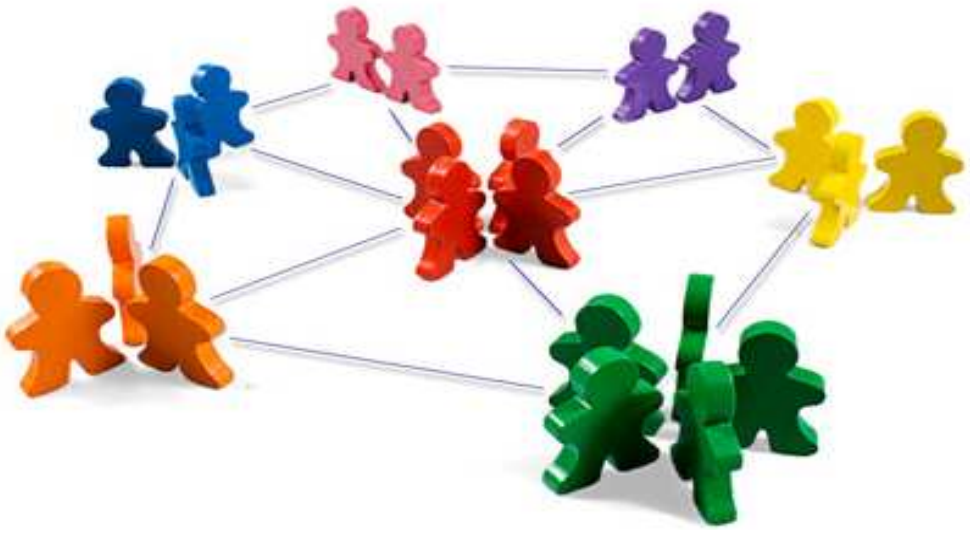
TU Delft



MSc Thesis

Network Coded Flooding

Assignment: Master of Science Thesis
Instructor: Ir. J. Goseling
Supervisor: Dr.Ir.J.H.Weber
Date: 19 June 2009
Student: Haiyan Wang

Network Coded Flooding

Name: Haiyan Wang
Student number: 1385356
Date: 19 June 2009

Instructor: Ir. J. Goseling
Supervisor: Dr.Ir.J.H.Weber

A thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

presented at

Delft University of Technology,
Faculty of Electrical Engineering,
Mathematics, and Computer Science,
Department of Telecommunications,
Wireless and Mobile Communication (WMC) Group.

June 2009

Preface

My master thesis is done at the Wireless and Mobile Communication (WMC) Group, Faculty of Electrical Engineering, Mathematics, and Computer Science (EEMCS), Delft University of Technology, on which I worked from August 2008 to June 2009. The work presented in this thesis is done independently, but it could not be accomplished without the guidance and aid of multitude of individuals.

Many of you have had significant influences on me during my time at Delft University of Technology, in a variety of ways, both academic and personal. I express my sincere appreciation to all of you who have given me encouragement and help during the completion of this study, and I hope that I can repay you in some ways as I am able. I would like to single out the following people who had a major impact on me.

I would especially like to express my gratitude to my supervisor, Dr. ir. J. H. Weber, for providing me invaluable guidance, advice, support and criticism. It was because of him that my master studies were so enjoyable and so intellectually rewarding. He has taught me not only how to do research, but how to write papers. His wide knowledge and great foresight have provided a good basis for the present thesis. His knowledge and instructions helped me to conquer a lot of challenges in the process of my thesis. Furthermore, I would like to thank him for giving such perfect and useful lectures, especially in “Error Correcting Codes”. Besides, I would also like to express my appreciation to him for encouraging and helping me when I met some serious difficulties in my personal life.

I would also like to thank my daily mentor, Ir. J. Goseling, for the weekly discussions on temporal reasoning that were both enjoyable and useful; for giving me positive and important suggestions even if I made mistakes instead of criticizing me like ordinary people. He has been a great source of information and his ideas and suggestions have really helped me in my research. Moreover, although Jasper does not work in TUD every day, he has been available to help me whenever I need, no matter how busy he is.

In addition, I want to express my appreciation to my mother and my friends who have given me encouragement and happiness, especially when I was in difficult situation.

List of Abbreviations

ADBP	Ad Hoc Broadcast Protocol
ARP	Address Resolution Protocol
CBS	Counter-Based Scheme
CD	Collision Detection
CSMA	Carrier Sense Multiple Access
CTS	Clear To Send
DBS	Distance-Based Scheme
DTN	Delay-Tolerant Network
GF	Galois Field
ID	Identity
IFq	Interface Queue
LBS	Location-Based Scheme
LENWB	The Lightweight and Efficient Network-Wide Broadcast Protocol
LL	Link Layer
MAC	Medium Access Control
MPR	Multi-Point Relaying
NCBA	Network Coding Broadcast Algorithm
NCF	Network Coded Flooding
NetIF	Network Interface
NS-2	Network Simulator 2
PDR	Packet Delivery Ratio
PFA	Probabilistic Flooding Algorithm
PHY	Physical
RP	Routing Protocol
RTS	Request To Send
SBA	Scalable Broadcast Algorithm

Abstract

Prior work of network coding is mainly focusing on multicast traffic. In this thesis, we propose a new network coding based communication algorithm called Network Coded Flooding (NCF) which is related to network wide broadcast. This designed algorithm is an integration of network coding and one of the commonly used broadcasting techniques in wireless networks.

In this thesis, we choose Probabilistic Flooding Algorithm (PFA) to integrate with network coding since it is a simple and robust flooding algorithm; it can be used in random wireless networks; it can work without any network topology information. As with PFA, NCF has a parameter of rebroadcast probability that controls packets' rebroadcasts when receiving innovative packets.

During the process of designing NCF, we also consider the issues how efficient network coding can achieve in a random wireless network even if the system process ability is low and available memory space is also limited. Therefore, the idea of generation is used and we also propose a specific generation management method in this thesis that is able to let system occupy little system memory space while good network performance (such as successful packet delivery, low packet delay and great energy savings) and relatively low system process complexity are guaranteed.

NCF is a practical network coding based flooding algorithm that can be used in random wireless networks; that does not need any network topology information; that huge amount of data is allowed to be transmitted during the communication process; that the requirements of buffering and network process ability are at a relatively low level.

We simulate such algorithm in Network Simulator 2 (NS2), and the simulation results show that NCF can realize the benefits in terms of reliability, working efficiency and energy saving if related parameters (such as generation size or maximum number of generations per node has) of NCF are set accurately. In addition, reasonable trade-off schemes are also given through analyzing the obtained simulation results, which give general ideas about how to accurately use and change the related parameters of NCF in order to efficiently balance the relationship between network requirements and network performance.

Key words: Broadcasting; Flooding Algorithm; Generation Management Method; Network coding; Wireless Networks.

TABLE OF CONTENTS

Preface 5

List of Abbreviations 7

Abstract 9

1. Introduction 13

1.1 Motivation 13
1.1.1 Benefits of Network Coding 13
1.1.2 Flooding the networks 17

1.2 Related Work 18
1.2.1 Work Related to the Problem of Wireless Broadcasting 18
1.2.2 Open Problems in Related Work 19

1.3 Research Goal 19

1.4 Assignments 21

1.5 Thesis Overview 21

2. Flooding Algorithms 23

2.1 Blind Flooding Protocol 23

2.2 Mechanisms to Avoid Broadcast Storm 24
2.2.1 Overview of Different Schemes 24
2.2.2 Probabilistic Flooding Algorithm 26

3. Introduction to Network Coding 29

3.1 Main Network Coding Theorem 29
3.1.1 Linear Network Coding 30

3.2 Practical Network Coding 31

3.3 Previous Work of Flooding the Network using Network Coding 35
3.3.1 Network Coding Algorithm in DTNs [9] 35
3.3.2 Efficient Network Coding Broadcast Algorithm [10] 36

3.4 Limitations & Disadvantages 36

4. Network Coded Flooding Algorithm 41

4.1 System Model and Basic Functions 41

4.2 Packet Distributed Method 42

4.3 Generation Management Method in NCF 43
4.3.1 Generation List 45
4.3.2 Packet Format 48

4.4 Initial Sending, Receiving and Retransmission 49
4.4.1 Initial Sending Part 50
4.4.2 Receiving and Retransmission Part 51

4.5 Summary of NCF 54

5. Simulation Results and Analysis 57

5.1 Description of the Simulation 57

5.1.1 Simulation Scenario 57
5.1.2 Performance Metrics 59
5.2 Comparison of Three Algorithms 60
5.3 Analysis of NCF compared with PFA 67
5.3.1 Impact of Generations' Amount and Sending Rate 68
5.3.2 Impact of Rebroadcast Probability and Sending Rate 71
5.3.3 Impact of Network Density and Sending Rate 73
5.4 Trade-Off Schemes 76
5.5 Summary of Simulation Study 79
6. Conclusion and Future Work 81
6.1 Conclusion 81
6.2 Future Work 82
6.2.1 Unsolved Problems 82
6.2.2 Recommendations 84
References 87

1. Introduction

Network coding is such a new and significant technique which breaks the traditional way of packet transmission and has sufficient ability to improve network performance. We are confident that network coding will be an essential technology in the future network and will be used into practical applications soon. However, before this dream coming true, a prerequisite how to make network coding really used in today' networks by combining with current network protocols and then work efficiently in real applications must be carefully considered and solved.

The thesis is exactly focusing on one aspect of this problem. One commonly used network protocol, *flooding protocol*, is chosen, and we are aiming at designing a new algorithm which can integrate network coding with such protocol; in addition, this new algorithm should be able to further improve network performance.

In this introduction chapter, a motivation is presented firstly which includes the reason why we are interested in network coding; why a flooding protocol is chosen in our case. Then the previous works that related to the problem of network coding based broadcasting are briefly introduced. Thereafter, the contribution of our research work, a designed network coding based broadcast algorithm, called Network Coded Flooding (NCF) is briefly introduced including its features. Related assignments and an outline of this thesis report are also clearly shown at the end of this chapter.

1.1 Motivation

The reason why we are interested in the topic of NCF is that we realize the significant potentials and benefits of network coding; besides, when main research works of network coding are focusing on multicast traffic, it is also much necessary to do researches on broadcast traffic since broadcasting plays an important role in networks, for example, it is useful for building network protocols' blocks.

Here, the explanations of our motivation of doing this topic are presented from two main points of views:

1) Why network coding? (Section 1.1.1);

2) Why flooding the networks? (Section 1.1.2).

1.1.1 Benefits of Network Coding

Network coding is a new research field in information theory [1]. To some extent, it breaks the traditional packet transmission method. Using network coding, nodes are allowed to process the received incoming packets instead of simply forwarding or

repeating them; and thus, packets which are independently generated by their sources are not needed to be kept separately any more.

Due to the appealing property, network coding is able to offer benefits in various aspects of communication networks. In the following, we will introduce several main advantages of network coding. Terminologies used in the thesis are presented in the following table (Table 1.1).

Table 1.1: Terminologies used in this thesis.

Terminology	Definitions
$G=(V,E)$	A network with the set of nodes V and the set of edges E.
V	A set of nodes in the certain network.
$v_i \in V$ $(i=1,2,...,\|V\|)$	A random node in the network.
S ($S \subseteq V$)	A set of sources in the certain network.
$s_j \in S$ $(j=1,2,...,\|S\|)$	A random source node in the network
T ($T \subseteq V$)	A set of receivers in the certain network.
$t_n \in T$ $(n=1,2,...,\|T\|)$	A random receiver node in the network
M	A set of transmitted messages.
$m_k \in M$ $(k=1,2,...,\|M\|)$	A random transmitted message.

- **Throughput [1] [2] [3] [4] [5]**:

  Before we explain the benefit of throughput that network coding can provide, it is necessary to explain Max-flow Min-cut Theorem firstly [1] [2] [4] [5] which is related to a single-session *unicast* case.

Definition [5]:	Considering a single-session case, a node $s \in V$ wants to transmit messages to a node $t \in V$. A *cut* between s and t is a set of edges whose removal disconnects s from t. The *value* of the *cut* is the sum of the capacities of the edges in the cut, which is denoted as $rate(s,t)$. A *min-cut* is a cut with the smallest (minimal) value which is designated as $\min-cut(s,t)$. The minimum of the values of all such cuts between s and t is an upper bound on $rate(s,t)$, ($rate(s,t) \le \min-cut(s,t)$).

Max-flow Min-cut Theorem [2] [4] [5]:	Considering a single-session *unicast* case, for undirected graphs with unit capacity edges, there always exists a set of $h=\min-cut(s,t)$ paths between s and t. Thus, by routing information over this set of h unit-capacity paths, reliable communication can be achieved between s and t at the maximum possible rate, $\max- flow(s,t)=\min-cut(s,t)=h$.

In the single-session *broadcast* case, a node $s \in V$ wants to transmit messages to all the other nodes in the network (except for itself). Let $rate(s,V)$ be an achievable rate at which s can broadcast its messages in the network. Thus an upper bound on $rate(s,V)$ is $\min_{v_i \in V} \min-cut(s,v_i)$. In 1972, Edmonds [2] proved by routing information through certain paths in the network, reliable broadcast from s to V (except for s) can be achieved at the maximum possible rate, $\max- flow(s,V) = \min_{v_i \in V} \min-cut(s,v_i)$.

In a single-session *multicast* case where a single sender s wants to transmit messages to a set of receivers in the network $T \subseteq V$. Let $rate(s,T)$ be an achievable rate at which s can multicast its messages to the nodes belonging to the set of T. The upper bound on $rate(s,T)$ is still equal to the value of minimal cut, that is $rate(s,T) \le \min_{t_n \in T} \min-cut(s,t_n)$. However, this upper bound of $rate(s,T)$ cannot always be achieved when multicasting (more information can be found in [1] [2] [4]).

Fortunately, reliable multicast from s to $T \subseteq V$ can be realized at the upper bound ($\max- flow(s,T) = \min_{t_n \in T} \min-cut(s,t_n)$) if network coding is used which was proved by Alswede et al. in [1].

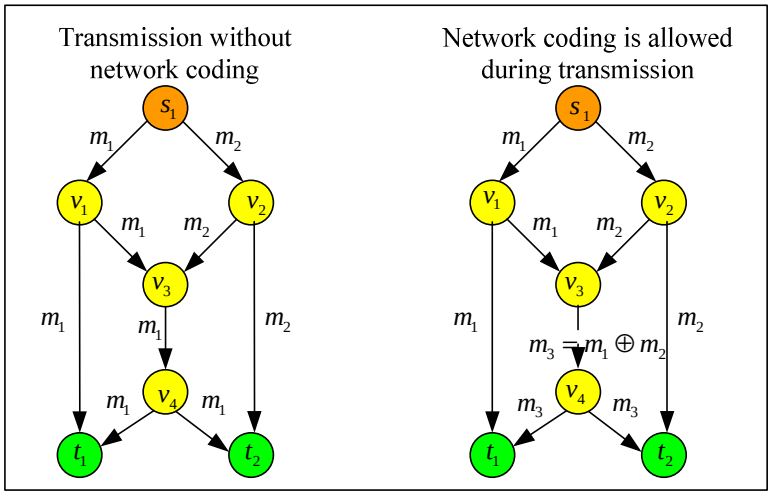


Figure 1.1: A single-source two-receiver network. Left: transmission without network coding; right: network coding is allowed during transmission [3].

A simple example of Figure 1.1 shows the basic operation of network coding when multicasting. It is a one-source (s) and two-receiver (t_1 and t_2) network with a butterfly topology. We assume the capacity of each edge in the network is one, such that the value of a cut is equal to the number of edges in the cut. It is easy to check in the example that the value of a max-flow of each sink is 2.

Without network coding, after two packets m_1, m_2 originate at their source s simultaneously, the path from node v_3 to node v_4 is overlapped which means both two receivers t_1, t_2 should share the network resources and thus the

communication rate has to be reduced [5]. In the example, we assume packet m_1 wins the contention and passes the contended path firstly, such that for receiver t_1, only packet m_1 can be received at that moment; meanwhile, for receiver t_2, both packets m_1, m_2 can be received. Therefore, the maximum possible rate of 2 cannot be achieved; instead, the average throughput of traditional packet transmission way is only 1.5.

On the contrary, if we allow information to be coded at node v_3 which means it is unnecessary for both packets m_1, m_2 to contend at the overlapping path (from node v_3 to node v_4); instead, m_1, m_2 can pass the path simultaneously based on a special encoding way (in this example, modulo 2 addition is used) , such that the performance of throughput is improved. For receiver t_1, m_2 can be recovered from the two received packets m_1 and m_3; similarly, m_1 can also be recovered by receiver t_2. Through network coding solutions, the average throughput achieves at 2.

In conclusion, regarding as the Max-flow Min-cut Theorem, reliable multicast can be realized at the max-flow information rate only if network coding is allowed to be used during the communication process.

- **Time Efficiency:**

Network coding can also save the wireless resources, such as packet delay. A simple example in Figure 1.2 can clearly explain such advantage.

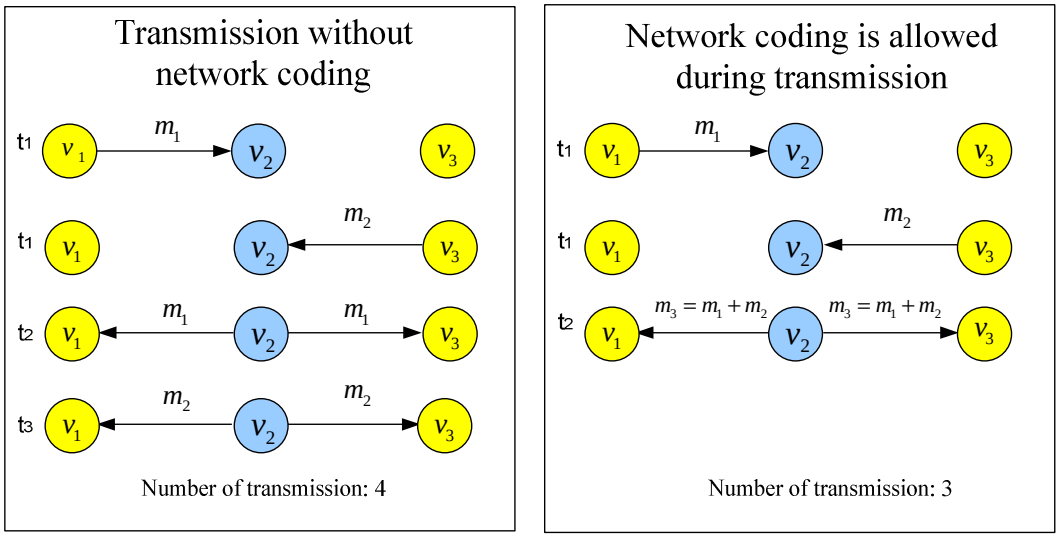


Figure 1.2: Example: network coding's benefits of efficiently using wireless resources (energy and delay) [3].

In Figure 1.2, it is a wireless ad-hoc network where node v_1 and node v_3 want to exchange their information called m_1, m_2 respectively via the relay node v_2. We assume that time is slotted and one transmission or receiving is occurred and finished during one timeslot. We can observe in the example (left), three time

slots are needed to finish the goal of information exchanging given that packet transmission follows the traditional way (simply forwarding packets without network coding); on the contrary, if network coding is allowed during communication (right), after receiving both packets from two senders, the intermediate node v_2 rebroadcasts a combined packet m_3, such that the goal of information exchanging can be reached only during two time slots which is less than the traditional transmission way.

- **Energy Efficiency:**

  Moreover, network coding promises to offer the benefit of energy saving which is one of the most important advantages that network coding can provide; furthermore, it is much meaningful for practical network communications.

  According to the same example in Figure 1.2, network coding' energy efficiency can be clearly shown. If network coding is allowed to be used, after node v_1 and v_3 successfully receiving and decoding the information from each other, only one transmission from node v_2 is needed to finish information exchanging instead of twice, and thus the energy is saved around 33% (total amount of transmissions is 3 through using network coding instead of 4 if packets are transmitted in traditional transmission way).

- **Other benefits:**

  Besides, network coding is able to offer benefits along other different dimensions of network communications such as security, complexity and resilience to link failures [3] [5].

Briefly speaking, network coding is the combination of transmission, encoding and re-encoding of packets arriving at nodes, such that the transmitted messages can be recovered at their destinations provided that there are sufficient innovative packets received by receivers in the network [4]. Therefore, based on the unique operation way and potentials of network coding, the issue how to bridge theory with practices; in other words, how to deliberately use network coding in real applications, and further discover new fields that network coding can be used is becoming more and more interesting and essential.

1.1.2 Flooding the networks

Prior research on network coding is mainly related to multicast traffic; but network wide broadcast (called broadcast traffic later in this thesis) is also an important part in network communications. Therefore, in this thesis, we change the current research direction of network coding to a different field that is broadcast traffic. With broadcast traffics, a message sent from its source should be received by all the other nodes in the same network, such that the purpose of flooding the network can be achieved.

We are confident that network coding will also significantly display its benefits and efficiently improve network performance only if suitable network coding based broadcasting algorithm is designed and used.

1.2 Related Work

As mentioned in the above section, we are aiming at designing an algorithm which is an integration of network coding and one of the current flooding algorithms used in wireless networks. Therefore, before explaining our research work, related work need to be investigated. We divide the related work into two parts which are presented in Section 1.2.1 and 1.2.2 respectively.

1.2.1 Work Related to the Problem of Wireless Broadcasting

- **Flooding Algorithms:**

The commonly used broadcasting techniques are mainly divided into four families [6]. The simplest algorithm called blind flooding protocol causes the problem of broadcast storm [7], and thus leads to the consumption of a large amount of system overhead**.** Therefore, several improved algorithms are proposed in order to alleviate the influence of broadcast storm [6] [7].

In our case, we are interested in such flooding algorithms that no network topology information is needed when it works in network communications. There are several candidates that satisfy with this requirement such as Counter-Based-Scheme (CBS) and probabilistic scheme [8].

- **Network Coding for Wireless Broadcasting:**

There are two related algorithms that are exactly related to our research area. Both of them do work without any network topology information.

In [9], a Network Coding based Protocol for Delay-Tolerant Networks (DTN) is proposed which is an exact communication algorithm that is similar to Probabilistic Flooding Algorithm (PFA) but is based on network coding. Good network performance and energy efficiency can be achieved using their algorithm.

Recently, another similar algorithm, called Network Coding Broadcast Algorithm (NCBA) [10] is proposed. A very simple distributed algorithm is proposed based on CBS and allows realizing network coding's benefits, especially energy efficiency. The properties of NCBA include that it can be used in random wireless networks where network can work well without any information about the network topology; furthermore, network resources (e.g. energy consumption) can be efficiently saved.

1.2.2 Open Problems in Related Work

According to the related simulation results in the literature [9] [10], the algorithms do get benefits of network performance (e.g. packet delivery ratio (PDR), packet delay, energy saving) against the commonly used broadcast techniques which are based on the traditional packet transmission way. However, there are some disadvantages and unpractical points that are still unsolved (we will discuss the related algorithms in Chapter 3):

1) The number of packets that can be transmitted during the communication have to be seriously limited;

2) System buffer has to bear heavy burden since all the transmitted data needs to be stored together for decoding;

3) Although in both related algorithms the idea of generations is used or mentioned in order to decrease the size of decoding matrix, the property of generation is still not apparently displayed and both algorithms are still not practical enough: for the algorithm in [9], the system process complexity is high with the increasing size of generation; for the algorithm in [10], synchronization is still needed to organize packets' transmissions; besides, no generation management method is really used [10], and all the received packets are still kept together during the simulation process;

1.3 Research Goal

As mentioned in the last section, although a better network performance can be achieved and no network neighborhood information is needed through using network coding based algorithms, they are still impossible to be used in practical scenarios because there are several limitations: such algorithms cannot be used in random wireless networks and the type of network should be carefully controlled; the total number of packets that could be transmitted during the communication process has to be limited; huge amount of data needs to be stored together which occupies too much network memory space; the system computation complexity is becoming higher and higher as with the increasing size of decoding matrix.

Therefore, what we need recently is a new and practical network coding based broadcast algorithm which can be used in any kind of wireless networks; achieve good network performance as the previous related algorithms do; meanwhile, the deficiencies existing in previous related work can be overcome.

We are exactly aiming at researching such kind of algorithm in order to improve current situation. The main contribution of our research is that we design such an algorithm called NCF.

In this thesis, evaluation and comparison for NCF and other related algorithms are mainly through investigating the performance metrics of PDR, end-to-end packet

received and decoded delay (packet delay), and network overhead (number of rebroadcast times).

PDR is measured as the percentage of the number of packets that can be received (for algorithms without network coding) or successfully decoded (for algorithms with network coding) [10]; packet delay is measured as the average time between the transmission of a packet by the original source and successfully received (for algorithms without network coding) or decoded (for algorithms with network coding) at a certain node; network overhead, which displays the number of retransmissions needed to achieve a certain PDR, is also investigated in this thesis (the detail definitions of these performance metrics are given Section 5.1.2). Moreover, the size of memory space that needed to be occupied and system process complexity are also necessary to be evaluated.

Here, we briefly give NCF's properties in the following and the detail contents of the proposed algorithm will be presented later (Chapter 4 and Chapter 5):

- NCF is the combination of network coding and one of the currently used flooding algorithms, PFA [8]; besides, based on PFA's property, network topology information is also unnecessary to be known in NCF;

- NCF can be used in real networks, where no synchronization is needed to control packet transmission and receiving; where packet losses and packet delay are allowed to happen…

- A new packet format and corresponding specific decoding method are proposed in order to support the specific operation way of NCF;

- Since the property of network coding is efficiently used, good network performance can be guaranteed (e.g. high value of PDR, short packet delay and low consumption of network overhead) through accurately making use of NCF's related parameters; moreover, through our research, we discover another new property of network coding: a remarkable reduction of packet delay can be achieved when the system sending rate is increased, which means network coding is more suitable in high-speed-transmission networks;

- We discuss how efficient network coding can be achieved even with little available memory. A new generation management method is proposed together with NCF. Therefore, a larger number of packets are allowed to be existed and transmitted in the network, while only small part of data has to be stored in the system memory space; in addition, although nodes in the network have to process huge amount of data, the network process complexity is still reasonable.

- Both parameters "generation size" and "number of generations (each node can keep simultaneously)" are set as variables in NCF in order to discover their influences; moreover, through analysis of simulation results, we propose general trade-off schemes for the purpose of conveniently and efficiently using NCF in different realistic situations.

1.4 Assignments

During the research process, in order to accomplish our goal, we have made a specific plan and divided our research work mainly into two parts including "theoretical part" and "implementation part":

- **Theoretical part:**

1) Reviewing currently used different flooding protocols especially PFA;

2) Reviewing the current network coding theory (what it is; how it works; which models the network and the nodes should have to satisfy network coding's special functions, and etc…);

3) Investigating existing algorithms related to the problem of network coding based broadcasting; analyzing and finding out their disadvantages and limitations;

4) Designing a new algorithm which can efficiently combine network coding with one of the current flooding protocols and overcome such disadvantages of former related algorithms;

- **Implementation part:**

1) Getting familiar with Network Simulator 2 (NS-2) which is a discrete event simulator targeted at networking research. New flooding protocols can be easily integrated in the framework in NS-2;

2) Learning the programming language C++ and Tcl;

3) Implementing the designed NCF algorithm;

4) Doing simulation of the designed protocol and analyzing the simulation results.

1.5 Thesis Overview

In the first part of the thesis, an overview of current flooding protocols, especially PFA (Chapter 2) is given; the concept of network coding theory and previous network coding broadcasting algorithms are presented in Chapter 3.

In the second part of the thesis, NCF is introduced specifically in Chapter 4 including its operation way and features; simulation results which are compared with PFA and NCBA are shown in Chapter 5; besides, specific analysis is also given in this chapter.

In the last chapter (Chapter 6), a conclusion of our research including the features, disadvantages and limitations are present; in addition, the future work is also presented.

2. Flooding Algorithms

As mentioned in the introduction chapter, we are interested in designing a network coding based flooding algorithm. Therefore, firstly, it is necessary to review and investigate the basic theories of flooding algorithms and network coding concepts respectively.

In this chapter, a brief review of blind flooding algorithm and its negative influences called broadcast storm [7] are presented. Several approaches have already been proposed for alleviation of broadcast storm problem, a brief review of which is also given here. Certainly, since we are focusing on such algorithms that do not need any topology information, one of the improvements called PFA is suitable for our case. Hence, a specific explanation of this probabilistic scheme including its advantages and the reason to be chosen is presented as well.

2.1 Blind Flooding Protocol

We are focusing on such a particular case that if a source in the network sends a message, all the other nodes should receive it. Therefore, flooding algorithms satisfy with our requirements. According to existing broadcasting techniques, the simplest flooding protocol is called "blind flooding" (Figure 2.1), and it is very feasible. No other parameters or network information is needed.

Blind Flooding Protocol
Assume a node v_i receives a packet called m_k
If the packet m_k is received by node v_i for the first time, it will forward this packet to its neighbour nodes once.
If the packet m_k has already been received by node v_i, it is defined as duplicated packet and dropped.

Figure 2.1: Blind flooding protocol [8].

Unfortunately, in [6] [7], the authors observe that serious problems, called "broadcast storm" are caused if flooding is done blindly:

- Because the radio propagation is omni-directional and a physical location may be covered by the transmission ranges of several hosts, many rebroadcasts are considered to be redundant. For example, when a node decides to rebroadcast a broadcast message to its neighbors, all its neighbors have already got the message such that these redundant packets must be dropped without contributing any useful help in packet transmissions and will also cause resource wasting.

- Heavy contention could exist because rebroadcasting hosts are probably close to each other. For instance, after a node broadcasts a message, if many of its

neighbours decide to rebroadcast the message, these transmissions (which are all from nearby nodes) may severely contend with each other.

- Collisions are more likely to occur because Request-To-Send (RTS) / Clear-To-Send (CTS) dialogue are inapplicable and the timing of rebroadcasts is highly correlated. Because of the lacks of RTS/CTS dialogue, and the absence of Carrier Sense Multiple Access (CSMA) (with Collision Detection (CD)), collisions are more likely to occur and cause more damage.

In order to alleviate these problems mentioned above, there are several options that can be remedies given in next section.

2.2 Mechanisms to Avoid Broadcast Storm

The key point to alleviate the broadcast storm problem is obviously to reduce useless retransmissions, so that the problem of broadcast storm such as redundancy, heavy contentions and collisions which are caused by blind flooding algorithm can be alleviated [6] [7].

2.2.1 Overview of Different Schemes

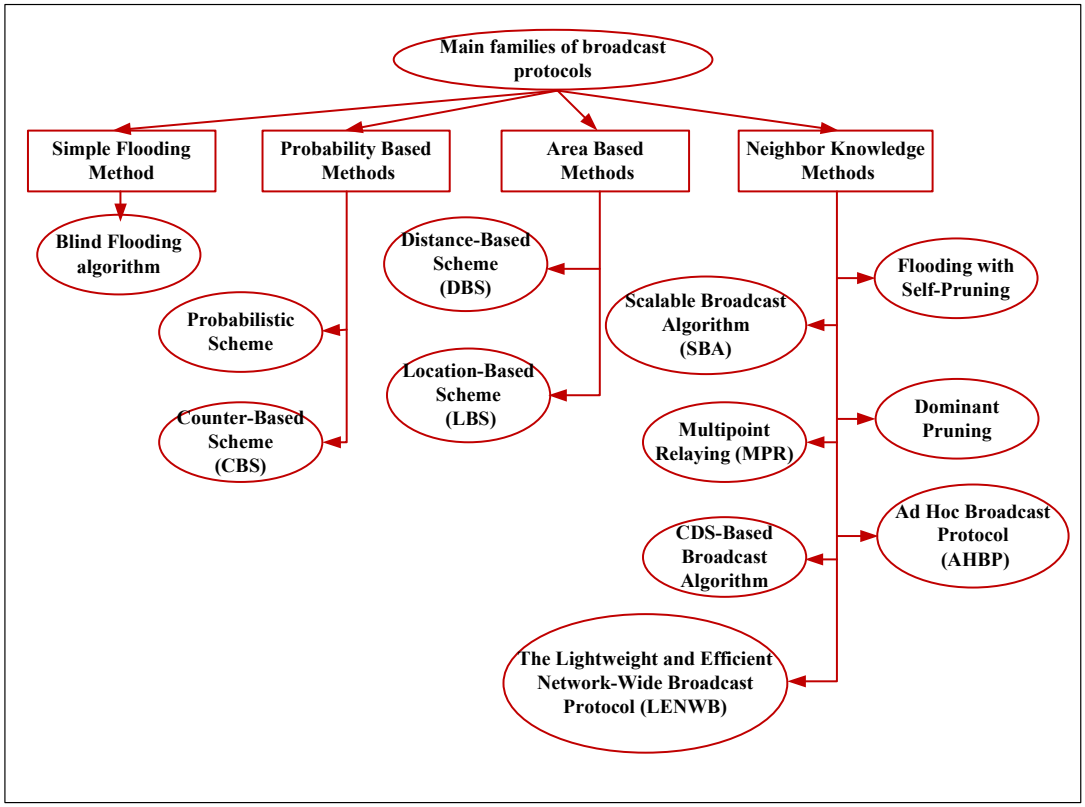


Figure 2.2: Classification of main broadcast protocols [6] [7].

The overview of various broadcasting techniques used in wireless networks are illustrated in Figure 2.2, which are mainly divided into four families and more details of specific operation way can refer to [6] [7]. Here, we only give the simple definitions of these four families [6] [7]:

1) Simple flooding method:

Blind flooding algorithm belongs to this family. The details of this algorithm have already been explained in Section 2.1.

2) Probability based methods:

It is also a relative simple family of broadcasting techniques. After receiving a new packet, one rebroadcast is activated by a node with an assigned probability and no network topology information is needed.

The value of probability can be decided depending on the density of the network. In a dense network area, the probability should be set relatively low in order to reduce the amount of useless retransmissions; in a sparse network area, more times of rebroadcasts are needed in order to guarantee good performance of packet delivery.

3) Area based methods:

Assume the transmission range of each node is identical. After receiving a new packet, a rebroadcast is activated by a node only if the rebroadcast will reach sufficient additional coverage area. The additional coverage area is evaluated based on all received redundant transmission. Depending on the operation way, network neighbors' information is necessary.

4) Neighbor knowledge methods:

Easily known from the name, network topology information is essential in this family. Nodes in the network have to keep updating their neighborhood states through "hello" packets which is used in the decision to rebroadcast after receiving a new packet.

In this thesis, as we need network to work without any topology information and also avoid the problem of broadcast storm, only the second family of broadcasting techniques, probability based methods, is satisfied with our requirement and can be chosen in our case.

According to Figure 2.2, there are two related schemes, probabilistic scheme and CBS belonging to this family:

- **Probabilistic Scheme:**

  Upon reception a new message, a rebroadcast is activated with a predetermined probability. If the probability is 100%, it is the same as blind flooding. Because

we are using probabilistic flooding as our distributed algorithm, more details of probabilistic scheme will be presented later (see Section 2.2.2).

- **Counter-Based Scheme:**

  Upon reception a new packet, the node initiates a counter with a value of one and set a small random delay before rebroadcasting the message. During the random delay, the counter is increased by one for each received duplicated packet. A counter threshold is chosen. If the value of the counter is larger than the threshold, a rebroadcast is avoided; otherwise, the packet is rebroadcast. This scheme is also simple and does not need knowledge about network topologies.

In this thesis, we choose probabilistic scheme (called PFA in our thesis) to integrate with network coding. Certainly, CBS can also be easily used in our case only through changing a small part of the NCF algorithm.

2.2.2 Probabilistic Flooding Algorithm

As explaining in the last section, PFA is a relatively simple flooding algorithm which is able to alleviate the broadcast storm problem; furthermore, it does not need any topology information. Exactly as its name, a rebroadcast is decided by a predetermined probability. An introduction of probabilistic routing is given in Figure 2.3.

.

Probabilistic Flooding Algorithm (PFA)
If a new packet originates at its source, it will be broadcast with probability $p = 100\%$.
Assume a node v_i receives a packet called m_k, and the rebroadcast probability is p:
If the packet m_k is received by node v_i for the first time, it will duplicate and forward it to its neighbor nodes with the probability p.
If the packet m_k has already been received by node v_i, it is defined as duplicated packet and has to be dropped.

Figure 2.3: Description of Probabilistic Flooding Algorithm [8].

Obviously, the influence of broadcast storm problems is reduced to some extent as one rebroadcast is activated only with a predetermined probability and then the number of redundant messages can be reduced efficiently. For example, in a dense network, several nodes are probably in the same transmission range; therefore, if some nodes do not retransmit their received innovative packets, the network overhead and thus the network resources will be saved without having any negative effect of packet delivery.

Although PFA has improved much against blind flooding algorithm due to the ability to alleviate the problem of broadcast storm; however, considering it still uses the traditional way of packet transmissions, neither the network performance (e.g. successful packet delivery) nor the situation of network overhead consumption are improved significantly. In addition, the information containing in each transmitted packet is still relatively less than the transmission way of network coding since nodes

are not allowed to process the received packets, instead, what they are only allowed is repeating or simply forwarding packets. Therefore, there is still enough space to improve the performance of PFA in wireless communications if network coding can be combined.

3. Introduction to Network Coding

In this chapter, the basic knowledge of network coding will be given. We mainly give the explanations such as what network coding is and how it works in networks. It contains main network coding theorem (Section 3.1), linear network coding (Section 3.1) and practical network coding (Section 3.2). Thereafter, we also introduce several interesting previous researches which are related to wireless broadcasting using network coding (Section 3.3); besides, the limitations and disadvantages of these previous related works are also listed and discussed (Section 3.4).

3.1 Main Network Coding Theorem

As introduced in Chapter 1, network coding is firstly proposed in a single-session multicasting case. In [1], the authors have proven that reliable multicast at an maximum possible information rate which is the minimum of the individual min-cut bounds can always be achieved in any network given that network coding is allowed to be used [5].

Under the basis of [1], specific encoding, re-encoding and decoding methods of network coding are designed, called linear network coding [11]. The output flow at a given node is obtained as a linear combination of its input flows which regards a block of data as a vector over a certain field and allows a node to do a linear transformation before transmitting it [11]. In other words, nodes have to perform linear operations, additions and multiplications and the coefficients of these linear combinations are, by definition, selected from a large finite field $GF(2^s)$. The size of finite field has to be decided carefully, because on one hand, the size of the Galois Field should be large enough in order to guarantee that all the transmitted packets are unique but dependent with other related packets; on the other hand, to reduce system the computational complexity, the size of the field should be chosen as small as possible.

A main theorem in network coding regarding to Min-cut Max-flow theorem is given in the following [5] which combine the basic idea of network coding and the specific coding way of linear network coding:

Main Network Coding Theorem [5]:	Consider a directed graph $G=(V,E)$ with unit capacity edges, h unit rate sources located on the same vertex of the graph and $T \subseteq V$ receivers. Assume that the value of the min-cut to each receiver is h. Through using linear network coding solution, the information delivered from the sources can be received simultaneously to each receiver at the maximum possible rate of h can be achieved when multicasting.

3.1.1 Linear Network Coding

After stating the main theorem of network coding, we will explain the operation way of linear network coding in detail which includes two main parts: "encoding" and "decoding".

- **Encoding:**

  We assume that at a given time a certain node in a network has N original source symbols $m_1, m_2, ..., m_N$ generated by one or several sources in a network.

  Using linear network coding, a transmitted packet contains a symbol, called "information vector" which is generated in the form of $g_1 m_1 + g_2 m_2 + ... + g_N m_N$, and the sequence of coefficients $g = (g_1, g_2, ..., g_N)$, called "encoding vector" which are selected from a finite field [11].

  Encoding can be performed recursively (to already encoded packets) [3] [4] and encoding coefficients can be chosen deterministically [12] [13] or randomly [14] [15]. In this thesis we use randomized network coding where encoding coefficients are randomly selected from a large finite field. It has been shown that network throughput can be improved significantly if encoding coefficients are selected randomly [14, 16-19].

  For example, assume at a certain moment, there are two messages in node v_i's buffer, which are $g_1 m_1 + g_2 m_2 + g_3 m_3$ and $g_4 m_1 + g_5 m_2 + g_6 m_3$. When there is an opportunity for node v_i to rebroadcast a packet, it will re-encode its previously received packets, the operation way of which is in the following:

  $$\alpha_1(g_1 m_1 + g_2 m_2 + g_3 m_3) + \alpha_2(g_4 m_1 + g_5 m_2 + g_6 m_3) = g_7 m_1 + g_8 m_2 + g_9 m_3,$$

  where $g_7 = \alpha_1 g_1 + \alpha_2 g_4$; $g_8 = \alpha_1 g_2 + \alpha_2 g_5$; $g_9 = \alpha_1 g_3 + \alpha_2 g_6$; since we use random linear network coding, α_1 and α_2 are chosen randomly from a finite field of large size.

- **Decoding:**

  Upon reception of these mixed packets, nodes also need to recover the original source packets and then forward them to upper layers. Therefore, in the decoding part, nodes have to solve a linear system of equations with N unknowns (the original source packets $m_1, m_2, ..., m_N$). Assume a node in the network has received L messages, it means there is a linear system with L equations and N unknowns. According to the basic knowledge of linear algebra, the number of received packets needs to be at least as large as the number of original symbols ($L \geq N$), such that the information is enough for this node to solve the linear system and then retrieve these source symbols .

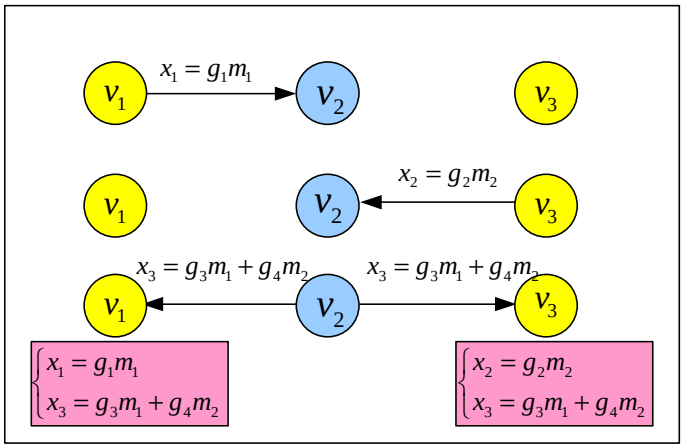


Figure 3.1: A simple example of linear network coding.

For the example in Figure 3.1, assume three nodes v_1, v_2, v_3 are in the same network, both node v_1 and node v_3 want to transmit their source packet m_1, m_2 respectively to each other and node v_2 is the intermediate (relay) node in the network. Assume node v_1 (v_3) sends out their source packet in the form of a linear combination, $x_1 = g_1 m_1$ ($x_2 = g_2 m_2$) to the network. Upon reception of both two packets from v_1 and v_3 respectively, assume node v_2 immediately gets the chance to transmit a packet with the version of a linear combination of all previous received packets ($x_3 = g_3 m_1 + g_4 m_2$, source packets are m_1, m_2 and the encoding vector is $g = (g_3, g_4)$). And then, after receiving the packet from node v_2, node v_1 (v_3) is able to recover source packets m_2 (m_1) and finish the purpose of information exchanging, because v_1 (v_3) has received 2 linear equations with 2 source packets, which are illustrated in the rectangles of Figure 3.1.

From [11], the authors proof that linear network coding is sufficient to achieve the max-flow bound when multicasting; furthermore, the result is somewhat stronger than the one in [1] since the code is a linearity which greatly reduce implementation overhead.

3.2 Practical Network Coding

When network coding is proposed first, it is assumed to be used in a theoretic network; however, if we expect network coding can be used in realistic applications, many potential problems need to be considered and solved. A comparison and analysis for the difference between theory and practical networks is listed in Table 3.1.

Considering the apparent differences between theory and practice, Chou et al. [14] propose a new network coding operation way in order to realize the practical usage of network coding. It allows information travel asynchronously in packets; allows packets subject to random delays and losses; allows edges have variable capacities

due to congestion or other traffic. Their work on practical network coding also addresses real networks where node and link failures are common. Moreover, no knowledge of the network topology is required. The highlights of this practical network coding are (assume the broadcast capacity is known):

Table 3.1: Comparison of theory and practice for wireless networks [14].

Theory	Practice
Symbols flow synchronously throughout network;	Information travels asynchronously throughout network;
Edges have unit (or known integer) capacities;	Edge capacities often unknown and time-varying;
Centralized knowledge of topology assumed to compute encoding and decoding functions;	Difficult to obtain centralized knowledge, or to arrange reliable broadcast of functions;
Assume packets are received without any delays and there are no losses or errors occurring during the communication process;	Packets subject to random delays and losses;

- **Decoding Matrix:**

  Assume that a certain node in a network has received several messages during the communication process; it is allowed to keep encoding vectors which are contained in the received messages, row by row in the form of so-called "decoding matrix" if they are "innovative". Whenever an "innovative packet" (the definition will be given later) is received, it is inserted as last row into the decoding matrix. For the same example in Figure 3.1, if the technique of decoding matrix is used, then after receiving a mixed packet from node v_2, node v_1 keeps these packets in the form of decoding matrix:

$$\begin{bmatrix} g_1 & 0 \\ g_3 & g_4 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_3 \end{bmatrix}.$$

- **Innovative or non-innovative packet:**

  A received packet is decided to be innovative if its encoding vector increases the rank of the decoding matrix; otherwise, non-innovative packets that do not increase the rank of the matrix is simply discarded from the matrix. The reason why the non-innovative packets are not needed and have to be dropped is that such packets contain redundant information and are useless for packet transmissions and exchanges.

- **Buffering Model:**

  Buffering is needed by nodes in the network to synchronize the packet arrivals and departures, such that asynchronous packets are allowed to throughout the network. With buffering, packet that arrives at a certain node is kept in a specified buffer corresponding by the generation number from the packet header (will be explain in next part of "Generation"). If the packet is innovative, the

node has an opportunity to retransmit a random linear combination of all the received packets in the same buffer.

- **Generation:**

For practical reasons, huge amounts of packets might be transmitted in real networks, such that if all the packets are kept together in one decoding matrix, the size of decoding matrix must be increased. With the increasing size of decoding matrix, the requirement for system buffer must be increased and it will also lead to high computation complexity.

Therefore, in order to limit the size of the decoding matrix, the idea of generation is proposed in [14]. Packets can be grouped into different generations [14] and only packets belonging to the same generation are allowed to be combined with other packets. The size of each generation is equal to the broadcast capacity.

A specific method used in [14] to make the idea of generation come true is explained in the following where a specific packet format is designed for supporting the operation way of practical network coding (see Figure 3.2): we assume the size of each generation is h which means a generation is allowed to consist of at most h source packets. The packet format is described in the following: a h-dimensional encoding vector (the size of encoding vector is also h) is appended to each packet that describes how the source packet(s) is (are) encoded; a generation number which is also called generation Identity (ID) and a symbol of information vector are also appended to the packet in order to make this possible.

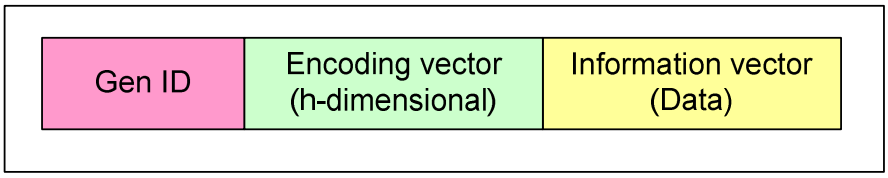


Figure 3.2: Packet format used in practical network coding.

After receiving the encoded packet, this packet will be put into a right generation buffer according to its generation ID if it is innovative, and then do earliest decoding if possible (earliest decoding will be described below). If the node gets a chance of rebroadcast, it will send out a linear combination of all the packets that are buffered in the same generation associated with an encoding vector, the encoding coefficients of which are randomly selected from a defined finite field in order to make the transmitted packet unique.

Besides, not all the generations have to be kept together, otherwise, it will occupy huge amount of nodes' memory space. The generation management method used in [14] is that the current generation is advanced and the old generation including all the packets of the certain generation is flushed from the buffer whenever the first packet of the current generation is received. It is called flushing policy. The example in Figure 3.3 illustrates the generation management method used in [14]. It shows the current buffer situation of node v_i. In the left part, assume so far, all the received packets kept in the buffer are belonging to the generation No.1.

However, when a new packet which is from another different generation, called generation No.2 in this example, arrives at node v_i, according to the generation management method used in [14], the generation No.1 with all its packets has to be flushed in order to create a new generation and store this new packet.

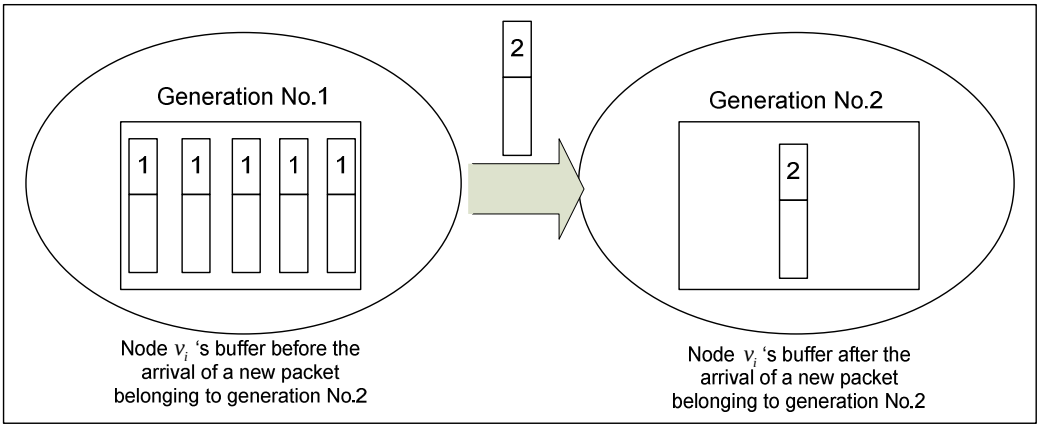


Figure 3.3: Illustration of the generation management method (flushing policy) used in [14].

Authors of [14] have proven that dividing packets into different generations allows the node to decode source packets faster since the size of the matrix is limited. Furthermore, the requirements of system buffering and network computation complexity are decreased as well.

- **Earliest Decoding [4] [14]:**

Normally, decoding can be achieved by a node if all the packets belonging to the same generation are received. Gaussian elimination could be performed on the corresponding decoding matrix in order to extract the source packets from the decoding matrix. The decoding delay in this block decoding method equals the length of time for the receiver to collect all the packets belonging to the generation, which is proportional to the generation size. Thus, the original decoding method might lead to a longer packet delay since decoding action can only be done if all the packets in the same generation are received.

Therefore, in order to decease the decoding delay, in [14], earliest decoding is proposed, in which Gaussian elimination is performed immediately when an innovative packet is received and kept in the right decoding matrix according to the generation ID of this received packet. For the example in Figure 3.4, assume the size of generation is 4. In normal way (left part of Figure 3.4), a node has to wait until receiving four related packets and then is able to decode the source packets m_1, m_2, m_3, m_4; however, in the way of earliest decoding (right part of Figure 3.4), since Gaussian elimination is performed each time that an innovative packet is received, although the decoding matrix is not full, information is still sufficient to retrieve source packets m_1, m_2. According to the simulation results shown in [14], earliest decoding yields a much lower decoding delay than that of block decoding.

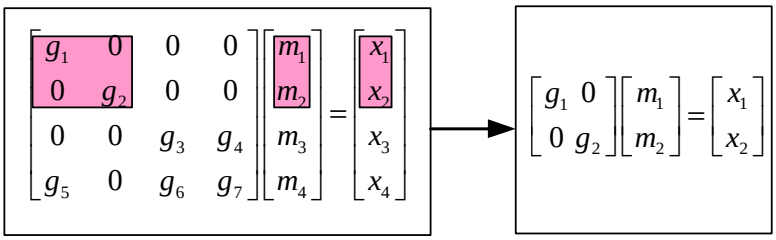


Figure 3.4: An example of earliest decoding.

3.3 Previous Work of Flooding the Network using Network Coding

After introducing the basic knowledge of network coding, in this section, some useful and interesting previous works which are related to the problem of wireless broadcasting using network coding are presented.

3.3.1 Network Coding Algorithm in DTNs [9]

In [9], one broadcast algorithm that combines PFA and network coding is proposed. But it is designed particularly to be used in DTNs which is called network coding algorithm in Delay-Tolerant Network (DTNs) in this thesis.

The distributed algorithm used in this algorithm in DTN is given in Figure 3.5, in which packet distributions are decided by the parameter of forwarding factor.

Distributed algorithm: assume forwarding factor $d > 0$:

For each source symbol that originates at a node, the node will send such packet $\max(1, \lfloor d \rfloor)$ times, and an additional packet is generated and send out with probability $p = d - \lfloor d \rfloor$ if $d > 1$.

When a node receives an innovative packet, $\lfloor d \rfloor$ information vector will be generated from the corresponding decoding matrix and broadcasted to the neighbors, and a further information vector is generated and sent with probability $p = d - \lfloor d \rfloor$.

Figure 3.5: Distributed algorithm used in network coding algorithm in DTNs [9].

The features of this algorithm are briefly given in the following [9]:

- A new method to manage generations called “generation hashing” is also proposed, and the authors of [9] believe that this generation management method performs better than any other related methods such as generation membership based on local scope (only packets that were originated few hops away from the node that created the generation are allowed to be in the same generation).

- They also discuss the problem of how efficient network coding can achieve even with little available memory space. A scheme called “information aging” is

proposed which allocates very little memory space to old information while maintaining a relatively high PDR. The simulation results show that this method achieves great network performance of reliability and robustness.

3.3.2 Efficient Network Coding Broadcast Algorithm [10]

Recently, based on the above algorithm, an improved algorithm which is designed to be used in random wireless networks instead of DTNs is also given here. Due to the functionality of this algorithm, in this thesis, we call it Network Coding Broadcast Algorithm (NCBA). The advantages of NCBA are also given in the following [10]:

- A simple distributed algorithm of NCBA that allows approaching the optimal performance in practice is designed (see Figure 3.6). It is an improved version compared with the distributed algorithm used in Network Coding Algorithm in DTNs which is based on CBS and is also easy to be implemented in real networks. Clearly, for the distributed algorithm use in NCBA, both the probabilities and number of packet transmissions and rebroadcasts are decided both by the values of sending counters and by the parameter of forwarding factor.

Distributed algorithm: constant forwarding factor d

- Each node maintains a sending counter s that is initially set to zero.

For each source symbol that originates at a node, the node increases s by $\max(1,\lfloor d \rfloor)$, and further increases s by one with probability $p = d - \max(1,\lfloor d \rfloor)$ if $p > 0$.

When a node receives an innovative packet it increases s by $\lfloor d \rfloor$, and further increases s by one with probability $p = d - \max(1,\lfloor d \rfloor)$ if $p > 0$.

If $s \geq 1$, a node attempts to broadcast a linear combination over the span of the received coding vectors. Each transmission reduces the send counter s by one.

Figure 3.6: distributed algorithm based on a constant forwarding factor [10].

- This algorithm is mainly focusing on the situation of energy saving. Authors of [10] have investigated benefits in terms of energy efficiency in an ad-hoc wireless network and they have proven that network coding can offer a constant factor of benefits over a fixed network and a $\log n$ factor over a network where the topology dynamically changes;

3.4 Limitations & Disadvantages

In previous sections, we have already explained the basic concepts and knowledge of network coding, and also analyzed the features and advantages of the previous proposed algorithms that are related to the problem of network coding based broadcasting. Here, we would also like to analyze and conclude the limitations and disadvantages of these related works. Practical network coding solution will be

analyzed firstly, and then deficiencies of two algorithms that are related to the problem of network coding based broadcasting are also given. At the end, several criterions of being a good network coding based broadcasting algorithm are listed.

- **Practical Network Coding:**

  According to [14], the contributions for making network coding more practical are significant; however, several problems have not been solved yet which impede network coding becoming completely practical:

  1) When a new generation occurs, an old generation with all its packets should be dropped although there are some source packets that have not been decoded. In other words, network might lose those packets that are not retrieved yet when they are dropped;

  2) The generation management way is far from practical. In order to make the idea of generation work, at the beginning of communication, the broadcast capacity should be known in order to define the size of generation and generation membership. However, in real networks, we never know what kind of source packets will be transmitted and thus it is impossible to divide source packets into different generations based on their generation management method.

  3) According the packet format, the whole encoding vector, the size of which is equal to the size of generation, should be sent associated with an encoded packet. Thus it incurs additional overhead in the header. For example, we assume source packets $m_1, m_2, ..., m_h$ belong to the same generation. At a given time, a packet is transmitted and actually it only includes one source packet e.g. m_2. But using practical network coding solution, the whole encoding vector which contain h encoding coefficients needs to be transmitted in the form of $(0, g_1, 0, ..., 0)$.

- **Network Coding Algorithm in DTNs:**

  The advantages of network coding algorithm in DTN [9] are: 1) nodes are allowed to have more than one source packets to send; 2) the technique of "information aging" does reduce the burden of system buffering for old information. The limitations of this algorithm are present in the following:

  1) The algorithm is designed specially in DTNs architecture; hence, it will probably incur some problems when it is directly used in a random wireless network.

  2) Because of the property of "generation hashing", generation IDs are selected from a set of small size, such that the number of packets kept in a same generation will be increased gradually in order to reach a higher reliability; in other words, a larger memory space is still needed, and meanwhile, the system operation complexity still cannot be reduced.

3) Although "information aging" is used aiming to save nodes' memory space, it might cause another problem: packet delay will also be increased.

- **NCBA:**

  Finally, a discussion of NCBA is also given here. According to [10], the simulation results show that in the static topology network coding using this algorithm can achieve better performance (PDR and packet delay) with much less network resources.

  However, there are still several unpractical problems and several unclear points existing in this algorithm:

  1) In this literature, the authors think NCBA can be easily extended to operate over generations. However, they do not give their generation management method in detail and how to really use generations in network coding is still obscuring, such as what is the generation membership standard to create different generations; what the size of generations is; how many generations one node can contain at the same time; how to manage generations in order to avoid inter-fluencies (e.g. an arrival of new generation might incur packet losses if the flushing policy is used as generation management method like [14]).

  2) According to the simulation part, they only use one single generation that holds the all the received packets together, such that several problem are caused:

     a) The amounts of packets should be seriously controlled (the way they use to control the amount of packets is that each node in the network is only allowed to transmit at most single source symbol (packet);

     b) Due to the large size of generation, it increase system processing burden and also increase the requirements of nodes' buffering since the memory space of each node in the network should be large enough to store all the received packets, the size of which is depending on the number of packets (the number of nodes in the network).

Considering the limitations of the algorithms "Network Coding Algorithm in DTNS" NCBA, and PFA (see Chapter 2), obviously there is still huge research space to further discover and display the benefits of network coding. We will give a brief conclusion of PFA and two related algorithm from the aspects of network performance and system processing requirements (see Table 3.2) in a whole view. The comparison results showing in the table are evaluated according to the related literatures.

We can conclude from Table 3.2 that although both related algorithms and PFA can achieve great network performance (such as high PDR, low packet delay), they also have their own unpractical limitations; besides, in the terms of network coding based algorithms, the network processing complexity is still too high to be used in practical application and the requirements of system buffering are still too much higher to stand

for a real network. Therefore, we would like to design a new algorithm and we hope that it can keep the advantages, and meanwhile, solve all the problems that are shown in Table 3.2.

Table 3.2: A conclusion and comparison of the related algorithms;

Items		Probabilistic Flooding Algorithm	Network Coding Algorithm in DTNs	NCBA
Network Performance	High PDR	**Yes**	**Yes**	**Yes**
	Low packet delay	**Yes**	**Yes**	**Yes**
	Low network overhead (saving energy)	**No**	**Yes**	**Yes**
Practical Scenarios	Algorithm is allowed to be used in random wireless networks;	**Yes**	**No**	**Yes**
	Using certain algorithm, each node can have multiple source packets to send; huge amounts of data is allowed to transmitted during the communication process;	**Yes**	**Yes**	**No**
Low System Processing Complexity		**Yes**	**No**	**No**
Requirement of System Buffering		**Low**	**High**	**High**

In the next chapter, we will introduce our algorithm NCF which is designed based on previous related algorithm. We hope it is able to both reach the great network performance and overcome such limitations and disadvantages described above. As presented in the table, our goal is to design such an algorithm which is able to make each comparison item to be positive (e.g. Yes) instead of negatives (e.g. No).

4. Network Coded Flooding Algorithm

In this chapter, we will introduce our approach NCF in detail which is aiming at seamlessly integrating network coding with one of the current network broadcast protocols, PFA, and is designed in order to make up with the disadvantages of previously proposed related algorithms.

The introduction of NCF include several specific parts: system model and basic network coding functions (Section 4.1), packet distributed method (Section 4.2), a different generation management method (Section 4.3); thereafter the whole operation way of NCF will also be presented, which is divided into two parts: "initial sending part" (Section 4.4.1) and "receiving and retransmission part" (Section 4.4.2).

4.1 System Model and Basic Functions

We are interested in broadcast traffic of wireless networks, where a broadcast message originating at its source is received by all its neighbors in the same network. A random-topology static wireless network model with a large number of nodes is used in our case. Each node in the network can be a sender that transmits its new packet(s) and each packet originating at one of these nodes should be received by all the other nodes in the same network in order to achieve the goal of flooding (it means each source packet should be received and successfully retrieved by all the nodes in the network). Moreover, multiple nodes are allowed to communicate with each other simultaneously. We use the same terminologies as what we have defined in Chapter 1.

In this algorithm, "practical network coding [14]" solutions are used. Whenever there is an opportunity of retransmission, a node will send out a rebroadcast message which is a linear combination of all its previously received packets (in the right generation). Each packet is always sent associated with one encoding vector, the encoding coefficients of which are randomly selected from a finite field with a large size (refer to Section 3.1). In NCF, we select such finite field with size $GF(2^8)$, so that each encoding coefficient selected from this field can be store in one byte; moreover, according to [5] [9], field size of $GF(2^8)$ is sufficient to highly limit the probability that two encoded packets are totally identical.

When a node receives one packet which is a linear combination of source symbol(s), according to the generation number of the packet, it will insert this packet into a right decoding matrix as the last row if it is an innovative packet.

In the decoding part, a different decoding method is proposed which is also designed for the purpose of time efficiency. After receiving an innovative packet, nodes will check whether the decoding matrix is a full matrix which means the number of columns in the decoding matrix is equal to the number of rows; if so, nodes can solve the matrix and retrieve related source symbols**.** We will further explain our decoding method later (Section 4.4.2).

4.2 Packet Distributed Method

As mentioned before, we choose PFA to combine with network coding. According to previously related algorithms introduced in Section 3.2, both of them (the algorithm in [9] is PFA based and the algorithm in [10] is CBS based) use a parameter of forwarding factor. Through related computations that depend on such parameter, the number and probability of packet retransmissions can be controlled and decided.

However, what we need is a simpler method without any calculations before rebroadcasting. Therefore, based on the property of PFA, the parameter of rebroadcast probability is used instead of using forwarding factor. Besides, according to the simulation results of these previous related algorithms [9] [10], they only focus on forwarding factor's range from 0 to 1. In that case, it is unnecessary to use the parameter of forwarding factor; instead, rebroadcast probability can make the same effect as forwarding factor does, given that the range is from 0 to 1.

Briefly speaking, PFA is chosen in our case to integrate with network coding, the packet distributed method in NCF is mainly based on such broadcasting technique and we use the parameter of rebroadcast probability to control the probability of packet retransmission instead of using the parameter of forwarding factor. The details of the packet distributed method used in NCF are described in Figure 4.1.

If a new packet originates at its source, it will be broadcast with probability $p = 100\%$.
Packet Distributed Method: assume rebroadcast probability is p
Assume a node v_i receives a packet, and the rebroadcast probability is p
If the packet is an innovative packet, it will be stored in node v_i's buffer and one rebroadcast is activated with the predetermined probability p. A rebroadcast packet is a linear combination of all the received packets belonging to the right buffer sorted by the generation ID.
If the packet is a non-innovative packet, it contains duplicated information and will not be helpful for packet transmission and will be dropped.

Figure 4.1: Packet Distributed Method in NCF.

According to Figure 4.1, we divide packet delivery process into three situations:

1) For the situation that new packets originate at their sources, sources must send these new packets with a probability of 100%. The reason why nodes must send their source packets once independent with a probability is that if sources decide not to send out their source packets, none of the nodes in the same network will receive those packets at all so that the performance of packet delivery will be seriously degraded.

2) When nodes in the network receive innovative packets from their neighbors, nodes get the chance to mix all the packets (do random linear combination of all the packets) in the corresponding generation and rebroadcast it with the predetermined probability, called rebroadcast probability.

3) If a node finds out that a received packet is a non-innovative packet, it will discard this packet since it contains duplicated information and will not be helpful for packet transmission and communication.

4.3 Generation Management Method in NCF

Our purpose is to make NCF work in real networks, where no synchronization is needed to control packets' arrivals and departures; where packet delay and packet losses are allowed to happen; where huge amount of data can also be transmitted during the communication process. Therefore, the idea of "buffering and generation" which is an important technique of practical network coding (refer to Section 3.2) is necessary to be used in our case, since:

1) Buffering model [14] allows asynchronous packets arrivals and departures with arbitrarily delay and loss;

2) Dividing packets into generations means smaller coding vectors are allowed and then controls the size of decoding matrix and system process complexity; besides, it will probably decrease the packet (decoding) delay and save the system memory space.

Before explaining our generation management method, we want to briefly review several proposed methods. According to [14], on one hand, given that only one generation with limited size is allowed to be kept in a node, when a new generation occurs at this node, the current generation with all its packets will be flushed and discarded, it will probably cause packet losses since there might be several un-decoded packets in the discarded generation**;** on the other hand, if all the generations are kept together like "network coding algorithm used in DTNs" [9], although a special generation management method is used in this case ("generation hashing", refer to Section 3.3.1), it cannot efficiently control the size of generation and keeping huge amount of data together will absolutely occupy much larger system memory space. In addition, considering the complex functionality of the generation management method, "generation hashing", it itself has already added to the burden of system.

Certainly, generation is a very helpful technique which can help network coding become more practical. However, if no suitable and reasonable generation management method is used, blindly using generations will make network perform worse than before. Therefore, in our case, in order to design a practical network coding based broadcast algorithm, we have to carefully consider the problems related to generations, such as how to control generations; how to divide packets into different generations (the way to control generation membership) and etc.; and also think of a flexible and reasonable generation management method.

In NCF, we propose a different generation management method which will be explained in the following.

There are two parameters that are used to control generations and thus to control the situation of system buffering and process complexity:

- **Number of generations one node is allowed to keep:**

  Nodes do not need to contain all the generations together in their buffers; instead, only parts of generations with parts of the packets are needed to be kept, such that the requirement of system buffering is reduced;

- **Size of each generation:**

  Controlling the size of generation means controlling the size of decoding matrix, such that nodes do not need to process a decoding matrix of a huge size, and then the system process complexity will be controlled.

The main idea of our generation management method is given in the following example (Figure 4.2). This figure shows a current buffer situation of a certain network node v_i. At this moment, four generations have already been kept in its buffer and we assume there is not enough memory space for node v_i to contain any new packet which is belonging to a new generation. However, if a new packet with a new generation arrives at node v_i at this moment, according to our generation management method, node v_i will discard one of the existing generations in its buffer, such that enough memory space can be released for collecting this new packet with a new generation.

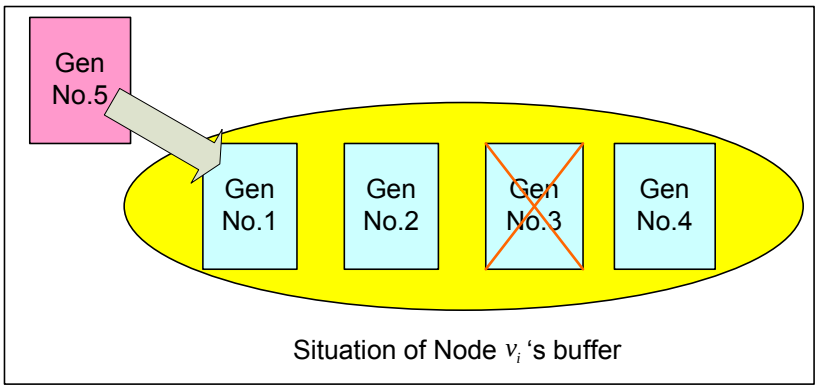


Figure 4.2: Illustration of main idea of generation management method used in NCF.

In order to make this idea possible, the related issues what kind of method can be used to separate packets into different generations, and what kind of standard can be used to decide generation's creation and dropping are becoming more and more essential in our case. Moreover, we also hope the discarded generations will cause relative less effect or even cause no negative influence on network performance.

In NCF, we propose the idea of "generation list" to solve the above mentioned problems. The detail operation way of generation list is explained in Section 4.3.1 and a particular packet format which supports the function of generation list is also given thereafter (Section 4.3.2).

4.3.1 Generation List

Each node is required to have its own generation list which contains almost all generations' current information. It helps nodes efficiently control packet transmission or receiving, and also generation's creation or discarding. Using generation list, it is easy for a node to know the information including the quantities and the identities of generations, the size of occupied memory space and available space, and etc, such that nodes can immediately decide whether and how to create or drop generations. The detail functions of generation list are given in the following:

- **Function of Generation List:**

  In our method, the evaluation standard to compare generations depends on the creation time of each generation and also available memory space. Therefore, generation list contains the information:"(Unique Generation ID, Created time of the generation, Number of source packets)" (e.g. (No.1, T(1), 3) means the creation time of the generation named No.1 is designated as T(1), and number of source packets that is also the number of columns of the decoding matrix in this generation is 3).

  The item of "unique generation ID" gives the information of generation's identity (the reason why generation ID has to be unique is explained in the part of "Creating or Discarding Generation"). In order to make the identity unique, each time when a new generation is created, the generation ID is randomly chosen in a large set which is composed with huge amount of integers; the item of "creation time of the generation" gives the information of the exact time when the generation is created which will be helpful for sorting the generation list (explain in the part of "Sorting the generation list"); the item of "number of source packets" gives the information that how large the remaining available memory space is.

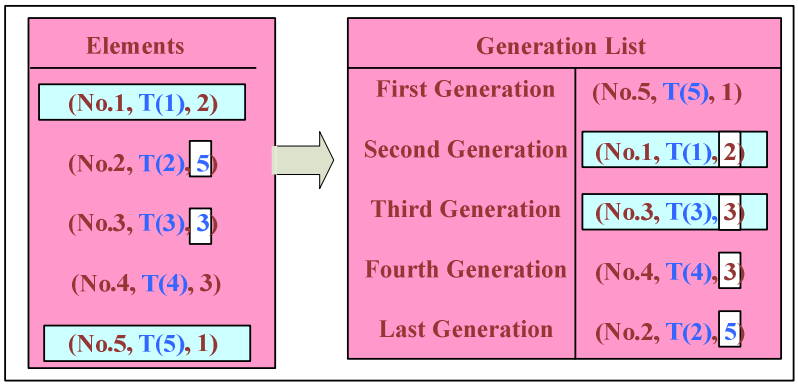


Figure 4.3: Example of generation list.

One example of a generation list is shown in Figure 4.3. We assume the order of creation time of each generation is $T(5) > T(1) = T(3) = T(4) > T(2)$ which means the earliest created generation is generation No.2 and the latest created generation is generation No.5. The elements in the generation list should be put in order according to the created time from the newest (named "first generation") to

the oldest (named “last generation”).

If the creation time of two or more generations is the same (e.g. generation No.1, No.3 and No.5 in the example), the node will further compare the number of source packets of these corresponding generations, and put them in the order from the smallest to the largest. However, if the numbers of source packets of these related generation are also identical (e.g. No.3 and No.4 in the example), the order of these generations will be arranged randomly. Certainly, the precision of the creation time depends on the implementation. In our case, we use NS-2 to implement NCF and the time defined in our case is extremely precisely such that the probability that two or more generations created at the same time is exceedingly low. In the future, we can also investigate how precise the creation time of generation could be defined for comparison in generation list in order to guarantee the good network performance of NCF.

As shown in the example, the elements’ order in generation list is mainly focuses on the creation time of generation. The main reason is that we think a relative new generation might contain more innovative packets and all the packets in a relative old generation might have already been successfully received and recovered. The number of source packets is also taken into account is because we have to control the size of each generation and thus, a generation with relative large available space is considered and used firstly.

Generation list plays an important role in NCF. In order to display the function of generation list more clearly, we give a practical example: we assume there are many students in a class. After examination, the teacher wants to know the entire students’ current situations such as who the best student in the class is or which student needs more help. If the teacher records all the students’ exam results in a list and sort them in the order from highest to lowest, it will be convenient for the teacher to know the students’ situations: the first student in the list means he (or she) is the best in the class, and the last student in the list means he (or she) needs more help.

Similarly, the information contained in the generation list is sorted in a particular way which is explained above. When a node wants to send a packet with one or several new source packets, only through checking its generation list, the node can decide which generation could be considered first (always choosing the first generation in the list or creating a new generation if necessary); when a node wants to drop one generation for releasing some memory space, the generation list can also let the node know which is the most suitable generation that could be discarded (always discarding the last generation in the list).

- **Sorting the Generation List:**

  As long as a node creates or drops one generation, its own generation list should be updated and sorted accordingly, which is named “sorting the generation list”. The way how to sort the generation list (compare the elements in the generation list and put them in order) after something changing is described in the above. For instance (see Table 4.1), if a node creates one new generation, before sorting the generation list, the node will insert the information of this created generation into

its generation list as the last row (in this table the new generation is No.4) and the initial number of columns is always equal to zero. A new generation can also be inserted at any position of the generation list which can be easily done through implementation. After sorting the list, the elements in the generation list are put in order again.

Table 4.1: Example of sorting generation list after changing. Assume the order of the creation time of each generation is T(1)=T(4)>T(2)>T(3).

	Before sorting the generation list	After sorting the generation list
First generation	(No.1, T(1), 2)	(No.4, T(4)=T(1), 0)
Second generation	(No.2,T(2), 5)	(No.1, T(1), 2)
Third generation	(No.3, T(3), 5)	(No.2, T(2), 5)
…	…	…
Last generation	(No.4, T(4)=T(1), 0)	(No.3, T(3), 5)

- **Creating or Discarding Generation:**

  Nodes are allowed to create new generations or drop old generations if necessary. The situations that generation has to be created or dropped will be explained in the Section 4.4.

  As long as a new generation is created, both a unique “generation ID” and the “creation time of this generation” should also be given and recorded.

  When a generation is decided to be discarded, all of the packets in this generation are also dropped.

  Whenever a node creates or drops one or several generations, information of the generation list needs to be updated and re-sorted (the details are explained in “function of generation list” and “sorting the generation list” in this section).

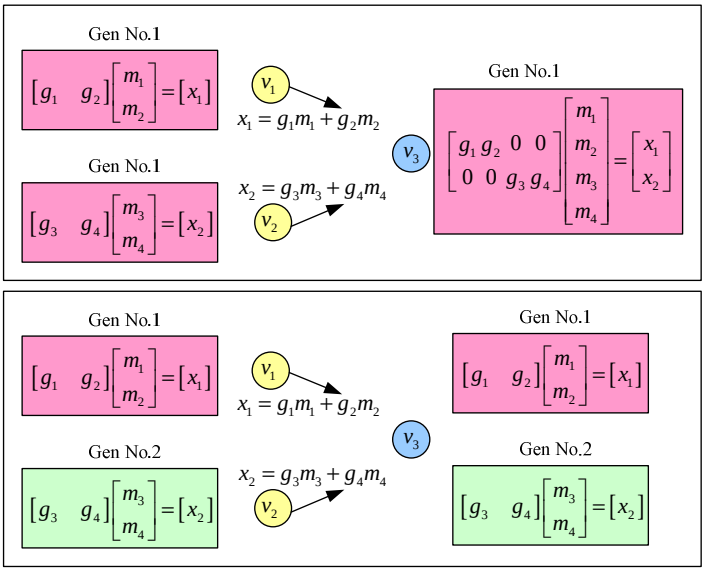


Figure 4.4: Explanation of choosing generation ID in a huge set. The first figure is the example that generation IDs are the same and the second figure shows an example that the collision of generation IDs is avoided.

There is one problem that is worth to being discussed: we requires each generation's ID to be unique such that the probability of creating generations with the same generation ID is relatively small and thus, such collisions might be prevented.

An example is shown in Figure 4.4 that explains the reason why the collision of creating generations with same generation ID should be avoided: assume there are three nodes v_1, v_2, v_3 in the network, and both node v_1 and v_2 want to transmit packets to node v_3 at the same time. According to this example, node v_1 wants to transmit a packet with source symbols m_1, m_2 and a packet from node v_2 contains source symbols m_3, m_4.

For the first figure, assume the size of the set that is used to choose the generation IDs is very small, the situation that both node v_1 and v_2 create their generations at the same time with the same generation ID might happen probably (the generation ID is assumed as generation No.1 in this example). Clearly shown, after these two packets that are transmitted from node v_1, v_2 respectively are received by node v_3, the size of decoding matrix for generation No.1 is increased to 4×2 which means node v_3 cannot retrieve any one of these four source symbols m_1, m_2, m_3, m_4 unless it receives at least two more related packets.

In contrast (see the second figure), assume the size of the set that is used to select generation IDs is larger enough to avoid such collision (selected generation ID is same and not unique). Given that both node v_1 and v_2 create their generations at the same time with different generation IDs (the generation ID is assumed as generation No.1 for node v_1 and generation No.2 for node v_2), after receiving these packets, node v_3 should keep two generations separately, the size of which are 2×1 respectively. Obviously, the computation complexity is decreased because solving two 2×2 matrixes is relatively easier than solving one 4×4 matrixes; packet delay could also be reduced with a relative higher probability since only receiving one more related packet, the source packets contained in generation No.1 (or No.2) could be recovered. However, there are other problems that can be caused by the way of generation identity composition, we will further discuss this problem (refer to Section 6.2.1).

Certainly, in order to make generation list work, a suitable packet format is also necessary. In next section, we propose a packet format that is different from the previous related algorithms.

4.3.2 Packet Format

In order to support the particular operation way of NCF, a corresponding packet format is necessary to be designed. A packet format using in NCF is illustrated in

Figure 4.5 which includes the generation ID and creation time of this generation; marks of source symbols; associated encoding vector and information vector which is a symbol generated from a linear combination.

According to the packet format used in NCF, one new item "creation time of this generation" is added as a part of packet. The main reason is because of the generation list. Only through comparing the creation time, the elements in the generation list can be arranged in order such that the function of generation list can be realized (see 4.3.1 "functions of generation list').

Besides, we do not need to send a whole encoding vector; instead, only the encoding coefficients that are corresponding with the transmitted source symbols are needed to be sent. For example, we assume the size of generation is 8. If we use the former packet format (Figure 3.2) and assume only the 3rd source packet is combined in this packet, the corresponding encoding vector should be $(0,0,g_1,0,0,0,0,0)$. However, in our method, the corresponding encoding vector is (g_1) together with a unique mark of this related source packet (e.g. m_3) such that the size of encoding vector is reduced.

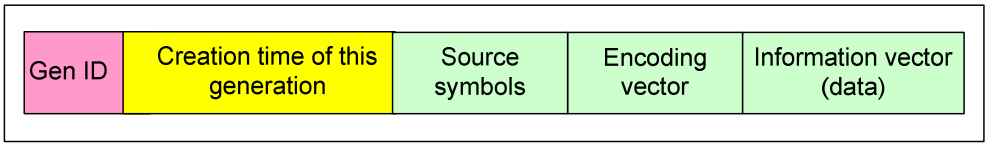


Figure 4.5: Packet format of NCF.

Depending on the different packet format, a different decoding way is also proposed correspondingly which will be explained in Section 4.4.2.

4.4 Initial Sending, Receiving and Retransmission

After introducing the packet distributed algorithm and generation management method used in NCF, in this section, we will explain the operation way of NCF in detail. We divide the operation way of NCF into two parts: initial sending part; receiving and retransmission part.

Table 4.2: Symbols' definitions about node v_i.

Symbols	Meaning	Initial value
M	maximum number of elements in a generation list (maximum number of generations a node can have simultaneously)	
T	maximum generation size of each generation	
n(i)	number of generations (generation buffers) node v_i has	n(i)=0
Gen (j)	unique ID of the generation j selected from a set of large size which is composed with huge amounts of integers	
#columns(j)	number of columns in the decoding matrix of generation j (equal to the amounts of source packets in generation j)	#columns(j)=0
#rows(j)	number of rows in the decoding matrix of generation j	#rows(j)=0

Firstly, the definitions of some symbols which are used to describe the state of one of the nodes v_i in the network are listed in Table 4.2.

4.4.1 Initial Sending Part

The initial sending part describes the situation when one or several source packets originate at a certain node in the network at random time during the communication process. The flow chat is described in Figure 4.6.

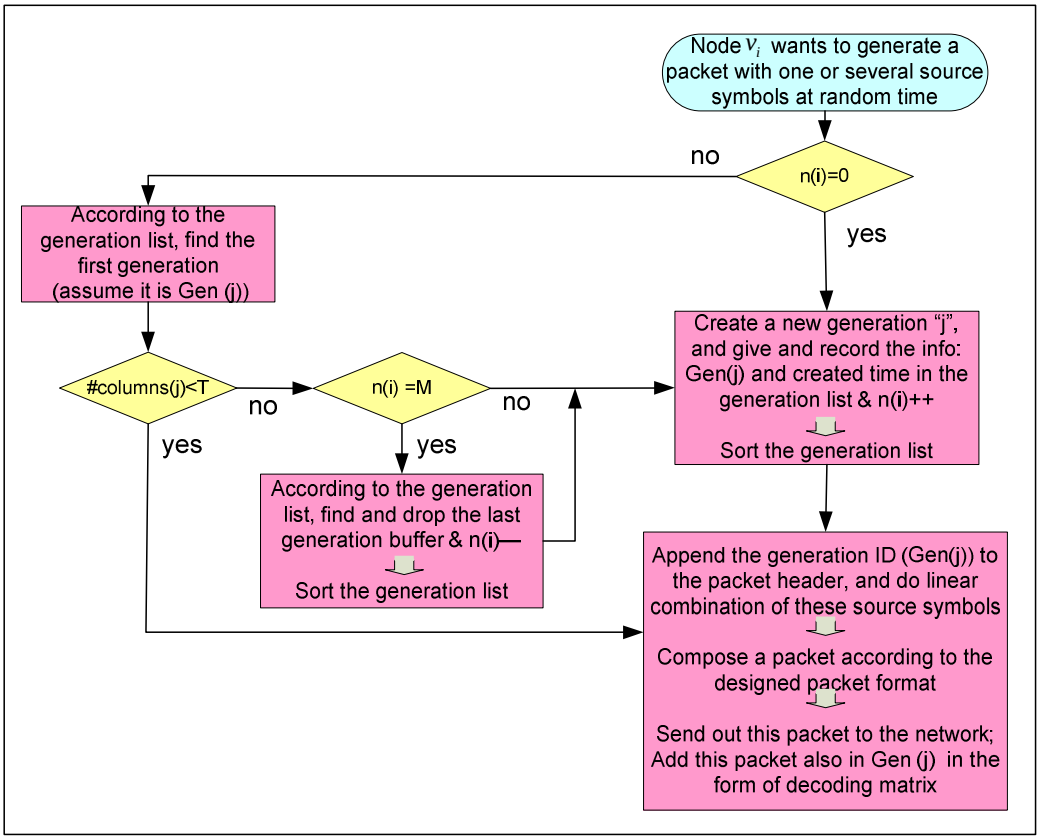


Figure 4.6: Initial sending part of NCF

According to Figure 4.6, we mainly define two situations that could be met in the initial sending part in the following. Whenever a source packet needs to originate at its source, the generation list has to be carefully checked firstly.

1) If there is no generation existing in the node ($n(i)=0$, the generation list is empty), the node will create one new generation (see Section 4.3, as long as creating one new generation, unique generation ID and creation time must be given and recorded, and the initial value of number of columns is set as zero); meanwhile, the related information needs to be recorded in its generation list. Then, the node sends out a linear combination of the source symbol(s) and stores it in its generation buffer for back-up.

2) If there are one or more generation buffers that have already been existing in node v_i ($n(i) \neq 0$, there are elements in the generation list), the node will choose the first generation according to its generation list (since the elements in the list are always put in order of creation time from the newest to the oldest, and node always choose the newest generation to send packet if it is available), and further check its number of source packets:

   a) If the first (newest) generation is not full which means this newest generation still has enough space to contain new source packets, the node will certainly choose this generation as the packet's generation, then store the new source symbol(s) in this generation buffer, and send out a linear combination of all received packets including the new source symbol(s).

   b) However, if the newest generation (the first generation in the list) is full, the node has to create one new generation instead of using this generation; otherwise, generation size would be unnecessarily enlarged. As the number of generations is also limited considering the limit memory space of a node, before creating a new generation, the node should check how many generations have already existed in. If the number of generations is smaller than the maximum, it means there is enough memory space for the node to create one new generation without influencing any other generations kept in this node. Otherwise, if the number of generations is already equal to the maximum, the only way is to discard one generation such that some memory space can be released for creating and keeping one new generation. In our case, removing the last generation in the generation list (dropping all the packets in this oldest generation) is a better choice. The reason is that the probability that nodes in the network still need the oldest data at the given time is relatively low.

      One important thing cannot be forgotten: as long as creating or discarding some generation, the generation list should be updated and re-sorted (refer to section 4.3.1).

4.4.2 Receiving and Retransmission Part

The receiving and retransmission part (Figure 4.7) is described the situation that a packet is received by a random network node during the communication process.

When a node receives one packet, the information of generation including the generation ID and its creation time can be obtained from the header of such packet. According to its generation list, the node must check whether the generation has already existed in this node. If the node has already got this generation and the packet is defined as innovative (non-innovative packets need to be dropped), there are three actions that are needed to be activated:

- **Sorting the generation list**:

  The number of source packets should be updated in the generation list if there are new source symbols added in this generation.

- **Receiving and decoding**:

  According to our packet format (see Section 4.3.2), the decoding method used in NCF is different. Let us make an example. We assume at this moment, according to the generation ID, the decoding matrix of node v_i is

$$[g_1][m_1]=[x_1].$$

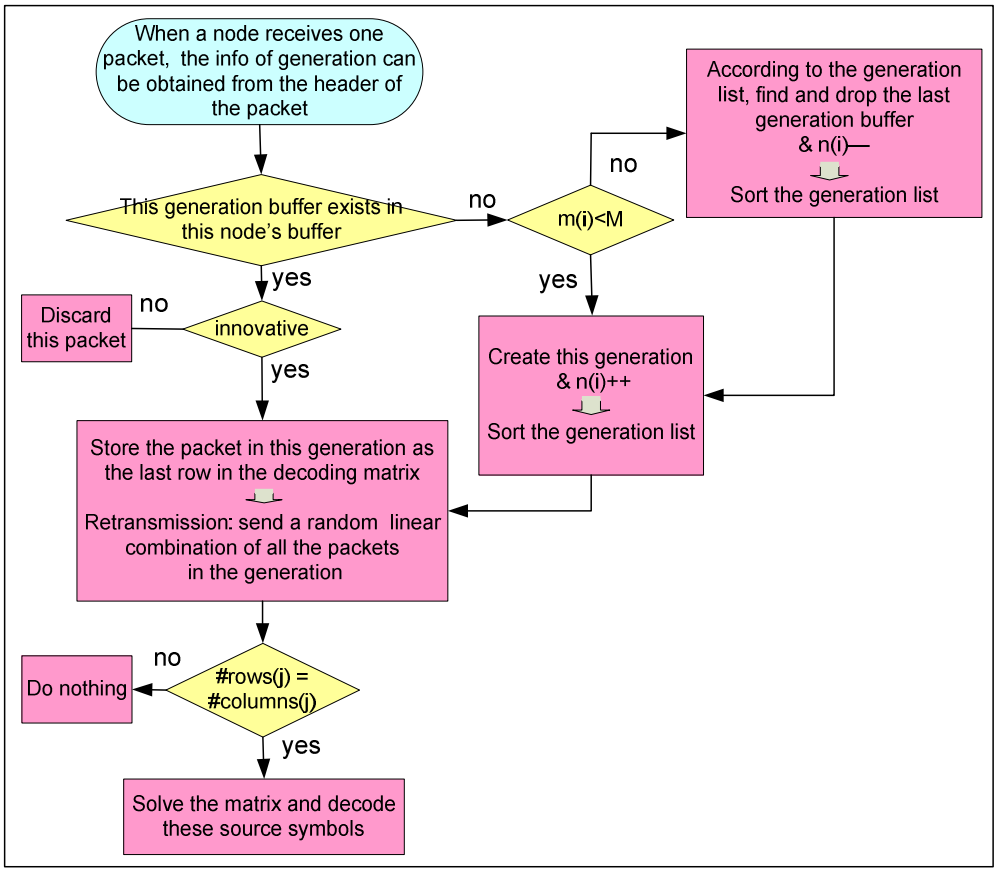


Figure 4.7: Receiving and retransmission part of NCF.

And then an innovative packet belonging to this generation is received and according to its packet format, the symbol of information vector is x_2; encoding vector is (g_2, g_3); related source packets are m_2, m_3. Therefore, node v_i will keep this packet in the form of decoding matrix which is in the form:

$$\begin{bmatrix} g_1 & 0 & 0 \\ 0 & g_2 & g_3 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}$$

Clearly shown in the above matrix, since the received packet does not include the source packet of m_1, the corresponding position in the decoding matrix is appended as 0; similarly, the second and third columns of the first line are also appended as 0.

In this algorithm the node can execute decoding process to recover the original source symbol(s) when the number of columns in the decoding matrix is equal to the number of rows. Then, the decoded data will be sent to the upper layers. This way of decoding also makes this protocol work more practically and efficiently as it further reduces the size of encoding vector; and it allows solving the decoding matrix early instead of solving the decoding matrix only if the whole generation is completed. In other words, the upper layers will receive the decoded data earlier instead of waiting until the end of communication process.

- **Retransmission**:

  As long as receiving an innovative packet, the node obtains an opportunity to retransmit one packet with a predetermined rebroadcast probability. If so, it will retransmit a random linear combined packet. In NCF, we define that retransmissions will be activated immediately without any delay. Later, we can also let nodes retransmit packets after some random delay in order to reduce the influence of collisions.

If the generation of the received packet does not exist in this node, the node should create this generation in order to store the received packet in a right generation buffer. Similarly, before creating, the node should check the number of generations it has so far. If it is still smaller than the maximum, the node will create this generation corresponding to the generation ID in the packet header; otherwise, the node has to discard the last (oldest) generation according to the generation list and create this new one. After creating a right generation corresponding to the received packet's generation, the actions including storing the innovative packet, updating and sorting the generation list, retransmission and decoding processes are also needed to be executed as mentioned above.

In the receiving part, another problem should also be explained here. If the node has already had this generation and the packet is defined as innovative (non-innovative packets need to be dropped), the node will store the packet in this generation and put it as the last row of its decoding matrix even if the size of the packet's encoding vector exceeds the maximum generation size, such that the number of packet losses might be decreased. An example in Figure 4.8 can clearly show this kind of situation.

We assume that before sending, both node v_1, v_2 have already kept Generation No.1 and one packet with source symbol m_1 has already been stored in their "Generation No.1" buffers. The maximum generation size is assumed as 2. According to the figure, given that at the same time both node v_1, v_2 want to transmit one packet with source symbols m_1, m_2 and m_1, m_3 respectively and they both choose generation No.1 since the size of it is smaller than the maximum. In the left part of this figure, extra packets are not allowed to be stored in the right generation if the number of

source symbols reaches the maximum, such that node v_3 can only keep one of both packets in its generation buffer (for the example in the figure, because one generation is only allowed to contain at most 2 source symbols, node v_3 only keeps the packet from node v_2 and discard the packet from node v_1 although it is also innovative). Clearly, for node v_3, a source symbol (for the example, it is m_2) might never been received and recovered which will degrade the network performance.

In contrast, in the right part of this figure which exactly describes the receiving operation of NCF, extra packets are allowed to be kept provided they belong to this generation and are innovative although this particular generation is already full, such that all these three source symbols m_1, m_2, m_3 can obtain the opportunity to be recovered.

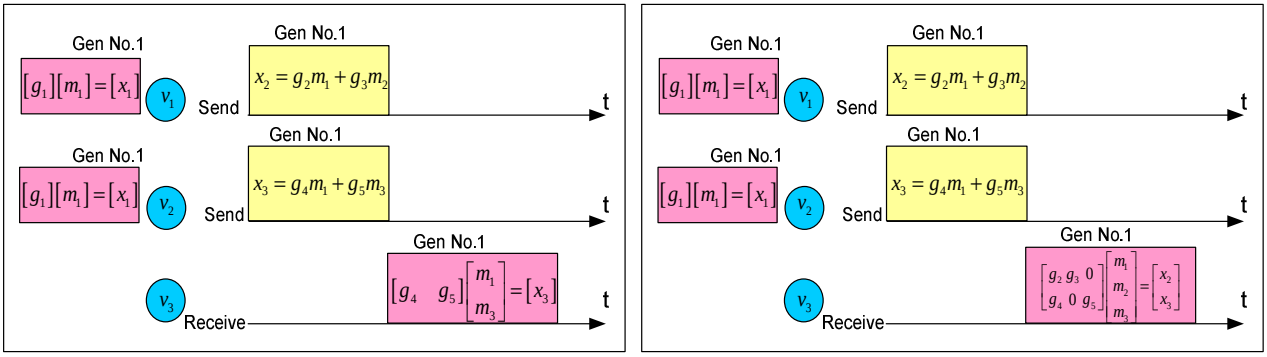


Figure 4.8: An example of packet receiving. Left part: packet is not allowed to be kept in the right generation buffer if the generation is already full; right part: packet can be kept in the right generation buffer no matter how many packets have already been in this generation. Assume the maximum generation size is set as 2.

4.5 Summary of NCF

In this chapter, we describe our algorithm NCF in detail including the packet distributed method, the generation management method and corresponding packet format. We also explain the operation way of NCF in two aspects of initial sending part and receiving and retransmission part. In this section, we would like to make a specific example in order to give a clear overview of NCF.

According to Figure 4.9 (this example illustrates one of the situations that NCF can meet during communication process), packets, tagged by generation number (generation identity are 1, 2 and 3 respectively and showing with different number) receive sequentially, subject to collision and loss [14]. We assume the creation time of each generation is in the order of $T(3) > T(2) > T(1)$, the maximum generation size is 2 and each node can contain at most 2 generations at the same time. There are two nodes in the same network and we assume at this time, node v_1 sends a packet to the network. When this certain packet which is belonging to generation No.3 and the information vector is $x_1 = g_1 m_1 + g_2 m_2$ (two source packets are included in this packet) arrives at node v_2, it will first check its generation list and also compare with

the generation ID retrieving from the certain packet's header. According to the contents in generation list, there is no such generation named generation No.3 in node v_2's buffer and the amounts of generations are already equal to the maximum. Therefore, in order to store this new arriving packet, node v_2 will discard the last generation which is generation No.1 in this example, and create a new generation called generation No.3 and then put this packet in the right buffer with the form of decoding matrix. Of course, the generation list will be updated and sorted. If the node gets one opportunity to rebroadcast a message, it will do the network coding solution of randomly linear combination of all the source packets in the right generation which is associated with an encoding vector.

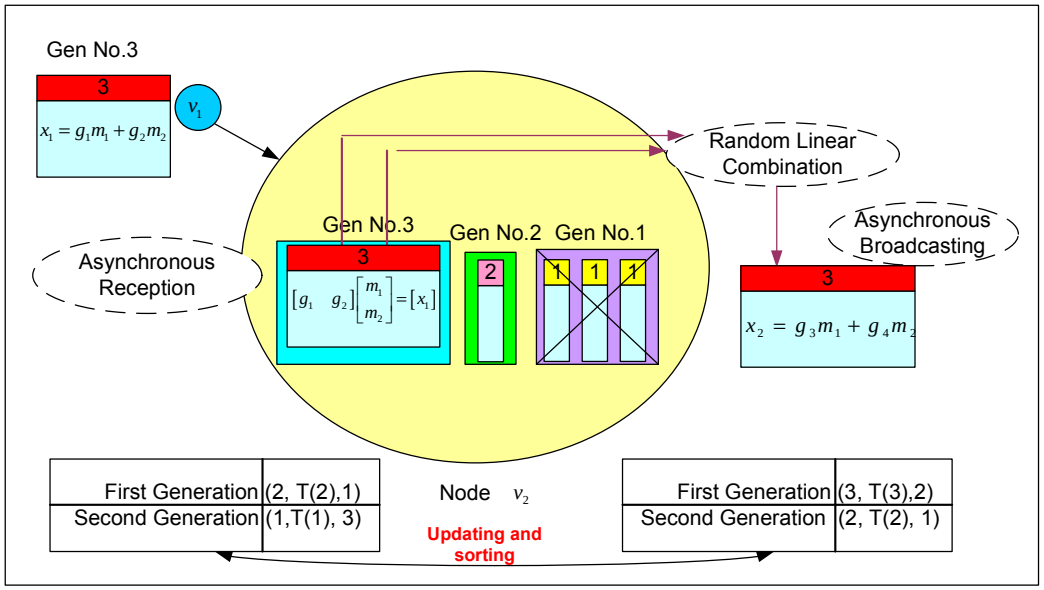


Figure 4.9: Brief conclusion for NCF. Assume a node can contain at most two generations at the same time [5].

In this chapter, we specifically introduce the designed algorithm "Network Coded Flooding Algorithm". Briefly speaking, it is a network coding based flooding algorithm which can both limit the size of each generation and the number of generations one node can have simultaneously aiming at achieving a better network performance while system processing complexity and the occupation of memory space are reduced to a reasonable level.

After theoretic analysis, in the next chapter, we want to show the simulation results of this designed algorithm. Through simulation, we can further analyze the pros and cons of NCF compared with other related algorithms.

5.Simulation Results and Analysis

In previous chapter, we have explained the details of NCF. In this chapter, we would like to analyze and compare the performance of NCF with several related algorithms, especially with NCBA and PFA. After briefly introducing the simulation environment in Section 5.1, specific simulation results are presented in Section 5.2-5.3, and several related parameters such as rebroadcast probability, generation size, number of generations, and network density are investigated. At the end, we propose several trade-off schemes in Section 5.4 which are based on obtained simulation results.

5.1 Description of the Simulation

5.1.1 Simulation Scenario

We use NS-2 to implement our algorithm, and the simulation scenario is implemented based on tcl script. Node configuration and related variables' setting-ups are described in the Figure 5.1.

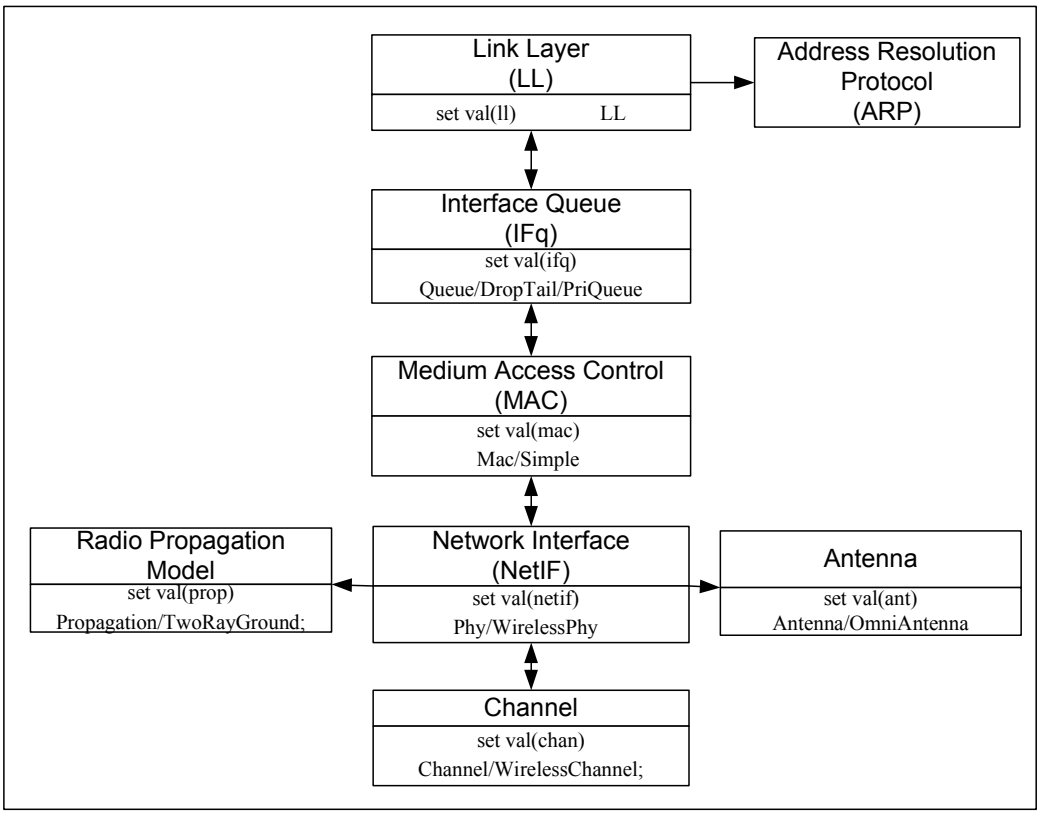


Figure 5.1: Implementation based on tcl script: node configuration and variables setting-ups' description.

For the topology of the simulation network, we use $500m \times 500m$ simulated wireless network and network nodes in the network are randomly distributed on such area. We use NS-2's default settings of wireless network physical (PHY) layer (denoted as "Phy/WirelessPhy" in NS-2) such that the transmission range and the receiving range of each node is $250m$. The specific default settings of "Phy/WirelessPhy" are given in the following [20]:

Table 5.1: Default settings of "Phy/WirelessPhy" in NS-2 [20].

Phy/WirelessPhy set CPThresh_ 10.0
Phy/WirelessPhy set CSThresh_ 1.559e-11
Phy/WirelessPhy set RXThresh_ 3.652e-10
Phy/WirelessPhy set Pt_ 0.28183815
Phy/WirelessPhy set freq_ 2.4e+9

Other related settings in tcl are also given here:

Table 5.2: Other related settings of our simulation.

set val(rate)	20	#number of broadcast packet generated per second
set val(traffic_interval)	10	# the interval while there is traffic in the network
set val(traffic_ start)	1	
Mac/Simple set	bandwidth_	1Mb

For the Medium Access Control (MAC) layer, we use a simple MAC type without RTS/CTS which can be directly set in NS-2 by the command of "Mac/Simple" without any implementations.

Figure 5.2 shows an example simulation network area which contains 100 randomly distributed network nodes.

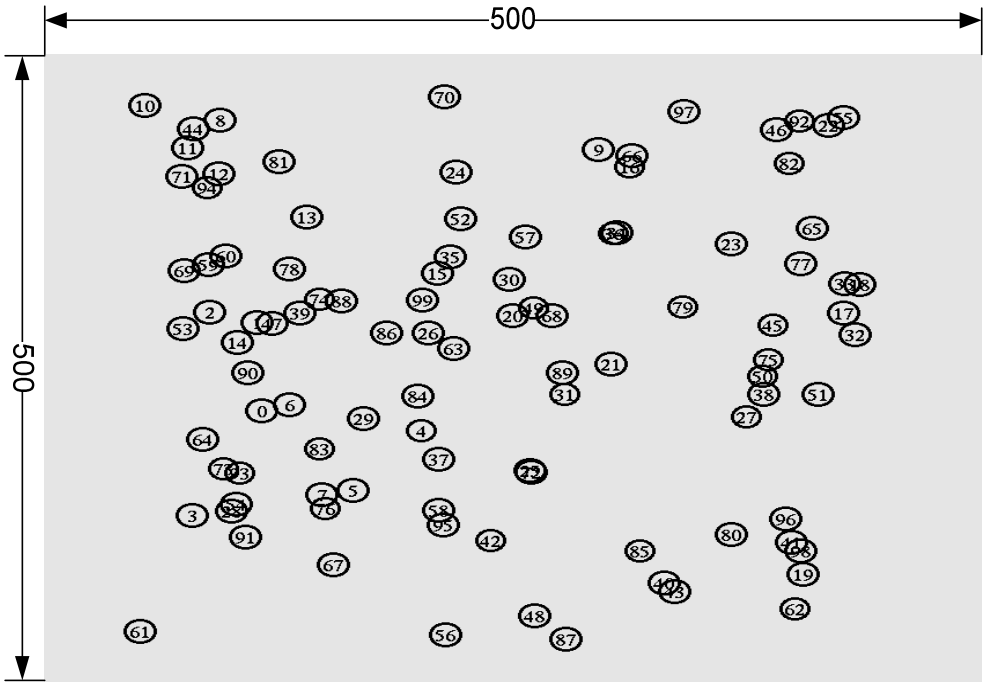


Figure 5.2: Example simulation network area with 100 randomly distributed nodes.

5.1.2 Performance Metrics

The performance metrics are "PDR"; "end-to-end packet received and decoded delay (packet delay)", and "network overhead (number of rebroadcast times)", the definitions of which are given specifically in the following:

- **PDR:**

It is the most important parameter which can clearly show the reliability and efficiency of the network.

For each source packet that originates at its source, with PFA, PDR is percentage of network nodes who accurately receive this new innovative packet; if network coding is allowed, PDR is the percentage of network nodes that receive and successfully decode this original source symbol.

Following the terminologies which are given in Chapter 1, we assume there are number of $|M|$ source packets transmitted during the simulation process, and for a random source packet m_k, the *PDR* of such packet is defined as $PDR(m_k)$. We use the average value of *PDR* to evaluate the network performance of packet delivery:

$$\overline{PDR} = (\sum_{k=1}^{k=|M|} PDR(m_k)) / |M|, \quad k = 1, 2, ..., |M|.$$

The reason why we prefer to use the average value is that because nodes are randomly distributed in the simulation area, it is possible that parts of the simulated network are dense and some areas of the network are sparse. Obviously, such situations might lead to results that on one hand, some packets which originate from the dense parts of the network can reach higher delivery ratio; on the other hand, packets that originate from an extreme sparse region of the same network might not be delivered at all, since no nodes are within the transmission range of the sources. As we want to investigate the ordinary performance, PDR with the average value is more suitable in our case.

- **End-to-end Packet Received and Decoded Delay (packet delay):**

The performance metric of packet delay shows how long a new innovative packet can be received or retrieved by all available nodes, and thus it shows the working efficiency of the network.

Similarly, for each source packet, if PFA is used, the packet delay is between the time that a certain packet originates at its source and the time that the packet is accurately received by a node in the network. We average the delay time by all the nodes who do successfully receive this packet; if network coding is allowed, it is measured as the decoding time that starts from the original transmission of a certain source packet and ends at successfully retrievals by a node in the network. Similarly, we also average the decoding delay time by the total number of network nodes who do receive and successful decode this source packet.

Certainly, there are many source packets that need to be transmitted in the network; we further average the packet delay by the total number of source packets existing in the given network for the same reason mentioned above.

- **Network Overhead (total number of retransmission times):**

Network overhead is exactly the same as total number of retransmission times which shows the system state of power consumption. The larger the value of retransmission times is, the more the network overhead is needed during communication. This performance metric can clearly show one of the NCF's benefits: energy saving, which will be illustrated through analyzing related simulation results.

In addition, we also add 95% confidence intervals in parts of simulation results that will be shown in Section 5.3 in order to analyze and compare the performance of NCF and related algorithms much clearly.

5.2 Comparison of Three Algorithms

Firstly, we want to compare the performance of NCF with both PFA (refer to Section 2.2.2) and NCBA (refer to Section 3.3.2). In order to compare these three algorithms clearly, we should simulate them in a same network environment. As the limitations of NCBA (refer to Section 3.4), the network environment used in this case is particular:

1) Network contains 100 nodes, which are randomly distributed in an area of $500m \times 500m$; the transmission range of each node in the network is around $250m$.

2) Each node can only have single source packet to send and thus, the total number of source packets transmitted during the communication process is equal to the total number of nodes in the network;

3) According to the simulation environment in [11], packets (not rebroadcast packets) are transmitted during the first 100 time units, at each time unit one packet originates at a randomly selected node. In our case, similarly, we set the time that a new packet originates at its source node is randomly chosen during the first 20 seconds of the simulation process; besides, the average sending rate is 5 packets/second (in one second, about 5 new source packets will originate at their sources in the form of linear combinations); certainly the parameters of system sending rate can be easily changed;

4) We set the whole simulation duration is 20 seconds. Thus, simulation will be stopped at the end automatically although there might be nodes who still want to rebroadcast packets.

Table 5.3: Parameters' setting in simulation.

Generation size	4	8	16	32
Number of generations	8	4	2	1

Moreover, for NCF algorithm, both "the size of each generation" and "the number of generations a node is allowed to contain at the same time" are important parameters in NCF which could significantly influence network behaviors; therefore, in this case, we assume that each node in the network can keep at most 40 packets and different settings of NCF used in simulations are listed in Table 5.3. It seems only 32 packets can be kept in a node instead of 40 because we should consider the situation that in the receiving part of NCF, node should put any innovation packets in the right generation no matter how many packets has already in this generation buffer (refer to the Section 4.4.2 and Figure 4.8).

After introducing the basic setting-ups of simulation, we would like to show several related simulation results. In Figure 5.3, 5.7-5.9, it respectively shows the performances of PDR, packet delay and network overhead for NCF, NCBA and PFA (we use "probabilistic flooding" to denote PFA in our simulation results) which are changing with rebroadcast probability. Since when the rebroadcast probability is large enough, the performance of these three algorithms are almost identical (all of them can achieve PDR of about 100%); we only give the simulation results changing with the rebroadcast probability of range from 0 to 25%. There is several useful information displaying in these simulation results which will be given in the following:

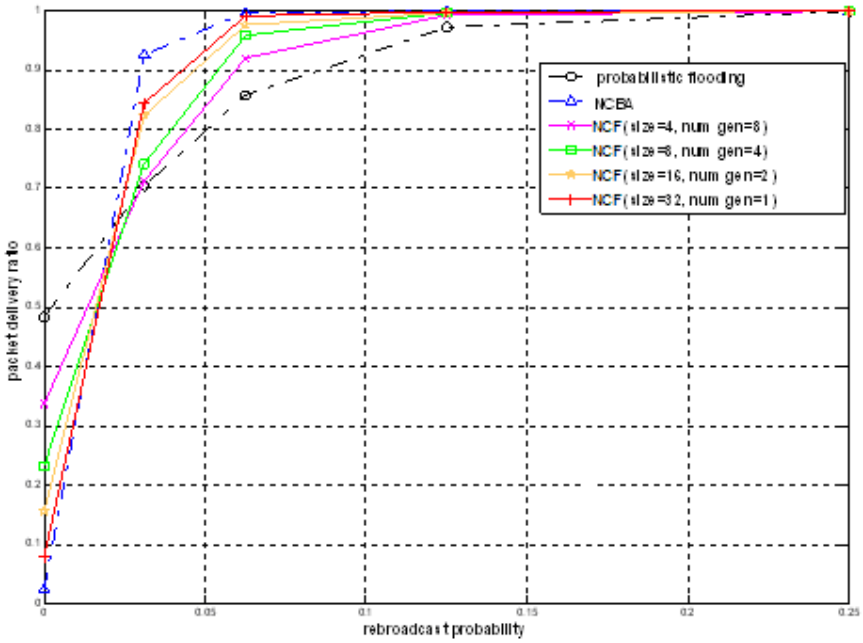


Figure 5.3: Rebroadcast probability-PDR for NCF, NCBA and PFA. (In NCF, the impacts of different size of generation).

- According to Figure 5.3, if nodes are almost not allowed to rebroadcast messages upon the reception of innovative packets (roughly in the range of $0\% < p \leq 3\%$ shown in Figure), we observe that network coding is not helpful; and in contrast, PFA performs better (it is able to get a relatively higher PDR).

However, as the rebroadcast probability increasing, the benefits of network coding are shown gradually. PFA needs a higher rebroadcast probability of larger than 25% to achieve PDR of around 100%; in contrast, network coding can almost achieve 100% PDR merely with a lower rebroadcast probability. Furthermore, the larger the generation size is, the better performance of PDR (higher value of PDR with lower rebroadcast probability) the network can achieve. For the example of NCF algorithm with the size of generation 32, PDR of around 100% is reached only with about 12.5% rebroadcast probability.

Besides, when the rebroadcast probability is larger enough (according to Figure 5.3, it is larger than 25%), PFA can also reach the PDR of almost 100% as network coding algorithms do.

Examples in Figure 5.4-5.6 clearly explain the reason why such behaviors could be obtained through simulation; why network coding does not perform well where rebroadcast is not allowed, and why network coding can achieve higher PDR than PFA if the rebroadcast probability is at a reasonable level. These three examples are given in different situations: rebroadcast probability is extremely low which belongs to the range of around $0\% < p \le 3\%$ (the rebroadcast probability is denoted as p); rebroadcast probability is relatively large (e.g. according to Figure 5.3, rebroadcast probability is in the range of around $3\% < p \le 25\%$); rebroadcast probability is large enough (e.g. larger than 25%):

a) Rebroadcast probability is about 0% ($0\% < p \le 3\%$ **):**

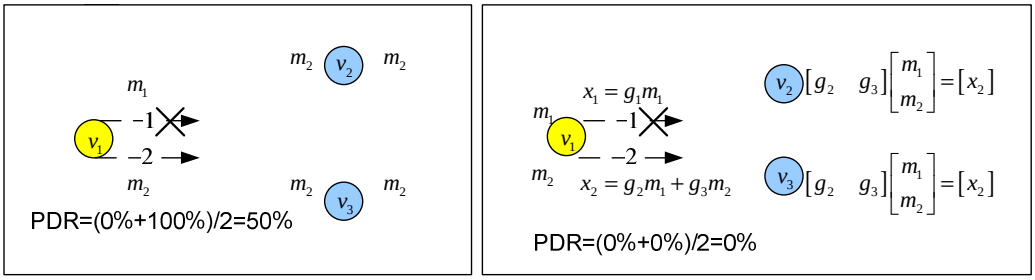


Figure 5.4: Example for comparison of PFA and NCF with an extremely low rebroadcast probability (around $0\% < p \le 3\%$).

For the example in Figure 5.4, it illustrates the situation with extremely low rebroadcast probability, which means nodes are probably not allowed to rebroadcast packets when they receive innovative packets. We assume there are three nodes v_1, v_2, v_3 in the network and node v_1 sends two source packets with source symbols m_1, m_2 respectively, and $(g_1, g_2, ...)$ are the coefficients of encoding vectors used in network coding which are selected in a large finite field. Besides, we also assume only the second transmitted packet is successfully received by the neighbors of the source because of packet loss or collision that occurs during the communication process.

The example of PFA is shown in the left part. We can observe that both two nodes v_2, v_3 receive the second packet with source symbol m_2 and no rebroadcast is activated by node v_2, v_3. Hence, the average PDR is 50%.

For network coding (right part of this figure), both nodes v_2, v_3 receive the second packet of $x_2 = g_2 m_1 + g_3 m_2$; apparently, no source symbols can be recovered as sufficient packets are not received by node v_2, v_3 to recover any source symbols such that the average PDR in this case is 0%.

In conclusion, the above example gives the explanation that PFA does perform better than network coding if rebroadcast is hardly allowed during the communication process.

b) Rebroadcast probability is relatively large:

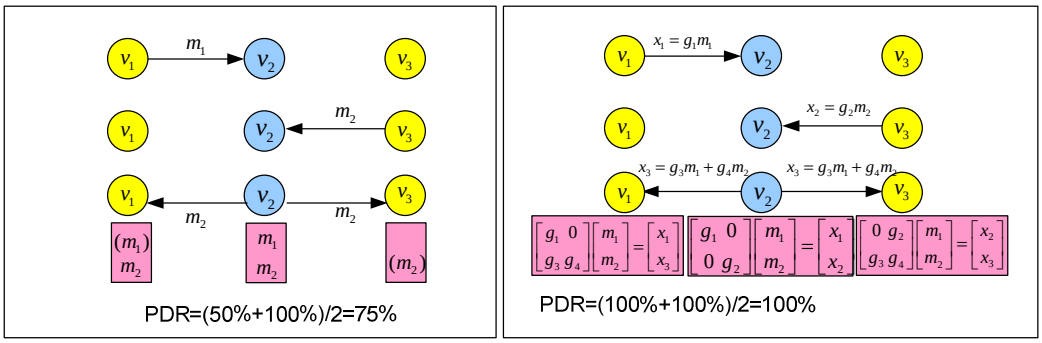


Figure 5.5: Example for comparison of PFA and NCF with relatively high rebroadcast probability (according to Figure 5.3, rebroadcast probability is in the range of around $3\% < p \le 25\%$).

For the example in Figure 5.5, it illustrate the situation that rebroadcast is allowed with a rebroadcast probability in the range of about $3\% < p \le 25\%$. We assume there are three nodes v_1, v_2, v_3 in the same network and node v_1, v_3 want to exchange their information with each other through the relay node v_2.

The example using PFA is shown in the left part of Figure 5.5. We assume after node v_2 receives both messages m_1 and m_2, it gets a chance to forward one of the received message (assume m_2 in this example). We also assume that since the rebroadcast probability is not very large, node v_2 does not get the chance to forward the message of m_1. Then, the average PDR in this case is 75%.

Compared with PFA, an example of network coding solution is also given in the right part of this figure. Similarly, node v_1, v_3 want to exchange their information with each other. The transmitted messages are in the form of linear combinations. We also assume that after receiving both packets from node v_1, v_3, node v_2 only gets one chance to rebroadcast a packet which is

in the form of $x_3 = g_3 m_1 + g_4 m_2$. Then after receiving the packet from the relay node, information is enough for both node v_1, v_3 to retrieve new source packets, and the average PDR in this case is 100%.

Briefly speaking, network coding solution can achieve higher PDR compared with PFA when the rebroadcast probability is relatively large (through observing from our simulation results, we define the rebroadcast probability in this case roughly belongs to the range of $3\% < p \leq 25\%$).

c) Rebroadcast probability is large enough:

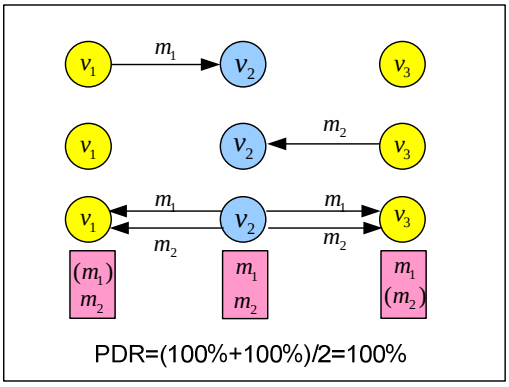


Figure 5.6: Example of PFA for the performance of PDR when rebroadcast probability is larger enough (according to Figure 5.3, rebroadcast probability is roughly larger than 25%).

If the rebroadcast probability is large enough, according to Figure 5.6, we assume node v_2 gets retransmission opportunities to forward both two received messages m_1 and m_2 respectively, such that the purpose of information exchanging can be finished and the average PDR can also be achieved at 100%. It means that if retransmissions are allowed to happen with a large enough probability, both PFA and network coding solutions can reach a very high value of PDR.

- According to Figure 5.7, the packet delay of PFA is not changing apparently with the increasing rebroadcast probability, and it always keeps at a relatively lower value as the operation way of PFA is simple without any encoding or decoding process and the transmission range of each node in our simulation is large enough.

  In contrast, with network coding, the performance of packet delay changes apparently related to the value of rebroadcast probability. If no rebroadcast is allowed, we can observe that packet delay is extremely low. The reason is that network nodes can only decode such source packets which can be immediately retrieved at the first time when it is received; otherwise, they will never be retrieved during the communication process. Hence, only the decoded time of retrieved packets can be recorded which is extremely small and used for the calculations of packet delay.

If rebroadcast is allowed but not with a high probability ($3\% < p \leq 12.5\%$, the range is given according to Figure 5.7), a node may wait for a relatively long time to receive sufficient number of innovative packets in order to recover these source symbols, which means a longer packet delay might be needed for network coding in this case.

However, as the value of rebroadcast probability increasing (larger than 12.5%), both NCBA and NCF can also reach lower packet delay as PFA does. The reason is that with a larger rebroadcast probability, in same duration, the number of innovative rebroadcast packets is increased correspondingly such that the time that is needed to decode source symbols is certainly decreased.

The changing tendency of packet delay is different from the results shown in previous related work [9] [10] and we will further discuss this problem later (Section 6.2.1).

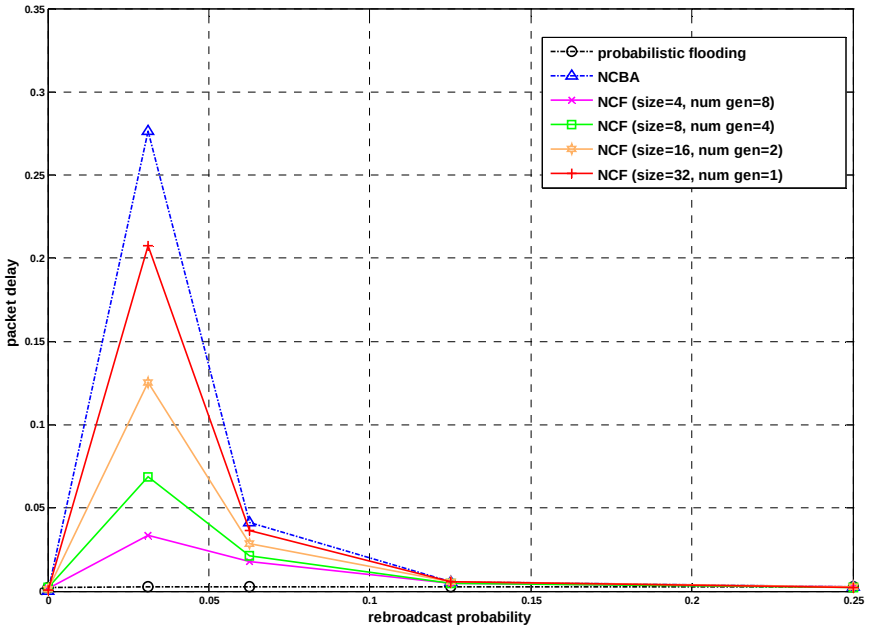


Figure 5.7: Rebroadcast probability-packet delay for NCF, NCBA and PFA. (In NCF, the impacts of different size of generation are also shown).

- In addition to compare PDR and packet delay, we also compare the situation of network overhead changing with the increasing rebroadcast probability. Through simulation, we find that the performances of all three algorithms are almost the same; hence, we only give one result shown in Figure 5.8 which is about NCF with generation size of 32 and the number of generations one node can keep is 1.

  Clearly shown in Figure 5.8, with the increasing rebroadcast probability, number of rebroadcast times is increasing almost linearly. Therefore, a serious disadvantage of PFA is shown obviously: according to Figure 5.3, when rebroadcast probability is larger enough (at least larger than 25%), PFA can also reach almost 100% PDR which is the same as NCF and NCBA; however, according to Figure 5.8, much more times of rebroadcast are needed which means

PFA needs to consume much more network overhead in order to achieve the same performance of PDR. A more apparent comparison is shown in Figure 5.9.

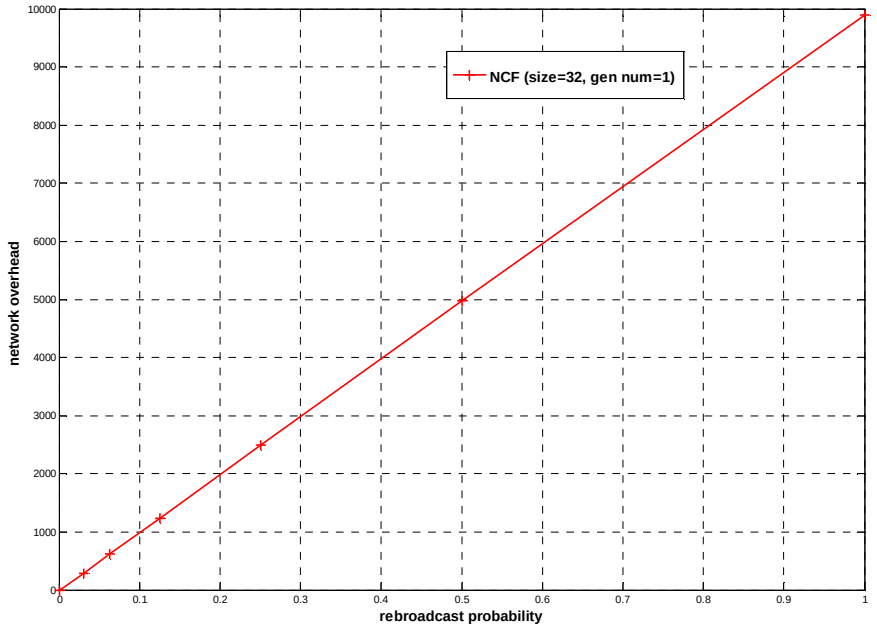


Figure 5.8: Rebroadcast probability-network overhead for NCF.

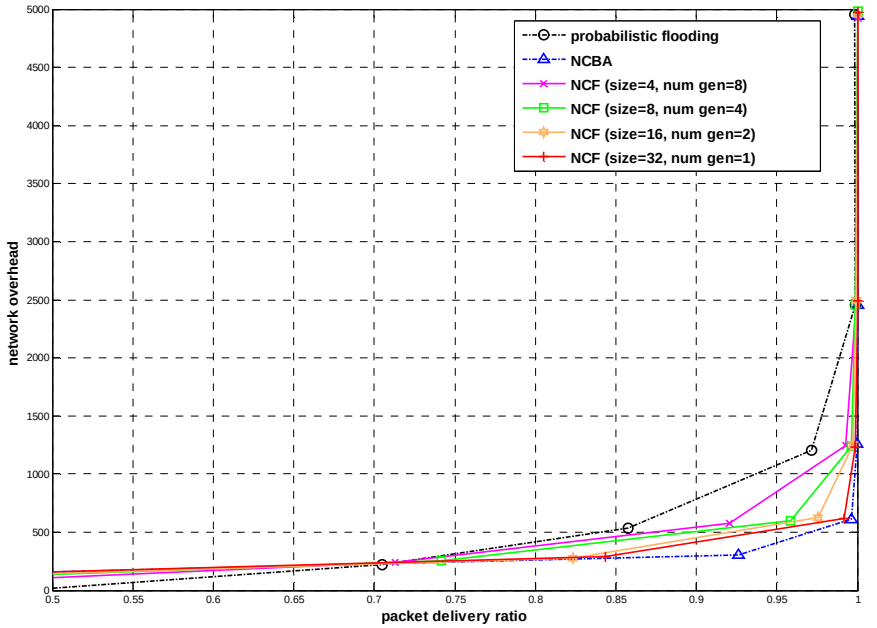


Figure 5.9: PDR-network overhead for NCF, NCBA and PFA. (In NCF, the impacts of different size of generation are also shown).

According to Figure 5.9, assume we want to achieve around 98% PDR, PFA needs the total rebroadcast times of around 1300; in contrast, using network coding, this amount will be reduced significantly which are related to different generation size. For the example of NCF with generation size of 32, only 600

times of rebroadcast are needed to achieve the same PDR, and the amount is decreased greatly by about 700 times.

- Different generation size of NCF leads to different network performance, especially when the rebroadcast probability is smaller.

  The larger the generation size is, better performance of PDR can be achieved and fewer network overhead is consumed while longer packet delay might be caused. The impact of different generation size has already been explained in previous work [16]: buffering sufficient number of useful encodings at an intermediate node before encoding increases the ability of generating useful encodings, such that the correlation between different packets is increased and thus the network reliability (PDR) is also improved**.** But buffering more packets means that more storage space is occupied and larger size of encoding might increase packet delay.

  Therefore, although we are aiming at achieving PDR as high as possible, blindly increasing the generation size is not a good idea since it will probably result in increasing packet delay, network process complexity and buffering requirement. As shown in Figure 5.3, compared with NCBA which needs network nodes to store all the received innovative packets together, NCF's performance proves that only if both the size of generation and the rebroadcast probability are set reasonably, it is unnecessary for nodes to keep all the packets; instead, only keeping a small part of these packets is enough to reach the same PDR. For example, NCF with generation size of 32 can also reach PDR of almost 100% when the rebroadcast probability is larger than around 12.5% in our simulation result, which is the same as NCBA. Besides, using NCF, the occupation of memory space is greatly saved, and both the packet delay and the computation complexity are also reduced compared with NCBA because the size of generation is limited.

  However, compared with NCBA, a disadvantage of NCF can also be found out that the level of NCF's energy saving is relatively lower than NCBA. For instance, in order to achieve around 98% PDR, NCF with generation size of 32 needs roughly 550 times of rebroadcast while NCBA only needs 500 times. This is due to frequently using the idea of generation in NCF and we will further discuss it in Section 6.2.1.

In conclusion, compared with NCBA, except for the energy saving problem, NCF is a better algorithm since only through accurately setting the related parameters of NCF, it can achieve higher PDR with less packet delay. Moreover, it efficiently decreases system process complexity and buffering requirement.

5.3 Analysis of NCF compared with PFA

In the previous section, we compare NCF with NCBA and PFA; however, due to the limitations of NCBA, the simulation environment we use is far from practical and not enough to display NCF's properties. In this section, a more practical scenario is employed:

1) Network size is also equal to $500m \times 500m$, and the transmission range of each node is still around 250m;

2) However, each node in the network can transmit several source packets instead of only one; huge amounts of packets are allowed to be transmitted in the network in parallel;

3) The time a new packet is initiated by its source is chosen randomly during the whole simulation process. Simulation will be stopped at the end automatically although there are nodes who still want to rebroadcast packets.

We will investigate the influences of different generation size, different number of generations and different sending rate as described in Section 5.3.1, and later analyze the situation that the rebroadcast probability is changed (Section 5.3.2) and the influence of network density (Section 5.3.3). We simulate NCF in different aspects in order to fully understand the potentials and even its weaknesses.

5.3.1 Impact of Generations' Amount and Sending Rate

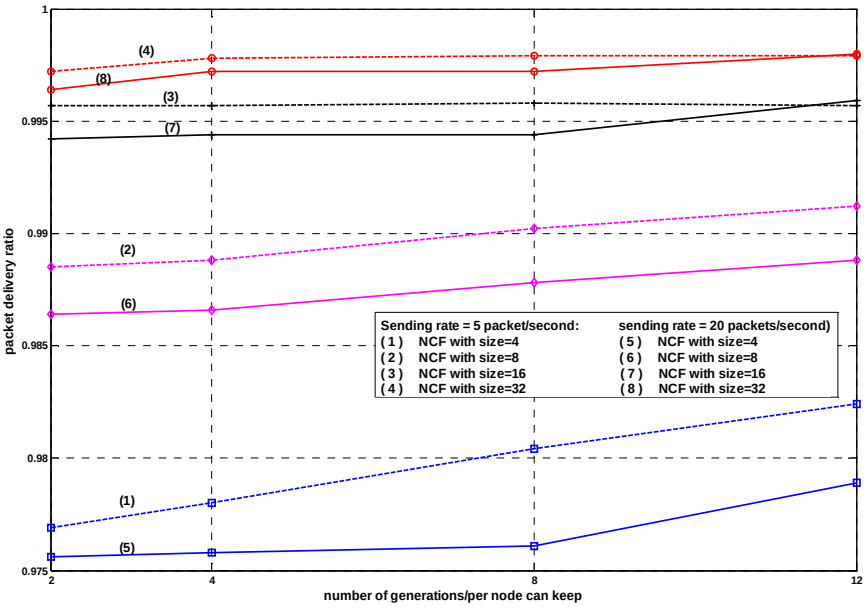


Figure 5.10: PDR for NCF with different generation size and different number of generations per node can have. We test the different performance of NCF with generation size 4, 8, 16 and 32. Different sending rates are also simulated (5 packets/second & 20 packets/second).

According to the introduction of generation management method used in NCF (section 4.3.1), we have already known that one of the properties of NCF is to create new generations or drop relatively helpless generations if necessary for the purpose of reducing the burden of system buffer and further improve network's work efficiency. We have also mentioned that the above operations are decided by node's own generation list which mainly contains two parameters including the number of generations per node is allowed to contain and generation size. In the section, we want

to know through simulation how these two parameters influence the performance of NCF and how to efficiently and accurately set and use the parameters in order to satisfy different network requirements.

The whole simulation duration is 40 seconds. The $500m \times 500m$ network also contains 100 nodes which are randomly distributed in the network area, and the rebroadcast probability is fixed at 10% (since the network environment we simulate is a dense network, rebroadcast probability of low value is enough to finish the purpose of network flooding; besides, according to previous simulation results, we think the rebroadcast probability of 10% can clearly show the differences between NCF and PFA). Since we are aiming at analyzing the impacts of different generation size and different amounts of generations per node has, we compare NCF with generation size of 4, 8, 16 and 32 respectively. Besides, we also change both the number of generations per node is allowed to keep and sending rate of 5 packets/second or 20 packets/second, which means the total number of packets originating during the communication process is 200 and 800 respectively, and investigate its influence.

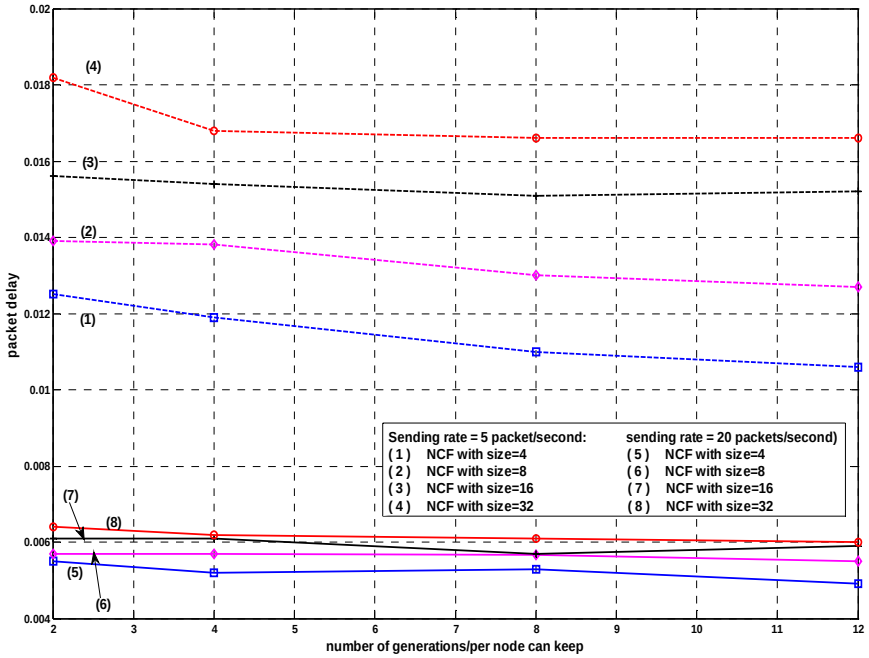


Figure 5.11: Packet delay for NCF and PFA with different generation size and different number of generations per node can have. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

The simulation results are given in Figure 5.10 and 5.11:

1) Similarly to Section 5.2, increasing generation size will improve PDR (Figure 5.10); meanwhile, packet delay will be increased (Figure 5.11).

2) Through studying the simulation results, we find that with the same generation size, keeping more generations together in one node might improve the network performance of PDR and packet delay. Let us make an example which is shown in Figure 5.12. There are three nodes in the networks and all of them can communicate with each other directly. We assume at the same time, both nodes

v_1, v_2 transmit their packets: the packet sent from node v_1 belongs to generation No.1 and the packet sent by node v_2 belongs to generation No.2 (the contents of such packets are clearly shown in the figure). If we assume only one generation can be kept in one node's buffer, in this case, after receiving packets from v_1, v_2, node v_3 can only store one packet which belongs to one generation (for the example of Figure 5.12, node v_3 keeps the packet from generation No.2 and drop generation No.1 with its packets). Then, since source packets of m_1, m_2 have not been retrieved when it is dropped, the performance of PDR is degraded. However, if we let node can contain two or more generations together at the same time, generation No.1 with its packets will not be dropped in this case, such that the probability that more source packets can be successfully retrieved is increased.

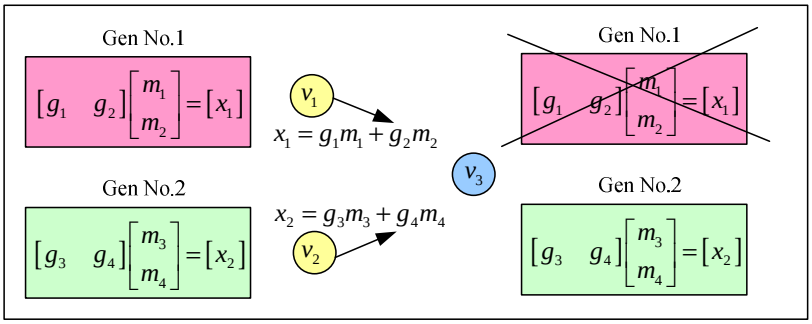


Figure 5.12: Explanation for the impact of the amount of generations one node can keep at the same time.

Brief speaking, for NCF, given that the generation size is fixed, the more generations one node is allowed to be kept at the same time, the better network performance might be achieved.

3) According to the performance of packet delay, we observe another interesting phenomenon: for network coding, increasing sending rate will remarkably reduce packet delay compared with the delay that is obtained in the network with lower sending rate. For example, if the sending rate is 5 packet/second, NCF with 16 generation size and 4 number of generations needs the packet delay of around 0.015 second, instead, if the sending rate is 20 packets/second, the packet delay is decreased by above 2 times.

It means NCF is very suitable in realistic situations, especially for high-speed –transmission networks: a better performance of packet delay will be reached when the sending rate of the network is increased. The reason is simple to explain: because of the property of network coding that each packet is dependent with each other. Although a packet which contains a new source symbol is not received by a certain node at this moment, it is still possible that this node can recover this new source symbol with the help of other packets. Assuming a network with a relatively lower rebroadcast probability, increasing sending rate will probably decrease the time of receiving sufficient number of innovative packets and then to recover source symbols.

5.3.2 Impact of Rebroadcast Probability and Sending Rate

This section shows simulation results that are related to the impact of rebroadcast probability on NCF. We use the same network environment as last section, and set rebroadcast probability as variable changing from 0% to 25%. Since we are aiming at comparing NCF with PFA, we only choose NCF with generation size of 16 and generations number of 4 as a sample to compare (also in next Section 5.3.3). The simulation results are similar with those of Section 5.2:

1) According to Figure 5.13, NCF can probably achieve PDR of 100% for a rebroadcast probability of around 12.5%, but PFA requires at least 25% rebroadcast probability to reach the same PDR. The changing of sending rate in this situation does not apparently influence the performance of PDR.

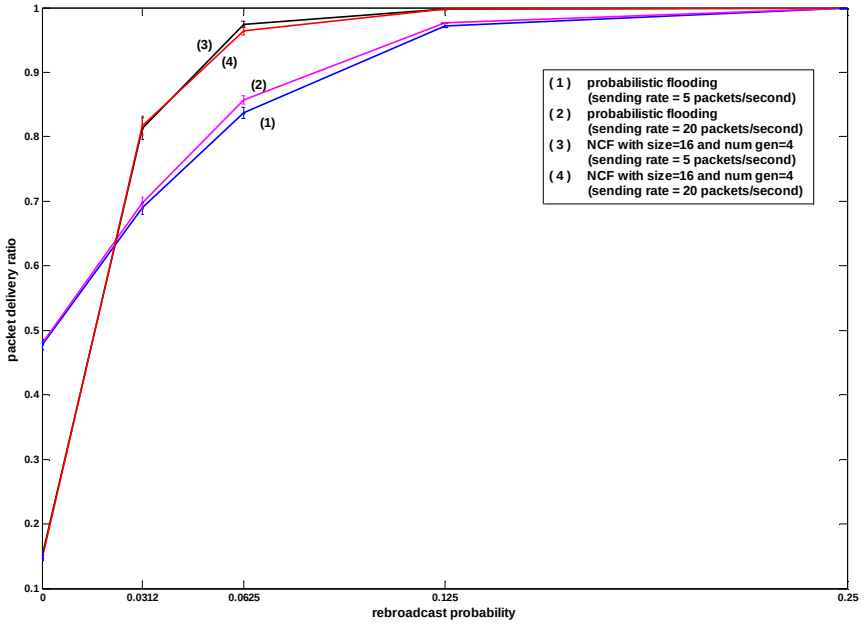


Figure 5.13: Rebroadcast probability-PDR for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

2) According to the performance of packet delay in Figure 5.14, the performance is the same as Figure 5.7. When the rebroadcast probability is extremely low, decoding delay of NCF is much longer than the performance of PFA; with increasing rebroadcast probability, the decoding delay of NCF will gradually decrease almost to the same level as PFA.

3) Obviously, if we hope NCF reach lower packet delay when the rebroadcast probability is extremely small, increasing the sending rate of the system can exactly solve this problem (reason is already given in the previous section); moreover, it will make system more practical. As shown in Figure 5.14, we assume the rebroadcast probability is around 3%, NCF with sending rate of 20

packets/second can reach the average packet delay of about 0.05 which is approximately 4 times shorter than NCF with sending rate of 5 packets/second. Of course, decreasing the generation size of NCF is another option to improve the performance of packet delay, while it will probably degrade the performance of PDR (refer to Section 5.3.1 and Figure 5.10).

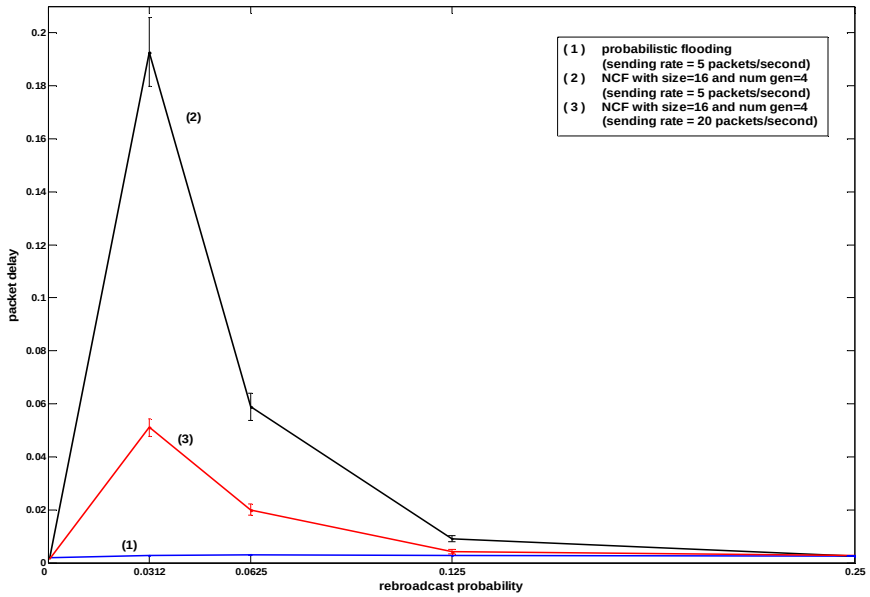


Figure 5.14: Rebroadcast probability-packet delay for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

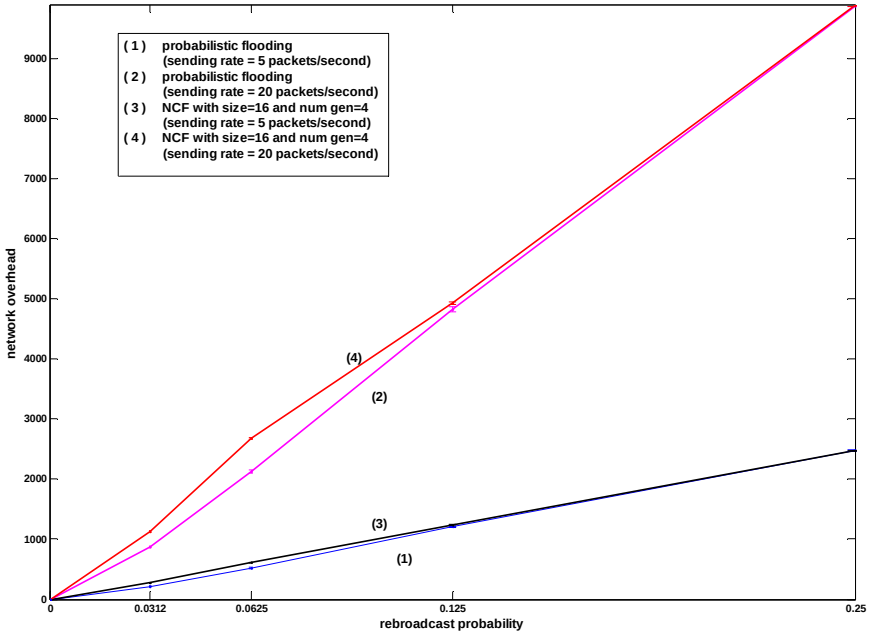


Figure 5.15: Rebroadcast probability-network overhead for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

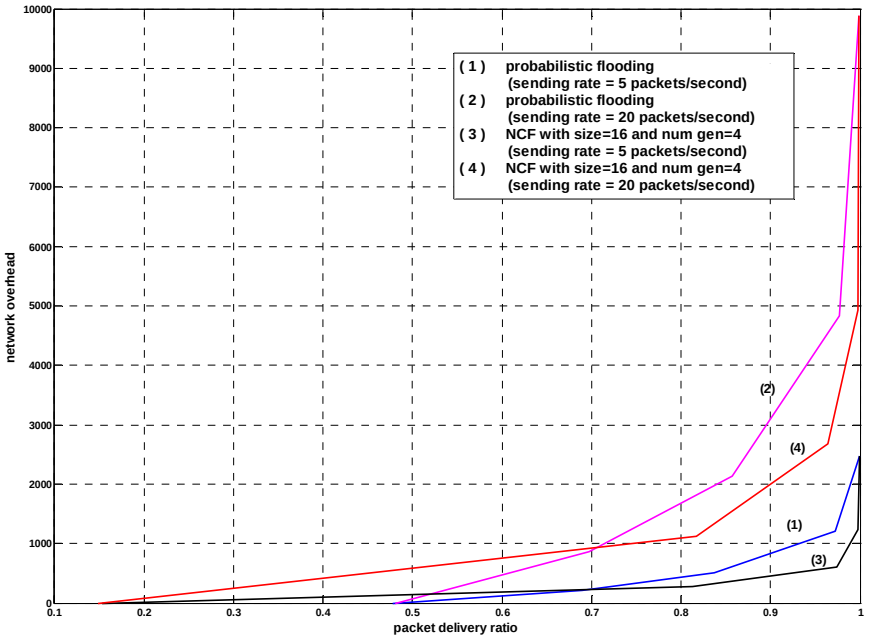


Figure 5.16: PDR-network overhead for NCF and NFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

4) The energy efficiency of NCF is also shown in Figure 5.15 and 5.16 through observing the performance of the total number of retransmission times for both two algorithms. From Figure 5.15, we find out that the number of rebroadcast times is increasing almost linearly as increasing rebroadcast probability.

   From Figure 5.16, we can clearly observe one of the advantages of NCF. Assume we are aiming at reaching PDR of 100%, the amount of retransmission times that NCF needs in both scenarios (5 packets/second and 20 packets/second sending rate) are at least one time less than those of PFA

5.3.3 Impact of Network Density and Sending Rate

In this section, we are discussing the impact of network density through changing the number of nodes in the same network from 10 to 100. We use the same network area of $500m \times 500m$ and chose rebroadcast probability of 10%. The influences of different sending rates are also considered here.

As shown in Figure 5.17, in an extremely sparse network (10 nodes in the network), PFA can reach relatively high PDR and low packet delay. As density of network is increasing, NCF shows its benefits more and more apparently. For the example of Figure 5.17, if there are 60 nodes in a network of $500m \times 500m$, the PDR of NCF is higher than PFA for at least 10 percent. According to the performance of packet delay in Figure 5.18, decoding delay of NCF is higher than PFA, but this disadvantage can be improved easily and significantly through increasing sending rate.

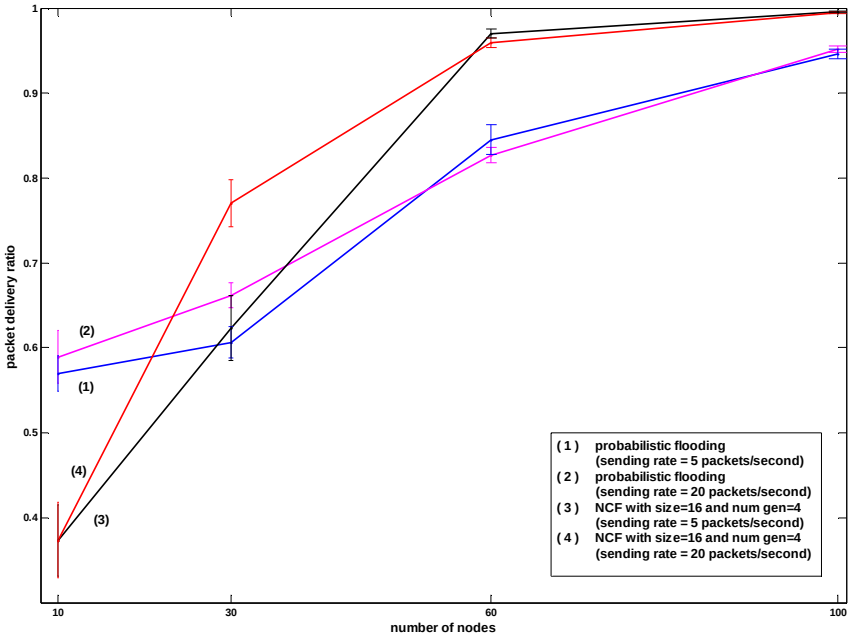


Figure 5.17: Network Density-PDR for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

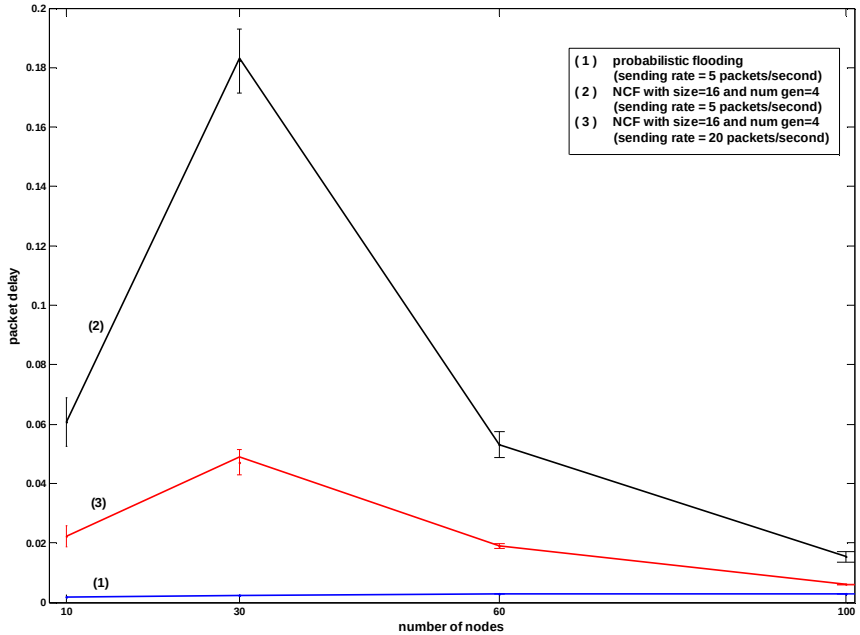


Figure 5.18: Network Density-packet delay for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

The advantage of energy efficiency of NCF can be discovered in Figure 5.19 and 5.20. According to Figure 5.19, one of the reasons that NCF is more energy efficient is displayed. For instance, assume the network density is the same, NCF can get more retransmission times than PFA. As we know, a rebroadcast can be initiated by a certain node only if it receives an innovative packet, which means for the same

network density, NCF can PFA, as each packet is independent with each other, multiple duplicates might be generated such that nodes have to drop these duplicated packets without initiating any novel rebroadcast.

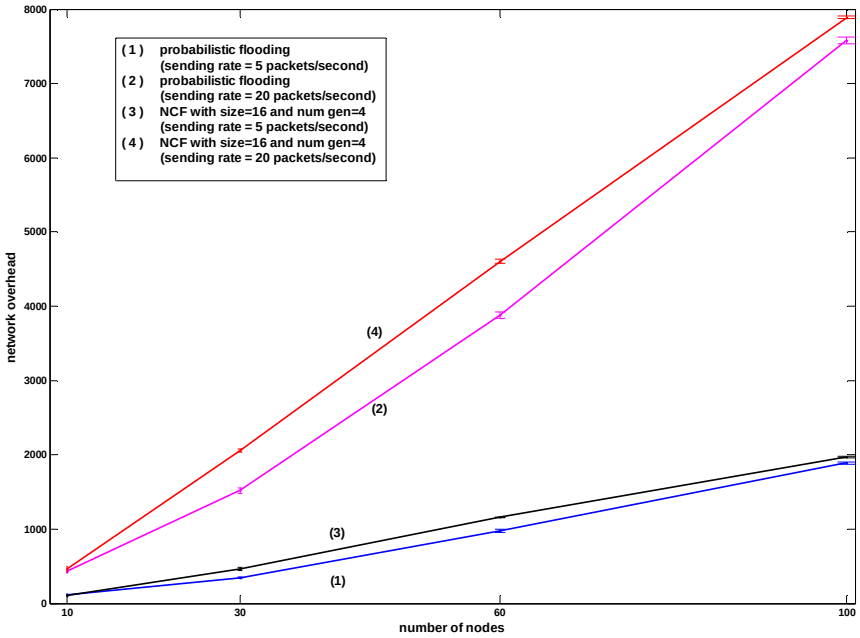


Figure 5.19: Network Density-network overhead for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

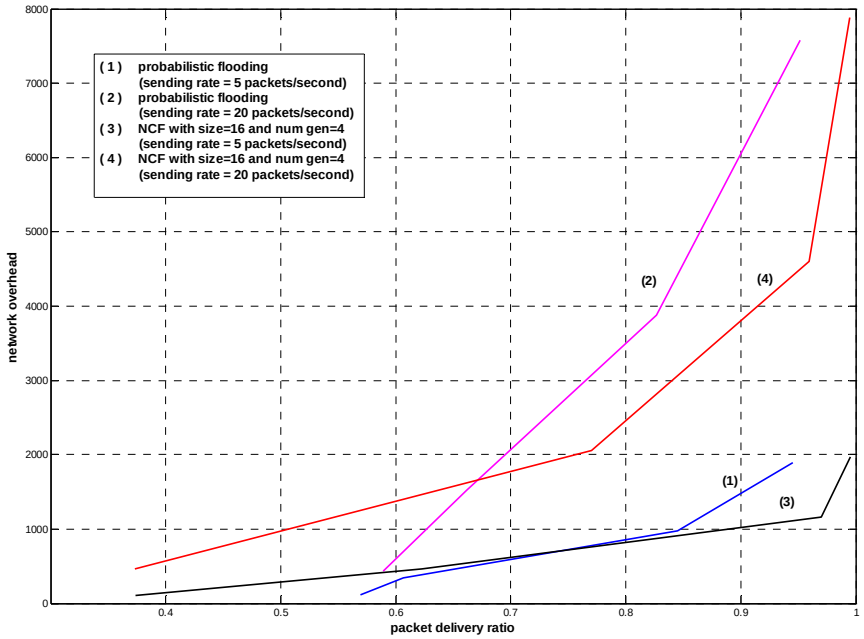


Figure 5.20: PDR-network overhead for NCF and PFA. With NCF, we choose NCF with generation size of 16 and number of generations of 4. Different sending rates are also simulated (5 packets/second & 20 packets/second).

Figure 5.20 can also clearly show NCF's advantages of energy saving. Assume network can achieve PDR of the same value (e.g. 90%), NCF needs fewer retransmission times in the contrast of PFA: for sending rate of 20 packets/second, the difference is at least 2000 times; for 5 packets/second, the difference is at least 500 times.

5.4 Trade-Off Schemes

As shown in Section 5.3, for NCF, different parameters' setting-ups will lead to totally different results. Only through correctly setting the values of related parameters, required network performance can be achieved. Therefore, after grasping the influences of these related parameters (generation size, or number of generations one node is allowed to contain), the issue how to make network efficiently balance what it needs through accurately using the related is essential to be discussed. In this section, related trade-off schemes are proposed.

Since we just want to give general ideas of how to accurately make NCF work through setting its particular parameter, the trade-off schemes proposed here are only depending on the effect of rebroadcast probability. Certainly, more concrete trade-off schemes can be discovered in the future depending on more testing results.

Based on the different effects of rebroadcast probability for network performance, we hope to give schemes from three parts (the specific values of rebroadcast probability shown in the following are obtained from our simulation results and it cannot represent general situations):

1) Network with rebroadcast probability of $0\% < p \le 3\%$;

2) Network with higher rebroadcast probability (according to our simulation results, we assume that a rebroadcast probability of larger than 25% is higher in our case);

3) Network with the rebroadcast probability of $3\% < p \le 25\%$.

We will take network performance of PDR, packet delay and network overhead into account; besides system process complexity and buffering requirement are also considered:

- **Rebroadcast probability of around** $0\% < p \le 3\%$ **or** p>25%:

  If the rebroadcast probability is around 0% which means that rebroadcast is probably not allowed upon reception of innovative packets, PDR is impossible to reach a high value since packets only can be received by its neighbors that are in the transmission range of its source. In such situation, it is better to choose the routing protocol of PFA since relatively higher PDR, less packet delay and network overhead can be achieved; furthermore, due to the simplicity of PFA, implementation cost and process complexity are at a low level. Certainly,

compared with NCF that only keeps a small part of received packets, it might need larger storage space in order to store the received packet.

If the rebroadcast probability is larger than 25%, we observe that PFA is also a better choice. After receiving an innovative packet, nodes will get opportunities to rebroadcast a packet with a higher probability such that although network coding is not used, sufficient amount of transmission will still make network reliable.

However the above network environment cannot present the common networks. Therefore, we will mainly focus on the following situation.

- **Rebroadcast probability of** $3\% < p \leq 25\%$**:**

We are mainly interested in this area since there are obvious differences between NCF and PFA and more advantages of network coding can be shown. However, different parameters' setting will lead to different network performance; therefore, we will propose several related trade-off schemes in the following which depend on different network requirements.

1) Energy-Limited Network (Figure 5.21):

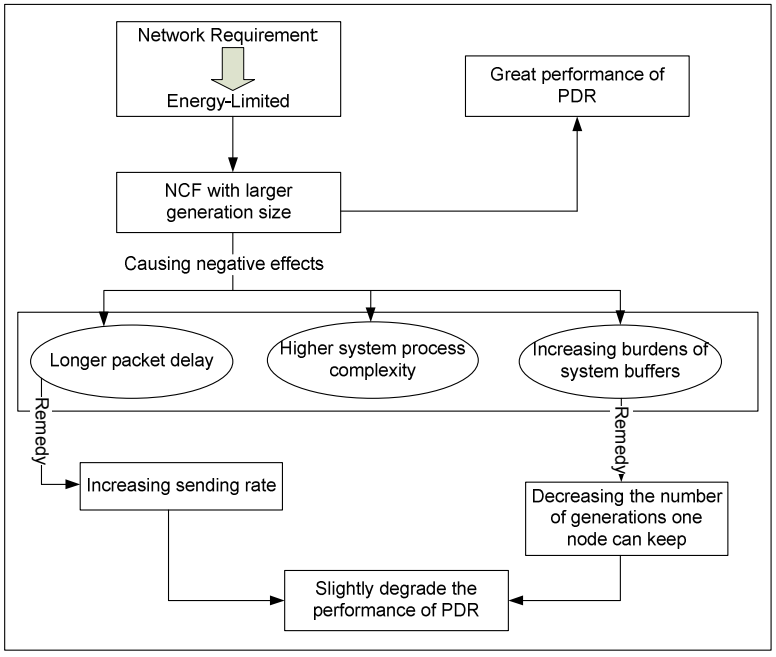


Figure 5.21: A trade-off scheme of energy-limited networks.

If it is an energy-limited network, according to Figure 5.9, 5.15, 5.16, 5.19, 5.20, reasonably increasing the generation size can save system energy; and the larger the generation size, the more system energy can be saved. Moreover, it will also lead to relatively great performance of PDR compared with using smaller generation size.

However, using larger generation size will also cause some negative effects including longer packet delay; higher system process complexity and increasing the requirement of system buffers. Through analyzing the simulation results, there are several remedies which could reduce such negative effects: increasing sending rate can significantly reduce the packet delay; if there is no enough available space to keep data, we can also decrease the number of generations one node can keep. Certainly, according to Figure 5.10 and 5.11, both remedies will slightly degrade the performance of PDR.

2) Delay-Limited Network (Figure 5.22):

If the network requires lower delay, in NCF, a best option is to make generation size smaller, which can also bring other advantages: lower system process complexity and reducing the burden of system buffering. Of course, compared with NCF with larger generation size, more energy will be consumed and the performance of PDR will also be degraded. There is one remedy that can be used to relatively improve the performance of PDR which is increasing the number of generations one node can keep (according to Figure 5.10 and 5.11).

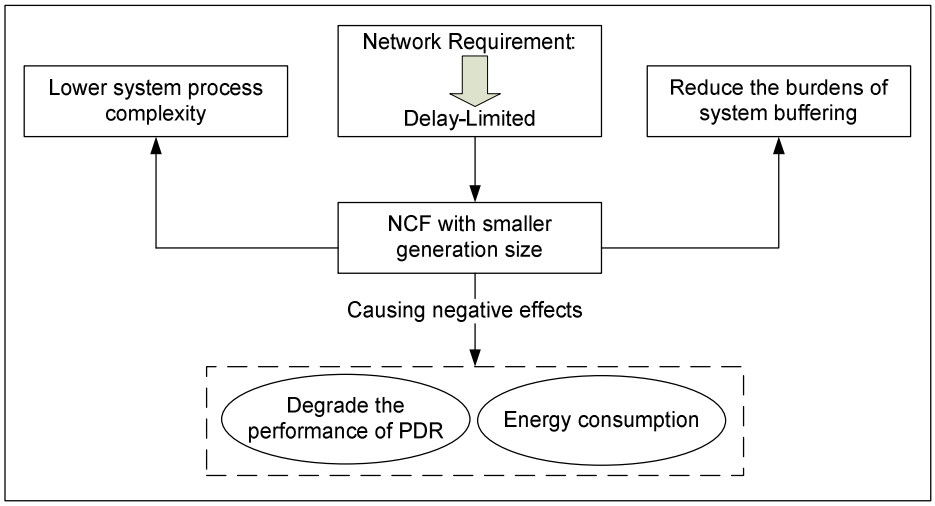


Figure 5.22: A trade-off scheme of delay-limited networks.

We propose the trade-off schemes between the network performance and limited resources, and also propose several remedies which can be used to improve the efficiency of NCF if some network requirements are limited.

Nevertheless, the schemes are proposed only through analyzing the simulation results which are implemented in our model; therefore, there are many other problems that might happen when NCF is used in other different environments or in real networks. Moreover, we only discuss the problems under the basis of rebroadcast probability; more schemes are also worth to being proposed according to other different network parameter such as different densities of network, different network areas. In addition, there is something obscure, e.g. we think NCF with larger generation size is suitable for the energy-limited networks, but we have not given the exact size of generation.

Therefore, the schemes can be further improved when more practical resultes are collected.

5.5 Summary of Simulation Study

In this chapter, we first compare NCF with NCBA and PFA. Because of limitations, the simulation environment is particular and far from practical where the amount of transmitted packets and the time that a packet is allowed to originate at its source are needed to be seriously controlled. Through simulation, we observe although NCBA can achieve about 100% PDR in a network with much less rebroadcast probability, because of its larger generation size, it causes longer packet delay, apparently increases system process and buffer's burden. In contrast, NCF can remedy NCBA's disadvantages provided the related parameters (amount of generations and generation size) are set correctly.

Thereafter, we compare NCF with PFA in a more practical scenario eliminating the unpractical method NCBA. In such environment, the amount of packets is not limited and the time of packets' origination is also random. We observe that as long as efficiently making use of the properties of NCF, more benefits can be obtained, furthermore, we also find out that NCF is much suitable for high speed transmission network which is another practical advantage.

Besides, we also propose general trade-off schemes of NCF which can be a guide for applying NCF in different network scenarios with different network demands. Through the trade-off schemes, we can also find out that NCF is more flexible than other related algorithms and it can satisfy different network requirements only through changing certain related parameters.

6. Conclusion and Future Work

6.1 Conclusion

During our research process, we have designed a new algorithm, called NCF, which can efficiently integrate network coding with one of the commonly used flooding algorithms, PFA. A simulation study is also given in last chapter including comparisons with other related algorithms and also analysis NCF's own performance. In this section, conclusions of NCF are drawn in the following:

- NCF is an integration of network coding and PFA, which is one of the commonly used network broadcasting techniques;

- NCF is a practical algorithm and designed to be used in real networks, where no synchronization is needed to control packet transmission and receiving or packets arrivals and departures; where no network topology information is needed to be known; where packet delay are allowed to happen; where huge amounts of packets are allowed to be transmitted;

- A specific "generation management method" is designed in NCF that is based on the creation time of generations and the number of source packets in a certain generation at the given time (refer to section 4.3). According to the basic idea of our generation management method, the requirement of system buffering and system process complexity is reduced compared with previous proposed related algorithms;

- A new packet format and corresponding decoding method are proposed in order to support the operation way of NCF;

- Based on the property of network coding, we observe that NCF is more suitable to be used in high-speed-packet-transmission network since the packet delay will be significantly reduced with the increasing packet sending rate;

- NCF mainly uses two related parameters, size of generations and number of generations per node has to control network performance. Depending on different network requirement, different effects can be achieved and network can easily balance its needs through correctly setting the values of these two parameters;

- NCF makes up with several limitations of previous proposed related algorithms:

    1) Compared with the generation management method used in [14] the problem of packet losses is alleviated because a different generation management is designed in NCF;

2) Each node in the network can transmit several new packets instead of only one (compare with NCBA [10]) and huge amount of packets can be transmitted simultaneously in the network;

3) NCF is suitable for random wireless networks, and there is no limitations on network architectures or topologies;

6.2 Future Work

6.2.1 Unsolved Problems

Certainly, during the research process, we find out that NCF also has several disadvantages that are worth to being solved in further research which are explained in the following:

1) According to the packet format designed in NCF, compared with former packet format (Figure 3.2), more information needs to be contained. Although the header does not need to include the whole encoding vector, it occupies too much space of packet header.

   Therefore, in the future, remedies could be designed to decrease the burden of packet header.

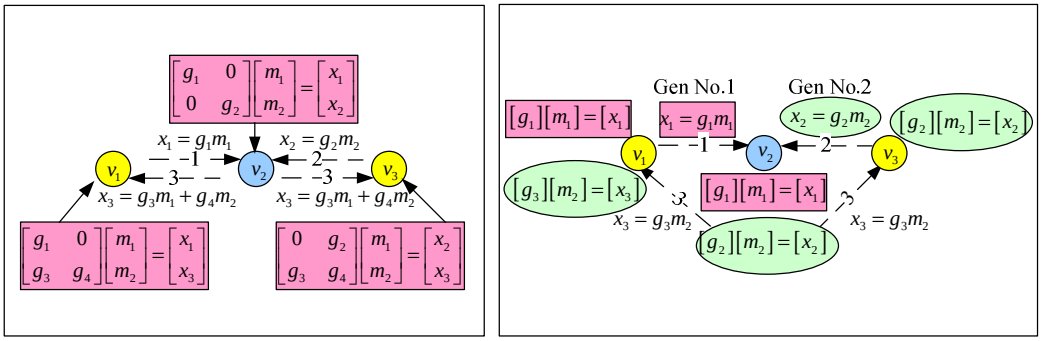


Figure 6.1: Explanation of deficiency of "Generation". Left part: no generation is allowed, and all the received packets are kept together; right part: generation is allowed, and only packets in the same generation are allowed to be encoded.

2) The main deficiency is about the problem of energy saving caused by "Generation". According to Figure 5.8, 5.15, 5.16, 5.19 and 5.20, the degree of energy saving of NCF is relatively lower than that of NCBA. Considering a simple example in Figure 6.1, there are three nodes v_1, v_2, v_3 in the network, and both node v_1, v_3 send their packets including source symbols m_1, m_2 respectively at the same time. For the left figure, no generation is allowed and all the received packets are kept in the same buffer, encoding vectors of which are also kept together in the same decoding matrix. We can see that PDR of 100% can be achieved only through "three" times of transmissions; in contrast, in the

right part, generation is allowed to be used. Node v_1, v_3 transmit their packets which are appended by different generation ID (Gen No.1 and Gen No.2 in the example). Because only packets that are from the same generation can be encoded, node v_2 has to store these two packets in different generation buffers. Clearly, only three times of transmissions are not enough to reach 100% PDR and more transmissions are needed (at that moment, node v_3 in the figure cannot receive and decode source symbol m_1). Therefore, more energy might be needed in NCF.

However, because "Generation" is the main technique used in NCF which reduces computation complexity and buffering burden; besides probably decrease the packet delay, we cannot eliminate "generation".

In the future, we can focus the research on designing a different technique which can both keep generation's advantages and make up with the energy problem.

3) In the receiving part of NCF, there is another problem needed to be solved in the future. After reception of innovative packets, there are three actions might be activated by a certain network node which include "sorting the generation list", "retransmission" and "decoding". The problem we would like to point out is caused by the "decoding" action.

In NCF, when a node receives an innovative packet, the node will add the encoding vector of this packet in the end of right decoding matrix. If the information is enough for the node to solve such matrix (the number of rows is equal to the number of columns), the node will solve the whole matrix and transmit the decoded source symbols to the upper layers. As long as the matrix is solved, all the decoded source symbols have to be transmitted to the upper layers, many source symbols might be transmitted several times which are useless and thus waste the system energy since the upper layer might have already received such source symbols.

For example, assume at the moment, a certain node has a generation which contains the decoding matrix

$$\begin{bmatrix} g_1 & g_2 & g_3 \\ g_4 & g_5 & g_6 \\ g_7 & g_8 & g_9 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Since the above matrix is full, the node will solve it and transmit the decoded source symbols m_1, m_2, m_3 to the upper layer. And at the next time unit, assume the node receives a new packet, the information vector of which is $x_4 = g_{10} m_4$ and also belongs to the same generation. Certainly, the new packet has to be put in the right generation buffer and also added in the right matrix. Therefore, the decoding matrix becomes to be

$$\begin{bmatrix} g_1 & g_2 & g_3 & 0 \\ g_4 & g_5 & g_6 & 0 \\ g_7 & g_8 & g_9 & 0 \\ 0 & 0 & 0 & g_{10} \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

We can find the source symbols containing in the decoding matrix are decodable, and they will be transmitted to the upper layer as well. At this time packets x_1, x_2, x_3, x_4 have to be transmitted. Apparently, source symbols m_1, m_2, m_3 has already been received by upper layers before, extra transmissions make them duplicated and consume unnecessary energy. In the future, a reasonable method needs to be proposed to reduce the number of duplicated packets.

6.2.2 Recommendations

In this section, we would like to propose several schemes that are also worth to researching in the further work in order to further improve the functions of NCF.

1) The performances of PDR and packet delay of PFA and NCF in our simulation network model are shown in Chapter 5. However, according the previous work [9] [10], the behaviors of packet delay with the function of forwarding factor are different: their simulation results show that for PFA, the packet delay will continuously increase with increasing rebroadcast probability; for NCBA, the packet delay starts from zero and increases with forwarding factor until such parameter is larger than a certain value.

   Through our analysis, we think it is due to the different settings of both MAC layer and PHY layer in our simulation model. In [10], the MAC layer set in the simulated model is an idealized version of IEEE 802.11 with perfect collision avoidance. Therefore, a simple MAC type (it can be directly realize in NS-2 through using the command of "mac/Simple") is chosen in our model which is also an idealized version of MAC layer without RTS/CTS. However, the information about the MAC layer settings given in [10] is not very precise. Therefore, the difference between the simple MAC type (used in our model) and the MAC type used in [10] is not clear.

   We have also tried setting the MAC type of 802.11 standard including other related parameters' settings in our network model (it can also be directly realize in NS-2 through using the command of "mac/802_11"). Some preliminary results show that it will remarkably influence the simulation results and we are confident that more realistic phenomenon could be obtained. Therefore, in the future, we can simulate NCF with the MAC layer of 802.11 and other more practical scenarios in order to further discover the properties of our algorithm.

2) At this moment, we have not considered the situation that part of packet data is lost during the transmission process. In our case, we assume if a transmitted packet is received, it is a completed packet; otherwise, such transmitted packet is not received. Hence, in the next step, we can take the situation that part of data is

lost into account. Based on NCF, more functions such as packet acknowledgement can be added in order to increase the system efficiency and availability.

3) In the simulation part of the thesis, we only propose rough trade-off schemes through analyzing and studying the simulation results. For example, in an energy limited system, a larger generation size should be set such that the energy consumption of the network can be saved; or in a time-limited network, a generation with small size should be set in order to avoid longer packet delay.

   However, we have not proposed a specific trade-off scheme which can let the system easily know how larger the generation size is needed in real applications. Therefore, in future work, we can simulate NCF in real applications and propose much concrete trade-off schemes in order to efficiently balance the requirements of a particular network.

4) In our case, the standard we use to sort the contents of generation list and then control the current situation of system buffer is mainly based on the creation time and remained available memory space.

   Of course, it is only one of the methods that can be used to choose a suitable generation. Many different comparing standards can also be proposed to be used in generation list. For instance, we can also separate packets in different generations depending on their priorities in the network communications. The packets' priorities can be divided into several levels and packets that belong to the same priority level are said to be in the same generation. The order of generations in the generation list can be set from highest priority level to lowest. Thus, each time when a node wants to use a generation, it will first check the generation of a highest priority and the generation with a lowest priority level will be chosen to discard if necessary.

References

[1]. R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204-1216, 2000.

[2]. J. Edmonds, "Edge-disjoint branchings", *Combinatorial Algorithms, R. Rustin, Ed., Algorithmics Press*, New York, pp. 91-96, 1972.

[3]. C. Fragouli, J-Y. L. Boudec and J. Widmer, "Network Coding: an instant primer," *ACM SIGCOMM Computer Communication Review*, vol. 36, pp. 63-68, 2006.

[4]. P. A. Chou and Y. Wu, "Network Coding for the Internet and Wireless Networks," *IEEE Signal Processing Magazine*, vol. 24, pp. 77-85, 2007.

[5]. C. Fragouli and E. Soljanin, "Network Coding Fundamentals," *Foundations and Trends in Networking,* vol2, pp.1-133, 2007.

[6]. B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile ad hoc networks,"*ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc '02),* pp.194-205, 2002.

[7]. S-Y. Ni, Y-C. Tseng, Y-S. Chen and J-P. Sheu, "The Broadast Storm Problem in a Mobile Ad Hoc Network," *the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pp.151-162, 1999.

[8]. Y. Sasson, D. Cavin, and A. Schiper, "Probabilistic broadcast for flooding in wireless mobole ad hoc networks," *the 11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, pp. 1124-1130, 2003.

[9]. J. Widmer, J-Y.L. Boudec, "Network Coding for Efficient Communication in Extreme Networks,"*ACM SIGCOMM workshop on Delay-tolerant networking*, pp. 284-291, 2005.

[10]. C. Fragouli, J. Widmer, and J-Y.L. Boudec, "Efficient Broadcasting using Network Coding," *IEEE/ACM Transactions on Networking*, vol.16, pp. 450-463, 2008.

[11]. S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, pp. 371-381, 2003.

[12]. P. Sanders, S. Egner , and L. Tolhuizen, "Polynomial time algorithms for network information flow," *the 15th annual ACM symposium on Parallel algorithms and architectures*, pp. 286-294, 2003.

[13]. C. Fragouli, and E. Soljanin, "Decentralized Network Coding," *IEEE Information Theory Workshop*, pp. 310-314, 2004.

[14]. P. A. Chou, Y. Wu, and K. Jain, "Practical Network Coding," *the 51st Allerton Conference on Communication, Control and Computing*, 2003.

[15]. T. Ho, M. Medard, J. Shi, M. Effros and D. R. Karger, "On Randomized Network Coding," *41st Annual Allerton Conference on Communication, Control, and Computing*, 2003.

[16]. M. Halloush and H. Radha, "Network Coding with Multi-generation Mixing," *Information Sciences and Systems 42nd Annual Conference*, pp. 515-520, 2008.

[17]. A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth codes: maximizing sensor network data persistence," *SIGCOMM Computer Commun*ication *Review*, vol. 36, pp. 255-266, 2006.

[18]. C. Gkantsidis, and P. Rodriguez, "Network Coding for Large Scale Content Distribution," *INFOCOM 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 24, pp. 2235-2245, 2005.

[19]. Z. Guo, P. Xie, J-H. Cui, and B. Wang, "On applying network coding to underwater sensor networks," *the 1st ACM international Workshop on Underwater Networks*, pp. 109-112, 2006.

[20]. "NS: physical wireless default values," ~ns-2/tcl/lib/ns-default.tcl, v. 1.223, *Copyright @ Regents of the University of California*, 1996-1997.