 8 shows the packet format for this protocol. Table 3 shows the format of the flag field from the communication protocol. Other fields of this protocol are described below the Table 3.

Flag 8Bits	Version 8Bits
Packet length 16 Bit	
SenderID 16 Bit	
Location ID 16 Bit	
Content fields variable length (<Type 8 Bit><Length 0-16 Bit><Value length defined by length field>)	

Figure 8
Self Synchronization Communication Protocol Packet Format

Table 3

Format of the Flag in the Self Synchronization Communication Protocol Packet

Type	**Length**	**Value**
Communication type	1bit	0 = Broadcast, 1=Unicast
Packet Type	4bit	0000=Contextual Information from vehicle
		0001=Message from static transceiver (Infrastructure)
		0010= Register Request by vehicle
		0011= Drop request by vehicle
		0100= Emergency request packet
		0101= Registration acceptance information
		0110= Periodic Message from cell DataBridge for resource advertisement
		0111= Transfer request for Cell DataBridge Responsibility
		1000= Acceptance for Cell DataBridge Responsibility
		Can add 7 more packet type
Duplicate bit	1 bit	0-not a duplicate 1-duplicate packet
Unused	2 bits	Can be used in future

Version. This field tells about the version of this protocol. At present this is 1.0 where the last three bits (least significant bits) for decimal digits and 5 bits (most significant bits) are for integer parts. This version is a laboratory version and can be enhanced for outdoor use.

Packet length. This field shows complete length of the packet in terms of bytes.

Sender ID. This field recognizes the sender whether it is a static transmitter or a transmission from a car. In either case this sender id uniquely identifies the sender e.g. the car id.

Location ID. This field recognizes the cell from where the message is being transmitted. Location is the combination of cell id and intersection id. This field will be used to identify the Inter-Cell or Intra-Cell communication.

Content fields. This field is a variable length field. There can be multiple content fields in a packet. A content field is the combination of three field's i.e.

Type	1 byte	2 MSB tells the size of length field, 6 LSB tells the type of field. Some field types are explained below.
Length	0-2 bytes	tells the length of the field value.
Value		is the value of content field

Following Table 4 displays possible values for content fields:

Table 4

Content fields defined in this version of protocols are.

Type	header Field Name	Description
1	Current Time (Fixed Width 8 Bytes)	Contextual information related to sender.
2	Longitude (Fixed Width 4 Bytes)	Current Longitude of vehicle
3	Lattitude (Fixed Width 4 Bytes)	Current Lattitude of vehicle
4	Speed (Fixed Width 1 Byte)	Current Speed of vehicle
5	Intersection Lists Variable Width with 2 Bytes of length field	List of intersection up to 10 that are in order the vehicle is about to travel. It will be a list of the type length value list.
6	Intersectionid (Fixed Width 8 Bytes)	I.D. of the Intersection
7	Seconds to Reach (Fixed Width 2 Bytes)	Time to reach the intersection from the current time in seconds.
8	Car List (Variable Width with 2 Bytes of Length Field)	List of cars cronologically ordered with respect to their arrival in synchronization space of the intersection.
9	Decision List (Variable Width With 2 Bytes of Length Field)	A schedule of vehicles that defines the order in which vehicles are going to access the Intersection
10	Virtual VehicleID (Fixed Width 2 Bytes)	Virtual Vehicle ID of a vehicle in a cell

The above mentioned content fields are defined for the laboratory based model. The content field has been designed as a highly dynamic field so that it can accommodate any type of future requirement.

It is fairly simple to recognize the message source i.e. static base stations (Static transceiver) or moving vehicles, using the packet type. Similarly, the message content can be easily processed using the content type.

4.3.5. Setup for the Protocol

The self-synchronization communication protocol is an application layer protocol, which is developed on top of DSRC/WAVE protocol stack. This protocol should enable short range as well as long range communication channels. To achieve it requires a special cell setup which is shown in Figure 9 below:

Figure 9 Communication Protocol Cell Structure

Following are the assumption for protocol setup:

a. Every adjacent cell uses a different service channel for data transmission.

b. Radius of each cell is 250 meters.

c. The maximum transmission range with high power transmission is 1000 meters and the transmission range for normal transmissions is 500 meters.

d. Channel 184 is assigned for High Power transmissions. This channel will be used for inter-cell transmission.

e. All vehicles are equipped with 3 radios: a) For control channel, b) For service channel inside the cell and c) For inter-cell communication.

4.3.6. Procedure of Self-Synchronization Protocol

Initialization.

a. We assume that there is no other vehicle or infrastructure in communication range therefore initially all the lists will be initialized with null.

b. Every Vehicle has to obtain its own contextual information (using a location management device e.g. GPS) before starting the communication and before they start moving on the roads.

c. As soon as a vehicle gathers its own contextual information, it starts listening to the communication channel for control information and then follows the steps below:

There will be two types of communication networks in this protocol: a) Intra-cell communication, b) Inter-cell communication.

Intra-Cell Communication.

a. A vehicle listens to the control channel for the broadcast from the cell master. If no cell master is present, it will announce itself as the cell master. To avoid two or more vehicles announcing themselves as cell masters at the same time, this protocol will use the CSMA/CA technique.

b. Once the cell master is recognized, each vehicle sends its identification number to the cell master on the control channel. The identification number is unique to each vehicle just like a MAC ID.

c. The cell master registers the vehicles and sends back a unique virtual id, within the cell, to the registered vehicle.

d. The cell master maintains the list of vehicles that are present in the cell. When it receives a new vehicle id, it adds the id to the list while maintaining a chronological order of the vehicle's arrival.

e. The cell master broadcasts the list of vehicles in the cell in its broadcast on the control channel.

f. There may be a data collision in the first step. Therefore, the above two steps are repeated until the car receives a list with its own vehicle id included in the list.

g. Each car calculates a time slot based on the number of vehicles in the list and its own location in the list. The time slot calculation uses some assumptions about the amount of data to be communicated by each vehicle.

h. Each car continues to listen to the communication channel and keeps its own time slot updated based on the time taken by previous transmissions.

i. The car starts transmission of its contextual information when its own time slot occurs. It is possible that more than one car is trying to transmit at the same time. This situation is taken care by the CSMA/CA protocol.

j. The steps 4, 5 and 6 keep repeating as long as there is at least one car in the synchronization space.

k. The cell master removes the vehicle id from the list as soon as the vehicle moves out of the current cell.

l. When a cell master is about to leave the cell, it selects a new candidate to be cell master based on how long the new candidate is going to remain in the cell.

Inter-cell Communication. Only the cell-masters will be involved in inter-cell communication. The long range radio will be used for inter-cell communication among cell-masters.

a. Each cell master will transmit the contextual information about the vehicles in its own cell as well as for all the vehicles in its adjacent cells.

b. Communication among cell masters will be handled using CSMA/CA protocol. There will be at most 19 Cells that are competing for inter-cell transmission at any given point in time (Figure 9).

4.3.7. States of Self-Synchronization Protocol

Below are the states of each node and the state diagram:

a. Enter the cell as unregistered node

b. Waiting for cell-master

c. Working as cell-master

d. Working as registered node but not cell-master

e. Calculating the transmission slot.

f. Listening to the channel for contextual information.

g. Transmitting the contextual information.

h. Leaving the cell as not a cell-master

i. Perform registration process as the cell-master.

j. Leaving the cell as a cell-master

4.4. Solution For Self-Synchronization At One Intersection

Existing Scheduling Solutions. Existing scheduling solutions such as FCFS (First-Come First-Serve), scheduling of network resources, and burst-based scheduling for non-blocking ATM switches with multiple input queues [6] are not suitable for real-time scheduling. The self-synchronization approach concentrates on resolving the scheduling problem by the entities involved in the conflict instead of the controlling agent.

4.4.1. Possible Approaches for Finding Scheduling Solution

The self-synchronization scheduling problem can be solved using two approaches.

a. Static approach: a simple stop-and-go solution where each vehicle will wait for all other vehicles which are already in the queue before passing the intersection.

b. Dynamic approach: a complex expert system based upon the algorithm which generates the near optimal schedule for all the vehicles at the intersection based upon the optimization criteria.

The static approach works only at four-way stop intersections. This approach is similar to EDD and is useful when priority and processing times for all the vehicles are the same, but with batch dependent setup times this approach is not feasible.

The work done in [7] shows us the various programming approaches to address the scheduling issues that involve genetic algorithms, heuristics, neighborhood searching techniques and expert systems. The problem of the self-synchronization at one intersection is

a real time problem and must be solved within a very short time frame. For example, at 40Mph, a vehicle travels 17.78 meters per second. This means that the dynamics of the problem change every second. The number of vehicles in the problem area, their position, and their release time will be different in one second. The solution for the problem could be very different when a new vehicle joins. Therefore, the solution must be found within a very short stipulated time slot. We assume that problem dynamics remain constant during this time slot. Genetic algorithms and neighborhood searches require the generation of multiple solutions and their comparison, which could be very time consuming, hence may not be effective in this scenario. This is a very special problem, therefore, we propose using the problem specific expert system to generate fast and near optimal solutions. In special cases the expert system approach may work as well as the static approach and will be capable of handling all sorts of scenarios at an intersection. We, therefore, focus our investigation on the expert system approach.

4.4.2. Self-Synchronization Solution Representation

It is very important to represent the candidate solutions in effective ways such that the feasibility of solutions can be tested quickly and easily. The representation of solutions must be simple to understand. For the self-synchronization at one intersection problem, the solution should provide a schedule for the vehicles in the problem domain. Using this schedule, any vehicle in the problem domain must be able to determine when it can access the intersection.

As described in section 3.5, the solution vector generated at base time T_0 is represented as following:

$$SV_I = \{V_1, V_2 \ldots\ldots\ldots\ldots V_n\}$$

In this solution vector, vehicles are arranged in the order that they should access the intersection. To calculate the feasibility of this solution, it requires the determination of the completion time C_i for each vehicle. That can be done by coupling the start time S_i and processing time p_i with each vehicle node. To make the feasibility calculation easy, we should also provide the due date information of each vehicle. In our problem there are direction dependent setup times. Since we include start time, processing time and due date with each node, it is not required to include setup time in the resulting schedule. We can also add wait time for each vehicle in the resulting schedule to ease the calculation of $wait_{max}$ calculations. After considering all the points noted above, the modified solution vector can be as below:

$$SV_I = \{(V_1, s_1, p_1, d_1, w_1), (V_2, s_2, p_2, d_2, w_2) \ldots\ldots\ldots\ldots\} \qquad \text{(Solution 2)}$$

4.4.3. Upper and Lower Bound for $Wait_{max}$

When vehicles are ready to access the intersection from all the conflicting directions, $Wait_{max}$ will be minimum if we change the direction in a round robin way after every passing vehicle. $Wait_{max}$ will be maximum if we allow all the vehicles from one direction to go before changing the direction. The upper bound of $Wait_{max}$ can be calculated as below:

a. Add the processing time for all the vehicles in each of the directions separately.

b. Discard the total processing time for the direction with the minimum processing time.

c. Add the total processing time for the remaining directions.

d. Add the setup time for all the directions.

The upper-bound calculated for $Wait_{max}$ using the above procedure may not be feasible when the intersection is saturated from at least one direction. That means at least one direction

has a very long queue waiting to access the intersection. In this kind of scenario, we must define a threshold value for $Wait_{max}$. For example, at a 4-way intersection, if we allow each conflicting direction for a maximum of 30 seconds, then the threshold limit for $Wait_{max}$ should be 90 + ∑ST. The max time for each direction can be defined with intersection properties.

When traffic is very low, the optimization criteria of $Wait_{max}$ may cause the change of directions frequently to get the optimal result, but it is not feasible to change the direction in the low traffic scenario. Instead, all the vehicles from one direction should be allowed to go before changing directions in order to increase the throughput. In the low traffic scenario, $Wait_{max}$ doesn't have any significance. Hence, we must set a Lower Bound for $Wait_{max}$. The $Wait_{max}$ optimization criteria should be ignored if upper bound of $Wait_{max}$ is less than its Lower Bound. The lower Bound for $Wait_{max}$ should also be defined with intersection properties.

4.4.4. Dynamic Programming Solution for Self-Synchronization at One Intersection

Let there be 4 conflicting directions and A, B, C and D are a set of vehicles from each of the conflicting directions. Let V be the set of all the vehicles. So the following properties hold:

$$A \cap B = A \cap C = A \cap D = B \cap C = B \cap D = C \cap D = \Phi$$

$$A \cup B \cup C \cup D = V$$

If Pi be the permutation set of all possible sequences of n number of vehicles from V and precedence rule of vehicles in A, B, C, and D is ignored then there will be $!n$ possible sequences. In our problem, we need to keep the precedence of vehicles from A, B, C and D. if |A| = w, |B| = x, |C| = y, |D|= z then the total possible permutations in set Pi are:

$$!n/_{!w * !x * !y * !z} \quad (1)$$

If S be one of the arbitrary solution sequence from Pi and if total tardiness is optimization criteria which is represented as function f(S) for solution S then:

$$f(S) = \{\sum_{i=1\ to\ n} \max(c_i - d_i, 0)\} \quad (2)$$

Suppose we find an optimum solution for this problem and S* = [{v_1*}, {v_2*}, {v_3*},……..(v_n*}} is the optimum schedule of vehicles. Then using (2)

$$f(S^*) = \{\sum_{i=1\ to\ n} \max(c_i^* - d_i^*, 0)\}$$

The above equation can be decomposed for any 1 <= k <= n

$$f(S^*) = \{\sum_{i=1\ to\ k} \max(c_i^* - d_i^*, 0)\} + \{\sum_{i=k+1\ to\ n} \max(c_i^* - d_i^*, 0)\}$$

$$f(S^*) = f(S_1^*) + f(S_2^*) \quad (3)$$

where S_1* = {{v_1*}, {v_2*} …….(v_k*}} and S_2* = {{v_{k+1}*}, {v_{k+2}*} …….(v_n*}}

Let $A_1 = S_1^* \cap A$, $A_2 = S_2^* \cap A$

$B_1 = S_1^* \cap B$, $B_2 = S_2^* \cap B$

$C_1 = S_1^* \cap C$, $C_2 = S_2^* \cap C$

$D_1 = S_1^* \cap D$, $D_2 = S_2^* \cap D$

Then the following Lemma is true:

Lemma 1: If S* is the optimum sequence for vehicles in A, B, C, and D then the subsequence S_1* obtained in (3) is the optimum sequence for vehicles in reduced sets A_1, B_1, C_1, and D_1.

Proof: if S_1* is not the optimum sequence for the vehicles in reduced sets A_1, B_1, C_1, and D_1 then we can find an optimum sequence for this subset. Let this optimum sequence be S_{11}* such that S_{11}* <> S_1* and f(S_{11}*) < f(S_1*).

Now we can construct a new sequence for the vehicles in A, B, C, and D by combining S_{11}* and S_2*. The objective function value of this new sequence will be $f(S_{11}^*) + f(S_2^*)$ which is less than $f(S_1^*) + f(S_2^*)$. This contradicts the assumption that S* is the optimum sequence for vehicles in A, B, C, and D. Thus, the converse is true that S_1* is the optimum sequence for the vehicles in the reduced sets A_1, B_1, C_1, and D_1.

Let V_1 be the subset of V and $V_2 = V - V_1$ which is the complement of V_1. Let C_{V1} be the sum of processing time for all the vehicles in V_1.

$$C_{V1} = \sum_{v_i \in V_1} p_i \tag{4}$$

Suppose a solution sequence S for V is created such that all the vehicles in V_1 are accessing the intersection before all the vehicles in V_2. If S is the optimum solution for V, then according to lemma 1 the vehicle from V_1 must be optimally sequenced irrespective of how vehicles from V_2 are sequenced. Because no vehicle in V_2 will access the intersection before C_{V1}, we can construct the reduced conflicting direction subsets A_1, B_1, C_1, and D_1 as $A_1 = V_1 \cap A$, $B_1 = V_1 \cap B$, $C_1 = V_1 \cap C$, $D_1 = V_1 \cap D$.

So the minimum tardiness value for V_1 will be:

$$F(V_1) = f(S_{V1}^*) = \min_{S_{V1} \in P_1} \left\{ \sum_{v_i \in V_1} \max(c_i - d_i, 0) \right\} \tag{5}$$

Where P_1 is permutation set V_1 while maintaining the precedence order of the vehicles in A_1, B_1, C_1, and D_1. If v_k is the last vehicle to access the intersection from V_1, then its corresponding access time will be C_{V1}. So, (5) can be written as below:

$$F(V_1) = f(SV_1^*) = \min_{S_{V1} \in P_1} \left\{ \max(C_{V1} - d_{v_k}, 0) + F(V_1 - \{v_k\}) \right\} \tag{6}$$

If V_1 is empty then we define $F(\Phi) = 0$ which is the boundary condition. We can generalize (6) for the entire problem as below:

$$f(V) = f(S^*) = \min_{S \in Pi}\left\{\sum_{v_i \in V} \max\left((c_i - d_i), 0\right)\right\}$$

$$= \min_{S \in Pi}\left\{\max(C_V - d_{v_k}, 0) + F(V - \{v_k\})\right\} \quad (7)$$

The iterative equations (6) and (7) and the boundary condition $F(\Phi) = 0$ allows us to find the minimum value of the optimization function for the whole problem. We can take the bottom up approach and start from only one vehicle in V_1 and using (6) find the minimum value for the first vehicle from each conflicting direction. Then, we can move on with two vehicles in V_1 having different combination of vehicles from each conflicting direction. The minimum values we find in prior steps will be stored to be used in later steps so that we can save the additional iterations. Using this method, we can avoid to iterating through all possible permutations. It therefore provides a faster solution for our problem.

Though using this dynamic programming method stated above, we avoid iteration through every possible set of permutations, yet the complexity of this algorithm is still very high which expands quickly with the increase in number of vehicles in V.

4.4.5. Expert System Solution for Self-Synchronization at One Intersection

The self-synchronization of one intersection problem is a unique scheduling problem where the vehicle's arrival rate at the intersection is random. At some point in time, the arrival rate may be very high and at other times it may be very low. It causes delay if vehicles have to stop at the intersection which is referred to as the batch dependent setup time.

The job families of this problem are both compatible as well as incompatible. If A, B, and C are three families at the intersection, there may be a case where A and B are compatible, A and C are compatible, but B and C are not compatible. At a 4-way intersection, directions like turn right can be grouped together with two different groups of direction. We mark them

as minor directions. The directions like Turn Left and Go Straight are major directions. While scheduling, we give priority to major directions. It doesn't cause any delay to vehicles moving in minor directions, since they can be batched twice for processing in one scheduling cycle of all directions. Based on the compatibility matrix for a 4-way intersection (Table A1) there are four conflicting groups of directions.

Due to the setup time that is required while switching the direction for processing vehicles, it is very important to find feasible lot sizes from each direction. In this problem, the total processing time of a lot depends on jobs included in the lot. If all the vehicles have the same processing time, then including all the vehicles from one direction before moving to the others can be a good strategy to optimize the average tardiness. This approach results in high $Wait_{max}$ values. On the other hand, to get optimal $Wait_{max}$ value, changing the direction with every passing vehicle is best. When we bind these two optimization criteria, the best result is when the lot size is half the number of vehicles from one direction (assume vehicles are evenly distributed).

In the real world this is not the case. The processing time for vehicles varies. Also the vehicles are not distributed across the direction of movement. Considering the real world scenario, it becomes more important to find a feasible lot size from each direction. Due to the batch dependent setup time, selecting the next family to process does impact the optimization value.

Let A, B, C and D are a set of vehicles from 4 conflicting directions and V is the set of all the vehicles. If Pi is the set of the permutation sequence on V, S be one of the sequences

from Pi. The optimization function f(S) can be defined as (2) in section 4.4.4. Size of Pi is defined in (1) in section 4.4.4.

Finding the optimal solution from all the possible permutations is very difficult. We propose the following approach to find a feasible solution for this problem:

a. Find a direction to process next.

b. Find a feasible batch size to process from a selected direction.

c. Mark the batch as scheduled and repeat the steps for the remaining vehicles until all the vehicles are marked scheduled.

If there are K conflicting directions, there are !K possible ways to select the next direction for processing. We propose introducing preferred cycle rules for conflicting directions. For example, in this case the preferred cycle of direction could be A -> B -> C -> D and repeat. Further, selection of the next direction to process will be based on the availability of vehicles from the direction and wait time of the direction. The Solution S can be shown as below:

$$S = \{A_1, B_1, C_1, D_1, A_2, B_2, C_2, D_2 \ldots\ldots\ldots\ldots A_n, B_n, C_n, D_n\}$$

Where $A_i \subset A, B_i \subset B, C_i \subset C, D_i \subset D$

$$A_i \cap A_j = \phi \; \forall \; i \neq j \quad \text{and} \quad \cup A_i = A$$

Suppose S* be the optimal sequence of batches as below:

$$S^* = \{A_1^*, B_1^*, C_1^*, D_1^*, \ldots\ldots\ldots\ldots A_n^*, B_n^*, C_n^*, D_n^*\}$$

Then optimization function for set S* is

$$f(S^*) = \min_{S^* \in P_1} \left\{ \sum_{v_i \in V} \max(c_i - d_i, 0) \right\}$$

Suppose A_1* precedes all the remaining batches in S* and let A_2* = A- A_1* then we can decompose the above equation as:

$$f(S^*) = \min_{S^* \in P_1} \left\{ \left(\sum_{v_i \in A_1} \max(c_i - d_i, 0) \right) + \left(\sum_{v_i \in S^* - A_1} \max(c_i - d_i, 0) \right) \right\}$$

$$f(S^*) = f(S^*_1) + f(S^*_2) \quad (8)$$

Where $S^*_1 = \{A_1\}$ and $S_2^* = \{B_1^*, C_1^*, D_1^*, \ldots\ldots\ldots A_n^*, B_n^*, C_n^*, D_n^*\}$

Using lemma 1 we can prove that S_2* must be the optimal batch sequence for the reduced set $V_2 = \{B^*, C^*, D^*, A_2^*\}$. The A_2 is placed at the end of the sets to establish next processing batch selection precedence. We can generalize this equation for the whole problem with boundary condition $F(\Phi) = 0$ as below:

$$F(V) = F(A, B, C, D) = f(S^*) = \min_{S^* \in P_1} \{(f(A_1) + F(B, C, D, A_2)\} \quad (9)$$

When A_1 precedes all the vehicles in schedule S*.

In the last section, we have shown that the complexity of finding an optimal solution using this approach is very high. To find a feasible solution that is near optimal, we assume that all the vehicles from direction B, C and D will be ready to process when we finish the processing of batch A_1. While calculating the batch size of A_1, we also assume that processing all the vehicles from B, C, and D before processing vehicles from batch A_2 yields a feasible solution. Let C_1 be the total processing time of the last vehicle from batch A_1. If all the vehicles from directions B, C, and D are ready to access the intersection, the wait time for every vehicle will be increased by C_1. If C_2 is the total processing time for all the vehicles from directions B, C and D, including setup time for all the directions, then wait time for the vehicles in A_2 will be increased by C_2. If n_1 is the size of A_1, n_2 is the size of A_2 and n_{bcd} is the size of {B, C, and D} combined. Then, below will be the total wait time for all the vehicles:

$$f(V) = 0*n_1 + C_1 * n_{bcd} + C_2 * n_2 \quad (10)$$

The optimal value of the equation (10) will provide the near optimal batch size for A_1. The complexity of finding the optimal value for equation (10) is linear. Using the equation (9) and (10) we can iteratively find a feasible solution for this problem in the polynomial time complexity.

To improve the quality of the solution, we apply the following rules while calculating the batch size from each direction.

a. The batch processing size must not exceed the $wait_{max}$ cap for that conflicting direction.

b. The processing time for the other conflicting directions B, C, and D are capped using the $wait_{max}$ cap of the corresponding direction. Also the count of vehicles are capped using this parameter.

c. The problem of Self-Synchronization of the vehicles at one intersection involves groups of non-conflicting directions that can access the intersection together while conflicting with other groups. This property adds another level of complexity to the equation. While selecting the batch size for one conflicting group, we need to account for all the non-conflicting directions in that group. Therefore, equation (9) is optimized for a group of directions in place of just one direction.

d. A batch is selected from every conflicting group in a round robin sequence. The batch size can be 0 from any conflicting group based on the value of the optimization equation. To eliminate the recursive wait, we apply a rule that if at least one vehicle waiting from at

least one non-conflicting direction in a processing group, then the batch size for that group can't be zero.

The self-synchronization problem is a real time problem. Vehicles join and leave the queue at the intersection arbitrarily. If we calculate a whole new solution for every changing scenario, it will not be feasible in the real world. The vehicles that are set to access the intersection in the next few seconds can't be stopped immediately, doing this may cause hazardous situations. Hence while generating a new schedule we must consider if there is a schedule exists for prior situation at the intersection. If a schedule exists, the vehicles that joined the intersection recently have two options a) to be allotted to an existing batch or b) form a new batch. The new vehicles can be allotted to an existing batch, if the batch was terminated due to unavailability of vehicles and have enough capacity to accommodate new vehicles such that adding them to the existing batch does not impact the overall value of optimization function or reduce it.

4.5. Solution For Multiple Intersection Synchronization (Green Channel)

The immediate question arises why we can't use the solution algorithm of the one intersection problem to solve the green channel problem. It is straight forward that the solution algorithm for one intersection can be replicated at every intersection such that it provides conflict free movement throughout the route of a vehicle. But if we use the same approach at every intersection, it doesn't guarantee that vehicles have non-stop movement at every intersection before accessing it. Though we get a conflict free movement of vehicles, it may cause vehicles to stop frequently. This will result in longer travel times for each vehicle and hence may cause an increased number of vehicles in the problem domain. In other words, it

may cause a congestion scenario. Therefore, to reduce the total time taken by a vehicle to travel from point A to point B through multiple intersections, it requires non-stop movement of a vehicle throughout its journey. The green channel problem addresses this issue and ensures a vehicle passes at least a few consecutive intersections non-stop.

In section 3.6.3, we have shown that the self-synchronization of moving vehicles through multiple intersections or the green channel problem is a job shop scheduling problem. There is plenty of work done on job shop scheduling problems. The work in [8] provides a practical approach towards general job shop scheduling problems which involves different jobs on different type of machines. The work in [42] shows a genetic algorithm approach for resource constraint scheduling. In [43] an advanced heuristic approach is used to solve non preemptive open shop scheduling problems with sequence dependent setup times. There are several different approaches that can be used to obtain a feasible schedule for given job shop scheduling problems. Some of the approaches are discussed in section 2.2.

4.5.1. Possible Approaches for Finding Scheduling Solutions

The vehicular environment is a very dynamic environment. It requires obtaining of a scheduling solution before the current problem scenario changes. Approaches like genetic algorithms, neighborhood searches, neural networks and dynamic programming require processing through multiple possible scheduling sequences to find a feasible solution. In the vehicular environment, we do not have the luxury of time. The algorithm must be fast and should produce the feasible solution within a couple of seconds.

We can opt for advanced heuristics or environment specific expert systems to obtain a feasible schedule. The problem of providing green channel to the maximum number of

vehicles in the problem set has several unique properties such as machine constraint, transfer time, setup time, conflicting and non-conflicting families of jobs, etc. Considering the uniqueness of the problem, we propose to apply the expert system approach.

4.5.2. Setup and Approach

Setup for multiple intersection problem is shown in Figure 7. Suppose a vehicle V has three consecutive intersections X, Y and Z on its route. The vehicle is reaching intersection X at time T_0. It takes t_1 time to travel from intersection X to Y and t_2 from Y to Z. The goal of this problem is to make intersection X accessible for V at time T_0, intersection Y accessible at time $T_0 + t_1$, and intersection Z at time $T_0 + t_1 + t_2$. Figure 5 shows that there can be 12 freedom of movement at a 4-way intersection. At a 4-way intersection, every direction of movement conflicts with 6 other directions. Figure 10 displays a scenario of vehicles conflict when synchronized for the three intersections ahead. All the vehicles that are reaching the intersection indicated with a green circle at time T_0 can reach the intersection indicated with a red circle at the same time. So, if the vehicle at the green circle intersection at time T_0 has the red circle intersection in its route, it must synchronize with the vehicles that are present at the other green circled intersection at time T_0. Here we assume that the travel time between each intersection is the same. If the travel time between different intersections differs, a vehicle may conflict with a very different set of vehicles. Here our emphasis is on the fact that though at a given intersection, movement of a vehicle may conflict with 6 other directions of movement. If we plan to synchronize its movement three intersection ahead, the degree of conflicting directions of movement increases remarkably.

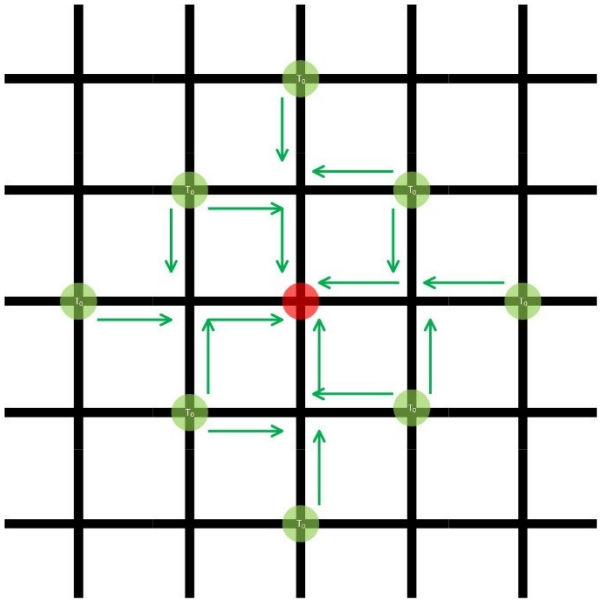


Figure 10 Three Intersection Conflict Scenario

Finding the optimal schedule for this problem is very complex. Due to the precedence rule and machine restrictions a slight change in scheduling at one intersection may cause huge differences in scheduling for the other involved intersections. Therefore, to ease the complexity of the problem we propose the following assumptions:

a. At every intersection there are 12 freedom of movements or conflicting directions. We assume that each batch from each direction is a job instead of every vehicle. Processing time of each batch depends on the number of vehicles included in the batch.

b. We define four major conflicting groups and 4 non-major non-conflicting groups at an intersection similar to the groups defined in the one intersection problem. A 4-way intersection has two cross roads. In general, the directions of cross roads are north-south and east-west. Therefore, we group the conflicting groups moving in the same direction as soft-conflicting groups. So, at a 4-way intersection there would be two soft-conflicting groups.

Each soft-conflicting group involves 2 major conflicting groups and 2 non-major conflicting groups.

c. We define a processing cycle time for each intersection. Each major conflicting direction (or major conflicting group) accesses the intersection one by one in the round robin approach. When each of the directions or groups has been provided access, we call it a processing cycle. For example, there are conflicting groups A, B, C and D at an intersection I. A starts accessing the intersection at time T_0. Then, B starts at T_1. Then C starts at T_2 and D starts at T_3. When D finishes at time T_4, then time ($T_4 - T_0$) is a cycle time for intersection I. Therefore, cycle time at an intersection defines the maximum wait time for a direction. The $Wait_{max}$ for an intersection defines the maximum cycle duration for it. Actual processing cycle time can be obtained by adding the processing time for the major conflicting groups. The processing time for the major conflicting groups is equal to the maximum batch processing time for the directions involved in it.

d. We also define max duration a direction can access the intersection uninterrupted using $wait_{max}$ cap for the direction. $Wait_{max}$ cap of a soft-conflicting group can be obtained by adding the $wait_{max}$ cap for its major conflicting group. $Wait_{max}$ cap for a conflicting group equals the maximum value for the $wait_{max}$ cap for its major conflicting direction.

4.5.3. Solution Representation

Properties of solution for green channel problems are very similar to that of the self-synchronization of vehicles at one intersection. The differences are a) this problem involves more than one intersection. b) The optimization criteria is to maximize the number of vehicles with no wait. In our solution approach for green channel problems, we propose using different

speed settings for every vehicle at every intersection. We can use the solution representation as shown in section 4.4.2. With the addition of the speed setting, for this problem we also need a schedule at every intersection. Therefore, the solution for the green channel problem can be shown as below:

$$SG = \{SV_{I1}, SV_{I2}, \ldots\ldots\ldots SV_{In}\}$$

Where $SV_{Ix} = \{(V_1, s_1, p_1, d_1, w_1, sp_1), (V2, s_2, p_2, d_2, w_2, sp_2) \ldots\ldots\ldots\ldots\}$ represent schedule at intersection I_x. sp_i is speed required by vehicle V_i to arrive at the intersection on or before the start time (s_1). All other symbols are explained in section 4.4.2.

4.5.4. Solution Algorithm for the Green-Channel Problem

Figure 11 displays the setup of three intersections. If transfer time between each intersection is constant, and if the availability of the vehicles is consistent, then finding the solution for the green channel problem is very simple. We can simply set the cycle time for each intersection as twice the transfer time and batch size for each major conflicting group to half of the transfer time. Then, we can achieve non-stop movement for all the vehicles. For example, if there are 4 major conflicting groups A, B, C and D at each intersection, and transfer time between the intersections is 30 seconds, then we set the cycle time as 60 seconds and the batch time for A, B, C and D to be 15 seconds. With reference to Figure 11, if $T_1 = 0$ then $T_2 = 30$, $T_3 = 60$. Delta will be 30 seconds (since there are two major conflicting groups on each cross road.) At Intersection I_{22} vehicles from left to right will reach the intersection at 30 second mark and finish at 60 second mark. Similarly, vehicles from top to bottom will arrive the intersection at 60 second mark and will finish at 90 second mark. This way no vehicle has to stop at any intersection.

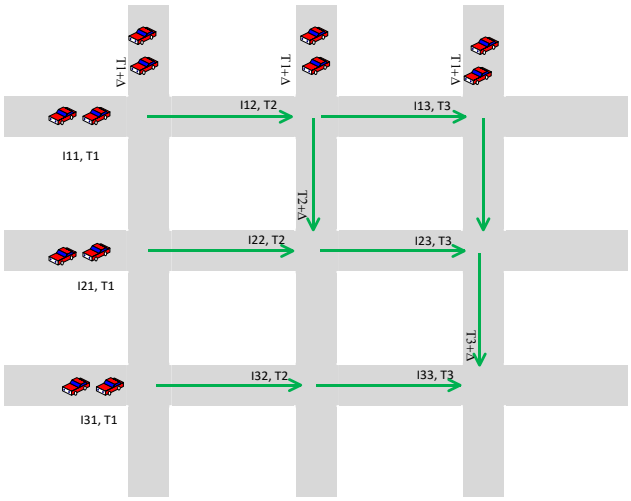


Figure 11. Three Intersection Setup

The scenario presented above is an ideal scenario and it is not always the case in the real world. The transfer times between intersections are arbitrary and the availability of vehicles is also not consistent. Though we can't control the availability of vehicles, the transfer time taken by each vehicle can be controlled by manipulating the travelling speed of the vehicles.

If we know the cycle time for intersections and can calculate a time slot for each of the conflicting groups, we can adjust the release time for a vehicle such that it reaches the intersection when the intersection is available for it. Suppose b1 and b2 are two successive batches from direction A. Batch b1 starts at T_0 and its processing time is P_0. If the cycle time for the intersection is CT then the batch b2 will start at $T_0 + CT$. If a vehicle reaches an intersection between $T_0 + P_0$ and $T_0 + CT$, it will have to stop at the intersection. We call it idle time interval for a direction and it can be calculated as $(CT - P_0)$. A vehicle has to make

decisions if it can reach the intersection before $T_0 + P_0$ by speeding up or if it can slow down and reach the intersection at $T_0 + CT$.

The time vehicles can make-up or lose using speed manipulation depends on the distance between the two intersections. The worst case scenario for a vehicle is when with normal speed limits it is arriving at the intersection exactly at the middle of idle time. If the idle time interval for a direction is large, it will be difficult for a vehicle to gain or lose the sufficient time using speed manipulations, but if the idle time is small then it is easily possible. The time a vehicle can gain or lose can be calculated using maximum/minimum speed limit and distance between two intersections. If there are two intersections I and J, $AvgS_{IJ}$ is the average speed limit, $MaxS_{IJ}$ is the maximum speed limit and $MinS_{IJ}$ is the minimum speed limit between the intersections I and J. TR_{IJ} is the transfer time defined. Then, we define gain time G_{IJ} and lose time L_{IJ} for the intersection I, J as

$$G_{IJ} = |(TR_{IJ} * AvgS_{IJ} / MaxS_{IJ}) - TR_{IJ}|$$

$$L_{IJ} = |TR_{IJ} - (TR_{IJ} * AvgS_{IJ} / MinS_{IJ})|$$

The cycle time depends on the number of vehicles traveling from each direction. Since we cap the maximum batch processing time using $Wait_{max}$ cap for each direction, we limit the idle time for each conflicting direction.

It is very difficult to find an algorithm to get the optimal solution for the green channel problem. We propose the following scheduling algorithm that will ensure non-stop movement for vehicles moving straight, but it can't ensure non-stop movement of vehicles turning left or right.

1. *Select a grid of adjacent intersections to be synchronized. The grid dimension must be at least 3x3.*
2. *Find the min (G_{ij} + L_{ij}) for the grid.*
3. *Set the $Wait_{max}$ for each intersection in grid as (min (G_{ij} ,+ L_{ij})).*
4. *Set $Wait_{max}$ cap for major directions as $Wait_{max}$ / 2.*
5. *Set $Wait_{max}$ cap for minor directions as $Wait_{max}$ / 4.*
6. *For each vehicle V at the intersection I, do the following:*
7. *Find the size and finish time of the last batch from direction, from which vehicle V is moving.*
8. *Find the idle time for the intended direction based on the current cycle processing time.*
9. *If adding V to the last batch causes it to violate $wait_{max}$ cap for the direction or If V will not be able to arrive at the intersection before the finish time of the last batch, then create a new batch starting with vehicle V, and update its release time and required speed.*
10. *If adding V to the last batch doesn't violate the $wait_{max}$ cap for the direction and If V can reach the intersection at the finish time of the last batch, then add V to the last batch. Update the release time and speed for V. Update the start time and speed for all the vehicles impacted by this insert.*

Since we are scheduling vehicles at least three intersections ahead, the cycle times and idle time for each direction and for each batch will be known well in advance. Hence, a sudden

change in schedule is highly unlikely. If the arrival rate of vehicles at an intersection from a particular direction is extraordinarily high and other directions at the same intersection are not crowded, then it will require the vehicles from that direction to slow down or wait at the entry point of the problem grid. (The grid of the intersections for which the scheduling algorithm has been applied.)

CHAPTER 5

SELF-SYNCHRONIZATION SIMULATION AND STASTISTICS

We have developed a computer based simulation program that provides a graphical representation for solution developed for self-synchronization of one intersection problem. The simulation program is divided into four modules: a) contextual information generator, b) message transmitter, c) schedule generator and d) graphics generator. The first three modules are combined into an executable module. One instance of this module is executed for every vehicle in test environment. Only one instance of graphics generator module is executed. These modules are developed using Visual Studio 2012. It is executed on a machine that has Intel I5 processors, 4GB of RAM and Windows 10 OS.

The contextual information generator module is responsible for generating the initial data for a vehicle. This module also recalculates the contextual information for corresponding vehicle after a small stipulated time period.

The message transmitter module is a simulation of communication protocol which broadcasts the contextual information of one vehicle to all the other instances of simulator. Along with contextual information this module is responsible for transmission of calculated schedule.

The schedule generator module is core of this simulation. It gathers contextual information from every vehicles. Using this information it generates a schedule and pass it to transmission module for broadcast.

The graphics generator module collects contextual information for all the vehicles and plot them on the screen based on current longitude and latitude of the vehicle. The graphics refreshed very frequently to smooth out the animation of vehicle movements.

5.1. Analysis and Statistics

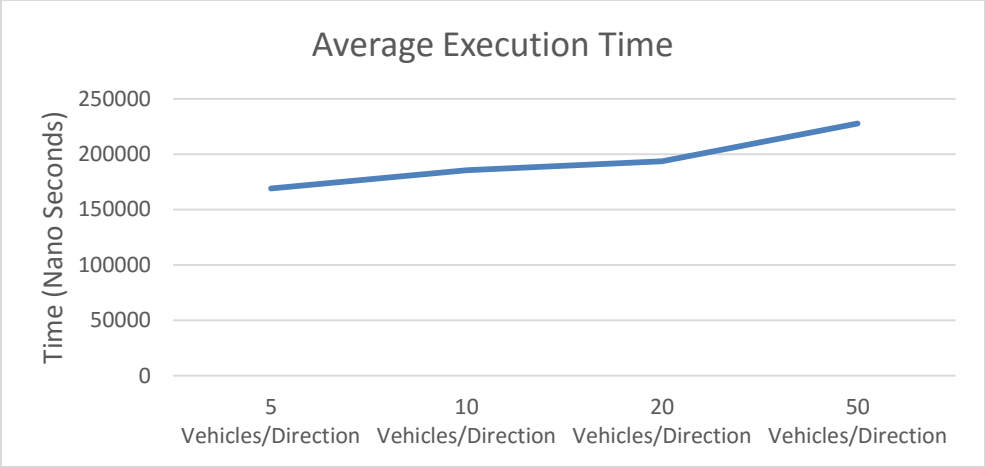
To analyze the scheduling algorithm, we perform numerous test with various sample data. We have used random number generator to determine processing time and release time for every vehicle so that we can obtain arbitrary values for it. The random value range for processing time is from 1 to 6. Random time interval between release-time of two consecutive vehicles has range from 1 to 10. Our purpose of generating sample data is to mimic various scenarios of vehicle formation at an intersection. Following are the scenarios for which we have generated sample data and have performed scheduling on it.

a. Evenly distributed vehicles across all directions. 10 Samples each for 5, 10, 20 and 50 vehicles in each direction.

b. Low number of vehicles in directions turning left or right, high number of vehicles going straight. 10 samples each for min 5, 10 vehicles going straight.

c. Random distribution of vehicles, 10 samples.

Below Chart 1 show execution time taken by scheduling algorithm. Analysis of this data shows that execution time for algorithm increases in linear order with increase in number of vehicles per direction. Moreover the maximum execution time for this algorithm for 50 vehicles per direction, total 600 vehicles, is less than 0.3 seconds. This fulfill the requirement of real time execution condition.



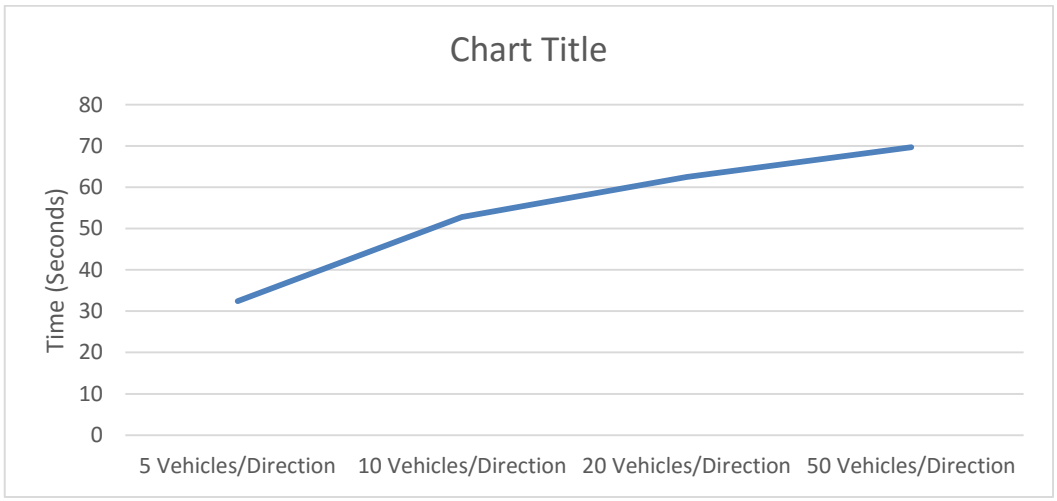
Number Of vehicles	1	2	3	4	5	6	7	8	9	10
5 Vehicles/Direction	180332	170111	170099	170105	170140	170135	155103	170111	170129	165184
10 Vehicles/Direction	180126	200128	180127	180120	180126	180097	210131	180103	190136	175218
20 Vehicles/Direction	210198	205211	205181	195213	179022	190100	190124	185192	180114	195068
50 Vehicles/Direction	230151	220117	240155	220171	230151	235204	240239	210131	230145	220238

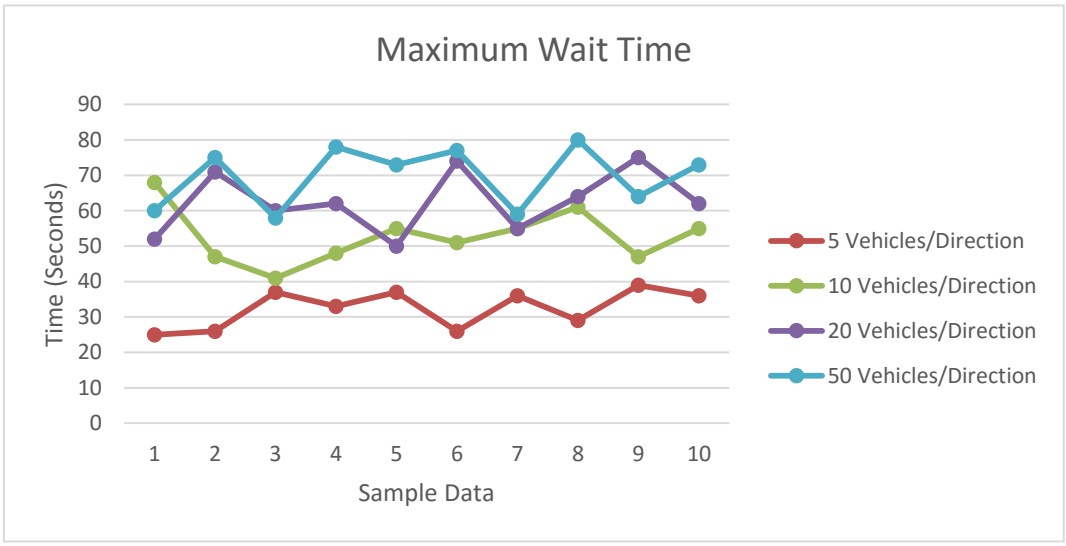
Execution Time

Chart 1 Execution Time for Even Distribution of Vehicles

Below Chart 2 show maximum wait time for even distribution of vehicles at intersection. Analysis of this data shows that value for wait max doesn't correlate with number of vehicles directly. Because the wait max cap is enforced, the maximum wait time value

remains below or very close to this limit. The value of maximum wait time depends on distribution of vehicles and their release time.



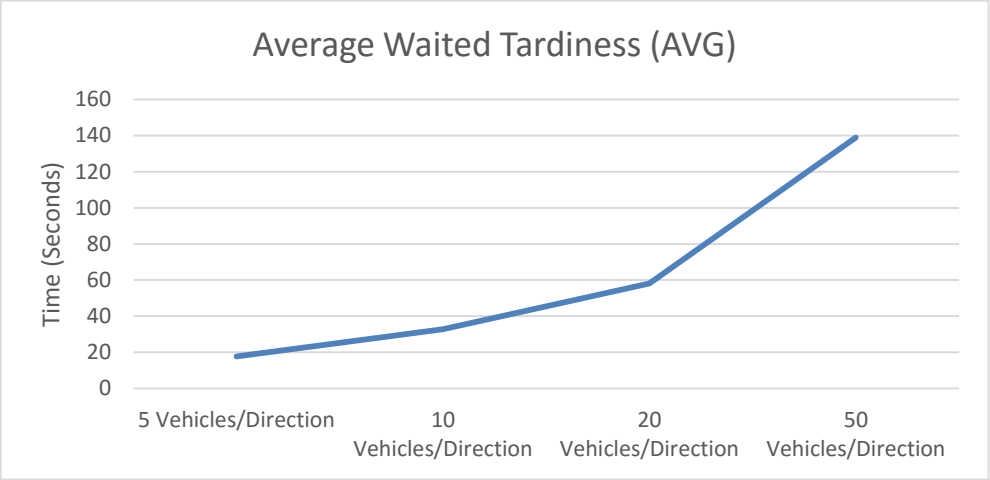


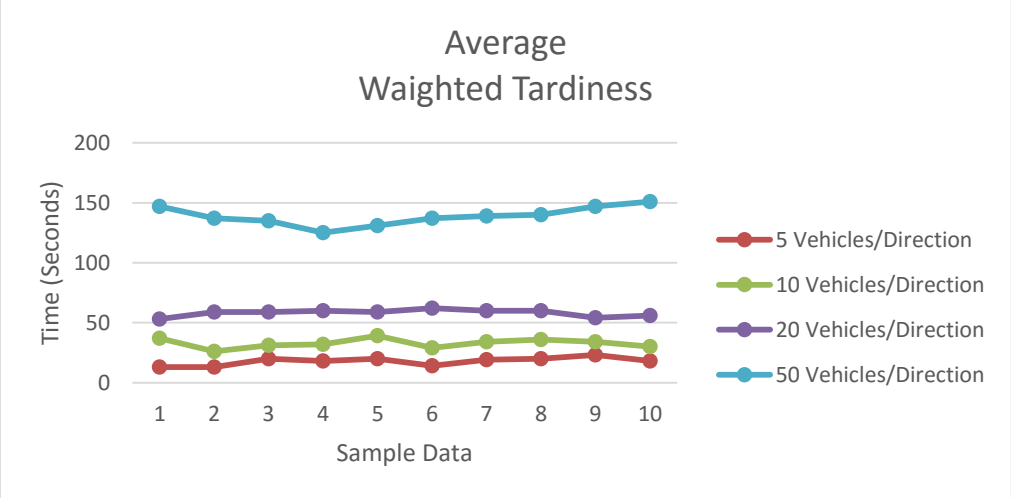
Number Of vehicles	1	2	3	4	5	6	7	8	9	10
5 Vehicles/Direction	25	26	37	33	37	26	36	29	39	36
10 Vehicles/Direction	68	47	41	48	55	51	55	61	47	55
20 Vehicles/Direction	52	71	60	62	50	74	55	64	75	62
50 Vehicles/Direction	60	75	58	78	73	77	59	80	64	73

Maximum Wait Time

Chart 2 Maximum Wait Time for Even Distribution of Vehicles

Below Chart 3 shows average weighted tardiness for evenly distributed vehicles at intersection. It shows that the average weighted tardiness have higher values if number of vehicles increases at the intersection. The average waited tardiness remains in close range for different samples of same series. Hence we can conclude that using this algorithm we can estimate the average tardiness if we know number of vehicles at the intersection.





Number Of vehicles	1	2	3	4	5	6	7	8	9	10
5 Vehicles/Direction	13	13	20	18	20	14	19	20	23	18
10 Vehicles/Direction	37	26	31	32	39	29	34	36	34	30
20 Vehicles/Direction	53	59	59	60	59	62	60	60	54	56
50 Vehicles/Direction	147	137	135	125	131	137	139	140	147	151

Average Weighted Tardiness

Chart 3 Average Weighted Tardiness for Even Distribution of Vehicles

Following charts display statistics for execution time, maximum wait time and average tardiness for uneven distribution of vehicles as well as random distribution of vehicles. Analysis of these chart shows that the values for optimization variables follows the same pattern displayed above.

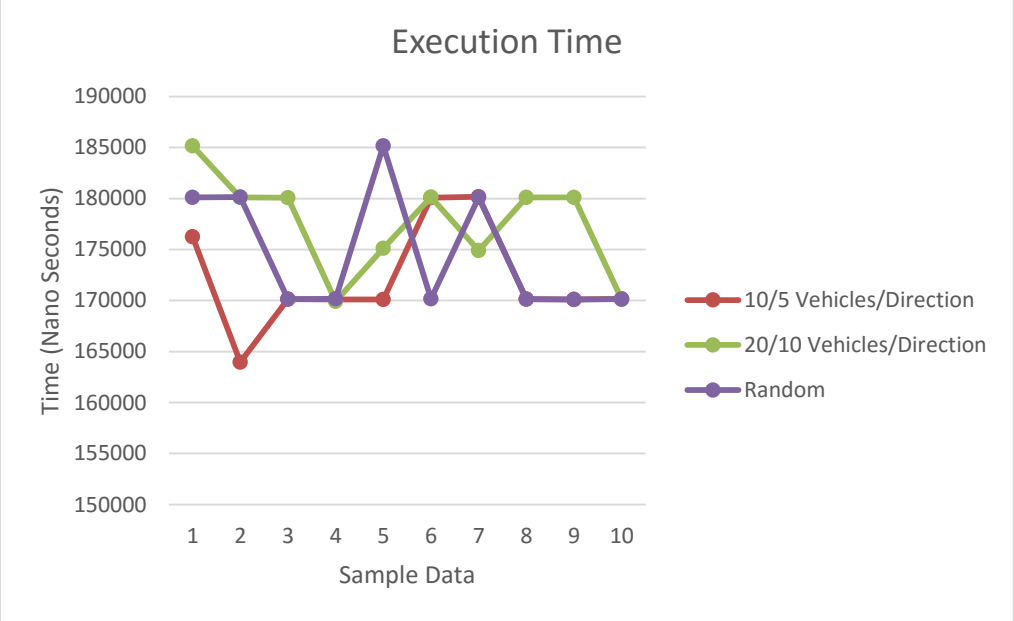


Chart 4 Execution Time for Uneven/Random Distribution of Vehicles

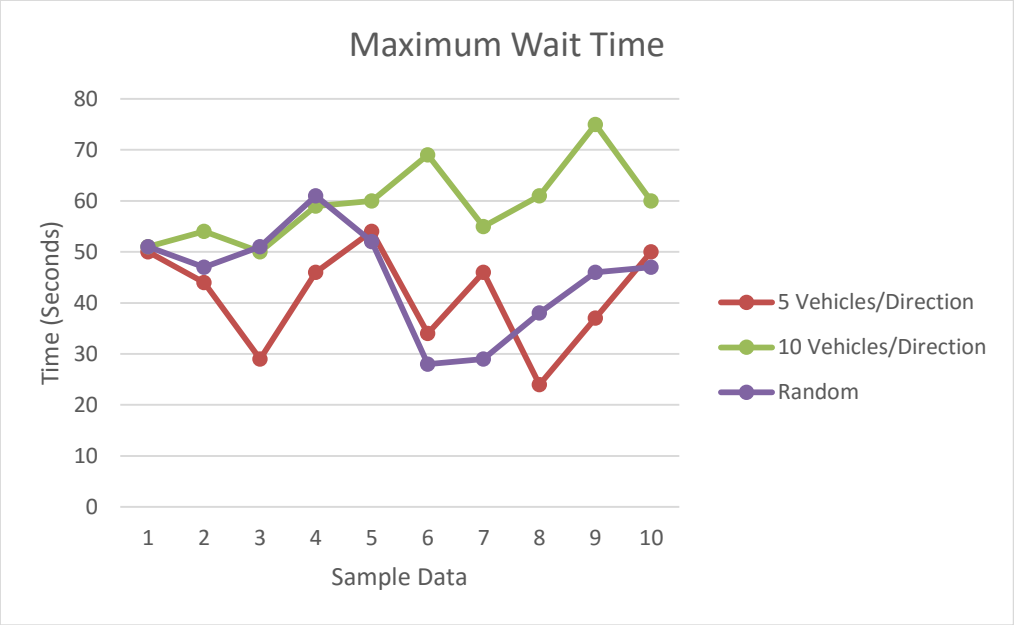


Chart 5 Maximum Wait Time for Uneven/Random Distribution of Vehicles

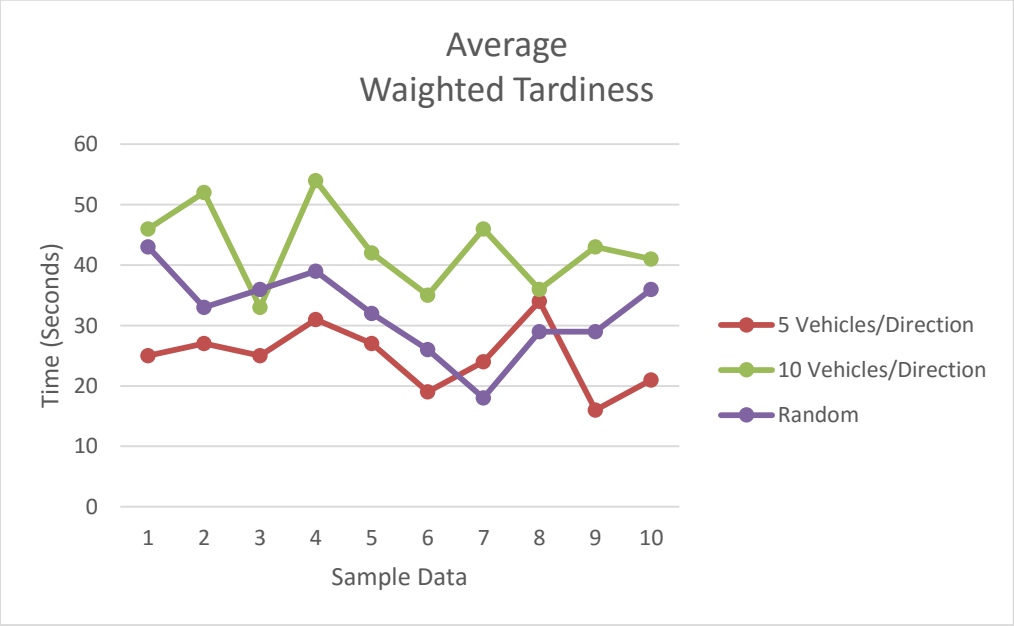


Chart 6 Average Weighted Tardiness for Uneven/Random Distribution of Vehicles

CHAPTER 6

PROOF OF CONCEPT

6.1. For communication protocol

Avoid Deadlock. Every node start transmitting its information on a calculated time slot therefore there is no "wait and hold" scenario. This eliminates a possibility of deadlock.

Avoid Starvation. The base station arranges the car id in chronological order (in order they enters the synchronization space) and the time slot for transmission is calculated based on number of cars in the list and the cars own position in the list therefore each car will have a chance to transmit its information once per cycle.

Avoid Data collision because each car tries to transmit at a unique time slot, probability of them transmitting at the same time is very low. Moreover if two or more cars tries to transmit at the same time, it is been handle by the inbuilt CSMA/CA protocol.

Packet Delivery in time. Below is the statistic of self-sync communication protocol which shows that it would be always able to transmit the contextual information in time.

a. Header of this protocol is only 8 Bytes.

b. The fields other than Decision Schedule cost than 50 Bytes.

c. The Decision Schedule is made of list of vehicles in order they should access the intersection. A vehicle can be identified by its identification number which can be obtain using Combination of junction ID (8Byte) & virtualID (2 Byte).

d. Other fields related to Schedule (Start Time, processing Time, Due Date, Wait Time) are relative to base time, hence can be represented in one to two byte each.

e. Total packet size for transmission depends on number of vehicles in the schedule. If a cell has 200 vehicle at a given point of time then total packet size would be

$$8+ 50 +10*200 + 8 * 200 =3659 \text{ Bytes.} < 4 \text{ KB}$$

f. Total data to be transmitted= 200 * 4 = 800 KB

g. On a straight 500m road, there can be 100 vehicles in a row on single lane (assuming that one vehicle require 5m space including space between two consecutive vehicles.).

h. If we assume there are 3 lanes per side of road, and intersection is at the middle of the 500 meter stretch (so at each side there would be max 50 vehicles.), total number of vehicle at the intersection would be

$$50 * 3 \text{ (lanes)} * 4 \text{ (sides)} = 600 \text{ vehicles}$$

i. Hence in worst case scenario there would be 600 vehicle at an intersection, hence total packet size would be

$$8 + 50 + 10 * 600 + 8 * 600 = 10858 \text{ Bytes} < 11 \text{ KB}$$

j. Total data to be transmitted

$$11 * 600 = 6600\text{KB} < 6.5 \text{ MB}$$

k. Maximum speed of network using DSRC/WAVE protocol is 27Mbps >= 3.3MBps

l. Hence in worst case scenario it will require less than 2 Seconds to transmit data for every vehicles in the vicinity of an Intersection. When an intersection is saturated and have very heavy traffic we expect a bit slower traffic then average speed limit. Hence we conclude that 2 seconds of time is sufficient and safe for transmission of contextual information of all the vehicles.

6.2. Self-Synchronization at One Intersection

Avoid Deadlock. In this issue a deadlock scenario will appear if there are vehicles present on at least two of the conflicting direction and every side selects 0 size batch. We eliminate this using

a. To eliminate the recursive wait, we apply a rule that if at least one vehicle is waiting (Release Time <= Current Time) from at least one non-conflicting direction in a processing group, then the batch size for that group can't be zero.

b. The optimization formula used to calculate batch size from each direction group is defined as equation (10) in section 4.4.5. For any given set of vehicles, if there are vehicles on two conflicting direction, due to existence of set up time, it is not possible to obtain optimal value for equation (10) with 0 batch size on each of the direction.

Avoid Starvation. Implementation of $Wait_{max}$ and $Wait_{max}$ cap for each direction ensures that no vehicle from any direction will have to wait for more than a stipulated time period. Hence it avoids any starvation scenario.

Fairness. The fairness has different meaning in different scenarios in vehicular environment. For example if all the direction have same number of vehicles, wait time should for every direction should be similar. On the other hand if there is huge traffic on one direction whereas moderate or low traffic on other direction then difference in wait time is understandable. Even in the skewed traffic scenario none of the direction should be waiting for too long. The optimization criteria of minimize the average tardiness as well as minimize the maximum wait time, ensures that every vehicle will be provided the intersection access without un-necessary delay.

Feasibility: It is very difficult to find an optimal solution sequence for self-synchronization at one intersection problem. On a 4 way intersection there are 12 conflicting directions. If we assume 2 vehicles are present on each conflicting direction, Total possible permutations would be $(!24 / (!2)^{12}) = 1.5 * 10^{19}$.

We define solution as a feasible solution if it provide conflict free movement of vehicles while maintaining fairness on intersection. Also the solution must be obtained with in a very short time slot (1 – 2 Seconds).

The solution we presented in this dissertation fulfill all of the requirements stated above. Hence we conclude that the solution is feasible.

6.3. Self-Synchronization Through Multiple Intersection

We can achieve nonstop movement of vehicles through multiple intersection if a vehicle can reach the next intersection either before last batch from same direction finishes or when the next batch from same direction at the intersection begins.

We have defined the processing cycle time at every intersection as

$$\min(G_{ij} + L_{ij})$$

Where $(G_{ij} + L_{ij})$ is sum of maximum gain time and lose time for any vehicle between intersections I and J. Let release time for vehicle v at intersection J is T_v and v is moving in direction d. Let the last batch finish time at intersection J for direction d is T_d. Hence T_v falls between end time of last batch from direction d and start time of next batch from same direction.

$$T_d \leq T_v \leq T_d + \min(G_{ij} + L_{ij})$$

If $T_v \leq T_d + G_{ij}$ then $T_v - G_{ij} \leq T_d$

Hence Vehicle v can accelerate and gain enough time to go with current batch.

If $T_v \geq T_d + G_{ij}$ then $T_v + L_{ij} \geq T_d + G_{ij} + +L_{ij}$

Hence vehicle v can slow down and lose enough time so that it reaches the intersection when next batch from direction d is about to process.

Hence proved that if the processing cycle time at every intersection is $\min(G_{ij} + L_{ij})$ then all vehicles can achieve nonstop movement through multiple intersections.

CHAPTER 7

CONCLUSION

The Self Synchronization approach solves the problems of present traffic signal light approach as follows

a. It eliminates any kind of dilemma by having decision of either stop or go.

b. If the intersection is available i.e. no other vehicle is on conflicting side, the vehicle will always pass through without stopping.

c. It eliminates the situation when at the peak hour one side is waiting and there is no vehicle on other side. It schedules the vehicles as soon as intersection is available to access.

d. It saves all the electric power required to keep signals working.

e. It provides more understanding among vehicle drivers passing through an intersection (a better solution for yield, because every vehicles follows a specific command for either stop or go therefore it doesn't require to yield).

f. It reduces average stopping time at junction by auto synchronizing vehicle movement at intersection.

g. It reduces the congestion problem by reducing average stopping time at intersection.

The self-synchronization of moving vehicles has successfully established an innovative method for traffic management and synchronization among vehicles at intersection without the need of an outside entity. It has also provided a way for nonstop movement of vehicles through multiple intersection. For future research topic such as advancement of

algorithm for multiple intersection synchronization and alternative route calculation based on traffic trends can be used.

ANNEXURE A

Table A1 shows conflicting direction on a junction. The true value represent a conflict whereas blank represent no conflict.

Table A1

Matrix of conflicting direction on a 4 way intersection

Direction	(1,2)	(1,3)	(1,4)	(2,1)	(2,3)	(2,4)	(3,1)	(3,2)	(3,4)	(4,1)	(4,2)	(4,3)
(1,2)								True			True	
(1,3)				True	True	True		True			True	True
(1,4)				True		True	True		True		True	True
(2,1)		True	True				True	True		True	True	
(2,3)		True										True
(2,4)		True	True				True	True	True			True
(3,1)			True	True		True				True	True	True
(3,2)	True	True		True		True					True	True
(3,4)			True			True						
(4,1)				True			True					
(4,2)	True	True	True	True			True	True				
(4,3)		True	True	True	True	True	True	True				

For example if a vehicle is going from road 1 towards road 2 (1, 2) then it will not have any conflict with vehicle going from road 2 to road 3 but it will have conflict with vehicle

going from road 4 to road 2. Vehicles from conflicting direction can't use the intersection simultaneously. These type of predefined matrices can provide a tool to identify conflicting vehicle.

REFERENCES

1. A. Ng. C. Allahverdi, T. Cheng, M. Kovalyov. A survey of scheduling problems with setup times or costs, *European Journal of Operational Research*, *187*, 985–1032, 2008.

2. Albert Jones, Luis C. Rabelo. Survey of Job Shop Scheduling Techniques, *NISTIR. Nationall Institute Standards and Technology*, 1998.

3. Alexandra Knaup. Automatically into the parking space, *Mercedez-Benz Automation*, Oct 27, 2014, Retrieved Oct 30, 2015 from Mercedez-Benz site: https://www.mercedes-benz.com/en/mercedes-benz/next/automation/automatically-into-the-parking-space/

4. Ari Hottinen, Olav Tirkkonen, Risto Wichman. *Multi-Antenna Transceiver Techniques for 3G and beyond*, John Wiley & Sons Ltd., 2003.

5. Bilal Ahmed Khan, Nai Shyan Lai, An Intelligent Traffic Controller Based On Fuzzy Logic, in *The First International Conference on Green Computing, Technology and Innovation (ICGCTI2013)*, The Society of Digital Information and Wireless Communication, 2013

6. C L Liu, James W Layland. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, in *Journal of the Assomation for Computing Machinery, 20 (I)*, 1973

7. Chinyao Low, Yuling Yeh. Genetic Algorithm-Based Heuristics for Open Shop Scheduling Problems with Setup, Processing and Removal times Separated, *Robotics and Computer-Integrated Manufacturing 25,* 314-322, 2009.

8. Debra J Hoitomt, Peter B Luh, Krishna R Pattipati. A Practical Approach to Job-Shop Scheduling Problem, *IEEE transactions on Robotics and Automation*, 9(1), 1993.

9. E. L. Lawler. On Scheduling Problems With Deferral Costs, *Management Science 11(2)*, 280-288, 1964

10. E. L. Lawler, J. K. Lenstra, A. H. R. Kan, D. B. Shmoys. Sequencing and scheduling: Algorithms and complexity, *Handbooks in operations research and management science, 4*, 445-522, 1993.

11. Erico Guizzo. How Google's Self-Driving Car Works, *IEE Spectrum*, Retrieved Oct 30, 2015 from IEEE: http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

12. European Telecommunications Standards Institute, ETSI EN 302 571 (V1.1.1), 2008-2009

13. Federal Communication Commission: Code of Federal Regulations, Title 47, Office of the Federal Register National Archives and Records Administration, Oct 2010.

14. Ford Motors :Media News, Dearbrn, Mich., June 26, 2012, Retrieved Oct 30, 2015, from Ford Motors official media news: https://media.ford.com/content/fordmedia/fna/us/en/news/2012/06/26/ford-developing-new-technologies-to-help-ease-traffic--parking-s.html

15. Ford Motors :Media News, Dearbrn, Mich., Dec 12, 2013, Retrieved Oct 30, 2015, from Ford Motors official media news: http://media.ford.com/content/fordmedia/fna/us/en/news/2013/12/12/ford-test-car-that-automatically-steers-around-stopped-or-slowin.pdf

16. Francis Sourd. Earliness–tardiness scheduling with setup considerations, *Computers & operations research, 32(7)*, 1849-1865, 2005.

17. G. F. Ali Ahammed, Reshma Banu. Analyzing the performance of Active Queue Management Systems, *International Journal of Computer Networks and Communications (IJCNC), 2 (2)*, 2010.

18. Gary Long, Acceleration Characteristics of Starting Vehicles, in Transportation Research Board 79th Annual Meeting (Washington, DC, 2000), Transportation Research Board, Paper No. 00-0980.

19. Google Official Blog: Blog News, Aug 7, 2012, Retrieved Oct 30, 2015, from Google Blog: https://googleblog.blogspot.hu/2012/08/the-self-driving-car-logs-more-miles-on.html

20. H. K. Heidarsson, G. Sukhatme. Obstacle Detection From Overhead Imagery Using Self-Supervised Learning For Autonomous Surface Vehicles, In *IEEE/RSJ International Conference* on *Intelligent Robots and Systems,*(Sept. 2011), vol., no., pp.3160,3165,
25-30, doi: 10.1109/IROS.2011.6094610

21. IEEE Computer Society, Part 11 Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Amendment 6- Wireless Access in Vehicular Environment, in IEEE Standards For Information Technology- Telecommunication and Information Exchange Between Systems Local and Metropolitan Area Networks Specific Requirements, IEEE Standards 802.11p™ – 2010, New York, 2010.

22. IEEE Computer Society, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)— Multi-channel Operation, IEEE Std 1609.4™_2010 (Revision of IEEE Std 1609.4-2006), New York, 2011

23. International Telecommunication Union, Radio Regulation Articles Edition 2004, Article 5

24. Japan—MIC Equipment Ordinance (EO) for Regulating Radio Equipment Articles 7, 49.20, 49.21a.

25. Joseph Y-T. Leung, Gilbert H. Young. Minimizing Total Tardiness on a Single Machine With Precedence Constraints, *ORSA Journal on Computing, 2(4)*, 346-352, 1990.

26. Jan Karel Lenstra, A H G Rinnooy Kan, Peter Brucker. Complexity of machine scheduling problems, *Annals of discrete mathematics, 1*, 343-362, 1977.

27. M. Debes, A. Lewandowska, J. Seitz. Definition and Implementation of Context Information, in *Proc. of the 2nd workshop on positioning, navigation and communication (WPNC'05) & 1st ultra-wideband expert talk (UET'05), pp 63-68, 2005*

28. M. R. Garey. The Complexity of Flowshop and Jobshop Scheduling, *Mathematics of Operations Research*, 1 (2), 117–129, 1976

29. Mark Whittington. Law Proposed in Texas to Require Licensed Driver in Self-Driving Vehicles, *Yahoo! News*, Fri, Mar 8, 2013, Retrieved Oct 30, 2015, from Yahoo news site:

30. Md. Imrul Hassan, Hai L. Vu, Taka Sakurai, Performance Analysis of the IEEE 802.11 MAC Protocol for DSRC Safety Applications, *IEEE Transactions On Vehicular Technology, Vol. 60, No. 8*, 2011

31. Mercedez-Benz: Media Article, Retrieved Oct 30, 2015, from Mercedez-Benz Innovation:
https://www.mercedes-benz.com/en/mercedes-benz/innovation/with-intelligent-drive-more-comfort-in-road-traffic/

32. Mercedez-Benz: Media Article, Retrieved Oct 30, 2015, from Mercedez-Benz Innovation:
https://www.mercedes-benz.com/en/mercedes-benz/innovation/ces-2015/

33. Mercedez-Benz: Media Article, Retrieved Oct 30, 2015, from Mercedez-Benz Innovation:

https://www.mercedes-benz.com/en/mercedes-benz/innovation/autonomous-long-distance-drive/

34. Muller. With Driverless Cars, Once Again It Is California Leading The Way, *Forbes.com*, September 26, 2012, Retrieved Oct 30, 2015, from Forbs official site: http://www.forbes.com/sites/joannmuller/2012/09/26/with-driverless-cars-once-again-it-is-california-leading-the-way/

35. Michael L. Pinedo. Scheduling: Theory Algorithms, and Systems, Springer Science & Business Media, 2012.

36. R. Graham. Bounds for certain multiprocessing anomalies, *Bell System Technical Journal, 45*, 1563–1581, 1966.

37. R. L. Graham, E. L. Lawler, J. K. Lenstra, Rinnooy Kan. Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey. in *Proceedings of the Advanced Research Institute on Discrete Optimization and Systems Applications of the Systems Science Panel of NATO and of the Discrete Optimization Symposium. Elsevier, (5)*, 287–326, 1979.

38. Safety and Mobility Test field Germany: Project Details, 2011, Retrieved Oct 30, 2015, from, SIM TD web site: http://simtd.de/index.dhtml/enEN/index.html

39. USDOT pilot safety program: Connected Vehicle Safety Pilot, 2015, Retrieved Oct 30, 2015, from Office of the Assistant secretary for research and technology, Intelligent Transportation System Joint Program Office: http://www.its.dot.gov/safety_pilot/index.htm

40. US Department of Transportation (NHTSA), Crash Factors in Intersection-Related Crashes: An On-Scene Perspective, DOT HS 811 366, Sep 2010.

41. USDOT Safety Pilot Model Deployment: About, 2012, Retrieved Oct 30, 2015, from UMTRI safety pilot program official web location: http://safetypilot.umtri.umich.edu/index.php

42. Wall Matthew Bartschi. A genetic algorithm for resource-constrained scheduling, Diss. Massachusetts Institute of Technology, 1996.

43. V. Roshanaei a, M.M. Seyyed Esfehani a, M. Zandieh. Integrating non-preemptive open shops scheduling with sequence-dependent setup times using advanced metaheuristics, *Expert Systems with Applications, 37,* 259–266, 2010

44. Y. J. Li. An overview of the DSRC/WAVE technology. In *Quality, Reliability, Security and Robustness in Heterogeneous Networks*, (2012), Springer Berlin Heidelberg, 544-558.