

Network Coding in Multihop Wireless Networks: Throughput Analysis and Protocol Design

by
Zhenyu Yang

A Dissertation
Submitted to the Faculty
of the
WORCESTER POLYTECHNIC INSTITUTE
In partial fulfillment of the requirements for the
Degree of Doctor of Philosophy
in
Electrical and Computer Engineering

April, 2011

Approved:

Prof. Wenjing Lou
ECE Department
Dissertation Advisor

Prof. Xinming Huang
ECE Department
Dissertation Committee

Prof. Andrew G. Klein
ECE Department
Dissertation Committee

Prof. Craig E. Wills
CS Department
Dissertation Committee

Abstract

Multi-hop wireless networks have been widely considered as promising approaches to provide more convenient Internet access for their easy deployment, extended coverage, and low deployment cost. However, providing high-speed and reliable services in these networks is challenging due to the unreliable wireless links, broadcast nature of wireless transmissions, and frequent topology changes. On the other hand, network coding (NC) is a technique that could significantly improve the network throughput and the transmission reliability by allowing intermediate nodes to combine received packets. More recently proposed symbol level network coding (SLNC), which combines packets at smaller symbol scale, is a more powerful technique to mitigate the impact of lossy links and packet collisions in wireless networks. NC, especially SLNC, is thus a particular effective approach to providing higher data rate and better transmission reliability for applications such as mobile content distribution in multihop wireless networks.

This dissertation focuses on exploiting NC in multihop wireless networks. We studied the unique features of NC and designed a suite of distributed and localized algorithms and protocols for content distribution networks using NC and SLNC. We also carried out a theoretical study on the network capacity and performance bounds achievable by SLNC in mobile wireless networks.

We proposed CodeOn and CodePlay for popular content distribution and live multimedia streaming (LMS) in vehicular ad hoc networks (VANETs), respectively, where many important practical factors are taken into consideration, including vehicle distribution, mobility pattern, channel fading and packet collision. Specifically, CodeOn is a novel push-based popular content distribution scheme based on SLNC, where contents are actively broadcast to vehicles from road side access points and further distributed among vehicles using a cooperative VANET. In order to fully enjoy the benefits of SLNC, we proposed a suite of techniques to maximize the downloading rate, including a prioritized and localized relay selection mechanism where the selection criteria is based on the usefulness of contents possessed by vehicles, and a lightweight medium access protocol that naturally exploits the abundant concurrent transmission opportunities. CodePlay is designed for LMS applica-

tions in VANETs, which could fully take advantage of SLNC through a coordinated local push mechanism. Streaming contents are actively disseminated from dedicated sources to interested vehicles via local coordination of distributively selected relays, each of which will ensure smooth playback for vehicles nearby. CodeOn pursues a single objective of maximizing downloading rate, while CodePlay improves the performance of LMS service in terms of streaming rate, service delivery delay, and bandwidth efficiency simultaneously. CodeOn and CodePlay are among the first works that exploit the features of SLNC to simplify the protocol design whilst achieving better performance.

We also developed an analytical framework to compute the expected achievable throughput of mobile content distribution in VANETs using SLNC. We presented a general analytical model for the expected achievable throughput of SLNC in a static wireless network based on flow network theory and queuing theory. Then we further developed the model to derive the expected achievable accumulated throughput of a vehicle driving through the area of interest under a mobility pattern. Our proposed framework captures the effects of multiple practical factors, including vehicle distribution and mobility pattern, channel fading and packet collision, and we characterized the impacts of those factors on the expected achievable throughput. The results from this research are not only of interest from theoretical perspective but also provide insights and guidelines on protocol design in SLNC-based networks.

Dedicated To My Beloved Parents and Sister

Acknowledgements

Four years ago when I arrived at WPI, I was not sure what I as a graduate from CS department was going to do in the ECE department's wireless network and security lab. I am deeply indebted to my advisor, Professor Wenjing Lou, for making this difficult transition a lot easier with her wide knowledge and kindly advices. She was always there to meet and discuss about my ideas, to challenge me with sharpening questions, to give suggestions about the structure of the paper and to do proofread with great patience. She is the most responsible one and without her help, I could not have finished this dissertation. Her hard working and passion for research also has set an example that I would like to follow.

Besides Professor Lou, I like to thank Professor Xinming Huang, Professor Andrew G. Klein and Professor Craig E. Wills for serving on my dissertation committee, and for their good questions and insightful comments on my work.

I also wish to thank my previous colleagues Dr. Kai Zeng, Dr. Shucheng Yu and other labmates: Ming Li, Ning Cao, Hanfei Zhao and Qiben Yan for creating an intellectual and enjoyable atmosphere in the lab. I am especially indebted to Ming Li, who has put great effort on our collaboration.

This dissertation is dedicated to my parents, who always give me selfless love and support during all my life.

Contents

Abstract	i
Acknowledgements	iv
Table of Contents	v
List of Tables	ix
List of Figures	x
1 Introduction	1
1.1 Motivation	1
1.2 Network Coding	5
1.2.1 Brief Overview	5
1.2.2 Important Theoretical Results	8
1.2.2.1 Intra-flow network coding	8
1.2.2.2 Inter-flow network coding	10
1.3 Thesis Overview	11
2 Reliable Broadcast in Wireless Mesh Networks	15
2.1 Introduction	15
2.2 Related Work	18
2.3 Preliminaries	21
2.3.1 Network Model	21
2.3.2 Network Coding	21
2.4 Existing Schemes Analysis	22
2.4.1 Pacifier	22
2.4.2 AdapCode	24
2.5 R-CODE Design	25
2.5.1 Idea	25
2.5.2 Design	28
2.5.2.1 Basic Scheme	29
2.5.2.2 Dynamically Maintain the Guardian-Ward Relationship . .	30
2.5.2.3 Source Rate Limiting	31

2.6	Performance Evaluation	34
2.6.1	Simulation Settings	34
2.6.2	Number of Transmissions	36
2.6.3	Broadcast Latency	40
2.6.4	Discussion	41
2.7	Summary	41
3	Cooperative Popular Content Distribution in VANETs using SLNC	43
3.1	Introduction	43
3.2	Problem Formulation and Related Work	47
3.2.1	Problem Formulation	47
3.2.1.1	Model and assumptions	47
3.2.1.2	Objectives	49
3.2.2	Related work and our contributions	50
3.2.2.1	Network coding for content downloading	50
3.2.2.2	Transmission coordination in content downloading	52
3.2.2.3	Multi-channel compatibility	53
3.2.2.4	Other related works	53
3.3	Symbol-level Network Coding	54
3.3.1	A Brief Review of Symbol-level Network Coding	54
3.3.2	How VANET content distribution benefits from SLNC	56
3.4	The Design of CodeOn	58
3.4.1	Overview	59
3.4.2	Network Coding Method	61
3.4.3	Efficient Exchange of Content Reception Status	63
3.4.4	Distributed Relay Selection in Cooperative PCD	64
3.4.4.1	Node utility calculation	64
3.4.4.2	Transmission coordination	65
3.4.4.3	The merit of carrier sense under SLNC	68
3.4.5	Broadcast Content Scheduling	68
3.4.5.1	Content scheduling at APs	68
3.4.5.2	Content scheduling at vehicles	69
3.5	Performance Evaluation	70
3.5.1	Methodology	70
3.5.2	Simulation Settings	73
3.5.3	Results	73
3.5.3.1	Downloading performance	73
3.5.3.2	Fairness	75
3.5.3.3	Protocol efficiency	76
3.5.3.4	Discussion	78
3.6	Summary	80

4	Live Multimedia Streaming in VANETs using SLNC	85
4.1	Introduction	85
4.2	Related work	89
4.2.1	NC-based streaming schemes	89
4.2.2	Streaming schemes for VANETs	89
4.3	Problem Formulation	91
4.3.1	Model and Assumptions	91
4.3.2	Objectives	92
4.4	The Design of CodePlay	92
4.4.1	Design Rationale of CodePlay	92
4.4.1.1	Push-based Network Coding is Good for LMS	92
4.4.1.2	SLNC Potentially Performs Better than PLNC in VANETs	93
4.4.1.3	Make All Ends Meet — Coordinated Local Push with SLNC	94
4.4.2	Design overview	94
4.4.2.1	Global View	95
4.4.2.2	Local View	96
4.4.3	LMS Using Symbol Level Network Coding	99
4.4.4	Coordinated and Distributed Relay Selection	101
4.4.4.1	Distributed Coordinator Selection	102
4.4.4.2	Relay Selection	103
4.4.4.3	Determining the Segment Length	105
4.4.5	Transmission Coordination of Relays	107
4.4.5.1	Spatial Coordination	107
4.4.5.2	Temporal Coordination	109
4.4.6	OLRR: Opportunistic LRR Scheduling for Sparse VANETs	109
4.5	Performance Evaluation	113
4.5.1	Effect of Number of LMS Sources	115
4.5.2	Effect of D_{opt}	116
4.5.3	Effect of Initial Buffering Delay	117
4.5.4	Effect of Traffic Density	119
4.5.5	Effect of Opportunistic Scheduling	119
4.6	Summary	120
5	Throughput Analysis of Cooperative Mobile Content Distribution in VANETs using SLNC	125
5.1	Introduction	125
5.2	Related Work	127
5.2.1	Capacity Scaling Law of Wireless Networks	127
5.2.2	Achievable Throughput of MCD in VANETs	129
5.3	Achievable Throughput of Symbol-level Network Coding in Wireless Network	130
5.3.1	System model for SLNC in Wireless Network	130
5.3.2	Achievable Throughput for SLNC in Wireless Network	132
5.4	Throughput analysis of cooperative MCD system using SLNC	137
5.4.1	Problem Formulation	138

5.4.2	Symbol-Level Link Average Arrival Rate	140
5.4.2.1	Link Average Arrival Rate Considering Channel Fading . .	141
5.4.2.2	Link Average Arrival Rate Considering Collision	143
5.4.3	Achievable throughput of cooperative MCD system	147
5.4.4	Expected Downloading Volume with Mobility	148
5.5	Numerical Evaluation and Discussion	149
5.6	Summary	152
6	Conclusions and Future Research	157
6.1	Summary	157
6.2	Further Research	158
6.2.1	Multi-Flow MCD in VANETs	158
6.2.2	Multi-Rate Adaptive SLNC	158
6.2.3	MCD under Disconnected VANET (DTN) Scenarios	159
	Bibliography	161

List of Tables

2.1	Comparison between Pacifier and R-Code	28
2.2	Optimal coding schemes	35
2.3	Simulation parameters	35
3.1	Frequently used notations	59
3.2	Simulation parameter settings	71
3.3	Protocol efficiency (Total number of pieces in the file: 1600).	76
4.1	Parameter Settings	113
5.1	Simulation parameters	150

List of Figures

1.1	An illustration of the concept of network coding.	6
1.2	A simple intra-session network coding example.	6
1.3	A simple inter-session network coding example.	7
2.1	A simple example of AdapCode. The packet delivery probability of each link is given.	24
2.2	A simple example to show the intuition of R-Code. The weight of each link is given.	27
2.3	R-Code packet header format.	29
2.4	A simple example of dynamically maintaining the guardian-ward relationship	32
2.5	Grid size=200m	38
2.6	Grid size=250m	39
3.1	The architecture for PCD. Inside the AP coverage, AP broadcasts and vehicles receive; outside the AP coverage, vehicles distribute their received contents cooperatively.	48
3.2	The time and channel utilization of each vehicle and each AP.	49
3.3	The topology for the example in Fig. 3.4. Left: numbers on the edges (links) show the symbol error probabilities; right: corresponding packet error probabilities.	55
3.4	Symbol level network coding in VANET content distribution. S : source node; V_1 and V_2 : downloading vehicles & relays.	57
3.5	Overview of cooperative content distribution in CodeOn.	58
3.6	Comparison between the overhead of piece division and packet division, when both uses run-length SLNC.	61
3.7	The average rank representation of a file's reception status at node u	64
3.8	(a) Highway scenario. (b) Urban scenario.	70
3.9	Downloading progresses.	81
3.10	Downloading delays and rates.	82
3.11	The distributions of downloading delays.	83
4.1	The architecture for LMS.	91
4.2	The illustration of smooth propagation. Shaded segments are selected to transmit during the corresponding service time slots.	97

4.3	The concept of coordinated local push.	98
4.4	playback buffer and priority generations.	102
4.5	The format of piggybacked information (in Byte), where N is the size of the playback buffer (in generation).	103
4.6	The average symbol reception probability when CR=277m, ER=700m. Data rate is 12Mbps, and Nakagami fading model is used.	106
4.7	The optimal distance between two adjacent relays under various data communication ranges.	106
4.8	Snapshots of 3 sparse VANETs (T=1, R=3). The road segment ID is illustrated above each road segment and the vehicles represent corresponding coordinators. Those dark shaded segments in each snapshot are designated to be scheduled in this time slot.	110
4.9	Comparison between using one and two APs, dense highway, source rate=12KB/s, initial buffering delay=24Sec.	114
4.10	Comparison between different distance of adjacent concurrent transmitting relays, dense highway, source rate=12KB/s, initial buffering delay=24Sec.	116
4.11	Fixed initial buffering delay, varying source rates. Sparse highway.	121
4.12	Fixed rate, varying initial buffering delay. Sparse highway.	122
4.13	Impact of traffic density. Dense highway.	123
4.14	Effect of opportunistic transmission scheduling.	123
5.1	The example of flows from the perspective of SLNC ('X' means the corresponding symbol is corrupted).	131
5.2	Graph models for PLNC and SLNC	133
5.3	two-link tandem network. Left: packet level queueing network; right:symbol level multi-queueing network.	133
5.4	The architecture for MCD. AP owns contents to broadcast to vehicles; vehicles distribute their received contents cooperatively.	138
5.5	Average achievable throughput of MCD in vehicular network with exponential inter-distance distribution	154
5.6	Comparison of different inter-distance distribution	155
5.7	The expected downloading volume of different average inter-distances with fixed velocity of 20m/s	156
5.8	The expected downloading volume of different average inter-distances with traffic model	156

Chapter 1

Introduction

1.1 Motivation

In the last decade, multi-hop wireless networks, including wireless sensor networks (WSNs), wireless mesh networks (WMNs), mobile ad hoc networks (MANETs), vehicular ad hoc networks (VANETs), etc, have emerged as promising approaches to provide more convenient Internet access due to their extended cover range, easy deployment and low cost [8,9,17,36]. For example, WSN deployed on an active volcano enables scientists to monitor the situation there conveniently by letting the sensor nodes transfer collected data to remote server through Internet [109], and WMN deployed at rural area makes the residents enjoy the Internet access without relying on costly wired infrastructure [9]. However, to provide high-speed and reliable services in these networks is challenging due to the following reasons. Firstly, the quality of wireless links are time-varying which are mainly caused by various kinds of propagation fading and therefore may incur intermittent link behavior. Secondly, since the wireless medium is broadcast in nature, the transmissions on neighboring links tend to interfere with each others. Lastly, the node mobility in MANETs and VANETs can

cause frequently unpredictable topology changes and make effectively and efficiently finding and maintaining the multiple hop routes a critical concern in the network protocol design.

Traditional routing protocols [49,88] for multihop wireless networks have followed the concept of routing in wired networks, where the links are considered as point-to-point and the routes are usually maintained by finding the shortest path(s) between a source and destination based on given metrics (least number of hops, least transmission energy cost, etc). However, since the fact that unreliable and broadcast wireless links are totally different from reliable and point-to-point wired links, those routing protocols tend to experience poor performance such as low packet delivery ratio and high control overhead. To better exploit the broadcast nature of wireless links, a new routing paradigm, known as **opportunistic routing** (OR) [12,13,29,61,96,132] has recently been proposed. OR integrates the network and MAC layers. Instead of picking single relay node in each forwarding hop, OR selects a set of candidate nodes to forward a packet in network layer where each of them is closer to the destination than the last hop relay. At the MAC layer, only one node is selected as the actual forwarder dynamically based on the instantaneous wireless channel condition and node availability at the time of transmission. Since it takes advantages of the spatial diversity and broadcast nature of wireless communications, OR is an efficient mechanism to combat the time-varying links and could improve the network throughput [12,29,122–124] compared with traditional routing. However, OR also introduces a difficult challenge for protocol designs based on it. Without proper coordination between multiple candidates in each hop, all of them that overheard the packet from the last hop relay node will attempt to forward it, which creates spurious retransmissions and wastes bandwidth. To address this challenge, a complicated and costly coordination scheme is needed to run among the candidate nodes, so that they can reach a consensus on which one should actually forward the packet. Network

coding (NC) [6], a totally new packet forwarding paradigm proposed recently, has been shown that could address this challenge in a natural and efficient way with significantly less coordination overhead [19, 54, 60, 121]. Network coding basically breaks the traditional store-and-forward packet forwarding paradigm. It allows intermediate nodes to combine previously received packets to generate coded packets on the outgoing links. In protocols combined OR with NC techniques, different forwarding candidates in each hop will forward coded packets based on their current reception status and all those coded packets are naturally different with each other with high probability [38]. Thus there is no need for complicated and costly coordination between multiple forwarding candidates. Besides simplifying protocol design, NC also could bring other important benefits in terms of throughput, reliability, robustness and adaptability [6, 28, 31, 73], which will be illustrated in detail in section 1.2.

Reliable content distribution is a key function to enable various multi-hop wireless networks operate well. For instance, it is necessary for software updates which may happen at the initial deployment and maintenance period, or is used in multimedia services like video/audio broadcasting in WSNs and WMNs. In mobile VANETs, such applications includes: live video broadcast of road traffic and condition to vehicles driving towards it for intelligent navigation, which is especially useful during inclement weathers; periodical broadcasts of multimedia advertisements of local businesses in a city to vehicles driving through a segment of suburban highway (like a digital billboard); the disseminations of an accurate update of the GPS map about a city or a scenic area, etc. A key requirement of these applications is to strictly guarantee that every receiver has to download every bit of the broadcasted content. Benefits of NC described above make it widely adopted in reliable broadcast protocol design as an effective approach to improving the bandwidth efficiency and protocol robustness [59], [121], [86], [19], [27], [119]. More recently, symbol level network coding (SLNC) has been proposed [54]. In contrast to traditional packet level network

coding (PLNC), which combines packets at the packet level, SLNC allows intermediate nodes to combine packets at symbol level, where a symbol is typically composed of several physical layer symbols. SLNC allows a node to recover correctly received symbols from erroneous packets. Since symbol error rate is smaller than the packet error rate, in addition to the benefits one can gain from PLNC, SLNC provides better error tolerance thus increased successful packet reception rate. A further study of SLNC also shows that due to the better error tolerance, SLNC in fact enables better spatial reusability by allowing concurrent transmissions within shorter distances [66, 118]. Clearly, SLNC is a more powerful technique to mitigate the impact of lossy links and packet collisions in wireless networks. In reality, to determine if the symbols are received correctly or not, the intermediate node uses soft hints [107] as an estimator, which refers to the confidence values computed in physical layer when decoding symbols, this information could be exposed to the upper network layer through SoftPHY interface [45] in a PHY-independent manner by annotating bits with additional hints. Only those being estimated as correctly received symbols would be processed and forwarded. because there do exist erroneous symbols being estimated as correct and seep through, SLNC uses a rateless end-to-end error correcting component (maximum rank distance codes) to correct them at destinations, which works independently with the network coding component.

How NC can help to improve the network capacity has been extensively studied in recent years [6, 19, 24, 27, 28, 38, 52, 57, 72, 75, 125]. There have also been many studies on NC-based content distribution in both wired and relatively stable wireless networks [20, 33, 62, 67, 106, 115, 119, 131]. However, these works have not exploit mobility fully, especially in highly mobile VANETs. There have been a few works on NC-based content distribution in vehicular networks [7, 63, 87] and their focus has been on practical protocol design and the results are rudimentary. The benefits of NC tend to be offset by severe packet collisions due to lack of proper transmission coordination

mechanisms among vehicles. As a very powerful but relatively new concept, SLNC's potential to improve the bandwidth efficiency has not been fully exploited too. There is a lack of theoretical foundation and understanding on the performance limits such as achievable throughput by SLNC, especially content distribution in high mobility scenario. Practical protocol design is another issue. SLNC-based content distribution protocol design for mobile networks is not a trivial task. How to devise the protocol such that these great potentials can be fully realized is also challenging and rarely addressed.

This dissertation carries out a comprehensive study on the throughput analysis and protocol design of network coding in multi-hop wireless networks, with specific focus on the SLNC and VANETs.

1.2 Network Coding

1.2.1 Brief Overview

The concept of network coding is firstly introduced in the seminal paper [6] in 2000, which basically allows the intermediate relay nodes to combine the received packets for generating encoded packets on the outgoing links, as illustrated in Fig.1.1. Generally, network coding in packet networks can be classified into two types: (1) intra-session coding where coding is restricted to packets belonging to the same session or connection. This can be illustrated by the Fig.1.2. S multicasts to node E and F. All links have capacity 1. With network coding applied (by xoring packet P_a and P_b on link CD), the achievable throughput are 2 for both receivers, the same as each of them enjoys the whole network. Without network coding, however, the total achievable throughput for E and F is less (if we assume they share the critical link CD

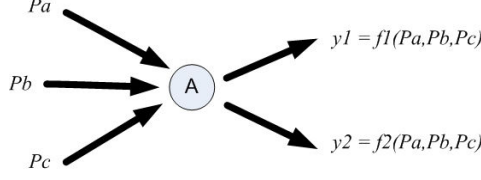


Figure 1.1: An illustration of the concept of network coding.

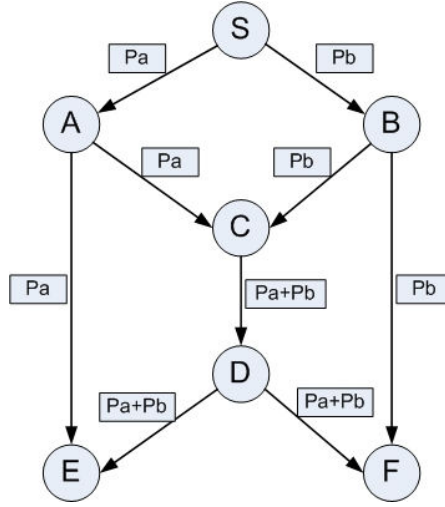


Figure 1.2: A simple intra-session network coding example.

equally, the achievable throughput for both of them is 1.5). (2) inter-session coding where coding is allowed among packets belonging to possibly different sessions. This can be illustrated by Fig.1.3. Two nodes A and B want to exchange a packet. Due to the limited wireless communication range, an intermediate node S should play the role of relay. Without network coding applied, it is easy to deduce that 4 transmissions in total are needed to make the exchange. While if we combine the P_a from A and P_b from B at relay S and let it broadcast the coded packet, the total required transmission number reduces to 3.

Usually network coding techniques operate above network layer in the standard OSI model, like the PLNC and SLNC. However, the idea of network coding could also be applied in other lower layers. Zhang et al. [127] proposed physical-layer network

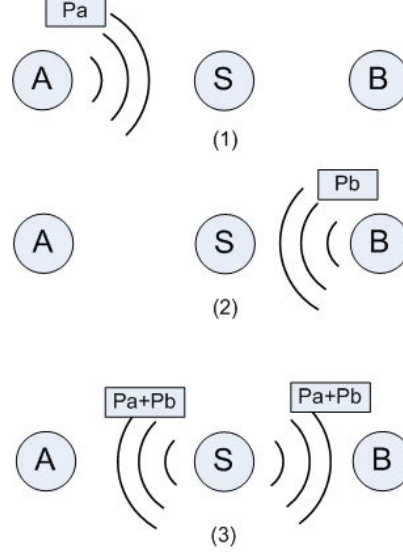


Figure 1.3: A simple inter-session network coding example.

coding (PNC) for wireless networks. The main idea of PNC is to deal with electromagnetic signal reception and modulation directly at lower physical layer. Through a proper modulation-and-demodulation technique at relay nodes, the mix of electromagnetic signals can be mapped to the mix of digital bit streams, so that the interference becomes part of the arithmetic operation in network coding. However, PNC assumes symbol ¹-level synchronization, carrier-frequency synchronization, and carrier-phase synchronization, which is hard to satisfy in practice since unlikely that two signals arrive at the exact same time at the router and incur the same distortion over the wireless medium. Katti et al. [53] presented a novel algorithm for analog network coding (ANC) and also implemented it in software radio platform. The main idea of ANC is similar to PNC but without the need for restrict synchronization assumption.

¹The symbol here refers to the physical layer symbol (PHY symbol) by the modulation scheme (e.g., groups of 4 bits in a 16-QAM scheme), which is different with the symbol defined in SLNC, where the symbol length is a system parameter and each symbol can include multiple PHY symbols. In the rest of the dissertation, we refer the latter as network-level symbol.

Since PNC and ANC deeply involve into the cross-layer design, accommodating of them into the currently widely used TCP/IP architecture needs large amount of effort [55]. On the other hand, PLNC and SLNC work on network layer and combining them with TCP/IP architecture mainly relies on software development rather than hardware modification, which is much easier to be implemented and widely deployed on the current legacy network infrastructure [33, 131]. Thus the research efforts in this dissertation are mainly focus on throughput analysis and protocol design based on PLNC and SLNC.

1.2.2 Important Theoretical Results

1.2.2.1 Intra-flow network coding

Ahlswede et al. proved that by adopting intra-session network coding², the single source could disseminate information to multiple receivers in a rate approaching the smallest minimum cut between the source and any receiver, which is also the theoretical upper bound, based on the assumption that the encoding symbol size approaches infinity [6]. This shows the fundamental benefit of network coding since under the store-and-forward transmission paradigm, the capacity limit is not achievable for multicast session in general networks. The basic reason for this benefit comes from the fact that network coding considers the flows of packets as information flows where each of the flows could effectively be considered as the only user of the whole network, which is totally different from commodity flows where all the flows in the networks have to share the communication links. In the following works, Li et al. [68] proved that linear coding with finite symbol size is sufficient for achieving capacity

²Without explicit explanation, network coding in the rest of the dissertation is referred to PLNC technique.

of multicast network. Koetter and Medard [57] extended the conclusions to arbitrary networks and robust networking with delay and cycles. Ho et al. [38] presented a distributed random linear network coding scheme for data transmission in more general multisource multicast networks, where network nodes independently and randomly select linear mappings from inputs onto output links over given finite field \mathbb{F} . One prominent result of [38] is that: if the source file is divided into k original packets, each receiver could recover the whole file from any k received coded packets with probability $1 - 1/|\mathbb{F}|$. For sufficiently large finite field, e.g., widely used \mathbb{F}_{2^8} , this probability is very close to 1 [113]. In [39], they also considered the problem of multiple multicast sessions with intra-session network coding in time-varying networks and identified the network-layer capacity region of input rates that can be stably supported. Lun et al. [72, 74] showed that random linear network coding could approximately achieve the throughput capacity for both single unicast and single multicast sessions in both wireline and wireless lossy networks.

Besides the capability of improving throughput, network coding could also benefit in terms of energy efficiency, reliability, robustness and adaptability. Wu et al. showed in [114] that the minimum-energy-per-bit for multicasting in a mobile ad hoc network can be attained by adopting network coding and NC enables the optimal solution to be found in polynomial time. This is in sharp contrast with the NP-hardness of constructing the minimum-energy multicast tree as the optimal routing solution under traditional store-and-forwarding paradigm. Lun et al. [73] showed that when network coding is used, the problem of establishing minimum-cost multicast connections equates to two effectively decoupled problems: one of determining the subgraph to code over (the rate to inject packets on each link) to support the multicast session, and the other of determining the code to use over that subgraph (the contents of those packets). While the latter problem has already been solved in completely decentralized approach by random linear coding technique, Lun et al. tackled the former

problem by presenting an optimal, decentralized method to find the minimum-cost subgraphs. Compared with [114] where the cost function (energy paid for each transmitted bit) considered is linear, the approach proposed in [73] is applicable for more general cost function including linear and strictly convex cost functions. Ghaderi et al. [31] analyzed the performance of reliability mechanisms based on network coding for tree-based reliable multicasting in wireless networks and compared them with rateless codes and automatic repeat request (ARQ). It is showed that the reliability gain of network coding compared to link-by-link ARQ is $\Theta(\log K / \log \log K)$, where K is the fan-out degree of the nodes in the tree.

1.2.2.2 Inter-flow network coding

Compared with intra-session network coding, which has been extensively studied, inter-session network coding is more complicated and less understood from the theoretical point of view. In [23], Dougherty et al. proved that there exists particular multisource network scenarios where linear coding operation is not sufficient to achieve the optimal network coding capacity. For networks with multi-pair unicast sessions, where each node is a source and will randomly select another node in the network as destination, Liu et al. [70] showed that network coding provides no order difference improvement on throughput compared to routing, which still scales as $\Theta(1/\sqrt{n \log n})$ [35] where n is the total number of nodes in the network. However, the constant factor improvement of network coding is still important in practice. [25, 40, 104] presented a class of suboptimal, yet practical code construction techniques. In this class of codes, network coding is limited to XOR coding between pairs of sessions. Two uncoded packets belonging to different sessions can be coded together to form a coded packet in order to share capacity on one or more hops. This coded packet is subsequently replicated along the routing paths of the two sessions. In the end of the shared hops, the

coded packet could be decoded by XORing with the corresponding original packet, then the decoded packets can be subsequently re-encoded. These class of network codes could achieve some capacity region which is not necessarily feasible with pure store-and-forwarding, however, they failed to provide eligible network codes for all the possible network capacity region that are feasible with linear network coding.

1.3 Thesis Overview

The contents of each chapter are described as follows.

Chapter 2 of this dissertation studies the reliable broadcast in static wireless mesh networks. In this chapter, we put forward R-Code, a network coding-based reliable broadcast protocol. The main idea of R-Code composed of two parts: firstly, we introduce a guardian-ward relationship between neighboring nodes that effectively distributes the responsibility of reliable information delivery: from the global responsibility of the source to the localized responsibilities of guardians to their corresponding wards; secondly, we use a link quality-based minimum spanning tree as a backbone to guide the selection of guardians adaptively and the transmission of coded packets accordingly. Besides, opportunistic overhearing is also utilized to improve the performance of the protocol. Extensive simulation results show that R-Code achieves 100% packet delivery latency, compared with state-of-the-art reliable broadcast protocols.

Chapter 3 of this dissertation studies the popular content distribution in VANETs. In this chapter, we introduce CodeOn, a novel push-based popular content distribution scheme based on SLNC, where contents are actively broadcasted to vehicles from road side access points and further distributed among vehicles using a cooperative VANET. In order to fully enjoy the benefits of SLNC, we propose a suite of techniques to maximize the downloading rate, including a prioritized and local-

ized relay selection mechanism where the selection criteria is based on the usefulness of vehicles possessed contents, and a lightweight medium access protocol that naturally exploits the abundant concurrent transmission opportunities. We also propose additional mechanisms to reduce the protocol overhead without sacrificing the performance. Extensive simulation results show that, under a wide range of scenarios, CodeOn significantly outperforms state-of-the-art schemes based on network coding.

Chapter 4 of this dissertation studies how to provide live multimedia streaming (LMS) services in VANETs. In this chapter, we introduce CodePlay based on SLNC, which fully takes advantage of SLNC through a coordinated local push mechanism. Streaming contents are actively disseminated from dedicated sources to interested vehicles via local coordination of distributively selected relays, each of which will ensure smooth playback for vehicles nearby. Different from CodeOn, where the single objective of maximizing downloading rate is pursued, CodePlay is designed to simultaneously improve the performance of LMS service in terms of streaming rate, service delivery delay and bandwidth efficiency. Extensive simulations show that simply replace the SLNC with PLNC technique in previous schemes which provide live multimedia streaming can not provide satisfiable user experience, and special scheme design based on the unique characteristics of SLNC proposed in CodePlay is necessary for future live multimedia streaming services in VANET.

Chapter 5 of this dissertation endeavors to develop a theoretical model to compute the achievable throughput of cooperative mobile content distribution in VANETs using SLNC. In this chapter, we first present a general analytical model for the achievable throughput of SLNC in fixed wireless networks based on flow network and queuing theory, and then tailor this model to derive the achievable accumulated throughput of a vehicle driving through the area of interest under a mobile highway VANET scenario. Our proposed model is unique since it captures the effects of multiple prac-

tical factors, including vehicle distribution and mobility pattern, channel fading and packet collision. We also study the case when the physical symbol reception in each packet is correlated. Through numerical results, we demonstrate how the achievable throughput decreases with the distance of a vehicle from the source, and compares the throughput gain of SLNC over PLNC under different vehicle distributions and mobility. This analytical framework can provide valuable insights and guidelines for future protocol design in mobile content distribution scheme for VANETs.

Chapter 2

Reliable Broadcast in Wireless Mesh Networks

2.1 Introduction

Wireless mesh network (WMN) emerges as a promising technique to provide high-bandwidth Internet access to a large number of mobile devices in a specific area. Broadcast is an important function in WMNs. For example, it is necessary for software updates which may happen at the initial deployment and maintenance period, or is used in multimedia services like video/audio downloading. A key requirement of these applications is to strictly guarantee 100% packet delivery ratio (PDR), which means every node has to download every bit of the broadcasted file. In addition, efficiency is another important concern. Since other normal unicast traffics may exist in the network at any time, broadcast applications should have good coexistence with these traffics, which translates into consuming minimal amount of network bandwidth and disseminating the file with low latency.

It is nontrivial to design an efficient reliable broadcast protocols for real WMNs. The fundamental challenge comes from the unreliable nature of the wireless link [78], which is due to packet losses caused by channel fading and interferences. In order to guarantee 100% PDR with those unreliable links, some previous schemes [85, 97] use automatic repeat request (ARQ) technique, which requires the receivers to provide explicit feedbacks of the packet reception status to the source. However, this will cause “ACK implosion” problem which may incur a large amount of redundant transmissions. Other schemes [56, 84, 94] combine ARQ with forward error correction (FEC) technique to reduce the transmission overhead while still guaranteeing 100% PDR. Yet, these techniques still consider the wireless link as point-to-point, and neglect the fact that wireless medium is broadcast in nature. This leads to duplicate transmissions at intermediate nodes, which are not efficient enough.

Recently, network coding (NC) has been proposed as an effective technique to increase the network bandwidth-efficiency [28]. In contrast to FEC, NC gives intermediate nodes the ability of randomly encoding different packets received previously into one output packet. Thus, although multiple intermediate nodes may receive the same packet, they will broadcast different re-coded packets that are linearly independent with each other with high probability. Each of these re-coded packets can benefit other nodes that overhear it, which avoids the duplicate transmissions. Theoretical analysis has demonstrated that NC is able to approach the multicast and broadcast capacity in multi-hop wireless networks [5, 72]. Due to the high complexity of implementing network coding, practical NC-based broadcast schemes have also been proposed [19, 27, 42, 59, 86, 121], where NC is shown to have a noticeable gain in bandwidth-efficiency. However, most of these schemes only provide reliability with best effort rather than guaranteeing 100% PDR, with the exception of AdapCode [42] and Pacifier [59]. However, AdapCode is purposefully designed for wireless sensor networks and is not efficient when directly applied into WMNs. While MORE

and Pacifier focuses on multicast in WMNs, they do not consider to exploit the specific characteristic of broadcast, that is, every node has to receive the whole file. In fact, when a node has received a certain amount of information, it can be regarded as a temporary source, which can guarantee the reliable reception of its neighbors in an efficient way.

In this chapter, we propose R-Code, an efficient distributed **R**eliable broadcast protocol in WMNs based on network **C**oding which guarantees 100% PDR. The core idea of R-Code is to establish a guardian-ward relationship between neighboring nodes, so that the global responsibility of the source to ensure the reliable reception of all the nodes in the network is distributed to all the guardians. This is because a guardian is a temporary source that is much closer to its wards than the original source, thereby it can guarantee their reliable reception of the file more efficiently. A link quality-based minimum spanning tree is constructed to guide the selection of guardians and packet transmissions accordingly. A guardian is the best node in a ward's neighborhood to ensure the reliable reception of the ward with the least number of transmissions. The guardian-ward relationship is adaptively maintained throughout the broadcast session to exploit the benefit of opportunistic overhearing. In addition, intra-flow NC is adopted to further reduce the total number of transmissions and simplify the coordination between multiple transmitters. Moreover, R-Code applies a source rate limiting mechanism to alleviate the collisions in the network. We evaluate R-Code and compare it with AdapCode by extensive simulations. The simulation results show that R-Code uses up to 15% less number of transmissions and 65% shorter broadcast latency than that in AdapCode to disseminate the same file.

The rest of the chapter is organized as follows: we give related work in Section 2.2. In Section 2.3, we describe the preliminaries. The analysis of existing schemes is presented in Section 2.4. In Section 2.5, we introduce the design of R-Code protocol

in detail. Section 2.6 presents the simulation results and Section 2.7 wraps up the chapter.

2.2 Related Work

Broadcast in multi-hop wireless networks has been studied for decades. From the perspective of reliability, those proposed schemes can be divided into two categories: (1) schemes that provide reliable broadcast services with best effort, a good survey of which can be found in [111]; (2) schemes that guarantee 100% PDR strictly. Some of them use ARQ technique [85, 97], where the source requires feedbacks from all the receivers, leading to the well-known ACK implosion problem and also incurring large amount of redundant transmissions. Others combine FEC with ARQ [56, 84, 94], which can increase the throughput. However, all these schemes do not explicitly take advantage of the broadcast nature of wireless medium, and thus suffer from duplicate transmissions.

Many NC-based scheme also have been proposed. Among them, Fragouli et al. [27] study NC-based efficient broadcast from both theoretical and practical point of views. They show that NC is able to increase the bandwidth/energy efficiency by a constant factor in fixed networks. They also propose a probabilistic forwarding-based algorithm for random networks which shows significant overhead improvement over probabilistic flooding. However, their algorithm does not guarantee 100% PDR. MORE [19] is the first practical NC-based routing protocol that achieves high throughput and guarantees 100% PDR, for both unicast and multicast sessions. The main idea of MORE is to combine opportunistic routing [22] with network coding, which eliminates the need of complicated coordination mechanism between multiple forwarders in pure opportunistic routing. However, MORE is inefficient when applied to multi-

cast, since almost every node in the network may become a forwarding node (FN), which can cause heavy congestion [59]. Moreover, due to the constraint on the encoding/decoding complexity, the source has to divide the file into batches and transmit them sequentially. In particular, the source works in a stop-and-wait fashion, which means it needs to wait till a batch is received by all receivers before moving to the next batch. This makes MORE suffer from the “crying baby” problem [41]. Namely, when one receiver has poor connection to the source, trying to ensure 100% PDR of this receiver will make other receivers wait unnecessarily which can heavily degrade the performance of the whole protocol.

In order to address those weaknesses of MORE, Koutsonikolas et al. propose Pacifier [59], a high-throughput, reliable multicast protocol. In Pacifier, the source builds a shortest-ETX [22] tree, which is the union of all the shortest-ETX paths from the source to each receiver, to guide the multicast process. Since only non-leaf nodes of the tree are selected as FNs, the number of forwarding nodes is reduced significantly compared with MORE. Similar to MORE, Pacifier also applies intra-flow network coding and lets the source assign a TX_credit for each FN. This TX_credit is the expected number of transmissions this FN should make for each coded packet it receives from an upstream FN in order to ensure that each of its children receives at least one packet. To solve the crying baby problem, Pacifier lets the source send batches of packets in a round-robin way rather than the stop-and-wait fashion adopted in MORE. It also applies a source rate limiting mechanism to further reduce the congestion. However, both MORE and Pacifier are source routing protocols, which includes the list of FNs and their $TX_credits$ in the header of each transmitted packet. This makes them less scalable. Moreover, they do not exploit the specific characteristic of broadcast session and each receiver’s reliable reception has to be guaranteed by the source through end-to-end ACKs, which is inefficient.

AdapCode [42] is a reliable broadcast protocol which is used for code updates in wireless sensor networks. Because we will compare R-Code against AdapCode in our evaluation, a brief overview of its two major components is given below.

Compressed forwarding. AdapCode works in a similar way like probabilistic forwarding. However, the forwarding is based on batch rather than single packet. That is, each node only transmits packets after receiving the whole batch. For a received batch which contains k packets, k/N coded packets is transmitted. The number N is called the “coding scheme”, which is selected adaptively according to the number of neighbors. In this way, Adapcode reduces the number of transmissions significantly compared with pure flooding.

“Timer+NACK” mechanism. AdapCode ensures 100% PDR by a Timer+ NACK mechanism, which runs as follows: each node i keeps a count-down negative ACK (NACK) timer and this timer will be restored to initial value once the node receives a packet. When the NACK timer counts to 0 and i still does not get enough packets for decoding, it broadcasts a NACK to request for the needed packets. Each of i ’s neighbors that overhears this NACK and possesses those required packets, is eligible to respond. In order to reduce the risk of unnecessary simultaneous responses, all those eligible responders will go through a coordination process. That is, each of them delays for a random period of time before responding to see if any other node is replying to this NACK. If no reply is heard during this period, it will respond to this NACK by sending all the requested packets. AdapCode also adopts a “lazy NACK” mechanism to reduce the number of NACKs, which requires each node to reset its NACK timer to avoid sending duplicate NACK if it overhears another one.

2.3 Preliminaries

2.3.1 Network Model

The WMN considered in this chapter consists of a number of wireless mesh routers that communicate with each other by radio transmission. Those mesh routers are static but not energy limited. The WMN connects to the Internet through some gateway routers. The broadcast application is one-to-all, where a gateway router is always the source that wants to disseminate a file to all the other routers in the WMN. The WMN is modelled as a weighted undirected graph $G(V, E)$, where V is the set of nodes (mesh routers) and E is the set of links. Two nodes i and j are considered to be connected if $w_{i,j} \geq w_{threshold}$, where $w_{i,j}$ is the weight of link (i, j) and $w_{threshold}$ is a given threshold value. $w_{i,j}$ is defined as the expected transmission count (ETX) [22] between i and j ¹. We also assume that for one transmission, the packet losses in different receivers are independent [79].

2.3.2 Network Coding

We use intra-flow random linear network coding in this chapter. In order to reduce the packet header overhead and encoding/decoding complexity, the source divides the broadcast file into small batches and send them sequentially. Each batch contains k original packets, denoted as $p_i, i = 1, 2, \dots, k$. The source keeps broadcasting coded packet of the current batch until all the receivers decode this batch successfully, then it moves to the next batch. We choose k to be 32 in our scheme, which is the same as in [19, 59]. Each coded packet x is a linear combination of all the packets in the

¹Each node will broadcast beacon messages every T seconds to estimate the link qualities to its neighbors.

batch: $x = \sum_{j=1}^k \alpha_j p_j$, where $\langle \alpha_j \rangle$ is the encoding vector. Each α_j is randomly selected from a Galois Field $GF(2^q)$. We choose q to be 8 in our scheme, the same as in [19, 59]. Every coded packet in transmission includes the encoding vector in the packet header. When a node receives a packet, it checks if the encoding vector of this packet is linearly dependent with all the other encoding vectors of the packets received previously. If so, this packet is discarded since the information it carries can be deduced from those already received packets; otherwise, this packet is called an innovative packet and stored in a buffer. When transmission opportunities appear, an intermediate node broadcasts coded packets generated by the packets stored in the buffer currently. Once a node receives k such innovative packets, it can decode this batch to retrieve all the original packets by doing Gaussian elimination, whose computational complexity is $O(k^3)$.

2.4 Existing Schemes Analysis

In this section, we carry out a thorough analysis of Pacifier and AdapCode.

2.4.1 Pacifier

In MORE and Pacifier, the selection of FNs and the calculation of credit for each FN are both based on ETX-metric. As stated in [60], this makes their performance heavily depends on the accuracy of the link quality measurement. Unfortunately, with currently widely used measurement mechanisms which are based on periodical hello packets, the precise link quality measurement can only be obtained with very high overhead, which is not applicable in practice.

Moreover, although Pacifier addressed the crying baby problem by letting the

source send batches in a round-robin fashion, within the transmission of the same batch, it still suffers from the “ACK delay” problem [121]. That is, during the time between the ACK is sent from the receiver and it is received by the source, those coded packets sent by the source for the current batch are, in fact, unnecessary.

To solve this problem, Pacifier lets the source maintain a counter C_{S_i} for each batch i . C_{S_i} is the expected number of packets the source need to send to guarantee that at least one of the receivers receives the whole batch successfully. C_{S_i} decreases by one after each of the source’s transmission. The source moves to the next batch either when it receives an ACK or when C_{S_i} reaches zero. However, since the calculation of C_{S_i} is based on the link quality measurement, the performance of this mechanism is also quite sensitive to the accuracy of those estimations.

We can see that the ACK delay problem is caused by the requirement of end-to-end ACKs (from the receiver to the original source), which can be further ascribed to that, in Pacifier, the source has to directly guarantee the reliable receptions of every node. In detail, since all the selected FNs in Pacifier only act as forwarders, which passively relay what they received from upstream nodes. For a given node i that still needs more packets, only the original source can actively inject the required packets into the network which then are relayed by those FNs to the receiver. Thereby the original source has to take the responsibility of ensuring 100% PDR for all the nodes.

However, since in broadcast, every node has to receive the whole file reliably sooner or later, a FN can also play the role of a temporary source after receiving the whole file. Thus for a given node i that still needs more packets, if some neighbor of i is a temporary source, i can get packets more efficiently from this temporary source than from the original source. In section 2.5 we will show that by taking advantage of those temporary sources, we can design a reliable broadcast scheme which not only avoids ACK delay problem, but also does not require accurate link quality measurement.

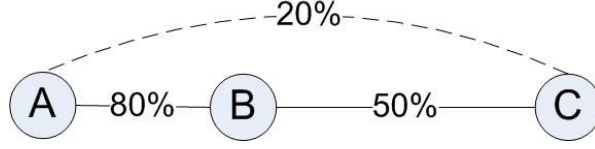


Figure 2.1: A simple example of AdapCode. The packet delivery probability of each link is given.

2.4.2 AdapCode

In AdapCode, each eligible responder has the responsibility of replying to the overheard NACK by sending those required packets. In order to avoid that some nodes are too heavily loaded to die out quickly, which can potentially disconnect the wireless sensor network, AdapCode designs a random selection mechanism for the NACK's responder which makes every eligible responder has the same probability to reply. However, we argue that this random selection mechanism makes AdapCode inefficient for WMNs. This could be illustrated by a simple example.

Fig. 2.1 presents a scenario of local broadcast process, where node A just decoded a batch and tries to disseminate it to its neighbors. For convenience, we let the batch size to be 10 and the coding schemes of all the nodes to be 1, which means A will transmit 10 coded packets for this batch. Thus, B, C will receive 8 and 2 packets², respectively. Since C has poorer connection to A, it will receive a packet with longer time interval expectedly which causes its NACK timer to fire earlier. Thus, it firstly sends out NACK, which in turn suppresses B's NACK because of the lazy NACK mechanism. In this NACK, C indicates the requirement of 8 more packets for successful decoding. Node A, as the only ³ eligible responder now, replies to this NACK by transmitting 8 packets. Because of the broadcast nature of wireless

²The calculation is based on expected values in all the examples in this chapter.

³The probability that B coincidentally possesses all the 8 required packets is quite small, we ignore it here for the convenience of analysis.

transmissions, B can also overhear these packets and obtain enough packets to decode the batch successfully. After decoding the whole batch, B will broadcast 10 coded packets according to the coding scheme, from which C can overhear 5 more packets. Now C has received $2 + 8 \times 0.2 + 10 \times 0.5 = 8.6$ packets, which is still not able to decode the batch. When its NACK timer fires again and sends another NACK, both A and B are eligible responders now. The total number of transmissions of this broadcast process is 32.9.

However, since the link quality of (B,C) is better than that of (A,C), if we let A cover ⁴ B at first and then B cover C deterministically, the total number of transmissions required is 27.5, which is more efficient. Since in practice, a broadcast session will experience similar situations quite often, we explicitly take the link quality into consideration in the design of R-Code.

2.5 R-CODE Design

2.5.1 Idea

The basic idea of R-Code is to distribute the responsibility of reliable information delivery from the original source to some selected nodes, called guardians. A guardian selects several nodes from its neighbors, called wards, and ensures the reliable reception of those wards. In order to promise that every node could be covered by a local guardian efficiently, R-Code uses a link quality-based minimum spanning tree as a backbone to guide the selection of guardians and wards.

In particular, the parent-child relationship between the node pairs in the tree

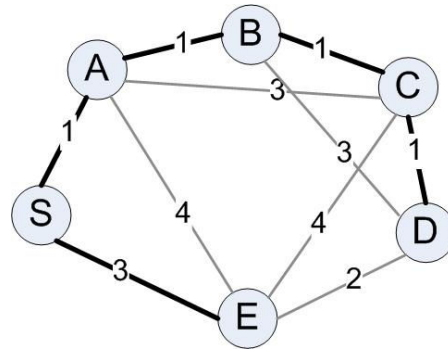
⁴A covers B means A has the responsibility of ensuring B's reliable reception.

can be translated into guardian-ward relationship. Therefore the original source only covers its children by keeping sending packets until get positive ACKs from all of these children. Upon receiving packet, each non-leaf child sends independently coded packets. A child stops sending only after receiving positive ACKs from all its children. This hop-by-hop cover process goes on until all the nodes receive this whole file reliably. Since these guardian-ward relationships are based on MST, which ensure that each node (except for the original source) will be assigned unique guardian, 100% PDR for all receivers are guaranteed. Also since all the ACKs only travel one hop distance, the ACK delay problem is avoided. Moreover, the optimal property of MST ensures each ward can always choose the best neighbor to be its guardian, avoiding the inefficiency of AdapCode. In addition, since R-Code defines specific guardian-ward relationship between neighboring node, it can use positive ACK rather than NACK adopted in AdapCode, which can also reduce the broadcast latency.

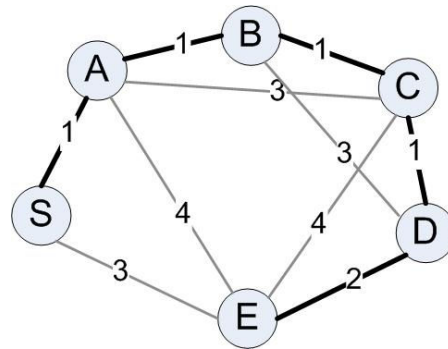
We use an example to explain the intuition underlying R-Code and compare the performance of R-Code with Pacifier. Without loss of generality, we assume the time is divided into slots and each transmission only happens at the beginning of a slot. Since every node has to send an ACK after receiving the packet, we ignore the cost of those ACKs for both protocols, which does not affect the comparison results. The broadcast latency is the number of slots from the time when the packet is broadcast firstly by the source to the time when it is received by the receiver ⁵.

Fig. 2.2 presents a network, which consists of 6 nodes. S is the source and wants to broadcast a batch reliably, the size of which is k . If we build the broadcast tree like Pacifier does, which is to combine all the best unicast paths from source to every other nodes, then we obtain the tree that is shown in Fig. 2.2(a) with bold lines. The numbers of transmissions generated by every node are shown in Table 2.1

⁵For the simplicity of analysis, we calculate the broadcast delay for one receiver at a time.



(a) Pacifier



(b) R-Code

Figure 2.2: A simple example to show the intuition of R-Code. The weight of each link is given.

and the average number of transmissions is $0.91k$. However, we observe that E is not covered by its best neighbor. For example, it can get a packet from D with 2 transmissions, which is more efficient than getting it directly from the source S . In a comparison, MST can make each node to be covered by its best neighbor, which is shown in Fig. 2.2(b) with bold lines. Now the average number of transmissions needed is $0.65k$, which is reduced by almost 30%.

However, this gain comes with cost in average broadcast latency, which increases about 7%, from 2.02 to 2.17. This tradeoff between the number of transmissions and broadcast latency reflects the difference of goals between R-Code and Pacifier.

Table 2.1: Comparison between Pacifier and R-Code

Node ID	Number of transmissions		Broadcast latency	
	Pacifier	R-Code	Pacifier	R-Code
S	$3k$	$1k$	0	0
A	$1k$	$1k$	1	1
B	$2/3k$	$2/3k$	2	2
C	$7/9k$	$7/9k$	$8/3$	$8/3$
D	0	$4/9k$	$31/9$	$31/9$
E	0	0	3	$35/9$
Avg.	$0.91k$	$0.65k$	2.02	2.17

That is, the primary goal of R-Code is to minimize the total number of transmissions while that of Pacifier is to achieve higher throughput, which, for a give file, can be translated into shorter broadcast latency. For this reason, we will not compare them in the evaluation part.

2.5.2 Design

R-Code works on top of the IP layer and the packet header format is shown in Fig. 2.3, which contains a type field that identifies data packet from control packet, the source's IP address, broadcast session id, batch index, the total number of batches for the file and code vector, which exists only in data packet and indicates the coefficients based on which the coded packet is generated from the original packets in this batch.

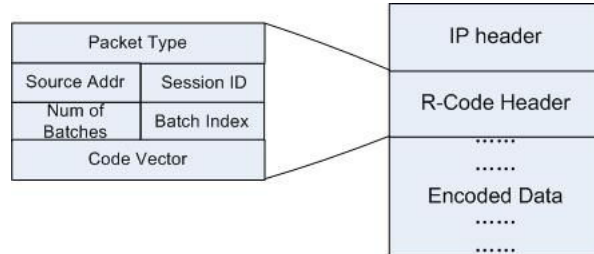


Figure 2.3: R-Code packet header format.

2.5.2.1 Basic Scheme

Generally, R-code can be divided into two stages.

(1) Initialization. During this stage, a distributed algorithm [30] is applied to construct a MST⁶. During the whole broadcast session, this MST is considered to be fixed. As stated in Section 2.3.2, the source divides the file into batches with size of k and broadcasts those batches in a round robin fashion to avoid crying baby problem.

For each batch, each node i initially selects its parent in the MST as guardian, denoted as G_i . As the broadcast session goes on, for some specific parent-child pair, their guardian-ward relationship maybe reversed⁷. However, we note that each node always has only one guardian at any time, either its parent or its child.

⁶In R-Code, the MST only need to be built up for once and can be shared by multiple broadcast sessions, this is different from Pacifier whose multicast tree structure needs to be constructed for every multicast session and reconstructed during the session when some receiver receives one batch successfully.

⁷Parent-child relationship is defined for MST which is fixed during the life time of the tree structure. However, guardian-ward relationship is defined according to the proceed of the broadcast session, which will be dynamically adjusted.

(2) Broadcast. During this stage, the source keeps sending coded packets generated from the current batch. When a node i receives a packet, it firstly checks whether this packet is innovative. If not, the packet is discarded; otherwise, i buffers this innovative packet and runs Gaussian elimination to check if it has gathered enough packets for decoding the whole batch. If not, it continues to keep silent and waits for more packets; else if i decodes the whole batch, it sends an ACK back to its guardian G_i by unicast. Then, if i is currently not in the process of disseminating some other previously decoded batch, it begins to play the role of guardian for its wards and keeps sending coded packets of this batch. Each guardian also works in a round-robin fashion to disseminate those successfully decoded batches received currently. After receiving ACKs from all the wards for a specific batch, the guardian will eliminate this batch from its buffer.

2.5.2.2 Dynamically Maintain the Guardian-Ward Relationship

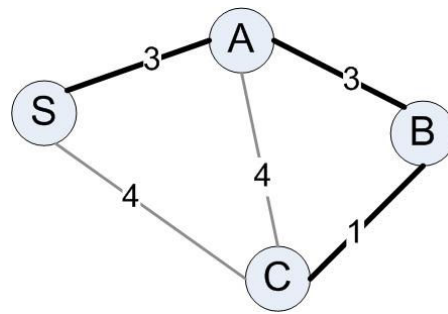
In the initialization stage, selecting the parent to be guardian for its children is based on the expectation that the parent is closer to the original source and thus will receive the whole batch ahead of the children. However, since a node i can overhear packets not only from parent, but also from ancestors and siblings due to the broadcast nature of wireless transmissions, it could happen that i gets the whole batch successfully before its parent. In this case, if w_{i,G_i} is less than $w_{G_i,G_{G_i}}$, which means that node G_i can be covered by i with less number of transmissions than be covered by G_{G_i} , the better choice for G_i is to take i as the new guardian. We adaptively adjust the guardian-ward relationship to capture this opportunistic overhearing gain. This can be done by sending two notification packets to i and G_{G_i} separately by unicast. Otherwise, if w_{i,G_i} is greater than $w_{G_i,G_{G_i}}$, the guardian-ward relationship between G_{G_i} and G_i keeps untouched.

A simple example is shown in Fig. 2.4 to illustrate this adjusting process. The network consists of 4 nodes and S is the source. The MST is indicated by bold links in Fig. 2.4(a). In the initialization stage, each node is assigned a guardian, G_A is S , G_B is A and G_C is B , as illustrated by arrows in Fig. 2.4(b). Suppose the batch size is k . If the guardian-ward relationship is fixed, both S and A need to transmit $3k$ packets to broadcast this batch, totally $6k$ number of transmissions. However, we notice that after A 's first k transmissions, C is expected to receive the whole batch successfully. Since $w_{C,B} = 1/3 \times w_{A,B}$, then B chooses A as its new guardian, as shown in Fig. 2.4(c). C needs to transmit another $2k$ times to finish the broadcast session. The total number of transmissions is $5k$, which is 17% less than keeping the guardian-ward relationship fixed. The extra cost for this change is only 2 control packets, from B to A and C separately. Since the common value for k is 32 or 64, it worths the effort to dynamically maintain the guardian-ward relationship.

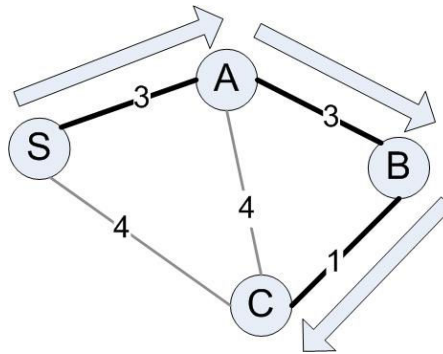
Moreover, this dynamic mechanism makes the initialization of guardian-ward relationship less critical. In specific, even the measuring of the link qualities are not very precise which leads to a sub-optimal MST being built at the initialization stage, R-Code can still keep high performance during the broadcasting process. In another word, the performance of R-Code is not sensitive to the accuracy of link quality measurement, in sharp contrast with Pacifier.

2.5.2.3 Source Rate Limiting

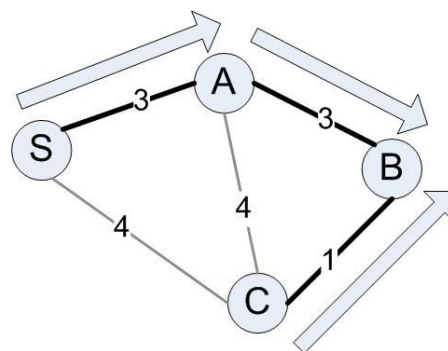
The importance of source rate limiting, through which the contention level of the network can be reduced, has already been shown in recent studies [43]. Pacifier applies a simple back pressure-based rate limiting by exploiting the broadcast nature of wireless transmissions. The basic idea is to let the source wait before sending the next packet until it overhears that the child has already forwarded the previous



(a) MST only



(b) MST with initial guardian-ward relationship



(c) MST with adjusted guardian-ward relationship

Figure 2.4: A simple example of dynamically maintaining the guardian-ward relationship

packet it sent. AdapCode's approach is to let the source wait for a given period of time $T_{batchInterval}$, after finishing the current batch and before starting the next batch. Moreover, each node has to backoff for a random period of time before transmitting each packet. This time is uniformly chosen between $10ms$ and $74ms$.

However, the situation in R-Code is different. Because now besides the original source, other guardians also generate packets actively rather than passively forwarding. Thus, R-Code can be considered as a multiple source broadcast scheme, which is different from flow-based Pacifier and AdapCode. Those backpressure-based approaches, which is derived from flow theory, is not directly applicable.

R-Code applies a simple rate limiting approach to all the guardians in the network. For a specific guardian, it has to backoff a random period of time before each transmission, the range of which should be proportional to the number of concurrent transmitters this node knows in the neighborhood. For example, in an area with large number of concurrent transmitters, the waiting time should be longer in order to reduce the probability of collision; otherwise each transmitter waits for shorter time for fast dissemination. We apply the moving weighted average (MWA) [76] method to determine the average number of transmitters: $avgTransmitter = \alpha \times avgTransmitter + (1 - \alpha) \times curTransmitter$, where the $curTransmitter$ is the number of concurrent transmitters that the node knows during the previous hello packet interval. The value of α should be determined according to the traffic stability of the network. Since most of the nodes in R-Code will be guardian for some batch during the whole broadcast session and the topology of WMN is stable, the number of transmitters among the neighbors of each node does not change often. Thus, we choose $\alpha = 0.8$ in our simulations. The wait period of time is uniformly chosen between $T_{pktInterval} \times 0.5 \times avgTransmitter$ and $T_{pktInterval} \times avgTransmitter$. $T_{pktInterval}$ is called packet broadcast interval, the value of which should be deter-

mined by the current traffic load in the network. We will evaluate the performance of various $T_{pktInterval}$ s in the following section. Note that this approach is different from what is adopted by AdapCode, where the chosen range of the random waiting time is identical to all the nodes and the local environment is not taken into consideration.

2.6 Performance Evaluation

We evaluate the performance of R-Code and compare it with AdapCode through extensive simulations. Since AdapCode is purposefully designed for wireless sensor networks, for fairness, we slightly modify it to be applied in WMNs. (1) In our implementation of AdapCode, we let the nodes transmit coded packets generated by linear combination of the whole batch rather than a portion of it. As claimed in [42], this could make AdapCode have better performance on bandwidth efficiency with the cost of higher computation overhead for encoding and decoding. (2) We let the responder of NACK send coded rather than original packets, which can benefit other nodes which overhear these packets, besides the sender of the NACK. (3) The range of the random backoff time is selected the same as R-Code does, which is more adaptive. (4) We also choose other parameters like batch size, the Galois field size, etc, to be the same as R-Code. Note that all those modifications make AdapCode performs better in WMN than its original version.

2.6.1 Simulation Settings

We use Glomosim simulator in our simulation. The network consists of a 7×7 grid of static nodes. We choose the grid size to be $200m$ and $250m$, where the former simulates a relatively dense network and the latter simulates a relatively sparse

Table 2.2: Optimal coding schemes

Average number of neighbors	0-5	5-8	8-11	11-
Best coding scheme	$N = 1$	$N = 2$	$N = 4$	$N = 8$

Table 2.3: Simulation parameters

Simulation parameter	Value
Batch size	32
Galois field size	2^8
$w_{threshold}$	5
$T_{batchInterval}$	100ms
Data transmission rate	11Mbps
ACK transmission rate	1Mbps
Retry limit	7
Pathloss model	two-ray
Fading mode	rician
Rician k factor	4
Hello packet interval(T)	1s

network. The average radio transmission range is 317m. Note that the WMN we considered in this chapter consists of only routers, which are usually deployed in a well-considered way for the purpose of balancing the network coverage and deployment cost, so a grid topology is more reasonable than a random topology for simulation. For AdapCode, We follow the optimal coding schemes presented in [42], which is shown in Table 2.2.

The source is fixed to be node 0 for all the simulations. The broadcasted file is 4MB, consisting of 1KB packets. Other related simulation parameters are listed in

Table 2.3. We run both protocols in the two grid sizes 10 times and show the average results over all 10 runs. For each run, the $T_{pktInterval}$ ranges from $2ms$ to $9ms$, with the step equals to $0.5ms$. We use the following metrics for comparison:

Average broadcast latency: the total time required for a node to receive the whole file, averaged over all nodes.

Average number of transmissions: the total number of transmissions of all the nodes divided by the number of nodes.

Average number of collisions: the total number of collisions experienced by all the nodes divided by the number of nodes. Note that one transmission can cause several collisions at different nodes.

Average number of linearly dependent packets: the total number of linearly dependent packets received by all the nodes divided by the number of nodes. Note that the count of this metric includes the case that a node who has already received the whole batch overhears coded packet from the same batch.

We do not compare PDR performance, since both R-Code and AdapCode can guarantee 100% reliability.

2.6.2 Number of Transmissions

We first compare the average number of transmissions introduced by both protocols. Besides data packets, we also count the NACKS of AdapCode, and ACKs and those control packets for maintaining the guardian-ward relationships of R-Code.

From Fig. 2.5(a) and Fig. 2.6(a) we can see that whatever the grid size is, R-Code uses less number of transmissions than AdapCode does for accomplishing the

broadcast session, the maximum reduction can be 13% when the grid size is $200m$ and 15% when the grid size is $250m$. The key reason for R-Code's better performance is its local optimal decision. For each node, it always chooses the best neighbor to be guardian. In comparison, when a node i in AdapCode sends a NACK, it randomly chooses a node from those that overhears this NACK and possesses the required packets to be responder. This randomly selected node, as we argued in section 2.4.2, maybe not the best one and thus incurs more redundant transmissions. This is shown clearly in Fig. 2.5(b) and Fig. 2.6(b), where we can observe that AdapCode yields more linearly independent receptions in most cases.

The only exception appears when $T_{pktInterval}$ is small, e.g., less than $3.5ms$ when the grid size is $200m$, and $2.5ms$ when the grid size is $250m$, where R-Code uses more transmissions. The reason for this exception is that in these cases, all the nodes inject packets too fast to be sustained by the network. This causes heavy contention between nodes, which is shown clearly in Fig. 2.6(d), where we can see that the average number of collisions experienced by each node is almost 700 when $T_{pktInterval}$ is $2.0ms$ and grid size is $200m$. The heavy collision further leads to the large amount of overheard linearly dependent packets, in other words, useless for the receiving nodes. This is shown in Fig. 2.6(b). Since the average number of neighbors for each node is approximately 6 for the network with grid size of $200m$, when $T_{pktInterval}$ is $3.0ms$, all the nodes' broadcasting rate has already above $400Kbps$, which is quite a high speed. Note that the reason for AdapCode's better performance in these cases is the NACK+Timer mechanism, which makes AdapCode not sensitive to the variation of $T_{pktInterval}$. However, this comes with the cost of much longer broadcast delay, which is shown in Fig. 2.5(c) and Fig. 2.6(c).

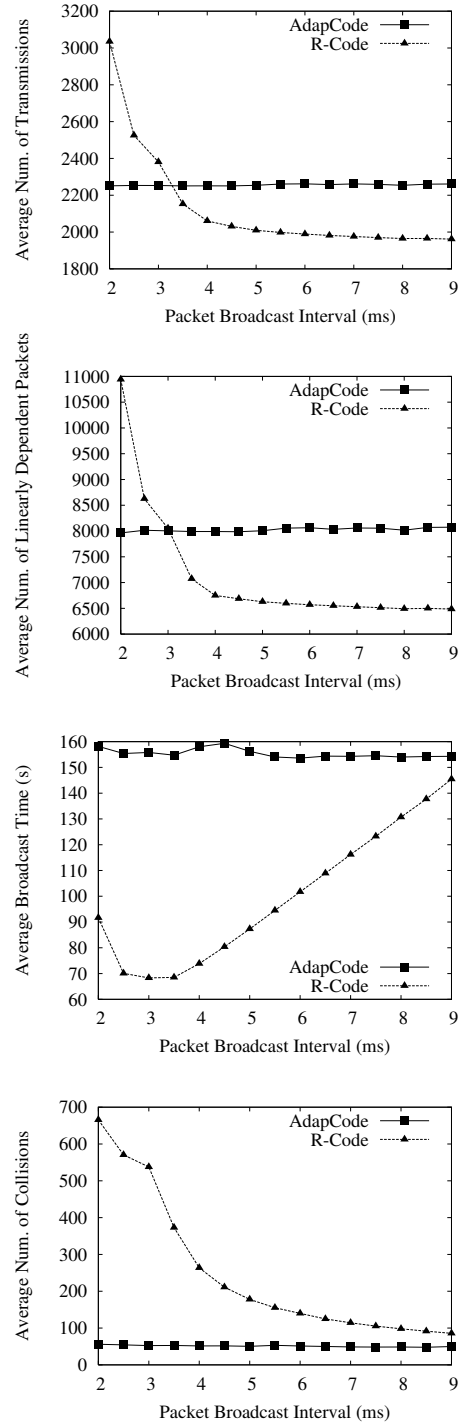


Figure 2.5: Grid size=200m

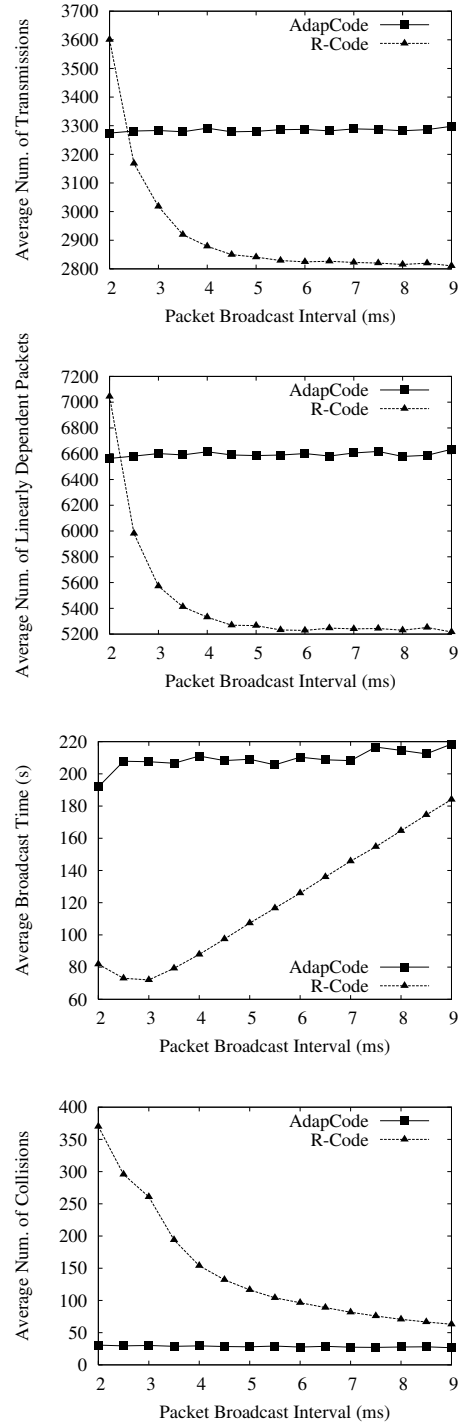


Figure 2.6: Grid size=250m

2.6.3 Broadcast Latency

Compared with the performance of number of transmissions, R-Code gains larger advantage over AdapCode when considering the broadcast latency. We can see that under all settings, the average broadcast latency of R-Code performance better than AdapCode. When $T_{pktInterval}$ is small, the reduction ratio can be up to 65%, which is achieved when the grid size is 250m and $T_{pktInterval}$ is 3.0ms. This is consistent with our analysis in Section 2.5.1, where we point out that NACK mechanism inherently tends to elongate the broadcast latency. We also observe that the broadcast latency of R-Code grows almost linearly to $T_{pktInterval}$. This is because that before trying to inject packets into the network, each guardian has to delay for a random short period of time that is proportional to $T_{pktInterval}$. We also observe that this linear property does not hold when $T_{pktInterval}$ is small, e.g., less than 3ms when grid size is 200m. In these cases, the broadcast latency of R-Code is even higher than the case when the packet broadcast latency is 3ms, which is also the minimum broadcast latency for all the cases. This seemingly abnormal performance can also be explained by the high congestion level in these cases. Under such high congestion level, each transmitter will encounter many collisions that causes it to backoff longer time, according to the CSMA/CA mechanism, which offsets the benefits of shorter $T_{pktInterval}$ s. In contrast, AdapCode's performance is quite stable under various $T_{pktInterval}$ s. The reason is that the broadcast latency of AdapCode is mainly decided by the initial value of the NACK timer for each node, which is set to be $2 \times T_{batchInterval}$ and not related to $T_{pktInterval}$.

Although the performance of R-Code on both number of transmissions and broadcast latency is better than AdapCode, we still can see that there is a tradeoff between transmission overhead and broadcast latency. The network designer need to choose proper values for parameters according to specific applications.

2.6.4 Discussion

We compare the average number of collisions between R-Code and AdapCode and observe that as the increasing of $T_{pktInterval}$, the average number of collisions experienced by AdapCode are quite stable. In a comparison, the average number of collisions experienced by R-Code decreased rapidly at first and then approaches to a stable value, which is a little higher than AdapCode. The reason for higher number of collisions experienced by R-Code in all settings is “hidden terminal” problem. Since R-Code encourages simultaneous transmissions to enhance the performance of spatial reuse, so it tends to suffer from the hidden terminal problem more. On the opposite, the NACK mechanism of AdapCode makes each node’s transmission more passive and provides less chance for the happening of hidden terminal problem. However, We argue that this does not means AdapCode will introduce less number of transmissions, because its inefficiency in bandwidth usage comes from the random selection mechanism for NACK’s responder, which incurs large amount of linearly dependent packets. How to deal with hidden terminal problem in R-Code will be our future work.

2.7 Summary

In this chapter, we propose R-Code, a distributed and efficient broadcast protocol which guarantees 100% PDR for all receivers. By introducing a guardian-ward relationship between neighboring nodes, R-Code effectively distributes the global responsibility of reliable information delivery from the original source to those locally selected guardians. R-Code uses a link quality-based MST as a backbone to guide the selection of guardians adaptively and the transmission of coded packets accordingly. R-Code also prevent guardians from sending duplicated packets with no extra

overhead by adopting network coding technique. Extensive simulations show that R-Code reduces the average number of transmissions and broadcast latency up to 15% and 65%, respectively, compared with AdapCode, a state-of-the-art reliable broadcast protocol under unreliable links.

Chapter 3

Cooperative Popular Content Distribution in VANETs using SLNC

3.1 Introduction

Vehicular communications have attracted lots of attentions recently. Since the advent of dedicated short range communications (DSRC) [1,48], and IEEE 802.11p and IEEE 1609 standards [4], people have envisioned and designed numerous tempting applications of vehicular networks, ranging from safety warning [64], intelligent navigation to mobile infotainment [63]. A particularly promising type of application is related to both safety-related and commercial services. That is, the distribution of “popular” multimedia contents to vehicles inside a geographical *area of interest* (AoI) by road side infrastructure (e.g. access points (APs)), which is referred to as *popular content distribution* (PCD) in this chapter. Examples of PCD may include: an ads

company periodically broadcasts multimedia advertisements of local businesses in a city to vehicles driving through a segment of suburban highway passing by that city (like a digital billboard); a traffic authority delivers real-time traffic and accident information about the roads in an urban area for intelligent navigation or emergency warning purposes, or disseminates an accurate update of the GPS map about a city or a scenic area.

Different from the usual “content downloading” services where various vehicles are interested in downloading different files from the Internet [26,81], the popular contents in PCD are often commonly “interested” by most of the vehicles driving through an AoI, and sometimes may even be disseminated mandatorily such as emergency videos [87]. An important aspect in common about popular contents is their potentially large file sizes, because multimedia files including video and audio are more vivid and effective, thus are always preferred over text-only files. For example, an advertisement video may be as large as 100 MB. Indeed, disseminating such large contents is possible in vehicular networks, given that four sub-channels in DSRC are allocated as service channels, while the IEEE 802.11p supports data rates up to 27 Mbps.

The primary requirement of PCD in vehicular networks is to achieve short *downloading delay*, or equivalently, high *downloading rate*. The former is the average time required for end-vehicles to receive a file completely. From a driver’s point of view, fast reception of a video about an accident or traffic condition may help the driver to plan his/her route in advance to avoid possible traffic jams or accidents. From the content provider’s viewpoint, shorter downloading delay improves the ratio of vehicles that can receive the content. Thus, a short delay is essential for both commercial and non-commercial contents. In addition, it is also critical for PCD to maintain a high degree of efficiency, i.e., to introduce low protocol overhead and reasonable amount of data traffic, so that PCD is readily compatible with other potential services running

under the same channel.

Due to the relatively high cost of deploying APs, the access to wireless Internet is quite limited in vehicular networks. In the initial deployment phase APs may be rare, which could be placed in highway service areas, gas stations or road intersections. Since it takes usually less than 1 minute for moving vehicles to drive through the coverage of an AP, vehicles may not finish downloading a large file within such a short time period. When vehicles are out of the coverage of the APs, they form a vehicular ad hoc network and cooperative distribution of the popular content is thus necessary.

However, it is non-trivial to design a high-rate and efficient cooperative PCD scheme. The main challenges come from the lossy wireless medium under vehicular environments, and the highly mobile and dynamical nature of VANETs. First, the lossy wireless links cause frequent packet losses and collisions, leading to prolonged downloading delay and decreased efficiency, and negatively affects the protocol performance. In addition, the ever-changing VANET topology prevents real-time acquisition of precise neighbor information (such as reception status) which forms the basis of optimized, distributed transmission decision making. If there lacks a well-devised coordination mechanism among the transmitting vehicles, duplicate transmissions may fill up the channel and waste the precious VANET bandwidth. Also, a PCD scheme could potentially incur large protocol overhead spent in collecting those information needed to achieve high performance.

Towards solving these problems, many existing works [7,50,62,63,87] have adopted NC for content downloading in VANETs, because NC effectively reduces duplicate transmissions and simplifies the transmission scheduling. Most of these protocols employ a pull-based cooperative content downloading approach [62,63], where vehicles transmit passively upon others' downloading requests, which suffers from low effi-

ciency. When downloading popular files, many vehicles make requests for the same content and many vehicles respond to their requests. Due to the lack of coordination, these protocols cannot avoid severe packet losses and collisions, especially under a dense VANET. This could lead to extremely low efficiency and large downloading delay. Thus the performance gain obtained from network coding is under-exploited and even offset by unrefined protocol design.

In this chapter, we put forward CodeOn, a high-rate cooperative PCD scheme for vehicular networks. We explore SLNC technique for cooperative PCD. In contrast with traditional packet level network coding, SLNC performs network coding on finer granularity of physical layer symbols. Since the error rate of a symbol is smaller than that of a packet's, SLNC has better error tolerance, enhances reception reliability and thus the downloading rate. Fully exploiting the advantage of SLNC for PCD necessitates non-trivial protocol design, whereas we make the following main contributions.

(1) CodeOn provides a whole new set of push-based content distribution protocol design for VANETs. The popular contents are actively broadcasted from a few APs to all vehicles within an AoI, through the cooperation of a set of dynamically selected relay nodes. In order to maximize the usefulness of every piece of content broadcasted by those relays, we propose a prioritized relay selection mechanism to coordinate the transmissions of vehicles, in which every vehicle's transmission priority is proportional to how much additional useful content it can provide to its neighbors. In addition, we use a simple medium access control (MAC) mechanism based on carrier sensing, which fully exploits the increased transmission concurrency enabled by SLNC so as to maximize the downloading rate.

(2) To reduce the protocol overhead without degrading the performance, we propose a scalable and efficient average-rank method for vehicles to represent and ex-

change their content reception status under SLNC. By taking advantage of the multi-channel property of VANET, vehicles piggyback this tiny information in their safety messages sent in control channel, which incurs *zero* overhead for content downloading.

(3) We implement CodeOn in NS-2 and evaluate its performance by extensive simulations. We compare CodeOn with an enhanced version of CodeTorrent, which is a pull-based, network coding based content distribution protocol and represents the current state-of-the-art. Simulation results show that CodeOn performs significantly better than CodeTorrent, in terms of average downloading delay, protocol efficiency and fairness. Significant improvements in average downloading rate are obtained for both highway and urban scenarios. To the best of our knowledge, this is the first time that cooperative PCD has been studied under lossy VANET environments.

The rest of the chapter is organized as follows. Sec. II formulates the PCD problem in vehicular networks and discusses related works, Sec. III introduces symbol level network coding and its benefits for content downloading. The main design of CodeOn is presented in Sec. IV. Sec. V contains the performance evaluation and results. Finally, Sec. VI summarizes this chapter.

3.2 Problem Formulation and Related Work

3.2.1 Problem Formulation

3.2.1.1 Model and assumptions

In this chapter, we consider the following PCD service architecture for vehicular networks. The content provider (e.g. a city wide traffic administration bureau) wants to distribute some popular files to all vehicles inside the AoI, which can be either

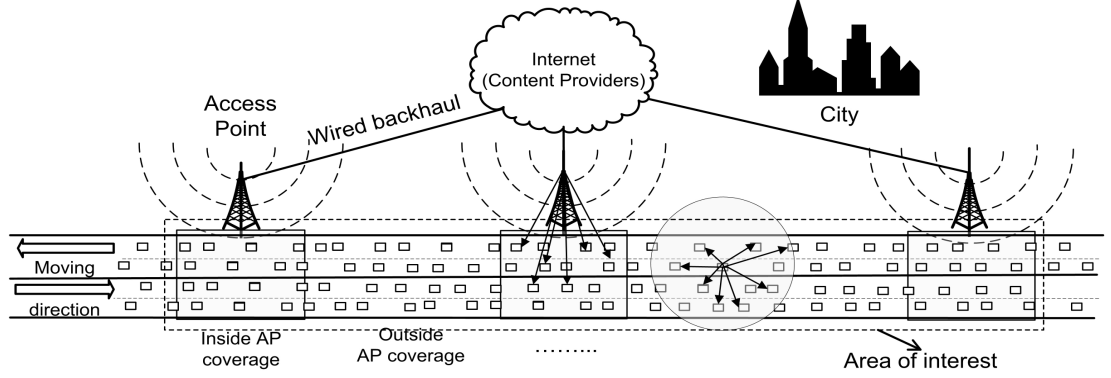


Figure 3.1: The architecture for PCD. Inside the AP coverage, AP broadcasts and vehicles receive; outside the AP coverage, vehicles distribute their received contents cooperatively.

a highway segment or an urban area. There are multiple APs (or road side units) deployed in an AoI, and APs are connected together through a wired backhaul. APs are controlled by the service provider to actively disseminate popular contents to the vehicles within the AoI. APs can be placed either deterministically or randomly and optimal placement is outside the scope of this chapter. The service architecture is illustrated in Fig. 3.1.

Each vehicle is equipped with an on board unit including a wireless transceiver (single radio). The wireless interface operates on multiple channels [1, 48]. To model the coexistence of safety and commercial applications, we consider two representative channels. The control channel is used to broadcast safety messages, which may contain vehicles' locations, speeds etc.; one service channel is dedicated for PCD. In order to guarantee the quality of service of safety messages (the interval between two consecutive safety messages should be smaller than 100ms [77]), time is divided into periodical, 100ms slots and all vehicles and APs are synchronized to switch simultaneously between the control channel and service channel. The utilization of time and channels is depicted in Fig. 3.2. Although there are advanced MAC protocols that dynamically adjust the time shares of control channel and service channel for better

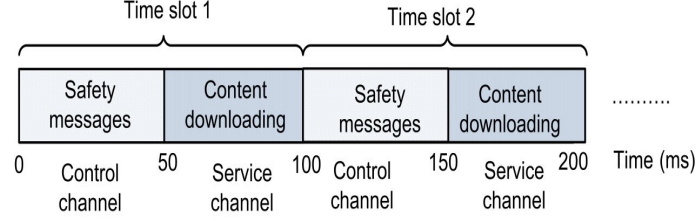


Figure 3.2: The time and channel utilization of each vehicle and each AP.

service [77], we fix it to $1/2 : 1/2$ for simplicity.

In the control channel, each AP and each vehicle broadcasts one beacon message in each slot. When a vehicle is in the range of an AP, it merely listens to the AP's content broadcast in the service channel; otherwise, it may share its received content with neighboring vehicles cooperatively. Vehicles outside the AoI do not involve in content distribution.

In addition, we assume all vehicles are equipped with Global Positioning System (GPS) devices, from which vehicles obtain their real-time locations and synchronize their clocks (error smaller than 100ns). GPS devices are low-cost and are available to most of the drivers nowadays. When vehicles are temporarily out of satellite coverage, they can use auxiliary techniques to determine their location, and rely on their own hardware clocks. Note that, GPS time synchronization is required by the IEEE 1609.4 standard for multi-channel operations [4].

3.2.1.2 Objectives

For any content distributed by the PCD service, the primary objective is to achieve low average downloading delay, which is equivalent to high average downloading rate. For each vehicle in an AoI, its *downloading delay* is defined as the elapsed time from downloading start to 100% completion. Meanwhile, it is desirable to achieve a high

degree of fairness, i.e., the variation of downloading delays among different vehicles should be small. Finally, high-rate content distribution cannot come at the cost of incurring too much protocol overhead and data traffic, otherwise the PCD service would be less compatible with other possible services in the service channel. Thus it is also important to maintain high protocol efficiency.

3.2.2 Related work and our contributions

In [81], Nandan *et.al.* first studied cooperative downloading in VANETs. They proposed SPAWN, a pull-based, peer-to-peer content downloading protocol for VANETs that extends BitTorrent. Later, they proposed “AdTorrent” [81], which is a semi push-based peer-to-peer protocol for vehicles to download advertisements they are interested in. In both SPAWN and AdTorrent, the peer and content selection mechanisms have high overhead and are not scalable, especially when most of the vehicles are interested in downloading popular contents. Also, they suffer from the “coupon collector problem” which enlarges downloading delay. Moreover, they use TCP for content delivery, which performs poorly over multi-hop lossy wireless links in highly mobile VANETs.

3.2.2.1 Network coding for content downloading

To avoid such problems, many researchers resort to network coding (NC) [6, 38]. NC mixes the packets by coding them together at every intermediate node and exploits the broadcast nature of wireless medium, so that the usefulness of each coded packet is increased. Lee *et.al.* proposed CodeTorrent [63], a pull-based content distribution scheme using NC, where vehicles need to explicitly initiate requests to download a piece of content. CodeTorrent restricts the peer selection and content delivery to the

one-hop neighborhood of a vehicle, thus eliminating the need of multi-hop routing. Also, the use of NC mitigates the peer and content selection problems.

Later, Lee *et.al.* further studied the practical effects of content distribution in VANETs using NC [62] based on a variation of CodeTorrent. It is shown that the resource constraints such as disk access, computation and buffer have significant impacts on the performance. They discussed approaches to reduce the communication and computation overhead of NC while maintaining the gain of it. Since this chapter focuses on dealing with the lossy wireless links in content downloading for VANETs, our work is orthogonal to [62].

The above schemes are all pull-based in essence. They could suffer from large downloading delay, since nodes passively respond to their neighbors' requests and the bandwidth is wasted (i.e., being idle much of the time). For example, in CodeTorrent it takes 200 seconds to download a 1 MB file in an urban scenario [63]. If a node wants to receive new information continuously, it must send out requests frequently. The transmissions from multiple responders tend to collide with each other, leading to low-efficiency in turn. Park *et. al.* proposed a push-based content delivery scheme for emergency related video streaming using NC [87]. However their "push" protocol design essentially reduces to controlled flooding, which tends to be inefficient.

In fact, with PLNC, it is difficult to achieve high downloading performance especially under lossy wireless links in VANETs, whether or not push based protocol design is adopted. The wireless medium in VANET has been shown to be lossy by empirical analysis [91, 100, 103]. In practice, network coding for a large file is usually done within each block of the file, namely a *generation* [7, 62, 63]. In order to maintain reasonable coding/decoding complexity while reducing the protocol overhead, the basic coding unit (coded piece) shall be larger than a usual packet. During the transmission of such a coded piece, any error to the coding vector or message body

will render the whole piece useless, leading to degraded downloading performance.

In this chapter, we put forward CodeOn, a whole new set of push-based protocol design that can well solve those problems. Instead of using PLNC, we take advantage of SLNC which has much better resiliency to transmission errors due to symbol-level diversity.

3.2.2.2 Transmission coordination in content downloading

Transmission coordination is an important issue for content distribution in VANETs. Bad coordination could result in severe packet collisions that affects the downloading performance. However, this issue has not been well addressed in previous works. In [87], a simple time out mechanism is used for each vehicle to decide when to transmit a coded packet. However, this mechanism does not take into account vehicles' content reception status, which leads to a non-negligible chance of duplicate information. Also, packet collisions are severe when the network is dense.

In [126], Zhang *et. al.* studied this problem from link layer, and proposed VC-MAC, a cooperative medium access control (MAC) protocol for gateway downloading scenarios in vehicular networks. In order to avoid possible interference among multiple transmissions, and to maximize the “broadcast throughput”, a heuristic relay selection algorithm with a backoff mechanism is proposed. However, the “broadcast throughput” is purely based on link quality, which is not content-aware. The relay chosen by VC-MAC may have nothing innovative to transmit to its neighbors.

In CodeOn of this chapter, we explicitly consider the content usefulness of nodes for higher rate content downloading. A dynamic set of relay nodes, which are selected based on their content availability and usefulness, actively broadcast (push) useful contents to neighboring nodes, and make medium access decisions based on both their

content usefulness and local channel status.

3.2.2.3 Multi-channel compatibility

Few existing work considered the compatibility of content downloading with other channels. In [77], the authors propose mechanisms to adjust the time share of the service channel to enhance the performance of content downloading while guaranteeing the QoS of safety messages. This chapter considers the coexistence of a service channel with the control channel, with the difference that we design a better PCD protocol given a fixed time share of service channel. Also we novelly utilize the control channel for better content downloading.

3.2.2.4 Other related works

In [130], Zhao *et. al.* proposed data pouring, a push-based data dissemination protocol for VANETs. They focus on broadcasting small data items to all vehicles inside an area, while we aim at disseminating large popular files. In [129], Zhao *et. al.* also studied the problem of drive-thru access to roadside APs, and proposed a vehicle-to-vehicle relay strategy to extend the coverage of APs. In [119], Yang *et. al.* proposed a push-based, reliable broadcast protocol for wireless mesh networks using network coding.

In addition, Fiore *et. al.* focused on cooperative downloading in urban VANETs [26]. The Roadcast [128] is a popularity-aware content sharing protocol in VANETs. These protocols are mainly suitable for applications where each vehicle may be interested in downloading different files, while we consider the popular content distribution.

3.3 Symbol-level Network Coding

In this section, we first describe the symbol-level network coding technique. Then, we give a motivating example to show the potential advantage of exploiting symbol-level diversity in content distribution in VANETs.

3.3.1 A Brief Review of Symbol-level Network Coding

SLNC arises from the observation that in wireless networks, even if a packet is received erroneously, some small groups of bits (“symbols”) within that packet are likely to be received correctly. SLNC gathers these correctly received (i.e., “*clean*”) symbols aggressively, and performs network coding on the granularity of symbols. In contrast to PLNC, SLNC gains from both symbol-level diversity and network coding. In addition, since more bit errors are tolerated than PLNC, SLNC can also gain higher throughput by encouraging more aggressive concurrent transmissions.

In general, SLNC works as follows. A symbol is defined as a group of consecutive bits in a packet, which may correspond to multiple PHY symbols of a modulation scheme. Assume the source has K packets to send, each of them expressed as a vector with elements from a Galois field \mathbb{F}_{2^q} . The j th symbol \mathbf{s}'_j in a coded packet at the source is a random linear combination of the j th symbol in all K source packets:

$$\mathbf{s}'_j = \sum_{i=1}^K v_i \mathbf{s}_{ji}. \quad (3.1)$$

where \mathbf{s}_{ji} is the j th symbol (at j th position) in the i th original packet, coefficient v_i is randomly chosen from \mathbb{F}_{2^q} , and $\mathbf{v} = (v_1, \dots, v_K)$ is the coding vector of the coded packet, which is also the coding vector for each symbol. Each receiver node v maintains a decoding matrix for every symbol position. A newly received coded symbol for position j is called *innovative* to v , if that symbol increases the *rank* of

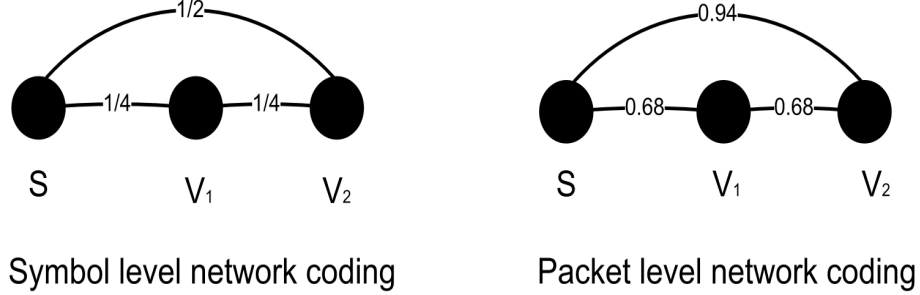


Figure 3.3: The topology for the example in Fig. 3.4. Left: numbers on the edges (links) show the symbol error probabilities; right: corresponding packet error probabilities.

the decoding matrix of the j th symbol position, referred to as *symbol rank*. Only innovative clean symbols are buffered.

Each coded packet transmitted by a relay node consists of random linear combinations of buffered clean symbols. For a source, every symbol in a packet is clean and shares the same coding vector. However, at a relay node, coding vectors may be different across symbols. For a coded packet to be sent by relay u , the j th coded symbol is expressed as

$$\mathbf{s}_j'' = \sum_{i=1}^R v_i' \mathbf{s}_{ji}' = \sum_{i=1}^R (v_i' \sum_{l=1}^K v_{li} \mathbf{s}_{jl}) = \sum_{l=1}^K (\sum_{i=1}^R v_i' v_{li}) \mathbf{s}_{jl}, \quad (3.2)$$

where R is the number of buffered clean symbols at position j , \mathbf{s}_{ji}' is the i th buffered clean symbol (row) at position j (column), and $\mathbf{v}_i = \{v_{1i}, \dots, v_{Ki}\}$ is the coding vector for that symbol. \mathbf{s}_{jl} is the j th symbol of the l th source packet. From Eq. (3.2), \mathbf{s}_j'' is still a random linear combination of source symbols, and its new coding vectors are $\mathbf{v}' = (\sum_{i=1}^R v_i' v_{1i}, \dots, \sum_{i=1}^R v_i' v_{Ki})$.

In the extreme case, every symbol's coding vector is different and needs to be sent along with a packet, which incurs high overhead. To minimize this overhead, optimized run-length coding method can be adopted [54], where consecutive clean symbols are combined into a “run”.

3.3.2 How VANET content distribution benefits from SLNC

To illustrate how SLNC works and see the potential performance gain of SLNC over PLNC, we give a 3-node simple example for content distribution in VANET (Fig. 3.3 and Fig. 3.4). The corresponding topology is shown in Fig. 3.3. Assume source S has two original packets X and Y to broadcast. Assume a simple scheduling: S broadcasts coded packets until V_1 can decode the original packets, and then V_1 broadcasts until V_2 decodes all original packets.

Suppose S generates and broadcasts three coded packets A , B and C , each of them divided into 4 symbols. Let the symbol error probability from S to V_1 be $P_{se}(S, V_1) = \frac{1}{4}$, and it happens that each packet received by V_1 contains an erroneous symbol (Fig. 3.4). Luckily, for each symbol position at least two clean symbols are received. Since any two coding vectors among $\mathbf{v}, \mathbf{v}', \mathbf{v}''$ of A , B and C are independent¹, V_1 can decode X and Y by solving 4 linear equations. When V_1 broadcasts two packets (say, D and E), it generates two new coded symbols at each position, and packs the 8 coded symbols into D and E . Each new coded symbol is also a random linear combination of original symbols. Thus, V_2 can recover all original symbols after collecting 2 innovative coded symbols at each position, which may come from both S and V_1 .

The key insight of SLNC is that, for each symbol position, every correctly received coded symbol is equally useful for decoding, or it does not matter which symbol is received. While for PLNC, the reception granularity is a whole packet. Since the symbol error rate will be much less than the packet error rate, it is not hard to imagine SLNC will take less transmissions to collect the information needed for decoding the same amount of content.

¹This happens with high probability when the size of \mathbb{F}_{2^q} is large.

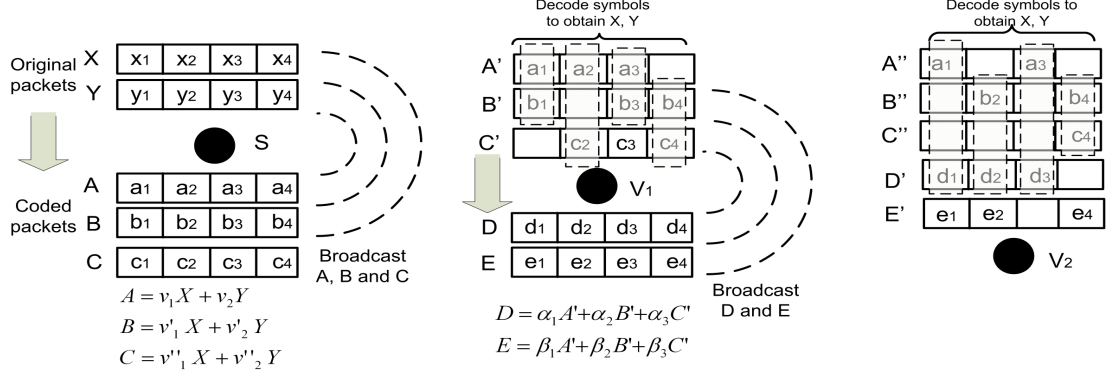


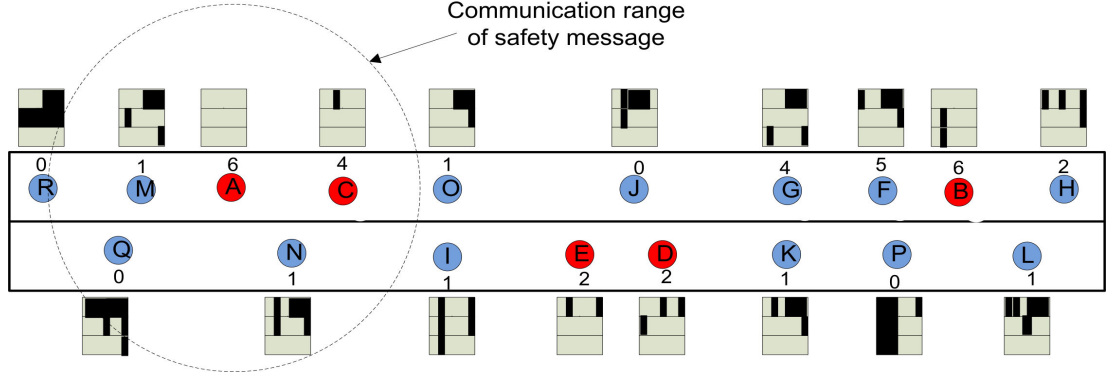
Figure 3.4: Symbol level network coding in VANET content distribution. S : source node; V_1 and V_2 : downloading vehicles & relays.

To confirm the above intuition, we compute the expected number of packets ($\mathbb{E}[Z]$) transmitted by S for node V_1 to decode using the simple example. After some calculation (refer to [65]), we obtain

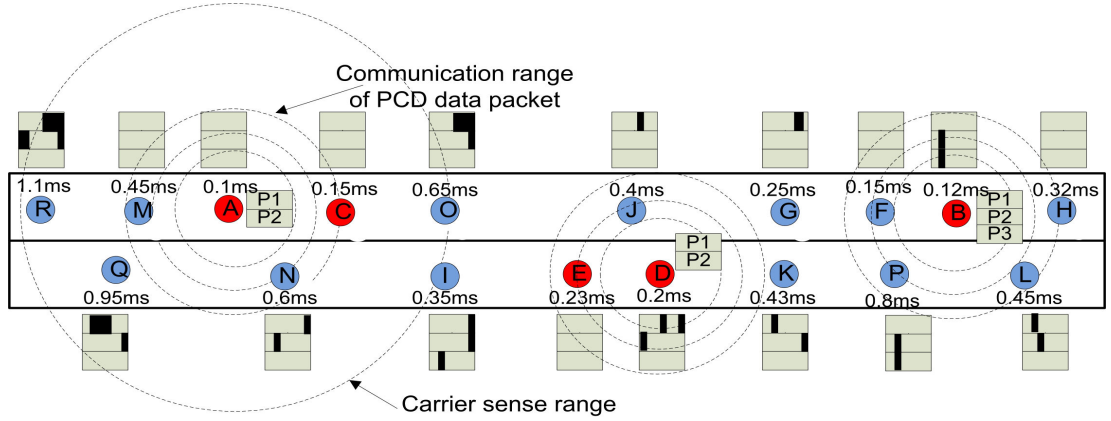
$$\mathbb{E}[Z] = \sum_{k=0}^{\infty} P(Z > k) = \sum_{k=0}^{\infty} [1 - P(Z \leq k)]. \quad (3.3)$$

Plugging in the parameters in the example, we obtain $\mathbb{E}[Z] = 3.67$. That is, 3.67 coded packets should be sent by S on average for V_1 to decode X and Y .

Next we compare SLNC to using PLNC for the same case. We compute the expected number of packets $\mathbb{E}[Z']$ sent by S for V_1 to receive 2 source packets. Assuming independent packet reception, we obtain that S must transmit 6.26 packets on average for V_1 to decode [65]. Thus, the number of transmissions (proportional to downloading delay) of V_1 has been reduced by $\frac{6.26-3.67}{6.26} = 41\%$ due to the use of SLNC. Similar conclusions can be drawn for node V_2 . From the above, the advantage of using SLNC than PLNC is evident for content distribution in VANET, i.e., it leads to higher downloading rate and incurs fewer transmissions.



(a) Exchange of neighbor information and utility calculation based on nodes' reception status. All nodes' reception status are depicted. Black parts in a piece indicate corrupted symbols in a node's buffer.



(b) Transmission coordination among potential relays, based on both node priority and carrier sense. Backoff delays are inversely related to nodes' utilities (nodes A-F have the least delays, but only A, B, D become relays).

Figure 3.5: Overview of cooperative content distribution in CodeOn.

3.4 The Design of CodeOn

We first give the main notations used in this chapter in Table. 3.1.

Table 3.1: Frequently used notations

Notation	Definition
F	The file to be distributed
N	Data packet size (bytes)
L	File length (number of generations)
K	Generation size (number of pieces)
J	Piece size (bytes)
M	Number of symbols in a packet
G_i	Generation i
\mathbb{F}_{2^q}	The Galois field used in network coding
$U(v)$	The utility of a node v
$\mathcal{N}(u)$	The neighbor set of node u
$\bar{r}_{v,i}$	Average symbol rank of G_i in vehicle v
γ	Average received SNR or SINR for a symbol

3.4.1 Overview

CodeOn is a push-based cooperative content distribution protocol, where a large file F is actively distributed from the APs to the vehicles inside the AoI through the help of a dynamic set of relay nodes. Each AP is a source for F , and F is divided into equal-sized generations (chunks), and the SLNC is performed within each generation. In Fig. 3.5, we illustrate the general process of content distribution in CodeOn, assuming F has only one generation consisting of 3 pieces.

Each AP/source broadcasts the source file to vehicles in its range based on vehicles' reception status, which is not shown in Fig. 3.5. Outside the ranges of APs, vehicles distribute the file cooperatively by agreeing on a set of relay nodes. This is the core

to CodeOn, which consists of three steps.

(1) Exchange of neighbor information. This is done in each control time slot, where every vehicle broadcasts a safety message that piggybacks a sketch of its content reception status, which will be used as an implicit content request for step (2). In this way, zero overhead is incurred in the service time slots. To limit the impact of piggyback overhead on control time slots, we will introduce a fuzzy representation of nodes' reception status later.

(2) Node utility calculation. This is the first step of distributed relay selection. In the beginning of each service time slot, every node computes its own utility based on neighbors' reception status information collected from step (1). The utility reflects each node's priority in relay selection, i.e., the total amount of useful content that this node can provide to all of its neighbors. Under such a priority assignment, the usefulness of each relay's transmission will be maximized, which enhances both the downloading rate and protocol efficiency. The utility of every node is shown in Fig. 3.5 (a).

(3) Transmission coordination among potential relays. As the last step of relay selection, we need to determine which nodes should actually access the channel, based on both node priority and the channel status. Each node computes a backoff delay that is inversely related to its utility, and upon the expiration of the delay it will sense the channel. If it cannot detect signal energy, it will broadcast coded contents without delay. Otherwise, it remains silent throughout the time slot. This process is captured by Fig. 3.5 (b). Thanks to SLNC's better error tolerance, this aggressive way of channel access, although simple, will be shown to achieve close to maximum overall downloading rate in the following.

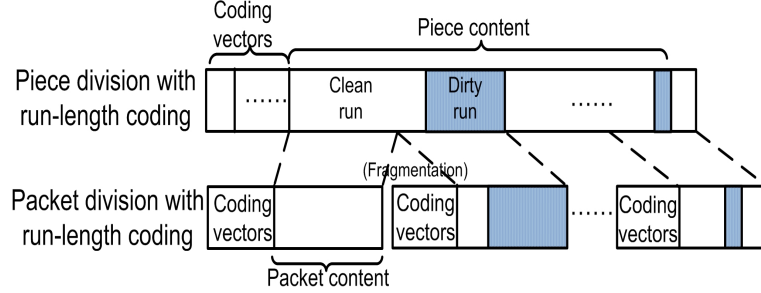


Figure 3.6: Comparison between the overhead of piece division and packet division, when both uses run-length SLNC.

3.4.2 Network Coding Method

In this section, we describe the way that SLNC actually operates in CodeOn. Assume F with size $|F|$ is divided into L generations G_1, G_2, \dots, G_L , where each generation contains K pieces. A piece has size J and contains $\lceil J/N \rceil$ packets. Then, $|F| = L \cdot K \cdot J$. In order to reduce the overhead brought by SLNC, we adopt “*piece-division, run-length SLNC*”.

The reasons are two fold. On the one hand, if a generation is divided into packets (packet-division), in order to keep small computational overhead we must use relatively small K (the computation complexity of decoding is usually $O(K^3)$), thus a large number of generations is required for large F . This reduces the gain of NC due to the “coupon collector’s problem” [62], and increases the communication overhead for exchanging the content availability. On the other hand, using multi-packet pieces (piece-division), K can be maintained at a reasonable value by scaling the piece length linearly with file size. However, the number of symbols in a piece ($\frac{J \cdot M}{N}$) increases with the piece length. In the extreme case if every symbol in a piece has a different coding vector, the communication overhead is at least $\frac{J \cdot M \cdot K \cdot q}{N}$ bits, which equals to 10KB if $J = 20\text{KB}$, $N = 1\text{KB}$, $K = 32$, $M = 32$, $q = 8$. This is clearly unacceptable. Fortunately, run-length coding method [54] can be used to reduce the communica-

tion overhead of SLNC, in which one coding vector is used for each sequence of consecutive clean symbols (*run*). Dynamic programming is used to choose appropriate combination of runs to minimize the overhead [54]. Therefore, in CodeOn, we combine run-length SLNC with the piece division to achieve higher network coding gain and reduce the communication overhead, which we call *piece-division run-length SLNC*. When a coded piece is transmitted, it is separated into several packets; only the header of the first packet contains the coding vectors of runs that composing the piece, while subsequent packets only have normal small headers. Thus, a piece can be regarded as a “big packet”.

Compared with PLNC, the gain from symbol-level diversity can be easily seen from the analysis in Sec. III. Meanwhile, the overhead of our method is always smaller than run-length SLNC combined with packet division. Generally, the number of coding vectors in a piece equals to the number of runs. However, using packet division a run may be fragmented into more than one runs, which needs more coding vectors in total. In the worst case, each symbol is a run and the overheads are equal. This is illustrated in Fig. 3.6. In reality, since the symbols errors are often bursty (due to packet collisions), the number of runs is usually much smaller compared with the number of symbols. For example, if there are 20 runs in a 20KB piece the overhead is about 640B, which is 3.2% of piece size.

In order to balance the gain and overhead of SLNC in CodeOn, we fix the number of pieces in a generation (K) and the number of generations (L) (e.g. 32 and 50, respectively). Although the piece size J scales linearly with the file size, since SLNC tolerates symbol errors, the size of a piece has small impact on the protocol performance.

3.4.3 Efficient Exchange of Content Reception Status

An important piece of information exchanged in CodeOn is every node's *content reception status* (i.e., how much content is downloaded for each generation), which is essential to enabling optimized, distributed transmission decisions. It could be obtained by sending gossip messages in each service time slot, but this consumes a large portion of a service time slot. In CodeOn, we choose to piggyback the reception status in safety messages, thus adding zero overhead in the service channel.

However, for SLNC, it will incur large overhead to represent the exact reception status of each generation. The decoding matrix can be represented by a single null-space vector [63]. However, the size of the reception status information adds up to $\frac{L \cdot J \cdot M \cdot K \cdot q}{N}$ bits, where Kq is the maximum size of one null-space vector. For $L = 50, J/N = 20, M = 32, K = 32, q = 8$, this amounts to 1MB which is too large.

Therefore, in CodeOn we propose a fuzzy *average rank* method to represent the reception status in an efficient way. An important property of network coding is that the rank of the decoding matrix determines the amount of received information. For two nodes u and v with symbol ranks $r_{u,i,j}$ and $r_{v,i,j}$ for position j in G_i , respectively, if $r_{u,i,j} > r_{v,i,j}$, then a recoded symbol s'_j sent from u is innovative to v with high probability [6]. Otherwise, this does not hold². Therefore, we can substitute each null-space vector with a rank, which has $\log_2 K$ bits. For a generation G_i received by node u , there are many symbol positions with different rank values. But since the size of a piece is relatively small (e.g., $J = 20\text{KB}$) compared to what can be transmitted in a 50ms slot using DSRC (55KB when data rate is 11Mbps), the ranks of various symbol positions are expected to increase at similar rates thus are similar to each other.

²The property was originally proved under random linear packet level NC, assuming $|\mathbb{F}_{2^q}|$ is large. The same applies to SLNC, which is also based on random linear coding.

	$\lfloor \bar{r}_{u,1} \rfloor$	$\lfloor \bar{r}_{u,2} \rfloor$	$\lfloor \bar{r}_{u,3} \rfloor$	$\lfloor \bar{r}_{u,4} \rfloor$	$\lfloor \bar{r}_{u,L} \rfloor$
Average symbol rank	K	K-1	9	5	0
Generation index	1	2	3	4	L

Figure 3.7: The average rank representation of a file's reception status at node u .

Therefore, we use the average rank $\lfloor \bar{r}_i \rfloor$ across all symbol positions in G_i to represent how much information is received for G_i . It is rounded to an integer, because it is more meaningful to interpret the average rank as how many “pieces” are received. It does not make much difference when the variation of \bar{r}_i is smaller than 1. The range of the rank is in $[0, K]$; if $\lfloor \bar{r}_{u,i} \rfloor < K$, this means “some information in G_i is received”; and $\lfloor \bar{r}_{u,i} \rfloor = K$ means “ G_i is received completely”. Therefore, the total overhead becomes $L \cdot (\log_2 K)$ bits, which equals 31B when $L = 50, K = 32$. Note that, this is independent of the piece size and also the file size. Now, the overhead takes an acceptable percentage ($\approx 10\%$) of the typical size of a safety message (300B) and is small enough to be piggybacked without affecting the QoS of safety applications [117]. The average rank representation is illustrated in Fig. 3.7.

3.4.4 Distributed Relay Selection in Cooperative PCD

Once vehicles are out of the range of an AP, they begin distributing the content cooperatively through the VANET. Due to the mobile nature of the VANET, the very notion of “cooperative” is captured in that vehicles distributively agree on a set of relay nodes, based only on local information.

3.4.4.1 Node utility calculation

In order to determine a set of relay nodes that can bring the largest useful amount of content to the others, each node needs to calculate its own “*utility*” based on

neighbors' content reception status collected from the safety messages in the control channel. The *utility of a generation* at node u is defined as:

$$U(G_i, u) = \sum_{v \in \mathcal{N}(u)} \text{Step}(\lfloor \bar{r}_{u,i} \rfloor - \lfloor \bar{r}_{v,i} \rfloor), \quad (3.4)$$

where $\text{Step}(x) = x$, if $x > 0$, otherwise, $\text{Step}(x) = 0$. This quantity measures how much innovative information G_i of node u can provide to its neighbors in total.

The *utility* $U(v)$ of node v is defined as the maximum value among all generations' utilities of v . This estimates the maximum additional amount of innovative information v can provide to all neighbors, and reflects v 's priority in accessing the wireless medium. We do not look at the aggregate utility of multiple generations, because to transmit many generations takes a long time while the VANET topology could change dramatically.

3.4.4.2 Transmission coordination

After nodes' priorities are determined, only a subset of the high-priority nodes (relays) will become the ones who actually broadcast their contents, in order to achieve high downloading rate and prevent from severe interference. Those relays are decided via a contention process, in a local and opportunistic way. In particular, the vehicles with the highest priorities in their locality should access the channel first, and suppress the others to avoid unnecessary packet collisions.

To this end, at the beginning of each service time slot, each vehicle v sets a backoff delay Δt which is inversely proportional to its utility before it makes channel access decision. When the timer expires, v senses the channel; if it is clear v will broadcast a short control message which is sent immediately by the MAC layer, even without additional random backoff in 802.11. Note that, an AP always has the highest utility,

so it will be a relay every time if there are vehicles still in need of the file in its local range.

Backoff delay function. A straightforward one is as follows:

$$\Delta t(v) = \left(1 - \frac{U(v)}{K \cdot |\mathcal{N}(v)|}\right) \cdot \Delta t_{max}, \quad (3.5)$$

where parameter Δt_{max} is the maximum allowable backoff delay (e.g., 2ms). However, Eq. (3.5) suffers from a major problem. That is, each node v has different neighborhood and $\mathcal{N}(v)$. If v merely has one neighbor but its generation utility for G_i is K , it will have the highest priority and $\Delta t(v) = 0$. However, compared with another node w who has 10 neighbors and utility $5K$, v is obviously not as beneficial to the whole network as w . Ideally, the $|\mathcal{N}(v)|$ should be a maximum possible neighborhood size ($|\mathcal{N}|_{max}$) and be the same for all vehicles, so that they have a common basis of priority comparison. However, setting it to be a fixed value is undesirable since the vehicle density will change.

Therefore, we estimate the maximum local neighborhood size. To do so, each node broadcasts its neighborhood size to others, and propagates its own estimation about the maximum neighborhood size. After several rounds, all nodes can obtain the same $|\mathcal{N}|_{max}$. Although the VANET topology may change every tens of time slots so that $|\mathcal{N}|_{max}$ varies over that time, we actually need not to maintain the same $|\mathcal{N}|_{max}$ for all nodes in the network. Rather, it is sufficient for vehicles in a local 1-hop range to agree on the same estimated $\widehat{|\mathcal{N}|_{max}}$, while the local propagation requires only very few rounds to converge. To achieve this, each vehicle will attach its local estimate of $\widehat{|\mathcal{N}|_{max}}$ in the safety message, and update it in a way similar to distance updates in distance vector routing.

In addition, to resolve ties, a random jitter is added to the backoff delay of each vehicle. Thus, in CodeOn, each vehicle sets its backoff delay according to the follow-

ing:

$$\Delta t(v) = \left(1 - \frac{U(v)}{K \cdot |\mathcal{N}|_{max}}\right) \cdot \Delta t_{max} + Rand(0, T_J). \quad (3.6)$$

where T_J is the maximum jitter.

Discussion of parameter selection. First, Δt_{max} must be large enough to distinguish two vehicles with adjacent utility rankings. For a common neighbor v_c of two vehicles v_1 and v_2 , the minimum difference between $U(v_1)$ and $U(v_2)$ is 1. Therefore, the minimum difference between v_1 and v_2 's backoff delays is $\min\{|\Delta t(v_1) - \Delta t(v_2)|\} = \frac{1}{K|\mathcal{N}|_{max}} \cdot \Delta t_{max}$, which should be larger than the signal propagation delay. When their distance $d(v_1, v_2) = 300\text{m}$ the propagation delay is $\frac{300}{3 \times 10^8} = 1\mu\text{s}$. Therefore, we can choose $\Delta t_{max} > 2\text{ms}$, i.e., when $|\mathcal{N}|_{max} = 50$, $K = 32$, $\min\{|\Delta t(v_1) - \Delta t(v_2)|\} > 1.2\mu\text{s}$. On the other hand, Δt_{max} shall not be too large since it will waste bandwidth. For $\Delta t_{max} = 2\text{ms}$, if transmission of one generation spans 10 service time slots (500ms), the percentage of wasted time can be as low as $2/500 = 0.4\%$.

Second, T_J should be both large enough to distinguish two contending nodes v_1 and v_2 with the same utility, and small enough to preserve the priorities between nodes with different utilities. Assume all the contending nodes have the same neighbor set. Since node utility is an integer, for node v_1 , the utility of the node v_3 with priority next to v_1 is at most $U(v_1) - |\mathcal{N}(v_1)|$ (since $\lfloor \bar{r}(v_3, i) \rfloor = \lfloor \bar{r}(v_1, j) \rfloor - 1$ for some G_i, G_j and every neighbor is counted once). Thus, the utility difference is at least $|\mathcal{N}(v_1)|$. Therefore, we have $T_J \approx \frac{\Delta t_{max}}{K}$ (e.g. 0.1ms). Note that, we do not consider $U(v_1) - U(v_3) \ll |\mathcal{N}(v_1)|$ since this is rare in reality, i.e., contending nodes always share a large portion of neighbors.

3.4.4.3 The merit of carrier sense under SLNC

We have used *carrier sense* in the contention process for transmission opportunities by potential relay nodes. That is, *a node quits the contention for channel access whenever it detects the energy of an ongoing transmission, otherwise it is allowed to transmit concurrently with others*. Traditionally for packet-level broadcast (with/without NC), this leads to the well-known “hidden terminal” problem, since such concurrent transmissions may cause interference at their neighbors³. Various mechanisms have been proposed to solve this problem, such as clearing the channel within a range larger than carrier sensing range [64]. However, due to SLNC’s better tolerance in transmission errors and interference, more aggressive concurrent transmission is possible. In [65], we show that the simple carrier sensing rule actually provides near-optimal performance in terms of average downloading rate, as the impact of hidden terminals is greatly alleviated by SLNC.

3.4.5 Broadcast Content Scheduling

Finally, we briefly highlight the way that broadcast content scheduling is dealt with in CodeOn.

3.4.5.1 Content scheduling at APs

In CodeOn, the APs broadcast the contents in a round-robin way to maintain the “information difference” between vehicles moving out of the AP range at different times. In order to make more efficient use of the VANET bandwidth, the content

³With packet-level broadcast, carrier sense is shown to work well under a two transmitter setting in [15]. Here we focus on a multi-transmitter setting instead, using SLNC.

scheduling should also be aware of local vehicles' reception status. Therefore an AP will sort its file generations according to their utilities; in addition to round-robin, it transmits the one with both larger ID and the highest utility that hasn't been transmitted in the last "batch".

3.4.5.2 Content scheduling at vehicles

After a vehicle becomes a relay node, it broadcasts the generation with the maximum utility. To avoid from transmitting duplicate information, it is important for vehicles to decide when to stop the transmission.

To this end, we estimate the number of pieces that each relay should send in one batch. The intended number of (innovative) pieces that v sends to a neighbor w for G_i is estimated as $K_{v,w} = \text{Step}(\lfloor \bar{r}_{v,i} \rfloor - \lfloor \bar{r}_{w,i} \rfloor)$. Then, the number of pieces that v should send to all neighbors for G_i is computed as

$$Z_v(G_i) = \lceil \frac{1}{|\mathcal{N}(v)|} \sum_{w \in \mathcal{N}(v)} K_{v,w} \rceil, \quad (3.7)$$

which is also the size of a batch. When the average rank $\bar{r}_{v,i}$ and those of all of its neighbors are equal to K (full rank), we set $Z_v(G_i) = 0$. Note that, the above is a conservative estimation, which treats the link qualities as perfect.

In addition, we need to deal with two situations. (1) If a batch spans multiple service time slots, relay v accesses the channel deterministically by setting its $\Delta t(v) = 0$ during the following time slots in order to finish transmitting its batch. (2) If the transmission of a batch terminates before the end of some service time slot k , to avoid waste of VANET bandwidth, v will fill the rest of the channel by transmitting additional coded pieces from the same G_i until time slot k is used up.

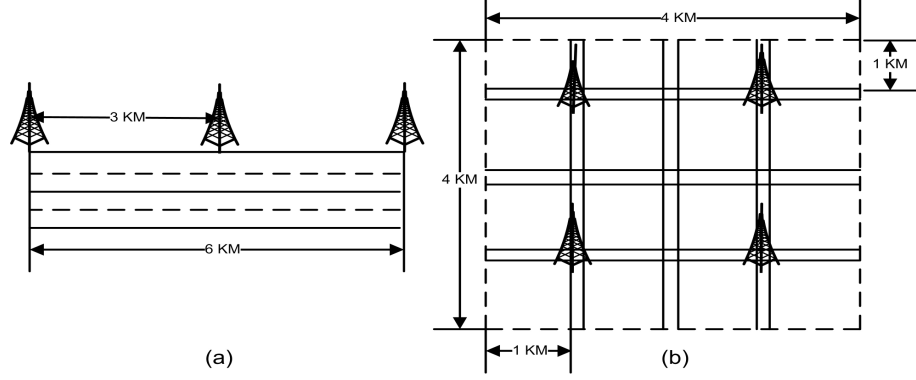


Figure 3.8: (a) Highway scenario. (b) Urban scenario.

3.5 Performance Evaluation

3.5.1 Methodology

In this section, we evaluate the performance of CodeOn by simulations. We compare CodeOn with an enhanced version of CodeTorrent [62], which is pull-based and uses PLNC. The AP is treated as a normal node. Each node periodically broadcasts a gossip message to tell others about its content availability. Based on this, a node v periodically broadcasts a downloading request, asking for the index of the rarest generation G_i among its neighbors, and attaches a null-space vector of G_i computed from v 's corresponding decoding matrix. Each neighbor w , upon receiving the request, checks if it has G_i . If yes, and if the null-space vector is not orthogonal to the subspace spanned by w 's coding vectors of G_i , w responds v with one coded piece from G_i via unicast, after waiting for a random backoff delay to reduce collisions. Only the first packet in a piece contains the coding vector; if that packet is lost then the whole piece is lost. Upon successful reception of a piece, node v continues sending another downloading request. Otherwise, v waits till the next period to broadcast its request. Nodes other than v exploit opportunistic overhearing, i.e., buffer a piece sent to v if

Table 3.2: Simulation parameter settings

CodeOn/CodeOnBasic		CodeTorrent	
Δt_{max}	2ms	Maximum random backoff delay	5ms
T_J	100 μ s	Gossip interval	0.5s
		Periodic Request interval	0.5s
		Unicast retry limit	7
Common parameters			
$ F $		16MB	
L		50	
K		32	
M		16	
q		8	
J/N		10 ($J = 10$ KB)	
CR, ER		250m, 700m	
Data rate/base rate		12Mbps (16QAM)/3Mbps (BPSK)	
SNR thresholds		15dB, 4dB	
Data capture threshold		20dB	
Data/safety message sizes		1KB, 256B (without header)	
Propagation model		Nakagami $m = 3$	

that piece is useful and received correctly.

We made the following additional modifications to CodeTorrent. We equip it with multi-channel capability as in CodeOn. To ensure a fair comparison, we apply the same channel switching mechanism in CodeTorrent, which results in a 1/2 reduction in the downloading rate. Also, in order to increase the success probability of overhearing, each node is allowed to overhear multiple different pieces during the same period, and

there are no reserved time for receiving one piece. Moreover, the packets in a piece do not have to arrive in order; a node flushes an incomplete piece after a certain time from its first reception, say 0.5s.

In addition, we introduce a variation of CodeOn, *CodeOnBasic*, which is also push-based, piece-division but based on PLNC. A piece is used as a whole for encoding and decoding. A node buffers any overheard piece as long as it receives the coding vector in the first packet of that piece, and the same buffer flushing mechanism as in CodeTorrent is adopted. Moreover, in content scheduling a relay node pads a service slot with whole pieces. If the remaining service slot time is not enough for sending a whole piece, it terminates the current batch, rather than filling with individual packets. Other than that, CodeOnBasic is the same with CodeOn.

We implemented CodeOn, CodeOnBasic and CodeTorrent in NS-2.34 [2]. For CodeOn, we implemented the run-length coding with dynamic programming algorithm to minimize the communication overhead in sending each coded piece [54]. We simulate independent symbol errors in a packet, and in simulation the number of runs seldom exceeds 20 for 10KB pieces. Packet capture effect is enabled; and when two packets collide, if no packet can be captured, the symbols from the point of collision are all discarded. Otherwise, the captured packet is received as usual. We do not consider vehicular buffer constraints.

We have a few notes on broadcast data rate selection. First, the safety message's communication range shall be larger than that of PCD data packets, so that the neighbor set used in relay selection can cover the set of nodes that can receive a data packet. Otherwise, the utility cannot truthfully reflect a node's total content usefulness. Considering the reliability of safety messages, we chose the base rate (3Mbps) for broadcasting safety messages. Second, we want to achieve high downloading rate for PCD. For SLNC, choosing a higher data rate is beneficial because it has better

error-tolerance. Since a too high rate is also undesirable due to very small communication range, the data rate of PCD packets is set to be 12Mbps throughout the chapter. The determining of optimal data rates is left for the future works.

3.5.2 Simulation Settings

We consider both highway and urban scenarios (Fig. 3.8). We use a VANET mobility generator [3] to generate the movement patterns. Vehicles are placed uniformly at random in the road area; when a vehicle hits the boundary it randomly selects another entry point of the map. This removes the boundary effect; equivalently, the AoI is infinitely large. Table. 3.2 is a list of parameters.

The highway scenario consists of a bi-direction, four lane highway with length 6km. Vehicles' speeds are randomly drawn from $[20, 30]$ m/s with a maximum acceleration of $0.5m/s^2$. The urban scenario is $4km \times 4km$ as shown in Fig. 3.8. In order to evaluate the impact of topology and traffic density, we simulate sparse and dense traffic for both scenarios. The sparse settings simulate delay-tolerant network (DTN), where the total number of vehicles is 100 for highway and 160 for urban. The dense highway setting has 300 vehicles while the dense urban has 400 vehicles.

3.5.3 Results

3.5.3.1 Downloading performance

We evaluate the downloading performance from three aspects: (1) downloading progress, which is the change of average downloaded percentage of the file with the elapsed time (averaged upon each vehicle); (2) average downloading delay: the average elapsed time

from downloading start to 100% completion; (3) average downloading rate, where the downloading rate for each vehicle is the file size divided by its downloading delay.

We present the downloading progresses in Fig. 3.9 for all four scenarios. It can be seen that CodeOn significantly outperforms both CodeOnBasic and CodeTorrent. The downloading progress of CodeOn is the fastest (Figs. 3.9 (a)–(d)), especially when the average downloaded file percentage is below 90%. The comparison between CodeOnBasic over CodeTorrent demonstrates the effectiveness of our new set of push-based protocol design, while the comparison between CodeOn and CodeOnBasic shows the advantage of the use of SLNC, which we will discuss later.

Next, we evaluate the average downloading delays and rates in Fig. 3.10. Some of the average delays are not shown since their downloading progresses cannot reach 100% within the given simulation period. There are two key observations. First, the average downloading rates of CodeOn are much higher than both CodeOnBasic and CodeTorrent, for both highway and urban scenarios and both sparse and dense traffic. Second, CodeOn maintains high downloading rate in all cases shown, especially for the two extremes, i.e., sparse urban scenario and dense highway cases which represent the lowest and highest traffic density, respectively. This means CodeOn is the most robust to variations in topology and vehicle density.

The first phenomenon above is attributed to the push-based protocol design combined with SLNC. In CodeOn, using a prioritized relay selection mechanism with the transmission coordination that avoids heavy packet collisions, the contents can be distributed proactively to the vehicles in the AoI so that the VANET bandwidth is fully utilized. Moreover, each piece of transmitted content brings the maximum usefulness to a relay's whole neighborhood. In addition, with SLNC, the symbols in content pieces are received with higher rate from APs and relays, which results in higher downloading rate.

The robustness of CodeOn under low traffic density is mainly attributed to the enhanced reception reliability brought by SLNC. Compared with PLNC, SLNC actually enables vehicles in a larger range to receive some useful information in a piece. In the sparse urban setting, although the vehicular contact opportunities are much less, CodeOn is able to mitigate the impact of low traffic density.

On the other hand, CodeOn is less affected under dense VANET. For the dense scenarios, the differences between CodeOn's downloading rates and those of CodeOn-Basic and CodeTorrent are both larger than the sparse scenarios (Fig. 3.10 (b)). For CodeTorrent, the performance degradation is due to lack of coordination and using of PLNC for a large file. (1) Under dense VANET, the number of requesting vehicles in a node's neighborhood increases. Since there may be more than one responder for each requester, the chance of packet collisions also increases. The unicast-with-overhearing mechanism retransmits packets after they are collided, which aggravates the problem. (2) For both CodeTorrent and CodeOnBasic, the use of PLNC prevents a requester from receiving a whole piece under frequent packet collisions. However, through prioritized relay selection and the use of SLNC, CodeOn alleviates the above problems dramatically.

3.5.3.2 Fairness

The fairness is embodied in the distribution of downloading delays of all vehicles, shown in Fig. 3.11. We show the distributions for all three cases. The most fair situation has zero variance, i.e, all the delays are equal. From Figs. 3.11 (a)-(c), one can see that the distributions of CodeOn are more concentrated (more fair) than those of CodeOnBasic and CodeTorrent. Few vehicles need very long time to receive the whole file. Again, the same robustness of CodeOn to variations in traffic density can be observed.

Table 3.3: Protocol efficiency (Total number of pieces in the file: 1600).

Protocols	Percentage of noninnovative received pieces	Average # of failed overheard piece per received piece	Average # of piece sent by a vehicle	Average # of piece sent by an AP
Sparse highway scenario				
CodeOn	N/A	N/A	2202.12	26023.00
CodeOnBasic	0.476	4.26	4054.87	51578.00
CodeTorrent	0.325	27.27	32889.87	53665.00
Sparse urban scenario				
CodeOn	N/A	N/A	1031.14	43445.25
CodeOnBasic	0.228	3.47	3525.31	143905.00
CodeTorrent	0.167	80.74	52465.69	222287.50

The superiority of CodeOn in fairness is still attributed to the use of SLNC. SLNC enables more reliable reception of the coded symbols, since an overhearing node will buffer any innovative clean symbol it received. In CodeOn, since the granularity of information reception is smaller, and vehicles have similar opportunities to contact with APs and other vehicles within a time period of order 1000s, their reception progresses have small variance. However in CodeOnBasic and CodeTorrent, a vehicle either receives a whole piece or receive nothing, so the variance among reception progresses is larger. Again, the results on fairness demonstrate the benefit of using SLNC and the effectiveness of CodeOn’s protocol design.

3.5.3.3 Protocol efficiency

One may wonder if CodeOn achieves fast push-based downloading by sacrificing protocol efficiency. To further investigate this issue, we present the results on protocol efficiency in Table. 3.3.

As we have shown in Sec. IV-C, the protocol overhead of CodeOn is small. To evaluate the amount of incurred data traffic, we show the average number of pieces sent by a vehicle and an AP during the whole simulation time (a node will not transmit when all of its neighbors receive 100% of the file). CodeOn has the fewest number among the three protocols. Its high protocol efficiency comes from both the high symbol reception probability due to SLNC, and the high usefulness of the transmitted symbols due to relay selection. As CodeOnBasic adopts the same relay selection mechanism, it enjoys similar high protocol efficiency to CodeOn. However, CodeTorrent sends many pieces due to a large number of failed overhears explained in the following. Note that, the APs are always the most advantageous nodes so they transmit a lot in all three protocols.

To further study the role of relay selection, we compute the percentage of total number of non-innovative pieces out of the total number of received pieces, which reflects the usefulness of the received content. Also, we calculate the average number of failed overheard pieces (in which the coding vectors are received but not all the subsequent packets) per received piece. For the former, CodeOnBasic is slightly higher than CodeTorrent; but for the latter, CodeOnBasic is much lower than CodeTorrent. This is because in CodeTorrent a responder uses the requester's null-space vector to decide whether to transmit a coded piece, which is definitely innovative to the requester. However, in CodeTorrent a responder's transmission mainly benefits the requester itself but few others due to uncoordinated transmissions. On the other hand, in CodeOnBasic the selected relays can benefit their whole neighborhood, while the broadcasted contents are still highly useful. As a result, both the downloading rate and efficiency are high.

3.5.3.4 Discussion

Finally, we give some insights that can be obtained from our results.

Push v.s. pull. First we compare the *push versus pull* based content distribution in VANETs. CodeOnBasic and CodeTorrent are both based on PLNC, but the former performs much better than the latter for all scenarios in Figs. 3.9 and 3.10. An obvious reason is the difference on the bandwidth utilization. CodeOnBasic let the APs and relays broadcast proactively (push), so that the service time slots are almost fully utilized. However, in CodeTorrent each node make requests (pull) periodically and responders transmit passively. Whenever received a piece in error, a requester will wait until the next period to make subsequent requests. Due to the lossy property of the wireless channel in VANETs, this happens frequently so that the service channel is under-utilized.

However, a more fundamental reason that the push method in CodeOn and CodeOnBasic is better, goes to the relay selection mechanism. If there was no transmission coordination between vehicles, the push-based content distribution could easily lead to frequent packet collisions. For CodeTorrent which is pull-based, its high chance of packet collisions is already evident from the large number of failed overheard pieces of CodeTorrent in Table 3.3. One can imagine that this situation will be aggravated if CodeTorrent is changed to push-based where nodes transmit more aggressively.

Apart from transmission coordination, in designing a push-based protocol, it is always critical to maximize the usefulness of the broadcasted content from each relay nodes. Since nodes do not make explicit downloading requests, and since “push” uses broadcast transmission in nature, it is basically impossible to ensure the usefulness of broadcast content of a relay for all its neighbors. In CodeOn and CodeOnBasic,

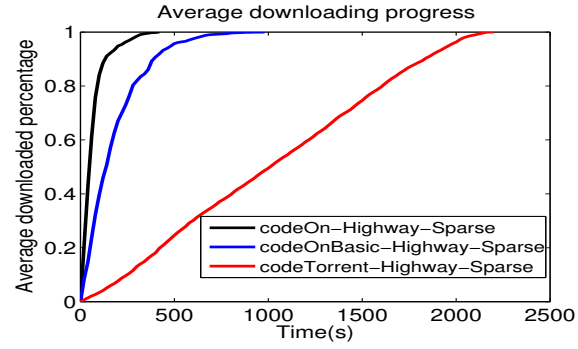
our approach is to select a relay to be the one that can bring maximum amount of useful contents to all its neighbors, by implicitly calculating node utilities based on fuzzy average rank differences. In contrast, in CodeTorrent each responder will only ensure the content to be 100% innovative for one requestor, using accurate null-space indicators. Interestingly, as one can see from the number of non-innovative pieces in Table 3.3, the number of CodeOnBasic is quite close to that of CodeTorrent, which can be regarded as a lower-bound. This proves the effectiveness of our relay selection approach.

SLNC v.s. PLNC. The advantage of using SLNC is evident by comparing CodeOn with CodeOnBasic in Fig. 3.9, which are only different in the network coding method. With PLNC, in CodeOnBasic a coded packet is discarded whenever it is received in error, which leads to unsuccessful reception of the whole piece. However, with SLNC, CodeOn records every innovative received symbol in a piece, and then combines innovative symbols to decode the piece.

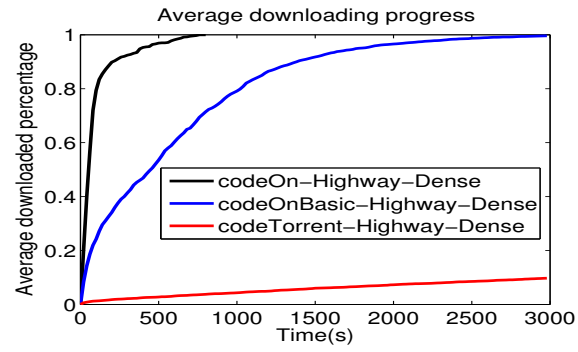
As previously mentioned, SLNC is superior in tolerating transmission error. This is a direct reason of why CodeOn has the best robustness under dense traffic scenarios. By both coding and receiving according to a small granularity of symbols (yielding higher content diversity), vehicles have higher chances of receiving some useful information, even when packet collisions are frequent due to dense traffic, or when there are few vehicles or APs around. However, with PLNC, the content diversity is lower. Although our push-based protocol design is able to choose the best relay nodes and alleviate collision, without SLNC, small downloading delays and a high level of fairness are very hard to achieve for all topologies and traffic densities.

3.6 Summary

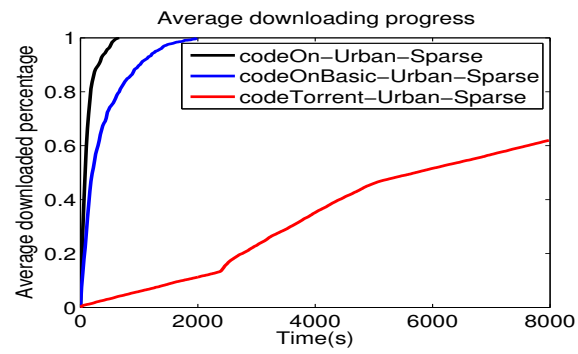
In this chapter, we have presented CodeOn, a novel push-based popular content distribution scheme for vehicular networks, where large files are broadcasted proactively from a few APs to vehicles inside an interested area. CodeOn is designed to primarily achieve high downloading rate and high protocol efficiency. To combat the lossy wireless transmissions in VANETs, we leverage symbol level network coding, which enjoys the benefits of both network coding and symbol-level diversity. The use of SLNC contributes as a key factor for the superior and robust performance of PCD across VANETs with different traffic densities and topologies. In addition, to allow “push” efficiently with maximized information usefulness, and to avoid from incurring frequent packet collisions, we designed a prioritized relay selection algorithm along with a lightweight transmission coordination mechanism, which are shown to improve greatly upon a previous pull-based protocol, CodeTorrent. Compared with CodeTorrent, CodeOn achieves a significant gain in terms of average downloading rate, where one important part of it comes from the use of SLNC, and the other is attributed to the new push-based protocol design. Our work demonstrates the strong potential to achieve fast PCD in realistic vehicular networks.



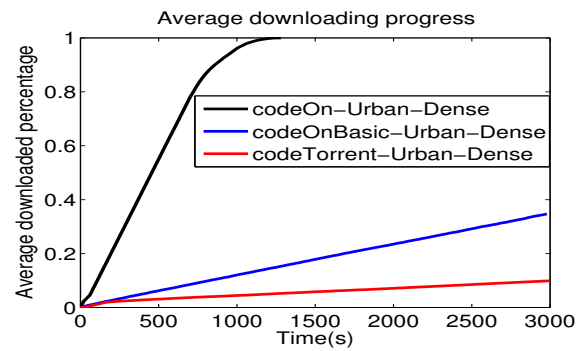
(a) Sparse highway.



(b) Dense highway.

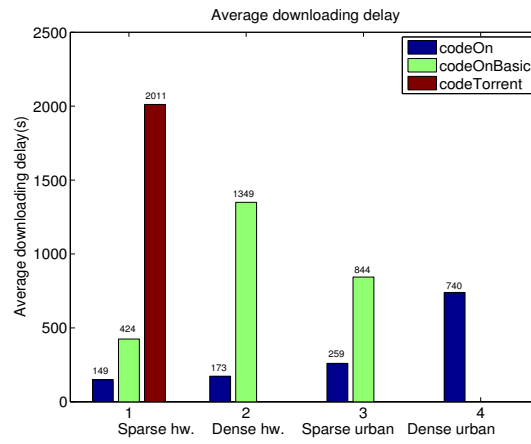


(c) Sparse urban.

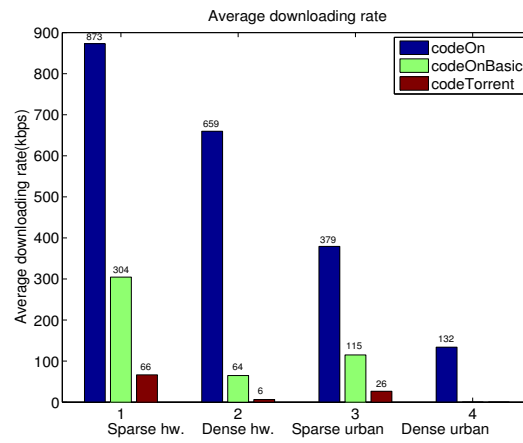


(d) Dense urban.

Figure 3.9: Downloading progresses.

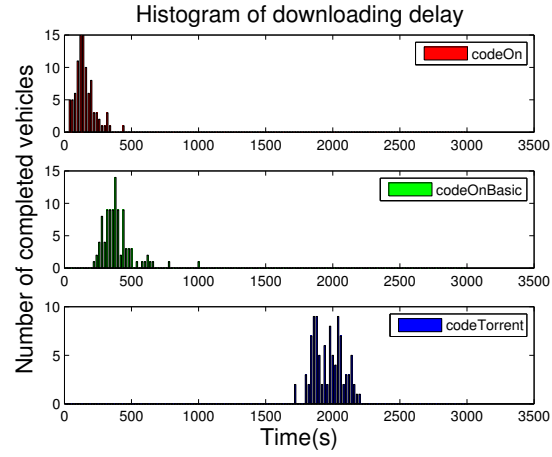


(a) Average downloading delay.

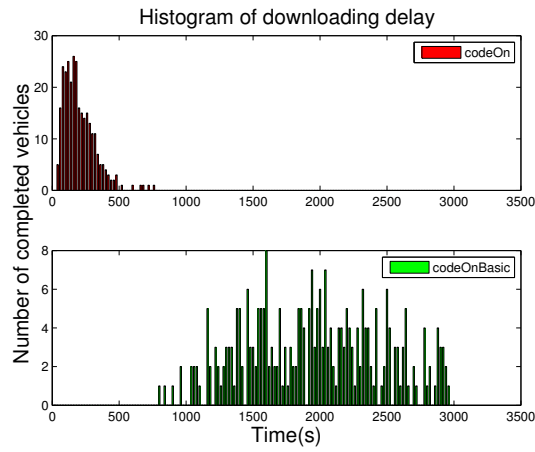


(b) Average downloading rate.

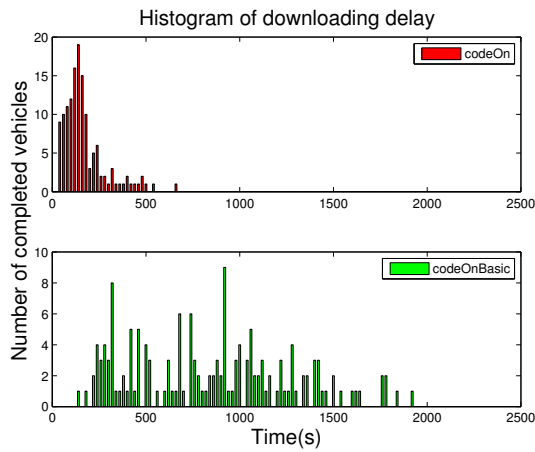
Figure 3.10: Downloading delays and rates.



(a) Sparse highway.



(b) Dense highway.



(c) Sparse urban.

Figure 3.11: The distributions of downloading delays.

Chapter 4

Live Multimedia Streaming in VANETs using SLNC

4.1 Introduction

In this chapter, we focus on how to provide another important but quite different application in VANETs, which is live multimedia streaming (LMS). The content of LMS usually consists of video, audio which could provide more precise, comprehensive and user friendly information than plain text based applications. Typical scenarios for LMS applications could be illustrated as the following example. A roadside access point (AP) continuously broadcasts the streaming video of the current road traffic conditions to vehicles driving towards it for intelligent navigation, which is especially useful in inclement weathers. Also, when a police vehicle spots an accident, it disseminates emergency-related LMS content of the accident to vehicles following several miles behind for early warning. Then the paramedics can also make preparation more purposefully in advance based on the collected LMS content on their way

to the accident scene.

Different from non-streaming services, such as popular content distribution studied in the previous chapter, where the main focus is on the average downloading rate, LMS services require not only high average streaming rate but also stable streaming rate for the purpose of smooth playback. LMS services are also different from non-live streaming services such as video-on-demand, where various vehicles maybe interested in different contents and those contents are not closely related to real world's time. For LMS services, the streaming contents are generated as time progresses, usually wanted by most of the vehicle and only useful to vehicles within a short period of time, e.g., several to tens of seconds. However, these time constraints are usually not as tight as those of real-time services, like intelligent collision avoidance, which usually requires delay smaller than hundreds of milliseconds.

Generally speaking, there are three primary requirements for LMS services in VANETs. Firstly, considering the large volume of each LMS content, all the receivers should achieve stable and high streaming rate for smooth playback. Note that the rate only needs to reach the requirements of related multimedia standards and higher rate is not necessary. Secondly, the service delivery delay should be short for all the receivers, and the delay variation should be small for neighboring receivers for possible coordinated actions between them, for example, bypassing a blocked road. Thirdly, LMS services should consume minimal amount of bandwidth resource for better coexistence with other competing services, since the bandwidth is a precious resource in VANETs. Essentially, this corresponds to improving bandwidth efficiency.

These requirements are conflicting and it is very challenging to achieve them simultaneously, especially considering VANETs' specific characteristics. In order to ensure smooth playback of LMS content, we have to combat with the lossy vehicular wireless links and highly mobile and dynamic topology of the underlying VANETs.

In vehicular communications, packet loss is a frequent phenomenon due to channel fading. To ensure stable streaming reception within short time delay, a large number of (re)transmissions would be incurred, which severely decreases the bandwidth efficiency. In addition, smooth playback requires vehicles to make local optimal transmission decisions, such as which vehicle should transmit what content to which neighbors. This means vehicles need to learn precise and in-time neighbor information (such as reception status). However, under VANETs with ever-changing topology, this learning process may lead to high communication overhead. Thirdly, VANETs tend to experience frequent partitions [112], which increases the difficulty of determining the best relay nodes and proper transmission opportunities for them. This may result in major performance degradation without careful protocol design. In sum, all these factors make it hard to ensure smooth playback while keeping low bandwidth consumption for LMS services in VANETs.

Most existing works [86,93,95,106] on live multimedia streaming focused on traditional wired or wireless networks, where either the links are reliable or the topology is relatively stable over time. Many of these works adopted packet level network coding technique. Compared with traditional store-and-forward communication paradigm, NC, by allowing nodes to combine different packets received previously together to generate coded packets for transmitting, has been shown to be an effective approach to improve the network bandwidth efficiency and simplify the protocol design for LMS services in those networks. On the other hand, only a few works [87] have applied NC to providing LMS services in VANETs. However, the gain of NC tends to be offset by severe packet collisions due to lack of proper transmission coordination mechanism among vehicles. To the best of our knowledge, none of existing works can well satisfy all the requirements simultaneously.

In this chapter, we try to exploit the more advanced symbol-level network coding

for designing a distributed live multimedia streaming scheme in VANETs and we make the following main contributions.

- We proposed CodePlay to fully exploit the benefits of SLNC in VANETs, the core of which is a coordinated local push mechanism. In order to disseminate the streaming content from sources to all the receivers timely and smoothly, a group of spatially separated relays are selected distributively, whose transmissions can bring most useful information to vehicles nearby. Each relay actively pushes coded information to cover its neighborhood. By taking advantage of SLNC's better tolerance for transmission interference, The concurrent transmissions of all relays could be optimally coordinated locally, which could provide continuous streaming coverage for the whole VANET efficiently.
- To enable CodePlay to perform well under various VANET densities, we also proposed an opportunistic transmission scheduling algorithm based on well-designed carrier sensing mechanism, where the network's spatial reusability can be adaptively enhanced with negligible overheads.
- We implemented CodePlay in NS-2 and carry out extensive simulations to evaluate its performance by various practical metrics. We showed both the potential and the constraints of providing LMS services in VANETs. Compared with traditional PLNC technique, the adoption of SLNC can provide more and better design choices for VANET designers. Also the particular topological characteristic of VANETs [112] (the vehicles running on the highway tends to form disjoint clusters rather than uniformly distributed) needs to be specifically considered into the scheme design. As far as we know, CodePlay made the first step towards this direction.

4.2 Related work

4.2.1 NC-based streaming schemes

Streaming services are widely deployed on the Internet nowadays, such as PPLive, PPStream, etc. In particular, network coding (NC) has been shown to be an effective technique that can improve the user experience of video streaming service for large scale systems. For example, Wang *et.al.* proposed R^2 [106], a random push-based P2P scheme using network coding. Also, Liu *et.al.* deployed a NC-based on-demand streaming scheme in a large-scaled commercial system [131], which showed the benefits of NC for multimedia streaming in a real P2P network. In wireless mesh networks, Seferoglu *et.al.* proposed a video-aware opportunistic network coding scheme across different flows [95]. However, all these schemes are for traditional wired or wireless networks and are not suitable for VANETs, due to VANETs' unique characteristics described previously.

4.2.2 Streaming schemes for VANETs

Previous schemes for supporting various kinds of streaming services in VANETs can be divided into two categories, which are introduced as follows:

(1) Schemes focusing on application layer. Bucciol *et.al.* carried out a series of experiments using two vehicles under different scenarios, which proved the possibility of video streaming between moving vehicles [16]. Mancuso *et.al.* presented a resource management mechanism based on proxy server equipped on the vehicle to support various streaming services for the customers in the same public vehicle, e.g. a moving train connected to the network via a satellite link. Maurizio *et.al.* proposed a

real-time video transmission scheme in vehicular networks [14]. This scheme only considers unicast sessions and heavily relies on fast and reliable feedback from receiver side, which itself is hard to be guaranteed in VANETs. Qadri *et.al.* showed that by adopting error resilience coding, state-of-the-art routing protocols can support multicast video streaming in city VANETs when the network is not dense [90]. These works mainly showed the possibility of video streaming in VANETs and did not consider more practical issues such as dealing with dynamically changing network density, minimizing bandwidth cost, conforming to standards for wireless access in VANETs, etc., all of which are carefully considered in this chapter.

(2) Schemes focusing on network and MAC layers. Park *et.al.* proposed NCDD for emergency related video streaming in VANETs using NC [87]. In this scheme, the transmission of each vehicle is triggered by a timer set upon the reception of every new packet. Since neighbors' current reception status is not considered, the broadcasted packets are not always useful for neighboring vehicles, which decreases the bandwidth efficiency. Also due to lack of coordination between concurrent transmitting vehicles, the scheme tends to suffer from severe collisions, especially under dense vehicular traffic.

Soldo *et.al.* introduced SMUG, a TDMA-based scheme to support streaming media dissemination in city VANETs [99]. A tree structure is established for broadcasting streaming video content. However, it is hard to maintain a stable and up-to-date communication structure for dynamic VANETs, thus stable streaming rate is difficult to achieve. In [34], Guo *et.al.* proposed V3, a live video architecture for VANETs, where directed broadcast is adopted for remote video request scenarios, which are quite different from the LMS based applications discussed in this chapter.

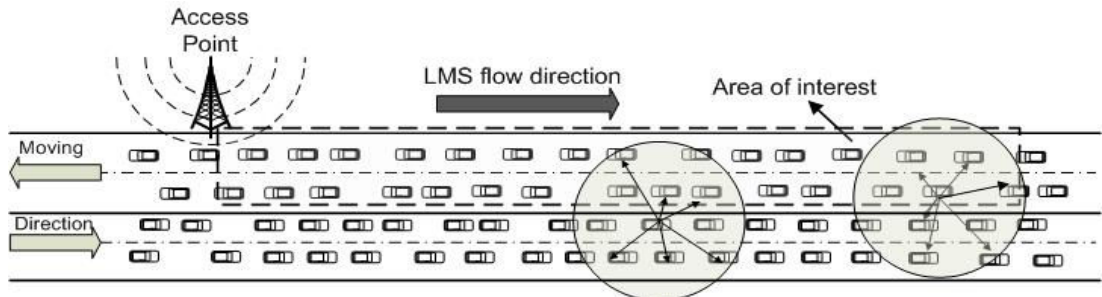


Figure 4.1: The architecture for LMS.

4.3 Problem Formulation

4.3.1 Model and Assumptions

We consider the following LMS services in VANETs, which is based on the network model presented in chapter 3. Several dedicated sources actively broadcast LMS contents (e.g., local road traffic monitoring videos) with constant streaming rate to vehicles inside AoI. As a motivating scenario, we assume a highway with bidirectional traffics. At the left end of the road, an AP is deployed, which continuously broadcasts LMS contents about its local traffic condition to all the vehicles driving towards it for providing intelligent navigation¹. The service architecture is illustrated in Fig. 4.1, where a live multimedia stream propagates against the moving direction of vehicles within AoI. We assume that the vehicles in the opposite road segment of AoI also assist the propagation of the multimedia stream, although they are not intended receivers.

We make the same assumption about the vehicles as in the chapter 3. First, each vehicle is equipped with an on board unit with a wireless transceiver (single radio), and operates on two-channel mode, one control channel and one service channel.

¹We can imagine that many such APs are deployed along the highway; here we show a typical part of the whole system. Also, for simplicity we only consider single streaming flow in this chapter.

The time is divided into 100ms slots and all nodes (including vehicles and APs) are synchronized to switch simultaneously and alternatively between the control channel and service channel. Second, each vehicle is equipped with a GPS device which could obtain real-time precise location (in the order of meters) information and synchronizes its clock (error smaller than 100ns).

4.3.2 Objectives

The design of CodePlay pursues the following primary objectives.

- Smooth playback at all the interested vehicles, which can be translated into providing stable and high streaming rate.
- Prompt service delivery, which can be translated into short end-to-end delay for all the receivers. For a receiver, this delay is defined as the elapsed time from the generation of specific LMS content at the source to the start of playback of this content at the receiver.
- Minimized bandwidth cost, which can be translated into incurring small protocol overhead and data traffic. This is for better coexistence with other possible services.

4.4 The Design of CodePlay

4.4.1 Design Rationale of CodePlay

4.4.1.1 Push-based Network Coding is Good for LMS

Most LMS schemes adopting store-and-forward communication paradigm are pull-based, where each receiver sends explicit requests to other nodes for retrieving the

missing contents. These schemes incline to suffer from low bandwidth efficiency in VANETs due to high protocol overheads and dependence on TCP-based content retrieving [106], which is well-known for low efficiency in lossy wireless networks [63].

The design of CodePlay is partially inspired by the following two works. R^2 [106], a push-based peer-to-peer LMS scheme that exploits NC technique for Internet, where seeds actively push coded packets to downstream peers without the need of costly requests and collaboration. NCDD [87], a push-based scheme that exploits NC technique for emergency related video streaming in VANETs, which constraints the content retrieving process within one hop, while not suffering from scarcity of useful neighbors due to the use of NC. Both schemes apply UDP-based content retrieving, in sharp contrast to the traditional pull-based schemes adopting TCP-based communications.

4.4.1.2 SLNC Potentially Performs Better than PLNC in VANETs

Although NC benefits, the PLNC technique suffers from unnecessary performance degradation in VANETs. In PLNC, a small portion of the packet which is not received correctly will render the whole packet useless, and this happens frequently under the lossy wireless medium in VANETs. SLNC, however, by operating on smaller symbols and thus benefiting from both symbol-level diversity and NC, can potentially achieve higher bandwidth efficiency than PLNC. And this has been shown in content distribution in VANETs by [66].

However, to provide satisfiable LMS services using SLNC in a dynamic and lossy VANET, the biggest challenge is how to achieve multiple objectives (stable high streaming rate, small service delivery delay and minimal bandwidth consumption) simultaneously. Essentially, this corresponds to the following design problem of Code-

Play: **which vehicles should transmit what content to whom at which service time slots?** In particular, since broadcast is adopted as the basic transmission paradigm, and multiple receivers may have different stream reception and playback statuses, how do we select proper relay nodes to ensure smooth playback of multiple vehicles? How to coordinate the transmission of multiple relays so that spatial reusability is maximized? How to efficiently achieve the above with small overhead? All these key issues imply that wholly new design considerations are needed.

4.4.1.3 Make All Ends Meet — Coordinated Local Push with SLNC

Corresponding to the above issues, our solution is a *coordinated local push* (CLP) mechanism based on SLNC, which mainly consists of two parts: distributed relay selection and transmission coordination of relays. The core idea is as follows. In each service time slot, a set of spatially separated relay nodes are dynamically and distributively selected. By actively pushing coded LMS contents in the locality, each relay node can provide most useful information to its neighbors so that their collective and individual needs for smooth playback can be both well satisfied. In time, the relays belonging to consecutive road segments are scheduled in a round-robin fashion, so as to achieve smooth propagating of the multimedia stream continuously throughout the network to reduce the end-to-end delay. In space, the transmission of all the relays are coordinated in a way that maximizes the overall streaming rate, by exploiting the increased spatial reusability enabled by SLNC.

4.4.2 Design overview

For proof-of-concept, in this chapter, we describe CodePlay under a one-dimensional highway scenario (Fig. 4.3). However, CodePlay can be easily extended to the urban

scenario, i.e., two-dimensional case. In the following, we will illustrate the core design of CodePlay, from both global and local point of views, by using a simple example.

4.4.2.1 Global View

Basically, we want to have smooth propagation of LMS content from the source to vehicles inside the AoI. Due to the limited transmission range of wireless communication, the source can not cover all the vehicles directly and needs multiple vehicles to be selected as relays to help it, the coordination of which shall be facilitated in an efficient way, yet, in a dynamic VANET. Considering the broadcast nature of wireless medium, the primary goal of such a coordination scheme is to ensure bounded channel access delay for each vehicle, which renders most of the random based channel access schemes not appropriate.

The idea of CodePlay is that, *we introduce road segmentation during initialization so that the relay selection could be made locally within each segment and allow relays of adjacent segments to share the wireless channel resource in a round-robin fashion.*² Specifically, each road is divided into fixed segments of equal length and is uniquely numbered, which can be pre-configured and provided by the access points with the help of GPS. Every vehicle is assumed to possess this information before entering the AoI. For each time slot, a unique relay will be locally selected from all the vehicles within the same road segment based on the mechanism presented in the following section. The round-robin fashioned channel access for those relays is illustrated in Fig. 4.2. Suppose the length of the round-robin cycle is 3. Initially (we denoted as service time slot 0), the AP, relay in segment 3 and relay in segment 6 are selected to used the service time slot 1; then the relays in segments 1, 4 and 7 will continue

²We note that similar segmentation approach has been used for solving different problems in previous works [50, 64].

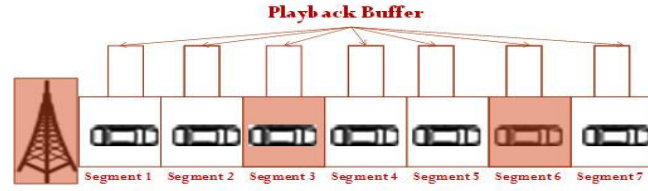
to use the service time slot 2 and so on. In this way, the LMS content is gradually and stably propagated from the source (AP) to all the vehicles, which is illustrated by the increasing reception level in the playback buffer of each vehicle, the definition of which is given in section 4.4.3.

4.4.2.2 Local View

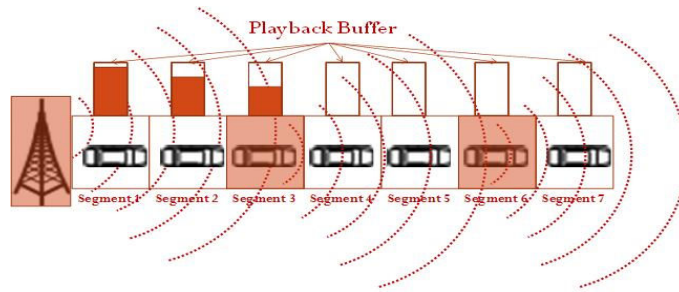
To achieve the performance described above, several design choices need to be made, which are introduced in the following.

1. *Local coordinator selection.* We should ensure that only unique optimal relay could be selected within each segment for the purpose of avoiding heavy collisions. However, how to achieve this under error-prone wireless vehicle-to-vehicle communication is a challenging problem, because a node needs to know its neighbors' current reception statuses for the purpose of selecting optimal relay. Unfortunately, there is no efficient approach to frequently exchange such large amount of information reliably between nodes due to those characteristics of VANETs described in section 4.1.

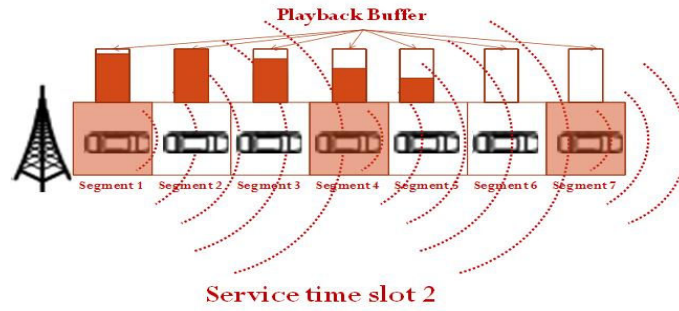
Our solution is to divide the relay selection into two steps: firstly, let vehicles within the same road segment achieve an agreement on the selection of a local “*coordinator*”; secondly, this “*coordinator*” selects the unique relay on behalf of other nodes. This is achieved by taking advantage of the obligated safety message service in the control channel required by the IEEE 802.11p standard, where every vehicle has to broadcast a safety message to inform its current location in each control time slot. CodePlay lets each vehicle piggyback a short piece of additional information in the safety message. This information contains the minimum Euclidean distance to the geographical center of the road segment that this vehicle currently knows, and also the vehicle's current LMS content reception and playback status (Fig. 4.3(a)). We will introduce an efficient representation of this information later. The piggybacked



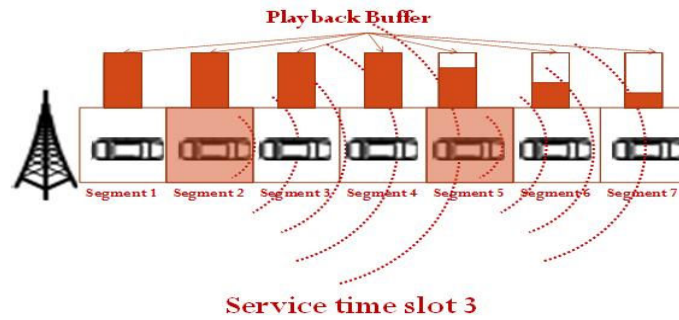
(a)



(b)



(c)



(d)

Figure 4.2: The illustration of smooth propagation. Shaded segments are selected to transmit during the corresponding service time slots.

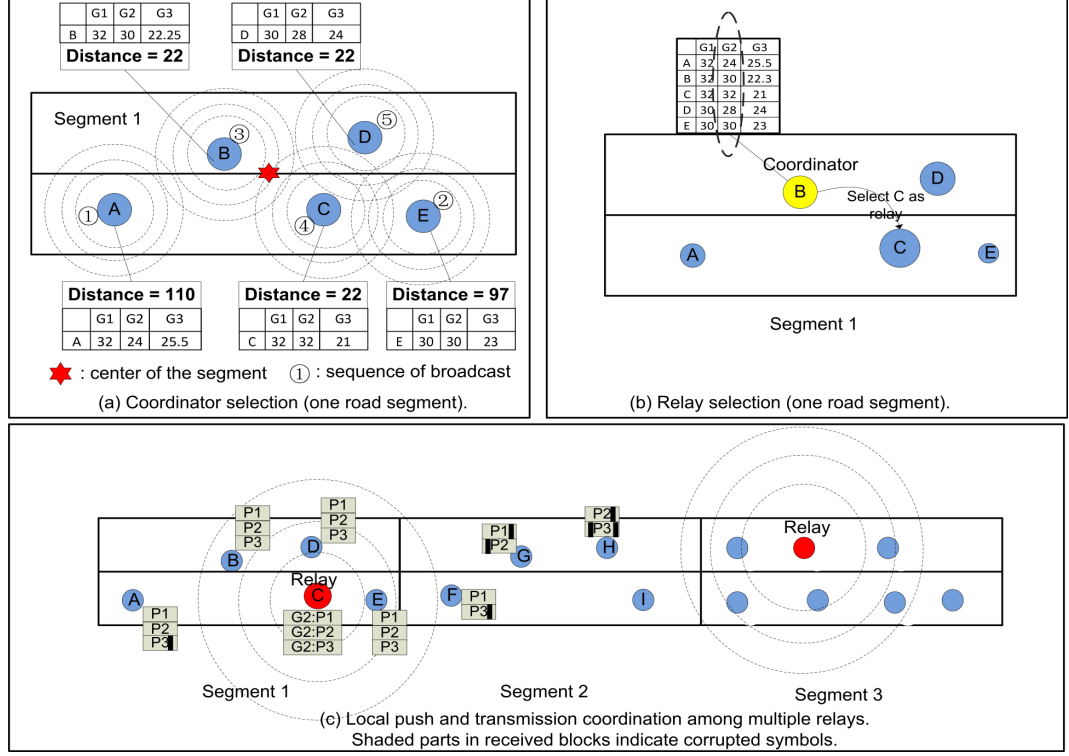


Figure 4.3: The concept of coordinated local push.

Euclidean distance could either be the vehicle's own distance to the center or the broadcasted distance overheard from another vehicle in the same segment. For example, in Fig. 4.3(a), vehicle A firstly broadcasts its safety message, thus it considers itself as the closest one to the segment center and piggybacks its distance 110 within the safety message. Vehicles E and B, which are the following ones to broadcast, will do the same as A. However, for vehicle C and D, since they overheard B's safety message and knew that B is closer to the segment center, they will piggyback B's distance in their safety messages. In this way, vehicle B, the closest to the center of the segment, will be selected as local coordinator with consensus by all the vehicles within the segment.

2. *Distributed relay selection.* The coordinator selects real relay based on the reception and playback statuses of all nearby vehicles, i.e., what LMS contents each of

them have received or are needed for playback in the immediate future. In particular, the coordinator computes the “*utility*” of each node in its segment as how much useful information can that node provide to its neighbors, and designates that node as relay via unicast. This is shown in Fig. 4.3 (b), where coordinator B designates vehicle C as relay and the generation G_2 as the broadcasted content. Since from B’s reception table, we can see that C’s G_2 could providing the most useful information to all the other neighboring vehicles. One generation represents a short period of LMS content and the precise definition will be given in the following section.

3. *Local push and transmission coordination of relays.* In order to create a stable and continuous LMS flow, only relays in certain segments are allowed to transmit concurrently in each service time slot. Those relays actively “push” coded LMS blocks to their vicinity, which will be received by neighboring vehicles. To maximize spatial reusability, we exploit SLNC’s symbol-level diversity by purposely reducing the distance between two concurrent transmitting relays (thus introducing a proper amount of signal interference). In the snapshot given in Fig. 4.3 (c), the two relays are separated by two road segments, which maybe too close if packet level collision avoidance mechanism is adopted. Specifically, we address the following issues: i) what is the optimal number of segments between two adjacent transmitting relays? ii) how can we opportunistically schedule the relays’ transmission if the density of the VANET is so sparse that some road segments are empty and no relay could be selected for them?

4.4.3 LMS Using Symbol Level Network Coding

SLNC is used throughout the design of CodePlay, and in this section we present the way SLNC actually operates in CodePlay. The source divides the original streaming content into equal-sized blocks or *generations* $G_1, G_2, G_3, G_4, \dots$, each representing

T seconds of playback. Every generation is again divided into K *pieces*, each of them consisting of M symbols. K is also called generation size. To reduce the encoding/decoding complexity, SLNC is carried out within each single generation [21]. At the source, the j^{th} symbol (at j th position) \mathbf{s}'_j in a coded piece is a random linear combination of the j^{th} symbols of all the K original pieces within the generation [38]:

$$\mathbf{s}'_j = \sum_{i=1}^K v_i \mathbf{s}_{ji}. \quad (4.1)$$

where \mathbf{s}_{ji} is the j th symbol in the i th original piece, and $\mathbf{v} = (v_1, \dots, v_K)$ is called the *coding vector* of this coded symbol, each element of which is randomly chosen from a Galois field \mathbb{F}_{2^q} . The coding vector, which is shared by all the coded symbols, will be transmitted along with the coded piece for the purpose of easily decoding at the receiver side. The coding process at a relay node is a little different, since the correctly received symbols may be in positions not consecutive due to packet corruptions. To reduce the overhead incurred by potential multiple coding vectors, we adopt piece-division, run-length coding algorithm [66], where consecutive clean symbols share a coding vector. Due to the space limitations, we will not go into details about this algorithm and interested readers are referred to our previous work [66]. By using SLNC, the bandwidth efficiency of each coded transmission could be improved.

Each receiver v maintains a *playback buffer* for generations to be played in the immediate future, which buffers all the received useful coded symbols. Note that v also maintains a decoding matrix for each symbol position j of each generation, which consists of the coding vectors of all the j^{th} symbols it received currently. The rank of each matrix is called *symbol rank*. A coded symbol is called *useful* in CodePlay if: i) it is received correctly [54]; ii) it can increase the corresponding symbol rank (*innovative*); iii) it belongs to a generation that is after v 's current playing point. When receiving enough useful symbols for a position, the receiver can decode the original symbols by performing Gaussian elimination on the corresponding matrix.

For nodes to make decisions on transmission, coding and coordination, every node needs to disseminate its *reception status* to neighbors, i.e., symbol rank of each useful generation. Although this information is piggybacked on periodical safety messages which adds no extra overhead to the service channel, we still should keep it minimal to reduce its impact on the reliability of safety messages. Here is a back-of-the-envelope calculation: for 10 generations with packet length of 30 symbols, the piggybacked information is 300 bytes, which is obviously too long for a safety message that is usually several few hundreds bytes long. To decrease the size of the piggybacked information, we use the fuzzy *average rank*. That is, for generation G_i , an average rank value $\lfloor \bar{r}_i \rfloor$ across all symbol positions is computed and transmitted. Now, the piggybacked information is only 10 bytes, which can be easily embedded in a safety message without affecting its reliability [117].

Each node plays the buffered generations sequentially and keeps eliminating older generations to make room for newer content. Those generations within α seconds after the current playback time is called *priority generations*. The piggybacked reception status which contains a priority generation with average rank less than K is considered as an implicit *urgent request*. The above definitions are depicted in Fig. 4.4. Note that vehicles on the opposite road of the AoI behave exactly the same as described above, except that they do not need to playback the received LMS content.

4.4.4 Coordinated and Distributed Relay Selection

The main purpose of the relay selection is to maximize the utility of each transmission to save the precious bandwidth resource in the VANET. A selected relay should best satisfy all its neighbors' smooth playback needs, which can be inferred through those vehicles' reception statuses. Here three components are needed: i) a local coordinator that serve as an arbitrator, with which a consensus on relay selection can be reliably

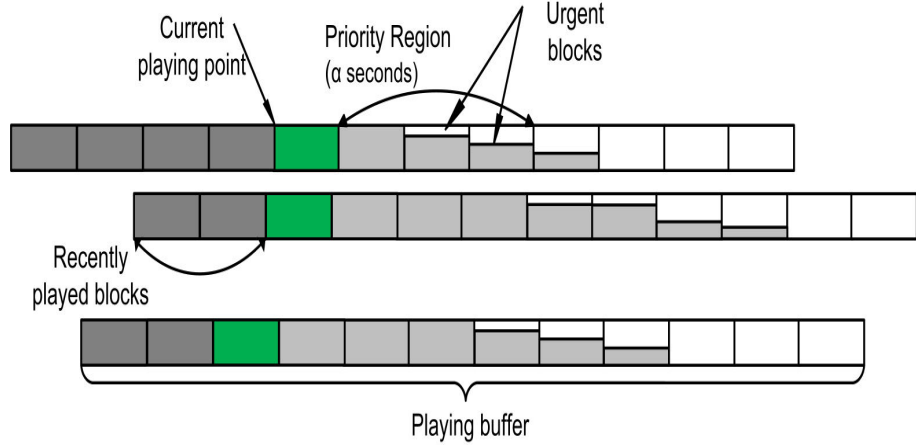


Figure 4.4: playback buffer and priority generations.

and efficiently achieved; ii) the computation of nodes' "utilities" that represents their capability to satisfy others; iii) The selection of appropriate parameters (such as segment length (SL), etc), for fast LMS propagation and continuous coverage.

4.4.4.1 Distributed Coordinator Selection

All vehicles in the same road segment agree on an unique local coordinator at the end of each control time slot, based on geographic information. For both reliability and efficiency considerations, we propose an accumulated consensus mechanism based on information piggybacked in the safety messages. We firstly define a *temporary coordinator* as the vehicle closest to the segment center that a vehicle currently knows. Each vehicle considers itself as the default temporary coordinator at the beginning of each control time slot. For each overheard safety message originated from a vehicle in the same segment, the receiver checks if the temporary coordinator piggybacked (Fig. 4.5) is closer to the segment center than the one known to itself presently. If yes, the receiver replaces its temporary coordinator with the overheard one. Since the vehicle closest to the segment center will be repeatedly claimed as temporary coordinator by multiple safety messages (like vehicle B in Fig. 4.3 (a)),

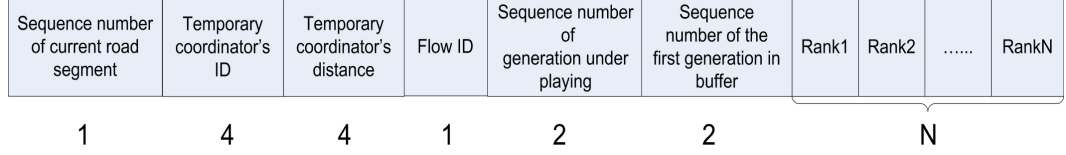


Figure 4.5: The format of piggybacked information (in Byte), where N is the size of the playback buffer (in generation).

this accumulated consensus mechanism makes the probability of selecting multiple coordinators within one segment negligible, no matter there are lossy wireless links or sparse connections.

4.4.4.2 Relay Selection

At the beginning of the following service time slot, each coordinator C firstly checks if its segment is scheduled to transmit in this slot or not, where the scheduling algorithm will be introduced in the next section. If yes, C will then calculate the *node utility* for each vehicle in $\mathcal{V}(C)$, the set of all the vehicles in the same segment as C , and designate the one with the highest utility as relay. If a tie appears, the vehicle located in the LMS propagation direction wins. The calculation of node utility consists of two steps:

i) Find the range of *interested generations* for all vehicles in $\mathcal{N}(C)$, which is the neighbor set of C and we require $\mathcal{N}(C) \supseteq \mathcal{V}(C)$. Only the generations representing streaming contents after the earliest playback time among vehicles in $\mathcal{N}(C)$ are regarded as interested ones. If there exists some urgent generations $UrgentGen$, C will give strict priority to the transmission of $UrgentGen$ during this time slot to ensure smooth playback at those vehicles. Otherwise, all the interested generations will be considered by C .

ii) Calculate node utility for each vehicle in $\mathcal{V}(C)$. If $UrgentGen \neq \emptyset$, only the

generations in it will be considered in this calculation. With SLNC, the usefulness of a potential relay v 's generation G_i is determined by the difference in the symbols' ranks of G_i between v and its neighbors. Due to wireless medium's broadcast nature, G_i 's utility to others increases with both the average usefulness of G_i and the number of vehicles it can benefit. Thus, for $v \in \mathcal{V}(C)$, the *generation utility* of G_i is defined as:

$$U(G_i, v) = \sum_{v' \in \mathcal{N}(v)} \text{Step}(\lfloor \bar{r}_{v,i} \rfloor - \lfloor \bar{r}_{v',i} \rfloor) \times \text{Urgent}(G_i, v') \quad (4.2)$$

where $\lfloor \bar{r}_{v,i} \rfloor$ is the fussy average rank of node v 's generation i . $\text{Step}(x) = x$, if $x > 0$; otherwise, $\text{Step}(x) = 0$. And $\text{Urgent}(G_i, v') = \text{priValue}$, if G_i is urgently requested by vehicle v' , otherwise, $\text{Urgent}(G_i, v') = \frac{\text{priValue}}{2^{i-i_0}}$, where i_0 is the index of the urgent generation closest to the physical world's time. The *priValue* is an adjustable system parameter which controls the relative importance of priority generations. Note that, since the coordinator does not know $\mathcal{N}(v)$ under the single-hop piggyback mechanism, we substitute $\mathcal{N}(v)$ by $\mathcal{N}(v) \cap \mathcal{N}(C)$. In fact, if we assume the safety messages are sent at the basic rate which can reach larger range (e.g. $2\times$) than normal data packets, then $\mathcal{N}(v)$ can be further reduced to nodes within v 's data communication range ($\mathcal{N}'(v)$) (which will be explained later), which can be estimated by C .

This utility measures how much innovative information node v can give to other vehicles in $\mathcal{V}(c)$ in total if it broadcast coded packets generated from G_i . Currently we do not consider the link qualities between v and the receivers. The *node utility* $U(v)$ of vehicular node v is defined as $\max_{G_i \in \text{interested generations}} \{U(G_i, v)\}$, which estimates the maximum amount of innovative information v can provide to other vehicles in $\mathcal{N}(v)$ for one generation. We do not look at the aggregate utility of multiple generations, since transmitting many generations takes a long time which may cross multiple time slots and the VANET topology has already changed.

The coordinator C designates R , the vehicle having the maximum $U(R)$, as the relay using a unicast message, which enables R to use the current service time slot. R then actively pushes coded packets generated from G_R with the maximum $U(G_R, R)$. Note that, the required number of coded pieces to send during one service time slot can be estimated based on $\frac{1}{|\mathcal{N}(R)|} \sum_{v' \in \mathcal{N}(R)} \text{Step}(\lfloor \bar{r}_{R,i} \rfloor - \lfloor \bar{r}_{v',i} \rfloor)$, which will not be elaborated here.

4.4.4.3 Determining the Segment Length

The length of the segment, SL , is an important parameter that affects the utility of relay selection and propagation speed of the LMS flow. On the one hand, if SL is too large, a relay at one end of a segment may not convey enough information to the neighboring segment in its scheduled time slot, and in the next slot the relay in the neighboring segment would have few innovative information to transmit, which affects smooth playback of LMS. On the other hand, if SL is too small, vehicles in adjacent segments tend to have similar reception statuses and their relays probably will transmit duplicate information. Both extremes could lead to low bandwidth efficiency and large service delivery delay.

In general, we should ensure that for a pair of sender and receiver of distance SL , the symbol reception probability is sufficiently high. However, under realistic fading channel, it is hard to define such a range since symbol reception is probabilistic. For a simpler alternative approach, we define an equivalent *data communication range* CR under free space propagation model(Friis)³, $CR = \sqrt{\frac{T_p G}{Th_{CR}}}$, where T_p is the transmission power, G is the antenna gain and Th_{CR} is the data reception threshold. Thus we set $SL \approx CR$ in this chapter.

³Although this range is originally defined for packet reception in 802.11p standards, it is also a meaningful approximation for symbol reception.

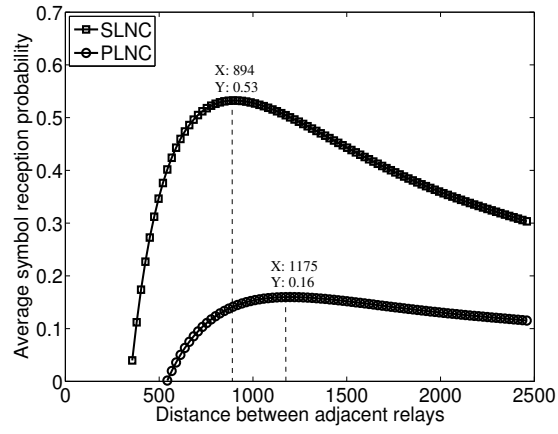


Figure 4.6: The average symbol reception probability when CR=277m, ER=700m. Data rate is 12Mbps, and Nakagami fading model is used.

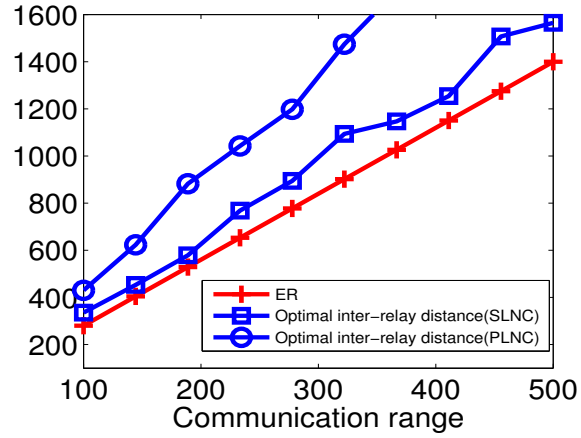


Figure 4.7: The optimal distance between two adjacent relays under various data communication ranges.

4.4.5 Transmission Coordination of Relays

We have determined which vehicles should transmit what content to whom. In this section, we answer the last question: in which time slots should each relay actively push the coded LMS? This is addressed from both spatial and temporal aspects.

4.4.5.1 Spatial Coordination

Due to the use of SLNC, concurrent transmissions of more relays are encouraged to take advantage of spatial reusability [54]. But two transmitting relays that are too close will cause heavy collisions which in turn degrades the bandwidth efficiency. There exists an optimal average distance between two concurrent transmitting relays, D_{opt} , under which the relays can convey highest amount of useful information to their neighbors within unit time. In other words, the bandwidth can be used the most efficiently.

Next we discuss how to determine the D_{opt} . First we define “optimal inter relay distance”. Consider a straight highway of length L , where vehicles are uniformly distributed. n relays, v_1, v_2, \dots, v_n , lie on the highway with equal inter-distance. All the relays simultaneously and continuously transmit coded streaming content to other vehicles, and each symbol is assumed to be useful if it is correctly received. The *average symbol reception probability* Pr_{avg} for all the vehicles in the VANET is defined as: the average probability that each vehicle receives one symbol from any of the n relays during the period of one symbol’s transmission. We assume a vehicle cannot receive more than one symbol at the same time. The inter-relay distance is considered as D_{opt} if the achieved Pr_{avg} is maximized.

Under wireless propagation models with channel fading (such as Nakagami model),

it is very hard to derive a closed form solution for Pr_{avg} . Therefore, we approximate Pr_{avg} by Monte-Carlo simulations⁴:

$$Pr_{avg} = \frac{\text{Total \# of symbols correctly received by all vehicles}}{\text{Total \# of symbols sent by all relays} \times \text{total \# of receivers}} \quad (4.3)$$

where for each receiver, the received signal to noise ratio (SNR) is randomly sampled from the propagation model. We generate 10 random topologies on a highway of $10km$ with 1000 vehicles and all the relays transmit 100 pieces simultaneously, each containing 30 symbols. The value of n varies according to $\frac{L}{d}$. We also evaluate the performance of PLNC under the same setting, where a packet is considered as correctly received if all of its symbols are correctly received. The results are given in Fig. 4.6 and Fig. 4.7.

We can see that when $CR = 277m$, for SLNC, $D_{opt} \approx 900m$, under which Pr_{avg} is above 0.5; if PLNC is applied, $D_{opt} \approx 1200m$, under which $Pr_{avg} \approx 0.16$. This confirms that SLNC tolerates transmission errors better than PLNC, which allows more aggressive concurrent transmissions and achieves higher bandwidth efficiency. Similar conclusions could be found under other CR values, which is omitted here.

In addition, SLNC's shorter D_{opt} simplifies protocol design. In Fig.4.7, one can observe that SLNC's D_{opt} is quite close to *energy detection range* ER under various communication ranges, especially when the CR is relatively short. ER is again an equivalent range defined under free space propagation model(Friis). The implications are that, by adopting SLNC, CodePlay can make the channel access decisions largely based on simple carrier sense mechanism. However, this is not the case for protocols adopting PLNC, which must deal with the well-known hidden terminal problem [120]. We will exploit this benefit of SLNC in the opportunistic scheduling algorithm in the next section.

⁴The details are omitted due to space limitations. Please refer to [66].

4.4.5.2 Temporal Coordination

To provide continuous streaming coverage and to satisfy the strict time constraint of LMS services, the traditional random medium access mechanisms are not appropriate since their channel access delays are not bounded [98]. We propose to use local round-robin (LRR) scheduling to coordinate the transmissions of neighboring relays. Since it is impossible to know the inter-relay distance before those relays actually transmit, in practice, we convert D_{opt} into the number of separating segments (W_{opt}) between two adjacent concurrent transmitting relays. The observation is that, relays selected from one segment will tend to be uniformly distributed in it over time, and their average location is the segment center. As an approximation, we have $W_{opt} \times SL < D_{opt} < (W_{opt} + 1) \times SL$, therefore $W_{opt} = \lfloor \frac{D_{opt}}{SL} \rfloor$. The round length R in LRR is exactly $W_{opt} + 1$. For a relay in segment i , its scheduled slots T_i are determined as: $T_i \equiv i \bmod (W_{opt} + 1)$. For example, assume $W_{opt} = 2$, then segment 1 is scheduled to use time slots 1, 4, 7, 10, etc. Using this local round-robin schedule, LMS can flow from the source to receivers within the AoI smoothly. From a receiver's point of view, if the VANET is well-connected, it is always able to obtain new LMS content for playback within determined waiting time.

4.4.6 OLRR: Opportunistic LRR Scheduling for Sparse VANETs

Due to the highly dynamic nature of VANET, it tends to experience partitions frequently [87], especially when the traffic density is low. In sparse VANET, some road segments will be devoid of relays and the scheduled transmission opportunities would be wasted if the original LRR is adopted, which results in low bandwidth efficiency. This could be illustrated in Fig. 4.8(a), where the segments 4,7,10 contain no vehicles, and the scheduled transmission opportunities in this time slot for them are

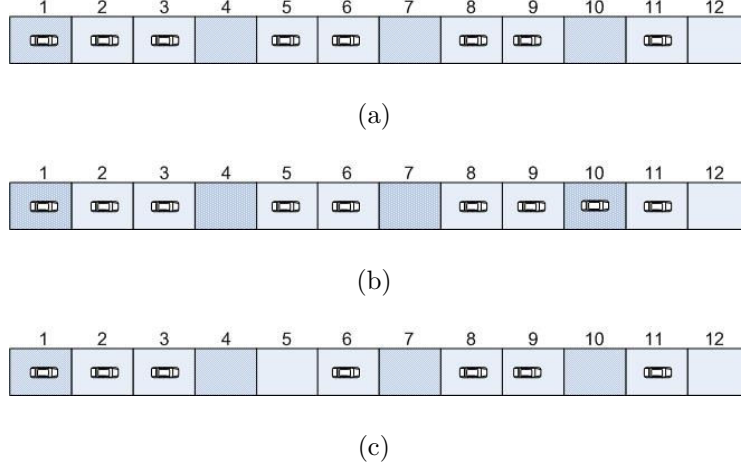


Figure 4.8: Snapshots of 3 sparse VANETs ($T=1$, $R=3$). The road segment ID is illustrated above each road segment and the vehicles represent corresponding coordinators. Those dark shaded segments in each snapshot are designated to be scheduled in this time slot.

wasted. To solve this problem, we propose an opportunistic LRR (OLRR) scheduling algorithm by taking advantage of those available slots.

The OLRR operates in a way resembling cognitive radio [37], which leverages nodes' capability of carrier sensing. Essentially, during each service time slot, the coordinators in each segment will detect if there are relays in the nearby “*primary segments*”, which are scheduled segments by LRR in that time slot, like segments 1, 4, 7 and 10 in time slot 1. (Fig. 4.8(a)). If not, certain *secondary segments* will gain channel access according to some priority assignment. Thanks to SLNC, each coordinator/relay does not need to consider the transmitters out of its energy detection capability, which greatly simplifies protocol design.

For enabling coordinators to sense the relay selection statuses of their neighboring road segments reliably and efficiently, we use a small period of time at the very begin of each service slot and divide it into $3 \times (W_{opt} + 1)$ subslots. Coordinator of road segment i that could find a relay will broadcast a short signal during the subslot

$i \bmod 3 \times (W_{opt} + 1)$ to notify other neighboring coordinators which will keep sensing the channel during those subslots. We note that the actual data transmissions start after those subslots. The reason we need $3 \times (W_{opt} + 1)$ subslots is to ensure that each coordinator will be able to determine a unique segment (w.h.p) that is transmitting in each subslot (cause for each subslot, there could be 2 possible notification signals broadcasted from both side). Since the sensing process is purely based on detecting the energy, the time overhead can be negligible. In CodePlay, we set the sensing signal length to be 50 bytes and the length of each sub-slot to be $100 \mu s$, which takes preamble, SIFS, etc. into consideration. For $W_{opt} = 2$, the total extra time is $3 \times (2 + 1) \times 100 = 900 \mu s$, which is less than 2% of a service time slot with length of $50ms$.

The algorithm is described in Alg. 1. In line 3, there are two cases where a relay cannot be selected: C_i is the only node in i , or no node can provide innovative information to others. $ConflictSet(i)$ is the set of coordinators (also segments) that has higher transmission priority than i . The nearer a segment is to a primary segment (with lower ID), the higher its priority. If two secondary segments happen to have the same distance to their primary segments, they will both access the channel as is the case in LRR.

We use the examples in Fig. 4.8 to illustrate the basic idea of OLRR. Suppose $R = 3$ and C_1, C_4, C_7, C_{10} are scheduled to use the channel simultaneously in the current service time slot $T = 1$. In Fig. 4.8(a), If we apply original LRR, only C_1 will use this service time slot. If we apply OLRR, C_5 will decide to take this time slot since it senses that C_4 and C_7 do not exist. The same for C_8 and C_{11} and thus this service time slot will be consumed by C_1, C_5, C_8 and C_{11} simultaneously which obviously improved the spatial reusability. For VANET snapshot shown in Fig. 4.8(b), If OLRR is adopted, although C_8 will give up this opportunity, since otherwise it will

Algorithm 1 Opportunistic LRR scheduling at each coordinator (at the beginning of a service channel slot)

- 1: **Input:** Segment ID i , coordinator C_i , round length $R = W_{opt} + 1$
 - 2: **Output:** Whether to allow the relay access channel
 - 3: If C_i is able to select a relay from i
 - 4: Broadcast a short signal in the subslot $i' \leftarrow i \bmod 3R$
 $ConflictSet(i) \leftarrow \emptyset$
 - 5: For subslot j' from 0 to $3R - 1$ //determine which segments have relays
 - 6: If sensed signal during j'
 - 7: $ConflictSet(i) \leftarrow ConflictSet(i) \cup C_{j'}, C_{j'} \in Segment\ j$,
 where $Segment\ j$ is the nearest one to i between the two:
 $j' + i - i'$ and $j' + i - i' \pm 3R$ //the most probable segment
 - 8: Prune from $ConflictSet(i)$ the segments that are more than R segments away from i //re-
 garded as not conflicting
 - 9: Prune from $ConflictSet(i)$ segments j with $j \bmod R > i \bmod R$ //the one nearer to a
 primary segment has higher priority
 - 10: If $ConflictSet(i) \neq \emptyset$
 - 11: C_i tells relay in i to abort transmission
 - 12: Else, C_i tells relay in i to access the channel in current service time slot
-

incur unnecessary heavy interference to the transmission of C_{10} , C_5 still could use this service time slot along with C_1 and C_{10} . The operation of OLRR under the situation shown in Fig. 4.8(c) is a little more complicated. Now both *secondary segments* C_6 and C_8 will try to take the extra transmission opportunities left by empty segments 4 and 7 respectively. To avoid heavy collision between them, OLRR assigns each secondary segment a priority based on its distance to the primary segment with lower ID. In this case, C_6 is two segment away from the primary segment 4 and C_8 is only 1 segment away from the primary segment 7, thus C_8 , which has higher priority, will take this transmitting opportunity and C_6 will keep silent during this service time slot.

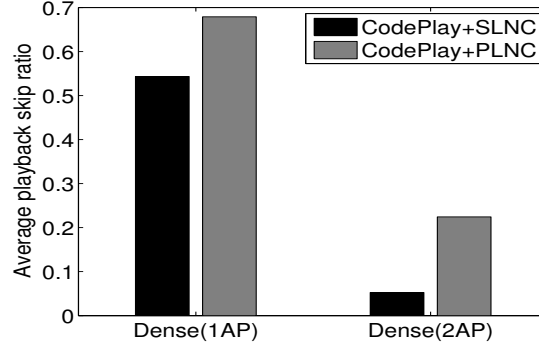
Table 4.1: Parameter Settings

Data rates for LMS and safety msg.	12Mbps, 3Mbps
Data communication range	$CR = 250\text{m}$
Time per generation, piece size	2s, 1KB
Safety message length (with piggyback)	130B
Buffer capacity	15 generations
$PriValue$	32
# of generations in priority region	$\alpha = 1$
Default D_{opt} for CdePlay+SLNC	900m
Default D_{opt} for CdePlay+PLNC	1200m

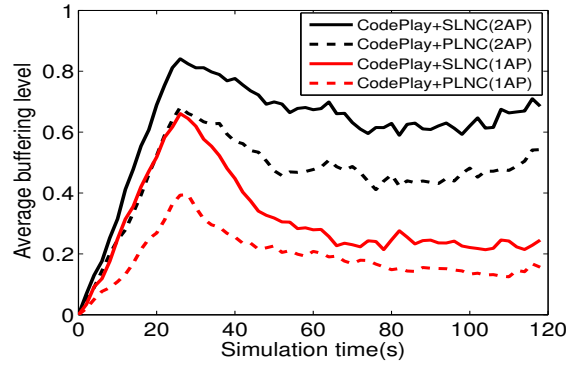
4.5 Performance Evaluation

We implemented and evaluated CodePlay by simulations using NS-2.34. The SLNC is implemented based on [54], with an enhanced run-length coding technique [66] which is more suitable for consecutively broadcasting a generation of coded pieces in CodePlay. To ensure unique coordinator selection within the same segment, at the beginning of service time slots use an additional broadcast round (shorter than 1ms) to resolve collisions between potential coordinators. The simulation scenario consists of a straight 4-lane highway with length of 3000 meters, and one or two LMS source(s) (e.g., access points) can be located at one or either ends of the highway. The upper part of the highway (west bound) is regarded as the AoI. We simulate both dense and sparse VANETs by using two traffic densities: 100 cars/km and 40 cars/km. The vehicular speeds are randomly selected from 20-30 m/s. The simulation parameters are shown in Table I.

The protocol for comparison is the PLNC version of CodePlay (CodePlay+PLNC)



(a)



(b)

Figure 4.9: Comparison between using one and two APs, dense highway, source rate=12KB/s, initial buffering delay=24Sec.

and the W_{opt} for PLNC is used. By default, the OLRR is applied for all the protocols. State-of-the-art LMS scheme to ours is emergency video dissemination in VANETs using PLNC (NCDD, [87]). However NCDD was not designed to meet the practical application layer requirements defined in this chapter, and it is hard to evaluate those metrics based on NCDD protocol. Thus we chose not to implement NCDD, but compare our results with the reported ones in [87]. Each point shown in the simulation results is averaged over 10 runs.

The performance of CodePlay is evaluated by multiple metrics: (1) Initial buffer-

ing delay, which is the user experienced service delay. In the simulation, we impose the same initial buffering delay for all receiving vehicles. (2) Source rate, which reflects the supported LMS generation rate from the application layer. (3) Skip ratio, the fraction of generations skipped due to incomplete reception before playback time over all the generations that are played. We note that this will make the skip ratio looks higher since any missing bit will render the whole generation useless. However, in practice, the LMS content usually does not need to achieve 100% reception ratio for playback. The playback video quality will gradually improved as the increase of the reception ratio by applying advanced video coding methods like multi-layer coding [32, 110] and multi-description coding [93]. Since the adoption of those advanced video coding approaches are orthogonal to our work, we just use this all or nothing rules for simplicity in this chapter. (4) Buffering level, the percentage of the buffered LMS contents between current playback time and physical world time. Both the skip ratio and buffering level could reflect the playback quality, i.e., smoothness [106].

Effects of the following factors are also considered: the number of LMS sources (APs), vehicle traffic density, and whether LRR/OLRR is enabled.

4.5.1 Effect of Number of LMS Sources

We first consider how the LMS performance is affected by the number of sources (AP), i.e., only one AP which is placed on one end of the highway, or two at both ends of it. Our main finding is that, the two-source case significantly outperforms the single-source case. Fig. 4.9 shows the difference between using one and two APs under the dense highway. Both protocols, CodePlay+SLNC and CodePlay+PLNC perform much better under the two-AP case than the single-AP case. Another observation is that CodePlay+PLNC can not work well even in the two-AP case, the skip ratio of which is as high as 22%. However, the adoption of SLNC can reduce the skip ratio

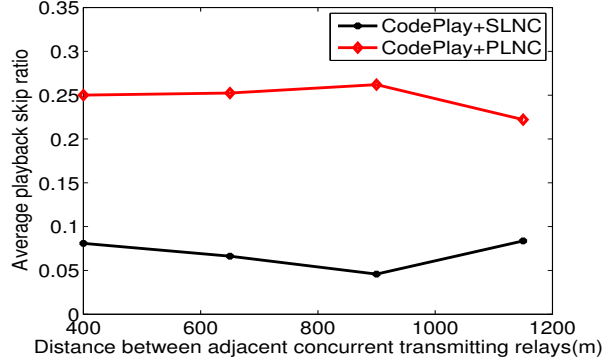


Figure 4.10: Comparison between different distance of adjacent concurrent transmitting relays, dense highway, source rate=12KB/s, initial buffering delay=24Sec.

to less than 5%, which enables a much better playback experience.

This can be explained as follows. Because of lossy wireless links, a single flow is not able to sustain smooth playback of the LMS content after traversing a large number of hops in the VANET, which is also in line with the conclusion of throughput degradation after multi-hops in wireless networks [44]. For two crossing flows with the same content, the packet losses are compensated by innovative symbols/packets from both directions. This can also be proved by the higher buffering levels in the two-AP case shown in Fig. 4.9(b). Therefore, in the following we evaluate CodePlay based on the two-AP case. The conclusions for sparse highway scenario is similar, where the average skip ratio is 10% and 14% for CodePlay+SLNC and CodePlay+PLNC, respectively under single AP and all equals 0 under the two-AP case.

4.5.2 Effect of D_{opt}

To see the impact of different D_{opt} on the performance, we run the protocols with varying optimal distances between adjacent concurrent transmitting relays under dense highway scenario, specifically, 400m, 650m, 900m and 1200m are used. Since the

segment length $SL = 250m$ in our simulations, these D_{opt} values represents the cases where the length of the round robin are 2, 3, 4 and 5 respectively. The result is shown in Fig. 4.10. We can see that the playback skip ratio for CodePlay+SLNC gradually decreases as the D_{opt} increased from 400m and achieves the minimum when the D_{opt} equals 900m, which also stands for the optimal playback video quality. After that, the skip ratio goes up again as the D_{opt} increased. For CodePlay+PLNC, the average playback skip ratio decreases continuously as the increase of the D_{opt} . This is consistent with our observation in Fig 4.6, which shows that the optimal distance for PLNC case is 1200m. Besides, we also can see that under $D_{opt} = 1200m$, Although the skip ratio of CodePlay+SLNC increases from 5% to about 8%, it is still much better than the corresponding performance of CodePlay+PLNC, which again demonstrate the benefits of SLNC over PLNC.

4.5.3 Effect of Initial Buffering Delay

To further illustrate the advantage of CodePlay in providing better LMS services under various VANET scenarios, we investigate the relationship between initial buffering delay, source rate and the metrics for smooth playback under a relatively sparse highway scenario. In the first simulation set, we fix initial buffering delay as 16 seconds, and increases the source rate from 24 KB/s to 36 KB/s. The results are presented in Fig. 4.11. We can see that the skip ratio for CodePlay+SLNC is much lower than its PLNC based opponent, where the former's skip ratio is 0 under 24 KB/s, 5.3% under 30 KB/s and 15% under 36 KB/s. This suggest that rate higher than no greater than 30KB/s could be supported without affecting smooth playback. Also, for each rate CodePlay+PLNC's buffering level decreases faster over time, and is less stable compared with that of CodePlay+SLNC. This reflects that CodePlay+SLNC achieves a more stable flow of multimedia streaming, which shows the effectiveness of

the integration of SLNC with the coordinated local push mechanism. We note that, the NCDD protocol only provided 10 KB/s source rate for video dissemination [87].

Another interesting observation when we look into details of the performance is the changing of the average reception ratio for those skipped generations, which is defined as the ratio between the average symbol rank over all symbol positions within the generation and the generation size. The result is shown in Fig 4.11(c). We can see that the CodePlay+SLNC not only achieves lower skip ratios, but also achieves much higher average reception ratio compared with CodePlay+PLNC. For example, the average reception ratio for the former is always higher than 90% while that of the CodePlay+PLNC is no greater than 60% for both 30KB and 36KB/s cases (in the 24KB/s case, the reception ratio for CodePlay+SLNC is 0 due to no skipped generation existed). This means those skipped generations in CodePlay+SLNC contain more useful information and if advanced video quality criterion [32,93] rather than the simple all-or-nothing policy is adopted, the advantage of SLNC based scheme will be enhanced. We also notice that as the increase of the source rate, the average reception ratio for CodePlay+PLNC gradually decreases while that of the CodePlay+SLNC is relatively stable. This is because that higher source rate bring more communication and thus incur more transmitting contention into the network, which will significantly affect the probability of correct reception of the whole packet. On the other hand, the SLNC based scheme will accumulate all the correctly received symbols even for an error packet thus the impact of the higher contention is greatly alleviated.

In the second simulation set, we fix the source rate as 24KB/s and 30 KB/s, then increase the initial buffering delay from 16 to 24 seconds, respectively. From Fig. 4.12, we can see an obvious reduction in the skip ratio and an increase in the buffering level for both protocols under both rates. This result is consistent with intuitions, and implies that initial buffering delay plays an important role in VANET

LMS services.

The CodePlay+SLNC works well through all source rates no greater than 30 KB/s, and for buffering delays of 16s and 24s. We argue that those delays are acceptable in VANETs. For example, for delay equals to 16s and vehicular velocity of 30m/s, a car will travel about 500m after it enters the AoI to begin playing an emergency multimedia content. For $L = 3000\text{m}$, the car will be at 2500m from the accident spot and may still have enough time to take actions.

4.5.4 Effect of Traffic Density

Next we study the performance of CodePlay under the dense traffic condition. Fig. 4.13 shows the whole set of simulation results with various source rates. Though CodePlay+SLNC still outperforms CodePlay+PLNC, compared with the sparse case, the skip ratio of both protocols are higher and corresponding buffering levels are lower. Especially, only the skip ratio of CodePlay+SLNC under 12KB/s could be kept lower than 5%, and the skip ratios of all the other cases are higher than 18% which could be unacceptable from receivers' point of view. The worse performance can be mainly ascribed to the limitations in the node utility functions, which is directly associated with how much innovative information a relay can deliver to all neighboring nodes. For broadcasting in a dense VANET, since there could be too many vehicles urgently demanding different portions of the LMS content, it is intrinsically hard to satisfy all their needs in a short time. Due to the time constraints of LMS applications, this leads to more frequent playback skips than in the sparse VANETs.

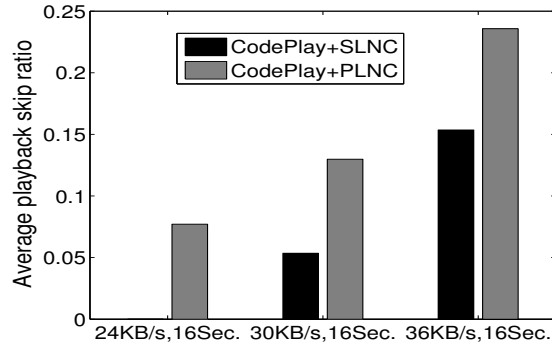
4.5.5 Effect of Opportunistic Scheduling

In the previous simulations, we have the OLR scheduling enabled by default. Yet it is interesting to see how the opportunistic scheduling affects the protocol performance.

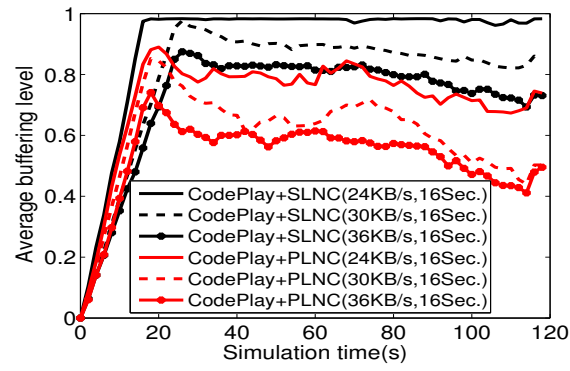
Thus, we presented in Fig. 4.14 the results of enabling and disabling the OLRR algorithm (using LRR instead). All the protocols run with source rate of 12 KB/s under dense network and 24 KB/s under sparse network, the initial buffering delays of both are 16 Sec. We can see that the OLRR greatly improves the performance over the basic LRR algorithm for all the running cases. By opportunistically utilizing the idle scheduled transmission slots left by primary segments, the OLRR can adaptively “fill” the unnecessary gaps created during the propagation of the LMS flow. We note that OLRR could take effect not only under sparse network, but also under dense network. The reason is that according to [112], the vehicles running on the highway tends to form disjoint clusters rather than uniformly distributed even under relatively dense traffic.

4.6 Summary

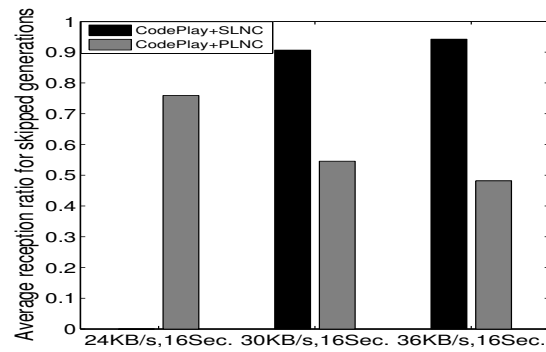
In this chapter, we presented the design and performance evaluation of CodePlay for live multimedia streaming in the dynamic and lossy VANETs. Multiple objectives are pursued at the same time, including short buffering delay, smooth playback, and high source rate. The core of CodePlay is a coordinated local push mechanism with symbol level network coding, which establishes local and distributed coordination among vehicles to ensure stable and high streaming rates. Through the above mechanisms, the benefits of SLNC is fully exploited for better LMS performance in VANET. Our main conclusions in this chapter is that symbol-level network coding is a good technique to support bandwidth consuming, delay constrained LMS applications in extreme environmental like VAENTs. Even using SLNC, we may still need the help of few additional infrastructure (APs) along the road and well designed channel usage mechanisms to facilitate the dissemination of LMS content to end users.



(a)

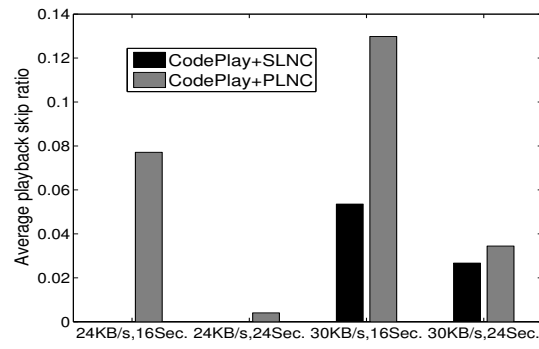


(b)

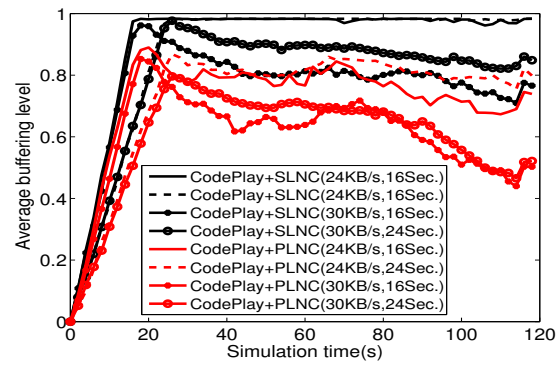


(c)

Figure 4.11: Fixed initial buffering delay, varying source rates. Sparse highway.

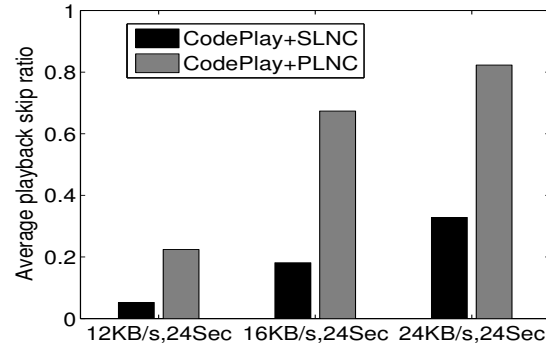


(a)

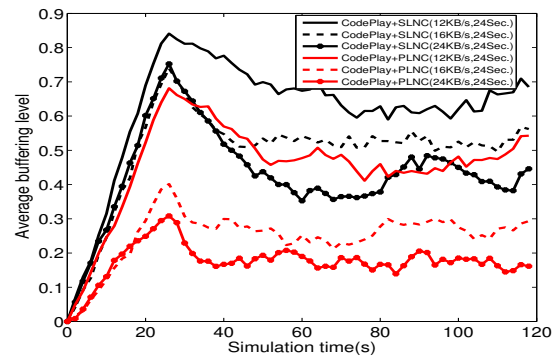


(b)

Figure 4.12: Fixed rate, varying initial buffering delay. Sparse highway.

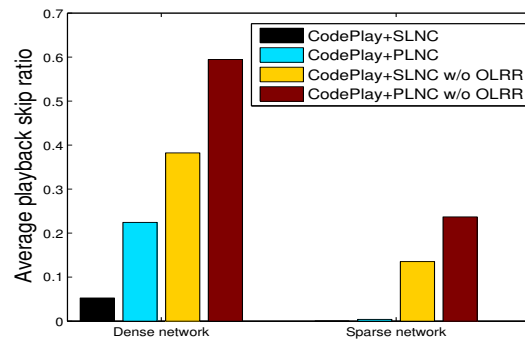


(a)



(b)

Figure 4.13: Impact of traffic density. Dense highway.



(a)

Figure 4.14: Effect of opportunistic transmission scheduling.

Chapter 5

Throughput Analysis of Cooperative Mobile Content Distribution in VANETs using SLNC

5.1 Introduction

In the previous two chapters, we describe the design of two SLNC-based reliable content distribution schemes in details. In this chapter, we endeavor to carry out more in-depth study about those SLNC-based mobile content distribution (MCD) schemes from theoretical point of view. We are interested in answering two questions: 1) Regarding realistic issues of channel fading, transmission interference and node distribution, how does the expected achievable throughput at a node changes with its distance from the AP? 2) Given a specific vehicle mobility pattern, what is the

expected downloading volume (in bytes) a vehicle can obtain from the AP during the time period it drives through the AoI? The results can provide valuable guidelines to optimize the source broadcast rate and plan the access point deployment.

The main contributions of this chapter can be summarized as follows:

(1) We propose an analytical model to compute the achievable throughput of single-flow unicast and multicast using SLNC in general wireless networks, based on network flow and queueing theory, given a static network topology. We show that the gain of achievable throughput of SLNC over PLNC comes from the symbol-level diversity.

(2) We apply the above method to model the throughput of cooperative MCD system with SLNC in VANETs. Under a static snapshot of VANET topology instance, given a channel fading model and a medium access control scheme, the achievable throughput of SLNC in stable conditions for a vehicle at a certain distance d from the AP can be derived. We compute the per-link average rates with saturated transmissions, which represents stable conditions, based on the concept of “communication density” [46], which is the product of vehicle density, packet injection rate and communication range. The expected achievable throughput for that distance is obtained via averaging among all topological instances following the same vehicle distribution. Furthermore, the expected downloading volume of a vehicle can be obtained by aggregating the distance-limited achievable throughput over its distance to the source with the vehicular mobility pattern, which captures the dynamic property of VANET.

(3) We demonstrate numerical results from our model, and obtain several interesting results. In summary, the gain of SLNC over PLNC in achievable throughput increases when the channel turns from slow fading to fast fading. For a given NC technique, the distance-limited achievable throughput is mainly impacted by the interplay

between the network connectivity and interference level. The former is in turn determined by inter-vehicle distance distribution, where a distribution with larger variance yields a faster decrease of throughput with distance. When the vehicle density increases, the distance-limited achievable throughput at nearer hops from the source decreases due to heavier interference, but it decreases slower with the distance (or number of hops) due to better overall connectivity. Our findings provide insights on optimized choices for cooperative MCD system design in VANETs, which has not been enabled by previous theoretical works.

The remainder of this chapter is organized as follows. In Section II, we discuss the related works. Section III describes SLNC system model and gives an analytical model to compute achievable throughput of SLNC in generic wireless network, and then a theoretical framework of deriving achievable throughput of cooperative MCD system is presented in Section IV. Section V contains the numerical results and discussion. Finally, we summary this chapter in Section VI.

5.2 Related Work

5.2.1 Capacity Scaling Law of Wireless Networks

The seminal work of Gupta and Kumar studied the unicast capacity of wireless networks. They showed that the per node throughput of multi-pair unicast traffic scales as $\lambda(n) = \Theta(1/\sqrt{n \log n})$. For broadcast, [102] was the first to show that the broadcast capacity is bounded by $O(c/n)$. Li et al. [69] investigated the multicast capacity in a random wireless network within a square region and derived an asymptotic upper bound and lower bound. Zheng et al. [108] generalized all types of information dissemination and proposed a (n, m, k) -casting as a framework to study the capacity

problems.

Following the introduction of the network coding (NC) concept by [6], there have been lots of research on the capacity scaling of wireless network with NC. NC has been shown to be able to achieve the multicast capacity of a wired network [6], while its capacity has no order difference with not using NC under both unicast and broadcast traffic [70, 71]. Karande et al. [51] derived a tight bound for multicast per-session capacity with NC and demonstrated the multicast throughput capacity with NC is also bounded by a constant factor. However, the above works mostly focus on using the information-theoretic capacity definition, which may not result in much practical insight for MCD in VANET.

A similar approach reveals the expected capacity of NC in a wireless network via the “asymptotic throughput”. [92] used a weighted random geometric graph to model a dense wireless network. They showed that the network throughput converges asymptotically to $N \cdot E[C_{ij}]$ when $N \rightarrow \infty$ is the number of nodes in the network, and C_{ij} is the capacity of link $i \rightarrow j$ which is independent with each other [35]. [58] Further considered a realistic dense network model considering interference and noise, and provided a similar result as the above work. In contrast, in this chapter we consider an extended network model with constant node density, which is more realistic for the VANET.

Lun et. al. [72] proposed a theoretical model to compute the exact capacity region of random linear network coding in both wired and wireless networks. They prove a general result that NC can achieve the max-flow min-cut capacity in a network for packet arrivals with average rates, by modeling the flow of packets in the network as the propagation of jobs through a queueing network. This is the most related work in terms of technical approach, but we have several key differences: 1) we model the achievable throughput of SLNC instead of PLNC; 2) we provide a method to derive

the achievable throughput for MCD in VANETs under practical conditions including channel fading, medium contention, symbol correlation and vehicle distribution; 3) from our results we give several insights on the design of cooperative MCD systems in VANET.

5.2.2 Achievable Throughput of MCD in VANETs

The first cooperative MCD strategy in vehicular network “SPAWN” is proposed by Nandan et al. [80]. After that, network coding [33] [63] is adopted in cooperative MCD, because of its ability to exploit the broadcast nature of wireless network and enhance the bandwidth efficiency. Recently, SLNC is introduced into cooperative MCD to further improve the downloading performance [66, 118]. In [66], Li et. al studied the practical benefits of SLNC over PLNC in content distribution in VANETs, whose work focused on protocol design. However, there still lacks an analytical model to compute the achievable throughput of SLNC in MCD system and to quantify the gain of it compared with PLNC.

There are only a few works studying the capacity of unicast or multicast in a VANET setting. The asymptotic transport capacity of VANET was studied in [82], and they demonstrated the achievable throughput in VANET is limited by $\Theta(\frac{1}{d})$ without considering interference and vehicle distribution (they assume the network is always connected), where d is an upper limit to the communication distance. In [50], Johnson et. al. considered a similar scenario to ours, and used percolation theory to derive the maximum network flow by modeling the network as a random graph based on the Aloha MAC protocol. The maximum flow converges to a constant N/e , when $N \rightarrow \infty$ and N is the number of nodes in each road segment. However, they did not take into account channel fading or vehicle distribution; as a result, their achievable throughput has no connection with the source-destination distance and node density,

which is impractical.

5.3 Achievable Throughput of Symbol-level Network Coding in Wireless Network

Although SLNC has been applied into content distribution in VANETs, an analytical model to compute the achievable throughput of SLNC in MCD system is currently lacking in the literature. In this section, we propose a theoretical framework by queueing theory and flow network to analyze the achievable throughput of SLNC in generic wireless network.

5.3.1 System model for SLNC in Wireless Network

In this section, we will formulate a system model for analyzing SLNC in generic wireless network. Before we elicit the system model, first of all, we must have a scheduling strategy to determine how packets are injected onto each arc, which is essential to the throughput performance. However, the packet scheduling problem is a difficult problem, and there are many existing works [75, 114, 116] to address this problem. In our generic framework, we assume a scheduling scheme is given.

Wireless network is modeled as a directed hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{A})$ [72], where \mathcal{N} is the set of nodes, \mathcal{A} is the set of hyperarcs. A hyperarc (i, J) represents a lossy broadcast link, where i is a node from \mathcal{N} , J is a non-empty subset of \mathcal{N} . Some injected packets on (i, J) are received by a set K which is a subset of J . We define the average rate at which packets are injected to the hyperarc (i, J) and is received by $K \subset J$ as z_{iJK} . If the packet counting process during time τ is $A_{iJK}(\tau)$, the average

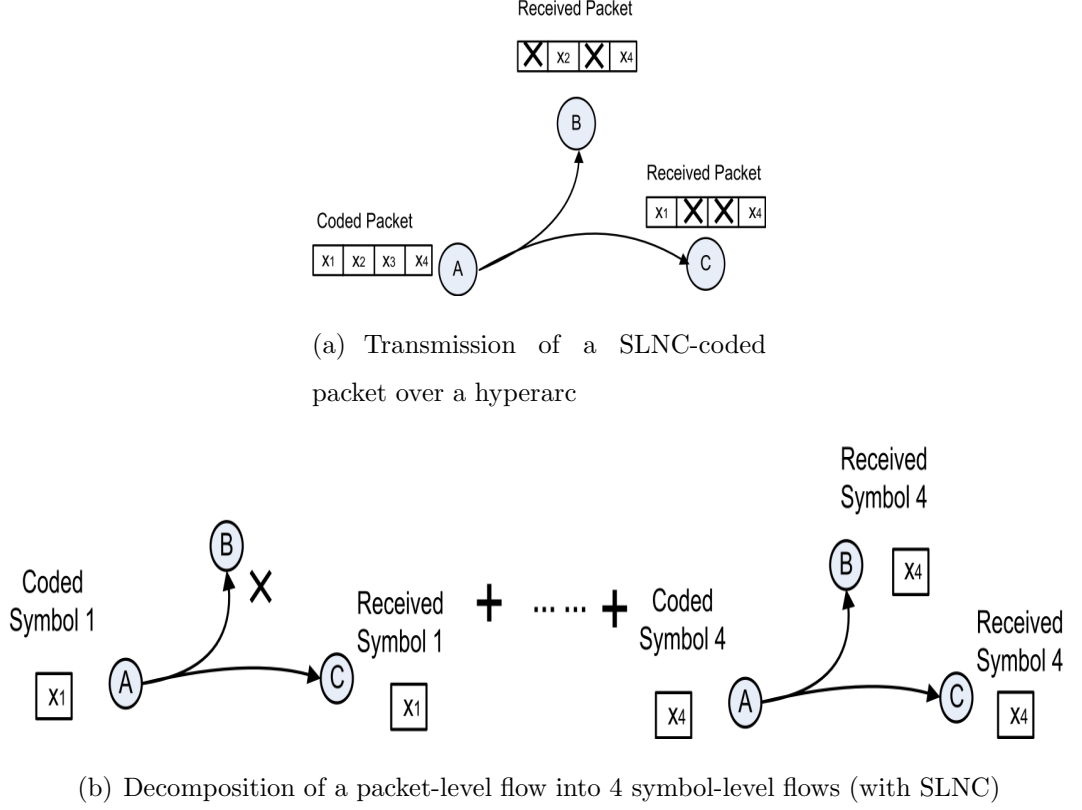


Figure 5.1: The example of flows from the perspective of SLNC ('X' means the corresponding symbol is corrupted)

$$\text{rate } z_{iJK} = \lim_{\tau \rightarrow \infty} A_{iJK}(\tau) / \tau.$$

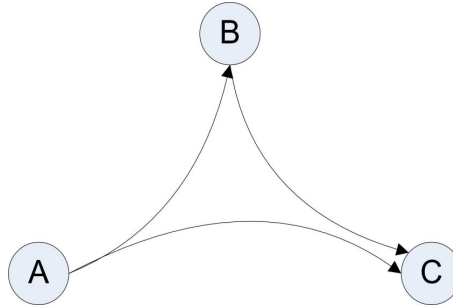
We believe the above hypergraph model is an accurate abstraction of packet-level wireless network, thus is a suitable model for analyzing PLNC. On the other hand, SLNC encodes the information at symbol-level. Whether a packet is received correctly is determined by the reception status of each symbol in the packet, thereby, making the hypergraph inappropriate to model SLNC. For instance, assume a coded packet $P = \{x_1, x_2, x_3, x_4\}$ is injected on arc $(A, \{B, C\})$ in Fig. 5.1(a), where x_m represents the m th coded symbol in P . Fig. 5.1(b) shows an instance of reception status on $\{B, C\}$ at symbol-level. If the transmission is viewed at the packet level

using hypergraph model, both B and C receives no useful information due to the failure of receiving a whole packet. However, if the transmission is viewed at the symbol level, the successfully received symbols (x_2, x_4 in B , x_1, x_4 in C) can be counted as “received” at the corresponding receivers. Since the existing network coding literature all view the information flow on the granularity of the whole packets, the above symbol-level reception phenomenon is neglected.

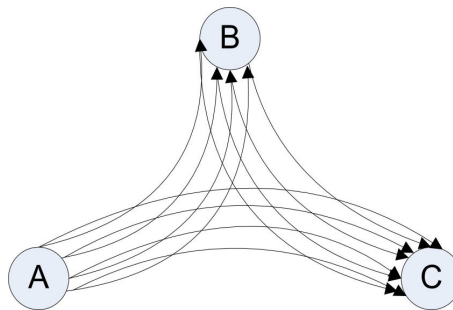
In order to fully capture SLNC performance, we convert the hypergraph wireless network model into a *multi-hypergraph* $\mathcal{H}^M = (\mathcal{N}, \mathcal{A}^M)$, where M is the number of symbols contained in each packet. In multi-hypergraph, as in Fig. 5.2(b), there are M corresponding hyperarcs $\{a_1, a_2, \dots, a_M\}$ in the hyperarc sets \mathcal{A}^M all with the same starting node i and end sets J . We call these M related hyperarcs as *multi-hyperarcs*. Conceptually, hyperarc a_m corresponds to the transmission of the m th coded symbol. By virtual of multi-hypergraph, the process of one packet injected to one hyperarc is decomposed into M injections of symbols to M hyperarcs. The multi-hyperarc can also be regarded as a special bundle of hyperarcs.

5.3.2 Achievable Throughput for SLNC in Wireless Network

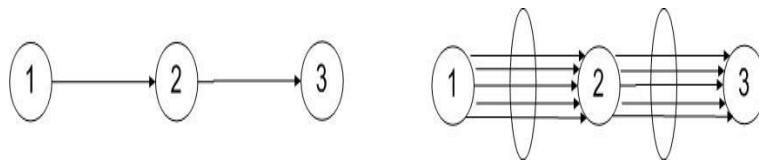
In this section, we give our general results for achievable throughput of SLNC from special cases. We first consider a two-link tandem network, which is a multigraph shown in Fig. 5.3. The tandem network with PLNC has been studied in [72]. With PLNC based tandem network, the propagation of innovative packets through a node follows the propagation of jobs through a single-server queueing station, which can be analyzed using fluid approximation for discrete-flow networks. The achievable throughput with PLNC is then proven to be determined by the average packet arrival rate on each arc.



(a) Hypergraph (for modeling PLNC)



(b) Multi-Hypergraph (for modeling SLNC)

Figure 5.2: Graph models for PLNC and SLNC*Figure 5.3:* two-link tandem network. Left: packet level queueing network; right: symbol level multi-queueing network.

On the other hand, the tandem network with SLNC has M arcs between two nodes (See Fig. 5.3). We virtually maintain one queue for each symbol position at the queueing station of one node, so that there are M multiple queues at each node. Then, the propagation of innovative symbols through arc (i, j) can be described as the propagation of jobs through a multiple single-server queueing system consisting of M

single-server queueing stations. We now demonstrate that the single-server queueing station at each symbol position works the same way as the single-server queueing station of PLNC.

As we mentioned above, the result of [72] shows that the propagation of packets with innovative information can be modeled with a single-server queueing station, which results in an achievable throughput determined by the average packet arrival rate z_{ij} on each arc (i, j) . If the arrival process of each symbol also has a same average rate, we can get the similar achievable throughput for each symbol.

Switch from the perspective of packets to symbols, the arrival of one received packet on arc (i, j) can be translated into the arrival of all received symbols in the packet on arc (i, j) , denoted by z_{ij} . If these symbols are independently received, it is clear that the arrival processes of symbols are i.i.d.. Assume symbols' reception probability is P_{sym} and packets' reception probability is P_{pkt} , then the average symbol arrival rate at the m th position is:

$$z_{ij}^{(m)} = \frac{P_{sym}}{P_{pkt}} z_{ij} \quad (5.1)$$

which apparently exists and is the same for each symbol. But if these symbols are received with correlated errors, by modeling the correlated channel as Markov chains [105], the arrival processes of packets or symbols still have average arrival rate from the demonstration of [72]. Furthermore, by scrutinizing a long term, each symbol will undergo the same fading process on average, through which all the symbols will show the same average symbol reception probability \bar{P}_{sym} , so that each of them must have the same average arrival rate.

Assume the m th symbol injection rate on arc (i, j) is $r_{ij}^{(m)}$, which is the same as packet injection rate r_{ij} . The average arrival rate of m th received symbol is thus

$z_{ij}^{(m)} = (1 - \varepsilon_{ij}^{(m)})r_{ij}^{(m)}$, where $\varepsilon_{ij}^{(m)}$ is the symbol loss rate. The achievable throughput of m th symbol in two-link tandem network is thus:

$$R^{(m)} \leq \min(z_{12}^{(m)}, z_{23}^{(m)}) \quad (5.2)$$

which is determined by average symbol arrival rate on each link. As the packet arrival rate on arc (ij) with PLNC is $z_{ij} = (1 - \varepsilon_{ij})r_{ij}$, thus the difference between symbol arrival rate with SLNC and PLNC on each link is relating to the loss rate: $\frac{1 - \varepsilon_{ij}^{(m)}}{1 - \varepsilon_{ij}}$. Note that we assume all the symbols are homogeneous. The symbol loss rate is smaller than packet loss rate, assume the symbol losses are independent with each other, the packet loss rate is $\varepsilon_{ij} = 1 - (1 - \varepsilon_{ij}^{(m)})^M$. For instance, symbol loss rate $\varepsilon_{ij}^{(m)} = 0.2$, then the packet loss rate is $\varepsilon_{ij} = 0.5904 > \varepsilon_{ij}^{(m)}$ with $M = 4$. Therefore, the gain of SLNC over PLNC on each link is derived from symbol-level diversity, which mainly comes from the increased transmission success rate at symbol-level. By virtual of Eq. (5.2), we make a further conclusion that the gain of achievable throughput of SLNC over PLNC comes from symbol-level diversity.

The result of two-link tandem network can be extended to L -link tandem network, thus the achievable throughput of m th symbol in L -link tandem network is given by:

$$R^{(m)} \leq \min_{1 \leq i \leq L} \{z_{i(i+1)}^{(m)}\} \quad (5.3)$$

Finally, we further extend the results into symbol-level wireless networks, in which the multi-hypergraph can then be separated into M independent hypergraph for each symbol. In m th hypergraph \mathcal{A}_m^M , the m th symbol injected on hyperarc (i, J) is received by $K \in J$. The average arriving rate of the symbol is $z_{iJK}^{(m)}$, then the achievable symbol rate from source s to destination t is:

$$R_t^{(m)} \leq \min_{\mathcal{Q} \in \Omega^{(m)}(s,t)} \left\{ \sum_{(j,J) \in \Gamma_+(\mathcal{Q})} \sum_{K \not\subset \mathcal{Q}} z_{iJK}^{(m)} \right\} \quad (5.4)$$

where $\Omega^m(s, t)$ is the set of all cuts between s and t on the m th conceptual hypergraph, and $\Gamma_+(\mathcal{Q})$ denotes the set of forward hyperarcs of the cut \mathcal{Q} , i.e. $\Gamma_+(\mathcal{Q}) := \{(i, J) \in \mathcal{A}_m^M | i \in \mathcal{Q}, J \setminus \mathcal{Q} \neq \emptyset\}$, where K represents the subsets of hyperarcs in the m th hypergraph \mathcal{A}_m^M .

There exists a flow vector f , which can be modeled with the following two constraints:

1) Conservation constraint for the flow at m th symbol position (holds true for all $i \in \mathcal{N}$):

$$\sum_{\{j|(i,J) \in \mathcal{A}_m^M\}} \sum_{j \in J} f_{iJj}^{(t)} - \sum_{\{j|(j,I) \in \mathcal{A}_m^M, i \in I\}} f_{jIi}^{(t)} = \begin{cases} R_t^{(m)}, & \text{if } i=s, \\ -R_t^{(m)}, & \text{if } i=t, \\ 0, & \text{otherwise.} \end{cases} \quad (5.5)$$

2) Capacity constraint (holds true for all $(i, J) \in \mathcal{A}_m^M$ and $K \subset J$):

$$\sum_{j \in K} f_{iJj}^{(t)} \leq \sum_{\{L \subset J | L \cap K \neq \emptyset\}} z_{iJL}^{(m)} \quad (5.6)$$

Thus, the achievable throughput of m th symbol's propagation is the right-hand side of Eq. (5.4), which is determined by the minimal cut of wireless hypergraph. In order to approach this achievable throughput, we can decompose the network flow f into a finite set of paths p_1, p_2, \dots, p_N [72], each carrying positive flow $R_{ti}^{(m)}$, such

that $\sum_{i=1}^N R_{ti}^{(m)} = R_t^{(m)}$. Each path is a tandem network and can deliver innovative symbols at a rate arbitrary close to $R_{ti}^{(m)}$, which will result in an arrival rate of innovative symbols at node t arbitrary close to $R_t^{(m)}$. Finally, the overall achievable throughput at symbol-level R_t is given by:

$$R_t = \sum_{m=1}^M \{R_t^{(m)}\} \quad (5.7)$$

Where R_t is measured in number of symbols per second, from which we can easily get achievable throughput in bytes or bits. In sum, with a given average symbol arrival rate on each hyperarc, the achievable throughput of SLNC in wireless network is determined.

5.4 Throughput analysis of cooperative MCD system using SLNC

In this section, we apply the above analysis to model the throughput of cooperative MCD system using SLNC in VANETs. We formulate the problem in a line-shaped road topology with one AP as the source. From the achievable symbol throughput of SLNC indicated in Eq. (5.4), we take three steps to approach the achievable throughput of cooperative MCD system: first, under a specific VANET topology, given a channel fading model and a medium access control scheme, we compute per-link average symbol arrival rate by “communication density”; second, with the average symbol arrival rate of each link, we address the max-flow min-cut problem to get the maximal divergence out of the source over all capacity-achievable flows, and acquire distance-limited achievable throughput at distance d away from the source by averaging among all topological instances; third, by integrating vehicle mobility, ex-

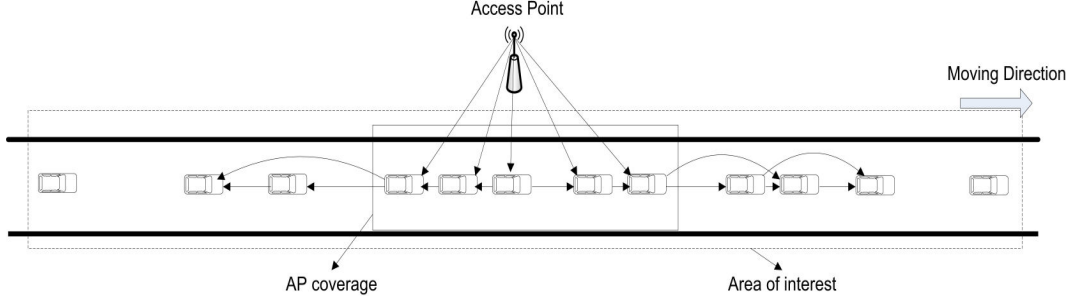


Figure 5.4: The architecture for MCD. AP owns contents to broadcast to vehicles; vehicles distribute their received contents cooperatively.

pected downloading volume is obtained via aggregating distance-limited achievable throughput of a measured vehicle inside an AoI.

5.4.1 Problem Formulation

In this section, we consider the following MCD service architecture for vehicular network with a line-shaped road topology. The content providers such as Public Transport Authorities disseminate safety information or commercial Ads to a roadside AP, which is the only data source in the network. The AP then actively broadcasts these files to the vehicles inside a geographical AoI in a way that ensures all the vehicles efficiently and continuously receive the complete files. The MCD service architecture is illustrated in Fig. 5.4.

The vehicles on the road are equipped with wireless transceivers, with which they can receive packets from AP and transmit them to their neighboring vehicles. All the vehicles inside a AoI are interested at the disseminated packets from the AP, so that they will actively download all the overheard packets, store them in the buffer and send coded packets to their neighbors. We use SLNC to encode the packets.

Briefly, we want to study the above single-flow multicast content distribution from

the AP to the moving vehicles on the road inside AoI using multi-hop vehicle-to-vehicle cooperation, and endeavor to attain the achievable throughput for a vehicle at a certain distance away from AP. This cooperative MCD system is a kind of Drive-thru Internet System [101]. But different from the normal Drive-thru Internet System, cooperative MCD system considers vehicle-to-vehicle cooperation to expand the coverage and increase overall downloading volume.

To make the problem tractable, we make some important assumptions: first, we assume the road is long enough such that the neighborhoods of a vehicle from both sides are extending to infinite; second, we assume all the vehicles on the road are homogeneous and have similar neighborhoods. The positions of vehicles follow homogenous node distribution. Nevertheless, vehicular density can be arbitrary which is dense with continuous lines of vehicles or sparse with only a few vehicles on the road.

As we consider a wireless network with broadcast links, an important issue we need to consider is medium access control scheme, which determines how to share the wireless medium among the nodes. As an amendment of WLAN standard IEEE 802.11, IEEE 802.11p [47] is proposed as the dedicated short range communications (DSRC) standard in VANETs, which employs IEEE 802.11 Distributed Coordination Function (DCF) MAC method [11]. Thus, we adopts 802.11 MAC protocol in the cooperative MCD system underlying SLNC scheme.

It is well known that the network with 802.11 MAC scheme exhibits an unstable behavior [10], which means as the offered load from source increases, the throughput of the network grows up to a maximal value. However, further even minor increases of the offered load will decrease system throughput significantly, making it impractical to operate at the maximal throughput for a long time especially in dynamic vehicular network with fast-changing topology. This form of instability exist in both single-

hop [10] and multi-hop [83] networks. Therefore, we would like to study the maximal load the system can carry in stable conditions, which can be expressed as saturation throughput. Then in the cooperative MCD using SLNC, with the restriction of 802.11 MAC, we focus on the achievable throughput in the scenario when all the nodes including source and relays potentially always have packets stored in their buffer and send coded packets whenever is possible.

Then we give some definitions to some frequently used phrases in this chapter:

Definition 1 (Achievable throughput). *the feasible flow rate a source node can send to its destinations over a long term. In this chapter, the achievable throughput is specified as the throughput in stable conditions, as mentioned above;*

Definition 2 (Link average arrival rate). *the average reception rate that a node can receive over a particular link (arc) in a long term;*

Definition 3 (Distance-limited achievable throughput (DLAT(d))). *the expected achievable throughput of a node with a certain distance d from the source in a network under some node distribution;*

Definition 4 (Expected Downloading Volume (EDV)). *the expected total amount of information that a vehicle can accumulate when driving through the AoI.*

5.4.2 Symbol-Level Link Average Arrival Rate

Given a specific topology of wireless network, its hyperarc flow is denoted by connections between one node and a set of receiving nodes, i.e. one symbol is placed on arc (i, J) , and will be received by $K \subset J$. However, the symbol reception rate $z_{iJK}^{(m)}$ of hyperarc flow connections is difficult to derive. Particularly, regarding realistic issues of channel fading, interference and node distribution, computing the achiev-

able throughput of single-flow multicast network with 802.11 MAC protocol is a hard problem.

Thus, we resort to using the results of “Communication Density” [46]. Communication density can be described as the product of vehicle density, message generation rate and transmission range. [46] shows that the node reception status of a line-shaped broadcast system with 802.11 MAC protocol is determined by the message size and communication density around the node under practical channel conditions. By taking advantage of communication density which already takes channel variety and packet collision into account, the reception rate of node A contributed by source S through the arc (S, A) , considering interference of its neighboring nodes, can be presented by a closed-form formula.

5.4.2.1 Link Average Arrival Rate Considering Channel Fading

As illustrated above, the link average arrival rate considered in this section is in symbol-level, which can also be denoted as average symbol reception rate. One important element impacting reception rate is the channel randomness which is brought by channel fading, causing the link average arrival rate to decay with the distance. In Rayleigh fading channels, the received signal amplitude obeys Rayleigh distribution. We assume the impact to channel randomness by variation of Gaussian noise is negligible compared with the variation of SNR caused by fading process. Then, the fading process is constant over the transmission of one frame and the subsequent fading processes are *i.i.d.*. Thus, The pdf of *physical symbol*¹ SNR γ_{ps} is given by [89]:

$$f(\gamma_{ps}) = \frac{1}{\gamma_{ps}} e^{-\frac{\gamma_{ps}}{\gamma_{ps}}} \quad (5.8)$$

¹It denotes the symbol generated by modulation schemes.

Where $\bar{\gamma}_{ps}$ is the average physical symbol SNR. Consider a simple path-loss model for signal propagation, the receiving power is decayed with distance as $d^{-\alpha}$. The transmission energy per physical symbol is denoted by E_{ps} , and Gaussian noise spectral density is denoted as N_0 . The average physical symbol SNR can then be written as $\bar{\gamma}_{ps} = \frac{E_{ps}}{d^\alpha N_0}$. We model the loss process with a SNR threshold γ_{th} , which means the physical symbols are correctly received if and only if the physical symbol SNR γ_{ps} is greater than the given threshold γ_{th} . Thus, the probability that the physical symbol is correctly received at a distance x from the source is given by:

$$P_{succ,ps}(x) = P(\gamma(x) \geq \gamma_{th}) = \int_{\gamma_{th}}^{\infty} f_{\gamma_{ps}}(x) dx = e^{\frac{\gamma_{th}}{\bar{\gamma}_{ps}}} = e^{-\rho d^\alpha} \quad (5.9)$$

Where ρ is $\frac{\gamma_{th} N_0}{E_{ps}}$. Here, we assume a SNR threshold model to represent the error model, which greatly simplify our analysis.

Now we consider two types of Rayleigh fading channel: Channel A is characterized by flat slow fading with no line-of-sight (LoS) path, while channel B is characterized by fast fading. These two types of channels indicate two extreme cases in real vehicular network, which will be brought by various vehicle density and surrounding environments. Assume a symbol in SLNC contains μ physical symbols. In channel A , physical symbols are fully correlated within the packet because of the larger channel coherence time in slow fading scenario. Then the probability that the symbol is correctly received at a distance x from the source in this case is give by:

$$P_{succ,ns-corr}(x) = P_{succ,ps}(x) = e^{-\rho d^\alpha} \quad (5.10)$$

In channel B , physical symbols are independent with each other within the packet because of the smaller channel coherence time in fast fading scenario. Then the

probability that the symbol is correctly received at a distance x from the source in this case is give by:

$$P_{succ,ns-ind}(x) = (P_{succ,ps}(x))^\mu = \left(\int_{\gamma_{th}}^{\infty} f_{\gamma_{ps}}(x) dx \right)^\mu = (e^{-\rho d^\alpha})^\mu \quad (5.11)$$

Finally, the link average arrival rate of m th symbol on arc (i, j) considering Rayleigh fading can be given by:

$$z_{fad}^{(m)}(i, j) = r_{i,j}^{(m)} \cdot P_{succ}(d_{ij}) \quad (5.12)$$

where d_{ij} is the length of arc (i, j) . $P_{succ}(x)$ can be derived from Eq. (5.10) and Eq. (5.11) for both fully-correlated symbol error and independent symbol error cases.

5.4.2.2 Link Average Arrival Rate Considering Collision

In our cooperative MCD system, another important element we need to consider is interference caused by simultaneous transmissions, which is mostly induced by well known “hidden terminal” problem. Even worse, when the system is saturated (nodes in the system effectively all have packets to send), the collision will potentially cause many packets to drop, which will greatly affect the throughput performance.

We take advantage of “Communication density” [46] to analyze the packet collision. Recall the assumption that all the vehicles on the road are homogeneous distributed and have similar neighborhoods. And then, the probability of message transmission in a slot is assumed to be constant and independent of nodes. Furthermore, we assume any packet collision will result in packet reception error, then the corresponding packet reception probability at a distance x can be given by [46]:

$$P_{succ}(x) = p \cdot (1 - p_{fad}) \cdot (1 - p_{col}) \quad (5.13)$$

where p is message transmission probability in one slot. p_{fad} is packet error probability caused by channel fading, and p_{col} is packet error probability caused by packet collision. Each of the above three parameters should be evaluated to determine packet reception probability. Next, we will evaluate p_{col} and p respectively, as p_{fad} has already been addressed in the last section.

As we consider a 802.11 MAC protocol running underlying the cooperative MCD system, the collision probability is determined by both carrier-sensing technique and hidden-node.

On one hand, collision probability at a node is closely related to its carrier sense capability. The carrier sense probability at distance d from the source is given by $\chi(d)$. Assume there is a SNR threshold for carrier sense. For Rayleigh channel, the probability of the source's carrier being "sensed" at distance d can be given by $\chi(d) = e^{-\chi d^\alpha}$ with $\chi > 0$, and χ depends on the transmission power. The carrier sense range d_{cs} is defined as the distance at which carrier sense probability falls below a certain threshold, thus d_{cs} is closely tied to χ . Then, the probability of sensing a new transmission during one time slot is given by [46]:

$$r = 1 - e^{-\sqrt{\pi}\delta p d_{cs}} \quad (5.14)$$

On the other hand, hidden-node is another important factor in determining collision probability. By considering hidden-node collision, from [46], the probability of packet collision occurred at the node is given by:

$$p_{col} = 1 - e^{-(2S_{msg}-3)\delta p d_{cr}} \quad (5.15)$$

Where S_{msg} is the message size in slots, p is the message transmission probability in one slot, δ is node density and d_{cr} is the communication range.

As illustrated above, in terms of saturated throughput, all the nodes including source and relays potentially always have packets stored in their buffer and send coded packets whenever is possible, thus the message generation rate for each node is infinite. Assume all the nodes in the network are homogeneous, according to [46], the message transmission probability in one slot p is given by:

$$\frac{1}{p} \cong T_0 + \frac{(W-1)}{2} T_1 \quad (5.16)$$

Where W is backoff window size. T_0 is the average amount of time a node spends in its own transmission including the extra waiting time due to sensible overlapping transmissions and T_1 is the average time spent to decrease the backoff counter by 1. Both of them are given in [46]. T_0 and T_1 are expressed in terms of r , and r is expressed in terms of p and model parameter. Thus from Eq. (5.16), we use numerical method to calculate the message transmission probability in one slot p .

The message pointed out above is in unit of time slots, here we define a packet as a partial message transmitted in one time slot. After calculating message transmission probability p , the data transmission rate at each node is given by:

$$R_{tr} = p \cdot S_{msg} \cdot \frac{L_{pkt}}{\tau_s} \quad (5.17)$$

Where S_{msg} is the message size in slots, L_{pkt} is the packet length which is also

the payload length in one time slot. τ_s is the time slot duration. Thus, $\frac{L_{pkt}}{\tau_s}$ is the effective data transmission rate. R_{tr} can be measured as bits per second or bytes per second.

The above formulations are all in perspective of packets. Now, we consider symbol-level reception rate, which is in the same form of Eq. (5.13). For one thing, the symbol error probability caused by channel fading is provided in the last section. For another, symbol collision probability can be derived based on packet collision probability of Eq. (5.15) as follows. For simplicity, we assume the packet collision is uniformly occurred in the whole packet and the symbols after occurrence of packet collisions are all discarded. Therefore, we can say in average, half of the symbols in the collided packet will be devastated over a long time period, i.e. the symbol collision probability is halved:

$$p_{col}^{(sym)} = p_{col}/2 \quad (5.18)$$

Combine the result of the last two sections Eq. (5.13), Eq. (5.15) and Eq. (5.18), the m th symbol's average reception rate at distance d is as follows:

$$R_{rec}^{(m)}(d) = R_{tr} \cdot e^{-\rho d^2} \left[1 - \frac{1}{2} (1 - e^{-(2S_{msg}-3)\delta p d_{cr}}) \right] \quad (5.19)$$

With ρ defined at the end of last section, and the pass-loss factor $\alpha = 2$. Then the link average symbol arrival rate of link (i, j) considering fading and collision is expressed as follows:

$$z_{i,j}^{(m)} = R_{rec}^{(m)}(d) \quad (5.20)$$

So up to now, we have attained symbol-level link average arrival rate with respect to distance d using 802.11 MAC protocol. Next, with a specific topology, according to Eq. (5.4), we are going to find the max-flow min-cut of this multicast network with each link weighted by its link average arriving rate.

5.4.3 Achievable throughput of cooperative MCD system

Although we are aware of the node distribution, analyzing the random MCD network under the above realistic considerations is a difficult problem. In this section, we employ analytical method to generate multiple specific network topology instances. And then, for each network topology instance, achievable throughput at distance d from source is derived through a max-flow min-cut algorithm. By averaging over all the achievable throughput from various instances, $DLAT(d)$ can be obtained.

We begin with a deterministic network topology instance \mathcal{D}_i generated by a certain node distribution, for example, Poisson distribution. \mathcal{D}_i can be viewed as a graph containing randomly distributed nodes. We consider a virtual node t located on graph \mathcal{D}_i , who has a distance d away from the source $s \in \mathcal{D}_i$. We are concerned with the achievable throughput at t . To derive it, we further define the hypergraph between source s and t as $\mathcal{G}_{st}^{(i)}(d) \subset \mathcal{D}_i$. From network flow's perspective, the max-flow from s to t is determined by $\mathcal{G}_{st}^{(i)}(d)$. Utilizing the results of last section, we search for max-flow in $\mathcal{G}_{st}^{(i)}(d)$ with each link weighted by its link average arriving rate. Then, Ford-Fulkerson algorithm is employed to find the maximal flow or minimal cut of $\mathcal{G}_{st}^{(i)}(d)$, which is the achievable throughput at distance d from source in this specific topology instance.

After that, we average over all these instances to obtain expected achievable throughput $DLAT^{(m)}(d)$ at the m th symbol position. Then the overall distance-

limited achievable throughput at symbol-level is $DLAT(d) = \sum_{m=1}^M DLAT^{(m)}(d)$.

Until now, we have computed the expected achievable symbol throughput in Eq. (5.4), which can be reached using SLNC.

5.4.4 Expected Downloading Volume with Mobility

Until now, all the throughput calculated above is that of static networks, where the mobility has not been considered. A following up question that “How much information volume can a vehicle download when driving across the AoI?” is what we are going to address in this section.

The time period of a vehicle driving across the AoI can be divided into several short time intervals, which should be much larger than the maximal end-to-end transmission delay in order to be counted as “a long term” for calculating achievable throughput. The maximal end-to-end transmission delay can be separated into two parts: the average time for channel contention and air time for transmission. In realistic condition, as the achievable throughput drops down dramatically with distance. We assume a meaningful AoI from source as $1200m$, which is nearly 5 hops from the source. We assume the message transmission probability in each slot is $p = 0.05$ ², then the average time spent for channel contention is $1/p = 20$ slots. And further assume the message transmission air time including the MAC overhead is 55 slots³ and one slot time is $20\mu s$, then, the maximal end-to-end transmission delay is approximately $7.5ms$. For example, we choose the short time interval as one second,

²In a MCD system with exponential inter-distance distribution, in which average distance is 40 meters, we calculate $p=0.05$.

³We assume packet payload is 1500 bytes, other 802.11 overhead is 150 bytes including preamble, header and ACK, the slot time is $20\mu s$. The data transmission rate is 12 Mbps, then the air transmission time is $\frac{(1500+150) \times 8}{12} = 1100\mu s$, which is 55 slots.

then more than 133 transmissions occurred inside AoI, which can maintain a proper network flow in a long term for reaching achievable throughput. On the other hand, with the vehicle velocity as $20m/s$, the vehicle movement of $20meters$ in one second will not change the whole network topology. Therefore, we can consider each network instance as static by dividing the time period into one second interval.

By the results of $DLAT(d)$ for each snapshot, we can obtain the expected downloading volume EDV_t at node t as follows:

$$EDV_t = \sum_{\Delta t=0}^{\lfloor \frac{T_{out}^t - T_{in}^t}{\Delta t} \rfloor} t \cdot DLAT(|d(\Delta t)|) \quad (5.21)$$

where T_{in}^t and (T_{out}^t) is the time when vehicle t gets into (departs) the AoI, which is determined by vehicle velocity, vehicle density and the length of AoI. $|d(\Delta t)|$ is the distance between the vehicle and AP, which gradually diminishes when approaching AP and increases when departing from AP.

5.5 Numerical Evaluation and Discussion

In this section, we demonstrate numerical results from our model. We first give the parameters used in the numerical evaluation in Table. 5.1.

We generated various network topology instances with node inter-distance following exponential distribution. Two different channel conditions are considered, one is fast fading with independent physical symbol error case and the other is slow fading with fully correlated physical symbol error case. In Fig. 5.5(a)-(c), we show the achievable throughput of different vehicle density variances from our models. Note that each sub-figure has three throughput curves, for packet error rate and symbol er-

Table 5.1: Simulation parameters

Parameters	Values
<i>Simulation runs</i>	10000 times
<i>Data transmission rate</i>	12 Mbps
<i>Message size</i>	15 slots
<i>CW (Backoff window size)</i>	15 slots
<i>Slot time τ_s</i>	20 us
<i>DIFS</i>	20 us
<i>Data transmission range</i>	250 meters
<i>Carrier sensing range</i>	400 meters
<i>Pass loss factor</i>	2
<i>Length of AoI</i>	1.2 km
μ	4

ror rate at both fast fading and slow fading channel respectively. From the numerical results denoted in each figure, during fast fading case, SLNC has a gain over PLNC, while during slow fading case, PLNC and SLNC appears to have the same achievable throughput. This result shows the benefit of SLNC over PLNC comes from symbol-level diversity. The benefit of SLNC through symbol-level diversity will diminish, when the correlation degree of physical symbol error increases. Therefore, the gain of SLNC over PLNC is upper bounded by the fast fading case with independent symbol error.

Now, let us focus on the influence of different vehicle density variances among three sub-figures. Comparing the expected achievable throughput in the case with fully-correlated symbol error, we notice some insights here: when the vehicle density increases, the expected achievable throughput at nearer hops from the source first in-

creases due to better connectivity (Fig. 5.5(a) – Fig. 5.5(b)), and then decreases due to heavier interference (Fig. 5.5(b) – Fig. 5.5(c)). With the increase of vehicle density, the slope of these curves become smaller, which means the achievable throughput decreases slower with distances. This is because better overall connectivity is guaranteed in the high density network, which determines the achievable throughput over long distance. So here we find out an interplay between network connectivity and interference level. This result provided insight on optimization of MCD system design in VANETs. In short, for different vehicle density scenarios, both network connectivity and inter-node interference should be considered to determine offered data rate and AP deployment for MCD in VANETs.

Then we study achievable throughput performance under two different node distributions in Fig. 5.6. One is uniform distribution and the other is regular network with equal inter-distance. Both of these two distributions have the same expected inter-distance as 40 meters. Fig. 5.6(a) and Fig. 5.6(b) shows that in fast fading case, the expected achievable throughput of PLNC is nearly zero. In this case, SLNC has much higher gain over PLNC. We also notice that regular network always shows a straight line because of its deterministic topology and homogeneous node density.

Finally, from Fig. 5.8, we show the expected downloading volume with different average inter-distances. The x-axis in the figure denotes the starting distance of vehicle from AP. As the movement of vehicles follow the routines of first moving forward AP and then moving away from AP, the distance that a vehicle passes through is two times of the starting distance. For example, at starting distance of 200 meters, we compute the downloading volume over the distance of 400 meters. Without embedding traffic model, Fig. 5.7 shows the expected downloading volume with fixed velocity of $20m/s$. In this case, the vehicle velocity has no relationship with vehicle density. The sparsest density case shows the lowest downloading volume, due to its

low achievable throughput near AP and fastest drop of achievable throughput with the increasing distance, illustrated in Fig. 5.5(a). However, the mediate density case shows the highest downloading volume until its starting distance from AP exceeds nearly 650 meters. Beyond 650 meters, the downloading volume of heaviest density case overtakes that of mediate density case. We use the same interplay between network connectivity and interference level, demonstrated in Fig. 5.5, to explain this phenomenon. Because of higher interference level, in the heaviest density case, a small volume is accumulated around the AP. However, downloading volume steadily rise up with the expansion of AoI, by merit of its stable overall connectivity.

Next, we use a practical traffic model to capture the relationship between vehicle velocity and vehicle density [18]:

$$v(m/s) = -0.15\delta + 30 \quad (5.22)$$

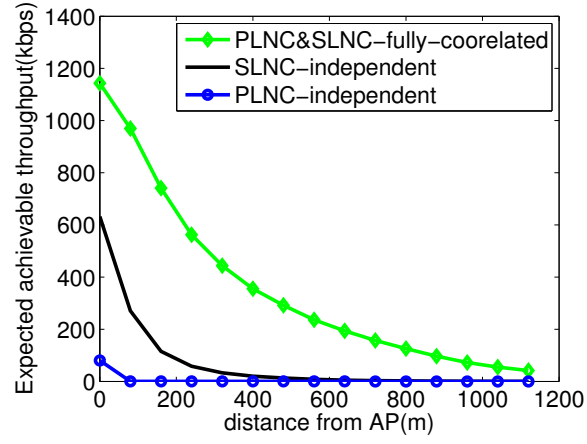
Where $v(m/s)$ is the vehicle velocity, δ represents the number of vehicles per kilometer, expressed as vehicle density.

Fig. 5.8 shows that the vehicles in the heaviest density network has the highest expected downloading volume and the downloading volume falls with the decrease of vehicle density. We notice in the traffic model of Eq. (5.22), higher density brings about lower velocity, which will induce higher downloading volume. This explains that in highest density case, the vehicle has more time spending in the AoI, thus can accumulate more contents.

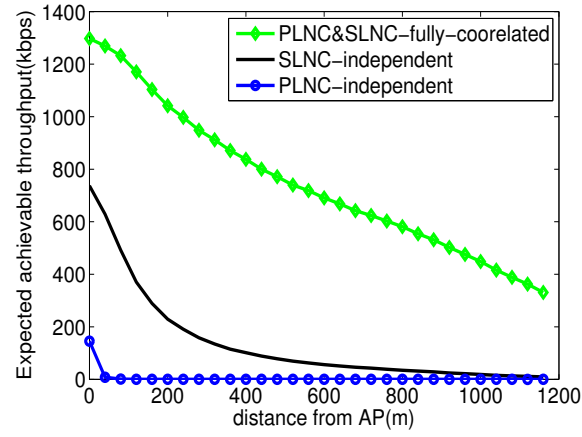
5.6 Summary

In this chapter, we have done an analytical analysis for the achievable throughput of cooperative MCD in VANETs using SLNC. We first propose a generic model to

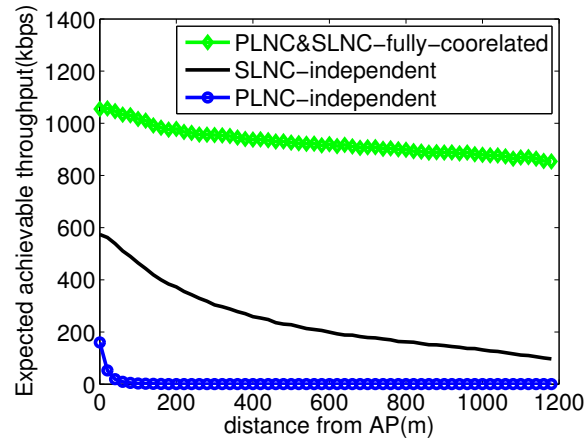
compute the achievable throughput of SLNC in wireless networks. We show that as long as the packet arrivals conform to an arrival process having average rates, the difference in achievable throughput between SLNC and PLNC is determined by symbol-level diversity, which mainly comes from the increased transmission success rate at symbol-level. We then propose a method to analyze the expected achievable throughput of cooperative MCD system with SLNC in VANETs. We consider realistic assumptions such as channel fading, interference and vehicle distributions, under which the distance-limited achievable throughput of a vehicle is characterized. Furthermore, by considering vehicular mobility pattern, we compute the expected downloading volume of a vehicle passing through an AoI with one AP as the source. Through numerical results, we reveal the impacts of using PLNC & SLNC under different channel fading levels, the vehicle distribution and vehicle density to the throughput limits of MCD, which provide valuable insights for optimized deployment of APs and the cooperative MCD system design.



(a) Average inter-distance is 80m

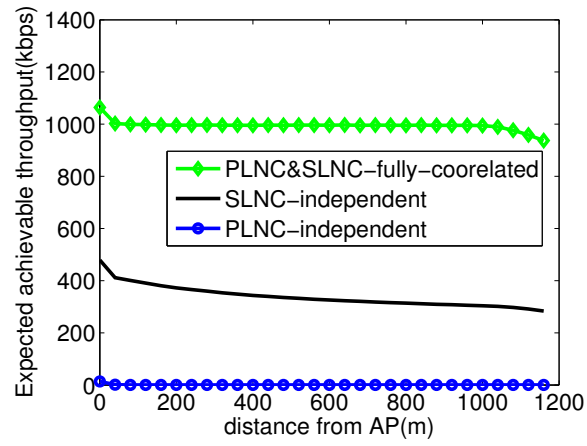


(b) Average inter-distance is 40m

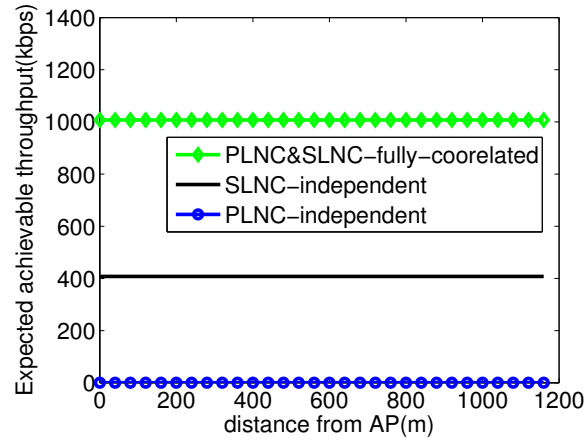


(c) Average inter-distance is 20m

Figure 5.5: Average achievable throughput of MCD in vehicular network with exponential inter-distance distribution



(a) Uniform inter-distance distribution with 40m average inter-distance



(b) Regular inter-distance with 40m inter-distance

Figure 5.6: Comparison of different inter-distance distribution

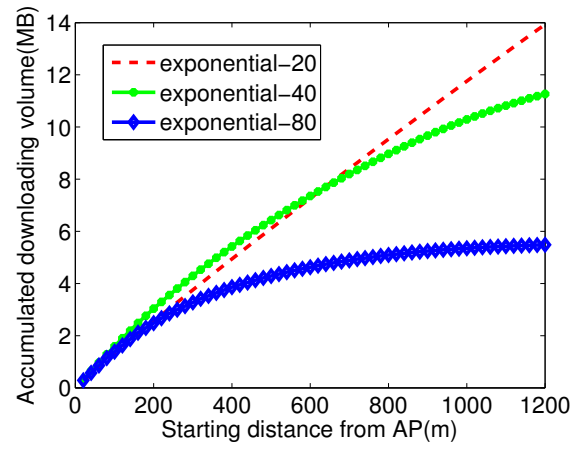


Figure 5.7: The expected downloading volume of different average inter-distances with fixed velocity of 20m/s

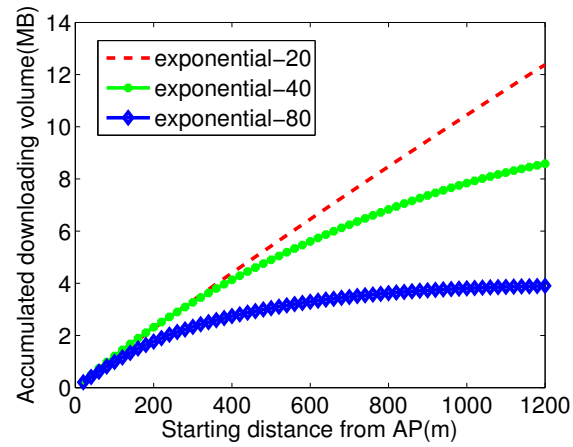


Figure 5.8: The expected downloading volume of different average inter-distances with traffic model

Chapter 6

Conclusions and Future Research

6.1 Summary

Network coding is an amazing simple but effective idea to improve the network throughput and transmission reliability, especially in the wireless scenarios, which makes it well suited for protocol design targeting on various multi-hop wireless networks. In chapter 2, we study the reliable broadcast in static wireless mesh networks and put forward R-Code based on the observation that once a node fully downloaded the file, it could play the role of source and disseminate the content of the file more efficiently to its neighbors than the original source node. In other word, the responsibility of reliable information delivery is transferred from the global responsibility of the source to the localized responsibilities of guardians to their corresponding wards. In chapter 3 and 4, we study the reliable content distribution in mobile VANETs, and proposed CodeOn and CodePlay, two novel SLNC based schemes. In order to fully enjoy the benefits of SLNC, we propose a suite of techniques to maximize the spatial reusability while avoid introducing too much concurrent transmission interference.

In chapter 5, we develop a theoretical model to compute the achievable throughput of cooperative mobile content distribution in VANETs using SLNC, which could provide valuable insights and guidelines for future protocol design in mobile content distribution scheme for VANETs.

6.2 Further Research

Below, we list a few challenging issues that need further study based on the work of this dissertation.

6.2.1 Multi-Flow MCD in VANETs

Primarily, the mobility of the VANET poses a bigger challenge for multi-flow MCD. This is because now we must jointly consider the dynamic relay selection, hyper-arc selection, and flow scheduling problems. In a single-flow MCD with all nodes in an AoI as the (geocast) destination, the selection of relays and hyperarcs is relatively easy, i.e., we could take all the current neighbors of a relay into account. However, for multi-flow MCD, since there are potentially many multicast groups and multiple flows, and the locations of those destinations are constantly changing. On the one hand, the traditional methods which are based on establishing multicast trees based on global topology are no longer suitable since they incur too much overhead.

6.2.2 Multi-Rate Adaptive SLNC

In this dissertation, we all assume using a single rate for all the transmission links, which may suffer from performance degradation under bad channel conditions in

VANET with fading, interference and mobility, and renders the bandwidth of VANETs under-exploited. However, meanwhile it is possible and interesting to investigate how to take advantage of the varying channel conditions by employing adaptive rate selection for intra-session SLNC-based MCD, as multi-rate functionality is enabled by the IEEE 802.11p standard. The primary challenge to this problem comes from the ever-changing VANET topology. For a node to decide which transmission rate to use, traditional methods usually need to measure the SNR or bit error-rate (BER) of every link in the network in advance, or estimating them on the fly. However in VANET, the channel coherence time is small due to high mobility, and it is very hard for every node to obtain accurate SNR or BER in real-time. Second, there is a trade-off between rate selection and multicast: the higher the rate, the fewer vehicles in the hyper-arc can receive a coded transmission and the throughput may even become lower. Third, the relay selection needs to consider both content usefulness and the relays channel condition to its neighbors, however the relay selection and rate selection are inter-dependent. This is because who are chosen to be the first transmitted relays affects the SNR of neighborhood of relays thereafter. In light of all the above, we believe that jointly optimized relay selection and rate selection maybe is necessary for achieving better performance.

6.2.3 MCD under Disconnected VANET (DTN) Scenarios

Another traffic regime we want to look into is the disconnected scenario, which is often referred to as delay-tolerant network (DTN). Under this paradigm, no contemporary end-to-end path may exist between a source-destination pair, which invalidates the traditional multicast methods. Store-carry-and-forward strategy is often employed to address this challenge, however, not for MCD or multicast with network coding. Our current works for SLNC-based multimedia streaming exploited the unused transmis-

sion opportunities for vehicles located near empty road segments, which was shown to improve significantly upon non-opportunistic transmission. However, a more fundamental issue is whether we can always achieve the practical throughput requirements under the DTN paradigm with SLNC, which has not been well-understood by existing works and is interesting research topic for future works.

Bibliography

- [1] Dedicated Short Range Communications.
- [2] Ns2. <http://www.isi.edu/nsnam/ns>.
- [3] The usc mobility generator tool.
- [4] Ieee trial-use standard for wireless access in vehicular environments (wave) - multi-channel operation. *IEEE Std 1609.4-2006*, pages c1–74, 2006.
- [5] C. Adjih, S. Y. Cho, and P. Jacquet. Near optimal broadcast with network coding in large sensor networks. In *First International Workshop on Information Theory for Sensor Networks, Sante Fe , USA*, June 2007.
- [6] R. Ahlswede, Ning Cai, S.-Y.R. Li, and R.W. Yeung. Network information flow. *IEEE Transaction on Infomation Theory*, 46(4):1204–1216, Jul 2000.
- [7] Shabbir Ahmed and Salil S. Kanhere. Vanetcode: network coding to enhance cooperative downloading in vehicular ad-hoc networks. In *IWCMC '06*, pages 527–532, 2006.
- [8] I.F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, August 2002.

- [9] I.F. Akyildiz and Xudong Wang. A survey on wireless mesh networks. *Communications Magazine, IEEE*, 43(9):S23 – S30, 2005.
- [10] Giuseppe Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *Selected Areas in Communications, IEEE Journal on*, 18(3):535–547, 2000.
- [11] Katrin Bilstrup, Elisabeth Uhlemann, Erik G. Strom, and Urban Bilstrup. Evaluation of the ieee 802.11p mac method for vehicle-to-vehicle communication. In *IEEE VTC Fall 2008*, pages 1–5, September 2008.
- [12] S. Biswas and R. Morris. Exor: Opportunistic multi-hop routing for wireless networks. In *SIGCOMM’05*, Philadelphia, Pennsylvania, Aug. 2005.
- [13] Aggelos Bletsas, Ashish Khisti, David P. Reed, and Andrew Lippman. A simple cooperative diversity method based on network path selection. *IEEE J. SELECT. AREAS COMMUN*, 24:659–672, 2006.
- [14] M.A. Bonuccelli, G. Giunta, F. Lonetti, and F. Martelli. Real-time video transmission in vehicular networks. pages 115 –120, may. 2007.
- [15] Micah Z. Brodsky and Robert T. Morris. In defense of wireless carrier sense. In *ACM SIGCOMM ’09*, pages 147–158, 2009.
- [16] P. Buccioli, E. Masala, N. Kawaguchi, K. Takeda, and J.C. De Martin. Performance evaluation of h. 264 video streaming over inter-vehicular 802.11 ad hoc networks. In *PIMRC 2005*, pages 1936 –1940, Sept. 2005.
- [17] Tracy Camp, Jeff Boleng, and Vanessa Davies. A survey of mobility models for ad hoc network research. *Wireless Communications and Mobile Computing*, 2(5):483–502, 2002.

- [18] J. M. Del Castillo and F.G. Benitez. On the functional form of the speed-density relationship-i: general theory. *Transportation Research Part B: Methodological*, 29:373–389, October 1995.
- [19] Szymon Chachulski, Michael Jennings, Sachin Katti, and Dina Katabi. Trading structure for randomness in wireless opportunistic routing. *SIGCOMM CCR.*, 37(4):169–180, 2007.
- [20] Chen-Mou Cheng, H.T. Kung, Chit-Kwan Lin, Chia-Yung Su, and D. Vlah. Rainbow: A wireless medium access control using network coding for multi-hop content distribution. In *Military Communications Conference, 2008. MILCOM 2008. IEEE*, pages 1 –10, 2008.
- [21] P. Chou, Y. Wu, and K. Jain. Practical network coding. In *Allerton Conference on Communication, Control, and Computing*, Oct. 2003.
- [22] Douglas S. J. De Couto, Daniel Aguayo, John Bicket, and Robert Morris. A high-throughput path metric for multi-hop wireless routing. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 134–146, New York, NY, USA, 2003. ACM.
- [23] R. Dougherty, C. Freiling, and K. Zeger. Insufficiency of linear coding in network information flow. *Information Theory, IEEE Transactions on*, 51(8):2745 – 2759, 2005.
- [24] J. Ebrahimi and C. Fragouli. Vector network coding algorithms. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 2408 –2412, 2010.
- [25] A. Eryilmaz, D.S. Lun, and B.T. Swapna. Control of multi-hop communication networks for inter-session network coding. *Information Theory, IEEE Transactions on*, 57(2):1092 –1110, 2011.

- [26] Marco Fiore and Jose Maria Barcelo-Ordinas. Cooperative download in urban vehicular networks. In *IEEE MASS '09*, Oct. 2009.
- [27] C. Fragouli, J. Widmer, and J.-Y. Le Boudec. A network coding approach to energy efficient broadcasting: From theory to practice. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–11, April 2006.
- [28] Christina Fragouli, Jean-Yves Le Boudec, and Jorg Widmer. Network coding: an instant primer. *SIGCOMM Comput. Commun. Rev.*, 36(1):63–68, January 2006.
- [29] H. Fussler, J. Widmer, M. Kasemann, M. Mauve, and H. Hartenstein. Contention-based forwarding for mobile ad-hoc networks. *Elsevier's Ad Hoc Networks*, 1(4):351–369, Nov. 2003.
- [30] R. G. Gallager, P. A. Humblet, and P. M. Spira. A distributed algorithm for minimum-weight spanning trees. *ACM Trans. Program. Lang. Syst.*, 1983.
- [31] M. Ghaderi, D. Towsley, and J. Kurose. Reliability gain of network coding in lossy wireless networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 2171 –2179, 2008.
- [32] S. Gheorghiu, L. Lima, A.L. Toledo, J. Barros, and M. Medard. On the performance of network coding in multi-resolution wireless video streaming. In *Network Coding (NetCod), 2010 IEEE International Symposium on*, pages 1 –6, 2010.
- [33] C. Gkantsidis and P.R. Rodriguez. Network coding for large scale content distribution. In *INFOCOM 2005.*, volume 4, pages 2235 – 2245, March 2005.

- [34] Meng Guo, M.H. Ammar, and E.W. Zegura. V3: a vehicle-to-vehicle live video streaming architecture. In *PerCom '05*, pages 171 – 180, march 2005.
- [35] P. Gupta and P.R. Kumar. The capacity of wireless networks. *Information Theory, IEEE Transactions on*, 46(2):388 –404, March 2000.
- [36] H. Hartenstein and K.P. Laberteaux. A tutorial survey on vehicular ad hoc networks. *Communications Magazine, IEEE*, 46(6):164 –171, 2008.
- [37] S. Haykin. Cognitive radio: brain-empowered wireless communications. *Selected Areas in Communications, IEEE Journal on*, 23(2):201 – 220, 2005.
- [38] T. Ho, M. Medard, R. Koetter, D.R. Karger, M. Effros, Jun Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, Oct. 2006.
- [39] T. Ho and H. Viswanathan. Dynamic algorithms for multicast with intra-session network coding. *Information Theory, IEEE Transactions on*, 55(2):797 –815, 2009.
- [40] Tracey C. Ho, Yu-Han Chang, and Keesook J. Han. On constructive network coding for multiple unicasts, 2006.
- [41] Hugh Holbrook, Sandeep K. Singhal, and David R. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation. In *ACM SIGCOMM*, pages 328–341, 1995.
- [42] I-Hong Hou, Yu-En Tsai, T.F. Abdelzaher, and I. Gupta. AdapCode: Adaptive network coding for code updates in wireless sensor networks. *INFOCOM 2008.*, April 2008.
- [43] Bret Hull, Kyle Jamieson, and Hari Balakrishnan. Mitigating congestion in wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international*

- conference on Embedded networked sensor systems*, pages 134–147, New York, NY, USA, 2004. ACM Press.
- [44] Kamal Jain, Jitendra Padhye, Venkata N. Padmanabhan, and Lili Qiu. Impact of interference on multi-hop wireless network performance. *Wireless Networks*, 11:471–487, 2005.
 - [45] Kyle Jamieson and Hari Balakrishnan. Ppr: partial packet recovery for wireless networks. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 409–420, New York, NY, USA, 2007. ACM.
 - [46] D. Jiang, Qi Chen, and L. Delgrossi. Communication density: A channel load metric for vehicular communications research. In *IEEE MASS 2007*, pages 1–8, October 2007.
 - [47] Daniel Jiang and Luca Delgrossi. Ieee 802.11p: Towards an international standard for wireless access in vehicular environments. In *IEEE VTC Spring 2008*, pages 2036–2040, May 2008.
 - [48] Daniel Jiang, Vikas Taliwal, Andreas Meier, Wieland Holfelder, and Ralf Hertrich. Design of 5.9 ghz dsrc-based vehicular safety communication. *IEEE Wireless Communications*, 13(5):36–43, October 2006.
 - [49] David B. Johnson and David A. Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile Computing*, volume 353, pages 153–181. Springer US, 1996.
 - [50] Mark Johnson, Luca De Nardis, and Kannan Ramch. Collaborative content distribution for vehicular ad hoc networks. In *Allerton Conference on Communication, Control, and Computing*, September 2006.

- [51] Shirish Karande and Zheng Wang. On the multicast throughput capacity of network coding in wireless ad-hoc networks. In *FOWANC'09*, pages 21–27, May 2009.
- [52] S. Katti, H. Rahul, Wenjun Hu, D. Katabi, M. Medard, and J. Crowcroft. Xors in the air: Practical wireless network coding. *Networking, IEEE/ACM Transactions on*, 16(3):497–510, jun. 2008.
- [53] Sachin Katti, Shyamnath Gollakota, and Dina Katabi. Embracing wireless interference: analog network coding. In *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '07, pages 397–408, New York, NY, USA, 2007. ACM.
- [54] Sachin Katti, Dina Katabi, Hari Balakrishnan, and Muriel Medard. Symbol-level network coding for wireless mesh networks. In *SIGCOMM '08*, pages 401–412, 2008.
- [55] V. Kawadia and P.R. Kumar. A cautionary perspective on cross-layer design. *Wireless Communications, IEEE*, 12(1):3 – 11, 2005.
- [56] Roger G. Kermode. Scoped hybrid automatic repeat request with forward error correction (sharqfec). *SIGCOMM Comput. Commun. Rev.*, 28(4):278–289, 1998.
- [57] R. Koetter and M. Medard. An algebraic approach to network coding. *Networking, IEEE/ACM Transactions on*, 11(5):782 – 795, 2003.
- [58] Z. Kong, S.A. Aly, E. Soljanin, E.M. Yeh, and A. Klappenecker. Network coding capacity of random wireless networks under a signal-to-interference-and-noise-ratio model. In *Allerton '07*, 2007.

- [59] D. Koutsonikolas, Y-C. Hu, and C-C. Wang. High-throughput, reliable multi-cast without crying babies in wireless mesh networks. In *CoNEXT*. ACM, Dec 2008.
- [60] D. Koutsonikolas, Chih-Chun Wang, and Y.C. Hu. Ccack: Efficient network coding based opportunistic routing through cumulative coded acknowledgments. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, 2010.
- [61] P. Larsson. Selection diversity forwarding in a multihop packet radio network with fading channel and capture. In *SIGMOBILE Mob. Comput. Commun. Rev.*, volume 5 of 4, pages 47–54, 2001.
- [62] Seung-Hoon Lee, Uichin Lee, Kang-Won Lee, and M. Gerla. Content distribution in vanets using network coding: The effect of disk i/o and processing o/h. In *Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON '08. 5th Annual IEEE Communications Society Conference on*, pages 117–125, 2008.
- [63] Uichin Lee, Joon-Sang Park, Joseph Yeh, Giovanni Pau, and Mario Gerla. Code torrent: content distribution using network coding in vanet. In *MobiShare '06*, pages 1–5, 2006.
- [64] Ming Li, Wenjing Lou, and Kai Zeng. Oppcast: Opportunistic broadcast of warning messages in vanets with unreliable links. In *IEEE MASS'09*, Oct. 2009.
- [65] Ming Li, Zhenyu Yang, and Wenjing Lou. Codeon: Cooperative popular content distribution for vehicular networks using symbol level network coding. *Technical Report*, 2010.

- [66] Ming Li, Zhenyu Yang, and Wenjing Lou. Codeon: Cooperative popular content distribution for vehicular networks using symbol level network coding. *To be appear on IEEE Journal on Selected Areas in Communications(JSAC)*, 2011.
- [67] Qiming Li, Dah-Ming Chiu, and J.C.S. Lui. On the practical and security issues of batch content distribution via network coding. In *Network Protocols, 2006. ICNP '06. Proceedings of the 2006 14th IEEE International Conference on*, pages 158–167, 2006.
- [68] S.-Y.R. Li, R.W. Yeung, and Ning Cai. Linear network coding. *Information Theory, IEEE Transactions on*, 49(2):371–381, 2003.
- [69] Xiang Yang Li, Shao Jie Tang, and Ophir Frieder. Multicast capacity for large scale wireless ad hoc networks. In *IEEE MobiCom'07*, pages 266–277, 2007.
- [70] Junning Liu, D. Goeckel, and D. Towsley. Bounds on the gain of network coding and broadcasting in wireless networks. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 724–732, May 2007.
- [71] Junning Liu, Dennis Goeckel, and Don Towsley. The throughput order of ad hoc networks employing network coding and broadcasting. In *IEEE MILCOM 2006*, 2006.
- [72] Desmond S. Lun, Muriel Medard, Ralf Koetter, and Michelle Effros. On coding for reliable communication over packet networks. *Physical Communication*, 1(1):3–20, 2008.
- [73] D.S. Lun, M. Medard, Tracey C. Ho, and R. Koetter. Network coding with a cost criterion, 2004.

- [74] D.S. Lun, M. Medard, R. Koetter, and M. Effros. Further results on coding for reliable communication over packet networks. In *Information Theory, 2005. ISIT 2005. Proceedings. International Symposium on*, pages 1848–1852, 2005.
- [75] D.S. Lun, N. Ratnakar, M. Medard, R. Koetter, D.R. Karger, T. Ho, E. Ahmed, and F. Zhao. Minimum-cost multicast over coded packet networks. *Information Theory, IEEE Transactions on*, 52(6):2608–2623, 2006.
- [76] Ya lun Chou. *Statistical Analysis*. Holt International, 1975.
- [77] T.K. Mak, K.P. Laberteaux, R. Sengupta, and M. Ergen. Multichannel medium access control for dedicated short-range communications. *IEEE Transactions on Vehicular Technology*, 58(1):349–366, Jan. 2009.
- [78] Niga Zamalloa Marco Zú and Bhaskar Krishnamachari. An analysis of unreliability and asymmetry in low-power wireless links. *ACM Trans. Sen. Netw.*, 3(2):7, 2007.
- [79] Allen Miu, Hari Balakrishnan, and Can E. Koksul. In *MobiCom '05: Proceedings of the 11th annual international conference on Mobile computing and networking*, New York, NY, USA.
- [80] Alok Nandan, Shirshanka Das, Giovanni Pau, Mario Gerla, and M. Y. Sanadidi. Co-operative downloading in vehicular ad-hoc wireless networks. In *WONS '05*, pages 32–41, 2005.
- [81] Alok Nandan, Saurabh Tewari, Shirshanka Das, Mario Gerla, and Leonard Kleinrock. AdTorrent: Delivering Location Cognizant Advertisements to Car Networks. In *WONS '06*, pages 203–212, 2006.

- [82] M. Nekoui, A. Eslami, and H. Pishro-Nik. Scaling laws for distance limited communications in vehicular ad hoc networks. In *Communications, 2008. ICC '08. IEEE International Conference on*, pages 2253–2257, May 2008.
- [83] Ping Chung Ng and Soung Chang Liew. Throughput analysis of ieee802.11 multi-hop ad hoc networks. *IEEE Transactions on Networking*, 15(2):309–322, April 2007.
- [84] J. Nonnenmacher, E.W. Biersack, and D. Towsley. Parity-based loss recovery for reliable multicast transmission. *Networking, IEEE/ACM Transactions on*, 6(4):349–361, Aug 1998.
- [85] Elena Pagani and Gian Paolo Rossi. Reliable broadcast in mobile multihop packet networks. In *MobiCom '97: Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking*, pages 34–42, New York, NY, USA, 1997. ACM.
- [86] Joon-Sang Park, M. Gerla, D.S. Lun, Yunjung Yi, and M. Medard. Codecast: a network-coding-based ad hoc multicast protocol. *Wireless Communications, IEEE*, 13(5):76–81, October 2006.
- [87] Joon-Sang Park, Uichin Lee, Soon Y. Oh, Mario Gerla, and Desmond S. Lun. Emergency related video streaming in vanet using network coding. In *VANET '06*, 2006.
- [88] Charles E. Perkins, Elizabeth M. Royer, and Samir R. Das. Ad hoc on-demand distance vector (aodv) routing. In *2nd IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, February 1999.
- [89] John G. Proakis. Digital communications, August 2000.

- [90] N. Qadri, M. Altaf, M. Fleury, M. Ghanbari, and Hanadi Sammak. Robust video streaming over an urban vanet. *IEEE Wireless and Mobile Computing, Networking and Communication*, pages 429–434, 2009.
- [91] K. Ramachandran, M. Gruteser, R. Onishi, and T. Hikita. Experimental analysis of broadcast reliability in dense vehicular networks. *IEEE Vehicular Technology Magazine*, 2(4):26–32, Dec. 2007.
- [92] A. Ramamoorthy, J. Shi, and R.D. Wesel. On the capacity of network coding for random networks. *Information Theory, IEEE Transactions on*, 51(8):2878–2885, 2005.
- [93] A.K. Ramasubramonian and J.W. Woods. Multiple description coding and practical network coding for video multicast. *Signal Processing Letters, IEEE*, 17(3):265–268, mar. 2010.
- [94] Luigi Rizzo and Lorenzo Vicisano. RMDP: an fec-based reliable multicast protocol for wireless environments, 1998.
- [95] H. Seferoglu and A. Markopoulou. Video-aware opportunistic network coding over wireless networks. *IEEE Journal on Selected Areas in Communications(JSAC)*, 27(5):713–728, June 2009.
- [96] R. C. Shah, A. Bonivento, D. Petrovic, E. Lin, J. van Greunen, and J. Rabaey. Joint optimization of a protocol stack for sensor networks. In *IEEE Milcom*, Nov. 2004.
- [97] A. Sobeih, H. Baraka, and A. Fahmy. ReMHoc: a reliable multicast protocol for wireless mobile multihop ad hoc networks. In *Consumer Communications and Networking Conference, 2004. CCNC 2004. First IEEE*, pages 146–151, Jan. 2004.

- [98] J.L. Sobrinho and A.S. Krishnakumar. Quality-of-service in ad hoc carrier sense multiple access wireless networks. *Selected Areas in Communications, IEEE Journal on*, 17(8):1353–1368, August 1999.
- [99] F. Soldo, C. Casetti, C.-F. Chiasserini, and P. Chaparro. Streaming media distribution in vanets. *GLOBECOM '08*, pages 1–6, 2008.
- [100] Vikas Taliwal, Daniel Jiang, Heiko Mangold, Chi Chen, and Raja Sengupta. Empirical determination of channel characteristics for dsrc vehicle-to-vehicle communication. In *ACM VANET '04*. ACM, 2004.
- [101] Wee Lum Tan, Wing Cheong Lau, OnChing Yue, and Tan Hing Hui. Analytical models and performance evaluation of drive-thru internet system. *Selected Areas in Communications, IEEE Journal on*, 29(1):207–221, 2011.
- [102] Bulent Tavli. Broadcast capacity of wireless networks. *IEEE Communications Letters*, 10(2):68–69, 2006.
- [103] M. Torrent-Moreno, F. Schmidt-Eisenlohr, H. Fussler, and H. Hartenstein. Effects of a realistic channel model on packet forwarding in vehicular ad hoc networks. In *IEEE WCNC*, pages 385–391, 2006.
- [104] D. Traskov, N. Ratnakar, D.S. Lun, R. Koetter, and M. Medard. Network coding for multiple unicasts: An approach based on linear optimization. In *Information Theory, 2006 IEEE International Symposium on*, pages 1758–1762, 2006.
- [105] Hong Shen Wang and Nader Moayeri. Finite-state markov channel-a useful model for radio communication channels. *IEEE Transactions on Vehicular Technology*, 44(1):163–171, February 1995.

- [106] Mea Wang and Baochun Li. R2: Random push with random network coding in live peer-to-peer streaming. *IEEE Journal on Selected Areas in Communications(JSAC)*, 25(9):1655–1666, December 2007.
- [107] M.M. Wang, Weimin Xiao, and T. Brown. Soft decision metric generation for qam with channel estimation error. *Communications, IEEE Transactions on*, 50(7):1058–1061, jul 2002.
- [108] Zheng Wang, Hamid R. Sadjadpour, and Garcia-Luna-Aceves J.J. A unifying perspective on the capacity of wireless ad hoc networks. In *INFOCOM 2008, IEEE*, pages 753–761, April 2008.
- [109] Geoffrey Werner-Allen, Konrad Lorincz, Matt Welsh, Omar Marcillo, Jeff Johnson, Mario Ruiz, and Jonathan Lees. Deploying a wireless sensor network on an active volcano. *IEEE Internet Computing*, 10:18–25, March 2006.
- [110] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the h.264/avc video coding standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, 13(7):560–576, 2003.
- [111] B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *MobiHoc*, pages 194–205. ACM, 2002.
- [112] N. Wisitpongphan, Fan Bai, P. Mudalige, V. Sadekar, and O. Tonguz. Routing in sparse vehicular ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications(JSAC)*, 25(8):1538–1556, Oct. 2007.
- [113] Y. Wu, P.A. Chou, and K. Jain. A comparison of network coding and tree packing. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, page 143, june-2 july 2004.

- [114] Y. Wu, P.A. Chou, and Sun-Yuan Kung. Minimum-energy multicast in mobile ad hoc networks using network coding. *Communications, IEEE Transactions on*, 53(11):1906 – 1918, 2005.
- [115] Yunnan Wu. Distributing layered content using network coding. In *Sensor, Mesh and Ad Hoc Communications and Networks Workshops, 2008. SECON Workshops '08. 5th IEEE Annual Communications Society Conference on*, pages 1 –4, 2008.
- [116] Yunnan Wu, Mung Chiang, and Sun-Yuan Kung. Distributed utility maximization for network coding based multicasting: A critical cut approach. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006 4th International Symposium on*, pages 1 – 6, 2006.
- [117] Qing Xu, Tony Mak, Jeff Ko, and Raja Sengupta. Vehicle-to-vehicle safety messaging in dsrc. In *ACM VANET '04*, pages 19–28, 2004.
- [118] Zhenyu Yang, Ming Li, and Wenjing Lou. Codeplay: Live multimedia streaming in vanets using symbol-level network coding. In *Network Protocols, 2010. (ICNP '2010) Proceedings. Eighteenth International Conference on*, November 2010.
- [119] Zhenyu Yang, Ming Li, and Wenjing Lou. R-code: Network coding-based reliable broadcast in wireless mesh networks. *Ad Hoc Networks*, In Press, Corrected Proof:–, 2010.
- [120] Wei Ye, J. Heidemann, and D. Estrin. An energy-efficient mac protocol for wireless sensor networks. In *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2002.

- [121] Lin Yunfeng, Baochun Li, and Liang Ben. CodeOR: Opportunistic routing in wireless mesh networks with segmented network coding. *Network Protocols, 2008. ICNP 2008. IEEE International Conference on*, pages 13–22, Oct. 2008.
- [122] Kai Zeng, Wenjing Lou, Jie Yang, and D. Richard Brown. On throughput efficiency of geographic opportunistic routing in multihop wireless networks. In *QShine'07*, Vancouver, British Columbia, Canada, August 2007.
- [123] Kai Zeng, Wenjing Lou, and Hongqiang Zhai. On end-to-end throughput of opportunistic routing in multirate and multihop wireless networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 816 –824, 2008.
- [124] Kai Zeng, Wenjing Lou, and Yanchao Zhang. Multi-rate geographic opportunistic routing in wireless ad hoc networks. In *Military Communications Conference, 2007. MILCOM 2007. IEEE*, pages 1 –7, 2007.
- [125] Chi Zhang, Xiaoyan Zhu, and Yuguang Fang. On the improvement of scaling laws for large-scale manets with network coding. *Selected Areas in Communications, IEEE Journal on*, 27(5):662–672, 2009.
- [126] Jin Zhang, Qian Zhang, and Weijia Jia. Vc-mac: A cooperative mac protocol in vehicular networks. *IEEE Transactions on Vehicular Technology*, 58(3):1561–1571, March 2009.
- [127] Shengli Zhang, Soung Chang Liew, and Patrick P. Lam. Hot topic: physical-layer network coding. In *Proceedings of the 12th annual international conference on Mobile computing and networking*, MobiCom '06, pages 358–365, New York, NY, USA, 2006. ACM.
- [128] Yang Zhang, Jing Zhao, and Guohong Cao. Roadcast: A popularity aware content sharing scheme in vanets. *IEEE ICDCS '09*, pages 223–230, 2009.

- [129] Jing Zhao, Todd Arnold, Yang Zhang, and Guohong Cao. Extending drive-thru data access by vehicle-to-vehicle relay. In *ACM VANET '08*, pages 66–75, 2008.
- [130] Jing Zhao, Yang Zhang, and Guohong Cao. Data pouring and buffering on the road: A new data dissemination paradigm for vehicular ad hoc networks. *IEEE Transactions on Vehicular Technology*, 56(6):3266–3277, Nov. 2007.
- [131] Baochun Li Shuqiao Zhao Zimu Liu, Chuan Wu. Uusee: Large-scale operational on-demand streaming with random network coding. In *INFOCOM '10*, 2010.
- [132] M. Zorzi and R. R. Rao. Geographic random forwarding (geraf) for ad hoc and sensor networks: energy and latency performance. *IEEE Transactions on Mobile Computing*, 2(4), 2003.