Downlink Traffic Scheduling in Green Vehicular Roadside Infrastructure

DOWNLINK TRAFFIC SCHEDULING IN GREEN VEHICULAR ROADSIDE INFRASTRUCTURE

BY

ABDULLA A. HAMMAD, M.Sc.

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Doctor of Philosophy (2014) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Downlink Traffic Scheduling in Green Vehicular Roadside Infrastructure
AUTHOR:	Abdulla A. Hammad M.Sc., (Computer Engineering) Cairo University, Giza, Egypt
SUPERVISOR:	Dr. Terence D. Todd
NUMBER OF PAGES:	xv, 124

Abstract

Vehicular Ad-hoc Networks (VANETs) will be an integral part of future Intelligent Transportation Systems (ITS). In highway settings where electrical power connections may not be available, road-side infrastructure will often be powered by renewable energy sources, such as solar power. For this reason, energy efficient designs are desirable.

This thesis considers the problem of energy efficient downlink scheduling for road-side infrastructure. In the first part of the thesis, the constant bit rate (CBR) air interface case is investigated. Packet-based and timeslot-based scheduling models for the theoretical minimum energy bound are considered. Timeslot-based scheduling is then formulated as a Mixed Integer Linear Program (MILP). Following this, three energy efficient online scheduling algorithms with varying complexity are introduced. Results from a variety of experiments show that the proposed scheduling algorithms perform well when compared to the energy lower bounds.

In the second part of the thesis, the variable bit rate (VBR) air interface option is considered. Offline scheduling formulations are derived that provide lower bounds on the energy required to fulfill vehicle requests. An integer linear program (ILP) is introduced which can be solved to find optimal offline VBR schedules. Two flow graph based models are then introduced. The first uses Generalized Flow (GF) graphs and the second uses time expanded graphs (TEGs) to model the scheduling problem. Four online scheduling algorithms with

varying energy efficiency, fairness and computational complexities are developed. The proposed algorithms' performance is examined under different traffic scenarios and they are found to perform well compared to the lower bound.

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my advisor Dr. Terence D. Todd for his continuous support of my Ph.D. study and research, for his patience, motivation, and immense knowledge. His guidance helped me with my research and the writing of this thesis. His insight and the critical thought provoking discussions we had were a huge influence on my way of thinking and his attention to detail showed me how quality work is crafted. Besides my advisor, I would like to thank the rest of my supervisory committee and the members of the examining committee for reading my thesis and for their valuable suggestions and comments. I thank my fellow members of the Wireless Networking Group for their support and comments. I would like to thank my mother for her continuous support throughout these years.

Last but not least, I would like to thank my wife and my two beautiful sons for their patience, understanding and for putting up with late working nights. It would have been impossible without you.

Abbreviations

AIFS Arbitrary Inter-Frame Space

AQPS Asynchronous Quorum-based Power Saving

ATIM Announcement Traffic Indication Message

AW ATIM window

BI Beacon Interval

CBR Constant Bitrate

CCH Control Channel

CW Contention Window

DHT Distributed Hash Table

DOT Department of Transportation

DSRC Dedicated Short Range Communication

EDCA Enhanced Distributed Channel Access

FCFS First Come First Serve

FDF First Deadline First

FF Fastest First

G-GF Greedy-Generalized Flow

G-TEG Greedy-Time Expanded Graph

GF Generalized Flow

GMCF Greedy Minimum Cost Flow

ILP Integer Linear Program

IP Integer Program

ITS Intelligent Transportation System

ITSA Intelligent Transportation Society of America

IVHS Intelligent Vehicle Highway System

LLC Link Layer Control

LSF Least Selected First

LTSF Longest Total Stretch First

LWT Longest Wait Time

MANET Mobile Adhoc Network

MILP Mixed Integer Linear Program

MLME MAC Layer Management Entity

MQIF Maximum Quality Increment First

MRF Most Request First

MSDU MAC Service Data Unit

NFS Nearest Fastest Set

OBU Onboard Unit

PLME Physical Layer Management Entity

PS Power Save

RR Round Robin

RSSI Received Signal Strength Indicator

RSU Roadside Unit

RWP Random Way Point

SCH Service Channel

SJF Short Job First

SRM Space Reliability Metric

SS Static Scheduler

TBTT Target Beacon Transmission Time

TEG Time Expanded Graph

TF Time Framed

TRM Time Reliability Metric

TS Time Slotted

V2I Vehicle to Infrastructure

V2V Vehicle to Vehicle

VANET Vehicular Adhoc Network

VBR Variable Bitrate

WAVE Wireless Access in Vehicular Environment

WBSS Wave Basic Service Set

WME Wave Management Environment

WSM WAVE Short Message

WSMP Wave Short Message Protocol

WSN Wireless Sensor Network

WWAN Wireless Wide Area Network

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

In the future, Vehicular Ad-hoc Networks (VANET) will play an increasingly important role in the operation of Intelligent Transportation Systems (ITS). They will enable a myriad of new applications, which include categories such as safety, traffic management and infotainment. In order to support these types of applications, VANETs can be formed using vehicle to vehicle (V2V), vehicle to infrastructure (V2I), or a combination of both modes of communications. In V2I communications, Roadside Units (RSUs) are fixed infrastructure which can store and relay locally relevant data and provide backhaul connectivity to other networks, such as the Internet. They also enable VANETs to meet the QoS requirements of certain applications, such as in situations where vehicle density is low and V2V communications cannot maintain full connectivity.

In many RSU deployments, wired electricity will be difficult to provide, and RSUs will often be powered by renewable energy sources such as solar and wind power. The cost associated with renewable energy solutions however, is a strong function of the energy

needs of the supplied device, which motivates the need for energy efficient RSU designs. This is the motivation behind this thesis.

1.2 Downlink Traffic Scheduling in Green Vehicular Road-side Infrastructure

This thesis focuses on packet scheduling with energy optimization as a goal. In certain scenarios, vehicle locations can be predicted with a high degree of accuracy, and this information can be used to reduce downlink infrastructure-to-vehicle communication energy usage. Two types of air interface assumptions are considered, namely, constant bit rate (CBR) and variable bit rate (VBR). The CBR approach uses variable transmission power to achieve a constant bit rate over the vehicle to infrastructure links. The VBR option uses constant transmission power, but varies the bit rate to accommodate changing link conditions.

In the first part of the thesis, the CBR case is studied. Offline scheduling algorithms are derived which provide lower bounds on the energy needed to satisfy arriving vehicular communication requirements. The scheduling complexity for both packet and slot based models is investigated. We show that the packet-based scheduling case can be formulated as a generalization of the classical single-machine job scheduling problem with a tardiness penalty, which is referred to as α Earliness Tardiness. A proof is given that shows that even under simple assumptions, the problem is NP-complete. We then focus on slot based scheduling. We formulate this problem as a Mixed-Integer Linear Program (MILP) that is shown to be solvable in polynomial time using a proposed minimum cost flow graph construction. Three energy-efficient online traffic scheduling algorithms are then introduced

for common vehicular scenarios. Results from a variety of experiments show that the proposed scheduling algorithms perform well when compared with the energy lower bounds in vehicular situations where path loss has a dominant deterministic component so that energy costs can be estimated. We also show that near-optimal results are possible but come with increased computation times compared with our heuristic algorithms.

In the second part of the thesis, the VBR air interface option is studied. We first formulate the scheduling problem as an Integer Program (IP) optimization. We then prove that even under a simple distance-dependent exponential radio path loss assumption, the problem is NP-hard using a reducing to the well known Santa Claus Problem. Two graph models are then introduced as an efficient approximation to the scheduling problem. The Generalized Flow (GF) graph model represents time slots as individual nodes while the Time Expanded Graph (TEG) model employs the concept of time frames. We then introduce four energy efficient online scheduling algorithms. Results from a variety of experiments show that the proposed algorithms perform well compared to the energy lower bound. It also shows that less computationally intensive algorithms can perform well under light load, but more complex algorithms can provide near optimal throughput and minimize packet dropping rates.

1.3 Thesis Organization

Chapter 2 introduces background information related to the focus of this thesis. VANET application classification, examples and requirements are presented. The chapter examines the specific characteristics that define VANETs and standardization efforts are reviewed. A comprehensive review of the research efforts related to VANET scheduling, with emphasis on energy efficiency, is presented.

In Chapter 3, the scheduling problem for RSUs with variable transmission power is presented, i.e., the constant bit rate (CBR) case. Theoretical bounds for packet based and slot based scheduling are formulated and three online scheduling algorithms are proposed. The first, Greedy Minimum Cost Flow (GMCF), is motivated by a minimum cost flow graph formulation. The other two algorithms have reduced complexity compared with GMCF. The Nearest Fastest Set (NFS) scheduler uses vehicle location and velocity inputs to dynamically schedule communication activity. The Static Scheduler (SS) performs the same task using a simple position-based weighting function. Results show the viability of the proposed algorithms and the performance tradeoffs encountered with the choice of each.

Chapter 4 investigates the case where the RSU transmission power is constant, resulting in variable bit rate (VBR) links. An offline Integer Program (IP) formulation is presented and the NP-hardness of the problem is proven. Two novel graph theoretic models are introduced to solve the offline bound model and four online algorithms are proposed. The first of these algorithms, First Come First Serve (FCFS), is based on treating all vehicles equally and in order of arrival. The second, Fastest First (FF), is designed to increase fairness between fast and slow moving vehicles. Greedy Generalized Flow (G-GF) and Greedy Time Expanded Graph (G-TEG) are online implementations of the two graph centric models. Representative results for the different algorithms are presented and recommendations are proposed for choosing between them depending on the traffic mix.

The thesis is then concluded in Chapter 5 with suggestions for possible future work.

Chapter 2

Background

2.1 Introduction

The number of cars in operation has now surpassed the one billion mark globally, and innovations in vehicle safety and efficiency can have a huge economic and environmental impact. Cars are increasingly equipped with sensory systems to assist in accident prevention, but cooperation between vehicles is still very primitive. Wide area communication systems such as WiMax and cellular networks can be used to facilitate cooperation between vehicles, but low data dissemination rates and high cost hinder their widespread use. On the other hand, systems based on local ad hoc networking created between vehicles and roadside infrastructure provide a more feasible approach in terms of latency, cost, scalability and locally relevant information. As a result, the VANET research field has received increasing attention over the last few years.

This chapter presents an overview of previous work related to scheduling in VANETs. It starts by discussing a wide variety of VANET applications and their diverse requirements.

We then explain the unique characteristics of VANETs that restricts the direct use of solutions that have been developed for mobile ad hoc networks (MANETs). The chapter then presents information relating to standardization efforts, such as the WAVE protocol stack. Classical scheduling approaches are reviewed with an emphasis on RSU scheduling with energy efficiency in mind.

2.2 VANET Applications

There have been many applications proposed for VANETs by standardization organizations, researchers and the automotive industry (Cheng *et al.*, 2011) (Karagiannis *et al.*, 2011). These applications generally fall into three categories: active road safety, traffic management and infotainment.

Active road safety applications aim to decrease the probability of traffic accidents. This is done by sharing information between vehicles, and between vehicles and RSUs. The type of information that is relevant may contain velocity, position, distance to intersection, and hazardous vehicle behavior. This information would be processed to predict collisions and dangerous locations, such as potholes and slippery surfaces. Examples of these types of applications include: Intersection collision warning, lane change assistance, overtake warning, head on/rear end warning, cooperative forward collision warning, emergency vehicle warning, cooperative merge assist, emergency e-brake light, stationary vehicle warning, signal violation warning, hazardous location notification and control loss warning. The common requirements for these types of applications are usually short and frequent message exchanges and low critical latency (less than 100 ms).

Traffic monitoring and management applications aim to improve traffic flow in road networks. They alleviate the need to install expensive legacy hardware such as loop sensors,

and instead, rely on Onboard Units (OBU) to deliver localized information relating to traffic conditions. After this information is processed, adjustments can be made to improve traffic flow and to prevent future deadlocks. This can be done, for example, by varying traffic light duration throughout a city road network. It can also be used to suggest the optimal driving speed such that fuel efficiency and driving experience is improved. Moreover, this data can help with cooperative navigation, where vehicles can be grouped into platoons and traffic can be distributed across different routes to improve traffic flow. QoS requirements for these types of applications are more relaxed than in the safety application case.

The third category is infotainment applications, which can either rely on local or global information. Examples of these are, local point of interest notifications, local commerce advertisements and free parking spots or global internet applications such as video on demand, browsing or VoIP.

Although WiMax or cellular networks can be used to support VANET applications, due to the high cost and low speed of data dissemination through wireless wide area networks (WWANs) (Liang and Zhuang, 2011), data dissemination based on RSUs provides a more feasible approach in terms of latency, cost, scalability and locally relevant information. Moreover, there are certain types of applications that depend specifically on RSUs. For example (Lochert *et al.*, 2008) proposes building a city-wide map by sending traffic information in each area to RSUs which in turn, cooperate to disseminate information. Reference (Mershad *et al.*, 2011) proposes routing packets via RSUs by using a distributed hash table (DHT). The authors in (Festag *et al.*, 2008) propose connecting RSUs to Wireless Sensor Networks (WSNs) to continuously send sensor readings to the RSUs. Other examples of applications enabled by RSUs are parking, toll, local ads and drive through Internet.

The effects of VANET protocols extend beyond the applications they enable. With gas prices and carbon emissions on the rise, efficient utilization of the road network becomes more important. As car manufacturers introduce more fuel efficient models such as hybrid and electric cars, improving VANET protocols with the intent of reducing CO_2 is becoming the focus of more research. An example of this type of application is engine start/stop, controlled by the RSU installed at traffic lights. Another example is suggesting travel speed to minimize acceleration and deacceleration in response to traffic conditions. Moreover, a group of vehicles can be organized to travel together and create a platoon, further reducing fuel consumption. Reference (Alsabaan *et al.*, 2012) examines the environmental effect of utilizing VANETs from different perspectives. It introduces "green measures" to compare different communication protocols such as fuel consumption and emissions and suggests new research directions to enhance VANET effects on the environment.

2.3 MANETs vs. VANETs

Mobile Adhoc Networks (MANETs) are created from self organizing mobile or stationary nodes. They usually do not have centralized control and deliver messages using multi-hop routing protocols. When the nodes in this type of network become vehicles, the network becomes a VANET. Although similar to MANETs in some respects, it has been shown (Blum *et al.*, 2004) that the direct use of many MANET protocols for VANETs is inefficient. In order to satisfy different requirements imposed by different application categories, the efficient use of VANET spectrum is required. This requires novel approaches to interference control, call admission control, medium access control, scheduling and QoS guarantees. Below are some unique characteristics (Dahiya and Chauhan, 2010) that create these research challenges.

- Large numbers of nodes: A high percentage of vehicles will eventually be equipped with OBUs. For this reason VANET designs will have to be scalable and able to handle these large numbers.

- Strict QoS constraints: VANET applications, especially when safety related, can require strong guarantees for data delay and reliability. Moreover, a vehicle passing by an RSU may spend very little time in its range, which requires efficient communication protocols.

- Rapid information change: traffic conditions and sensory data (such as parking information) can become obsolete in a very short time and can become dangerous in some safety applications if it is delivered past a certain time deadline.

- Mobility Models: Popular MANET mobility models, such as Random Way Point (RWP), have limited application in VANETs as vehicles are restricted to following the road network. Better models have been developed that take into account vehicle density as a function of the time of day, vehicle-to-vehicle interaction, traffic light effects, points of attraction and other factors that are unique to vehicle flow modeling.

- Geographical casting: In addition to unicasting and broadcasting, geocasting is an important means of communication that vehicles can use to distribute locally relevant data.

- Frequent network disconnection: Due to high vehicle speed, the effects of traffic lights, the difference in velocity and headings, VANETs experience high network disconnection. This can render some routes obsolete by the time they are established (Blum *et al.*, 2004).

- Communication range: Typical MANET range would be less that 100 meters but VANET range can be up to 1 KM.

- Different types of communication environments: Vehicles travel in both city and highway scenarios. The city is characterized by shadowing effects of buildings, high vehicle and RSU density but also suffers from rapid topology fragmentation and change. The highway scenario is simpler with less obstacles and less vehicle to vehicle interaction. Vehicles tend to travel at steady velocity but vehicle densities are less than in the city which introduces different types of challenges.

- Storage and energy: The OBU can store huge amounts of data and have a virtually uninterrupted amount of energy, since the radio uses a small percentage of the available vehicle electrical energy compared to other subsystems. On the other hand, if the RSU is not connected to wired electricity as in the case of highway environments, energy can be limited.

- Interaction with on-board sensors: each VANET OBU can utilize the array of sensors available in modern vehicles such as GPS, proximity sensors and cameras.

- Confidentiality of information and privacy: In safety applications, information is available to all vehicles with very little restriction, while in many other non life threatening applications, information should be confidential unless required by law. The identity of the vehicle passenger, velocity and destination has to be kept private.

2.4 VANET Standardization Efforts

Several standards and frameworks have been proposed for VANETs in Europe, Japan and the United States (Karagiannis *et al.*, 2011). In this section we focus on the efforts in the US, specifically the IEEE WAVE framework (Uzcategui and Acosta-Marum, 2009).

In 1991, the US Congress ordered the creation of the Intelligent Vehicle Highway System (IVHS) program with the goals of increasing safety, alleviating congestion, and making transportation more fuel efficient. To achieve this, the Department of Transportation (DOT) and other organizations created a framework for what is now known as the Intelligent Transportation System (ITS). In 1999 the FCC allocated 75 MHz of spectrum from 5.85 to 5.925 GHz for what is now known as Dedicated Short Range Communication (DSRC). The Intelligent Transportation Society of America (ITSA) suggested using a single standard for the PHY and the MAC layers based on IEEE 802.11 and it was approved by the FCC in 2004. In order to achieve this, IEEE Task Group p amended the IEEE 802.11 standard to be used in the vehicular environment. Meanwhile, a separate IEEE team called Working Group 1609 was responsible for developing the additional layers that would be required for the protocol suite. This resulted in the IEEE 1609.1, 1609.2, 1609.3 and 1609.4 standards. The collection of 802.11p and 1609.x are called Wireless Access in Vehicular Environment (WAVE).

The components of the WAVE system are shown in Figure 2.1. They consist of RSUs which can be mounted on traffic light poles or buildings, and OBUs which are installed in vehicles and can communicate while moving. The WAVE units act separately and can communicate over a default radio channel called the control channel (CCH). The WAVE units can also form communication groups called WAVE basic service sets (WBSS). The members of a WBSS exchange information over one of several radio channels called Service

Channels (SCH) and can also connect to wide area networks.

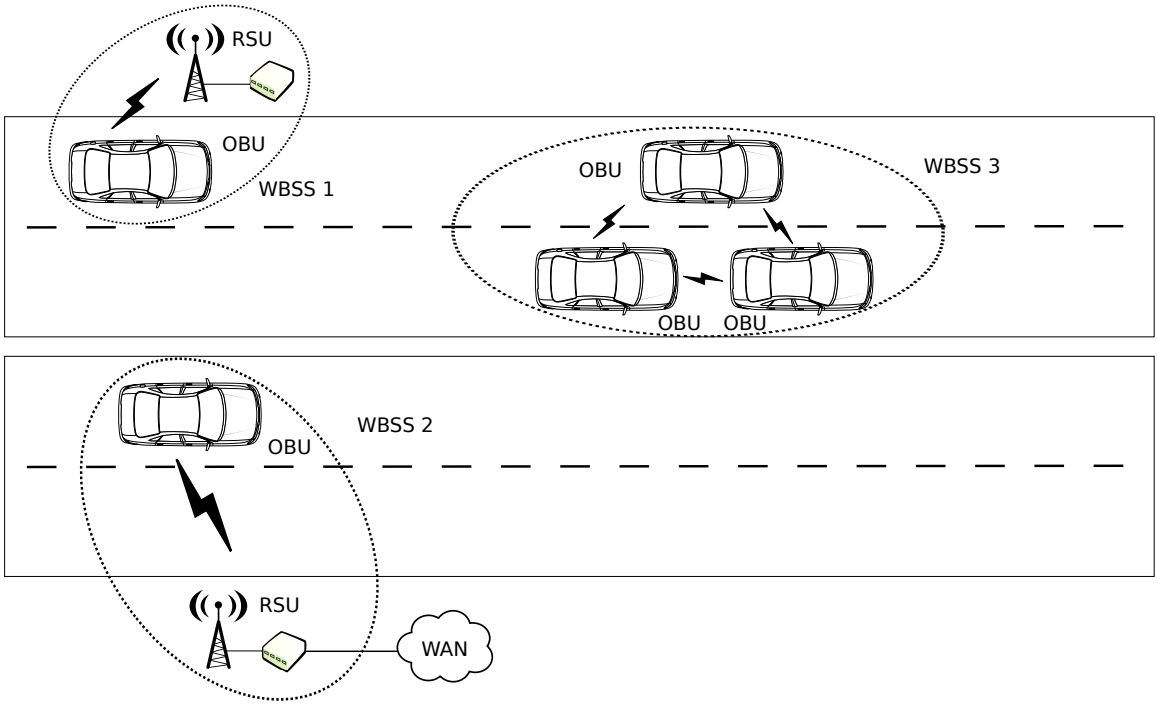

Figure 2.1: WAVE System Components

Two protocol stacks are defined in the WAVE architecture, and share the first two layers: physical and data link. They consist of a traditional Internet Protocol version 6 (IPv6) and a new protocol designed specifically for WAVE called the WAVE Short Message Protocol (WSMP). Using two protocols allows WAVE to be suitable for high priority time sensitive communication while accommodating traditional data exchange. The WAVE communication stack is shown in Figure 2.2. When compared to the ISO model, WAVE does not include session or presentation layers but includes two other layers (resource management and security services) that do not have counterparts in the ISO model. We will explain the layers briefly, before taking a closer look at the PHY and MAC layers. WAVE Layers 1 and

2 are built upon the IEEE 802.11 standard. Due to the differences introduced by the vehicular environment as explained above, the IEEE 802.11p standard amendment was created. It specifies the physical layer management entity (PLME) and the MAC layer management entity (MLME).

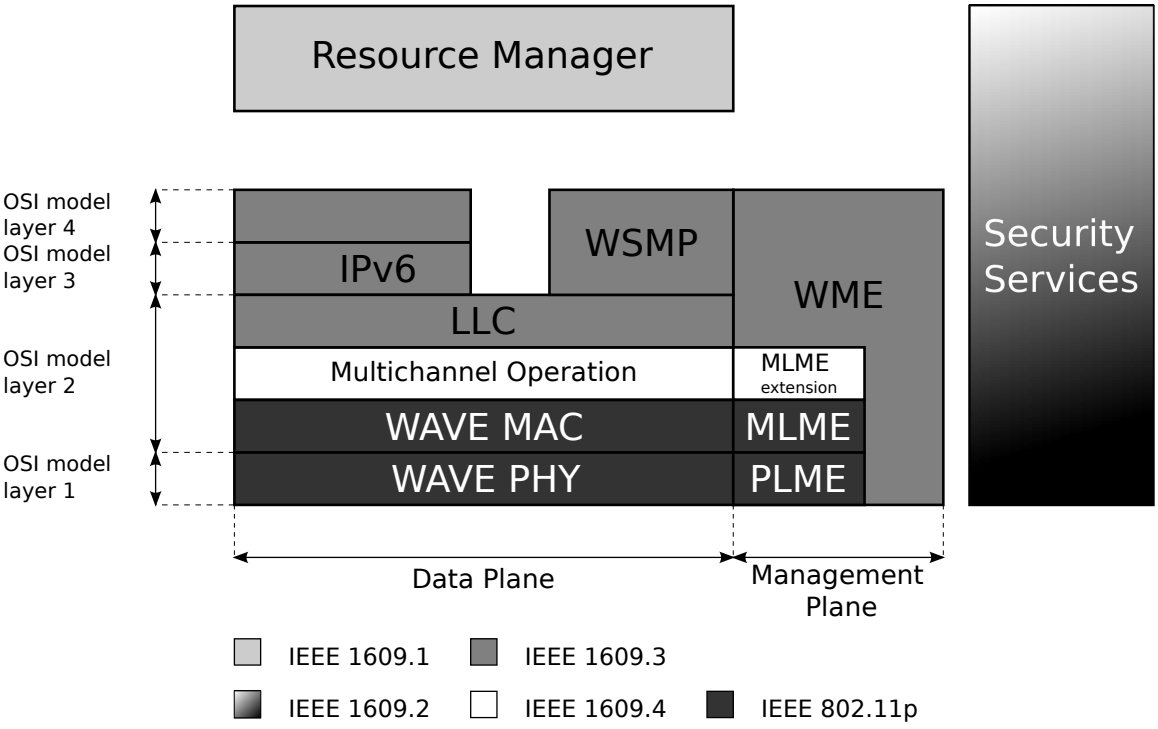

Figure 2.2: WAVE Stack

One of the unique features of the WAVE units is that they alternate between the control channel (CCH) and the service channel (SCH). This created the need for a sub-layer specified by 1609.4 that controls this multichannel operation. Layer 1609.3 is equivalent to ISO layers 3 and 4, i.e., it specifies WSMP and how to include IPv6 and TCP/UDP. In addition to these, it also defines a WAVE management entity (WME) that implements networking services. Resource manager and security services blocks do not have an ISO equivalent and are outside the current scope. In the following discussion, we will take a closer look at

the operations of layers 1 and 2.

2.4.1 MAC and PHY layers

The first two layers of the WAVE standard are based on the changes made to IEEE 802.11a that create IEEE 802.11p. This was done due to the existing stability of the IEEE 802.11a standard, which will make building compatible devices faster and more economical. The need for a different standard stems from the differences in the operating environment, i.e., longer radio ranges, vehicle speed, extreme multipath environments, and overlapping VANETs, and the fact that the application requirements are different. Some of these changes are listed in the Table 2.1 (Müller, 2009).

Parameters	IEEE 802.11a	IEEE 802.11p	Changes
Bit rate (Mbit/s)	6, 9, 12, 18, 24, 36, 48, 54	3, 4.5, 6, 9, 12, 18, 24, 27	Half
Modulation mode	BPSK,QPSK, 16QAM, 64QAM	BPSK,QPSK, 16QAM, 64QAM	No change
Code rate	1/2, 2/3, 3/4	1/2, 2/3, 3/4	No change
Number of subcarriers	52	52	No change
Symbol duration	4 μs	8 μs	Double
Guard time	0.8 μs	1.6 μs	Double
FFT period	3.2 μs	6.4 μs	Double
Preamble duration	16 μs	32 μs	Double
Subcarrier spacing	0.3125 MHz	0.15625 MHz	Half

Table 2.1: Comparison of The Physical Layers in IEEE 802.11a and IEEE 802.11p

In IEEE 802.11p, the communication channel width is 10 MHz, which is half of that defined for IEEE 802.11a in order to make received signals more robust against fading. This 10 MHz channel should allow a 3 to 27 Mb/s bit rate, and up to 54 Mb/s if it uses the optional 20 MHz channel. It also introduced a random MAC addressing mechanism and

requires a high accuracy for Received Signal Strength Indicator (RSSI) values. Additional priority and power control are included for QoS and interference control.

WAVE enabled devices must support a control channel (CCH) and 6 service channels (SCH) as shown in Figure 2.3. Coordination is needed for allocating wireless access for safety messages and other applications. The standard provides a split-phase multichannel medium access control protocol with a dedicated control channel. WAVE nodes switch alternatively between CCH and one of the SCHs and this is the reason it is called split-phase.

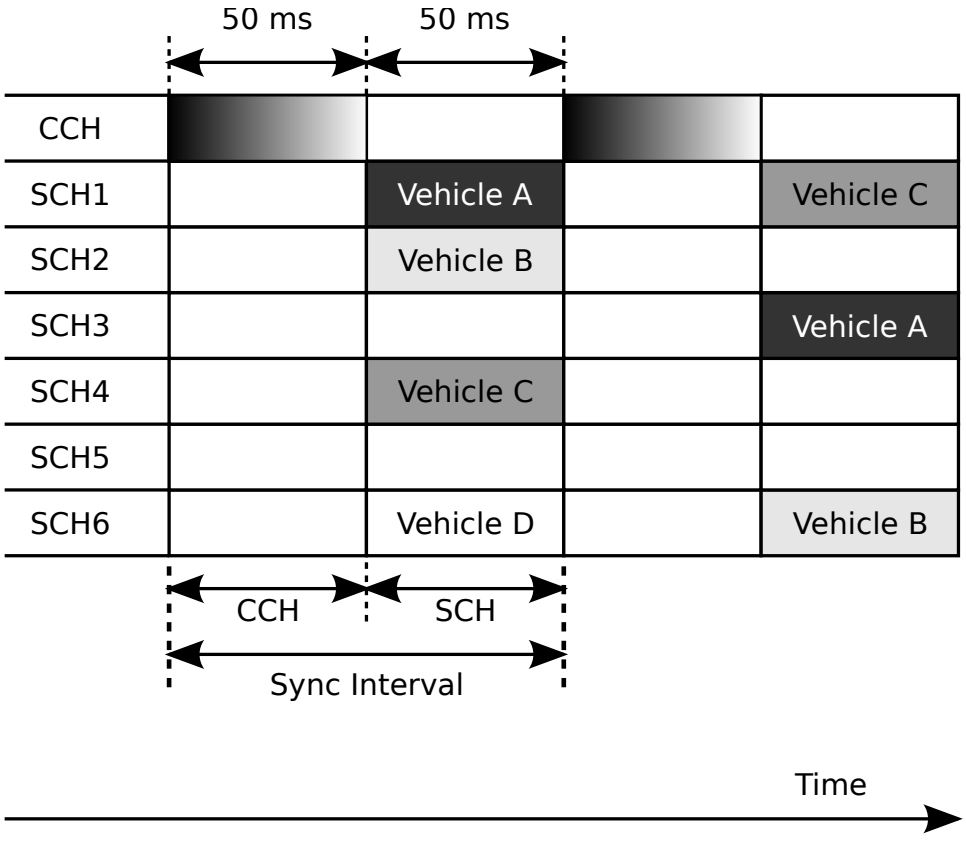

Figure 2.3: Example of Multichannel Operation of IEEE 1609.4

From a functional description point of view, the information exchanged over WAVE is

either control or data. For example, WAVE announcements are management frames that are transmitted over the CCH. Data frames such as WAVE short messages (WSM) can be exchanged over the CCH or the SCHs, while IP data frames are not allowed on the CCH. In order for exchanges over an SCH to happen, the nodes need to be members of the same WBSS. Setting the EDCA parameters is different for CCH and SCH, in that, in the CCH case they are predetermined to favour WSMP data transfers. In the SCH case, the EDCA parameters are transmitted using WAVE announcement frames.

To further clarify its operation, we present an example (Alsabaan *et al.*, 2012) of nodes operating using the DSRC standard in Figure 2.3. The DSRC radio switches between CCH and SCHs. For a duration equal to half the frame it is operating on the CCH. In this period, different nodes transmit beacon messages. These messages contain information regarding the vehicle, i.e., velocity, direction and location, among other information. The frequency of beacon generation is 10 Hz. Aside from the regular beacon, event based messages can be generated by the vehicles in case of emergency. After the CCH period, DSRC nodes can switch to one of the SCHs. In order to use a SCH, negotiations must be performed during the CCH period. These negotiations involve advertising its services using wave short message (WSM) and processing acknowledgments. As beacons, event-based messages and WSM advertisements are broadcast via the CCH, and it can become a performance bottleneck. This can cause congestion and high load on the channel.

The IEEE 1609.4 standard defines four services used to manage channel coordination and to support MAC service data unit (MSDU) delivery. The channel routing service delivers packets from the logical link control (LLC) to the designated channel. Medium access contention is performed by a priority service which applies enhanced distributed channel access (EDCA) from IEEE 802.11e. Channel coordination service ensures data packets

from the MAC are transmitted on the correct RF channel and finally, the MSDU data transfer service guarantees that WSMP obtains higher priority and direct access over IP data.

2.5 Data Scheduling in VANETs

Scheduling has been extensively studied in many different contexts, such as in operating system design (Silberschatz *et al.*, 1991). In this application, first come first serve (FCFS), round robin (RR) and shortest job first (SJF) schedulers were traditionally used for processor scheduling (Kumar and Chand, 2010). As these algorithms were designed for a point-to-point communication environment, they are sometimes not suitable for the broadcast nature of wired or wireless LANs. For this reason, many scheduling algorithms have been proposed in the literature to better utilize the shared nature of the communication medium. The design goals for these algorithms can be quite different, leading to the adoption of different performance metrics, some examples of which are: network connectivity, fairness, reliability, responsiveness, adhering to time constraints, throughput and service ratio. Before introducing some more advanced scheduling algorithms that have been recently proposed, we will cover some of the well known algorithms that serve both as a basis upon which new algorithms are built, and benchmarks against which their performance is measured. First, when data size is not considered a factor in the scheduling decision, earliness, deadline and frequency of request are among the most commonly used properties for which schedulers can be designed.

Some of the classic scheduling algorithms are:

First Come First Serve (FCFS) The simplest scheduling algorithm. The earliest requests are served first.

Longest Wait Time (LWT) The item for which the total waiting time of all outstanding requests is the largest, is scheduled first. It utilizes the broadcast nature of the communication medium by serving multiple requests with a single transmission if the requested data is identical.

Most Request First (MRF) Works in the broadcast environment where items with the highest frequency of being requested are served first.

First Deadline First (FDF) Regardless of the service time, the item with the most urgent deadline gets higher priority in scheduling.

Least Selected First (LSF) The message that had the least opportunity to be served before, would get higher preference.

When message size is taken into account, the following scheduling schemes are commonly used:

Longest Total Stretch First (LTSF) Reduces average waiting time by utilizing a metric called "Stretch" which is defined as the ratio between an item response time and service time.

Smallest Data Size First (SDSF) Requests asking for the least will be honored first.

Maximum Quality Increment First (MQIF) Serves the messages that result in the maximum quality increment first. Quality is based on time reliability metric (TRM) and space reliability Metric (SRM). TRM and SRM measure the percentage of messages that were delivered within their intended time and distance limits, respectively.

The above algorithms can be suitable for LANs, but are not necessarily adequate for the VANET environment. As vehicles travel at high speed, spend limited time inside the RSU

coverage range or in contact with one another, efficient scheduling becomes an important factor in increasing overall system performance. We review a representative sample of scheduling algorithms proposed specifically for VANET environments.

Ott and Kutscher, in their highly cited paper (Ott and Kutscher, 2004), studied the impact of vehicle velocity, transmission rate and packet size on throughput and delay of vehicle to RSU communication in what they called the drive-thru Internet scenario. In this situation, they measured transmission characteristics for sending and receiving high data volumes using UDP and TCP in vehicles moving at different speeds that pass one or more IEEE 802.11 access points at the roadside, concluding with the viability of the drive-thru internet scenario. This viability was further shown in (Gass *et al.*, 2005) where Grass et al. considered the possibilities of using opportunistic communication between RSU and vehicles.

In (Hadaller *et al.*, 2007), it was shown that widely used IEEE 802.11 protocols can only achieve 50% of the overall throughput possible in a drive-thru scenario even when DHCP was disabled. The researchers identified several problems caused by the mechanics of existing protocols that are responsible for this throughput loss, such as MAC management and TCP time outs, slow TCP adaptation for available MAC bit rate and application initialization delay. The paper recommends that algorithms be less sensitive to bursty data losses and should make use of location awareness by using special parameters at the edges of the coverage areas and during the initial setup. Also, MAC and TCP timeout values should be adjusted according to the location in the coverage area. Based on these suggestions, updates for the MAC layer were proposed to enhance opportunistic access for vehicles, as wireless conditions in the vicinity of an RSU are predictable, and by exploiting this information, throughput and reliability can be greatly improved.

In (Suthaputchakun and Ganz, 2007), the authors proposed a priority based scheduler for highway safety messaging based on IEEE 802.11e. They use the Enhanced Distributed Channel Access (EDCA) in IEEE 802.11e to provide priority. As the IEEE 802.11e EDCA MAC uses the Arbitrary Inter-Frame Space (AIFS) and Contention Window (CW) mechanisms to provide differentiated QoS, researchers used this in conjunction with repetitive transmission to enhance transmission priority for important messages. The objective was to maximize the throughput and minimize the delay for higher priority messages.

In (Zhang *et al.*, 2007), the D*S scheduler was proposed which considers both deadline (D) and data size (S) to give a weight for the data message. They improved upon their algorithm by making use of the broadcast nature of the channel and introduced D*S/N where N is the number of requests for the same data item. The proposed method provided significant improvement in terms of throughput and delivery ratio.

In (Shahverdy *et al.*, 2010), researchers continued where (Zhang *et al.*, 2007) left off as the latter paper did not address the application layer and treated each RSU independently. In this work, the authors differentiate between vehicles requesting new files and others wishing to continue to download the rest of a previously initiated download. They propose a scheduling scheme based on which requests are in two different queues. In the case where a vehicle requests a data item for the first time, its request queues in first time queue (f-queue) and if the vehicle requests the remaining part of a data item, its request goes to a continue queue (c-queue). The weights for the heads of the queues are then compared and the one with higher priority is selected. The result for applying the new algorithm shows improvement over the classical FCFS, FDF and SDSF algorithms.

In (Liu *et al.*, 2013), the RSU broadcasts data items in response to requests submitted by passing vehicles. The data items are associated with temporal constraints and are updated

periodically. Each request may ask for multiple temporal data items and is associated with a deadline. The authors introduce a real-time data dissemination model by formulating the time-constrainted requests and consistency requirements as an optimization problem. They propose an online scheduling algorithm to enhance the maximum request service and improve bandwidth utilization. The vehicle requests can be submitted to the RSU via the uplink channel (like CCH in IEEE 1609.4) and RSUs are assumed to be connected to a wired backbone network. The data items, for example traffic conditions at a certain street, are updated by the system periodically, thus multiple versions of the data can exist in the system and each is valid for a certain period of time. If a traffic accident happens at a certain segment of the street, the validity of the temporal data related to traffic at this segment would be overwritten by a new data unit. This uncertainty of the expiry time for a data unit makes the scheduling problem harder to tackle. A scheduling algorithm called Productivity, Status and Urgency (PSU) was proposed. It is designed to exploit the broadcast effect, improve bandwidth utilization and service chance. PSU utilizes the broadcast nature of the VANET to aggregate data units that are requested by multiple vehicles in the same transmission. It also enhances the bandwidth utilization by giving higher priority to requests that have less remaining parts to be satisfied. Compared to classical scheduling methods such as FDF, PSU outperforms them in terms of service ratio and bandwidth saving ratio.

In (Liang and Zhuang, 2011), network coding is used to improve the efficiency of data dissemination. The authors assume limited RSU storage capacity and messages with expiry times. No vehicle to vehicle communication is considered. The messages are predownloaded to the RSUs after encoding. Based on the given vehicle trajectory, resource allocation model and message pre-downloading profile, the optimal scheduling problem is mathematically formulated. The assumption that the vehicle trajectory between the RSUs

is predefined introduces limitations to the application of the method. Another limitation to their model is that they only consider communication between the vehicle and RSU inside the region where maximum transmission rate is used, which can be very limited in the VANET environment.

Reference (Mershad and Artail, 2012) takes a look at another RSU related scheduling problem. Usually it is assumed that the data vehicle request is readily available at the RSU by the time the vehicle arrives into the RSU coverage or that downloading them from a backhaul network requires little time. This research addresses the problem of utilizing the time slots the RSU is not using to communicate with a vehicle in order to prefetch the data that will be needed by another that will be arriving soon. The users in this system register for service beforehand and create a profile of frequently requested data items. The system pre-caches the data at the RSUs in anticipation for it to be requested by the users. As the system learns the user behavior, a default RSU is assigned to the user and his/her data is prefetched according to probabilities calculated from previous requests. This algorithm can be helpful when the vehicles use the same route frequently and there is consistency in the type of information they require.

2.6 Green Scheduling

The scheduling protocols surveyed so far have focused on network performance metrics such as throughput, delay and connectivity. Little attention has been given to power saving in VANETs since the OBUs are assumed to have an uninterrupted source of power and the RSUs are considered to have power grid connections. In this section, we take a look at power saving in the IEEE 802.11 standard and examine research that proposes new protocols to apply this to the VANET environment.

2.6.1 Using 802.11 Power Saving (PS) in VANET

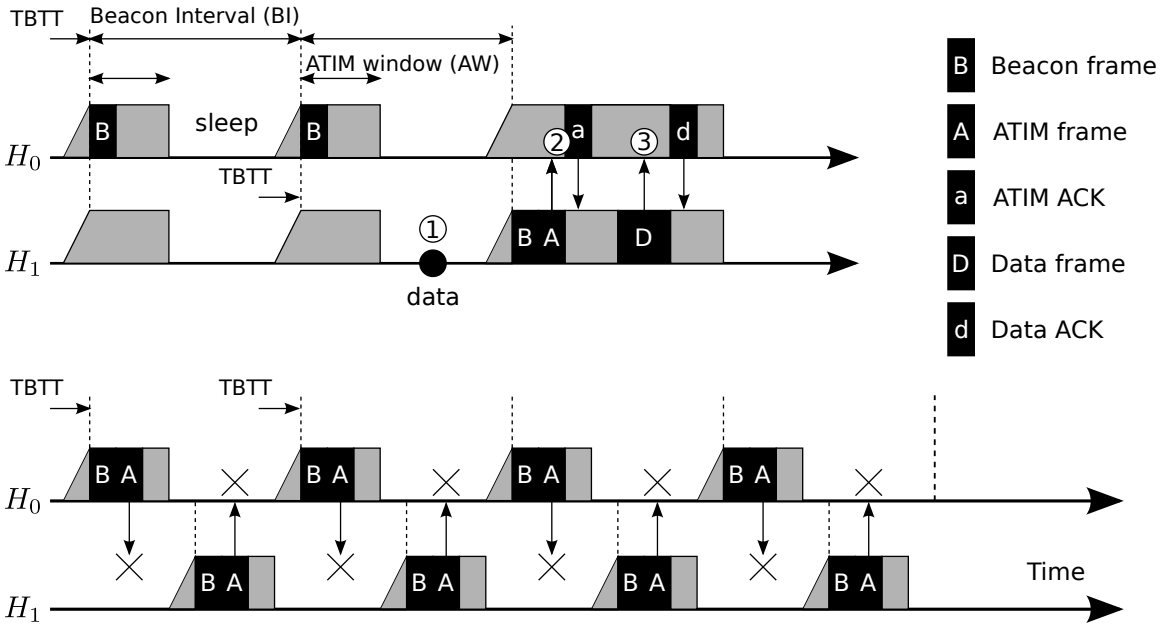

Figure 2.4: IEEE 802.11 Power Saving (PS) Mode

Figure 2.4 shows how the Power Saving (PS) mode in IEEE 802.11 operates. In this mode, the station is required to be awake each beacon interval for a period of time called the Announcement Traffic Indication Message (ATIM). Every Target Beacon Transmission Time (TBTT), a beacon frame is broadcast. In the upper half of the figure, station H_1 wants to transmit to H_0. It notifies H_0 during the ATIM window and waits for acknowledgement. After the end of the ATIM window, they use the usual DCF mechanism to exchange data. If no such data exchange was needed, the station enters the sleep mode after the ATIM. The successful operation of IEEE 802.11 PS protocol requires synchronization between the station timers. In the lower part of the figure, the synchronization between the stations has failed and it shows how the ATIM windows may not overlap, causing the two stations to be unable to communicate. Although there are several proposed methods to achieve

synchronization, due to the dynamic nature of VANETs, they are hard to apply between the vehicles and RSU. Synchronizing sleep/wake cycles between vehicles, which can encounter each other for a very brief period of time, can be impractical. This means that energy conservation between vehicles requires an asynchronous method.

2.6.2 Energy Conservation Protocols for Vehicular Networks

In (Wu *et al.*, 2010), an algorithm is proposed that can be implemented with the intent of preserving energy. As safety applications in VANETs are of great importance, the energy conservation mechanism must honor performance guarantees. A novel energy efficient DSRC based MAC protocol called DSRC-AA is proposed. In order to save energy at the OBU and the RSU, the protocol aims to put the radio in sleep (doze) modes. If it was possible to use IEEE 802.11 PS mode, the saving can be as high as 75%. But this requires the use of an auxiliary timer synchronization mechanism to ensure the overlap of wake times. As mentioned before, applying these mechanisms can be inefficient in the VANET environment.

Based on the IEEE 802.11 PS mode, the Asynchronous Quorum-based Power Saving (AQPS) method is proposed. The stations utilizing the DSRC protocol may sleep through beacon intervals, however, they must follow a schedule that ensures the wake time can overlap. A quorum system defines a pattern for each station to be awake. It guarantees, without need for synchronization, that during a station awake period, at least another station is awake too. As there is no need for a timer, it is more suitable for vehicular network environments. The cycle length, n, is defined as the number of beacon intervals needed for the wake/sleep pattern to repeat itself. One drawback is that there is a limit on the AQPS energy saving as each station has to be awake for a period equal to the square root of the

cycle length n.

In order to increase the energy savings, n needs to increase. But as n increases, neighbor discovery takes more time as awake periods may not overlap except once every full cycle. Another contradicting factor is that in certain scenarios, as in toll booths, contact time can be very limited, which requires n to be very small, thus limiting the energy saving. To address these conflicting requirements, DSRC-AA was proposed which is based on AQPS but with asymmetric properties. The improvement relies on grouping vehicles into clusters. When vehicles form clusters, they elect cluster heads to communicate with other cluster heads and RSUs. Members of the cluster relay schedule and data through cluster head. A symmetric quorum (s-quorum) is used by cluster heads and RSUs to establish symmetric links between them.

Cluster heads and the RSU are permitted to stay awake longer than other members of the cluster who instead use asymmetric quorum (a-quorum). Stations using s-quorum can communicate using the classical AQPS way. A-quorums are not guaranteed to intersect, instead, the cluster-head knows the sleep/awake schedule of all members, so it can relay data to and from members according to their schedules. The protocol is adaptive, for example, members with low relative speed to the cluster head can have more power saving. The published experiments showed a reduction of 44% in energy consumption compared to using classical AQPS method. Despite the energy improvement, one of the protocol weaknesses is that if a cluster head is lost or fragmented, members will have to use s-quorum, which results in lower efficiency. Another weakness is that the heavier duty cycles on s-quorum nodes (cluster heads and RSUs) result in unequal energy saving. Also the protocol becomes less efficient if no clusters can be formed for long enough periods of time or if cluster fragmentation is frequent.

Chapter 3

Energy Efficient Scheduling for Constant Bit Rate Channels

3.1 Introduction

In many highway locations, deploying roadside infrastructure is difficult due to the unavailability or prohibitive expense of wired electrical power. In these situations, an alternative to wired power connections is to operate some of the RSUs using an energy sustainable source such as solar power. In these types of node designs, it is well-known that the energy provisioning costs are a strong function of average power consumption and that they can be a significant fraction of the total node cost (Farbod and Todd, 2006)(Badawy *et al.*, 2010). This motivates the need for energy efficient vehicular roadside infrastructure. In vehicular infrastructure, proper traffic scheduling can lead to significant improvements in energy efficiency due to the strong dependence of power consumption on RSU-to-vehicle distance (Hammad *et al.*, 2010).

Unlike many traditional wireless networks, vehicles are often moving very quickly, and

may only remain in an RSU radio coverage area for a relatively short period of time. In addition, since multiple vehicles may be present in the RSU coverage area, the question arises as to the order with which vehicles should be served. This problem is in general a difficult one. One can argue for example, that faster moving vehicles spend less time close to the RSU and thus their requests should be more highly prioritized compared to that of slower moving vehicles. But this may lead to starvation for slower moving vehicles affecting the fairness of the scheduler.

In this chapter, energy efficient RSU scheduling is considered. The RSU transmission power is varied in order to obtain a constant bit rate channel. In certain vehicular installations, the location of vehicles passing through the RSU radio coverage area can be predicted with a high degree of accuracy. This information can then be used to reduce downlink infrastructure-to-vehicle energy communication costs. The chapter starts by presenting off-line scheduling bounds which provide lower limits on the energy needed to satisfy vehicular communication requests. The chapter considers both *packet* and *timeslot* based scheduling. In the former case, the problem can be formulated as a generalization of the classical single-machine job scheduling problem with earliness and tardiness penalties, referred to as α-Earliness-Tardiness. Even under a simple distance-dependent exponential radio path loss assumption, the problem is shown to be NP-complete. The chapter also considers timeslot-based scheduling. This version of the problem can be formulated as a Mixed Integer Linear Program (MILP), which is shown to be solvable in polynomial time by modeling it as a minimum cost flow problem and using the Integrality Property of its solution. The chapter then introduces online traffic scheduling algorithms for the common vehicular scenario where there is a strong deterministic radio path loss component.

The first algorithm, Greedy Minimum Cost Flow (GMCF) is based on a local optimization using our minimum cost flow model. Two other algorithms are proposed with reduced complexity compared with the GMCF Algorithm. The first algorithm, the Static Scheduler (SS), assigns time slots according to a simple position-based weighting function. The second is Nearest Fastest Set (NFS) scheduler that uses vehicle location and velocity inputs to dynamically assign communication slots. Results from a variety of experiments show that the proposed scheduling algorithms perform well when compared to the energy lower bounds in vehicular situations where path loss has strong deterministic components. Our results also show that near-optimal results are possible but come with increased computation times compared to our heuristic algorithms.

The remainder of the chapter is organized as follows. In Section 3.2 a brief overview is given of related work. In Section 3.3 we give a detailed description of our system assumptions. Section 3.4 formulates lower bounds on the energy performance of the offline RSU scheduling problem. In Section 3.4.1 the packet-based scheduling problem is shown to be a generalization of the classic single machine job scheduling problem with earliness and tardiness penalties. The complexity of this problem is considered in this section and a proof of NP-completeness is given in the Appendix. Following this, in Section 3.4.2 the timeslot-based scheduling problem is formulated as a mixed integer linear program and in Section 3.4.2 a minimum cost flow formulation is used for offline scheduling. These provide lower bounds on the energy consumption needed to satisfy arriving vehicular requirements. The chapter then introduces online scheduling algorithms in Section 3.5. In Sections 3.5.2, 3.5.3 and 3.5.4, the Greedy Minimum Cost Flow, Static, and Nearest Fastest Set schedulers are introduced. Performance comparisons are then presented in Section 3.6. Finally, in Section 3.7, the conclusions of the chapter is presented.

3.2 Related Work

VANET research has spanned a wide variety of topics in recent years. This includes applications (Khaled *et al.*, 2009), routing protocols (Li and Wang, 2007), authentication (Zhang *et al.*, 2009), and the performance analysis of the IEEE 802.11p standard (Mittag *et al.*, 2008). Several studies (e.g., (Bychkovsky *et al.*, 2006) and (Ott and Kutscher, 2004)), have illustrated the suitability of IEEE 802.11p for highway applications and in (Jhang and Liao, 2008), (Zhao *et al.*, 2008) and (Nandan *et al.*, 2005), proxy vehicles are used to improve roadside unit utilization, and to decrease vehicle contention.

Vehicle transmitter power control has been used as a mechanism for trading off network connectivity and reduced interference levels between vehicles (e.g., (Mittag *et al.*, 2008), (Festag *et al.*, 2007) and (Rawat *et al.*, 2009)). The energy efficiency for VANETs however, has typically not been an issue, as vehicles are usually assumed to have unlimited energy reserves. Moreover, from the roadside infrastructure point of view, most work assumes urban settings where wired power is available at reasonable cost.

Traffic scheduling at the RSU has been considered in (Zhang *et al.*, 2007) where simple schedulers are used based on data size and deadline but without considering the energy consumption of the infrastructure. In (Alcaraz *et al.*, 2009), an optimization is used to maximize the total throughput of a RSU given the locations and velocities of the vehicles in range. A scheduler was proposed which is suitable for use in the contention free period of IEEE 802.11e. The energy consumption for the RSU however, was not taken into consideration. It is this aspect that is considered in this chapter.

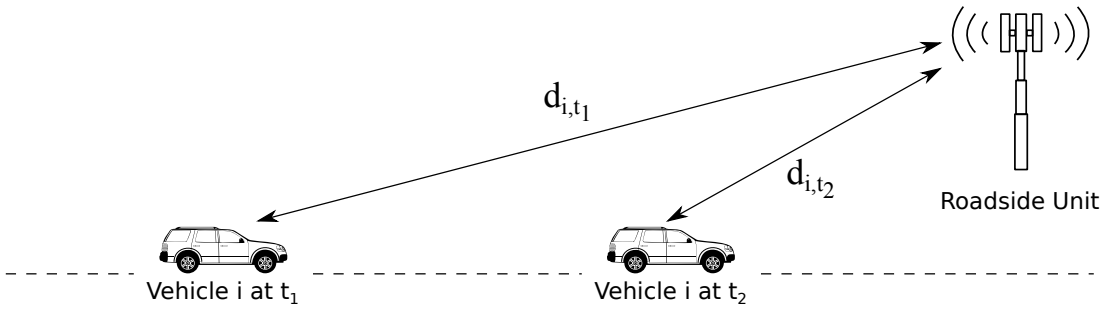

Figure 3.1: Roadside Unit (RSU) Example. Vehicle i is shown at two different times, t_1 and t_2, at distances d_{i,t_1} and d_{i,t_2} from the RSU, respectively.

3.3 System Description

A roadside unit is considered which is serving passing vehicles as shown in Figure 3.1. Although the figure shows traffic travelling in a single direction, the descriptions, scheduling algorithms, and results in this chapter are equally applicable to two-way vehicular traffic. The figure shows a single Vehicle i at two different times, t_1 and t_2, and at distances d_{i,t_1} and d_{i,t_2} from the RSU, respectively. We consider the energy consumption of the RSU radio interface that is used to communicate with the vehicles in the downlink (i.e., RSU-to-vehicle) direction [1]. Note that since vehicle radios are powered by the car engines, energy efficiency is not an issue on the vehicular side. We assume that the RSU uses transmit power control so that a constant bit rate reception is achieved at each vehicle with which the RSU communicates. For this reason, the power consumption needed when the RSU communicates to a nearby vehicle can be significantly lower than when it communicates with a more distant vehicle. In the example shown in Figure 3.1, the RSU power consumption needed for communicating with Vehicle i may be significantly less at time t_2 compared with that at time t_1 since $d_{i,t_1} >> d_{i,t_2}$. For this reason, communication with Vehicle i at

[1] Due to the coverage range normally associated with RSUs, the average power consumption of an energy efficient RSU design may be strongly dominated by downlink transmission power.

time t_2 is preferable compared to time t_1.

The RSU would like to minimize its long-term power consumption subject to satisfying the communication requests associated with the passing vehicles. We assume that a particular vehicular communication may occur any time throughout the vehicle's RSU transit time, i.e., the communications are delay tolerant for the period of time during which a vehicle is within the RSU coverage range. Some examples of delay tolerant applications are on-demand-video, file transfer, music and radio streaming. In the results in Section 3.6, it is also assumed that a given vehicle travels at a constant speed when moving through the coverage area of the RSU, which is typically the case in highway situations (Khabazian and Ali, 2008), however, different vehicles may be moving at different speeds. Accordingly, we assume that the vehicles do not interact significantly within the RSU coverage range considered (Helbing, 2001). When a vehicle first enters the coverage range area of the RSU, it communicates its current position and speed to the RSU, information which may be obtained via GPS inputs for example (Zhang *et al.*, 2010), and can be used to determine the vehicle's position.

3.4 Offline Energy Bounds

In this section we formulate lower bounds for the total RSU energy needed for downlink radio transmission to serve a finite set of vehicular arrival demands. The bound formulation consists of deriving the energy-optimal *offline schedule* where the entire vehicular arrival process and associated transmission demands are made available to the scheduler. For this reason these bounds are not generally achievable in practice since the scheduler has non-causal knowledge of future vehicular inputs. However, they are important in that they establish limits on what can be achieved in practice and are compared with online

scheduling algorithms later in the chapter.

Two cases are considered. The first assumes *Packet-Based Scheduling*. In this case contiguous downlink time is allocated to satisfy each vehicular communication requirement. It is shown that this can be formulated as a generalization of classic single machine job scheduling and a proof of NP-completeness is given via a reduction to the well-known PARTITION problem (Garey and Johnson, 1990). The second case assumes *Timeslot-Based Scheduling* where the each vehicle's transmission requirement can be independently allocated across non-contiguous timeslots. We present a mixed integer linear program and give an algorithm based on solving a minimum cost flow problem that runs in time which is polynomial in the number of timeslots.

3.4.1 Packet-Based Scheduling

In this section we will show that this case can be modeled using a classical single machine job scheduling problem with deadlines (Graham *et al.*, 1979). Machine scheduling is first described, then our problem is formulated as a generalization of this well-known problem.

Notation and Framework

In machine scheduling, n jobs are submitted for processing on m machines. The subscript i refers to a job while subscript j refers to a machine. The pair (i, j) refers to processing *job* i on *machine* j. Processing time is represented by p_{ij} if it is machine dependent or p_i if it is machine independent. The release date r_i is the arrival time of job i to the system. The due date, u_i, is the job deadline, and a penalty will occur if job i is completed after this time. The completion time of job i is denoted by C_i and the objective is also a function of

the due date u_i. The *lateness* of job i is defined as

$$L_i = C_i - u_i \,, \tag{3.1}$$

and the *tardiness* is defined by

$$T_i = \max(C_i - u_i, 0) = \max(L_i, 0). \tag{3.2}$$

Lateness and tardiness are two of the basic due date related penalty functions (Graham *et al.*, 1979).

We are interested in the *total weighted tardiness*, given by $\sum_{i=1}^{n} w_i T_i$, i.e., the sum of tardiness of all processed jobs, weighted by a weight w_i. The *earliness* of job i is defined in a symmetric way as

$$E_i = \max(u_i - C_i, 0). \tag{3.3}$$

Again, we are interested in the *total weighted earliness*, i.e., $\sum_{i=1}^{n} w_i E_i$, where the weights are the same as those used for the total weighted tardiness above. When tardiness and earliness are combined in one objective, the scheduler will try to have each job serviced exactly at its due date, otherwise a penalty of w_i will be paid by each job i for every unit of earliness or tardiness. In our case every vehicle defines a job whose deadline is its arrival time at the RSU, and we would like to have it serviced at that time, i.e., using the minimum estimated RSU transmit power.

Earliness-Tardiness Single Machine Scheduling

We now formulate the minimum energy scheduling problem as a generalization of single machine scheduling. Each vehicle has an associated request which is equal to its job size in

units of slot times. If we represent the RSU as a single machine, in order to promote serving vehicles closest to the RSU, we can choose the objective function to be the minimization of a combination of earliness and tardiness. In order to do this, we represent the due date u_i for each communication slot as the time at which vehicle i arrives *exactly* at a position closest to the RSU. Executing job i before reaching the RSU is penalized (energy-wise) by the earliness component E_i, while executing the job after leaving the RSU location is penalized (energy-wise) by the tardiness component, T_i, of the objective function. Therefore, we can think of our problem as a schedule on a single machine that minimizes the total weighted earliness and tardiness, i.e., $\sum_{i=1}^{n} w_i(E_i+T_i)$, or, if we use the standard scheduling notation of Graham et. al. (Graham *et al.*, 1979), $1||\sum_{i=1}^{n} w_i(E_i + T_i)$.

In our case, earliness and tardiness in time corresponds to increases in the power consumption with distance from the RSU. In this case, the energy dependence of the objective on E_i and T_i is not linear but is governed by RSU-to-vehicle path loss. For example, if we assume a standard distance-dependent exponential radio path loss propagation model (Rappaport, 1996) the relationship between the distance and the required transmission power needed to overcome path loss when the RSU is communicating with vehicle i at time t is given by

$$P_{i,t} = \rho P_0 \left(\frac{d_{i,t}}{d_0} \right)^{\alpha}, \tag{3.4}$$

where P_0 is a reference power, α is the assumed propagation path loss exponent, d_0 is a reference distance, ρ is a scaling constant, and $d_{i,t}$ is the distance between vehicle i and the RSU at time t (Rappaport, 1996).

To properly model the energy distance dependence in this case, we define for each vehicle i, $\hat{d}_i$ as the distance that the vehicle is from the RSU at the time that the *middle* of

it's message transmission occurs. This can be expressed as

$$\hat{d}_i = \bar{d}_i + d_0 + \tilde{d}_i, \tag{3.5}$$

where $\bar{d}_i$ is a normalized distance from the RSU to vehicle i at the time that the middle of the message is being transmitted, e.g., if $\bar{d}_i = 0$, exactly half the message has been transmitted when the vehicle is at the point closest to the RSU (i.e., the optimum power position). $\tilde{d}_i$ is the additional distance that the vehicle was from when the middle of the message was transmitted to where the vehicle was when the edge of the message was transmitted (it could be the front or back edge of the message). Therefore,

$$\bar{d}_i = v_i|C_i - u_i|. \tag{3.6}$$

i.e., C_i and u_i are defined as above in units of seconds referenced to the middle of the packet, i.e., $|C_i - u_i|$ is how long in seconds the packet was early or tardy as previously discussed. v_i is vehicle i's velocity. Also,

$$\tilde{d}_i = v_i p_i/2, \tag{3.7}$$

where p_i is vehicle i's packet transmission time. Then the transmit power needed for this packet is given by

$$\begin{aligned} P_t &= \rho P_0 \left(\frac{\bar{d}_i + d_0 + \tilde{d}_i}{d_0} \right)^{\alpha} && (3.8) \\ &= \rho P_0 \left(\frac{|C_i - u_i| v_i + v_i p_i/2 + d_0}{d_0} \right)^{\alpha} && (3.9) \\ &= h_i \left(|C_i - u_i| + r_i \right)^{\alpha}, && (3.10) \end{aligned}$$

where h_i and r_i are additional constants specific to vehicle i. The total energy required for this packet transmission is therefore $p_i P_t$. The total energy needed to communicate to all n vehicles is therefore given by

$$\sum_{i=1}^{n} p_i h_i (|C_i - u_i| + r_i)^{\alpha} . \tag{3.11}$$

Note that a minimum energy scheduler must minimize Equation 3.11. The scheduling system described above can be represented in the standard form of Graham et. al. (Graham *et al.*, 1979) as

$$1||\sum_{i=1}^{n} y_i (|C_i - u_i| + r_i)^{\alpha}, \tag{3.12}$$

where $y_i = p_i h_i$. It can be seen that when $\alpha = 1$ and $r_i = 0$ for all i, Equation 3.12 reduces to conventional earliness/tardiness as discussed in Section 3.4.1. This is therefore a generalization of the classical Earliness-Tardiness single machine scheduling problem, which we refer to as α-Earliness-Tardiness.

In the Appendix we provide a proof of the complexity of this problem showing its NP-completeness by a reduction to the PARTITION problem (Garey and Johnson, 1990). This result establishes the complexity of optimal packet-based energy scheduling even when propagation is governed by simple distance dependent exponential path loss.[2] The proof given can also be easily extended to less restrictive cases than this by specifying arbitrary path loss values and by modifying the weights used in defining the instance of PARTITION.

In the remainder of the chapter we focus on the timeslot-based scheduling case. This is more practical from a general scheduling viewpoint, and it will be shown that polynomial algorithms are possible even in the offline scheduling case.

[2]Note that this result is true for all $\alpha \geq 1$ so we do not need to know α's exact value in practice in order to establish the NP-completeness.

3.4.2 Timeslot-Based Scheduling

In this section we consider the timeslot-based scheduling case where packets that make up a vehicle's transmission requirement can be independently assigned to (non-contiguous) time slots. A mixed integer linear program is first presented and a polynomial complexity algorithm is given based on solving a minimum cost flow graph.

Mixed Integer Linear Program Formulation

An offline MILP optimization is formulated whose output gives a schedule that achieves minimum energy consumption and satisfies RSU-to-vehicle communication requirements. We are given an input vehicle traffic trace consisting of N vehicles indexed by the set $\mathcal{N} = \{1, ..., N\}$. Each vehicle has a communication requirement, R_i bits, for vehicle $i \in \mathcal{N}$ and the vehicles pass completely by the RSU during the time period $\mathcal{T} = \{1, ..., T\}$. Our objective is to minimize the total downlink energy needed to process the vehicular requests. This can be easily written as a mixed integer linear program (MILP), i.e.,

$$\underset{K_{i,t}}{\text{minimize}} \quad \sum_{t=1}^{T} \sum_{i=1}^{N} \rho \mathcal{L}_{i,t} K_{i,t} \tag{3.13}$$

$$\text{subject to} \quad \sum_{t \in \mathcal{T}} K_{i,t} \geq R_i / B, \quad \forall i \in \mathcal{N} \tag{3.14}$$

$$\sum_{i=1}^{N} K_{i,t} \leq 1, \quad \forall t \in \mathcal{T} \tag{3.15}$$

$$K_{i,t} \in \{0, 1\}, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}, \mathcal{L}_{i,t} \leq \mathcal{L}_{max}$$

$$K_{i,t} = 0, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}, \mathcal{L}_{i,t} > \mathcal{L}_{max}. \tag{3.16}$$

In Equation 3.13, $K_{i,t}$ is a binary decision variable equal to 1 if the RSU transmits to Vehicle i at time t and 0 otherwise. The objective uses the propagation path loss, $\mathcal{L}_{i,t}$, for

Vehicle i at time t to calculate the total downlink energy needed by the RSU to serve the demands of the vehicles and ρ is an energy scaling factor. Constraint 3.14 ensures that the scheduler satisfies the communication demands of all the vehicles where B is the number of packet (payload) bits carried per timeslot. Constraint 3.15 and the restrictions on $K_{i,t}$ ensure that the RSU communicates with at most a single vehicle during each time slot[3]. Note that in the above optimization, when a vehicle is outside of the maximum coverage range of the RSU, which corresponds to a path loss exceeding $\mathcal{L}_{max}$, the associated values of $K_{i,t}$ are set to zero.

The above model ensures that vehicle requirements are met with minimum expended energy and provides a lower bound on the energy needed by any realizable scheduling algorithm. Although we have solved this optimization directly for small problem sizes[4], solutions in general may require exponential time-complexity due to the MILP formulation. In the next section we present a polynomial complexity algorithm that can be used to perform this optimization instead, based on a minimum cost flow graph formulation.

Minimum Cost Flow Graph Formulation

In this model, we modify the above MILP formulation, without any loss of generality, to a graph problem that can be solved using a minimum cost flow algorithm. Letting

$$U_{i,t} = \rho \mathcal{L}_{i,t}, \tag{3.17}$$

[3]This result is very general and is not restricted to vehicular traffic passing the RSU in a single direction.
[4]This can be done using standard techniques such as branch-and-bound methods (Hammad *et al.*, 2010).

we can write the objective as

$$\underset{K_{i,t}}{\text{minimize}} \sum_{t=1}^{T} \sum_{i=1}^{N} U_{i,t} K_{i,t}. \tag{3.18}$$

Note that the RHS of Constraint 3.14 can be made integral without changing the feasibility set, and therefore can be written as

$$\sum_{t=1}^{T} K_{i,t} \geq \lceil R_i/B \rceil, \quad \forall i \in \mathcal{N}, \tag{3.19}$$

and as we are seeking the minimum, Equation 3.19 can be tightened to

$$\sum_{t=1}^{T} K_{i,t} = \lceil R_i/B \rceil, \quad \forall i \in \mathcal{N}. \tag{3.20}$$

The problem after modification can now be written as

$$\begin{aligned}
\underset{K_{i,t}}{\text{minimize}} \quad & \sum_{t=1}^{T} \sum_{i=1}^{N} U_{i,t} K_{i,t} && (3.21)\\
\text{subject to} \quad & \sum_{t=1}^{T} K_{i,t} = \lceil R_i/B \rceil, \quad \forall i \in \mathcal{N} && (3.22)\\
& \sum_{i=1}^{N} K_{i,t} \leq 1, \quad \forall t \in \mathcal{T} && (3.23)\\
& K_{i,t} \in \{0,1\}, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}, \mathcal{L}_{i,t} \leq \mathcal{L}_{max} && \\
& K_{i,t} = 0, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}, \mathcal{L}_{i,t} > \mathcal{L}_{max} && (3.24)\\
& U_{i,t} \geq 0, \quad \forall i \in \mathcal{N}, t \in \mathcal{T}. && (3.25)
\end{aligned}$$

In this form the optimization can be viewed as a standard Minimum Cost Flow Problem (Ahuja *et al.*, 1994). This is shown in the graph of Figure 3.2, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

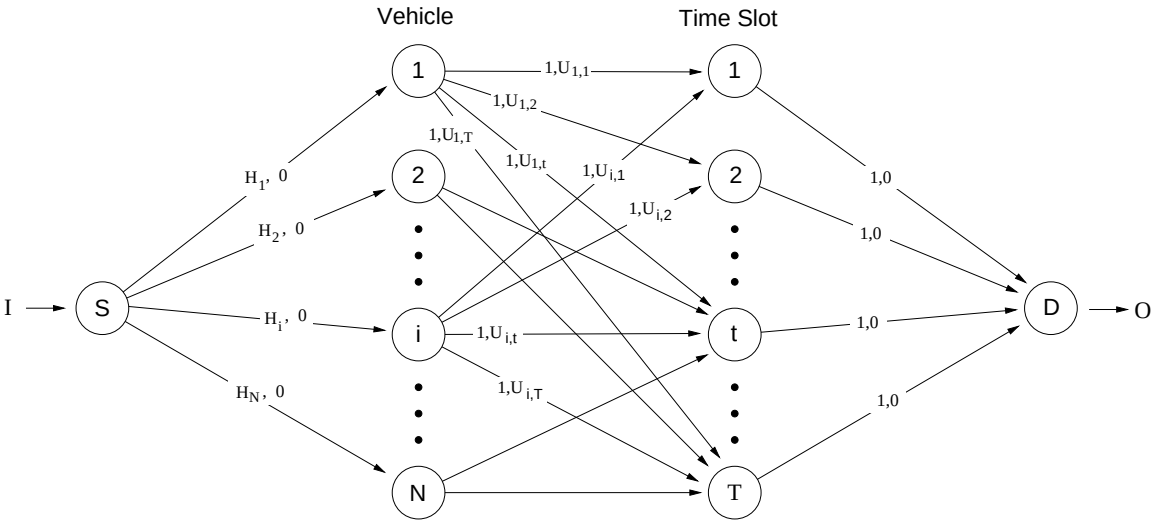

Figure 3.2: Minimum Cost Network Flow Graph Representation, $\mathcal{G}$. Each edge is labeled with an ordered pair, $(C_{i,j}, U_{i,j})$, where $C_{i,j}$ and $U_{i,j}$ are the capacity and cost of using edge (i, j), respectively. The input and output links, I and O, carry a flow of $\sum_{i=1}^{N} H_i$ with a 0 edge cost.

is defined by a set $\mathcal{V}$ of vertices (nodes) and a set $\mathcal{E}$ of edges (arcs) connecting the nodes. For each edge $(i, j) \in \mathcal{E}$ we associate a capacity $C_{i,j}$ that denotes the maximum flow on the edge. Each edge (i, j) also has an associated cost, $U_{i,j}$, that denotes the cost per unit flow on that edge. These are written as ordered pairs, $(C_{i,j}, U_{i,j})$, on each graph edge in Figure 3.2. For example, the capacity and cost of edge $(S, 1)$ in the figure is given by H_1 and 0, respectively.

We associate with each vertex $i \in \mathcal{V}$ a number b_i which represents the supply/demand of the vertex. If $b_i > 0$, the node is a supply node; if $b_i < 0$, node i is a demand node and node i is a transshipment node if b_i is zero. In our problem, the set of nodes is given by $\mathcal{V} = \{S\} \cup \mathcal{N} \cup \mathcal{T} \cup \{D\}$. The flow enters and exits the graph at dummy nodes S and D, respectively and all other nodes are transshipment. The first column of nodes represents all vehicles in $\mathcal{N}$ and the second column represents all time slots in $\mathcal{T}$. Each

vehicle node has edges connected to the time slot nodes during which the vehicle is inside the RSU coverage range. For this reason, slower moving vehicles will have larger numbers of vehicle-to-timeslot graph edges, i.e., higher total RSU-to-vehicle capacity. The capacity for an edge from the source S to a vehicle node $i \in \mathcal{N}$ is the communication requirement for vehicle i denoted H_i where

$$H_i = \lceil \frac{R_i}{B} \rceil. \tag{3.26}$$

The capacity for an edge from any time slot node to the destination D is 1. This capacity prevents time slots from being used more than once. The edges between a vehicle $i \in \mathcal{N}$ and a time slot $t \in \mathcal{T}$ also has a capacity of 1. This ensures that only one unit of transmission requirement can be assigned to a given time slot.

The cost for using the edges originating from Node S or terminating at Node D is zero as these are dummy flow collection nodes. The cost of using the edges between nodes $i \in \mathcal{N}$ and $t \in \mathcal{T}$ is given by $U_{i,t}$ which can be computed from Equation 3.17. Finding the minimum cost flow for graph $\mathcal{G}$ provides the minimum energy the RSU must consume to schedule vehicle transmission requirements for the input traffic trace.

Now the importance of transforming the data in Constraints 3.14 to the integral data in Constraints 3.19 becomes apparent, because we can used the Integrality Property Theorem (e.g., Theorem 9.10 in K. Ahuja et. al. (Ahuja *et al.*, 1994)), which states that "If all edge capacities and supplies/demands of nodes are integer, the Minimum Cost Flow problem always has an integer minimum cost flow". Accordingly, the resulting flow between the vehicle nodes $\mathcal{N}$ and time nodes $\mathcal{T}$ is integer. Coupled with the fact that the capacity of these edges is 1, the resulting flow is a binary matrix, $K_{i,t}$, for $i \in \mathcal{N}$ and $t \in \mathcal{T}$. When $K_{i,t} = 1$, the RSU communicates with vehicle i at time t and when $K_{i,t} = 0$ the slot t is not used for this communication. Since S and D are dummy nodes, the $K_{i,t}$ part of the flow is

the vehicle schedule that we can now compute using standard flow algorithms that run in time polynomial in T and N.

3.5 Online Timeslot-Based Scheduling Algorithms

3.5.1 Motivation and Notation

The results in Section 3.4.2 give a lower bound on the downlink RSU energy needed to fulfill vehicular packet requirements. In order to compute these bounds, the energy costs associated with a given packet transmission must be known. Although it is difficult to precisely know this information in general situations, in certain scenarios excellent estimates of this cost can be readily made (Wang, 2005)(C. Sommer and Dressler, 2011). Accordingly, we consider a highway scenario where vehicles travel at a constant speed through the RSU coverage area (Khabazian and Ali, 2008). The traffic flow may be bi-directional. When vehicles enter the RSU coverage area, they announce their location, direction and speed, information that can subsequently be used to estimate future energy transmission costs[5]. The results we present in Section 3.6 show that provided that the deterministic components of path loss are dominant, large improvements in performance are possible[6]. This would typically be the case in highway situations.

Three online algorithms are introduced which attempt to reduce total energy and which have different processing time complexity. In the following sections, t' is used to denote the current time slot. The vehicle arriving at the RSU coverage range at time t' is denoted v'. The algorithms are applicable to bi-directional traffic flow and when more than one

[5]This information can be obtained from GPS readings at the vehicle, for example.

[6]We assume that downlink power control is used during RSU-to-vehicle communication. This can be accomplished in a variety of ways such as using a short two-way handshake prior to user data packet transmission.

vehicle arrives in the same time slot, but for simplicity of the presentation we assume one vehicle can arrive during a time slot. The set $\mathcal{N}'$ contains all vehicles inside the RSU coverage range at t' with unsatisfied communication requirements. Vehicle $i \in \mathcal{N}'$ unsatisfied demand at time t' is denoted H'_i. The energy cost that the RSU expends to communicate with vehicle i at time slot t is denoted $U_{i,t}$ and is computed according to Equation 3.17. As each vehicle has a different arrival time to the RSU and different speed, the time slot representing its departure time t''_i from the RSU coverage can be different. For $i \in \mathcal{N}'$ the set of time slots between t' and t''_i is called $\mathcal{T}_i$. The time elapsed between current time and the departure of last vehicle of $\mathcal{N}'$ is $\mathcal{T}' = \bigcup_{i \in \mathcal{N}'} \mathcal{T}_i$.

3.5.2 Greedy Minimum Cost Flow (GMCF)

In Greedy Minimum Cost Flow (GMCF), the vehicle schedule is obtained using a greedy version of the bound formulation from Section 3.4.2. However, unlike the bound which incorporates all vehicle communication requirements at once, GMCF constructs a graph similar to the one in Figure 3.2 but limited to those vehicles that are currently inside the RSU coverage range.

GMCF is executed upon arrival of a new vehicle v' into the RSU range at time t'. The directed graph constructed for GMCF is similar to that of Figure 3.2 with $\mathcal{N}'$ replacing the vehicle column and $\mathcal{T}'$ replacing the time slot column. The capacity between Node S and Node $i \in \mathcal{N}'$ is given by H'_i which represents Vehicle i's *unsatisfied* communications requirements at time t'. The supply, I', to Node S is the demand O' from Node D, i.e.,

$$I' = -O' = \sum_{i \in \mathcal{N}'} H'_i. \tag{3.27}$$

Let this graph be denoted $G'(V', E')$, where $V' = \{S\} \cup \mathcal{N}' \cup \mathcal{T}' \cup \{D\}$ and E' is the set of edges between S, $\mathcal{N}'$, $\mathcal{T}'$ and D. Let flow $\mathcal{F}'$ be the minimum cost flow for graph $G'(V', E')$ and let the part of $\mathcal{F}'$ representing flows between vehicle nodes $i \in \mathcal{N}'$ and time slots $t \in \mathcal{T}'$ be $K'_{i,t}$. Then, $K'_{i,t}$ is the schedule for vehicles $i \in \mathcal{N}'$ during the time period $\mathcal{T}'$ or until a new vehicle enters the RSU range. In the latter case, the remaining unexecuted part of $K'_{i,t}$ is ignored and the algorithm is restarted. Whenever Vehicle i is served, its corresponding unsatisfied demand H'_i is reduced by 1.

Reference (Ahuja *et al.*, 1994) provides several algorithms for solving the minimum cost flow problem in polynomial time. Although algorithms such as Capacity Scaling, Cost Scaling and Double Scaling are polynomial, they are not strongly polynomial. For example, the Double Scaling Algorithm solves the problem in $O(nm \log U \log(nC))$, where n is number of nodes, m number of edges, C is the maximum cost and U is the maximum capacity. In GMCF, the cost over an edge can be quite large. Algorithms such as Repeated Capacity Scaling and Enhanced Capacity Scaling provide strongly polynomial time execution. For example, the complexity for Enhanced Capacity Scaling is $O((m \log n)(m + n \log n))$ is an improvement as they do not depend on C or U.

In GMCF the number of nodes, n is $v + t$, where v is the number of vehicles inside the coverage range and t is the number of time slots needed to exit the RSU coverage range. As for the number of edges, m it can be assumed to be $m = v \times t$ as if $\mathcal{N}'$ and $\mathcal{T}'$ are fully connected. Substituting these values in the Enhanced Capacity Scaling complexity, the GMCF complexity can be expressed as $O(v^2 t^2 \log(v + t))$.

To further clarify the different behavior of the online algorithms we discuss a simple example of two vehicles arriving at the same time, either from the same direction or opposite directions. The two vehicles have the same communication demand and travel at the same

speed. In the GMCF Algorithm, both nodes representing the vehicles in the flow graph will be connected to the same time slot nodes with the same cost over the corresponding edges. This leads the solver to treat them equally, and the assigned time slots between the two vehicles will be in arbitrary order, as one would expect since there is no preferential schedule from an energy viewpoint.

Although the GMCF algorithm achieves good energy usage, running the algorithm can be time consuming. In the following sections we introduce two heuristics, i.e., Static Scheduler (SS) and Nearest Fastest Set (NFS) which are more efficient from a time complexity viewpoint.

3.5.3 Static Scheduler (SS)

The basic idea in SS is to sort vehicles according to the energy they would use if they were served at energy optimal positions. The algorithm is static in the sense that these weights do not change as the vehicle propagates through the RSU coverage range. SS serves vehicles with high energy costs first in order to reduce the total energy required. For example, if two vehicles with equal demands are travelling a different speeds it is better to serve the faster vehicle first in order to avoid the extra energy costs of serving it at higher distances. SS would allocate the faster vehicle all its requested time slots and then search for other time slots to assign to the slower vehicles. In SS the scheduling is re-computed when a new vehicle enters the RSU coverage area.

SS is executed upon the arrival of a new vehicle v' into RSU range at time t'. The algorithm consists of two phases, namely, weight computation and scheduling. In the weight computation phase, the weight W_i' for each vehicle $i \in \mathcal{N}'$ is computed by finding its optimal energy cost. This can be best described by using a minimum cost flow graph as in

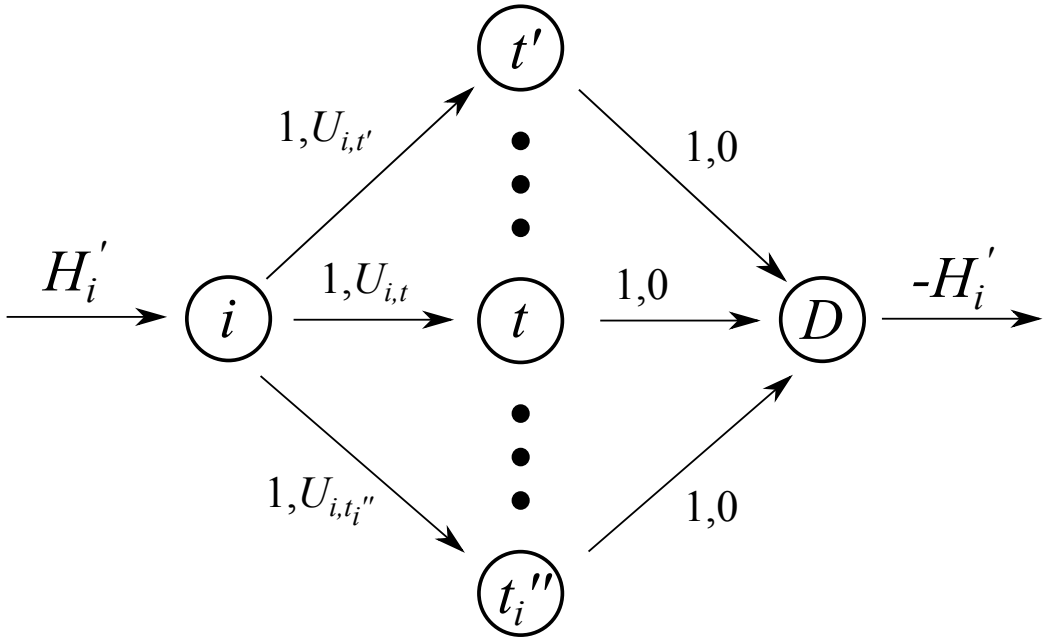

Figure 3.3: Example Flow Graph, G_i, for the SS Algorithm. This is used in the weight calculation phase.

Figure 3.2 but restricted solely to the vehicle in question. We denote that flow graph as G_i for Vehicle i. An example of this is shown in Figure 3.3, and contains one source node i which generates a flow equal to Vehicle i's remaining demand H_i' at time t'. It also contains node D which is a dummy destination node with demand equal to $-H_i'$. The intermediate nodes represent time slots of the set $\mathcal{T}_i$, starting at t' and ending with t_i''. The edge capacity between the vehicle i node and time node $t \in \mathcal{T}_i$ is set to 1. The cost $U_{i,t}$ over these edges is computed according to Equation 3.17. The capacity and cost for edges between time node $t \in \mathcal{T}_i$ and dummy destination D are 1 and 0 respectively. The minimum cost flow for graph G_i is computed and the cost associated with this flow is the weight W_i' for Vehicle i.

In the scheduling phase a single graph, M, is formed as shown in Figure 3.4. The graph consists of one vehicle source node, time nodes and a dummy destination node D. Initially, the time nodes consists of all elements of the set $\mathcal{T}'$. The edge capacity and cost from any

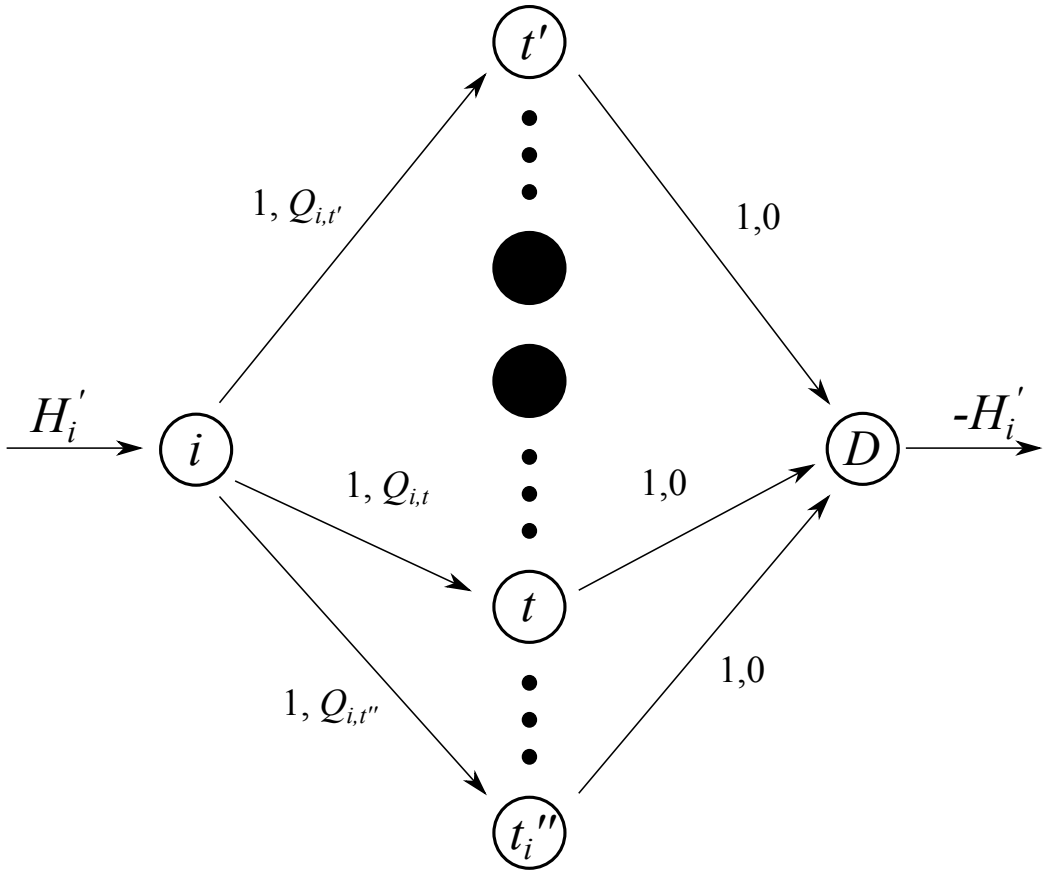

Figure 3.4: Example Flow Graph, M, for the SS Algorithm. This is used in the scheduling phase. Shaded slots have already been assigned for communication with higher priority vehicles.

time node to the D node are 1 and 0 respectively. Vehicles are sorted according to their weights W_i' where $i \in \mathcal{N}'$. In weight descending order, schedules are computed for each vehicle one at a time. Let vehicle $i \in \mathcal{N}'$ be the currently selected vehicle. The supply to the vehicle node i is set to be H_i' and demand by D node is set to $-H_i'$. The capacity of edges between vehicle i node and time $t \in \mathcal{T}'$ is set to 1. The cost, $Q_{i,t}$, of these edges is set as

$$Q_{i,t} = \begin{cases} d_{i,t}, & \text{if } t \in \mathcal{T}_i \\ \infty, & \text{otherwise.} \end{cases} \tag{3.28}$$

Setting the cost to ∞ for time slots when Vehicle i will be outside RSU coverage range prevents selecting them in the schedule. Let the flow $\mathcal{F}$ be the flow that minimizes the cost for graph M. Let the part of $\mathcal{F}$ specifying the flows between vehicle i and nodes $t \in \mathcal{T}'$ be $K'_{i,t}$. $K'_{i,t}$ is the schedule for vehicle i. The set of time nodes in graph M is updated according to

$$\mathcal{T}' = \mathcal{T}' - \{t | K'_{i,t} = 1, t \in \mathcal{T}'\}. \tag{3.29}$$

This removes the time slots already scheduled to serve Vehicle i from the set of available time slots for remaining vehicles. The vehicle following Vehicle i in weight order is selected. Graph M supply, demand, and edge costs are updated according to the following vehicle requirements and distances in the same way like vehicle i. The flow that minimizes the cost for the updated M is found for the new vehicle and the process is repeated until all vehicles are scheduled. After the scheduling phase has been executed for all vehicles $i \in \mathcal{N}'$, the schedule for all vehicles would be $K'_{i,t}$ where $i \in \mathcal{N}'$ and $t \in \mathcal{T}'$. This schedule will execute until the last vehicle of $\mathcal{N}'$ exits or a new vehicle arrives into the RSU range.

Continuing the example introduced near the end of Section 3.5.2 in the case of two vehicles with the same speed, demand and arrival time, the SS would assign the time slots differently. In the weight computation phase, both the vehicles would have the same weight value. This will result in a random selection of the two, with equal probability, for assignment of its desired time slots. All the demand for the chosen vehicle would be served in these time slots without interruption from the second vehicle. The second vehicle would be assigned time slots at a further distance from the RSU.

In determining the complexity of SS, we will use the same notation as in the complexity analysis of GMCF, where v is the number of vehicles inside the range. t is the number of time slots needed for them to exit the RSU range, and we add to them H_m as the maximum

number of slots a vehicle can demand.

SS is invoked upon the arrival of each new vehicle. In the weight computation phase, the process of finding the weight is executed for each vehicle. Finding the weight is equivalent in complexity to finding the minimum of an array of length t. As we find the minimum H_m number of times for each v vehicle, the complexity of weight computation phase can be stated as $O(H_m vt)$.

In the scheduling phase, the search for the highest weight among vehicles is $O(v)$. Scheduling the vehicle with the highest weight is similar to finding the minimum of an array of length t repeated H_m times. As this is repeated for each vehicle, the complexity of the scheduling phase can be stated as $O(H_m vt)$. Thus the total complexity of SS is $O(H_m vt)$. This is a big improvement over GMCF provided that $H_m < vt$, which is safe to assume.

3.5.4 Nearest Fastest Set (NFS) Scheduler

The Nearest Fastest Set (NFS) Scheduler uses vehicle inputs in a simpler and more dynamic way than SS. The motivation is to dynamically change the weight of the vehicles according to remaining demands. If a vehicle is selected for communication from the RSU at the current time slot, its weight is reduced while the weight of other vehicles is increased. The notion of "fastest" comes from the role that vehicle speed plays in weight computation. Consider the case where two vehicles are together and moving away from the RSU. If they are moving at different speeds, then serving the faster one first will lead to lower overall energy consumption. This is clearly due to the fact that in the next time step the faster vehicle will be farther away from the RSU. NFS uses this by embedding the effect of vehicle proximity and velocity in the weight calculation when considering which vehicle

to serve in a given time slot. The execution details of NFS are explained below.

NFS consists of preparation, execution and updating phases. The preparation phase is invoked upon the arrival of a new vehicle v' into RSU range at time t'. The time slots during which vehicle v' will be closest to the RSU are identified. This is done using a graph $G_{v'}$ similar to the one used in the weight computation phase of SS algorithm and shown in Figure 3.3. Vehicle v' requirements $H'_{v'}$ constitutes the supply to the vehicle node. Dummy destination node demand is set to $-H'_{v'}$. The time slot nodes in $G_{v'}$ represent the time slots in the set $\mathcal{T}_{v'}$. The capacities for all the edges in graph $G_{v'}$ are set to 1. The costs over the edges between vehicle v' node and time node $t \in \mathcal{T}_{v'}$ is set to $U_{v',t}$ which is computed according to Equation 3.17. The costs over edges ending in node D are set to 0.

Let $\mathcal{F}$ be the flow that minimizes the cost for graph $G_{v'}$. The weight $W_{v',t'}$ for vehicle v' is the cost of the flow $\mathcal{F}$. Let the array $Z_{v',t}$ represent the part of the flow $\mathcal{F}$ that specifies the flow between the vehicle v' node and time slots $t \in \mathcal{T}_{v'}$. As all supply, demand, capacities and cost of graph $G_{v'}$ are integers, and according to the Integrality Theorem, the flow $\mathcal{F}$ consists of integer values. As the maximum capacity of any edge in $G_{v'}$ is 1, the $Z_{v',t}$ is a binary array. Since the flow $\mathcal{F}$ minimizes the energy cost, in $Z_{v',t}$, the time slots during which vehicle v' is closest to the RSU are set to 1. These are the candidate time slots during which vehicle v' would like to communicate with the RSU.

As the preparation phase is executed for every vehicle $i \in \mathcal{N}'$ and $t \in \mathcal{T}_i$ currently inside the RSU range upon their respective arrivals, there is already a weight $W_{i,t'}$ and separate $Z_{i,t}$ for each vehicle $i \in \mathcal{N}'$ identifying the time slots each vehicle would like to use.

The execution phase happens every time slot. Let the current time be t'. If there is no vehicle $i \in \mathcal{N}'$ that requires the current time slot, then there is nothing to schedule and

the execution phase and update phase are terminated. But if not, let the set of contending vehicles be $\mathcal{E}$. The weights of these vehicles $W_{i,t'}$ are compared and the vehicle with the highest weight is allocated the current time slot t'. Let the vehicle with the highest demand be x. Then the remaining demand for vehicle x is decreased by 1 as it has been scheduled for downlink transmission in the current time slot. The candidate time slot array for vehicle x is updated by setting $Z_{x,t'} = 0$ so its weight will be reduced.

The update phase is for the vehicles that contended for time slot t'. A new set of candidate time slots and weights $W_{i,t'+1}$ for $i \in \mathcal{E}$ is computed. The start time is $t' + 1$ instead of t' because they will be contending for future time slots following t'. These new candidate time slots and vehicle weights are computed in the same way as in the preparation phase.

Continuing with the two vehicle example introduced in the last two sections, the NFS schedules them differently compared with that of GMCF and SS. The two vehicles have the same desired scheduling locations around the RSU. At the first contention, both have the same weight so only this time slot would be assigned, with equal probability, to one of them. In contending for the following time slot, the vehicle that was assigned the first time slot now has lower weight, thus this time slot would be assigned to the second vehicle. The NFS continues to assign time slots to each vehicle, one at a time in this manner, until they are served.

In determining the complexity of NFS, we again use the notation from before where v is the number of vehicles inside the range. t is the number of time slots needed to exit the RSU coverage area, and H_m as the maximum vehicle requirement. The NFS preparation phase is executed upon the arrival of a new vehicle. Unlike SS, this is executed not for all vehicles inside the range but only for the newly arriving vehicle. Thus, the complexity

of this phase is $O(H_m t)$. In the execution phase, determining if one or more vehicles requires communication with the RSU is an addition operation across all vehicle candidate slot arrays. This is equivalent to $O(v)$. Only when there are multiple candidates for a given time slot is the update phase executed. The update phase complexity is equivalent to the preparation phase, but it involves other vehicles inside the RSU, thus the update phase complexity is $O(H_m v t)$. When compared SS, NFS complexity is less. Only during the times of strong contention will NFS complexity become equal to that of SS.

3.6 Performance Evaluation

In this section, the performance of the proposed algorithms is investigated. The theoretical bound for energy required by the RSU to serve the input vehicle requirements as derived in Section 3.4.2 is referred to as *Bound* in the graphs. The bound is compared to the online algorithms (GMCF, SS and NFS) proposed in Section 3.5.

The online algorithms use knowledge of vehicle position and associated estimates of downlink transmission energy costs. For this reason two sets of results will be presented. The first assumes that an accurate prediction of energy costs is possible based on a deterministic path loss scenario using a distance dependent exponential path loss model. These results will give an indication of the best-case potential for energy savings at RSU using the proposed algorithms. In many practical systems however, there will be dominant deterministic propagation with random components due to effects such as shadowing. For this reason we also present results which include errors due to strong shadowing components. It is well-known that in highway scenarios with predominantly line-of-sight propagation, that average path loss has strong deterministic components. A recent measurement based paper has confirmed this even for the vehicle-to-vehicle case where average antenna heights are

low compared with the RSU-to-vehicle case (C. Sommer and Dressler, 2011) [7].

In (Harri *et al.*, 2009) and (Martinez *et al.*, 2011) vehicular traffic models were surveyed including those for intra-city and outside city scenarios. Highway traffic models are known to vary greatly from those in urban settings. For example, Reference (Helbing, 2001) showed that car-to-car interaction was a minimum requirement for realistic highway traffic flow models. A special characteristic of the highway environment is the tendency to maintain constant speed for long durations. Reference (Wang, 2005) for example, models vehicles as belong to different classes with different uniformly distributed speed ranges and in (Khabazian and Ali, 2008)(Bilstrup *et al.*, 2008), vehicle arrivals were taken to be Poisson distributed. The mobility model proposed in this chapter is from (Khabazian and Ali, 2008) and (Wang, 2005). The highway consists of several lanes, which permit vehicles to pass each other without a change in speed. Vehicles traveling on the highway belong to one of several classes. Each class has a Poisson arrival model and a desired speed at which the vehicles travel. Vehicles from each class do not overtake each other but can overtake vehicles from the other classes. There is a single RSU in the middle of the tested highway segment and vehicle arrivals may be viewed as travelling in one direction. Alternately, because of symmetry, one can view the experiments as consisting of a bi-directional highway segment with Poisson vehicle arrivals for each direction at half the one-directional arrival rate.

In the following two sections we present the results of experiments conducted using two and three classes of vehicles. Large scale path-loss using a distance dependent exponential path loss model is used in the first set of results. Following these results, we include random log-normal shadowing for two different levels of shadowing. The random

[7]The scenario that we consider also has the advantage that since the RSU is stationary, it can therefore measure and learn the propagation environment, which would lead to increased accuracy.

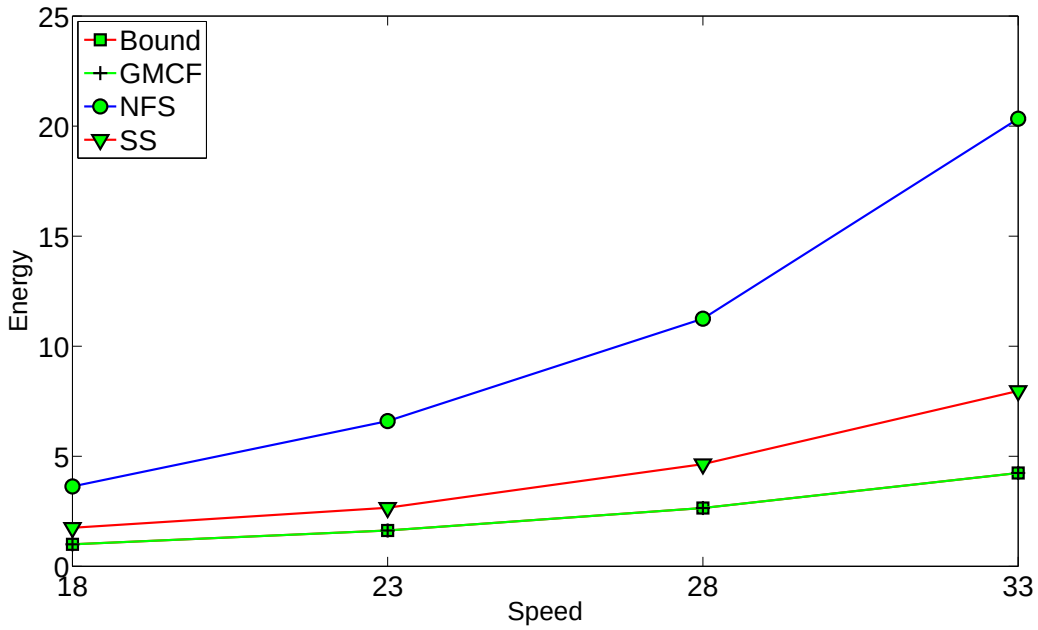

Figure 3.5: Performance of GMCF, SS and NFS for Two Vehicle Classes under Light Loading. No Shadowing Components.

shadowing components are unknown to the online schedulers which base their decisions on deterministic positional information. In the graphs, each plotted point is an average of multiple runs to ensure the results are true representation of the performance of the algorithms. The value of the points are normalized to the first point of the Bound graph in each figure. Traffic flow and scheduling algorithms were implemented in MATLAB. The optimization problems were expressed in AMPL and solved using CPLEX.

In the first experiment the traffic consists of two classes of vehicles with arrival rates of $\lambda_1 = 1/22$ and $\lambda_2 = 1/22$ vehicles/sec, respectively. The communication requirement for each vehicle was tested for low and medium fixed values. The results are presented in Figure 3.5 for low demand and in Figure 3.6 for high demand. Class 1 speed is maintained at 18 m/sec and Class 2 speed is tested for velocities: $18, 23, 28$ and 33 m/sec. Each point is the average of 17 runs.

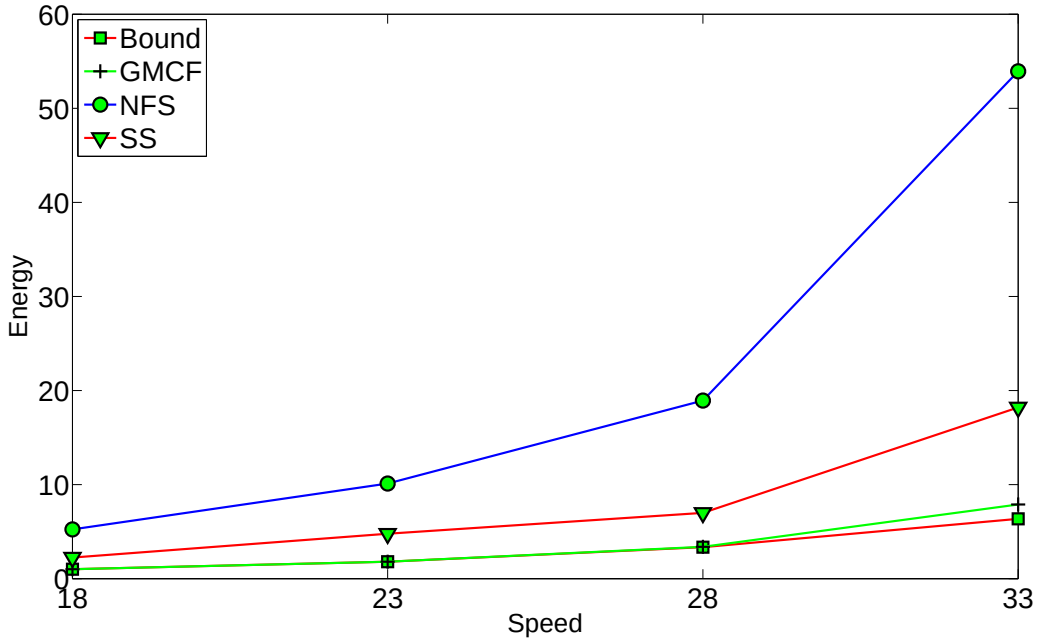

Figure 3.6: Performance of GMCF, SS and NFS for Two Vehicle Classes under Medium Loading. No Shadowing Components.

As vehicular speed increases, the number of time slots the faster moving vehicles, i.e., Class 2, can spend inside the communication range decreases. This forces the algorithms to communicate with the vehicles at more distant locations from the RSU in order to satisfy their communication requirements. So as expected, this shows that the total energy needed is an increasing function of average vehicle speed. Also, it can be seen in Figures 3.5 and 3.6 that the performance of the GMCF algorithm is very close to that obtained from the lower bound. Under low demand levels as in Figure 3.5, even with the increase of Class 2 speed which allows vehicles to pass Class 1 vehicles, these periods of scheduling contention do not seem to be long or frequent enough to result in significant increases in energy compared with the bound. Under heavier loading, as in Figure 3.6, with Class 2 speed at 33 m/sec, periods of contention become longer, thus forcing the GMCF algorithm to communicate with vehicles at larger distances. The price paid for GMCF's good performance

is a relatively long computation time needed to solve the associated minimum cost flow problem.

In these results the SS scheduler consistently requires more energy than the Bound or GMCF. But as shown earlier, its computational requirements are much less than that required by GMCF. Under light vehicular load, the energy required by SS is almost twice that required by the bound as shown in Figure 3.5 and grows linearly. Under heavier loading, as in Figure 3.6, the increase becomes more dramatic. This is due to the static nature of the algorithm. As the difference between Class 1 and Class 2 velocities increase, many more vehicle interactions occur, creating longer periods of scheduling contention which are more difficult for the scheduler to efficiently resolve. In these periods, unlike the GMCF algorithm which seeks minimum energy for all the vehicles currently inside the RSU coverage range, SS tries to minimize the total energy by seeking local minima for individual vehicles in order of their weightings. This drawback of static scheduling is illustrated next with the following example. Suppose two vehicles V_1 and V_2 are inside the RSU coverage range, and V_1's speed is much lower than that of V_2, and the remaining demands are 2 and 1 units of timeslots, respectively. Assume also that both vehicles are closest to the RSU at the same time. Since the candidate location for V_2 is at the same minimum RSU distance, SS serves V_1 at this location, causing V_2 to be served at a more distant location, significantly raising the energy cost for serving V_2, and resulting in a higher overall energy cost. If V_2 had been served at the location the lowest distance, the total energy would have been lower than what SS would choose.

As for the NFS scheduler, although it is a dynamic algorithm compared to SS in terms of continuously reevaluating weights for contending vehicles after assigning a timeslot, scheduling contention is not addressed until they actually happen. This way, if a contention

arises, it is resolved by assigning a slot in the future, which prevents more past energy efficient slots from being used. This explains the increased energy required by NFS compared with the others.

We have also simulated results using a first-come-first-served, i.e., FCFS, scheduler. In this algorithm the RSU places each vehicle request in a FCFS queue and immediately begins service in that order. As a result, FCFS tends to serve vehicles at the outer edge of the RSU coverage area, resulting in very high power consumption values. When compared with the results of our algorithms, FCFS usually results in total energy requirements which are orders of magnitude higher than that of the algorithms shown in our graphs. For this reason we have not presented these results, but this method is clearly a poor approach when energy efficiency is desired.

We now present results for three classes of vehicles. The first class maintains a speed of 18 m/sec across all experiments. The second class is tested for four different speeds, $18, 20, 22$ and 24 m/sec. The third class is tested also for four different speeds, $18, 23, 28$ and 33 m/sec. The arrival rate, λ, for each class is $1/30$ vehicles/second. The results presented here are the average of 10 independent experiments.

In Figure 3.7, all three classes travel at 18 m/sec. The total energy needed by the four algorithms are compared against different levels of demand, ranging from 4 to 10. It can be seen that as before, the GMCF algorithm is very close to the lower bound. This is partially due to the uniformity of the traffic, as all vehicles from all three classes are traveling at the same speed. The scheduling contention tends to be minimal, so there is less value in the bound's use of non-causal information. The behavior of the SS algorithm is similar to the former experiment where under low demand levels, the energy requirement growth is almost a linear relation to that of GMCF. But at high demand levels, the increase in SS

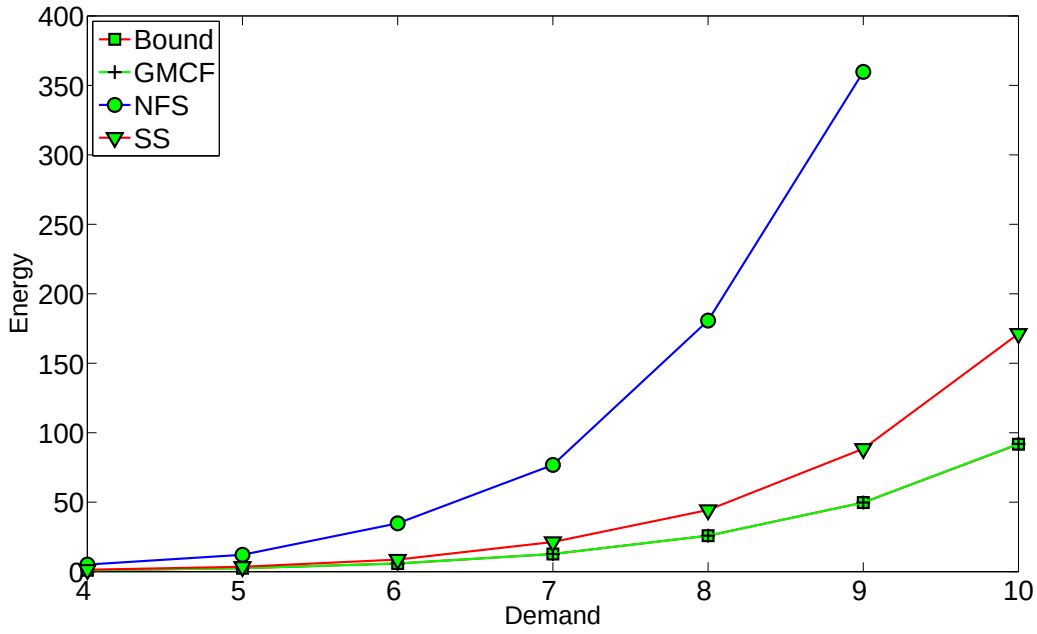

Figure 3.7: Energy Requirements for Three Vehicle Classes Traveling at the Same Speed. No Shadowing Components.

energy demand starts to increase with higher rates than that of GMCF and the bound. The weakness of the NFS algorithm is more evident here. As the demand increases, scheduling contention increases and NFS can only solve it by allocating slots in the forward direction, wasting more energy efficient slots in the backward direction.

In Figure 3.8, Class 1 vehicles are traveling at 18 m/sec, Class 2 at 24 m/sec and Class 3 are traveling at 33 m/sec. This causes the vehicles to interact with each other creating longer and more complex periods of scheduling contention not envisioned by the online heuristics compared to the bound computation. This is why we can see in this figure that when the demand is high, there is a clear difference between the bound and the GMCF algorithm. The NFS and SS behavior is close to that of Figure 3.7, but as the contention periods are longer and more frequent, the increase in energy requirements as demand increases is steeper than that of Figure 3.7.

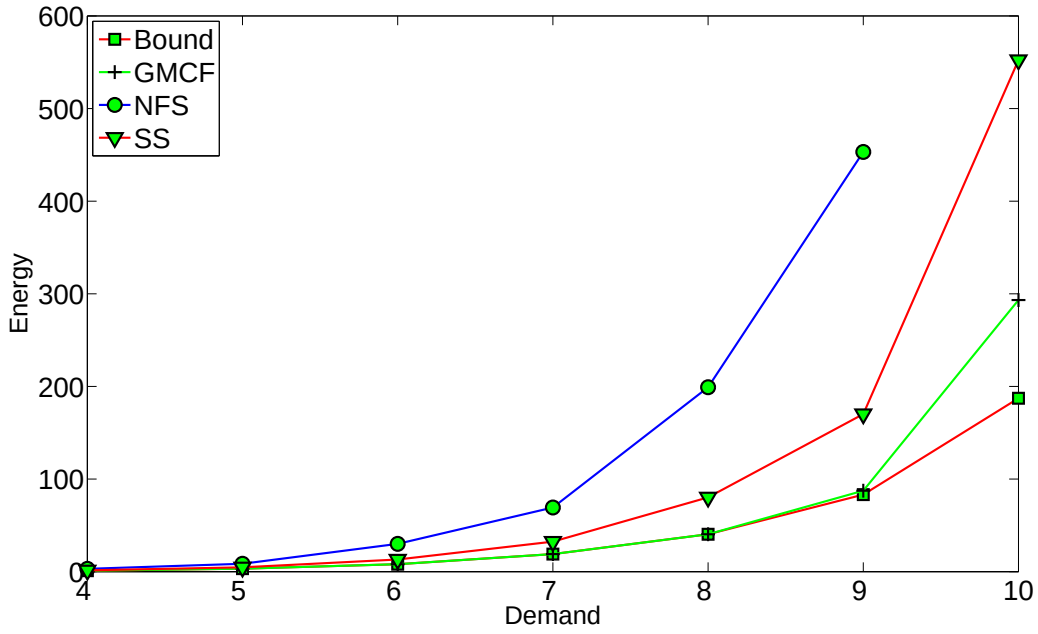

Figure 3.8: Energy Requirements for Three Vehicle Classes Traveling at Different Speeds. No Shadowing Components.

The results shown above consider the case where the schedulers are able to accurately predict the energy costs of each vehicle communication. These results clearly show that it is possible to significantly decrease the energy costs of RSU downlink transmission. In the next set of results we include propagation shadowing effects which results in unpredictable randomness in these estimates. A conventional log-normal shadowing model is used with zero mean and standard deviations of $\sigma \in \{4, 12\}$ dB (Gozalvez *et al.*, 2010) (Rappaport, 1996). The simulation results include three classes of vehicles. The performance of the different algorithms has been tested for various demand levels and different mixes of vehicle class speeds are examined. Class 1 always maintains a speed of 18 m/s for all experiments. Class 2 speeds are 18, 20, 22 and 24 m/s. Class 3 speeds are 18, 23, 28 and 33 m/s. The presented results are the average of 8 random iterations. The energy required by the two shadowing cases is given by the solid and dashed lines, respectively. In the graph legend,

"-s1" and "-s2" are appended to the algorithm names for the two shadowing cases. Note that for these two scenarios we have recomputed the lower bound using knowledge of the path loss values, which gives an absolute bound on system performance.

Figure 3.9 shows the performance when all classes have a speed of 18 m/s and Figure 3.10 when Classes 1, 2 and 3 have speeds of 18, 24 and 33 m/s, respectively. To simplify these two graphs we have not included results for the lower bound. Figure 3.11 shows the required energy to serve vehicles with a communication requirement of 8 timeslots under different velocity mixes. In this graph we have also included the lower bound calculation. An interesting phenomenon occurs when random shadowing components are included. Since the bound is aware of these values, it routinely schedules packet transmissions at greater distances from the RSU, provided that the shadowing gives a favourable path loss. This does not happen in GMCF, SS and NFS because the actual shadowing is not known to the algorithms.

From these figures, it can be seen that increasing the random shadowing component significantly increases the energy required by the RSU compared to the non-random case, as would be expected. However, the relative performance between the different algorithms is maintained, i.e., GMCF is still close to the bound except in situations where there is heavy demand and high differences in vehicle speeds.

When comparing the energy needed to satisfy vehicle communication requirements with and without random shadowing components we find that significant increases in power consumption result from the scheduler's unawareness of these values. However, Figures 3.9, 3.10 and 3.11 indicate that provided there is a known dominant component of path loss, then there is a high value in using the algorithms. These and other results that we have obtained show that although the overall energy costs increase with increases

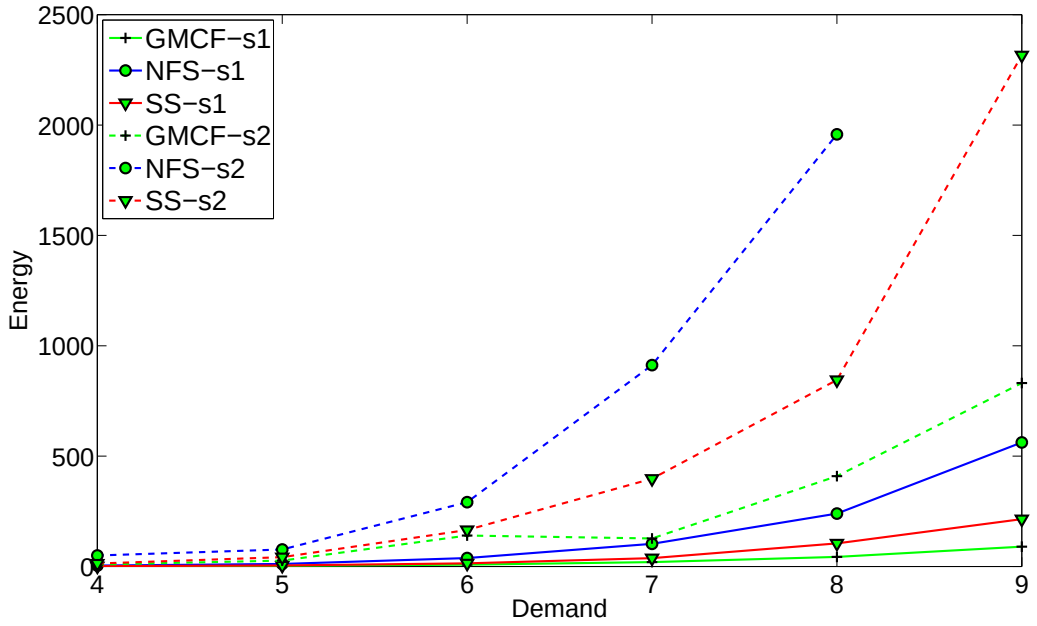

Figure 3.9: Scheduling Performance with Two Levels of Log-normal Shadowing.

in uncertainty, in practical highway scenarios there will clearly be a dominant enough deterministic path loss component that can be used to significantly increase energy savings using the proposed algorithms.

In the final set of results we increased the degree of time slot contention by incorporating vehicular platooning. Platooning is where vehicles follow each other closely in a chain over relatively long time periods. In these experiments, arrivals to the RSU coverage area follow the same arrival process as before but each arrival consists of multiple platooning vehicles. We also assume the same vehicular speed classes and vehicle packet demands as before. In these sets of results we assume the same RSU radio coverage arrangement as before and arriving vehicles can be viewed as approaching from either direction as discussed in Section 3.3.

In Figure 3.12 we show the total energy versus demand for different levels of platooning. In the case of "4 platooning", for example, each arrival consists of 4 vehicles. This

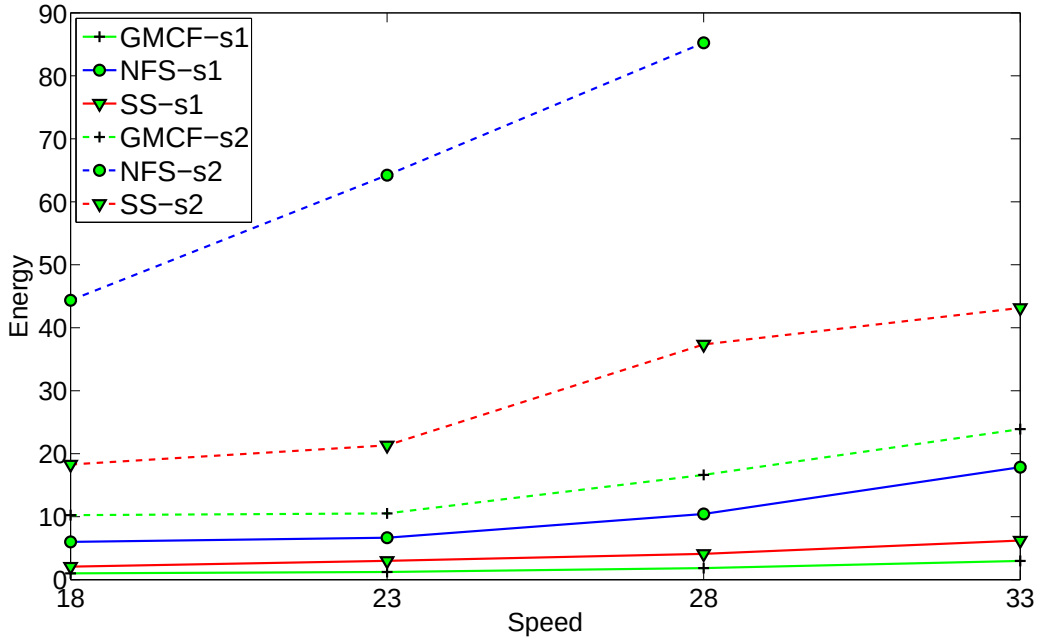

Figure 3.10: Scheduling Performance with Two Levels of Log-normal Shadowing.

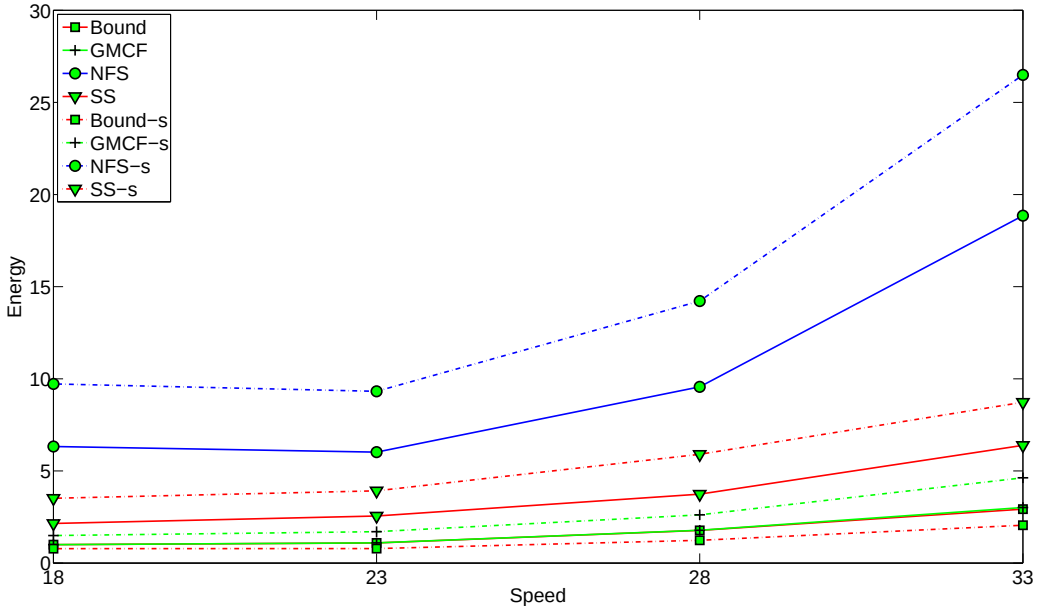

Figure 3.11: Scheduling Performance with Shadowing. Normalized vehicle demand of 8.

was done to create as much scheduling contention as possible. In this set of results we assume the vehicle classes travel at the same speed as in the results from Figure 3.7. In Figure 3.12 there are three families of curves in the graph corresponding to three levels of platooning (i.e., 2, 4 and 6). In each of these sets, results are shown for the proposed algorithms for the no-shadowing (solid lines) and log-normal shadowing cases (dotted lines). Because of the large number of curves in the graph we have chosen not to include the bound but it was computed and compared with the results, giving similar conclusions as we made previously. The graph shows that as platooning increases, the RSU energy requirements increase in proportion, as would be expected. It can also be seen that in all cases the relative performance of the algorithms is roughly the same as in the cases considered previously. This gives additional strength to the conclusions made previously regarding the relative performance of the algorithms.

Another set of results is given in Figure 3.13, but in this experiment, each class of vehicles travel at a different speed than that of the other classes. Class 1, 2 and 3 speeds are 18, 24 and 33 m/s, respectively, which corresponds to vehicles travelling at speeds of 65, 85 and 120 km/h. This arrangement creates more contention than in the Figure 3.12 case, and this is evident in the increase in required energy at comparable demand levels. Other than this, the same assumptions were used as in Figure 3.12 and it can be seen that the same relative comparisons of the algorithm performance hold. Again, this provides additional support for our algorithm performance conclusions.

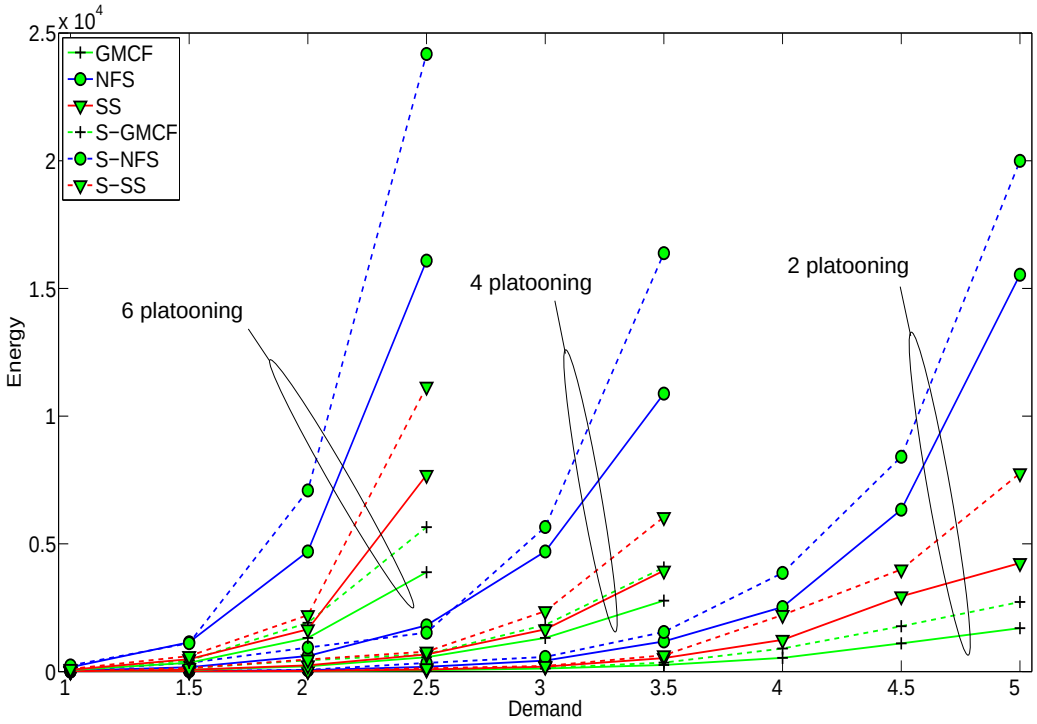

Figure 3.12: Vehicle Platooning Example with Log-normal Shadowing ($\sigma = 4$ dB). Three vehicle classes were used traveling at the same speed as in the results for Figure 3.7. The solid and dashed lines are the no-shadowing and shadowing cases, respectively.

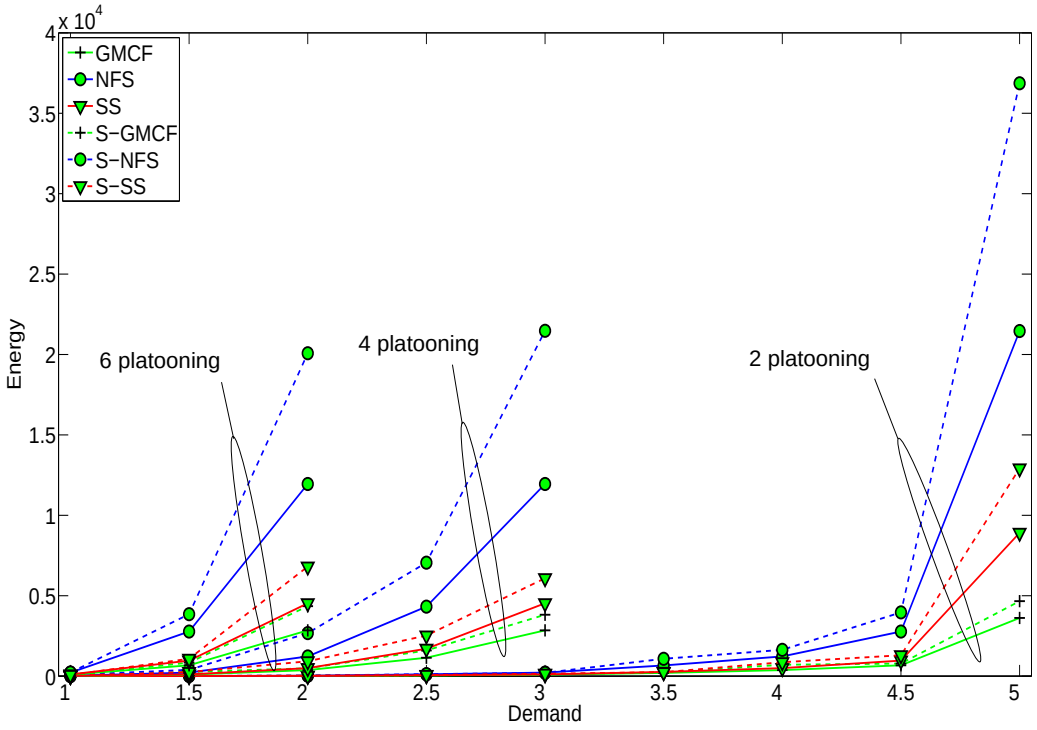

Figure 3.13: Vehicle Platooning Example with Log-normal Shadowing ($\sigma = 4$ dB). Three vehicle classes were used traveling at different speeds as in the results for Figure 3.8. The solid and dashed lines are the no-shadowing and shadowing cases, respectively.

3.7 Conclusions

Roadside infrastructure can be used to provide a wide variety of commercial services for vehicular networks. One particular challenge is that of providing roadside radio coverage in highway locations where wired electricity is not available. In this case, roadside units (RSUs) powered by renewable energy such as solar or wind power, is a viable alternative. The cost of provisioning this type of roadside infrastructure is dependent on the average power consumption of the RSU, and can be significantly reduced by energy efficient scheduling. In this chapter, we have considered the problem of satisfying vehicular communication requirements while minimizing the downlink energy needed by the roadside unit.

The associated scheduling can be either *packet-based* or *timeslot-based*. We first showed that for packet-based scheduling the offline problem can be formulated as a generalization of the classical single-machine job scheduling problem with earliness and tardiness penalties, referred to as α-Earliness-Tardiness. A proof was given which shows that the problem is NP-complete even under a simple distance-dependent exponential radio path loss assumption. In the timeslot-based scheduling case, we formulate the problem as a Mixed Integer Linear Program (MILP) which was shown to be solvable in time which is polynomial in the number of timeslots using a minimum cost flow graph construction. These formulations provide lower bounds on the energy requirements needed to satisfy arriving vehicular communication requests.

The chapter then introduced energy efficient online traffic scheduling algorithms. The first was motivated by a greedy implementation of the MILP formulation which optimizes over finite overlapping time windows. This is referred to as Greedy Minimum Cost Flow (GMCF). Two other algorithms with reduced complexity compared with GMCF were then

proposed. The Nearest Fastest Set (NFS) scheduler uses vehicle location and velocity inputs to dynamically schedule communication activity. The Static Scheduler (SS) performs the same task using a simple position-based weighting function. Results from a variety of experiments show that the proposed scheduling algorithms perform well when compared to the energy lower bounds in vehicular situations where path loss has strong deterministic components so that energy costs can be estimated. Our results also show that near-optimal results are possible but come with increased computation times compared to our heuristic algorithms.

Chapter 4

Energy Efficient Scheduling for Variable Bit Rate Channels

4.1 Introduction

In this chapter we focus on the problem of downlink schedule generation when the RSU-to-vehicle radios use a variable bit rate (VBR) air interface, rather than the constant bit rate case considered in Chapter 3. We first present offline scheduling formulations that provide lower bounds on the energy required to fulfill vehicle requests. An integer linear program (ILP) is introduced which can be solved to find optimal offline VBR time slot schedules. We then prove that this problem is NP-complete by a reduction from the well-known Santa Claus Problem. Two flow graph based models are then used to solve the minimum energy VBR scheduling problem. The first uses Generalized Flow (GF) graphs which represent time slots as individual graph nodes. The second uses time expanded graphs (TEGs) which model the temporal evolution of the system. Both of these models provide lower bounds on energy performance and provide the basis for energy efficient online schedulers. The

first scheduler introduced, First Come First Serve (FCFS) is very simple and treats all vehicles equally and in order of arrival. The second, Fastest First (FF), gives priority to faster moving vehicles since their transit time through the RSU coverage area is less than slower moving vehicles. The Greedy Generalized Flow (G-GF) and Greedy Time Expanded Graph (G-TEG) algorithms are two implementations of the two flow graph based models. The proposed algorithm performance is examined under different traffic scenarios and they are found to perform well compared to the lower bound. Our results show that less computational intensive algorithms, FCFS and FF, can perform well under light load, but G-GF and G-TEG, while more computational intensive, can provide near optimal throughput and with reasonable drop rates. The results also show that the G-GF and G-TEG algorithms are much more fair than the simpler algorithms in heavy load situations. The remainder of the chapter is organized as follows. Section 4.2 summarizes related research results. Section 4.3 described the assumptions used in our system model and Section 4.4 presents the ILP formulation of our problem for the variable bit rate time slot case. In Section 4.4.1 this scheduling problem is shown to be NP-complete. Following this, in Section 4.4.2 the offline energy bounds are introduced. In Section 4.5, four online scheduling algorithms are then proposed. The bound and the online algorithms performance are studied in Section 4.6 and in Section 4.7, the conclusions of our work are presented.

4.2 Related Work

Recent vehicular ad hoc networking work has covered a broad range of topics. For example, application requirements and their relation to networking technologies are explored in (Khaled *et al.*, 2009), and reference (Li and Wang, 2007) surveys routing protocols and related mobility models. The specific challenges relating to vehicular security have been

studied in reference (Zhang *et al.*, 2009). In (Bychkovsky *et al.*, 2006), (Mittag *et al.*, 2008) and (Ott and Kutscher, 2004), the performance of the IEEE 802.11p standard in highway environments was studied. Extending vehicular ad hoc network connectivity, improving utilization and decreasing contention were examined in (Jhang and Liao, 2008), (Zhao *et al.*, 2008) and (Nandan *et al.*, 2005).

Roadside unit traffic scheduling has been studied using simple schedulers based on message size and their deadlines but without considering the energy consumption of the infrastructure (Zhang *et al.*, 2007). Reference (Chen *et al.*, 2012) maximizes RSU throughput given messages with different priorities. The RSU channels are viewed as machines and messages as jobs using a model based on parallel machines. In (Alcaraz *et al.*, 2009), an RSU throughput optimization is used given the locations and velocities of the vehicles. A scheduler was also proposed which can be used in the IEEE 802.11e contention free period.

Energy efficiency in VANETs has typically not been an issue, since vehicles are usually assumed to have unlimited energy reserves. Also, from the roadside infrastructure viewpoint, most work assumes urban settings where wired power is available at reasonable cost. However, recent work addresses the energy consumption needed by a group of RSUs by switching them on and off in order to maintain connectivity for the vehicles while minimizing RSU energy use (Zou *et al.*, 2011). The purpose of this chapter is to optimize the RSU energy consumption, but with a focus on smart scheduling at the RSU.

4.3 System Description

The system considered consists of a single Roadside Unit (RSU) which communications with passing vehicles as shown in Figure 4.1. The figure shows the vehicles moving in

one direction, but multi-directional traffic is also easily accommodated. The focus is on the energy required by the radio interface used by the RSU to fulfill vehicle communication requirements in the downlink direction. We are not concerned about vehicular energy efficiency since the vehicle radio is powered by the car's engine.

The objective for the RSU is to generate and execute downlink transmission schedules that fulfill vehicular communication requirements. We assume that a vehicle's communication requirement is delay tolerant in that it can be satisfied at any point during the vehicle's RSU transit time (Hammad *et al.*, 2013). Some examples of delay tolerant applications are on-demand-video, file transfer, music and radio streaming. Upon entering the RSU range, each vehicle announces its requirements, location and velocity to the RSU. This information can then be used to estimate the downlink energy costs of communicating with the vehicle as it passes through the coverage area. The objective is to use these inputs to create a downlink transmission schedule so that energy use is minimized. In (Hammad *et al.*, 2013) schedule generation was considered when the air interface uses a constant bit rate (CBR). In the CBR case, power control is used to maintain a fixed bit rate over the RSU-to-vehicle links as the channel path loss changes. In this case the objective is to schedule downlink transmissions such that the required average downlink *power consumption* is minimized. However, as in (Azimifar, 2013), we assume a variable bit rate (VBR) air interface where the transmission power is assumed to be fixed, and changes in channel path loss are compensated for by varying the bit rate used over the RSU-to-vehicle links. A packet transmission may therefore occupy a different amount of time on the channel depending upon the bit rate used when it is transmitted. In this case, the objective is therefore to create a downlink transmission schedule such that the total time that the RSU spends transmitting is minimized. Note that the bit rate needed can be set in a variety of ways such

as using a short two-way handshake prior to actual vehicle data packet transmission. The schedule generation problem is more formally stated as follows.

INPUT: The scheduler is given a sample function of V vehicles indexed by the set $\mathcal{V} = \{1, 2, \ldots, V\}$, where each vehicle $v \in \mathcal{V}$ has a residual RSU-to-vehicle communication requirement, d_v, bits. We consider two different transmission arrangements.

1. Time Slotted (TS): The entire time over which scheduling is to occur consists of T fixed duration time slots given by the set $\mathcal{T} = \{1, 2, \ldots, T\}$. The scheduler is given estimates of the number of bits that can be carried in each time slot $t \in \mathcal{T}$ for vehicle v, denoted by $b_{v,t}$. This is referred to as the variable bit rate *time slotted* or TS case.

2. Time Framed (TF): The entire time over which scheduling is to occur consists of transmission frames given by the set $\mathcal{F} = \{1, 2, \ldots, F\}$ and their duration, τ_f, for all $f \in \mathcal{F}$. We are also given estimates of the bit rate that can be achieved in each frame f, for vehicle v, denoted by $B_{v,f}$. This is referred to as the *time framed* or TF case. Time frame durations are such that $B_{v,f}$ can be assumed constant for a given vehicle v during frame f.

OUTPUT: A scheduler output gives a downlink RSU-to-vehicle transmission schedule. Given the inputs defined above, the scheduler will try to generate a downlink transmission schedule so that all vehicle communication requirements are satisfied, and so that the downlink RSU energy cost is minimized, i.e., the scheduler tries to minimize the total time needed to fulfill all downlink vehicular transmissions. In the TS case the scheduler output gives $x_{v,t}$ for all $v \in \mathcal{V}$ and $t \in \mathcal{T}$, which are binary decision variables equal to 1 if vehicle v is allocated to time slot t, and 0 otherwise. In the TF

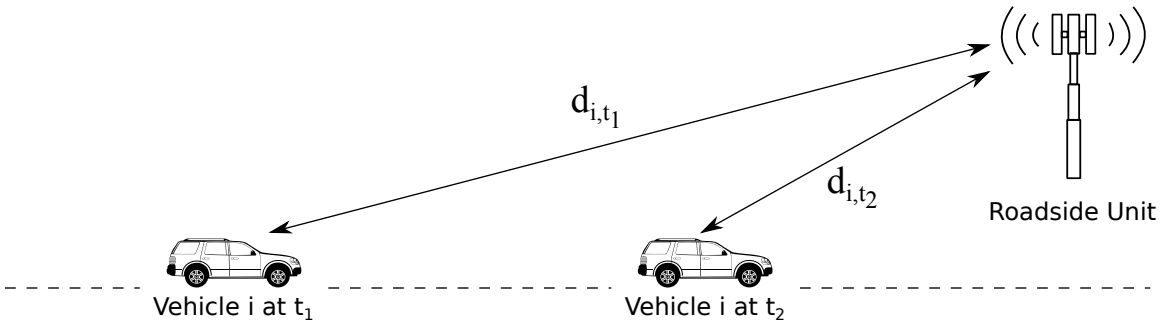

Figure 4.1: RSU Example. Vehicle i is shown at two different times t_1 and t_2 at distances d_{i,t_1} and d_{i,t_2} from the RSU, respectively

case, the scheduler output gives $x_{v,f}$ for all $v \in \mathcal{V}$ and $f \in \mathcal{F}$, which is the *number of bits* which are sent to vehicle v in time frame f.

The schedules that are generated can be either offline or online. In *offline scheduling*, the entire set of inputs for all $t \in \mathcal{T}$ (or for all $f \in \mathcal{F}$) are provided to the scheduler at once. Optimum offline schedules are derived in Section 4.4.2 and are used as lower bounds on energy performance for comparisons with the online algorithms. In *online scheduling* the inputs are provided to the scheduler in real time and it must create a schedule in real time using these causal inputs. Online scheduling algorithms are given in Section 4.5. In the following section, we formalize these definitions and derive optimum offline schedules for the energy needed for RSU-to-vehicle communication.

4.4 Offline Variable Bit Rate Scheduling

The offline scheduling task is modelled as an Integer Programming (IP) optimization. We first consider the time slotted (TS) model where the offline scheduling problem can be

written as an ILP, referred to as VBR-OPT, as follows.

$$\underset{x_{v,t}}{\text{minimize}} \quad \sum_{v\in\mathcal{V}}\sum_{t\in\mathcal{T}} x_{v,t} \tag{VBR-OPT}$$

$$\text{subject to} \quad \sum_{v\in\mathcal{V}} x_{v,t} \leq 1, \qquad \forall t \in \mathcal{T} \tag{4.1}$$

$$\sum_{t\in\mathcal{T}} b_{v,t}x_{v,t} \geq d_v, \quad \forall v \in \mathcal{V} \tag{4.2}$$

$$x_{v,t} \in \{0,1\}, \qquad \forall t \in \mathcal{T}, v \in \mathcal{V} \tag{4.3}$$

In VBR-OPT, $x_{v,t}$ are binary decision variables equal to 1 if vehicle v is allocated to time slot t and 0 otherwise. The objective is to minimize the total number of time slots the RSU will be transmitting. Constraint (4.2) ensures that the number of time slots allocated to each vehicle satisfies the vehicle demand, while Constraint (4.1) prevents time slots from being assigned more than once.

Note that since optimization VBR-OPT is an ILP, it may have an exponential worse-case complexity, which precludes its use for reasonable problem sizes. In fact, we show next that VBR-OPT is NP-complete. However, we can generate a less complex lower bound on energy by relaxing constraint (4.3) to $x_{v,t} \geq 0$ for all $t \in \mathcal{T}, v \in \mathcal{V}$. This permits us to obtain a bound by solving a conventional linear program. This is discussed further in Section 4.4.2.

4.4.1 Time Slotted Variable Bit Rate Scheduling Complexity

In this section we prove the NP-completeness of time slotted version of the offline scheduling problem formulated in Section 4.4. This is done by a reduction from the well known Santa Claus Problem (Bansal and Sviridenko, 2006). In the Santa Claus Problem, Santa

has a set $\mathcal{G}$ of presents (gifts) that he wants to distribute among a set $\mathcal{K}$ of kids. There is a given level of satisfaction, $p_{k,g}$, associated with giving present g to kid k, for all $g \in \mathcal{G}$ and $k \in \mathcal{K}$. The total satisfaction for a kid consists of the sum of satisfactions of each present received. Given these inputs, the decision version of the Santa Claus Problem must answer the question: *Given a satisfaction threshold, S, is there an assignment of presents to the kids such that each kid receives at least a satisfaction of S?*

The decision version of the Santa Claus Problem can be easily written as an integer linear program feasibility test as follows.

$$\begin{aligned}
&\underset{x_{k,g}}{\text{minimize}} \quad 0 && && \text{(SC-OPT)}\\
&\text{subject to} \quad \sum_{k\in\mathcal{K}} x_{k,g} \leq 1, && \forall g \in \mathcal{G} && (4.4)\\
&\qquad\qquad \sum_{g\in\mathcal{G}} p_{k,g}\, x_{k,g} \geq S, && \forall k \in \mathcal{K} && (4.5)\\
&\qquad\qquad x_{k,g} \in \{0,1\}, && \forall g \in \mathcal{G}, k \in \mathcal{K} && (4.6)
\end{aligned}$$

where $x_{k,g}$ is a binary decision variable, equal to 1 when kid k is given present g and 0 otherwise. Constraint (4.4) ensures that each present, is assigned to at most one kid and Constraint (4.5) guarantees that the each kid achieves the minimum satisfaction threshold, S. We now show the following result.

Theorem 1. *The time slotted variable bit rate vehicular scheduling problem is NP-complete.*

Proof. We first divide both sides of Constraint (4.5) by S, and define $p'_{k,g} = p_{k,g}/S$. This constraint now becomes

$$\sum_{g\in\mathcal{G}} p'_{k,g}\, x_{k,g} \geq 1, \quad \forall k \in \mathcal{K} \tag{4.7}$$

where $p'_{k,g}$ can be viewed as a normalized satisfaction. Similarly, we divide both sides of

Constraint (4.2) by d_v and define $b'_{v,t} = b_{v,t}/d_v$. This constraint now becomes

$$\sum_{t \in \mathcal{T}} b'_{v,t}\, x_{v,t} \geq 1, \quad \forall v \in \mathcal{V} \tag{4.8}$$

where $b'_{v,t}$ can be viewed as a normalized time slot capacity. Note the similarity that these two normalized constraints create for ILPs VBR-OPT and SC-OPT.

Now assume that we are given an instance of the Santa Claus Problem defined above. We define an instance of the variable bit rate scheduling problem where $|\mathcal{V}| = |\mathcal{K}|$ and $|\mathcal{T}| = |\mathcal{G}|$. We set $b'_{k,g} = p'_{k,g}$ for all $k \in \mathcal{K}$ and $g \in \mathcal{G}$. Clearly this transformation can be done in linear time.

Using the above inputs we now solve ILP VBR-OPT. If a valid solution is obtained, then from the above construction, the time slot assignments from ILP VBR-OPT are clearly a feasible solution to ILP SC-OPT. Now assume that ILP VBR-OPT is infeasible, i.e., no minimum energy solution to VBR-OPT exists. If a solution to the Santa Claus Problem instance exists however, then from the same construction, the solution from ILP SC-OPT could be used to create a feasible solution for ILP VBR-OPT. This contradicts the infeasibility of the ILP VBR-OPT solution. Therefore, the solution to any instance of the Santa Claus Problem can be answered by testing if a solution to the variable bit rate scheduling problem exists. This establishes the NP-completeness of the variable bit rate scheduling problem. □

4.4.2 Offline Scheduling using Generalized and Time Expanded Graphs

In this section we introduce two methods to compute the minimum energy that an optimal offline scheduler can achieve. Since they are offline, the entire input is given to the scheduler at once, and the schedulers may produce fractional time slot allocations. In Section 4.5 we adapt these models for online use. The first proposed method models the time slotted (TS) version of the scheduling problem as a Minimum Cost Generalized Flow Graph. Recent combinatorial algorithms (Wayne, 2002) can solve this type of problem in time complexity which is strongly polynomial compared to general purpose LP optimization solvers which at best can achieve weakly polynomial times. The second method models the time framed (TF) version of the scheduling problem as a Time Expanded Graph.

Generalized Flow (GF) Graph Model

In this section, we introduce an offline scheduling method using a generalized flow graph construction for the TS model. This technique has advantages over solving an LP relaxation of ILP VBR-OPT in that due to the cycle cancelling algorithms used for its solution (Wayne, 2002), it directly manipulates the underlying graph and tends to generate integral time slot allocations. This, combined with careful design of the generalized flow graph model for our problem has led to a dramatic decrease of the number of time slots that the scheduler shares between multiple vehicles. By reducing the percentage of shared time slots, the difference between the solutions obtained by the approximate schedule and an optimal IP scheduler is minimized. This is discussed further in Section 4.5. In addition, when general purpose LP methods are used to find an optimal schedule for a problem modeled as a generalized flow, they are weakly polynomial. Their complexity will depend on the size of the cost, capacity and gain factors (Wayne, 2002), while combinatorial algorithms

designed specifically for Generalized Flow problems can be strongly polynomial.

Generalized Flow Graphs (GFGs) (Ahuja *et al.*, 1994) are similar to conventional flow graphs in that they consist of nodes interconnected by directional edges. As in conventional flow graphs, each node can introduce flow into the graph (supply), subtract flow from the graph (demand), or be a transship node (the sum of flows entering the node is the same as the sum of the flows going out). Each GFG edge has a maximum flow capacity which must not be exceeded. The edges also have costs associated with the use of that edge, and for each unit of flow along an edge, the total edge cost incurred is the product of the flow along the edge and the edge cost. A feasible flow across the graph must satisfy all the nodes supply and demand, as well as flow conservation, while not violating the edge capacities.

Generalized flow graphs differ in that each edge also has a *multiplier*, which is referred to as the edge gain factor. Edge (i, j) has a multiplier, $\gamma_{i,j}$, and flow f entering this edge will emerge at the other end of the edge as flow $f\gamma_{i,j}$. The gain factors can be less (or greater) than 1 in which case the flow reaching the destination is reduced (or magnified), respectively.

We will present a simple example to show how the variable bit rate scheduling problem can be modelled as a GFG. For this example we assume a simplified case where only three bit rate levels are used, high (H), medium (M) and low (L). Clearly, the bit rate that can be used for a given vehicle will depend upon the quality of the downlink during a time slot. In the example we assume that a flow of 2 (in units of high bit rate time slots) is required for vehicle v. We also assume that two and three M and L time slots are needed to accommodate one H bit rate time slot, i.e., a flow of 1 H time slots can be serviced by 2 M bit rate time slots, etc. A section of the final generalized flow graph is shown in Figure 4.2. The input flow to node v is shown at the left. The L, M and H nodes correspond to

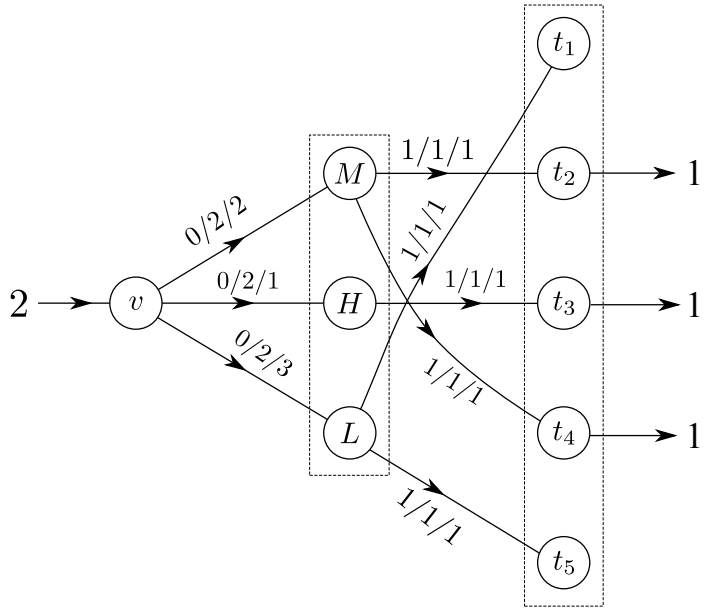

Figure 4.2: Generalized Flow Graph Illustration. Part of the graph is shown for a single vehicle v. The format of the edge labels is given by the triplet "cost/capacity/multiplier".

using time slots with those bit rates. Each node in the right hand column corresponds to a time slot, and the edges connecting the middle and right hand columns correspond to the bit rate which is possible for v during that time slot. The multipliers on the edges leaving node v correspond to the relative bit rates of the center column nodes used. For example, the v-to-M multiplier is 2 which will cause a unit flow input from v to become a two unit flow leaving node M. This will ensure that two M time slots will be required to satisfy this flow. In the example, the 2 units of flow entering v emerge as three units of flow from the time slot column. This indicates that one unit of flow each is passing along the H-to-t_3, M-to-t_2 and M-to-t_4 edges. The requirement for v has therefore used one H time slot and two M time slots. Note however, that the amount of flow that is leaving the section of the graph shown is not the same as the flow entering. The cost for a particular solution comes from the middle to right hand column edges, each of which are unit costs. The total cost

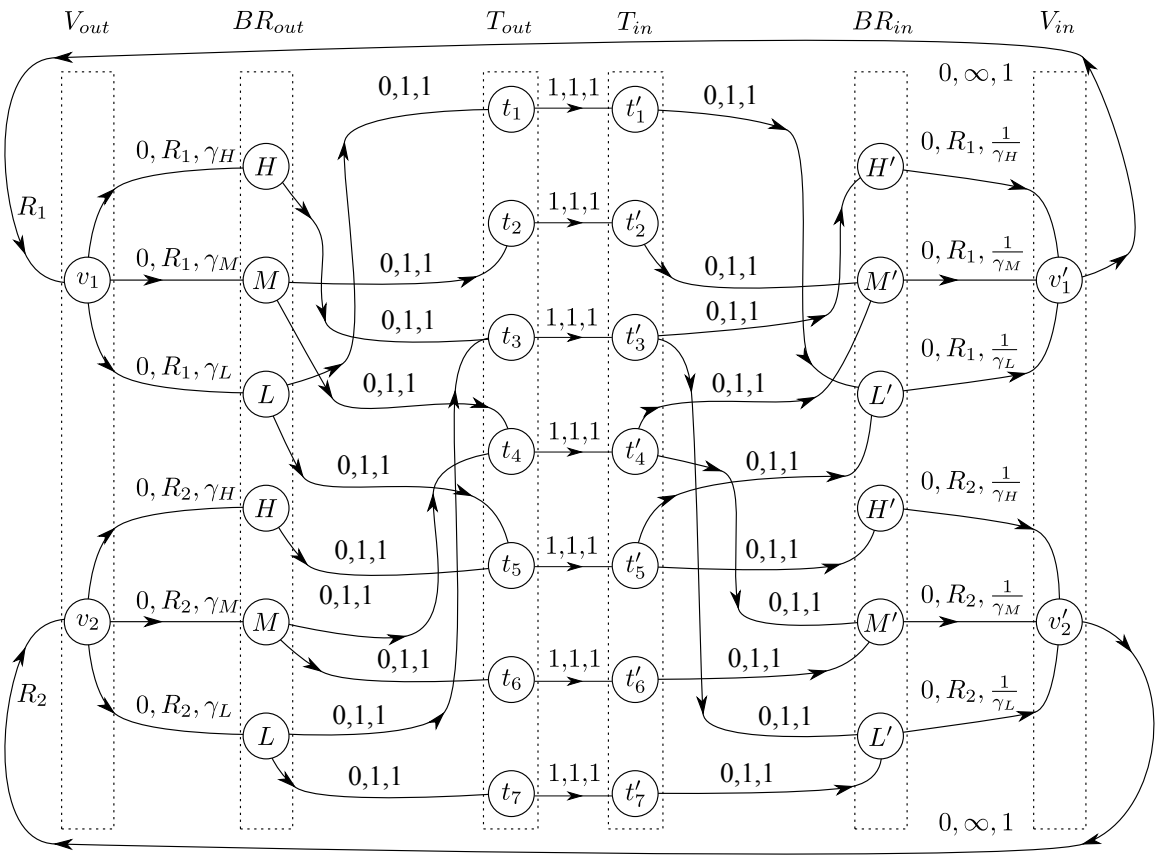

Figure 4.3: Generalized Flow Graph Model Example

for this solution is therefore 3.

In general a vehicle demand R is represented in terms of multiples of the highest bit rate level used for a full time slot. If vehicle v demand can be served during time slots connected to the H node, it can receive the highest bit rate achievable. Thus, it can be served in exactly R time slots. The multiplier on the edges connecting v to H, $\gamma_{v,H}$, is equal to 1. This is because a unit demand can be served by a unit of flow from H to the time slot node t_3, assuming the second highest bit rate level, M, is half the H bit rate. Serving vehicle v demand using only time slots connected to M would require the use of $2 \times R$ time slots, t_2 and t_4. This is reflected in the multiplier γ_M being equal to 2. The same

idea applies to lower bit rates. So we have $\gamma_H < \gamma_M < \gamma_L$. The multipliers over edges connecting nodes H, M and L and time nodes are set to be 1.

For algorithmic convenience reasons, *circulations* instead of flows are used. To achieve this, the above design is "mirrored" as shown in Figure 4.3. In this example we show a complete graph with the same time slot bit rates as before, showing 7 time slots and two vehicles. By mirroring, the flow coming from vehicle nodes in column $Vout$ flows to bit rate level nodes in column $BRout$ to time slots column $Tout$. At this stage the mirrored design starts where flow from $Tout$ nodes pass through Tin nodes. This in turn is collected in the bit rate level nodes in the column $BRin$ and finally to Vin vehicle nodes. The feedback flow allows excess flow to be fed back into the network, forming a circulation.

The values of capacity and multiplier over the edges in the mirrored part of the graph are kept the same, except for the edges connecting the mirrored bit rate level nodes $BRin$ to vehicles in the Vin column where the value of the multipliers are the inverse of those connecting $Vout$ vehicles nodes to $BRout$. In this design, the same time slot can be used by any vehicle that happens to be inside the RSU range during this time slot. In order to limit the flow from multiple vehicles into a single time slot to one unit, the mirror column of time slots is used where each time slot t'_i is connected to t_i with an edge capacity limited to 1.

This design allows combinatorial algorithms to detect flow generating cycles and flow absorbing cycles (Wayne, 2002). Together they form what are called "bicycles" which can be eliminated when searching for a minimum cost flow solution. After solving this minimum cost flow problem, the final schedule can be obtained from the flows between the bit rate level nodes $BRout$ and time nodes $Tout$. However, note that generalized flow graph solutions do not produce integral flows, so optimum solutions may allocate fractional

time slots. In Section 4.5 we propose a method based on rounding that generates integral solutions which can be used in practical schedulers.

Time Expanded Graph (TEG) Model

The generalized flow graph mode discussed in Section 4.4.2 specifically identifies time slots which are then assigned by the GFG optimization. In this section we consider the time framed (TF) version of the problem modeled as a time expanded graph. An advantage of this approach is that fractional flow assignments will be in units of bits rather than units of time slots. This makes it easier to accommodate practical packet constraints. Also, typically the achievable bit rate for a given vehicle does not change over many time slots, which means that the GFG size can be unnecessarily large.

Unlike a conventional flow graph, a TEG is time-expanded, i.e., the graph models the evolution of the system in time (Kotnyek, 2003). Accordingly, we consider the system to evolve over multiple time frames. An example of a TEG is shown in Figure 4.4 consisting of three time frames. In this example there are three vehicles V_1, V_2 and V_3 with downlink demands of 7, 5 and 15, respectively. The graph is time-expanded which means that the vehicle nodes and the RSU node are duplicated across the three time frames, as shown. The temporal node replications also model the carrying of flow (i.e., bits) over from one frame to the next, e.g., the RSU may fulfill a vehicle's job requirement over multiple frames. Note that node D is simply a flow collection node. The cost of communicating between the RSU and vehicle i at time frame j is denoted by $c_{i,j}$. When the vehicle is closest to the RSU and the achievable bit rate is highest, the cost is lowest, and as the achievable bit rate drops, the cost increases. The edge capacity denotes the maximum number of bits that can be transmitted to vehicle i if it used the entire frame j. The cost over this arc is inversely

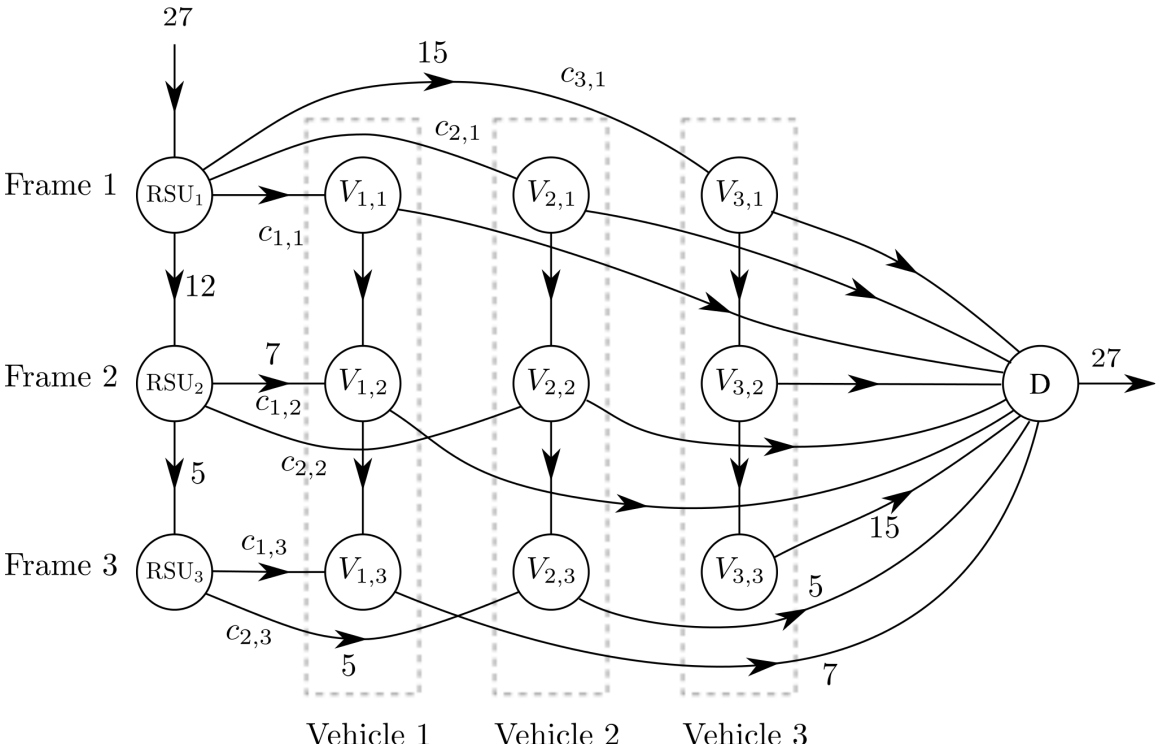

Figure 4.4: Time Expanded Graph Example

proportional to the bit rate $B_{i,j}$ vehicle i can achieve during frame j.

The TEG can easily be expressed in terms of the following LP.

$$\begin{aligned}
&\underset{x_{v,f}}{\text{minimize}} && \sum_{f\in\mathcal{F}}\sum_{v\in\mathcal{V}} x_{v,f}/B_{v,f} && && \text{(TEG-LP)}\\
&\text{subject to} && \sum_{f\in\mathcal{F}} x_{v,f} \geq d_v && \forall v\in\mathcal{V} && (4.9)\\
& && \sum_{v\in\mathcal{V}} x_{v,f}/B_{v,f} \leq \tau_f && \forall f\in\mathcal{F} && (4.10)\\
& && x_{v,f} \geq 0 && \forall v\in\mathcal{V}, f\in\mathcal{F} && (4.11)
\end{aligned}$$

where $\mathcal{V}$ is the set of vehicles, $\mathcal{F}$ is the set of time frames, $x_{v,f}$ is the number of bits sent to vehicle $v \in \mathcal{V}$ during frame $f \in \mathcal{F}$ and $B_{v,f}$ is the achievable bit rate to this vehicle during

this time frame. The objective in TEG-LP minimizes the total time that the RSU transmits. Constraint (4.10) ensures that the sum of all time durations allocated to vehicles during a frame f does not exceed the frame duration, τ_f. Vehicle v demand d_v (in bits) is guaranteed to be satisfied by Constraint (4.9). This LP can be solved in polynomial time and provides an offline bound to the energy required to serve the vehicles in $\mathcal{V}$ during time period $\mathcal{F}$.

4.5 Online Scheduling Algorithms

In this section, algorithms are proposed which perform energy efficient online scheduling in real time. The scheduling decisions made by the online scheduler can only be made using causal input data, i.e., consisting of past and current inputs.

4.5.1 First Come First Serve (FCFS) Algorithm

FCFS is a very low complexity scheduler motivated by the CBR version proposed in (Hammad *et al.*, 2010). The algorithm is executed whenever a new vehicle enters the RSU radio coverage range. Upon arrival, the scheduler assigns enough time slots to satisfy the vehicle's communication requirements. This is done by assigning time slots in "highest bit rate first" order for the arriving vehicle. A more formal description of the algorithm is shown in Algorithm 1.

In step 2 the algorithm executes the current schedule, $x_{k,t}$, for the current time slot, t. If $x_{k,t} = 1$, vehicle k is communicated to by the RSU for time slot t. Vehicle k therefore receives $b_{k,t}$ bits which is the available payload. The demand for vehicle k, d_k, is then reduced by $b_{k,t}$.

In step 5, if a vehicle v arrives, the current schedule is updated. This update does not

Algorithm 1 FCFS Algorithm

1: **for all** $t \in \{0, 1, \ldots\}$ **do**
2: **if** time slot t is assigned to vehicle k, i.e., $x_{k,t} = 1$ **then**
3: Transmit to vehicle k.
4: **end if**
5: **if** vehicle v arrives **then**
6: Set residual demand, r_v, to the vehicle v demand d_v.
7: **while** $r_v > 0$ **do**
8: Select an unassigned time slot, t', for vehicle v which maximizes its bit rate, i.e., $t' = \arg\max_{t \in \mathcal{T}_v} b_{v,t}$, where $\mathcal{T}_v$ is the set of unused time slots available to v.
9: Assign t' to vehicle v, i.e., $x_{v,t'} = 1$.
10: Reduce r_v by the payload available for vehicle v in time slot t', i.e., $r_v = \max(0, r_v - b_{v,t'})$.
11: **end while**
12: **end if**
13: **end for**

interfere with anything previously allocated for other vehicles. A temporary variable r_v is initially set to vehicle v demand d_v. In step 8, the algorithm searches for a free time slot t' during which newly arrived vehicle v can receive the highest possible bit rate from the RSU. Once this time slot is found, it is reserved in the schedule $x_{t',v}$ for some future time slot t'. This process is repeated, steps 10 to 14, until vehicle v is assigned enough time slots to satisfy its demand.

The description in Algorithm 1 is written in terms of variable bit rate time slots. However, the algorithm can also operate using the time framed (TF) model with very little change, i.e., instead of assigning maximum data rate time slots, variable length time periods are assigned in frames at the maximum data rate possible.

In Section 4.6 it will be shown that FCFS is most suitable when vehicle speeds and demands are relatively close and traffic load is light. When vehicle speeds are quite different, there are advantages to distinguishing them on this basis. This is the FF algorithm

introduced in the next section, and motivated by a similar algorithm in (Hammad *et al.*, 2010) used in the CBR air interface case.

4.5.2 Fastest First (FF) Algorithm

In this algorithm, the schedule is re-computed upon each vehicle arrival. Time slots are assigned using the same "highest bit rate first" priority as in the FCFS algorithm, however, the vehicles are processed in "fastest-first" order. As before, each vehicle is allocated enough time slots to satisfy its remaining demand. A more formal description of the algorithm is given in Algorithm 2.

Algorithm 2 FF Algorithm

1: **for all** $t \in \{0, 1, \ldots\}$ **do**
2: **if** time slot t is assigned to vehicle k, i.e., $x_{k,t} = 1$ **then**
3: Transmit to vehicle k.
4: **end if**
5: **if** a new vehicle arrives **then**
6: Reset schedule: $x_{k,t} = 0$ for all $k \in \mathcal{K}, t \in \mathcal{T}$
7: **while** unscheduled vehicles exist **do**
8: Select vehicle v with highest speed from unscheduled vehicles.
9: Set residual demand r_v to the vehicle v demand d_v.
10: **while** $r_v > 0$ **do**
11: Select an unassigned time slot, t', for vehicle v which maximizes its bit rate, i.e., $t' = \arg\max_{t \in \mathcal{T}_v} b_{v,t}$, where $\mathcal{T}_v$ is the set of unused time slots available to v.
12: Assign t' to vehicle v, i.e., $x_{v,t'} = 1$.
13: Reduce r_v by the payload available for vehicle v in time slot t', i.e., $r_v = \max(0, r_v - b_{v,t'})$
14: **end while**
15: **end while**
16: **end if**
17: **end for**

As in the FCFS algorithm, in step 2, the algorithm executes the current schedule, $x_{k,t}$, for the current time slot, t. If $x_{k,t} = 1$, vehicle k is communicated to by the RSU. Vehicle

k receives $b_{k,t}$ bits during that time slot. The demand for vehicle k, d_k, is then reduced by $b_{k,t}$.

In step 5, if a new vehicle v arrives, the current schedule is re-computed. In step 8, the vehicles are selected to be scheduled according to their speed. A temporary variable r_v is set to the currently selected vehicle v's remaining demand d_v. In step 11, the algorithm assigns a free time slot t' during which vehicle v can receive the highest possible bit rate from the RSU. This process is repeated, steps 10 to 14, until vehicle v is assigned enough time slots to satisfy its remaining demand d_v. All other vehicles are then processed in the same fashion. As in the FCFS algorithm, FF assumes fixed size time slots.

The description in Algorithm 2 is written in terms of variable bit rate time slots. As in the FCFS case, the algorithm can also operate using the time framed (TF) model with very little change, i.e., instead of assigning maximum data rate time slots, variable length time periods are assigned in frames at the maximum data rate possible.

The idea behind this algorithm is to provide better fairness across vehicles with different speed, since faster moving vehicles spend less time inside the RSU radio coverage range. Note that FF is more computationally expensive than FCFS, but as it will become clear later, it provides better fairness when there are vehicles with different speed.

4.5.3 Greedy Generalized Flow (G-GF) Algorithm

In section 4.4.2 we introduced two designs that compute bounds on the minimum energy of the offline scheduling case. These algorithms can be run in polynomial time, and in this section a greedy online version of these algorithms is proposed. The Greedy Generalized Flow Algorithm (G-GF) is based on the Generalized flow construction shown in Figure 4.3. Since it is an online algorithm it must schedule vehicles as they arrive to the RSU. The

algorithm is shown in Algorithm 3 and is described as follows.

Algorithm 3 includes a function declaration, GFG-SCHEDULER, which accepts a set of vehicles $\mathcal{V}$, time slots $\mathcal{T}$, remaining vehicle demand d_i and the number of bits available to vehicle i in time slot j, $b_{i,j}$. When this function is called, it constructs a generalized flow graph as in Figure 4.3 using the provided inputs. The GFG is then solved and the resulting outputs, $x_{v,t}$, become the new active schedule. As discussed previously, these outputs may be fractional.

When a new vehicle arrives (step 3), GFG-SCHEDULER is called and an updated schedule is obtained using all known information. This includes all known vehicles and the union of time slots available to this vehicle set. The algorithm then tests to see if any vehicles are scheduled for the current time slot, t, and if multiple vehicles are, it selects the one with the highest data rate, and transmits to that vehicle (step 8). Vehicle demand is then updated and if the vehicle in question is finished, the vehicle set is also updated. In step 10, if the vehicle had not been the only one assigned to time slot t, then a new schedule is obtained from GFG-SCHEDULER by excluding time slot t. Note that the same process occurs if a vehicle was assigned a fractional time slot, except that there is no need to call GFG-SCHEDULER since the system will use the existing schedule.

4.5.4 Greedy Time Expanded Graph (G-TEG) Algorithm

In this section we introduce an algorithm based on the design described in Figure 4.4. The idea is to create a time expanded graph for vehicles inside the RSU range. Upon vehicle arrival the scheduler creates a time expanded graph similar in design to the one in Figure 4.4. The vehicle nodes represent the ones currently inside the RSU range with unmet demands. The time frames set represents the frames during which these vehicles

Algorithm 3 G-GF Algorithm

$\mathcal{V}$: set of known vehicles
$\mathcal{T}$: set of time slots available to $\mathcal{V}$
d_v : vehicle demand $\forall v \in \mathcal{V}$
$b_{v,t}$: bits available to vehicle v in time slot t $\forall v \in \mathcal{V}, t \in \mathcal{T}$
$x_{v,t}$: current schedule

for all $t \in \{0, 1, \ldots\}$ **do**
  if a new vehicle arrives **then**
    GFG-SCHEDULER($\mathcal{V}, \mathcal{T}, d_i, b_{i,j}$) $\forall i \in \mathcal{V}, j \in \mathcal{T}$
  end if
  Let $\mathcal{K}_t$ be the set of all vehicles scheduled for time slot t, i.e., $\mathcal{K}_t = \{k | x_{k,t} > 0\}$.
  if one or more vehicles are scheduled for t, i.e., $\mathcal{K}_t \neq \emptyset$ **then**
    Select vehicle $k \in \mathcal{K}_t$ with maximum available bit rate, i.e., $k = \arg\max_{i \in \mathcal{K}_t} b_{i,t}$.
    Transmit to vehicle k and update its demand, i.e., $d_k = \max(0, d_k - b_{k,t})$.
    Let $\mathcal{V}' = \{v | v \in \mathcal{V}, d_v \neq 0\}$
    if t is not completely assigned to k: i.e., $x_{k,t} < 1$. **then**
      GFG-SCHEDULER($\mathcal{V}', \mathcal{T}', d_i, b_{i,j}$), where $\mathcal{T}' = \mathcal{T} - \{t\}, i \in \mathcal{V}', j \in \mathcal{T}'$
    end if
  end if
end for
function GFG-SCHEDULER($\mathcal{V}, \mathcal{T}, d_i, b_{i,j}$) $\forall i \in \mathcal{V}, j \in \mathcal{T}$
  Construct a new GFG as in Figure 4.3.
  Solve the GFG and create a new schedule $x_{v,t}$ for all $v \in \mathcal{V}, t \in \mathcal{T}$.
  return $x_{v,t}$ for all $v \in \mathcal{V}, t \in \mathcal{T}$
end function

maintain the same achievable bit rate. The frames extend until the last vehicle of those currently inside the range exits. The algorithm steps are similar to the steps shown in Algorithm 3, except for function GFG-Scheduler where a time expanded graph is built and solved instead of a generalized flow network. As with the G-GF, G-TEG algorithm can be executed in polynomial time.

4.6 Performance Comparison Examples

In this section we study the performance of the proposed algorithms by showing some representative examples of our results. The input data that was fed to the schedulers assumed a highway scenario where vehicles maintain constant speed throughout the RSU coverage area (Khabazian and Ali, 2008) (Hammad *et al.*, 2013). Good estimates of energy costs can be readily made in this type of situation (Wang, 2005)(C. Sommer and Dressler, 2011) and the bit rates used for the downlinks are based on these vehicle position estimates. A distance dependent exponential path loss model was assumed, using a path loss exponent of $\alpha = 3$. In many practical systems however, there will be dominant deterministic propagation with random components due to propagation effects such as shadowing. Therefore, we also include results where the scheduler input data includes random errors due to log-normal shadowing components.

Vehicular traffic models have been extensively studied and in general, vehicle behavior is highly variable (Harri *et al.*, 2009) (Martinez *et al.*, 2011). In highway scenarios however, vehicle interaction is considered to be relatively insignificant with vehicles tending to maintain constant speeds for relatively long periods of time (Helbing, 2001). The vehicle arrival process is often modeled to be Poisson (Khabazian and Ali, 2008) (Bilstrup *et al.*, 2008) and this what is assumed in our experiments. We also model the traffic in different lanes as belonging to different classes each with different speeds as in (Wang, 2005). The RSU is assumed to be in the middle of the highway segment of interest with vehicles passing by in one or more directions. The physical layer model is based on that obtained from (Demmel *et al.*, 2012) (Shivaldova *et al.*, 2011) where the bit rates vary according to distance from 3 to 27 Mbps. A 2 Km coverage area is assumed for the RSU. The time slot length for these experiments is set to one second. The energy values are given as multiple of

the energy required to transmit for one unit of time. The demand is expressed as multiples of the number of bits per time slot at the highest bit rate. For example, if the highest bit rate is 27 Mbits/s, one unit of demand is equivalent to 27 Mbits. Traffic flow and scheduling algorithms were implemented in MATLAB. The optimization problems were expressed in AMPL and solved using CPLEX. The theoretical offline minimum energy derived by the models in Section 4.4.2 are referred to as *TEG Bound* in the presented graphs. The values for *TEG Bound* computed either using the GF model or TEG are very close, and therefore only one of them, i.e., TEG, is reported in these experiments as the theoretical bound. This Bound is compared to the online algorithms (FCFS, FF, G-TEG and G-GF) we presented in Section 4.5.

Results are presented for experiments using two classes of vehicles, and the experiments show the effects of changing demand and vehicle speed. Later, the fairness of the proposed scheduling algorithms is studied. In the graphs, each plotted point is the average of 30 runs to ensure the results are representative of the performance of the algorithms. To heavily stress the scheduling algorithms, we often subject the RSU to traffic conditions which the RSU cannot fully satisfy. In these experiments we keep track of the vehicle loss rate, i.e., the probability that a vehicle's demand cannot be satisfied. Note that the online algorithms are designed to serve as much demand as possible, i.e., they will not drop vehicle demand in favor of minimizing total energy use. Demand is only dropped when no time slots are available.

When scheduling is only feasible by dropping demand, the set of vehicles that can resolve the situation is identified. They are sorted in ascending order based on residual demand and the first vehicle is selected. Its demand is reduced in increments of one unit until the situation is resolved or its residual demand is wholly dropped. If the infeasibility

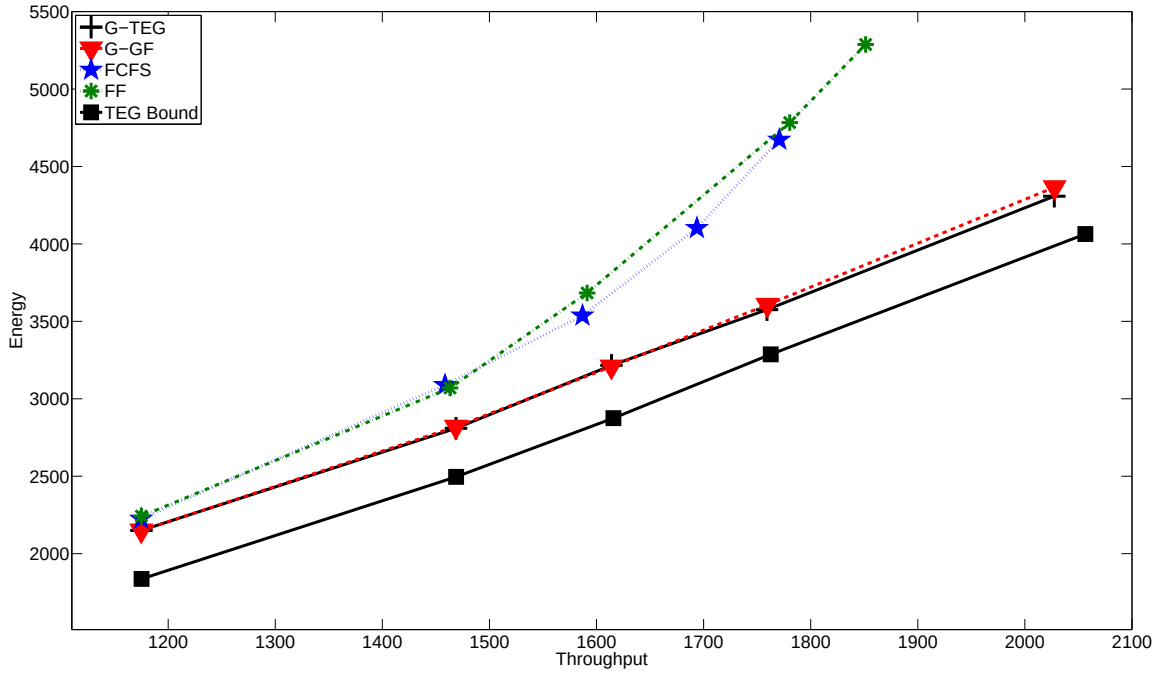

Figure 4.5: Throughput and Energy Comparisons. Low to Medium Demand.

persists, another vehicle from the set is selected and the dropping process is repeated until the scheduling is possible.

In the first experiment, we study the performance of the algorithms under light to medium loading demands. The vehicular traffic consists of two classes, c_1 and c_2 where the arrival rate for both is $\lambda = 1/28$ (vehicle/time slot). The two-vehicle platooning[1] percentage is 0.05 which means that 5% of vehicles arrive together. Vehicle speeds c_1 and c_2 are 18 and 30 m/s (65 and 108 KM/h), respectively.

Figure 4.5 shows the performance of the FCFS, FF, G-GF and G-TEG algorithms versus throughput. It can be seen that the G-GF and G-TEG algorithm performance is very similar and generally follows slightly about that of the Bound. The energy performance is about 25% higher than the bound under lighter throughput values and about 7.5% under the

[1] "Platooning" is where multiple vehicles travel together as a group.

highest values of throughput plotted. Note that the departure from the bound comes from several reasons. The bound is clearly non-causal and benefits from knowledge of future inputs which the online algorithms do not have. Although the G-GF and G-TEG algorithms use myopic versions of the generalized flow and time expanded graphs, the process of rounding the fractional outputs to integer schedules incurs some energy overhead. Finally, at larger values of throughput, the algorithms incur a loss rate, i.e., unfulfilled demand, that decreases their energy use compared to the bound which does not incorporate loss into its formulation. This accounts for the lower fractional spread between the G-GF/G-TEG algorithms compared with the bound at higher values of throughput.

Under light values of throughput, the FCFS and FF algorithms can be seen to require about the same total energy as G-GF and G-TEG. This is clearly because the scheduling problem is very simple when there is almost no contention for downlink time slots. It is not possible to do any better than greedily assigning the best available time slots to incoming vehicles. For this reason, under very light traffic conditions there is effectively no value in going to a much more sophisticated scheduling algorithm. Note that at lower values of throughput, all four curves eventually converge. As the throughput increases, however, this situation changes, i.e., the simplistic design of FCFS and FF begins to adversely affect their performance. For example, in the case considered in Figure 4.5, at the highest values of throughput plotted, the energy required by the FCFS and FF algorithms is over 30% higher than that for the G-GF and G-TEG algorithms. This is a significant improvement in energy performance.

The demand drop percentage for the experiment explained above is shown in Figure 4.6. Here, the load levels are the same as in Figure 4.5. Under light demand, the G-GF and G-TEG algorithms experience a very low dropping rate. This is due to the low probability

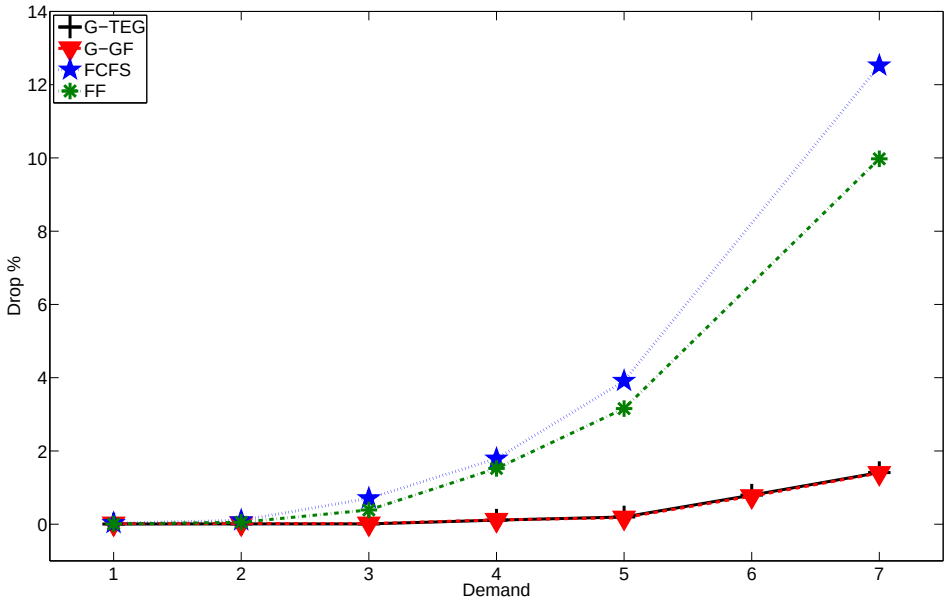

Figure 4.6: Demand Drop Percentage Under Low to Medium Demand

of contention arising from more than one vehicle competing for the same downlink time slots. The G-GF and G-TEG are able to resolve this contention without the need to drop vehicle demand.

Under medium loading, the G-GF and G-TEG dropping rates are in the 1.7% range. This happens for two reasons. First, the Bound has knowledge of all future arrivals so it can anticipate contention and schedule vehicles before and after the contention period to achieve a good schedule. G-GF and G-TEG are causal algorithms and can only identify contention once the vehicles creating it arrive to the RSU. Second, the sharing of a single time slot between more than one vehicle in the Bound model is not allowed in the online algorithms, which renders some schedules created by the Bound as infeasible.

The FF and FCFS algorithms experience zero dropping under very light load as in the first and second demand levels in Figure 4.6. Despite their relatively simple design, FF and

FCFS in this situation encounter very little contention which they are able to resolve without dropping demand, even if the resulting schedule is less than optimal. As the demand increases (points 3 and 4 on the plot) however, FF and FCFS start to experience dropping which is still below 2%. This is in contrast to the more computationally expensive G-GF and G-TEG algorithms which are experiencing little or no dropping. If the system was designed to allow for a small percentage of dropping (e.g., 2% or 3%) in exchange for simple to compute schedules, using FF or FCFS would be acceptable.

Under medium loading (last point on the plot), the dropping obtained by FF and FCFS are about 6 and 7.5 times the drop percentage for G-GF, respectively. Under these conditions, even a sophisticated algorithm like G-GF has no option but to start dropping demand while trading performance for computational simplicity. This indicates that under medium load, using a more efficient algorithm like G-GF and G-TEG becomes more reasonable.

It can be seen that the difference in the dropping percentage between FF and FCFS is less than 2% throughout most of these experiments. Although FF is designed to handle faster moving vehicles better than FCFS, under a low platooning percentage of 5%, the frequency of vehicles arriving together is limited. This allows for the simple FCFS algorithm to accommodate the demand of a newly arrived fast moving vehicle without dropping, even if it is in a less optimal location. This encourages the use of the less computationally expensive FCFS algorithm over the FF algorithm in traffic situations where vehicle flow experiences lower levels of arrival fluctuation.

To further study the performance of the online algorithms, Figure 4.7 shows the throughput and energy performance comparison under higher demand levels than in the previous experiment. Figure 4.8 shows the corresponding dropping behavior. In this experiment, the traffic consists of two classes of vehicles c_1 and c_2, where c_1 speed is 18 m/s and c_2 speed

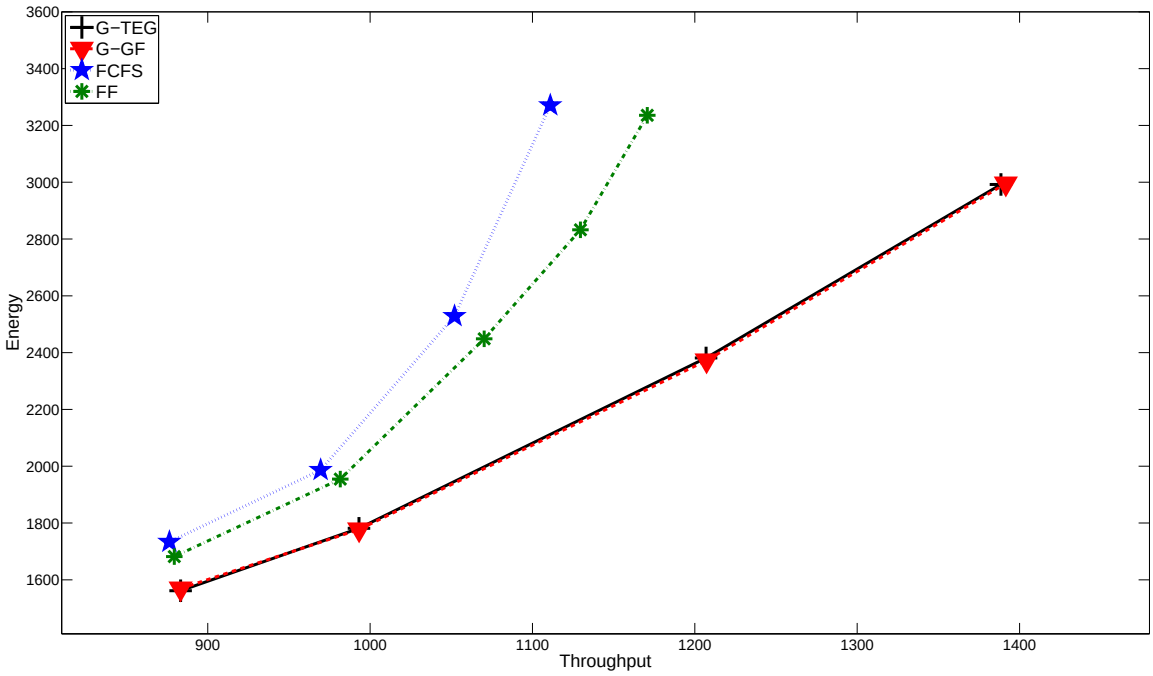

Figure 4.7: Energy Performance under High Demand

is 30 m/s. The vehicle arrival rate for both classes is $1/30$ vehicles/time slot and the two-vehicle platooning percentage is 10%. The increased platooning and demand levels allows for studying the algorithm behavior when experiencing heavy load conditions. The figures show that the performance of the two greedy algorithms (G-GF and G-TEG) is very close despite the two different modeling approaches. As slow moving vehicles or the choice of very small time slots can cause the problem size to grow very large in the G-GF case, it is best in this situation to use the G-TEG algorithm. Under heavy load, G-GF and G-TEG experience about 3% demand dropping. Despite the dropping, throughput continues to grow for these algorithms and they do not experience a saturation phenomenon where the throughput fails to increase while the energy use increases with demand. This is due to the fact that G-GF and G-TEG are able to resolve contention in an efficient way, even while some dropping is needed. On the other hand, the FCFS and FF algorithm throughput

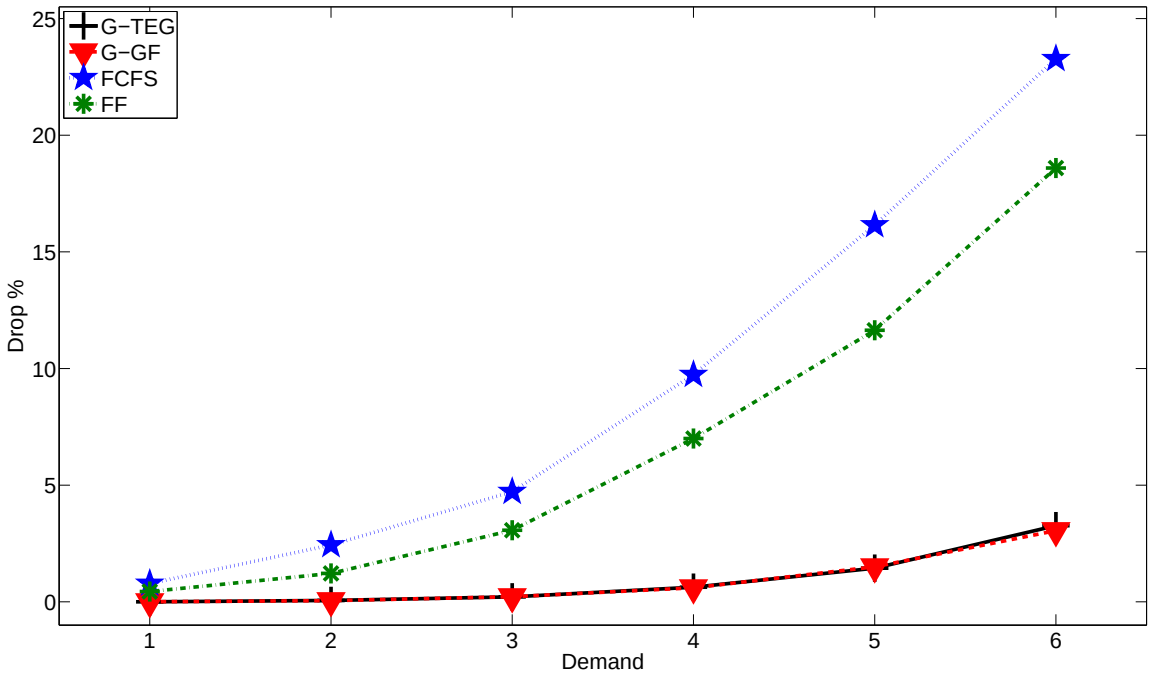

Figure 4.8: Dropping Performance under High Demand

saturates and the curve starts to grow vertically under heavy load, indicating that no more demand can be served. This means that the maximum performance for FF and FCFS is lower than their G-GF and G-TEG counterparts. Identifying the demand threshold would be important when using any of these algorithms. We can also see that in this experiment, there are more significant performance differences between the FF and FCFS algorithms. This is mainly due to the fact that the platooning percentage has increased which creates more frequent contention. In this case, FF offers faster moving vehicles a chance to make use of high bit rate time slots which the statically assigned schedule created by FCFS would deny. This shows that if the traffic to be served by the RSU is characterized by high variations in the arrival of vehicles, it is favorable to use FF over FCFS.

Vehicle speed is strongly related to the amount of energy required and the drop percentage that the vehicles demand experiences. The reasons behind this are that as velocity

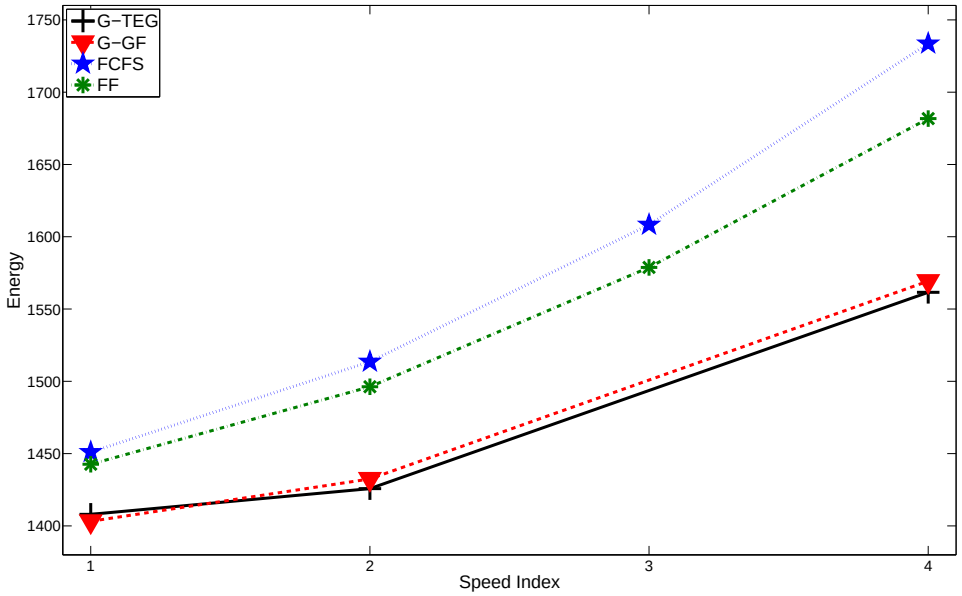

Figure 4.9: Energy Use under Light Load for Different Speed Indices

increases, the number of available time slots decreases for a particular vehicle. Also, in the case of two classes of vehicles traveling at different speeds, faster vehicles are catching up with more vehicles from the slower moving class. This creates more frequent contention periods.

The performance of the algorithms under different speed conditions is shown in Figures 4.9 and 4.10. The parameters for these experiments are the same as in the previous cases except for the vehicle speeds. While class c_1 maintains a velocity of 18 m/s, class c_2 is changed. The speed index from 1 to 4 refers to testing c_2 under velocities: 18, 22, 26 and 30 m/s. In the figures, the first point on the x-axis represents the two classes traveling at the same speed. In Figure 4.9, under light loading, the energy required by FF and FCFS is very close. This is expected as it is easy for either the static (FCFS) or the dynamic (FF) algorithms to find a good schedule. But in Figure 4.10, the difference between them

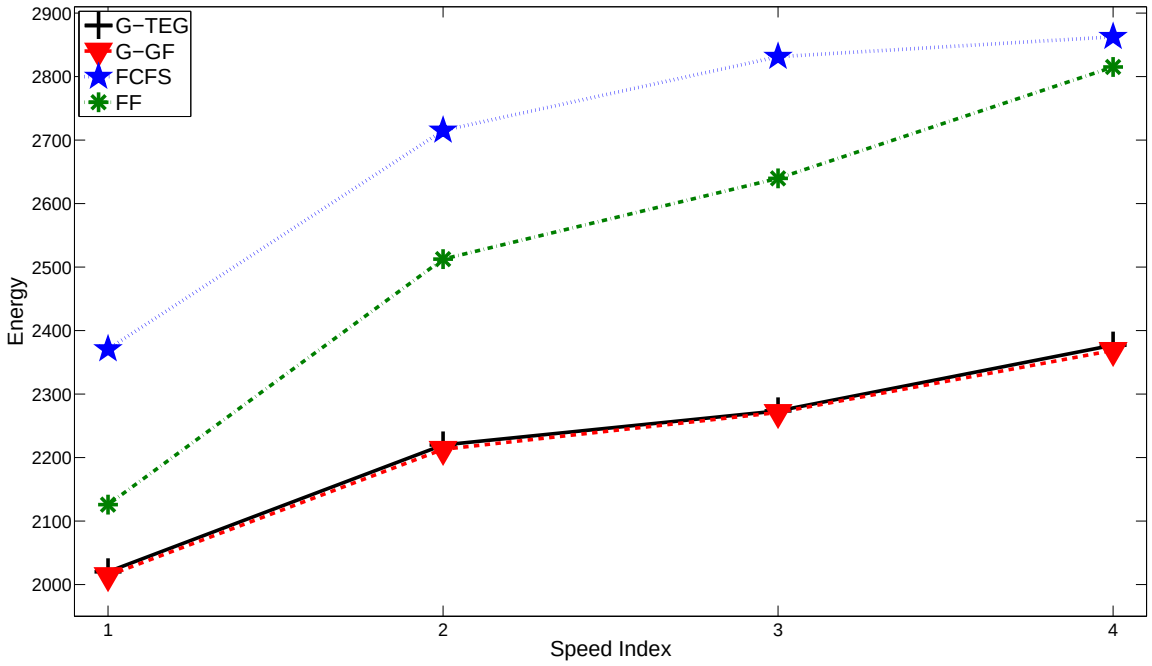

Figure 4.10: Energy Use under Medium Load for Different Speed Indices

is about 13%. Although FF does not differentiate and prioritize vehicles based on their speed, as both classes have the same speed at this point, the dynamic nature of FF allows the recalculation of the schedule every time a new vehicle arrives. With no prioritization, a newly arrived vehicle can be given the chance to communicate with the RSU during the time periods when it is enjoying a high bit rate. On the other hand, FCFS is a statically assigned schedule. Previously arrived vehicles keep all previously assigned time slots. In situations where the demand is high even if the traffic flow is steady, using the more computationally expensive FF is favourable. The above reasoning also explains why the gap between FF and FCFS under light load is smaller than the gap under heavier loading.

Table 4.1 shows results for the performance of the algorithms under different traffic speed mixtures. The first obvious observation is that under low or medium demand levels, and for all online algorithms, as the speed index increases, the required energy increases.

The reason for this is that the algorithms are forced to communicate with faster moving vehicles at further distances and lower bit rates, causing them to use more time slots. However, this does not necessarily translate into a change in the throughput. First we examine the low demand behavior. Both G-GF and G-TEG maintain constant throughput as no dropping occurs and they are able to successfully schedule all requests even as the speed for class c_2 is increasing. In FCFS and FF, the throughput decreases with the speed index increase. FF throughput decreases by about 1% while FCFS throughput decreases by about 2.3%. Under medium load the situation becomes clearer. While G-GF and G-TEG throughput drops by less than 1%, with energy increases of only 17% and 17.6%, respectively, FCFS and FF throughput drops by 7.5% and 6.9% with energy increases of 20% and 32%, respectively. These results show that G-GF and G-TEG are far more suitable for medium to high demand situations.

In the next set of results we show comparisons where the input data to the schedulers is not ideal. This was done by including random log-normal shadowing in the propagation model estimates. This is done to ensure that when the scheduler input data are not ideal, the algorithms do not produce results which would be biased in some way, due to this randomness. In the results presented, this was done by adding propagation shadowing effects to the extracted data, which result in unpredictable randomness in these estimates. The scheduling is therefore based on this input, but the actual costs incurred include the energy perturbations due to the random components. The random shadowing components are unknown to the online schedulers which base their decisions on deterministic positional information only, while the theoretical bound can make use of the shadowing information to produce a proper bound on the energy required to serve vehicle demand.

Figure 4.11 shows the performance differences between the online algorithms and the

			Speed Index			
	Demand		1	2	3	4
FCFS	Low	Throughput	993	983	981	970
		Energy	1549	1816	1883	1987
	Med	Throughput	1186	1147	1101	1096
		Energy	2370	2715	2831	2862
FF	Low	Throughput	993	992	989	981
		Energy	1534	1796	1866	1954
	Med	Throughput	1213	1198	1174	1129
		Energy	2125	2512	2639	2815
G-GF	Low	Throughput	993	993	993	993
		Energy	1513	1698	1731	1777
	Med	Throughput	1214	1213	1211	1207
		Energy	2014	2212	2271	2368
G-TEG	Low	Throughput	993	993	993	993
		Energy	1531	1705	1738	1781
	Med	Throughput	1214	1213	1211	1206
		Energy	2020	2219	2273	2377

Table 4.1: Throughput and energy requirement under different demand and velocity conditions for the online algorithms

Bound when these effects are introduced with shadowing standard deviation of 4 dB. In this experiment, the two-vehicle platooning percentage is 10%, two classes of traffic are simulated, and the arrival rate for the two classes is both $1/30$ vehicles/time slot. Vehicle speeds for c_1 and c_2 are 18 and 30 m/s, respectively.

The effect of including the shadowing shows in the increased difference between the Bound and the online algorithms when compared to the case in Figure 4.5. The reason behind this is that as the theoretical Bound takes into account the instantaneous shadowing effect, whereas the online algorithms are only aware of it at packet transmission time and not when the scheduling occurs. When the actual bit rate at the execution time is different than the one the online algorithm based its decision upon, rescheduling is triggered and

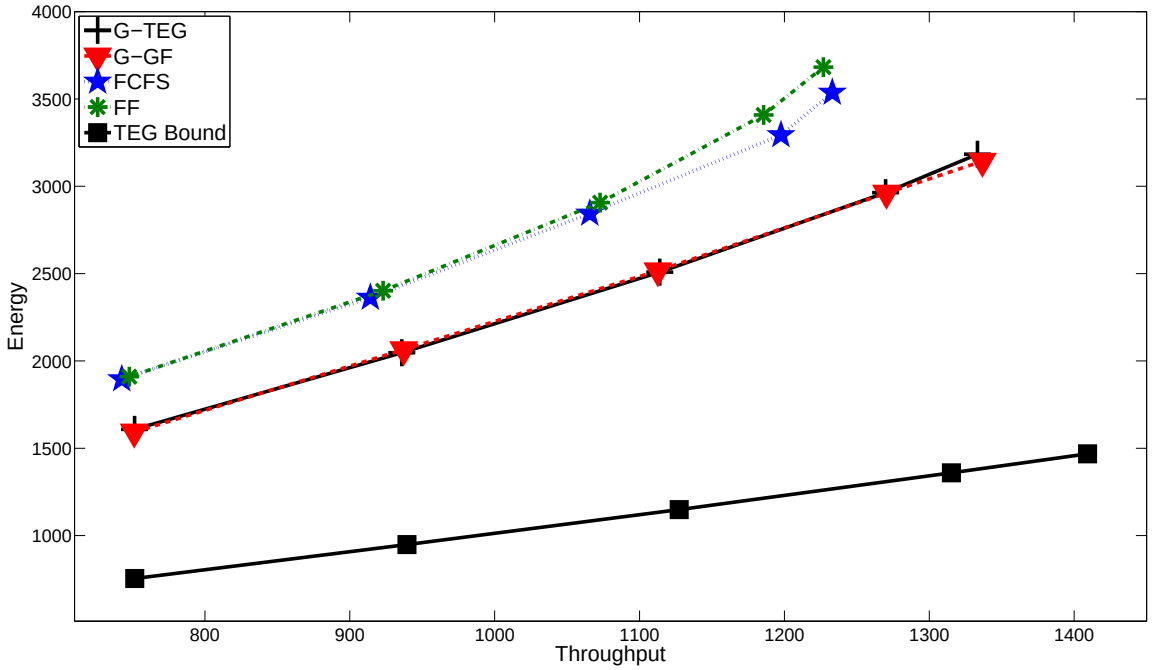

Figure 4.11: Throughput Performance with Log-Normal Shadowing

extra computations need to be done in order to accommodate the change in achievable bit rate. Figure 4.12 shows the dropping associated with this experiment. The main effect of including the log-normal shadowing is that the drop percentage is higher compared to the previous cases. For example, G-GF and G-TEG starts dropping at medium demand levels, which did not happen before. Faced with instantaneous bit rates different than the ones used to create the schedule, G-GF and G-TEG are forced to reschedule the rest of the time slots remaining for the vehicles which can result in a poorer schedule and increased dropping. However, these algorithms have maintained their superior performance over the simpler ones, i.e., FCFS and FF.

Figure 4.13 shows the effect of including the shadowing on the differences between the Bound and the two online algorithms, G-GF and G-TEG, where the throughput for both cases (with and without shadowing) is shown. The experiment is for low to medium

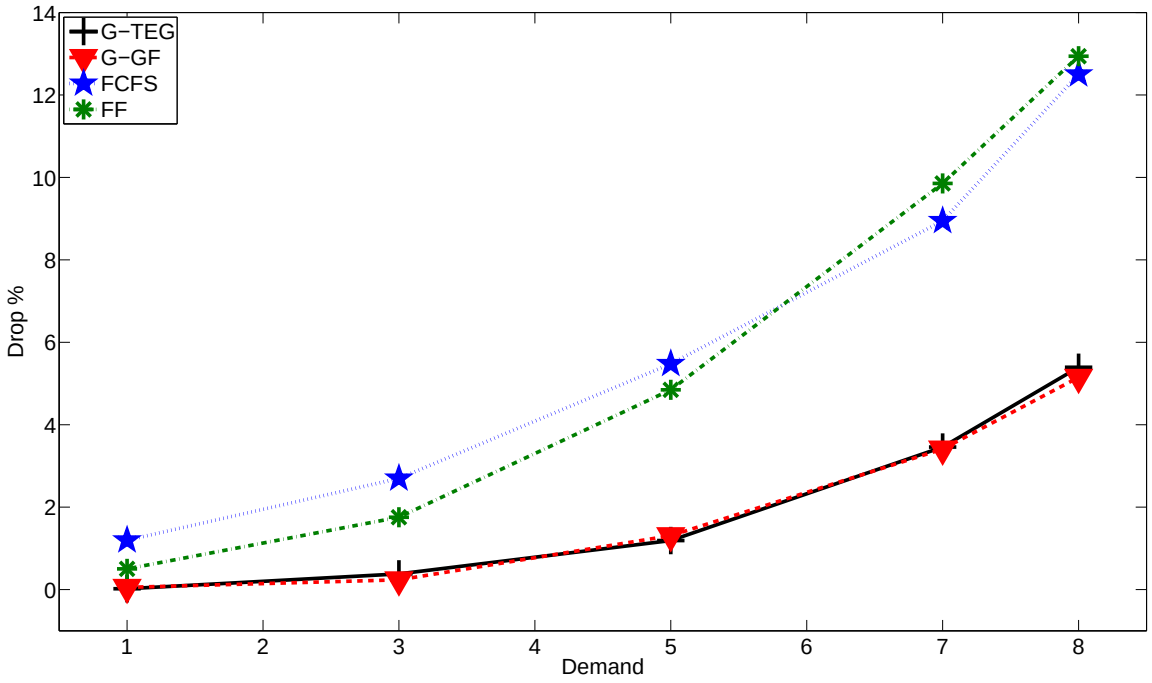

Figure 4.12: Demand Drop Percentage with Log-Normal Shadowing

demand to ensure the feasibility of the Bound calculations. The lines which represent the results of including the shadowing in both the bound and the two online algorithms starts with "sh". When including the log-normal shadowing, the differences between G-GF and G-TEG on the one side and the Bound on the other becomes much bigger than in the deterministic case. We also notice that the Bound for the shadowing case is *lower* than the Bound for the deterministic case. This is attributed to the fact that, unlike the online cases, the offline algorithms are aware of the random channel fluctuations and can exploit them in their scheduling.

An issue that arises when vehicular demands are not fully fulfilled is one of fairness. For example, an algorithm could choose to drop demands from faster moving vehicles whose fulfillment would result in disproportionately increased energy requirements. Clearly there is a tradeoff between fairness and total energy use. In order to assess whether vehicles

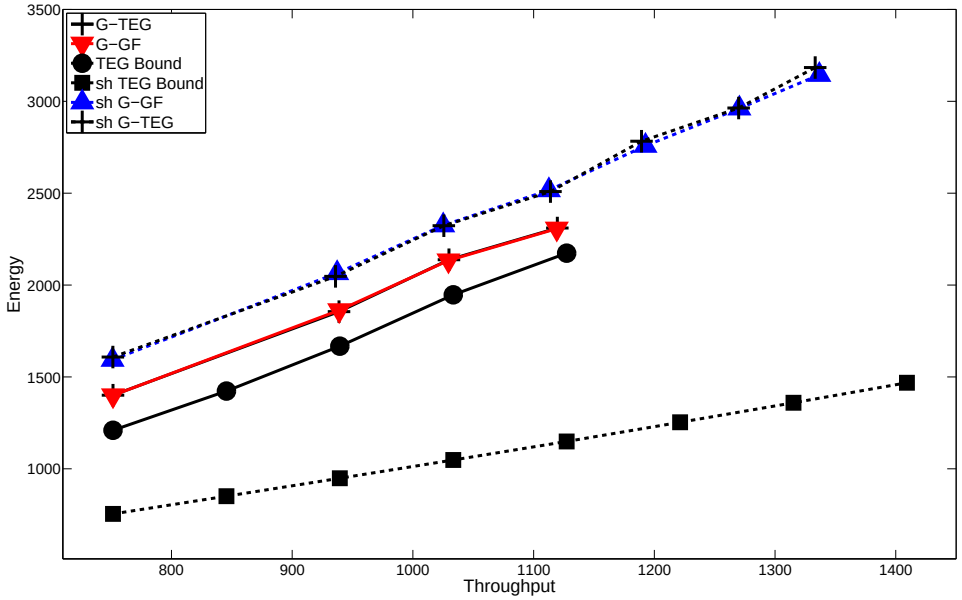

Figure 4.13: G-GF and G-TEG Performance with and without Shadowing

of different classes are treated fairly, *Jain's Fairness Index* is used to measure the fairness across the classes. This is defined as

$$f(x_1, x_2, x_3, \ldots, x_n) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} x_i^2} \tag{4.12}$$

where the index value is in the range $0 \leq f() \leq 1$. x_i represents the percentage loss each traffic class encounters individually. If all traffic classes are treated equally, Jain's Fairness Index will be 1. The worst case unfairness when there are two classes of traffic, is 0.5. We will show some representative examples from our experiments.

Table 4.2 shows that at high speed index, G-GF and G-TEG achieve almost perfect fairness across the vehicles. They are followed by the FF algorithm, while the FCFS suffers the most. The algorithms G-GF and G-TEG treat all vehicles equally when it comes to

Method	Average Load	High Load
G-TEG	0.98	0.99
G-GF	0.98	0.99
FF	0.83	0.91
FCFS	0.53	0.62

Table 4.2: Jain's Fairness for Different Algorithms

dropping demand. For example, in G-GF, the cost over the edges between vehicles and bit rate nodes depends on which bit rate is higher and does not depend on vehicle velocity. Thus, when the dropping is needed, no distinction between vehicles is made, which translates into good levels of fairness.

The fairness of FF compared to FCFS is shown in Tables 4.3 and 4.4. As mentioned earlier, the speed index number increase refers to the difference in velocity between classes c_1 and c_2. Table 4.3 shows that FF fairness actually increases as the difference in vehicle speed increases. This is because as the speed difference increases, the priority given to faster moving vehicles becomes higher. Therefore, despite the much shorter time they spend inside the RSU coverage range, they obtain treatment comparable to their slower moving counterparts. Table 4.4 shows that FCFS suffers as the difference between vehicle speeds increases. This is due the static nature of the FCFS algorithm where it treats all vehicles as jobs in a queue, despite the fact that faster vehicles spend less time inside the RSU coverage range, which increases their probability of dropping.

4.7 Conclusions

In this chapter we have considered the energy efficient roadside unit (RSU) scheduling problem when the RSU-to-vehicle radios use a variable bit rate (VBR) air interface. Offline

Speed Index	Low Load	Average Load	High Load
2	0.54	0.61	0.66
3	0.59	0.69	0.78
4	0.72	0.85	0.92

Table 4.3: FF Fairness for Different Speed Indices

Speed Index	Low Load	Average Load	High Load
2	0.74	0.82	0.87
3	0.59	0.69	0.73
4	0.52	0.58	0.62

Table 4.4: FCFS Fairness for Different Speed Indices

scheduling formulations were given which provide lower bounds on the energy needed to satisfy arriving vehicular demand. An integer linear program (ILP) was introduced which can be used to find optimal offline variable bit rate time slot schedules. It was shown that this problem is NP-complete by a reduction from the well-known Santa Claus Problem.

Two flow graph based models were introduced to solve the minimum energy VBR scheduling problem. The first uses Generalized Flow (GF) graphs which represent time slots as individual graph nodes. The second uses time expanded graphs (TEGs) which model the system as a time expanded flow graph. Both of these models can be efficiently solved and provide lower bounds on energy performance. They also provide the basis for some proposed online schedulers. Four energy efficient online scheduling algorithms were considered. The first, First Come First Serve (FCFS) is very simple and treats all vehicles equally and in order of arrival. The second, Fastest First (FF), gives priority to faster moving vehicles since their transit time through the RSU coverage area is less than slower moving vehicles. The Greedy Generalized Flow (G-GF) and Greedy Time Expanded Graph

(G-TEG) algorithms are two online implementations of the two flow graph models. Results from a variety of experiments showed that the proposed algorithms perform well compared to the energy lower bound. Our results show that less computational intensive algorithms, i.e., FCFS and FF, can perform well under light load, but G-GF and G-TEG, while more computational intensive, can provide near optimal throughput and minimum drop percent in demand. The results also show that the G-GF and G-TEG algorithms are much more fair than the simpler algorithms in heavy load situations.

Chapter 5

Conclusions and Future Work

Vehicular Ad-hoc Networks (VANET) will have a prominent role in the future of Intelligent Transportation Systems. Connected vehicle applications ranging from safety, efficient traffic management and infotainment will drive the demand for reliable and efficient networking solutions. Roadside units (RSUs) play an important role in VANETs, either storing and relaying information, or as gateways to the Internet.

In highway environments where wired electricity is prohibitively expensive to provide, RSUs will operate using renewable energy sources. Hence, an energy efficient RSU scheduling algorithms can play an important role in managing the cost of the infrastructure needed support and accelerate VANET adoption.

This thesis investigated RSU to vehicle scheduling from an energy efficiency point of view. In certain situations, the location of the vehicle can be predicted with high degree of confidence. This information can then be used to minimize the energy needed for RSU to vehicle communication. The air interface types studied were constant bit rate (CBR) and variable bit rate (VBR).

In the first part of the thesis, the CBR case was examined. In order to provide a theoretical lower bound on the energy needed to serve incoming vehicular communication requirements, offline scheduling models were created. The scheduling can be packet-based or timeslot-based. It was shown how the packet-based scheduling can be formulated as a generalization of the classical single-machine job scheduling with earliness and tardiness penalties, refereed to as α-Earliness-Tardiness. The problem was proven to be NP-complete even under simple propagation assumptions. For timeslot-based scheduling, a Mixed Integer Linear Program (MILP) model was developed. A novel model using minimum cost flow graphs was proposed that solve the scheduling problem in polynomial time.

Three energy efficient online scheduling algorithms were introduced. The first, Greedy Minimum Cost Flow (GMCF) was inspired by the minimum cost flow formulation used for the offline energy bound. The other two online algorithms were less computationally intensive alternatives for the GMCF. Nearest Fastest Set (NFS) made use of vehicle velocity and residual demand to dynamically schedule arriving vehicles and resolve contention. The Static Scheduler (SS) assigns vehicle communication slots statically and provides better performance than NFS by resolving contention before they happen but at a higher computational cost for the RSU. A wide variety of experiments showed the good performance of the proposed algorithms compared to the theoretical bound in situations where a dominant component of the path loss is deterministic, leading to accurate estimation of the energy cost. Near optimal results are possible but the algorithms incurred high computational cost required by the RSU.

In the second part of the thesis, the VBR interface was assumed for the RSU and energy efficient scheduling was studied. Offline scheduling models were introduced to study the minimum theoretical energy required by the RSU to serve vehicle requirements. The

scheduling problem was modeled as Integer Linear Program (ILP). It was shown that this problem is NP-compelete by a reduction from the Santa Claus Problem. Two models were introduced to solve the offline scheduling problem based on flow graphs. The Generalized Flow (GF) graph provided the basis for the first model which used time slots. Time Expanded Graphs (TEG) were used as a second model which employed the concept of time frames. These two models provided the basis for two out of the four online line algorithms proposed to solve the variable bit rate scheduling problem. These two algorithms were Greedy-Gernalized Flow (G-GF) and Greedy-Time Expanded Graphs (G-TEG), both of which provide near optimal performance. In addition, First Come First Serve (FCFS) is a simple scheduling algorithm that treats all vehicles equally and was shown to be suitable for low traffic and low demand situations. Fastest First (FF) was inspired by the fact that faster vehicles spend less time inside the RSU coverage range than slower moving vehicles. In order to maintain fairness across different vehicles, their velocity needs to be taken into account. Results from a variety of experiments showed the good performance of the proposed online algorithms compared to the theoretical offline bounds. Our results showed that less computational intensive algorithms, i.e., FCFS and FF, can perform well under light load, but G-GF and G-TEG, while requiring more time to compute, can provide near optimal throughput and minimum demand dropping. The results also show that the G-GF and G-TEG algorithms are much more fair than the simpler algorithms in heavy load situations.

The work in this thesis can be extended in the future by considering differentiated services, where urgent safety messages have higher scheduling priority than traffic information and entertainment data packets. This can be implemented by having earlier deadlines associated with high priority messages. Moreover, schedulers can be modified to assign

weights to vehicle requests, resulting in lower wait times for urgent messages. Scheduling algorithms like First Deadline First (FDF) and Longest Wait Time (LWT) surveyed in the background chapter can be applied to proposed algorithms to offer different classes of services. Moreover, admission control algorithms can be employed to increase the predictability and reliability of the system. The RSU can notify arriving vehicles with the probability of a request being granted. This can be calculated from historical traffic flow and achieved throughput values. As wind and solar power are intermittent energy sources, the energy available to the RSU can vary over time. The proposed scheduling algorithms behavior can adjust to accommodate these energy cycles, thus ensuring delivery of high priority messages.

Future extensions to the algorithms proposed in this thesis can take into account the energy required to execute the algorithms and to download request contents from backhaul networks, e.g., cellular network. Another extension is to create a hybrid system of more than one of the proposed algorithms, as their computational requirements and performance differ. Depending on the traffic conditions, residual battery energy and availability of sunshine (in the case of solar powered system, for example), the RSU can choose between different scheduling algorithms to use in different situations. Moreover, the broadcast nature of the communication medium invites the idea of assigning data items that are requested by more than one vehicle higher scheduling priority. This will maximize system capacity. Employing network coding techniques would be another extension to this work and can further improve the system performance.

Appendix A

NP-Completeness of α-Earliness-Tardiness

In this section we show the NP-completeness of the α-Earliness-Tardiness Problem formulated in Section 3.4.1. This is done by a reduction from the well-known PARTITION PROBLEM (Garey and Johnson, 1990). In the decision version of PARTITION we are given a set of n integers and must answer the question: "Can we divide the given set into two subsets such that the sum of the numbers in each set are equal?" Our proof is by reduction of PARTITION to the following version of α-EARLINESS-TARDINESS where $\alpha \geq 1$.

INPUT: n jobs, each job $i = 1, \ldots, n$ with its own processing time p_i, weight w_i, shift amount r_i, and due date D_i.

OUTPUT: A single-machine (non-preemptive) schedule of the jobs that minimizes

$$\sum_{i=1}^{n} w_i(|C_i - D_i| + r_i)^{\alpha}$$

where C_i is the completion time of job i in the schedule. Note that we assume that all our

input data are integers and that jobs can only start at integral time points.

Theorem 2. α-EARLINESS-TARDINESS *is NP-complete for any real* $\alpha \geq 1$.

Proof. Suppose we are given an instance of PARTITION with n objects, each with a value $p_i,\ i = 1, \ldots, n$. Let $P := \sum_{i=1}^{n} p_i$. We define an instance of α-EARLINESS-TARDINESS as follows:

- We have $n + 3$ jobs: Jobs $1, 2, \ldots, n$ correspond to the PARTITION objects; each has processing time p_i, weight 1, and due date D_1. Job $n + 1$ has processing time $p_{n+1} = 1$, weight w and due date D_1 (as jobs $1, \ldots, n$). Job $n + 2$ has processing time $p_{n+2} = l$ and weight w, defined as follows:

$$l := n^{\frac{1}{\alpha}}(1 + P), \quad w := n(2P + l + 1)^{\alpha} + 1$$

  and due date D_2. Job $n + 3$ has processing time $p_{n+3} = l$ and weight w (just like job $n + 1$), but due date D_3. Parameters D_1, D_2, D_3 are defined next.

- All the jobs will have one of the following three due dates:

$$D_2 := P + l, \quad D_1 := D_2 + \frac{P}{2} + 1, \quad D_3 := D_1 + \frac{P}{2} + l.$$

- Every job j has $r_j = \frac{p_j}{2}$.

First we show that any optimal schedule for this instance must schedule jobs $n + 1, n + 2, n+3$ so that they finish exactly on their due date. Indeed, suppose that there is an optimal schedule $\mathcal{S}$ such that at least one of these three jobs finishes earlier or later than its due date by at least 1 time unit. Then let $\mathcal{S}'$ be a schedule where these three jobs are neither early

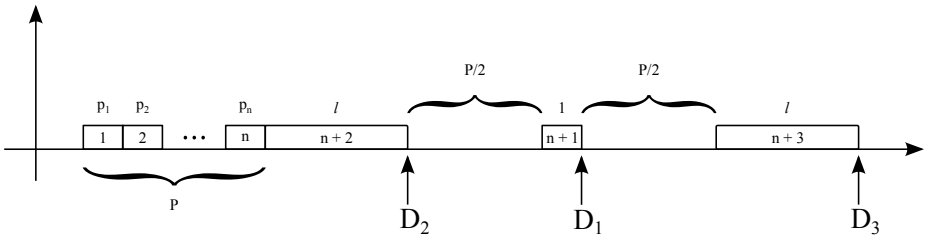

Figure A.1: Schedule $\mathcal{S}'$.

nor late, and the rest n jobs are scheduled right before job $n+2$ in any order (say in order $1, 2, 3, \ldots, n$). So $\mathcal{S}'$ is as shown in Figure A.1. Going from schedule $\mathcal{S}$ to schedule $\mathcal{S}'$, the following will happen.

- Due to jobs $n+1, n+2, n+3$, the cost decreases by at least w: Assume that job $n+1$ doesn't finish on its due date (the other cases are similar, or even better for our argument in case more than one of jobs $n+1, n+2, n+3$ don't finish on their due date). Then the cost decrease in $\mathcal{S}'$ due to the special jobs is at least

$$w(1+\frac{p_{n+1}}{2})^{\alpha} - w(\frac{p_{n+1}}{2})^{\alpha} \geq w.$$

- Due to jobs $1, 2, \ldots, n$, the cost increases by at most $\sum_{j=1}^{n}(3P/2 + l + 1 + \frac{p_j}{2})^{\alpha} \leq n \cdot (2P + l + 1)^{\alpha}$, since we are going from earliness/tardiness penalty of at least 0 in $\mathcal{S}$ (actually $\frac{p_j}{2}$ but 0 suffices for our purposes) to at most $(2P + l + 1)^{\alpha}$ in $\mathcal{S}'$ for each such job.

Therefore

$$cost(\mathcal{S}') - cost(\mathcal{S}) \leq n \cdot (2P + l + 1)^{\alpha} - w < 0,$$

which contradicts the optimality of $\mathcal{S}$. Hence, in any optimal schedule jobs $n+1, n+2, n+3$ finish exactly at times D_1, D_2, D_3 respectively. Next, we look at the optimal cost difference

between the cases of the existence of a partition and the non-existence of such a partition.

- If a partition exists, then there is a schedule that doesn't schedule any of jobs $1, 2, \ldots, n$ before or after jobs $n+2$ and $n+3$. In any such schedule, each of jobs $1, 2, \ldots, n$, say job j, incurs a cost of at most $(1 + P/2 + \frac{p_j}{2})^\alpha \leq (1+P)^\alpha$, for a total cost of the optimal $c_{YES} < n(1+P)^\alpha$ (note that $n \geq 2$, so not all processing times can be P).

- If a partition doesn't exist, at least one of jobs $1, 2, \ldots, n$, say job j, must be scheduled before job $n+2$ or after job $n+3$. Therefore, the cost of the optimal schedule will have a cost $c_{NO} \geq (l + P/2 + \frac{p_j}{2})^\alpha \geq l^\alpha = n(1+P)^\alpha$.

By our choice of l, we have $c_{YES} < n(1+P)^\alpha \leq c_{NO}$, and the decision problem "Is there a partition?" has the same answer as "Is there a schedule with cost smaller than $n(l+P)^\alpha$?"

□

Bibliography

Ahuja, R., Magnanti, T., and Orlin, J. (1994). Network Flows: Theory, Algorithms, and Applications. *Journal of the Operational Research Society*, **45**(11), 1340–1340.

Alcaraz, J., Vales-Alonso, J., García-Haro, J., *et al.* (2009). Control-Based Scheduling With QoS Support for Vehicle to Infrastructure Communications. *IEEE Communications*, **16**(6), 32–39.

Alsabaan, M., Alasmary, W., Albasir, A., and Naik, K. (2012). Vehicular Networks for a Greener Environment: A Survey. *IEEE Communications Surveys Tutorials*, **15**(3), 1372–1388.

Azimifar, M. (2013). *Vehicle-to-Vehicle Forwarding in Green Vehicular Infrastructure*. Master's thesis, McMaster University.

Badawy, G. H., Sayegh, A. A., and Todd, T. D. (2010). Energy Provisioning in Solar-Powered Wireless Mesh Networks. *IEEE Transactions on Vehicular Technology*, **59**(8), 3859–3871.

Bansal, N. and Sviridenko, M. (2006). The Santa Claus Problem. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 31–40.

Bilstrup, K., Uhlemann, E., Strom, E., and Bilstrup, U. (2008). Evaluation of the IEEE 802.11p MAC Method for Vehicle-to-Vehicle Communication. In *IEEE 68th Vehicular Technology Conference*, pages 1–5.

Blum, J. J., Eskandarian, A., and Hoffman, L. J. (2004). Challenges of Intervehicle Ad Hoc Networks. *IEEE Transactions on Intelligent Transportation Systems*, **5**(4), 347–351.

Bychkovsky, V., Hull, B., Miu, A., Balakrishnan, H., and Madden, S. (2006). A Measurement Study of Vehicular Internet Access Using In Situ Wi-Fi Networks. In *Proceedings of the 12th ACM Annual International Conference on Mobile Computing and Networking*, pages 1–5.

C. Sommer, D. Eckhoff, R. G. and Dressler, F. (2011). A Computationally Inexpensive Empirical Model of IEEE 802.11p Radio Shadowing in Urban Environments. In *Eighth International Conference on Wireless On-Demand Network Systems and Services*, pages 84–90.

Chen, F., Johnson, M. P., Alayev, Y., Bar-Noy, A., and La Porta, T. F. (2012). Who, When, Where: Timeslot Assignment to Mobile Clients. *IEEE Transactions on Mobile Computing*, **11**(1), 73–85.

Cheng, H. T., Shan, H., and Zhuang, W. (2011). Infotainment and Road Safety Service Support in Vehicular Networking: From a Communication Perspective. *Mechanical Systems and Signal Processing*, **25**(6), 2020–2038.

Dahiya, A. and Chauhan, R. (2010). A Comparative Study of MANET and VANET Environment. *Journal of computing*, **2**(7), 87–92.

Demmel, S., Lambert, A., Gruyer, D., Rakotonirainy, A., and Monacelli, E. (2012). Empirical IEEE 802.11p Performance Evaluation on Test Tracks. In *IEEE Intelligent Vehicles Symposium (IV)*, pages 837–842.

Farbod, A. and Todd, T. (2006). Resource Allocation and Outage Control for Solar-Powered WLAN Mesh Networks. *IEEE Transactions on Mobile Computing*, **6**(8), 960–970.

Festag, A., Baldessari, R., and Wang, H. (2007). On Power-aware Greedy Forwarding in Highway Scenarios. In *5th International Workshop on Intelligent Transportation (WIT), Hamburg, Germany*, pages 31–36.

Festag, A., Hessler, A., Baldessari, R., Le, L., Zhang, W., and Westhoff, D. (2008). Vehicle-to-Vehicle and Road-Side Sensor Communication for Enhanced Road Safety. In *Proceedings of the 15th World Congress on Intelligent Transport Systems*.

Garey, M. and Johnson, D. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Gass, R., Scott, J., and Diot, C. (2005). Measurements of In-Motion 802.11 Networking. In *Proceedings of the 7th IEEE Workshop on Mobile Computing Systems and Applications.*, pages 69–74.

Gozalvez, J., Sepulcre, M., and Bauza, R. (2010). Impact of the Radio Channel Modelling on the Performance of VANET Communication Protocols. *Telecommunication Systems*, **50**(3), 149–167.

Graham, R., Lawler, E., Lenstra, J., and Kan, A. (1979). Optimization and Approximation

in Deterministic Sequencing and Scheduling: A Survey. *Annals of discrete mathematics*, **5**, 287–326.

Hadaller, D., Keshav, S., Brecht, T., and Agarwal, S. (2007). Vehicular Opportunistic Communication Under The Microscope. In *Proceedings of the 5th ACM international conference on Mobile systems, applications and services*, pages 206–219.

Hammad, A. A., Badawy, G. H., Todd, T. D., Sayegh, A. A., and Zhao, D. (2010). Traffic Scheduling For Energy Sustainable Vehicular Infrastructure. *IEEE Global Communications Conference (GLOBECOM'10), Miami, Fla.*, pages 1–6.

Hammad, A. A., Todd, T. D., Karakostas, G., and Zhao, D. (2013). Downlink traffic scheduling in green vehicular roadside infrastructure. *IEEE Transactions on Vehicular Technology*, **62**(3), 1289–1302.

Harri, J., Filali, F., and Bonnet, C. (2009). Mobility Models for Vehicular Ad Hoc Networks: A Survey and Taxonomy. *IEEE Communications Surveys & Tutorials*, **11**(4), 19–41.

Helbing, D. (2001). Traffic and Related Self-driven Many-particle Systems. *Rev. Mod. Phys.*, **73**, 1067–1141.

Jhang, M. F. and Liao, W. (2008). On Cooperative and Opportunistic Channel Access for Vehicle to Roadside (V2R) Communications. *IEEE Global Telecommunications Conference (GLOBECOM'08)*, pages 1–5.

Karagiannis, G., Altintas, O., Ekici, E., Heijenk, G., Jarupan, B., Lin, K., and Weil, T. (2011). Vehicular Networking: A Survey and Tutorial on Requirements, Architectures,

Challenges, Standards and Solutions. *IEEE Communications Surveys & Tutorials*, **13**(4), 584–616.

Khabazian, M. and Ali, M. (2008). A Performance Modeling of Connectivity in Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, **57**(4), 2440–2450.

Khaled, Y., Tsukada, M., Santa Lozano, J., and Ernst, T. (2009). On The Design of Efficient Vehicular Applications. *Proceedings of IEEE 69th Vehicular Technology Conference (VTC'2009).*

Kotnyek, B. (2003). An Annotated Overview of Dynamic Network Flows. *INRIA, Technical Report No 4936*.

Kumar, V. and Chand, N. (2010). Data Scheduling in VANETs: a Review. *International Journal of Computer Science and Communication*, **1**(2), 399–403.

Li, F. and Wang, Y. (2007). Routing in Vehicular Ad Hoc Networks: A Survey. *IEEE Vehicular Technology Magazine*, **2**(2), 12–22.

Liang, H. and Zhuang, W. (2011). Optimal Scheduling for Roadside WLANs with Pre-Downloaded Messages. In *IEEE International Conference on Communications (ICC'11)*.

Liu, K., Lee, V. C., Joseph, K., and Son, S. H. (2013). Scheduling Temporal Data for Real-time Requests in Roadside-to-Vehicle Communication. In *IEEE 19th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 297–305.

Lochert, C., Scheuermann, B., Wewetzer, C., Luebke, A., and Mauve, M. (2008). Data Aggregation and Roadside Unit Placement for a Vanet Traffic Information System. In

Proceedings of the 5th ACM international workshop on VehiculAr Inter-NETworking, pages 58–65.

Martinez, F., Toh, C., Cano, J., Calafate, C., and Manzoni, P. (2011). A Survey and Comparative Study of Simulators for Vehicular Ad Hoc Networks (VANETs). *Wireless Communications and Mobile Computing*, **11**(7), 813–828.

Mershad, K. and Artail, H. (2012). SCORE: Data Scheduling at Roadside Units in Vehicle Ad Hoc Networks. In *IEEE 19th International Conference on Telecommunications (ICT'2012)*, pages 1–6.

Mershad, K., Artail, H., and Safa, H. (2011). Routing Packets to Distant Locations in VANETs. In *IEEE 11th International Conference on ITS Telecommunications (ITST' 2011)*, pages 33–38.

Mittag, J., Schmidt-Eisenlohr, F., Killat, M., Harri, J., and Hartenstein, H. (2008). Analysis and Design of Effective and Low-Overhead Transmission Power Control for VANETs. In *Proceedings of the 5th ACM International Workshop on Vehicular Inter-Networking*, pages 39–48.

Müller, M. (2009). WLAN 802.11p Measurements for Vehicle to Vehicle (V2V) DSRC. *Rohde & Schwarz*, pages 7–8.

Nandan, A., Das, S., Pau, G., Gerla, M., and Sanadidi, M. (2005). Co-operative Downloading in Vehicular Ad-Hoc Wireless Networks. *Second Annual Conference on Wireless On-demand Network Systems and Services (WONS'05)*, pages 32–41.

Ott, J. and Kutscher, D. (2004). Drive-Thru Internet: IEEE 802.11b for Automobile Users.

In *Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 1, pages –373.

Rappaport, T. (1996). *Wireless Communications : Principles and Practice*. Upper Saddle River, N.J. : Prentice Hall PTR.

Rawat, D., Yan, G., Popescu, D., Weigle, M., and Olariu, S. (2009). Dynamic Adaptation of Joint Transmission Power and Contention Window in VANET. *IEEE 70th Vehicular Technology Conference (VTC'2009)*, pages 1–5.

Shahverdy, M., Fathy, M., and Yousefi, S. (2010). Scheduling Algorithm for Vehicle to Road-Side Data Distribution. In *High Performance Networking, Computing, Communication Systems, and Mathematical Foundations*, pages 22–30. Springer.

Shivaldova, V., Maier, G., Smely, D., Czink, N., Alonso, A., Winkelbauer, A., Paier, A., and Mecklenbrauker, C. (2011). Performance Evaluation of IEEE 802.11p Infrastructure-to-Vehicle Tunnel Measurements. In *Seventh International Wireless Communications and Mobile Computing Conference (IWCMC'11)*, pages 848–852.

Silberschatz, A., Peterson, J. L., and Galvin, P. B. (1991). *Operating System Concepts*. Addison-Wesley Longman Publishing Co., Inc.

Suthaputchakun, C. and Ganz, A. (2007). Priority Based Inter-Vehicle Communication in Vehicular Ad-Hoc Networks Using IEEE 802.11e. In *IEEE 65th Vehicular Technology Conference (VTC '2007)*, pages 2595–2599.

Uzcategui, R. and Acosta-Marum, G. (2009). Wave: A Tutorial. *IEEE Communications Magazine*, **47**(5), 126–133.

Wang, S. (2005). The Effects of Wireless Transmission Range on Path Lifetime in Vehicle-formed Mobile Ad Hoc Networks on Highways. In *IEEE International Conference on Communications (ICC'05).*, volume 5, pages 3177–3181.

Wayne, K. D. (2002). A Polynomial Combinatorial Algorithm For Generalized Minimum Cost Flow. *Mathematics of Operations Research*, **27**(3), 445–459.

Wu, S.-H., Chen, C.-M., and Chen, M.-S. (2010). An Asymmetric and Asynchronous Energy Conservation Protocol for Vehicular Networks. *IEEE Transactions on Mobile Computing*, **9**(1), 98–111.

Zhang, L., Wu, Q., Solanas, A., and Domingo-Ferrer, J. (2009). A Scalable Robust Authentication Protocol For Secure Vehicular Communications. *IEEE Transactions on Vehicular Technology*, **59**(4), 1606–1617.

Zhang, Y., Zhao, J., and Cao, G. (2007). On Scheduling Vehicle-Roadside Data Access. In *Proceedings of the 4th ACM International Workshop on Vehicular Ad Hoc Networks*, pages 10–19.

Zhang, Y., Zhao, J., and Cao, G. (2010). Service Scheduling of Vehicle-Roadside Data Access. *Mobile Networks and Applications*, **15**(1), 83–96.

Zhao, J., Arnold, T., Zhang, Y., and Cao, G. (2008). Extending Drive-Thru Data Access by Vehicle-to-Vehicle Relay. *Proceedings of the 5th ACM International Workshop on Vehicular Inter-Networking (VANET'08)*, pages 66–75.

Zou, F., Zhong, J., Wu, W., Du, D.-Z., and Lee, J. (2011). Energy-Efficient Roadside Unit Scheduling for Maintaining Connectivity in Vehicle Ad-Hoc Network. In *Proceedings*

of the 5th ACM International Conference on Ubiquitous Information Management and Communication, page 64.