Design and Optimization of Cooperative Wireless Networks

BY

YASAMAN KESHTKARJAHROMI
B.Sc., Shiraz University, 2007
M.Sc., University of Tehran, 2010

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2017

Chicago, Illinois

Dedicated to: My parents, My spouse, Yashar, and My Sisters, Maryam, Mitra, and Marzieh

ACKNOWLEDGMENTS

Throughout my graduate studies at UIC, I have had the support and guidance of numerous people, most notably Prof. Hulya Seferoglu, Prof. Ashfaq Khokhar, and Prof. Rashid Ansari.

I would like to give my deepest thanks to my advisor, Prof. Hulya Seferoglu. I could not have wished for a more knowledgeable and inspiring mentor than her. She taught me by example how to become a tireless researcher and an independent thinker.

I would like to express my earnest gratitude to Prof. Ashfaq Khokhar and Prof. Rashid Ansari. I will never forget the years of continuous support and technical guidance they provided since I first started my studies at UIC. I hope to continue to enjoy their support in the future and throughout my career.

I would like to thank the rest of my thesis committee members: Prof. Salim El Rouayheb and Prof. Dan Schonfeld, for their insightful comments that exposed me to alternative perspectives and helped me to improve my work.

I also would like to thank my fellow labmates, Umer, Mohamed, Kamran, Nazaneen, Xi, Raheleh, Yuxuan, and Shanyu, for joining me in stimulating discussions, sleepless nights, and ups and downs of this journey. I will miss all the fun we have had together.

Last but not least, I would like to thank my family: my parents, who were my first teachers in life, my kind in-laws, my husband who is always my greatest supporter, and my three wonderful elder sisters who colluded to sway me into their field of medicine, but with no avail!

YK

CONTRIBUTION OF AUTHORS

A version of Chapter 2 has been published in IEEE Wireless Days Conference (Keshtkarjahromi et al., 2013). Prof. Rashid Ansari and Prof. Ashfaq Khokhar helped me with the development of main ideas and improvement of the proposed algorithms.

A version of Chapters 3 and 4 has been published in IEEE Trans. on Mobile Computing (Keshtkarjahromi et al., 2016a), IEEE ICNC Conference (Keshtkarjahromi et al., 2015a), and IEEE Milcom conference, and is under review in IEEE/ACM Trans. on Networking (Keshtkarjahromi et al., 2016b). This part of my work has been under the supervision of my advisor Prof. Hulya Seferoglu who helped me with the development of main ideas, and in collaboration with Prof. Rashid Ansari and Prof. Ashfaq Khokhar who provided important feedback that helped me to improve the proposed algorithms.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF ABBREVIATIONS

D2D	Device-to-Device
IDNC	Instantly Decodable Network Coding
NCMI	Network Coding for Multiple Interfaces
CCP	Computation Control Protocol

SUMMARY

With increasing usage of wireless networks, data traffic is rapidly increasing. For instance, cellular traffic is growing exponentially and it is expected to remain so for the foreseeable future. Thus device-to-device (D2D) networking/connections are promising to relieve traffic over traditional networks and improve throughput in wireless networks. Furthermore, such D2D connections could lead to applications such as (i) information gathering from sensor networks and detection in the fusion node, (ii) error recovery from the cellular links using D2D connection support, and (iii) cooperative computation in distributed systems. In particular, we solve the following problems in this thesis: (i) information gathering and detection in wireless sensor networks, (ii) content awareness for cooperative mobile devices that employ network coding, (iii) network coding design for multiple interfaces in cooperative mobile devices, and (iv) cooperative computation in heterogeneous wireless networks.

The first problem considered in this thesis is *Decentralized Detection in Wireless Sensor Networks*. Each node in the network makes an observation about the environment and then sends it to the fusion node for the final decision. The main drawback of the existing schemes in decentralized detection is that they are not energy efficient. In this thesis, we propose an energy efficient decentralized detection, in which the nodes cooperate to send the collective data to the fusion node in a multi-hop configuration. The proposed method has longer network lifetime and less probability of detection error compared to the existing methods.

The second problem considered in this thesis is transmitting a common content to a group of wireless devices that are in close proximity of each other through using cellular and D2D links. The default operation in today's cellular systems is that each mobile device receives its data via unicast transmission over a cellular link. In this thesis, we use broadcasting content through the cellular links and also cooperation among wireless devices through D2D links such as WiFi or Bluetooth to efficiently utilize the available resources. In particular, we consider a two-phase scenario; in phase one the whole content is broadcast to all nodes from the source and in phase two the missing content is recovered thanks to cooperation among the wireless devices. It has been shown that network coding reduces the completion time as compared to no network coding in this scenario. On the other hand, direct application of existing network coding schemes yields poor performance in terms of content quality and completion time for multimedia applications with deadline constraints. In this thesis, we propose novel content-aware network coding schemes for the recovery phase that (i) maximizes the quality under the completion time constraint, and (ii) minimizes the completion time under the quality constraint

The third problem considered in this thesis, is to design network coding schemes when multiple interfaces are used in the cooperative mobile devices. We consider the same two-phase scenario as the second problem. Cooperation among mobile devices and utilizing multiple interfaces such as cellular and local area links are promising to improve throughput in the recovery phase. In particular, thanks to using different parts of the spectrum, cellular links and D2D links operate simultaneously. Thus, each mobile device receives two simultaneous network coded packets; one from the base station or the access point through the cellular links and another one from one of the cooperating nodes through D2D links. In this thesis, we consider this system setup and propose efficient network coding schemes that minimizes the required amount of time for the complete acquisition of the content in wireless devices.

The forth problem considered in this thesis, is using cooperation among mobile devices to accomplish a computationally intensive task that needs to be accomplished at a central device, called collector. Here, in order to save energy at the collector side and also reduce delay, this task is divided into multiple subtasks, each of which is performed by a device called helper, in parallel. D2D links are used for transmitting the subtasks from the collector to the helpers and also transmitting the result of the subtasks from each helper device to the collector device. The key question in this system setup is (i) how to partition the task into subtask and (ii) which partition should be assigned to each helper device. In this thesis, we propose efficient algorithms with the goal of minimizing delay for partitioning the task into subtasks and assigning them to helper devices by taking into account the heterogeneity of the helper devices.

CHAPTER 1

INTRODUCTION

The increasing applications of wireless networks, introduce higher demand for throughput and increase the data traffic over wireless networks. On the other hand, device-to-device connections between the wireless nodes can be opportunistically used to construct a cooperative system. Using such cooperative system is promising to meet the throughput demand and relieve the traffic over the network. In this thesis, we consider three applications of cooperative systems: (i) the first system consists of cooperating wireless sensor nodes that are used to detect the status of the environment, (ii) the second system consists of cooperating wireless devices that are interested in receiving the same content from a base station or the access point through the cellular links, and (iii) the third system consists of cooperating smart devices that are used in parallel to accomplish a computationally intensive task.

Cooperating wireless nodes used for detecting the status of the environment: Decentralized detection, cast as a hypothesis testing problem, involves making noisy observations at sensor nodes, locally quantizing these observations based on some decision rules, and then sending the quantized data to the fusion node for the final decision. For binary hypothesis, the goal is to decide between two states, H_0 and H_1 in the fusion node with minimum probability of error. Although the final decision is binary, the decisions at each sensor do not need to be binary. In other words, the quantization levels at the sensors may be more than two. Decentralized detection was first introduced by Tenney and Sandell in (Tenney and Sandell, 1980). It was extended by Tsitsiklis (Tsitsiklis, 1993), Varshney (Varshney, 1996), Viswanathan and Varshney (Viswanathan and Varshney, 1997) and Blum et al. (Blum et al., 1997). In (Tsitsiklis, 1993), three different configurations are introduced for decentralized detection; tree, tandem, and parallel. In tree and tandem configurations, there exists a unique path from each sensor node to the fusion node through the other nodes and data is compressed further at each intermediate node (using the intermediate node's own observation) along the path to the fusion node. Thus, these methods would not be beneficial when the observations in the sensor nodes are independent. Also link and node failure in these configurations may lead to a shutdown of the whole network and failure of detection. A large body of research exists on parallel decentralized detection configuration, where each sensor node sends its quantized information directly to the fusion node (Chamberland and Veeravalli, 2007), (Chamberland and Veeravalli, 2003). The major drawback of parallel configuration is the large amount of energy needed for transmitting the sensors' quantized data. In such a configuration, nodes farther away from the fusion node require more energy to send their information directly to the fusion node, thus making them less attractive for large networks. In this thesis, we propose an energy efficient cooperative configuration for a one-dimensional sensor network, where each node quantizes its data and sends it to the fusion node via multiple intermediate nodes.

Cooperating wireless devices interested in receiving the same content: The widely-used and popular applications in todays wireless devices introduce higher demand for throughput, and put a strain especially on cellular links. In fact, cellular traffic is growing exponentially and it is expected to remain so for the foreseeable future (cis, 2010 2015). On the other hand, cooperation among wireless devices, facilitated by improved computational, storage, and connectivity capabilities of these devices, is a promising approach to meet these demands. In a cooperative setting, the goal is to satisfy all users' demands, while in a non-cooperative setting each node compete with the other nodes to utilize the available resources to satisfy its own demands (Kramer et al., 2007), (Sendonaris et al., 2003).

In this thesis, we consider an increasingly popular application of broadcasting a common content (e.g., video), to a group of cooperating wireless nodes within proximity, where device-to-device connections such as WiFi, WiFiDirect, or Bluetooth can be opportunistically used to construct a cooperative system (Keller et al., 2012), (Seferoglu et al., 2011). E.g., a group of friends may be interested in watching the same video on YouTube, or a number of students may participate in an online education class (Keller et al., 2012). In such scenarios, the common practice in todays network is that each mobile device connects to the Internet via its cellular or WiFi connection and downloads the content from the base station or the access point individually through the cellular link. In a cooperative system, the content server may just broadcast the whole content via cellular links. However, wireless nodes may receive only a partial content due to packet losses over wireless broadcast links. The remaining missing content can then be recovered thanks to cooperation among the nodes via local area links such as WiFi or Bluetooth. Some advantages of cooperative data exchange are:

- Improved reliability, higher transfer rate, and energy efficiency: short-range communications are often much more reliable, faster, and energy efficient.
- Offloading network traffic: the cooperating nodes requests the other nodes instead of the base station or the access point to send the missing content, thus offloading the network traffic to the local area links.

In todays communication networks, network components forwards data without any processing. Network coding breaks this fundamental assumption, and it advocates that, in addition to forwarding packets, intermediate nodes should be allowed to process and combine packets (Ahlswede et al., 2000), (Li et al., 2003), (Jaggi et al., 2005), (Ho et al., 2006). By taking advantage of network coding, the overall throughput of wireless networks can be improved significantly (Katti et al., 2008). On the other hand, for applications such as video streaming, not all packets have the same importance and not all users are interested in the same quality of content. Thus, the existing network coding schemes yield poor performance for multimedia applications with deadline constraints. In this thesis, we propose efficient content-aware network coding schemes that improve content quality and completion time.

In addition, thanks to using different parts of spectrum, multiple interfaces such as cellular links and D2D links such as WiFi and Bluetooth can be used simultaneously to deliver the missing content to the wireless devices. In this thesis, we consider this problem and propose efficient network coding schemes for cooperative systems with multiple interfaces.

Cooperating smart devices used to accomplish a computationally intensive task: Computationally intensive tasks are continuously emerging in different practical applications including health, entertainment (Hassanalieragh et al., 2015), and transportation systems over mobile devices and Internet of Things (IoT). Performing computationally intensive tasks lead to increased delay and energy consumption. On the other hand, cooperative computation is promising to reduce delay and use the available resources efficiently. E.g. MapReduce (Dean and Ghemawat, 2004) framework is a programming model that is used largely as a distributed computing model. In this framework, the computationally intensive task that needs to be performed in a central device (called collector) is partitioned into multiple subtasks. Then, the collector sends these subtasks to helper devices that can perform the subtasks in parallel. At last, the helpers send their results to the collector to obtain the result of the original task. It is shown that by using coding in cooperative computation systems, the delay is reduced significantly (Lee et al., 2016). However, the existing research in this area consider equal task allocation to all helper devices. This is while, in practice, the devices used as helpers are heterogeneous in terms of computation capabilities. In order to minimize delay, the task allocation to each helper device need to be proportional to its capability. In this thesis, we consider this issue and propose efficient task allocation to helper devices by taking into account the heterogeneity of helper devices.

1.1 Thesis Contributions

In this thesis, we consider wireless networks which consist of cooperating nodes. In such networks, the nodes cooperate to achieve a common goal. Here we consider three kinds of cooperating wireless networks: (i) decentralized detection in which the wireless sensors cooperate to make a decision about the status of the environment, (ii) delivering a common content to a group of cooperating wireless devices that are close to each other, and (iii) performing a computationally intensive task. In order to use the available resources such as energy and bandwidth efficiently, the nodes cooperate to achieve the common goal in the wireless network. The key contributions of this thesis are as follows:

- An energy efficient cooperative multi-hop configuration of WSNs is proposed to solve the detection problem in large networks with two objectives: maximizing network lifetime and minimizing probability of error in the fusion node. (Keshtkarjahromi et al., 2013)
- We study the problem of cooperative data exchange under realistic constraints such as bandwidth, delay, and energy and develop content-aware network coding algorithms that minimize delay and maximize the content quality. (Keshtkarjahromi et al., 2016a), (Keshtkarjahromi et al., 2015a)

- We characterize the performance of network coding in cooperative systems with multiple interfaces. We assume that the data is broadcast in a two-phase scenario; in phase one the whole content is broadcast to all nodes from the source and in phase two the missing content is recovered by using multiple interfaces. By using all available interfaces simultaneously, the amount of time required for all wireless devices to recover their missing packets is reduced. We use the minimum amount of time for delivering the whole content to all wireless devices as the performance metric. We establish a lower bound on the performance. We also propose efficient network coding schemes for cooperative systems with multiple interfaces and develop the upper bounds on their performance. (Keshtkarjahromi et al., 2016b), (Keshtkarjahromi et al., 2015b)
- We consider the coded cooperative computation in distributed systems with heterogeneous nodes. We propose a dynamic algorithm that estimates the capability of each device and assigns subtasks to each helper device based on the estimated capability.

1.1.1 Energy Efficient Multi-hop Configuration of Decentralized Detection

In (Li and Dai, 2008) multi-hop configuration of decentralized detection in Wireless Sensor Networks (WSNs) is studied using fusion and compression in the intermediate nodes. However, the main aim is to reduce the number and size of messages. The number of bits used to quantize information at different nodes in the multi-hop setup is assumed to be fixed. Such assumptions limit the ability to study alternate paths and variable bit allocation to achieve lower probability of error (or higher information) at the fusion node. In this work, an energy efficient multi-hop configuration of WSNs is proposed to solve the detection problem in large networks with two objectives: maximizing network lifetime and minimizing probability of error in the fusion node. Our method is based on allocating the optimal number of bits to the sensor nodes and path selection. Bit allocation consists of allocating the optimum number of bits to each sensor node. Path selection consists of choosing the best intermediate hops among the cooperating nodes to relay information to the fusion nodes from the sensor nodes. (Keshtkarjahromi et al., 2013)

1.1.2 Content-Aware Network Coding Over Cooperative Wireless Networks

It is shown that in a cooperative system setup, network coding reduces the required number of packet exchanges among the wireless nodes for all users to recover their missing packets (El Rouayheb et al., 2010), (A. Sprintson, 2010). Instantly decodable network coding (IDNC) focuses on the instant decodability of the coded packets (Sadeghi et al., 2009), (P. Sadeghi, 2010), (Sorour and Valaee, 2010b), (Sorour and Valaee, 2014). Therefore, the received network coded packet can be decoded at the time of reception without waiting for additional network coded packets. This characteristic of IDNC makes it feasible for real-time multimedia applications in which packets are passed to the application layer immediately after they are decoded. Yet, for applications such as video streaming, not all packets have the same importance and not all users are interested in the same quality of content. This problem is even more interesting when additional, but realistic constraints, such as strict deadline, bandwidth, or limited energy are added in the problem formulation. We assert that direct application of IDNC in such a scenario yields poor performance in terms of content quality and completion time. We propose a novel Content-Aware IDNC scheme that improves content quality and network coding opportunities jointly by taking into account importance of each packet towards the desired quality of service (QoS). Our proposed Content-Aware IDNC (i) maximizes the quality under the completion time constraint, and (ii) minimizes the completion time under the quality constraint (Keshtkarjahromi et al., 2015a).

1.1.3 Network Coding for Cooperative Mobile Devices with Multiple Interfaces

In order to use the scarce wireless resources efficiently, we use multiple interfaces simultaneously in the cooperative system. In particular, we consider a scenario that a group of cooperative mobile devices, exploiting both cellular and local area links and within the proximity of each other, are interested in the same content. In this setup, a common content is broadcast over cellular links. However, mobile devices may receive only a partial content due to packet losses over cellular links. The remaining missing content can then be recovered by utilizing local area links in a cooperative manner. Moreover, thanks to using different parts of the spectrum, cellular links and local area links operate simultaneously. Thus, a mobile device can receive two packets simultaneously; one via cellular, and the other via local area links. We develop an efficient network coding algorithm that takes into account cooperation among the mobile devices and multiple interfaces. The performance of network coding in single-interface systems has been considered in previous work, (Katti et al., 2008), (El Rouayheb et al., 2010), (A. Sprintson, 2010), (Wang, 2010), (El Rouayheb et al., 2007), (A. Sprintson, 2010), (Tajbakhsh et al., 2014), (Yu et al., 2014), in the context of broadcasting a common content over cellular links, and repairing the missing content via (i) retransmissions over cellular links, or (ii) by exploiting local area device-to-device connections. We consider the cooperative system with multiple interfaces and (i) develop network coding schemes for cooperative mobile devices with multiple interfaces, and (ii) characterize the performance of network coding by using the number of transmissions to recover all packets as a performance metric (Keshtkarjahromi et al., 2016b), (Keshtkarjahromi et al., 2015b).

1.1.4 Computation Control Protocol for Distributed Computing Systems

Cooperative computation is used to offload computation costs on the collector device by using multiple devices as helpers. Coding is shown to improve the performance of cooperative computation (Lee et al., 2016) (Li et al., 2015), however the existing work in this area considers the equal task allocation to each helper device, which is not efficient for systems with heterogeneous helper devices. Therefore, in order to efficiently use the available resources at the helper devices in a cooperative computation system, we consider a task allocation to each helper device which is proportional to its computational capabilities. In this thesis, we consider coded computation problem with heterogeneous helpers, and (i) find the optimum task allocation to helper devices, and (ii) develop a dynamic task allocation algorithm that performs very close to the optimum solution.

1.2 Thesis Organization

The rest of the dissertation is organized as follows. In Chapter 2, we present energy-efficient cooperative wireless networks in decentralized detection application. In Chapter 3, we present Content-Aware Instantly Decodable Network Coding proposed for the cooperative systems. In Chapter 4, we present the performance of network coding in the cooperative system with multiple interfaces. In Chapter 5, we present Computation Control Protocol proposed for cooperative computation in distributed systems. In Chapter 6, we conclude the thesis.

CHAPTER 2

ENERGY EFFICIENT DECENTRALIZED DETECTION USING MULTI-HOP COOPERATIVE WIRELESS NETWORKS

The contents of this chapters are based on our work that is published in the proceedings of 2013 IEEE Wireless Days (WD) conference (Keshtkarjahromi et al., 2013)

Existing information theoretic work in decentralized detection is largely focused on parallel configuration of Wireless Sensor Networks (WSNs), where an individual hard or soft decision is computed at each sensor node and then transmitted directly to the fusion node. Such an approach is not efficient for large networks, where communication structure is likely to comprise of multiple hops. On the other hand, decentralized detection problem investigated for multi-hop networks is mainly concerned with reducing number and/or size of messages by using compression and fusion of information at intermediate nodes. In this chapter, an energy efficient multi-hop configuration of WSNs is proposed to solve the detection problem in large networks with two objectives: maximizing network lifetime and minimizing probability of error in the fusion node. This optimization problem is considered under the constraint of total consumed energy. The two objectives mentioned are achieved simultaneously in the multi-hop configuration by exploring tradeoffs between different path lengths and number of bits allocated to each node for quantization. Simulation results show significant improvement in the proposed multi-hop configuration compared with the parallel configuration in terms of energy efficiency and detection accuracy for different size networks, especially in larger networks.

2.1 Related Work and Contributions

Decentralized detection was first introduced by Tenney and Sandell in (Tenney and Sandell, 1980). It was extended by Tsitsiklis (Tsitsiklis, 1993), Varshney (Varshney, 1996), Viswanathan and Varshney (Viswanathan and Varshney, 1997) and Blum et al. (Blum et al., 1997). In (Tsitsiklis, 1993), three different configurations are introduced for decentralized detection; tree, tandem, and parallel. In tandem and tree configurations, each sensor node quantizes its observation based on the quantized data it has received from the predecessor nodes and sends it to its successor nodes. The last node makes the final decision. In such networks, if one node runs out of energy or a link fails, the whole network shuts down. In parallel configuration, each node quantized its observations independently and sends it directly to the fusion node for the final decision. Chamberland and Veeravali, (Chamberland and Veeravalli, 2007) and (Chamberland and Veeravalli, 2003), investigate the problem of decentralized detection in sensor network applications, by considering resource constraints, such as spectral bandwidth, processing power, and cost, assuming that the information from the sensor nodes to the fusion node is transmitted over a wireless channel. In (Masazade et al., 2010), the decision thresholds at each sensor node is determined to minimize the probability of error and the total consumed energy in parallel configuration. The main drawback of parallel configuration is that it is not energy efficient. On the other hand, multi-hop cooperative configuration, in which the quantized data can be sent to the fusion node through multiple hops, energy consumption is reduced significantly. In (Li and Dai, 2008) multi-hop configuration of decentralized detection in Wireless Sensor Networks (WSNs) is studied using fusion and compression in the intermediate nodes. However, the main aim is to reduce the number and size of messages. The number of bits used to quantize information at different nodes in the multi-hop setup is assumed to be fixed. Such assumptions limit the ability to study alternate paths and variable bit allocation to achieve lower probability of error (or higher information) at the fusion node.

In this chapter, we study the problem of decentralized event detection, formulated as a binary hypothesis testing, for one dimensional sensor networks. In our current work, we assume that the observations at the sensors are independently distributed. We also assume that energy consumption at each sensor node is proportional to the number of quantization bits and the square of transmission distance. Thus sending data to the fusion node via multiple hops (multi-hop configuration) results in reducing the total energy consumed by the network compared with sending data directly (parallel configuration). Two optimization objectives, minimizing the probability of error in the fusion node and maximizing network lifetime, are considered under the constraint of total consumed energy to formulate our problem. The network lifetime is defined as the time it takes for the first node to deplete all its energy. Our method is based on allocating the optimal number of bits to the sensor nodes (with Maximum Likelihood Ratio (MLR) test as the decision rule at each sensor node) and determining the optimal quantization thresholds. To the best of our knowledge, no past work has investigated the benefit of optimal bit allocation amongst the sensor nodes in decentralized detection. In this chapter, multi-hop as well as parallel configuration are analyzed and compared with each other. The two objectives mentioned are achieved simultaneously in multi-hop configuration by taking advantage of the two degrees of freedom provided by multi-hop configuration of WSNs, namely path selection and bit allocation. Path selection consists of choosing the best intermediate hops to relay information to the fusion node from sensor nodes. Bit allocation consists of allocating the optimum number of bits to each sensor node. However in the case of parallel configuration these two objectives cannot be achieved simultaneously and thus are considered separately (with two different bit allocation methods). This is due to the fact that in parallel configuration we are not free to choose paths from sensor nodes to the fusion node by definition. Thus only bit allocation can be performed to satisfy either of the objectives but not both at the same time. As shown in the simulation results, multi-hop configuration significantly outperforms the parallel configuration in terms of information quality and energy consumption. More improvements were shown for larger network sizes.

2.2 Decentralized Detection Problem

Let us assume $\mathbf{Y} = [Y_1, Y_2, ..., Y_l, ..., Y_L]$ denotes a sequence vector of measurements observed over all sensor nodes, such that $Y_j = [y_j^1, y_j^2, ..., y_j^T]$ represents the measurements at sensor node j, $1 \leq j \leq L$, and y_j^t denotes a single instance of measurement at sensor node j at time instance t, $1 \leq t \leq T$. Let us further assume that the sequence of observations over all sensor nodes at a time instance, has the probability density function of $f_{\mathbf{y}|H}(\mathbf{y}|H_i), i = 0, 1$. Each sensor quantizes its observation according to the decision rule $\gamma_l^t : y_l^t \longrightarrow u_l^t$, and then sends the quantized information to the fusion node for final decision about the state according to the decision rule $\gamma_0 : \mathbf{U} = [U_1, U_2, ..., U_l, ..., U_L] \longrightarrow u_0$.
The goal in decentralized detection is to estimate the state in the fusion node with minimum probability of error. $\alpha = p(u_0 = H_1|H_0)$ is the probability of error, when the actual state in the environment is H_0, while the decision of the fusion node is H_1. Similarly, $\beta = p(u_0 = H_0|H_1)$ is the probability of error, when the actual state is H_1, while the decision of the fusion node is H_0. For hypothesis testing two different formulations have been used, Bayesian and Neyman-Pearson formulations. In (Chamberland and Veeravalli, 2003), the Chernoff information and Kullback-Leibler divergence are introduced as metrics to measure the probability of error in Bayesian and Neyman-Pearson formulations respectively.

Bayesian Formulation: In Bayesian formulation of binary hypothesis testing, a probability is assigned to H_0 and H_1 and the goal is to minimize the probability of error, $p_e = \pi_0\alpha + \pi_1\beta$, in which π_0 is the a priori probability of state H_0 and $\pi_1 = 1 - \pi_0$ is the probability of state H_1. The achievable upper bound for the error is given by ((Chamberland and Veeravalli, 2003)):

$$\lim_{T\to\infty} 1/T \log p_e^{(T)} \leq \log(\sum_{\mathbf{u}} p(\mathbf{u}|H_0)^s p(\mathbf{u}|H_1)^{1-s}), \tag{2.1}$$

$\mathbf{u}$ is the sequence of all decision rules available at the fusion node and the inequality is true for all values of $0 \leq s \leq 1$. To minimize the probability of error, we search for the decision rules that minimize the upper bound or equivalently maximize the Chernoff information at the fusion node, C_0 ((Cover and Thomas, 1991)):

$$C_0 = -\min_{0\leq s\leq 1}[\log(\sum_{\mathbf{u}} p(\mathbf{u}|H_0)^s p(\mathbf{u}|H_1)^{1-s})] \tag{2.2}$$

Neyman-Pearson Formulation: In Neyman-Pearson Formulation, a constraint is imposed on one of the error probabilities, α, and the goal is to minimize the other probability of error, β:

$$\underset{\gamma_1,\gamma_2,\dots,\gamma_L}{\text{minimize}} \beta(\varepsilon), \text{ subject to } 0 < \alpha < \varepsilon < 1/2 \tag{2.3}$$

According to Stein's lemma ((Cover and Thomas, 1991) and (Chamberland and Veeravalli, 2003)), we have:

$$\lim_{\varepsilon\to 0}\lim_{T\to\infty} 1/T \log \beta(\varepsilon)^{(T)} = -D(p(\mathbf{u}|H_0)||p(\mathbf{u}|H_1)), \tag{2.4}$$

in which $D(a||b)$ is the relative entropy of a with respect to b or Kullback-Leibler divergence. Therefore, for minimizing the error, we should maximize $D(p(\mathbf{u}|H_0)||p(\mathbf{u}|H_1))$, which is equivalent to:

$$D(p(\mathbf{u}|H_0)||p(\mathbf{u}|H_1)) = \sum_{\mathbf{u}} p(\mathbf{u}|H_0)\log(\frac{p(\mathbf{u}|H_0)}{p(\mathbf{u}|H_1)}), \tag{2.5}$$

2.3 Proposed Multi-hop Cooperative Decentralized Detection Configuration

Assuming a fixed total energy budget, E, over all the sensor nodes, the objectives are to minimize the probability of detection error at the fusion node and maximizing the network lifetime. The first objective can be achieved by maximizing the amount of information transmitted to the fusion node and the second objective can be achieved by allocating the same amount of energy budget to each sensor node. Most of the existing research related to decentralized detection in WSN has assumed parallel configuration of nodes for communicating local decisions to the fusion node (Chamberland and Veeravalli, 2007), (Chamberland and Veeravalli, 2003). In such a configuration, depicted in 1, the quantized information from the sensor nodes is sent directly to the fusion node. We assert that in parallel configuration, the two objectives cannot be simultaneously achieved. In our proposed multi-hop configuration, we maximize both the amount of information in the fusion node and the network lifetime at the same time. In addition, parallel configuration is inefficient in terms of energy and information quality compared with the proposed multi-hop configuration. The required energy for transmitting bits from one node to another is proportional to the number of transmitted bits and the square of distance between the two nodes. Therefore, we consider $bits \times distance^2$ as a metric to measure the consumed energy.

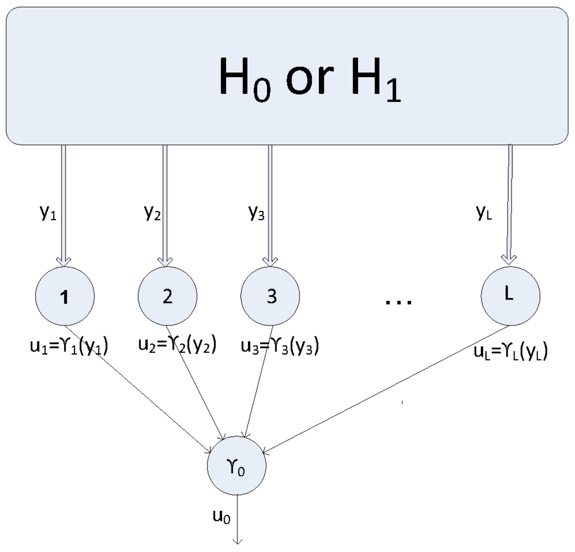


Figure 1: Parallel Decentralized Detection Configuration

2.3.1 Parallel Configuration

In this section, the two problems, maximizing the amount of information in the fusion node and maximizing network lifetime, are formulated for parallel configuration. We can solve either of the two problems by proposing the optimal bit allocation, but not both of them at the same time.

2.3.1.1 Maximizing information in the fusion node

The accuracy of estimation at the fusion node depends on the amount of information the sensor nodes relay to the fusion node, which increases with an increase in the number of bits allocated to each sensor node to quantize its observation. So, both the probability of error at the fusion node and the energy consumed for transmitting the sensors' decisions to the fusion node are functions of the number of quantization bits at each sensor node. Here, we assume that the observations of the sensors are conditionally independently distributed given each state: $p(\mathbf{u}|H_i) = \prod_{l=1}^{L} p(u_l|H_i)$ and senor node l quantizes its observations with the number of allocated bits equal to M_l. In Bayesian formulation, the Chernoff information of the fusion node is given by ((Chamberland and Veeravalli, 2003)):

$$\begin{aligned} C_0 &= -\min_{0\le s\le 1}[\log(\sum_{\mathbf{u}} p(\mathbf{u}|H_0)^s p(\mathbf{u}|H_1)^{1-s})] \\ &= -\min_{0\le s\le 1}(\log(\prod_{l=1}^{L}(\sum_{u_l=1}^{2^{M_l}} p(u_l|H_0)^s p(u_l|H_1)^{1-s}))) \\ &\le \sum_{l=1}^{L}(-\min_{0\le s\le 1}(\log(\sum_{u_l=1}^{2^{M_l}} p(u_l|H_0)^s p(u_l|H_1)^{1-s}))). \end{aligned} \tag{2.6}$$

The contribution of each sensor's decision to C_0 is no greater than: (Chamberland and Veeravalli, 2003)

$$C_l = -\min_{0\le s\le 1}(\log(\sum_{u_l=1}^{2^{M_l}} p(u_l|H_0)^s p(u_l|H_1)^{1-s})), \tag{2.7}$$

C_l increases with an increase in the number of allocated bits, M_l. Therefore, an upper bound for it is:

$$C_l < C_l^* = \lim_{M_l\to\infty} C_l \tag{2.8a}$$

$$= -\min_{0\le s\le 1}(\log(\int f(y_l|H_0)^s f(y_l|H_1)^{1-s} dy_l)). \tag{2.8b}$$

Since $C_l, l = 1,2,...,L$ is an increasing function of M_l and has the limit of C_l^* as $M_l \to \infty$, it is a concave function of M_l. Our minimization problem is aimed at determining the decision rules of the sensors that maximize the Chernoff information in the fusion node subject to the constraint of total consumed energy. The decision rule at each sensor node is completely identified by its number of allocated bits and the decision regions. To solve this optimization problem, first we find decision regions for each sensor, which maximizes its Chernoff information, C_l, for different values of allocated bits, M_l. The decision rule used at each sensor node for quantization is Maximum Likelihood Ratio (MLR) test; for a fixed number of bits, M_l, the optimum decision regions are obtained accordingly: $\gamma_l^{\text{opt}}(M_l) = \arg\max_{\gamma_l(M_l)} C_l(M_l)$, where,

$$\begin{aligned} C_l(M_l) = -\min_{0\leq s\leq 1}(\log(\sum_{u_l=1}^{2^{M_l}}(\int_{y_l\in\gamma_l^{-1}(u_l)} f(y_l|H_0)dy_l)^s \\ (\int_{y_l\in\gamma_l^{-1}(u_l)} f(y_l|H_0)dy_l)^{1-s})), \end{aligned} \tag{2.9}$$

Therefore, based on the probability density function of observations in the sensor nodes, the increasing and concave function of $C_l(M_l)$ can be obtained.

In the next step, we solve the bit allocation problem amongst the sensor nodes, which determines $\mathbf{M} = [M_1, M_2, ..., M_l, ..., M_L]$. In parallel configuration with equal bit allocation amongst nodes, data generated by the nodes farther from the fusion node consume more energy than the closer nodes to reach the fusion node. In our proposed method, bit allocation is performed based on the contribution of each quantized data to the information in the fusion node and its distance from the fusion node. Thus, the energy required for data to reach the fusion node is proportional to its contribution to the information in the fusion node. The energy required by M_l bits (quantization bits from sensor node l's observations with the distance of d_l from the fusion node) to reach the fusion node is equal to $E_l = M_l \times d_l^2 (bits \times distance^2)$ and its contribution to the information in the fusion node equals $C_l(M_l)$. Thus, for the two sensor nodes l_1 and l_2 we have:

$$\frac{M_{l_1} \times d_{l_1}^2}{M_{l_2} \times d_{l_2}^2} = \frac{C_{l_1}(M_{l_1})}{C_{l_2}(M_{l_2})} \Rightarrow \frac{C_{l_1}(M_{l_1})/M_{l_1}}{C_{l_2}(M_{l_2})/M_{l_2}} = \frac{d_{l_1}^2}{d_{l_2}^2} \tag{2.10}$$

Algorithm 1 Bit Allocation in Parallel Configuration for Maximizing Information in the Fusion Node

```
1: Number the sensor nodes in the order of their distances from the fusion node ($d_1 \geq d_2 \geq ...d_L$).
2: $M_1 \leftarrow 1$.
3: $M_l \leftarrow \arg\min_{M_l}(\frac{d_l^2}{d_1^2} \times C_l(M_1)/M_1 - C_l(M_l)/M_l)$ for $l = 2, 3, ..., L$, and $M_l = 0, 1, 2, ...$ (equation 2.10).
4: while $\sum_{l=1}^{L} M_l \times d_l^2 < E$ do
5:     $M_1 \leftarrow M_1 + 1$.
6:     $M_l \leftarrow \arg\min_{M_l}(\frac{d_l^2}{d_1^2} \times C_l(M_1)/M_1 - C_l(M_l)/M_l)$ for $l = 2, 3, ..., L$
7: if $\sum_{l=1}^{L} M_l \times d_l^2 > E$ then
8:     $M_1 \leftarrow M_1 - 1$.
9:     if $M_1 \equiv 0$ then
10:        Report the bit allocated to the first node as zero.
11:        Consider the original sensor network excluding sensor node 1; go to step 1
12: Report allocated bits as obtained in the algorithm.
```

In an unrealistic case, where M_l can be any real value, equation 2.10 is an exact equality. However, in practice M_l can only be an integer value. Thus, equation 2.10 will become an approximation. Since, $C_l(M_l)$ is a concave function, $C_l(M_l)/M_l$ decreases with an increase in M_l. If the observations in the sensor nodes are identically distributed, then $C_{l_1}(M) = C_{l_2}(M) = C_l(M)$. In this case, using equation 2.10 results in allocating more bits to the nodes with smaller d_l. In other words, the nodes farther from the fusion node are allotted less number of bits and nodes closer to the fusion node are allotted more bits. In our proposed method, we first determine the maximum possible value for M_1, the number of bits allocated to the farthest sensor node from the fusion node. Then the allocated bits to the other sensor nodes can be obtained from the ratio of their distances from the fusion node to d_1 (equation 2.10). Since M_l can take only integer values, algorithm 1 is proposed for determining bit allocation amongst the sensor nodes. This algorithm, which is based on equation 2.10, outputs the nearest integer values for the allocated bits that satisfy the inequality of $\sum_{l=1}^{L} M_l \times d_l^2 \leq E$.

2.3.1.2 Maximizing network lifetime

In a parallel configuration, the nodes farther from the fusion node consume more energy than the closer nodes and therefore run out of battery sooner. Therefore, they cannot send their information to the fusion node, which results in decreased amount of information and thus increased probability of error in the fusion node. Therefore, we define the network lifetime as the time elapsed until the first sensor node in the network depletes its energy. For maximizing the network lifetime, we try to distribute the energy consumption evenly among the sensor nodes, so that each sensor node consumes the same amount of energy as the others. Therefore, from the total energy budget, E, the amount of energy designated to each sensor node is E/L, with L be the number of sensor nodes in the network. Thus, in order to guarantee that the total consumed energy is less than E ($\sum_{l=1}^{L} M_l \times d_l^2 \leq E$), we have the following solution for bit allocation ($M_l, l = 1, 2, \ldots$ must be integer values) among the sensor nodes:

$$M_l = \left\lfloor \frac{E/L}{d_l^2} \right\rfloor, l = 1, 2, \ldots, L \tag{2.11}$$

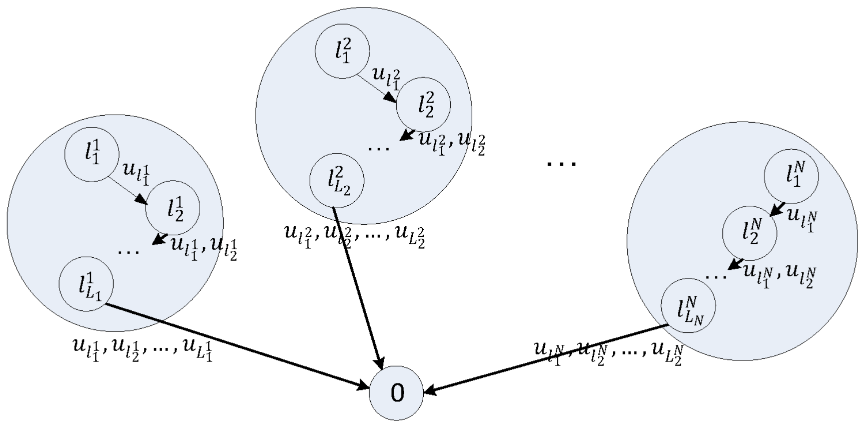

l_1^2
$u_{l_1^2}$
l_2^2
$\dots$ $u_{l_1^2}, u_{l_2^2}$
$l_{L_2}^2$
$u_{l_1^2}, u_{l_2^2}, \dots, u_{L_2^2}$
l_1^1
$u_{l_1^1}$
l_2^1
$\dots$ $u_{l_1^1}, u_{l_2^1}$
$l_{L_1}^1$
$u_{l_1^1}, u_{l_2^1}, \dots, u_{L_1^1}$
$\dots$
l_1^N
$u_{l_1^N}$
l_2^N
$\dots$ $u_{l_1^N}, u_{l_2^N}$
$l_{L_N}^N$
$u_{l_1^N}, u_{l_2^N}, \dots, u_{L_2^N}$
0

Figure 2: Multi-hop Configuration of Decentralized Detection

In practice, the number of quantization bits cannot be more than a fixed value (e.g. 8 bits). Here, we assume that the sensor nodes are located in the network such that the allocated bits to each sensor node are not more than a fixed number of bits.

2.3.2 Multi-hop Configuration

2 shows the multi-hop configuration of sensor nodes. As shown in the figure, each sensor node sends its quantization bits to the fusion node via multiple hops, which are selected among the other sensor nodes in the network. 2 consists of N groups such that group n consists of L_n sensor nodes. Each sensor node is assigned to a group based on its location in the network. The sum of the sensor nodes in all of the groups is the network size, L; in other words we have the equation: $\sum_{n=1}^{N} L_n = L$. Consider group $n = 1, 2, \dots, N$. The first sensor node in this group, l_1^n, quantizes its observations and sends the quantization bits to the second sensor node, l_2^n. The second sensor node also quantizes its own observations and sends its quantization bits as well as the first node's bits to the third sensor node. This

process continues until the last sensor node in the group, $l^n_{L_n}$, quantizes its own observations and sends the quantization bits from the entire sensor nodes in the group, $l^n_1, l^n_2, ..., l^n_{L_n}$, to the fusion node. In this configuration, if a sensor node runs out of energy, all of the quantization bits it relays to the fusion node are missed, which results in decreased information in the fusion node. Therefore, it is required that all of the sensor nodes consume the same amount of energy in order to maximize the network lifetime, the time until the first sensor node in the network runs out of energy (section 2.3.1.2). In addition, in order to maximize information in the fusion node, the energy spent by each sensor node's data to reach the fusion node should be proportional to its contribution to the information in the fusion node (section 2.3.1.1). We solve these two problems with the two degrees of freedom provided by multi-hop configuration: path selection and bit allocation under the constraint of total consumed energy. We propose a method in which best paths are selected to maximize the network lifetime and optimal bits are allocated to maximize the information in the fusion node.

Consider a random arrangement of L sensor nodes along with the fusion node on a straight line. Without loss of generality, we assume that the fusion node is located at the end of the line (3). If the fusion node is located in any other position (4), we divide the sensor network into two one dimensional networks such that in each of them the fusion node is located at the end of the line. Then the proposed method will be applied on each of them individually. The sensor nodes are ordered from the node farthest from the fusion node to the nearest: $l = 1, 2, ..., L$. In the beginning, the first group is formed. In 3, the sensor nodes in the first group are shown by shaded circles. In order to maximize the network lifetime, the following set of equations should be satisfied: (d_{ij} is the distance between the sensor node i and j and d_i is the distance of sensor node i from the fusion node.)

$$\begin{aligned} M_{l_1^1} \times d^2_{l_1^1 l_2^1} = (M_{l_1^1} + M_{l_2^1}) \times d^2_{l_2^1 l_3^1} = \ldots =& (\sum_{i=1}^{L_1} M_{l_i^1}) \times d^2_{l_{L_1}^1} \\ =& E/L, \end{aligned} \tag{2.12}$$

in which $d_{l_{L_1}^1}$ is the distance of the last sensor node in the first group from the fusion node.

According to the explanation given in section 2.3.1.1 for maximizing the information in the fusion node, the energy required for $M_{l_i^1}, i = 1, 2, ..., L_1$ to reach the fusion node, should be proportional to $C_{l_i^1}(M_{l_i^1})$. The consumed energy for transmitting $M_{l_i^1}$ through multiple hops in the group is equal to $M_{l_i^1} \times (\sum_{j=i}^{(L_1-1)} d^2_{l_j^1 l_{j+1}^1} + d^2_{l_{L_1}^1})$. Therefore, we have the following set of equations for the entire sensor nodes in the group:

$$\begin{aligned} & \frac{M_{l_1^1} \times (\sum_{j=1}^{(L_1-1)} d^2_{l_j^1 l_{j+1}^1} + d^2_{l_{L_1}^1})}{C_{l_1^1}(M_{l_1^1})} \\ =& \frac{M_{l_2^1} \times (\sum_{j=2}^{(L_1-1)} d^2_{l_j^1 l_{j+1}^1} + d^2_{l_{L_1}^1})}{C_{l_2^1}(M_{l_2^1})} \\ =& \ldots \\ =& \frac{M_{l_i^1} \times (\sum_{j=i}^{(L_1-1)} d^2_{l_j^1 l_{j+1}^1} + d^2_{l_{L_1}^1})}{C_{l_i^1}(M_{l_i^1})} \\ =& \ldots \\ =& \frac{M_{l_{L_1}^1} d^2_{l_{L_1}^1}}{C_{l_{L_1}^1}(M_{l_{L_1}^1})}. \end{aligned} \tag{2.13}$$

There are $2L_1 - 1$ independent equations in the above two equation sets 2.12 and 2.13. If the first

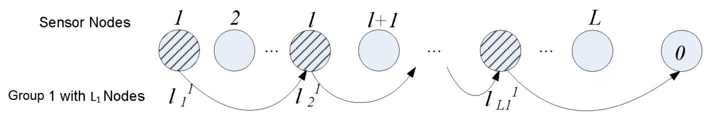


Figure 3: L Sensor Nodes Deployed on a Straight Line with the first Group of L_1 Nodes

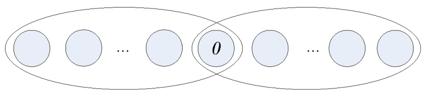


Figure 4: Sensor Network with the Fusion Node Located Somewhere Between the Nodes is Divided to Two Sensor Networks.

node of the group (l_1^1) and its allocated bits ($M_{l_1^1}$) are predefined, then the rest of the nodes in the group (determined by $d_{l_1^1 l_2^1}, d_{l_2^1 l_3^1}, ..., d_{l_{L_1-1}^1 l_{L_1}^1}$ and $d_{l_{L_1}^1}$), and their allocated bits ($M_{l_2^1}, M_{l_3^1}, ..., M_{l_{L_1}^1}$) are $2L_1 - 1$ unknowns that can be solved by the $2L_1 - 1$ equations. However, explicit solutions cannot be found by solving this system of equations for a one dimensional network with sensor nodes randomly located on a straight line. Algorithm 2 is proposed for assigning the sensor nodes to the first group and determining their allocated bits. This algorithm uses equation set 2.12 to assign nodes to the group and equation set 2.13 to determine the bit allocation. For initialization, the farthest node from the fusion node is chosen as the first node of the first group with one bit allocated to it: $l_1^1 = 1, M_{l_1^1} = 1$. At each step, the next hop for the current sensor node of the group is selected among the remaining sensor nodes in the network and then the new hop's bit allocation is determined. For instance, assume step i with the current sensor node of l_i^1. First, the sensor node l_{i+1}^1 is defined by calculating $d_{l_i^1 l_{i+1}^1}$, the distance between sensor node l_i^1 and l_{i+1}^1 using equation set 2.12. Then, $M_{l_{i+1}^1}$ is determined using equation set 2.13. At step i, the remaining sensor nodes of the group, $l_j^1, j = i+2, i+3, ..., L_1$ are not yet defined, so $d_{l_j^1 l_{j+1}^1}, j = i+1, i+2, ..., L_1 - 1$ and also $d_{l_{L_1}^1}$ are still unknown. Therefore, in using equation 2.13, the distance that the output of sensor node l_{i+1}^1 (which contains all of the quantization bits of $M_{l_j^1}, j = 1, 2, ..., i+1$) travels to reach the fusion node, is approximated by $d_{l_{i+1}^1}$. This process continues until the distance obtained from equation set 2.12 is bigger than the distance between the current sensor node and the fusion node. After forming the first group, the second group is formed similarly by considering the original sensor network excluding the sensor nodes in the first group. The next groups are formed in the similar way. The procedure ends when the entire sensor nodes are grouped.

The analysis for Neyman-Pearson formulation is similar to the analysis given for Bayesian formulation. The only difference is considering the Kullback-Leibler divergence instead of Chernoff Information. With the assumption of conditionally independently distributed observations, we have:

$$D(p(\mathbf{u}|H_0)||p(\mathbf{u}|H_1)) = \sum_{l=1}^{L} (\sum_{u_l=1}^{2^{M_l}} (p(u_l|H_0) \log(\frac{p(u_l|H_0)}{p(u_l|H_1)}))), \tag{2.14}$$

from which the contribution of each sensor and its upper bound are:

$$\sum_{u_l=1}^{2^{M_l}} (p(u_l|H_0) \log(\frac{p(u_l|H_0)}{p(u_l|H_1)})) \leq \int (f(y_l|H_0) \log(\frac{f(y_l|H_0)}{f(y_l|H_1)})) dy_l. \tag{2.15}$$

Algorithm 2 Determining the first group of multi-hop configuration and their allocated bits for a one dimensional sensor network

```
1: $i \leftarrow 1$
2: $l_1^1 \leftarrow 1, M_{l_1^1} \leftarrow 1$
3: **while** $l_i^1 < L$ **do**
4:     current sensor node $\leftarrow l_i^1$
5:     $d_{l_i^1 l_{i+1}^1} \leftarrow \sqrt[2]{\frac{E/L}{\Sigma_{j=1}^{i} M_{l_j^1}}}$
6:     **if** $d_{l_i^1 l_{i+1}^1} < d_{l_i^1}$ **then**
7:         Find a sensor node located between the sensor node $l_i^1$ and the fusion node as the next sensor node, $l_{i+1}^1$. This node should be the farthest sensor node from the current sensor node whose distance from the current sensor node is less than or equal to $d_{l_i^1 l_{i+1}^1}$.
8:         **if** No sensor node found **then**
9:             $j = \arg\max_k (M_{l_k^1}), k = 1, 2, ..., i$
10:            $M_{l_j^1} \leftarrow M_{l_j^1} - 1$
11:            **if** $\sum_{j=1}^{i} M_{l_j^1} \equiv 0$ **then**
12:                Set the nearest sensor node to sensor node $l_i^1$, which is located between sensor node $l_i^1$ and the fusion node as $l_{i+1}^1$.
13:                $M_{l_{i+1}^1} \leftarrow 1$.
14:                $i \leftarrow i + 1$.
15:            Go to step 5
16:        **else**
17:            $M_{l_{i+1}^1} \leftarrow \arg\min_{M_{l_{i+1}^1}} ((\frac{d^2_{l_{i+1}^1}}{d^2_{l_i^1 l_{i+1}^1} + d^2_{l_{i+1}^1}} \times C_{l_i^1}(M_{l_i^1})/M_{l_i^1}) - C_{l_{i+1}^1}(M_{l_{i+1}^1})/M_{l_{i+1}^1})$.
18:            $i \leftarrow i + 1$.
19:    **else**
20:        **if** $i \equiv 1$ **then**
21:            $M_{l_i^1} \leftarrow \left\lfloor \frac{E/L}{d^2_{l_1^1}} \right\rfloor$
22:        Go to 23
23: The first group with the size of $i$ nodes and their allocated bits are defined.
```

Using the procedure similar to what is discussed for Bayesian formulation, the optimum decision regions at each sensor node are obtained from:

$$\gamma_l^{\text{opt}}(M_l) = \arg\max_{\gamma_l(M_l)} \sum_{u_l=1}^{2^{M_l}} (p(u_l|H_0)\log(\frac{p(u_l|H_0)}{p(u_l|H_1)})). \tag{2.16}$$

Therefore, in Neyman-Pearson problem formulation, $\sum_{u_l=1}^{2^{M_l}}(p(u_l|H_0)\log(\frac{p(u_l|H_0)}{p(u_l|H_1)}))$ should be used instead of $C_l(M_l)$.

2.3.3 Comparison Between Parallel and Multi-hop Configurations

Direct transmission consumes more energy than transmission via multiple hops. Therefore, the total consumed energy in multi-hop configuration is less than parallel configuration with the same bit allocation among the sensor nodes. Equivalently, under the constraint of total consumed energy, more bits can be allocated to the sensor nodes in multi-hop configuration than the parallel configuration, which results in more information in the fusion node. In addition, as Chernoff information and Kullback-Leibler divergence are concave functions of number of bits, the information obtained with the first allocated bit is high (more than 60% of the upper bound as shown in 5) and with more number of bits, the increase in the information decreases. Therefore, for a fixed number of allocated bits, even distribution of bits among the sensor nodes results in more information than uneven distribution. In parallel configuration, more bits are allocated to the closer nodes to the fusion node and fewer bits are allocated to the farther nodes. While in proposed multi-hop configuration, bits are allocated among the sensor nodes more evenly. Thus, for a fixed number of allocated bits, the performance of multi-hop configuration is better than parallel configuration in terms of information in the fusion node (as shown in 7).

We briefly examine the delay incurred in multi-hop and parallel configurations. We assume that all sensor nodes use identical wireless transmission technologies. In wireless communication over a parallel configuration for a one dimensional sensor network with the fusion node located at one end, the sensor nodes cannot send their data simultaneously in order to avoid collisions when their transmission ranges overlap. As a result the maximum delay is $O(L)$, where L is the number of nodes. In our proposed multi-hop configuration, the transmissions are performed through multiple hops. As a result the maximum delay is at worst equal to that of the parallel configuration.

In the case of a link failure from sensor node l_1 to sensor node l_2 in multi-hop configuration, the entire information that sensor node l_1 should send to the fusion node is missed in the fusion node. If sensor node l_1 plays the role of intermediate node for other nodes, then the information from all of those nodes cannot reach the fusion node. While, in parallel configuration, the information of just one sensor node is missed in the fusion node in the case of link failure between any sensor node and the fusion node. From this point of view, multi-hop configuration is more sensitive to link failure than the parallel configuration.

2.4 Simulation Results

The performance of the proposed multi-hop configuration for the decentralized detection problem is evaluated by simulating in MATLAB and compared with the parallel configuration in terms of energy cost and information quality. Two sets of results are provided for the case of parallel configuration, one based on maximizing chernoff information and another based on maximizing network lifetime. We simulated Gaussian random variables of observations in the sensors; $f(y|H_0) \sim \mathcal{N}(-1,1)$ and $f(y|H_1) \sim \mathcal{N}(1,1)$ are considered for determining optimized decision rules within the sensor nodes.

We assume a single dimensional WSN field, where the nodes are randomly deployed on a straight line. The fusion node is located within 2 units distance from the end point of the line. All the results are average values over 1000 iterations.

2.4.1 Effect of Quantization

First, we need to determine decision rules of all the nodes for different values of M (number of allocated bits). Based on the fact that Gaussian distributions satisfy monotone likelihood ratio property, we determine quantization thresholds at each sensor node accordingly, for different values of M. Since the probability distributions at all of the nodes are identical, the optimization problem for all of the nodes is identical and is expressed as:

$$C_{opt}(M) = \max_{t_1,t_2,\ldots t_{2^M-1}} C_l(M), \tag{2.17}$$

and

$$\max_{t_1,t_2,\ldots t_{2^M-1}} \sum_{u_l=1}^{2^{M_l}} (p(u_l|H_0)\log(\frac{p(u_l|H_0)}{p(u_l|H_1)})) \tag{2.18}$$

for Bayesian and Neyman-Pearson formulations respectively. Here $t_i, i = 1,2,...,2^M-1$ are the thresholds that have the property $t_1 < t_2 < ... < t_{2^M-1}$ under the monotone likelihood ratio. The optimum thresholds for different number of allocated bits were obtained by simulation and are listed in I and II with four-digit accuracy of Chernoff information and Kullback-Leibler divergence. As the simulation results show, 5, the Chernoff information and Kullback-Leibler divergence are increasing concave functions of the number of allocated bits with the upper bounds of 0.5 and 2.0, respectively.

TABLE I: OPTIMIZED DECISION RULES AT EACH SENSOR NODE FOR DIFFERENT VALUES OF M, GAUSSIAN DISTRIBUTION OF BAYSIAN FORMULATION

M	thresholds	$C_{opt}(M)$
1	[0]	0.3137
2	[-1 0 1]	0.4399
3	[-1.8 -1.1 -0.5 0 0.5 1.1 1.8]	0.4824

TABLE II: OPTIMIZED DECISION RULES AT EACH SENSOR NODE FOR $M = 1,2$, GAUSSIAN DISTRIBUTION OF NEYMAN-PEARSON FORMULATION

M	thresholds	$\sum_{u_l=1}^{2^{M_l}}(p(u_l\|H_0)\log(\frac{p(u_l\|H_0)}{p(u_l\|H_1)}))$
1	[-0.6]	1.2788
2	[-1.7 -0.7 0.3]	1.7653

2.4.2 Multi-hop vs Parallel Configuration

In this section, we consider 100 sensor nodes deployed uniformly with the density of one node per unit length. The results obtained in the previous section are used in this section to simulate the performances of the proposed methods for parallel and multi-hop configurations. We have simulated

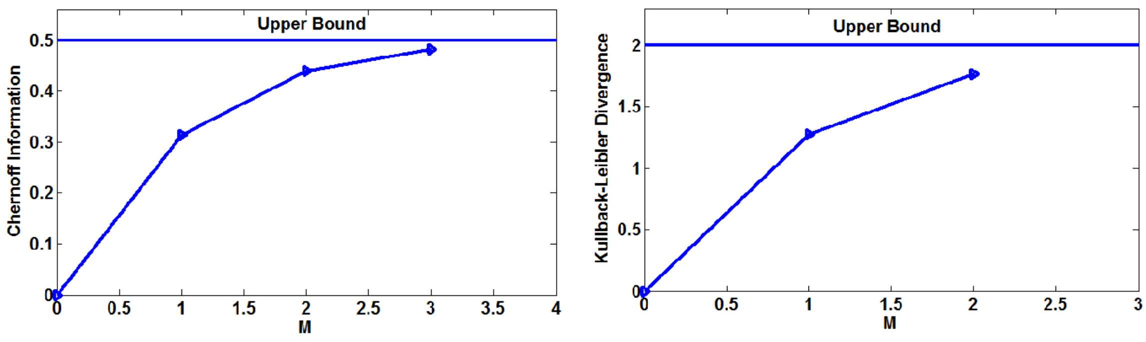


Figure 5: Optimized (a) Chernoff Information, and (b) Kullback-Leibler Divergence, for Different Values of Allocated Bits

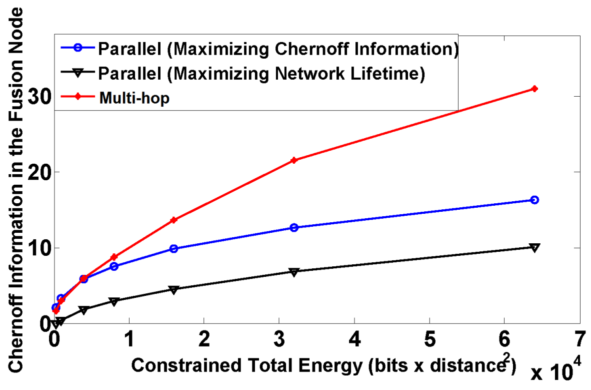


Figure 6: Chernoff Information in the Fusion Node Versus Constrained Total Energy

our proposed methods for both Bayesian and Neyman-Pearson formulations and the results show very similar trends for both cases. Here, we report only data for Bayesian formulation. 6 shows the amount of Chernoff Information in the fusion node for different values of the constrained total energy in parallel

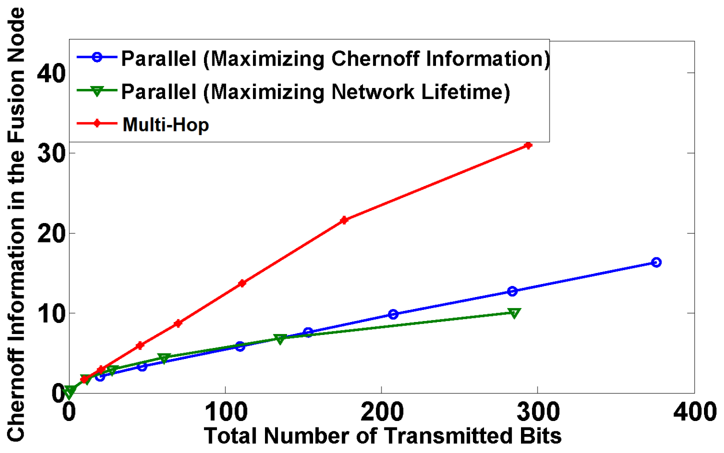


Figure 7: Chernoff Information in the Fusion Node Versus Total Transmitted Bits

and multi-hop configurations. As shown in the figure, the Chernoff Information in the fusion node obtained from the multi-hop configuration is more than the Chernoff Information obtained from the parallel configuration. Significant improvement is obtained for larger constrained total energy. 7 shows the Chernoff Information versus the total number of transmitted bits to the fusion node for multi-hop and parallel configurations. These curves are obtained by changing the value of constrained total energy and using the proposed methods for determining the bit allocation amongst nodes and computing the related Chernoff information in the fusion node. The allocated bits are divided amongst the sensor nodes more evenly in the multi-hop configuration than the parallel configuration, resulting in more Chernoff Information in the fusion node. Higher improvements were shown by considering larger total number of transmitted bits.

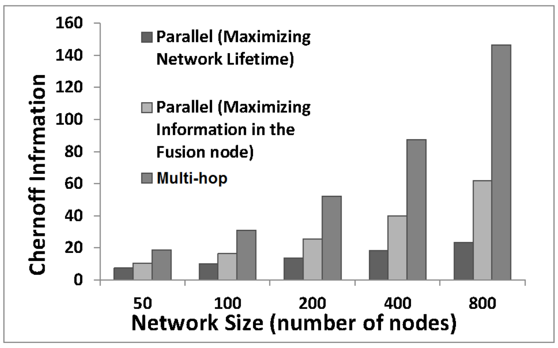


Figure 8: Chernoff Information in the Fusion Node for Different Network Sizes

2.4.3 Effect of Scaling the Network Size

In this section, we investigate the effect of network size on the performance improvement in the multi-hop configuration compared with the parallel configurations. Simulations are performed with the constrained total energy of $E = 64000(bits \times distance^2)$. 8 compares performances of the multi-hop and the parallel configurations for different network sizes. The results are presented for $L = 50, 100, 200, 400, 800$ sensor nodes, which are uniformly deployed on a straight line with the length of 100 units. We see that larger networks show significantly more improvement in terms of Chernoff Information in the fusion node.

2.5 Summary

In this chapter, an energy efficient decentralized detection was studied using a multi-hop cooperative configuration of the sensor nodes. We formulated the problem to achieve two objectives: maximizing information in the fusion node and maximizing network lifetime. We showed that in parallel configuration, where each node sends its data directly to the fusion node, the stated objectives cannot be simultaneously obtained. Whereas, in multi-hop configuration, these two objectives were achieved simultaneously using multi-hop transmission of data. Under the constraint of total energy, optimal bit allocations amongst the sensor nodes were proposed for parallel and multi-hop configurations. By taking advantage of optimal bit allocation amongst the sensor nodes, considerable improvements in terms of information quality and energy efficiency were achieved in the fusion node in multi-hop configuration as compared with parallel configuration.

CHAPTER 3

CONTENT-AWARE INSTANTLY DECODABLE NETWORK CODING OVER WIRELESS COOPERATIVE NETWORKS

The contents of this chapters are based on our work that is published in 2016 IEEE Trans. on Mobile Computing (Keshtkarjahromi et al., 2016a) and in the proceedings of 2015 IEEE International Conference on Computing, Networking and Communications (ICNC) conference (Keshtkarjahromi et al., 2015a)

Consider a scenario in which a source broadcasts a common content to a group of cooperating mobile devices that are within proximity of each other. Devices in this group may receive only partial content from the source due to packet losses over wireless broadcast links and these packet losses may differ for different devices. The remaining content missing at each device can then be recovered, thanks to cooperation among the devices by exploiting device-to-device (D2D) connections. In this context, the minimum amount of time that guarantees a complete acquisition of the common content at every device is referred to as the "completion time". It has been shown that instantly decodable network coding (IDNC) reduces the completion time as compared with no network coding in this scenario. However, for applications such as video streaming, not all packets have the same importance and not all devices are interested in the same quality of content. This problem becomes more interesting and challenging when additional, but realistic constraints, such as strict deadline, bandwidth, or limited energy are added in the problem formulation. We assert that direct application of IDNC in such a scenario yields poor performance in terms of content quality and completion time. In this chapter, we propose a novel

Content- and Loss-Aware IDNC scheme that improves content quality and network coding opportunities jointly by taking into account the contribution of each packet to the desired quality of service (QoS) as well as the channel losses over D2D links. Our proposed Content- and Loss-Aware IDNC (i) maximizes the quality under the completion time constraint, and (ii) minimizes the completion time under the quality constraint. We demonstrate the benefits of Content- and Loss-Aware IDNC through simulations.

3.1 Background

The idea behind network coding (Ahlswede et al., 2000), (Li et al., 2003), (Jaggi et al., 2005), (Ho et al., 2006) is that instead of sending a single packet through the transmission link, more packets are combined and transmitted in a single transmission. Random linear network coding (Ho et al., 2006) is a network coding scheme, in which the transmitter combines a random subset of the packets it has received and transmits through the transmission link. Network coding is effective in addressing the problem of reducing the number of packet exchanges among cooperating mobile devices (Ho et al., 2006; El Rouayheb et al., 2007; El Rouayheb et al., 2010; A. Sprintson, 2010). Instantly decodable network coding (IDNC) considers the same problem, but focuses on instant decodability (Sadeghi et al., 2009; P. Sadeghi, 2010; Sorour and Valaee, 2010b; Sorour and Valaee, 2014). In particular, a network-coded packet should be decodable by at least one of the devices in a cooperating group. This characteristic of IDNC makes it feasible for real-time multimedia applications in which packets are passed to the application layer immediately after they are decoded. However, for applications such as video streaming, not all packets have the same importance and not all devices are interested in the same quality of content. This problem becomes more interesting and challenging when additional, but realistic constraints, such as strict deadline, bandwidth, or limited energy are added in the problem formulation.

We assert that direct application of IDNC in such a scenario yields poor performance in terms of content quality and completion time. Thus, in this chapter, we propose a novel Content- and Loss-Aware IDNC scheme that improves content quality and network coding opportunities jointly by taking into account importance of each packet towards the desired quality of service (QoS) and the channel losses over D2D links. Next, we explain the operation of IDNC as well the importance of content-awareness via examples.

Example 1 *Let us consider Fig. 12, where the base station broadcasts the set of the packets* $\{p_1, p_2, p_3, p_4\}$ *to mobile devices A, B, C. These devices receive the set of packets,* H_A*,* H_B*,* H_C*, successfully from the base station and want to receive the missing packets, which are the sets* W_A*,* W_B*,* W_C*, respectively. Without network coding, four transmissions are required using D2D connections, so that each device receives all the packets. With IDNC, device A broadcasts* $p_2 \oplus p_3$ *to devices B and C, and device B broadcasts* $p_1 \oplus p_4$ *to devices A and C. After these transmissions, all devices have the complete set of packets. This example shows that IDNC has two advantages: (i) it reduces the number of transmissions from four to two, and (ii) packets are instantly decodable at each transmission;* e.g., *when device A broadcasts* $p_2 \oplus p_3$*,* p_2 *is decoded at device B and* p_3 *is decoded at device C without waiting for additional network-coded packets. These advantages make IDNC feasible for real-time multimedia applications.*
$\square$

In the context of IDNC, the minimum amount of time that can guarantee the complete acquisition of common content at every device is referred to as the "completion time". Previous works on IDNC mainly focus on reducing the completion time (Sorour and Valaee, 2010b; Sorour and Valaee, 2014). However, the interest of each device in receiving the remaining content may vary depending on the

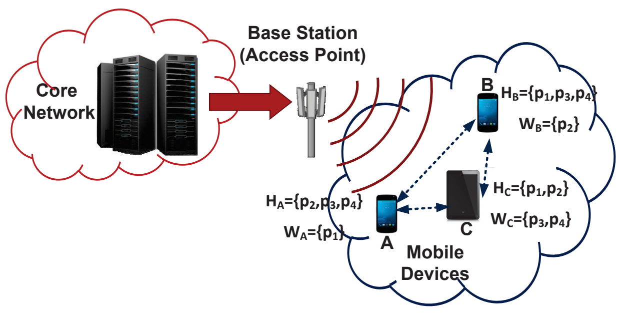


Figure 9: Mobile devices A, B, and C are in close proximity, and are interested in the same video content. As a simple example, let us assume that the video file is composed of four packets; p_1, p_2, p_3, p_4. Devices A, B, C want to receive the sets of packets; W_A, W_B, W_C, respectively. They already have the sets of packets; H_A, H_B, H_C, respectively.

information already received and the overall quality of service (QoS) requirements, such as bandwidth, energy, deadline, etc. Existing network coding or IDNC schemes under such realistic constraints yield poor performance in terms of desired QoS parameters. In the following, we further illustrate on this problem.

Example 1 - continued: Let us consider Fig. 12 again. Assume that there exists a constraint that devices should exchange their packets only in one transmission. (Note that IDNC requires two transmissions to deliver complete content to all devices.) This constraint may be due to (i) deadline or bandwidth; the packets may need to be played after one transmission, or (ii) energy; devices operating on batteries may put constraints on the number of transmissions. The question in this context is that which network code should be transmitted if there are such constraints, *i.e.,* a decision between $p_2 \oplus p_3$ or $p_1 \oplus p_4$ in the given transmission opportunity. This decision should be made based on the contents of the packets. The resulting optimization problem is the focus of this chapter. $\square$

We propose an efficient Content- and Loss-Aware IDNC which improves content quality and network coding opportunities jointly. The following are the key contributions of this work:

- We consider two content-aware optimization problems: (i) *completion time minimization* under the quality constraint, and (ii) *quality maximization* under the completion time constraint.
- We characterize the conditions that satisfy the constraints of our completion time minimization and quality maximization problems. We provide analysis of completion time and distortion by taking into account the constraints of these problems as well as the importance of each packet and the probability of channel losses over D2D links. We develop *Content- and Loss-Aware IDNC* algorithms for the quality maximization and completion time minimization problems based on our completion time and distortion analysis.
- We also provide *Content- and Loss-Aware IDNC (Heuristic)* algorithms, which are more practical as compared with *Content- and Loss-Aware IDNC* algorithms thanks to lower computational complexity.
- We evaluate our proposed Content- and Loss-Aware IDNC schemes for different number of devices and packets under the constraints of completion time and quality using real video traces. The simulation results show that Content- and Loss-Aware IDNC significantly improves completion time and quality as compared with IDNC.

3.2 Related Work

Broadcasting common content to a group of cooperating mobile devices within proximity and transmission range of each other is gaining increasing interest (cis, 2010 2015). In this scenario, mobile devices may receive only partial content due to packet losses over wireless broadcast link. The remaining missing content can then be recovered, thanks to cooperation among the devices by exploiting D2D connections. It has been shown that random network coding (Ho et al., 2006) reduces the number of transmissions necessary to satisfy all devices in the group. However, this kind of network coding, in general, requires that a block of packets be network-coded and exchanged among cooperating devices until all the devices decode all packets in the block, which makes block based network coding not suitable for delay sensitive applications.

Cooperative data exchange problems have considered designing network codes to reduce the number of transmissions in the same setup. The problem of minimizing the number of broadcast transmissions required to satisfy all devices is considered in (El Rouayheb et al., 2007). The total number of transmissions needed to satisfy the demands of all devices, assuming cooperation among devices and the knowledge of the packet sets available in each device, is minimized in (El Rouayheb et al., 2010). A deterministic algorithm that computes an optimal solution to the cooperative data exchange problem in polynomial time is proposed in (A. Sprintson, 2010). The cost and fairness issues of the cooperative data exchange problem have been considered in (Tajbakhsh et al., 2011). As compared with previous cooperative data exchange problems, the focus of this chapter is on instant decodability and content-awareness.

Instantly decodable network coding (IDNC) which requires instant decodability of the transmitted packets is introduced by (Sadeghi et al., 2009) and (P. Sadeghi, 2010). Minimization of the completion delay in IDNC has been considered in (Sorour and Valaee, 2010b), (Sorour and Valaee, 2014), and (Aboutorab and Sadeghi, 2016). Generalized IDNC which relaxes instant decodability constraint of

IDNC to target more receivers is introduced in (Sorour and Valaee, 2010a). The problem of minimizing the decoding delay of generalized IDNC in persistent erasure channels is considered in (Sorour et al., 2013). Minimization of the broadcast completion delay for IDNC with limited feedback is considered in (Sorour and Valaee, 2011a). Lossy feedback scenario is considered in (Sorour and Valaee, 2011b). IDNC is exploited in cooperative data exchange problem by making coding and scheduling decisions to generate IDNC packets in (Tajbakhsh et al., 2014). Capacity of immediately-decodable coding schemes for applications with hard deadline constraints is analyzed in (Li et al., 2011). IDNC is further relaxed in (Brahma and Fragouli, 2012), where the devices are satisfied if they receive any one message that they do not have, and in (Le et al., 2013), where the authors are interested in finding a code that is instantly decodable by the maximum number of devices. As compared with previous works on IDNC, our goal in this chapter is to develop Content-Aware IDNC.

Network coding and content-awareness have met in several previous works. Multimedia video quality improvement has been considered in (Nguyen et al., 2007), and multimedia-aware network coding scheme is developed for a broadcast and unicast scenarios for one-hop downlink topologies. One-hop opportunistic network coding scheme is considered for video streaming over wireless networks in (Seferoglu and Markopoulou, 2007). As compared with (Nguyen et al., 2007) and (Seferoglu and Markopoulou, 2007), in this chapter, we consider the packet recovery problem among cooperative mobile devices using IDNC and exploiting D2D connections. Packet prioritization is considered in IDNC (Muhammad et al., 2013), where packet prioritization is determined based on the number of requests for a packet, whereas in this chapter, content-based information is used for packet prioritization.

3.3 System Model & Problem Setup

We consider a networking model, which consists of cooperating mobile devices. Let $\mathscr{N}$ be the set of cooperating devices in our network where $N = |\mathscr{N}|$. These devices are within close proximity of each other, so they are in the same transmission range and can connect to each other via D2D links such as WiFi-Direct or Bluetooth.[1]

The cooperating mobile devices in $\mathscr{N}$ are interested in receiving the packets $p_m, m = 1, 2, \ldots, M$ from the set $\mathscr{M}$ where $M = |\mathscr{M}|$. Packets are transmitted in two stages. In the first stage, an access point or a base station broadcasts the packets in $\mathscr{M}$ to the cooperating mobile devices in $\mathscr{N}$. In this stage, the cooperating devices may receive only partial content due to packet losses over wireless broadcast link. We consider that there is no error correction mechanism in the first stage, which is dealt with in the second stage. After the first stage, the set of packets that device n has is $\mathscr{H}_n$, and is referred to as *Has* set of device n. The set of packets that is missing at device n is, $\mathscr{L}_n$ ($\mathscr{L}_n = \mathscr{M} \setminus \mathscr{H}_n$), and is referred to as *Lacks* set of device n. Each device n wants to receive all or a subset of its *Lacks* set, which is referred to as *Wants* set of device n and denoted by $\mathscr{W}_n$. Without loss of generality, we assume that for each packet $p_m \in \mathscr{M}$, there is at least one device that has received it successfully. In other words, $\forall p_m \in \mathscr{M}, \exists n \in \mathscr{N} \mid p_m \in \mathscr{H}_n$. If there exists a packet that is lost in all devices, this packet will be sent without network coding from the base station or the access point. We also assume that there is no packet in $\mathscr{M}$ that is received by all devices successfully. In other words, $\forall p_m \in \mathscr{M}, \exists n \in \mathscr{N} \mid p_m \in \mathscr{L}_n$. Note

[1]Note that we do not consider any malicious or strategic activity in our setup. We rely on possible social ties in close proximity setup for cooperation incentive and to prevent any malicious or strategic behavior.

that if there exists a packet that is received successfully by all devices, we delete this packet from the set of packets, $\mathscr{M}$. Therefore, $\mathscr{M} = \cup_{n \in \mathscr{N}} \mathscr{L}_n$.

In the second stage, the devices cooperate to recover the missing contents via their D2D connections such as WiFi-Direct or Bluetooth. Each device n is satisfied after receiving the packets in its *Wants* set; $\mathscr{W}_n$. In this stage, at each transmission opportunity the best network-coded packet with its corresponding transmitter is selected according to our Content- and Loss-Aware IDNC algorithms which we present in the next sections. Note that network coding and cooperation decisions are made by a central device, which is selected randomly among the cooperating devices. In this setup, at each transmission opportunity, a device selected as the transmitter device by the central device, broadcasts the selected network-coded packet to the other devices. The minimum amount of time that can guarantee the satisfaction of all devices $n \in \mathscr{N}$ is referred to as the "completion time"; T. In this chapter, T is defined as the number of packet transmissions that is required for all devices to be satisfied.

We denote the probability of packet loss for the D2D links by $\varepsilon_{i,j}$, where i is the transmitter device and j is the receiver device. In particular, when the transmitter device i broadcasts a packet in the local area, device j successfully receives the packet with probability $1 - \varepsilon_{i,j}$. We assume that the loss probabilities $\varepsilon_{i,j}, \forall i, j \in \mathscr{N}$ are i.i.d. according to a uniform distribution. The loss probabilities are predetermined by the central device as one minus the ratio of successfully received packets over transmitted packets in a time window, at the beginning of stage two.

Note that we consider the transmission of a large file that consists of several packets. However, for network coding and real-time operation purposes, the file is divided into smaller blocks, and network coding is performed over each block (consisting of a small number of packets). Thus, although network coding is performed over a small number of packets, loss can be calculated over a large number of packets. On the other hand, some files could be small due to their nature, *e.g.,* broadcasting small text messages. In this case, more practical and measurement-based loss calculation approaches, *e.g.,* (Draves et al., 2004) and (Kyasanur and Vaidya, 2006), can be employed, which is complementary to our work and algorithms in this chapter.

In our content-aware setup, each packet $p_m \in \mathscr{M}$ has a contribution to the quality of the overall content. We refer to this contribution as the *importance* of packet p_m. The importance of packet p_m for device n is denoted by $r_{m,n} \geq 0$.[1] The larger the $r_{m,n}$, the more important packet p_m is for device n. For example, in applications that the content is video or image, $r_{m,n}$ is calculated as the distortion of the content that device n experiences from lacking packet p_m. Therefore, the distortion value for device n is calculated as:

$$D_n = \sum_{m|p_m \in \mathscr{M}} r_{m,n} - \sum_{m|p_m \in \mathscr{H}_n} r_{m,n}. \tag{3.1}$$

The goal of traditional IDNC is “to minimize T” (Sorour and Valaee, 2010b), (Sorour and Valaee, 2014). On the other hand, Content- and Loss-Aware IDNC takes into account packet importances and distortion value D_n formulated in Eq. (3.1). In addition, we take into account the packet losses of D2D links in our formulations. In particular, we consider the following two problems:

[1]Note that the packets' importance values can be determined by the source and communicated to the central device so that it can make content-aware IDNC decisions. This information can be marked on a special field of the packet header. This field can be at the application level (*e.g.,* RTP headers) or part of the network coding header (Seferoglu and Markopoulou, 2007).

- Content- and Loss-Aware IDNC-P$_1$: Our first problem minimizes the completion time T under the quality constraint.

$$\text{minimize } T \tag{3.2}$$

$$\text{subject to } D_n \leq D_n^{cons},\ \forall n \in \mathcal{N} \tag{3.3}$$

where D_n^{cons} is the maximum tolerable distortion for device n. This problem is relevant if there are limitations on the number of transmissions due to available bandwidth or energy. *E.g.,* if devices are conservative in terms of their energy consumption, then the correct problem is to minimize the number of transmissions, which is equivalent to minimizing the completion time T, while satisfying a quality constraint; Eq. (3.3).

- Content- and Loss-Aware IDNC-P$_2$: Our second problem maximizes quality under the completion time constraint.

$$\text{minimize } f(\mathbf{D}) \tag{3.4}$$

$$\text{subject to } T \leq T^{cons}, \tag{3.5}$$

where T^{cons} is the maximum allowed completion time, $\mathbf{D}$ is the vector of per device distortions; $\mathbf{D} = [D_1, D_2, ..., D_N]$, and $f(\mathbf{D})$ is the function of the distortion vector; $\mathbf{D}$. $f(\mathbf{D})$ should be a convex function, and depending on the application, it can take different values (Ortega and Ramchandran, 1998). For example, in some applications the goal may be to minimize the sum distortion over all devices; *i.e.,* $f(\mathbf{D}) = \sum_{n\in\mathcal{N}} D_n$, while in some other applications the goal may be to minimize the maximum distortion over all devices; $f(\mathbf{D}) = \max_{n\in\mathcal{N}} D_n$ (Ortega and Ramchandran, 1998). We further explain our approach to select $f(\mathbf{D})$ in Section 3.4. The problem of minimizing $f(\mathbf{D})$ is relevant if there are constraints on delay. *E.g.,* if packets should be played out before a hard-deadline constraint; T^{cons}, then the goal is to improve the content quality as much as possible; Eq. (3.4), before the deadline; Eq. (3.5).

In the next section, we provide our solutions to *Content- and Loss-Aware IDNC-*P_1 and *Content- and Loss-Aware IDNC-*P_2.

3.4 Content- and Loss-Aware IDNC

3.4.1 Minimizing Completion Time under Quality Constraint

In this section, we present our approach to solve the problem; Content- and Loss-Aware IDNC-P$_1$ in Eqs. (3.2), (3.3).

Strategy to solve the problem: The main challenge while solving the optimization problem in Eqs. (3.2), (3.3) comes from the fact that the closed form expression for the completion time; T is an open problem. One possible approach, as also considered in previous work (Sorour and Valaee, 2010b), (Nguyen et al., 2007), is to formulate the problem as a Markov decision process, and we consider a similar approach in this chapter as explained next.

Let *state s* be the set of *Has* sets of all devices, *action a* be the selection and transmission of an IDNC packet, and the *terminating state* is any state, for which the constraint on the distortion values; Eq. (3.3) is satisfied. By considering the packet losses of D2D links, the system moves to one of the states in the Markov decision process from state s, by taking an action a. We define the completion times T and T^*

as the number of packets, required to be transmitted and received successfully at the targeted receivers, to reach the termination state (any state for which Eq. (3.3) is satisfied) from state s and s^*, respectively. Our approach is to take the action that gives the minimum average completion time over all next states in the Markov decision process. Motivated by this fact, we estimate the completion time T at the current state s as well as the completion time T^* at the next state s^*. Next, we provide the details.

Relating Completion Time to "*Wants* Sets": In our setup, as different from previous work (Sorour and Valaee, 2010b), each device does not have a fixed initial *Wants* set. Instead, each device is interested in receiving any set of packets so that Eq. (3.3) is satisfied. Indeed, for device n, L_n different *Wants* sets; $\mathscr{W}_n^l, l = 1, 2, \ldots, L_n$ could satisfy Eq. (3.3) as long as the following conditions are met.

- C$_1$: $\mathscr{W}_n^l \subseteq \mathscr{L}_n$

- C$_2$: $\sum_{m|p_m \in \mathscr{M}} r_{m,n} - \sum_{m|p_m \in (\mathscr{W}_n^l \cup \mathscr{H}_n)} r_{m,n} \leq D_n^{cons}$

- C$_3$: If $\mathscr{W}_n^{l_1} \supset \mathscr{W}_n^{l_2}, l_1, l_2 = 1, 2, \ldots, L_n$ then delete $\mathscr{W}_n^{l_1}$

It is obvious that a *Wants* set should be a subset of the *Lacks* set (the first condition; C$_1$). The second condition; C$_2$ is required to satisfy the constraint of our problem, *i.e.,* Eq. (3.3). The third condition; C$_3$ picks the set with the minimum cardinality between each pair of sets that are superset/subset of each other and deletes the other one. This condition is required to reach our objective of minimizing the number of packets to be transmitted. Let us explain the conditions; C_1, C_2, C_3 via the following example.

Example 2 *Assume that device $n \in \mathscr{N}$ is interested in receiving $M = 4$ packets with the importance values of: $r_{1,n} = 4, r_{2,n} = 5, r_{3,n} = 3, r_{4,n} = 1$, device n's* Has *set is $\mathscr{H}_n = \{p_1\}$, and its maximum tolerable distortion is equal to $D_n^{cons} = 5$. By applying the first and the second conditions, the potential* Wants *sets are: $\mathscr{W}_n^1 = \{p_2\}, \mathscr{W}_n^2 = \{p_3, p_4\}, \mathscr{W}_n^3 = \{p_2, p_4\}, \mathscr{W}_n^4 = \{p_2, p_3\}, \mathscr{W}_n^5 = \{p_2, p_3, p_4\}$. According to the third condition, ($\mathscr{W}_n^3 \supset \mathscr{W}_n^1, \mathscr{W}_n^4 \supset \mathscr{W}_n^1, \mathscr{W}_n^5 \supset \mathscr{W}_n^1$), only the first two sets are kept as potential* Wants *sets: $\mathscr{W}_n^1 = \{p_2\}, \mathscr{W}_n^2 = \{p_3, p_4\}$.* □

Now that we defined the conditions for the *Wants* sets for our problem, we can formulate the completion time in terms of *Wants* sets as follows. The completion time for device n, denoted by T_n, is equal to the minimum number of packets that it should receive successfully so that its distortion is equal to or less than its maximum tolerable distortion:

$$T_n = \min_{l=1,\ldots,L_n} |\mathscr{W}_n^l|. \tag{3.6}$$

Note that device n can benefit from a transmitted IDNC packet, if it is instantly decodable for device n and the decoded packet is a member of the set $\bigcup_{l=1}^{L_n} \mathscr{W}_n^l$. Assume that device n receives the transmitted packet successfully and decodes packet p_m, then the system moves from state s to state s^*. The completion time and the potential *Wants* sets for device n at state s^* are expressed as:

$$T_n^* = \begin{cases} T_n - 1, & \text{if } \exists l \leq L_n \mid (p_m \in \mathscr{W}_n^l \\ & \qquad \& |\mathscr{W}_n^l| = T_n) \\ T_n, & \text{otherwise} \end{cases} \tag{3.7}$$

$$(\mathscr{W}_n^l)^* = (\mathscr{W}_n^l \setminus p_m), l = 1, 2, \ldots, L_n. \tag{3.8}$$

Lower and Upper Bounds of T: The completion time, T, which is the minimum number of packets required to be transmitted and received successfully at the targeted receivers to reach the terminating state, has the lower and upper bounds of:

$$\max_{n \in \mathscr{N}} T_n \leq T \leq \sum_{n \in \mathscr{N}} T_n. \tag{3.9}$$

In particular, each device n needs at least T_n packet transmissions to be satisfied. In the worst case scenario, at each transmission, only one of the devices is targeted and its completion time is reduced by one if it receives the transmitted packet successfully. Therefore, $\sum_{n \in \mathscr{N}} T_n$ transmissions are required. This is equal to the upper bound of completion time; $T \leq \sum_{n \in \mathscr{N}} T_n$. On the other hand, the device with the maximum completion time needs to receive $\max_{n \in \mathscr{N}} T_n$ transmissions successfully. In the best case scenario, the $\max_{n \in \mathscr{N}} T_n$ packet transmissions can be chosen so that the other devices can also be targeted and satisfied by these transmissions. Note that a single transmission can benefit a subset of devices if they want the same packet or if the wanted packets are network-coded in the single transmission. The bounds in Eq. (3.9) are explained via the next example.

Example 3 *Let us consider three devices with the completion times of* $T_1 = 1, T_2 = 2, T_3 = 3$. *Obviously, device 3 needs at least three transmissions,* $T_3 = 3$, *to be satisfied,* i.e., *to receive all the packets in its* Wants *set with the minimum size,* $\min_{l=1,\dots,L_3} |\mathscr{W}_3^l|$ *(Eq. (3.6)). In the best case scenario, these three transmissions can also target and satisfy the other two devices. In other words, according to Eq. (3.7), the completion time is decreased by one for device 3 in all three transmissions, the completion time is decreased by one for device 2 in two of the transmissions and the completion time is decreased by one for device 1 in just one of the transmissions. Therefore, the lower bound for the completion time is (Eq. (3.9))* $T = \max_{n\in\mathscr{N}} T_n = 3$. *On the other hand, in the worst case scenario, at each successful transmission, just one of the devices is targeted and satisfied. Therefore, 3 transmissions are required to be received successfully at device 3 and satisfy it, 2 transmissions are required to be received successfully at device 2 and satisfy it, and 1 transmission is required to be received successfully at device 1 and satisfy it. Therefore, the upper bound for the completion time is* $T = \sum_{n\in\mathscr{N}} T_n = 1+2+3 = 6$. *In general, the completion time varies between the lower and upper bounds in Eq. (3.9).* □

Expressing T^* as a $p-$norm: As we mentioned earlier, our approach to solving Content- and Loss-Aware IDNC-P$_1$ is to take the action, *i.e.,* selecting the network code, that results in the next states s^* with the minimum average completion time. Consider the next state s^* with the completion time T^*.

Although we do not have analytically closed form formulation for the completion time; T, hence T^*, we have lower and upper bounds on T; Eq. (3.9), which also applies to T^*:

$$\max_{n \in \mathscr{N}} T_n^* \leq T^* \leq \sum_{n \in \mathscr{N}} T_n^*, \tag{3.10}$$

where T_n^* is characterized by Eq. (3.7).

Our goal is to find the network code that minimizes the average of T^* among all possible next states s^*, so let us examine the lower and upper bounds of T^* closely. The lower bound of T^* is $\max_{n \in \mathscr{N}} T_n^*$ which is actually the maximum norm (infinity norm or L_∞ norm) of the vector $\boldsymbol{T}^* = [T_1^*, T_2^*, ..., T_N^*]$, *i.e.,* the maximum norm of $\boldsymbol{T}^*$ is expressed as $\|\boldsymbol{T}^*\|_\infty = \max_{n \in \mathscr{N}} T_n^*$. On the other hand, the upper bound of T^* is $\sum_{n \in \mathscr{N}} T_n^*$, which is the L_1 norm of the vector $\boldsymbol{T}^* = [T_1^*, T_2^*, ..., T_N^*]$, *i.e.,* the L_1 norm of $\boldsymbol{T}^*$ is expressed as $\|\boldsymbol{T}^*\|_1 = \sum_{n \in \mathscr{N}} T_n^*$. Thus, $\|\boldsymbol{T}^*\|_\infty \leq T^* \leq \|\boldsymbol{T}^*\|_1$. Since the following inequality holds; $\|\boldsymbol{T}^*\|_\infty \leq \|\boldsymbol{T}^*\|_p \leq \|\boldsymbol{T}^*\|_1$, we can conclude that $T^* = \|\boldsymbol{T}^*\|_p$ for some p such that $1 < p < \infty$. Now that we know $T^* = \|\boldsymbol{T}^*\|_p$, we can select a network code which minimizes the p-norm of the average completion time among all next states; $\left\|\bar{\boldsymbol{T}}^*\right\|_p$.[1]

Taking Action: Since our goal is to select a network code which minimizes $\left\|\bar{\boldsymbol{T}}^*\right\|_p$, we should determine all possible instantly decodable network coding candidates. Then, we should select the best network code which minimizes $\left\|\bar{\boldsymbol{T}}^*\right\|_p$. A trivial approach would be to exhaustively list all possible

[1]Note that by minimizing $\left\|\bar{\boldsymbol{T}}^*\right\|_p$, instead of minimizing $\bar{T}^*$ itself, we loose optimality as we do not know the exact value of p. However, this relaxation allows us to tackle the problem. Furthermore, simulation results show that this approach provides significant improvement. The performance of IDNC for various p values is analyzed in (Sorour and Valaee, 2014). In this chapter, we consider $p = 2$.

network coding candidates, and calculate $\left\|\bar{\boldsymbol{T}}^*\right\|_p$ for each of them to determine the best one. A more efficient approach is to use a graph; IDNC graph (Sorour and Valaee, 2010b), (Sorour and Valaee, 2014). IDNC graph is constructed so that each clique in the graph corresponds to a network code. Thus, we can find the best clique to determine the best network code which minimizes $\left\|\bar{\boldsymbol{T}}^*\right\|_p$.

The IDNC graph $\mathscr{G}$ for our cooperative data exchange system consists of N disjoint IDNC local graphs. Each IDNC local graph $\mathscr{G}_t, t \in \mathscr{N}$ represents the network coding candidates that can be transmitted from device t. The IDNC local graph $\mathscr{G}_t$ for our problem is constructed as follows.

For device $n \in (\mathscr{N} \setminus t)$, $|(\bigcup_{l=1,\dots,L_n} \mathscr{W}_n^l) \cap \mathscr{H}_t|$ vertices, each vertex shown by $v_{n,m}^t$ such that $p_m \in (\bigcup_{l=1,\dots,L_n} \mathscr{W}_n^l) \quad \& \quad p_m \in \mathscr{H}_t$ are added to the graph. A pair of vertices, $v_{n,m}^t$ and $v_{k,l}^t$, are connected if one of the following conditions; $\mathrm{C}_1^{'}$ or $\mathrm{C}_2^{'}$ is satisfied:

- $\mathrm{C}_1^{'}$: $p_m = p_l$
- $\mathrm{C}_2^{'}$: $p_m \in \mathscr{H}_k \quad \& \quad p_l \in \mathscr{H}_n$.

The total number of possible actions when device t is the transmitter, *i.e.,* the number of network codes that device t can transmit, is equal to the number of cliques in the local graph $\mathscr{G}_t$. The action associated with clique $q_t \in \mathscr{G}_t$ corresponds to transmitting the network-coded packet generated by XORing all the packets associated with the clique, *i.e.,* XORing $\forall p_m$ such that $v_{n,m}^t \in q_t$. The best network code that can be transmitted from device t, hence the best clique in $\mathscr{G}_t$ is the one that minimizes $\left\|\bar{\boldsymbol{T}}^*\right\|_p$. We assign weights to each vertex in the graph so that the sum weight of all the vertices in clique q_t corresponds to $\left\|\bar{\boldsymbol{T}}^*\right\|_p$ which is resulted from sending the network code represented by the clique q_t from device t. Then, we consider graph $\mathscr{G}$, which is equal to the union of all local graphs; $\mathscr{G} = \bigcup_{t \in \mathscr{N}} \mathscr{G}_t$ and search for the clique that has the largest total weight summed over its vertices. Next, we determine the weight $w^t_{n,m}$ of vertex $v^t_{n,m}$ in clique q_t.

Assume that the network code corresponding to clique q_t is transmitted from device t. This packet is received successfully by any device n with probability of $1-\varepsilon_{t,n}$ and is lost with probability of $\varepsilon_{t,n}$. Therefore, the average of the resulting completion times, $\bar{T}^*$, will have $p-$norm; $\left\|\bar{\boldsymbol{T}}^*\right\|_p$, which is equal to:

$$\begin{aligned}\left\|\bar{\boldsymbol{T}}^*\right\|_p = (&\sum_{n|(\exists p_m|v^t_{n,m}\in q_t)} ((1-\varepsilon_{t,n})T^*_n + \varepsilon_{t,n}T_n)^p \\ &+ \sum_{n|(\nexists p_m|v^t_{n,m}\in q_t)} (T_n)^p)^{1/p} \qquad (3.11)\end{aligned}$$

Note that the completion time for device n changes from T_n to T^*_n (Eq. (3.7)) with probability of $(1-\varepsilon_{t,n})$ and does not change with probability of $\varepsilon_{t,n}$ if the selected clique covers device n, *i.e.,* it includes a vertex that represents a packet from *Wants* set of device n. The term $\sum_{n|(\exists p_m|v^t_{n,m}\in q_t)}((1-\varepsilon_{t,n})T^*_n+\varepsilon_{t,n}T_n)^p$ in Eq. (3.11) corresponds to this fact. On the other hand, the completion time for the devices that are not covered by the selected clique does not change. The term $\sum_{n|(\nexists p_m|v^t_{n,m}\in q_t)}(T_n)^p$ in Eq. (3.11) corresponds to this fact. Eq. (3.11) is expressed as;

$$\begin{aligned}\left\|\bar{\boldsymbol{T}}^*\right\|_p = (&\sum_{n\in\mathscr{N}} (T_n)^p + \\ &\sum_{n|(\exists p_m|v^t_{n,m}\in q_t)} ((1-\varepsilon_{t,n})T^*_n + \varepsilon_{t,n}T_n)^p - (T_n)^p)^{1/p} \qquad (3.12)\end{aligned}$$

Note that the first term in Eq. (3.12) is the same and fixed for all cliques in the graph. Therefore, in order to minimize $\left\|\bar{\boldsymbol{T}}^*\right\|_p$, the second term should be minimized, which corresponds to:

$$q_t^* = \arg\max_{q_t} \sum_{n|(\exists p_m|v_{n,m}^t \in q_t)} ((T_n)^p - \\ ((1-\varepsilon_{t,n})T_n^* + \varepsilon_{t,n}T_n)^p). \tag{3.13}$$

where q_t^* is the best clique and the corresponding network code is the best network code in the local graph $\mathscr{G}_t$. By substituting T_n^* from Eq. (3.7) into Eq. (3.13), the following weight assignment to vertex $v_{n,m}^t \in \mathscr{G}_t$ is obtained:

$$w_{n,m}^t = \begin{cases} (T_n)^p - (T_n - 1 + \varepsilon_{t,n})^p, & \text{if } \exists l \leq L_n \mid \\ & p_m \in \mathscr{W}_n^l \\ & \& |\mathscr{W}_n^l| = T_n \\ 0, & \text{otherwise.} \end{cases} \tag{3.14}$$

Using the weight assignments in Eq. (3.14), Content- and Loss-Aware IDNC-P$_1$ finds the network code that corresponds to the maximum weighted clique in graph $\mathscr{G}$ at each transmission opportunity until Eq. (3.3) is satisfied. Note that $\mathscr{G}$ is the union of all local graphs $\mathscr{G}_t$, $\mathscr{G} = \bigcup_{t \in \mathscr{N}} \mathscr{G}_t$.

3.4.2 Maximizing Quality under Completion Time Constraint

In this section, we present our approach to solve the problem; Content- and Loss-Aware IDNC-P$_2$ presented in Eqs. (3.4), (3.5). For the solution of Content- and Loss-Aware IDNC-P$_2$, we use a similar approach to the solution of Content- and Loss-Aware IDNC-P$_1$.

Expressing $f(\mathbf{D})$ as a $p-$norm: As we mentioned earlier, depending on the application, the distortion function $f(\mathbf{D})$ can take different values (Ortega and Ramchandran, 1998). If the goal is to minimize the sum distortion over all devices, then $f(\mathbf{D}) = \sum_{n\in\mathscr{N}} D_n$ which is actually the L_1 norm of the distortion vector; $\mathbf{D} = [D_1, D_2, ..., D_N]$, *i.e.,* $\|\boldsymbol{D}\|_1 = \sum_{n\in\mathscr{N}} D_n$. On the other hand, if the goal is to minimize the maximum distortion over all devices, then $f(\mathbf{D}) = \max_{n\in\mathscr{N}} D_n$, which is actually the maximum (infinity) norm of the distortion vector; $\mathbf{D} = [D_1, D_2, ..., D_N]$, *i.e.,* $\|\boldsymbol{D}\|_\infty = \max_{n\in\mathscr{N}} D_n$. For the sake of generality, we consider the objective function as $p-$norm of the distortion vector; $f(\mathbf{D}) = \|\boldsymbol{D}\|_p, \forall p \geq 1$.

Taking Action: Since our goal is to select a network code which minimizes $f(\mathbf{D}) = \|\boldsymbol{D}\|_p$, we should determine all possible instantly decodable network coding candidates. Then, we should select the best network code which minimizes $\|\boldsymbol{D}\|_p$. As we discussed in the solution of Content- and Loss-Aware IDNC-P$_1$, a trivial approach would be exhaustively listing all possible network coding candidates, and calculating $\|\boldsymbol{D}\|_p$ for each of them to determine the best one. However, constructing IDNC graph is more efficient. In Content- and Loss-Aware IDNC-P$_2$, the local IDNC graph $\mathscr{G}_t$ is constructed as follows. For device n, $|\mathscr{L}_n \cap \mathscr{H}_t|$ vertices, each shown by $v^t_{n,m}$ such that $p_m \in \mathscr{L}_n \quad \& \quad p_m \in \mathscr{H}_t$ are added to the graph. The vertex $v^t_{n,m}$ represents the missing packet p_m (with priority of $r_{m,n}$) in device n that can be transmitted from device t. The vertices in the graph are connected according to the rules C$_1^{'}$ and C$_2^{'}$ presented in the previous section. Each clique in the graph represents a network-coded packet.

Assume that the network code corresponding to clique $q_t \in \mathscr{G}_t$ is transmitted from device t. This packet is received successfully by any device n with probability of $1-\varepsilon_{t,n}$ and is lost with probability of $\varepsilon_{t,n}$. Therefore, $p-$norm of the distortion is equal to:

$$\begin{aligned}
\|\boldsymbol{D}\|_p = (&\sum_{n|(\exists p_m|v^t_{n,m}\in q_t)} (\varepsilon_{t,n}(D_n) \\
&+(1-\varepsilon_{t,n})(D_n - r_{m,n}))^p \\
&+ \sum_{n|(\nexists p_m|v^t_{n,m}\in q_t)} (D_n)^p)^{1/p} \\
&= (\sum_{n\in\mathscr{N}} (D_n)^p + \\
&\sum_{n|(\exists p_m|v^t_{n,m}\in q_t)} (D_n - r_{m,n} + \varepsilon_{t,n} r_{m,n})^p \\
&-(D_n)^p)^{1/p}. \qquad (3.15)
\end{aligned}$$

Note that the first term in the above equation, $\sum_{n\in\mathscr{N}}(D_n)^p$, is the same and fixed for all cliques in the graph. In order to minimize $\|\boldsymbol{D}\|_p$ in Eq. (3.15), the second term should be minimized. Therefore, the weight assigned to vertex $v^t_{n,m} \in \mathscr{G}_t$ is equal to:

$$w^t_{n,m} = (D_n)^p - (D_n - r_{m,n} + \varepsilon_{t,n} r_{m,n})^p. \qquad (3.16)$$

Our algorithm Content- and Loss-Aware IDNC-P$_2$ selects the clique with the maximum weight summed over its vertices in the graph $\mathscr{G} = \bigcup_{t\in\mathscr{N}} \mathscr{G}_t$. A network code corresponding to the maximum weighted clique is selected and transmitted to all devices from the corresponding transmitter.

3.4.3 Content- and Loss-Aware IDNC in Practice

Content- and Loss-Aware IDNC-P_1 and P_2 require the computation of the maximum weighted cliques in IDNC graph as explained in Sections 3.4.1 and 3.4.2. Although it is known that the maximum weighted clique problem is NP hard, the calculation may not be a bottleneck in the scenarios with small number of devices and packets as explained in Section 3.5. On the other hand, the maximum weighted clique problem can be easily solved using heuristics for larger number of devices and packets as explained next.

There exist several approaches to designing heuristics for the maximum clique problem (Pelillo, 2009). Among all these approaches, sequential greedy heuristics have the least complexity (Pelillo, 2009). The sequential greedy heuristics are based on the sequential addition of vertices to a clique based on the weights. We use the sequential greedy heuristics approach to find the maximum weighted clique in IDNC graph for Content- and Loss-Aware IDNC. Algorithm 3 summarizes this approach.

The input of Algorithm 3 is a graph $\mathscr{G}$, which consists of vertices $v_{n,m}^t$ with weights $w_{n,m}^t$. The basic idea behind Algorithm 3 is to greedily create a maximal weighted clique by adding vertices iteratively from the graph by taking into account their weights.

In particular, we first define $\widetilde{\mathscr{G}}$, which is initially set to the original graph $\mathscr{G}$. We also initialize clique $\tilde{q}_t^*$ as an empty set. (line 2) Note that $\tilde{q}_t^*$ will be updated at each step of the algorithm and eventually return the best clique in terms of the weight. Also note that $\widetilde{\mathscr{G}}$ (which is updated at each step), includes all candidate vertices, from which one is selected to be added to $\tilde{q}_t^*$ at each step. The original weight of each vertex $v_{n,m}^t$ is $w_{n,m}^t$. In the algorithm, $v_{n,m}^t$ will have an "assigned weight". The assigned weight of vertex $v_{n,m}^t \in \widetilde{\mathscr{G}}$ is calculated as follows: If the vertex has any neighbor in $\widetilde{\mathscr{G}}$, its assigned weight

Algorithm 3 Finding the best clique in terms of maximum weight in practice

```
1: Input: IDNC graph $\mathscr{G}$.
2: Initialization: $\widetilde{\mathscr{G}}$ is set to $\mathscr{G}$. Clique $\tilde{q}_t^*$ is set to an empty set.
3: while $\widetilde{\mathscr{G}}$ has vertices do
4:     for all $v_{n,m}^t \in \widetilde{\mathscr{G}}$ do
5:         Construct $\mathscr{N}_{n,m}^t$ as the set of all vertices that are connected to $v_{n,m}^t$ in graph $\widetilde{\mathscr{G}}$.
6:         if $\mathscr{N}_{n,m}^t$ is not empty then
7:             The assigned weight of $v_{n,m}^t$ is equal to $\tilde{w}_{n,m}^t = w_{n,m}^t \times \sum_{\forall\{k,l\}|(v_{k,l}^t \in \mathscr{N}_{n,m}^t)} w_{k,l}^t$.
8:         else if $\mathscr{N}_{n,m}^t$ is empty then
9:             The assigned weight of $v_{n,m}^t$ is equal to $\tilde{w}_{n,m}^t = w_{n,m}^t$.
10:    Find the vertex with the maximum assigned weight among all vertices in $\widetilde{\mathscr{G}}$; $v_{n,m}^{t*} = \arg\max_{\forall\{n,m,t\}|(v_{n,m}^t \in \widetilde{\mathscr{G}})} \tilde{w}_{n,m}^t$, and add it to the clique $\tilde{q}_t^*$.
11:    Update graph $\widetilde{\mathscr{G}}$ as the set of all vertices that are connected to $v_{n,m}^{t*}$ in graph $\widetilde{\mathscr{G}}$.
12: Return $\tilde{q}_t^*$ as the selected clique.
```

becomes equal to the multiplication of its original weight (*i.e.,* $w_{n,m}^t$) and the sum of its neighbors' original weights (line 6-7). On the other hand, if the vertex is not connected to any other vertices, its assigned weight is set to its original weight (*i.e.,* $w_{n,m}^t$ for vertex $v_{n,m}^t$) (lines 8-9). The assigned weights are calculated for all vertices in graph $\widetilde{\mathscr{G}}$ (lines 4-9). Then, the vertex with the maximum assigned weight, $v_{n,m}^{t*}$, is selected to be added to clique $\tilde{q}_t^*$ (line 10). At the end of each step, graph $\widetilde{\mathscr{G}}$ is updated. $\widetilde{\mathscr{G}}$ includes all vertices that can be added to clique $\tilde{q}_t^*$, so $\widetilde{\mathscr{G}}$ is updated as the set of the vertices that are connected to the selected vertex $v_{n,m}^{t*}$ (line 11). This procedure continues until there is no vertex left in $\widetilde{\mathscr{G}}$ (line 3).

We note that (Sorour and Valaee, 2014) and (Aboutorab and Sadeghi, 2016), which are the works that we used as baselines to evaluate our algorithms, also use heuristics inspired by the sequential greedy heuristics. Although Algorithm 3 and the heuristics in (Sorour and Valaee, 2014) and (Aboutorab and Sadeghi, 2016) share the same principles of the sequential greedy heuristics, our algorithm is an improved version of the heuristics in (Sorour and Valaee, 2014) and (Aboutorab and Sadeghi, 2016). As compared with (Aboutorab and Sadeghi, 2016), our algorithm calculates the weights of the vertices at every iteration based only on the updated graph $\widetilde{\mathscr{G}}$, which helps find the better connected cliques iteratively. In particular, updating $\widetilde{\mathscr{G}}$ at every iteration and calculating weights using only this updated graph condenses the connections/edges around the vertex $v_{n,m}^{t}$, while using $\mathscr{G}$ (or the combination of $\mathscr{G}$ and $\widetilde{\mathscr{G}}$ as in [28]) to calculate weights can possibly considers more edges in the graph than necessary, which overestimates the assigned weights of the vertices. On the other hand, (Sorour and Valaee, 2014) updates the graph $\widetilde{\mathscr{G}}$ at every iteration as in our algorithm. Yet, as compared with (Sorour and Valaee, 2014), our algorithm makes sure that vertices with large weights could also be selected for transmission even if they are not connected to other vertices (lines 8 and 9 in Algorithm 3), while the heuristic in [10] always gives preference to connected vertices, *i.e.,* network-coded packets. The approach of (Sorour and Valaee, 2014) is not appropriate in our setup as different packets have different contribution (priority) to overall quality, so it may make more sense to transmit uncoded packets (time to time) to improve the quality. The lines 8 and 9 of our algorithm in Algorithm 3 make this possible.

3.5 Simulation Results

3.5.1 Setup

We implemented the proposed Content- and Loss-Aware IDNC schemes; *i.e.,* Content- and Loss-Aware IDNC-P_1 and P_2 as well as their heuristic versions; *i.e.,* Content- and Loss-Aware IDNC-P_1 (Heuristic) and P_2 (Heuristic) by considering that there may be losses over D2D connections, and compared them with three baselines:

- Content-Aware and Loss-Unaware IDNC: This scheme is proposed in our previous work (Keshtkarjahromi et al., 2015a) assuming that D2D connections are lossless. In this method, each vertex in the local graph has a weight which is based on its contribution to minimizing the completion time for Content-Aware IDNC-P_1 and minimizing the distortion function for Content-Aware IDNC-P_2 without considering the probability of successful reception of the transmitted packet. We also implemented Content-Aware and Loss-Unaware IDNC (Heuristic) following Algorithm 3. Content- and Loss-Aware IDNC outperforms the method in (Keshtkarjahromi et al., 2015a) when D2D connections are lossy, as shown in the simulation results.
- Content-Unaware and Loss-Aware IDNC: This scheme, proposed in (Aboutorab and Sadeghi, 2016), takes into account the probability of D2D link losses among the cooperative devices, but it is not content-aware. According to (Aboutorab and Sadeghi, 2016), in Content-Unaware and Loss-Aware IDNC, the weight assignment to each vertex in the local graph is made based on the sizes of its targeted receivers' *Lacks* sets as well as the successful reception of the transmitted packet at the targeted receivers. We also developed Content-Unaware and Loss-Aware IDNC

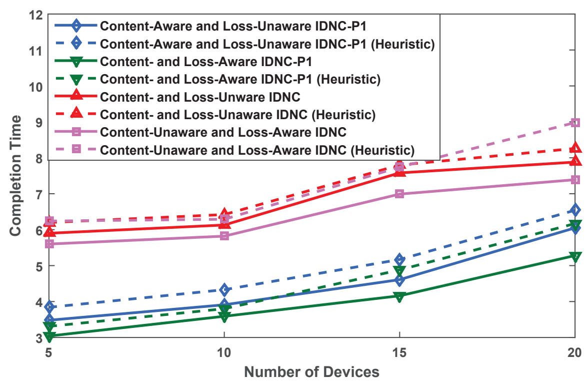
(Heuristic) based on (Aboutorab and Sadeghi, 2016). Content- and Loss-Aware IDNC outperforms the method in (Aboutorab and Sadeghi, 2016) under the realistic constraints of delay and quality, as shown in the simulation results.

- Content- and Loss-Unaware IDNC: This scheme, based on (Sorour and Valaee, 2014), does not take into account the probability of channel losses over D2D connections among the cooperative devices. In addition, it is not content-aware. We also implement Content- and Loss-Unaware (Heuristic) based on (Sorour and Valaee, 2014). Content- and Loss-Aware IDNC outperforms Content- and Loss-Unaware IDNC scheme when there are losses over D2D links and under the realistic constraints of delay and quality, as shown in the simulation results.

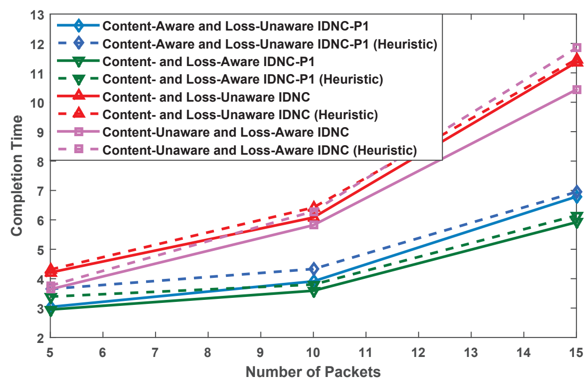
We consider a topology shown in Fig. 12 for different number of devices. First all packets are broadcast from the source in stage 1. Each device selects its loss probability uniformly from the region $[0.3, 0.8]$ for Figs. 10 and 11, and misses packets according to the selected loss probability. Then, in stage 2, the devices cooperate to recover the missing packets. The probability of loss for a packet transmission from device i to device j, $\varepsilon_{i,j}, i \in \mathscr{N}, j \in \mathscr{N}$ is selected from a uniform distribution in the region $[0, 0.3]$ for Figs. 10 and 11.

3.5.2 Simulation Results

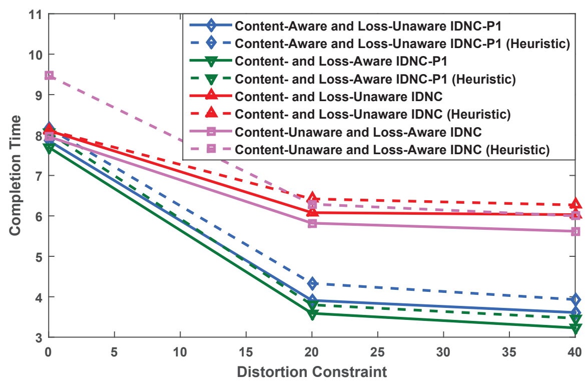
Completion Time & Distortion: Fig. 10(a) and 10(b) show the completion time required by Content- and Loss-Aware IDNC-P$_1$, Content-Aware and Loss-Unaware IDNC-P$_1$, Content-Unaware and Loss-Aware IDNC, and Content- and Loss-Unaware IDNC under the constraint of $D_n^{cons} = 0.2 \sum_{m|p_m \in \mathscr{M}} r_{m,n}$ for device n. In this setup, $r_{m,n}$ is generated according to a gamma distribution with mean 1 and variance 50. Fig. 10(a) shows the results for transmitting 10 packets to different number of devices. Fig. 10(b)



(a) Completion Time vs. Number of Devices



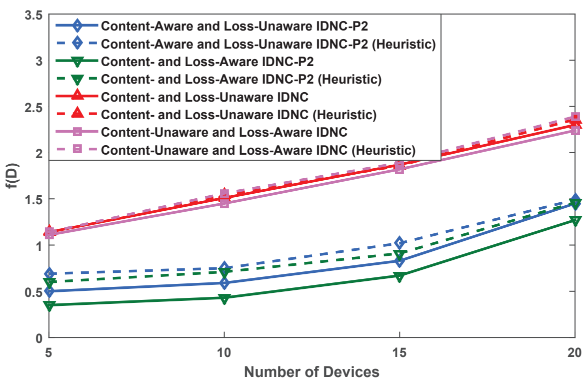
(b) Completion Time vs. Number of Packets



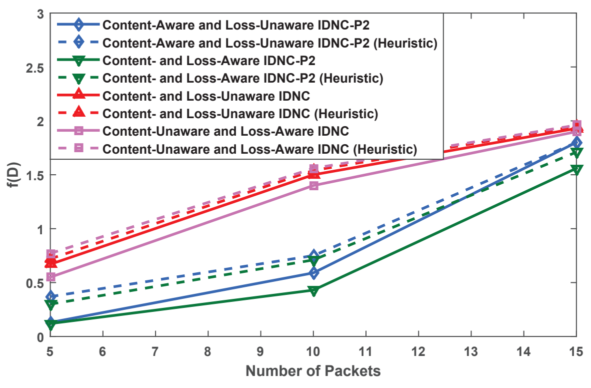
(c) Completion Time vs. Constrained Distortion

Figure 10: The performance of Content- and Loss-Aware IDNC-P$_1$, Content-Aware and Loss-Unaware IDNC-P$_1$, Content-Unaware and Loss-Aware IDNC, and Content- and Loss-Unaware IDNC.

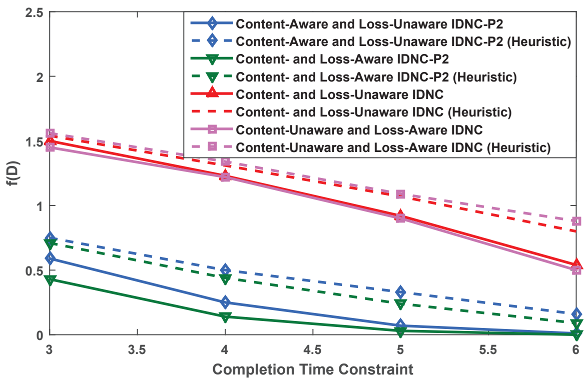
shows the results for transmitting different number of packets to 10 devices. In these graphs, the re-



(a) $f(\mathbf{D}) = \sqrt{\sum_{n \in \mathcal{N}} D_n^2}$ vs. Number of Devices



(b) $f(\mathbf{D}) = \sqrt{\sum_{n \in \mathcal{N}} D_n^2}$ vs. Number of Packets



(c) $f(\mathbf{D}) = \sqrt{\sum_{n \in \mathcal{N}} D_n^2}$ vs. Constrained Completion Time

Figure 11: The performance of Content- and Loss-Aware IDNC-P$_2$, Content-Aware and Loss-Unaware IDNC-P$_2$, Content-Unaware and Loss-Aware IDNC, and Content- and Loss-Unaware IDNC.

quired completion time increases with increasing number of devices/packets. As seen, the completion time using Content- and Loss-Aware IDNC-P$_1$, is smaller than the other methods.

Fig. 10(c) shows the required completion time for sending 10 packets to 10 devices, under the constraint of 0, 20%, and 40% distortion for each device. As expected, under the constraint of no distortion (*i.e.,* all packets are demanded by all devices), the performance of Content- and Loss-Aware IDNC-P_1 and Content-Unaware and Loss-Aware IDNC are almost the same and better than Content-Aware and Loss-Unaware IDNC-P_1 and Content- and Loss-Unaware IDNC. The more the tolerable distortion, the more improvement is observed by Content- and Loss-Aware IDNC-P_1.

Fig. 11(a) and 11(b) show the distortion function of $f(\mathbf{D}) = \sqrt{\sum_{n\in\mathcal{N}} D_n^2}$ for Content- and Loss-Aware IDNC-P_2, Content-Aware and Loss-Unaware IDNC-P_2, Content-Unaware and Loss-Aware IDNC, and Content- and Loss-Unaware IDNC under the constraint that $T^{cons} = 3$. Fig. 11(a) shows the results for transmitting 10 packets to different number of devices and Fig. 11(b) shows the results for transmitting different number of packets to 10 devices. As shown in the figures, the performance is improved significantly using Content- and Loss-Aware IDNC-P_2.

Fig. 11(c) shows $f(\mathbf{D}) = \sqrt{\sum_{n\in\mathcal{N}} D_n^2}$ for sending 10 packets to 10 devices, under the constraint of 3, 4, 5, and 6 packet transmissions. As shown in the figure, distortion resulted from Content-Aware IDNC-P_2 is small compared with the other methods. The distortion resulted from IDNC and No-NC schemes decrease and get closer to Content-Aware IDNC with increasing the constrained number of packet transmissions.

As shown in Fig. 10 and 11, Content- and Loss-Aware IDNC (Heuristic) outperforms the other heuristic methods, Content-Aware and Loss-Unaware IDNC (Heuristic), Content-Unaware and Loss-Aware IDNC (Heuristic), and Content- and Loss-Unaware IDNC (Heuristic). In addition, the performance of Content- and Loss-Aware IDNC (Heuristic), is close to the performance of Content- and Loss-

Aware IDNC, which is the optimum solution. This shows the effectiveness of the proposed heuristic in terms of reducing complexity without hurting the performance.

Real Video Traces: Table III shows the results for the total distortion improvement of Content- and Loss-Aware IDNC-P_2 over Content-Aware and Loss-Unaware IDNC-P_2, Content-Unaware and Loss-Aware IDNC and Content- and Loss-Unaware IDNC for real video traces (Akiyo and Grandma) under completion time constraint. Our video traces are CIF sequences encoded using the JM 8.6 version of the H.264/AVC codec (Rec., 2005), (h26, 2006). Each video trace is divided into blocks of packets, where block size is 10. The importance of each packet is determined by its contribution to overall video quality. The importance of each packet was determined by removing it from the video sequence, and measuring the total video quality distortion (when the packet is missing) using our H.264/AVC video codec. The video packets are delivered to 10 devices. Each device selects its loss probability in stage 1 uniformly from the region $[0.3, 0.4]$ and the loss probability for the D2D link between each two devices in stage 2 is selected uniformly from the region $[0, 0.5]$.

As seen, Content- and Loss-Aware IDNC-P_2 improves by 14.1% over Content-Aware and Loss-Unaware IDNC, 6.3% over Content-Unaware and Loss-Aware IDNC, and around 21% over Content- and Loss-Unaware IDNC, which is significant. Furthermore, the average of constrained completion time over all frames is 2.5 packet transmissions in our simulation, while the average completion time in Content-Unaware and Loss-Aware IDNC and Content- and Loss-Unaware IDNC is 8.5 and 9.5 transmissions, respectively. *I.e.,* Content-Aware IDNC improves by more than 70% over IDNC in terms of delay.

TABLE III: TOTAL DISTORTION IMPROVEMENT OF CONTENT- AND LOSS-AWARE IDNC-P_2 OVER (I) CONTENT-AWARE AND LOSS-UNAWARE IDNC-P_2, (II) CONTENT-UNAWARE AND LOSS-AWARE IDNC, AND (III) CONTENT- AND LOSS-UNAWARE IDNC

Video	(I)	(II)	(III)
Akiyo	15.5%	8.6%	24.3%
Grandma	12.8%	4.1%	17.6%

TABLE IV: TOTAL DISTORTION IMPROVEMENT OF CONTENT- AND LOSS-AWARE IDNC-P_2 (HEURISTIC) OVER (I) CONTENT-AWARE AND LOSS-UNAWARE IDNC-P_2 (HEURISTIC), (II) CONTENT-UNAWARE AND LOSS-AWARE IDNC (HEURISTIC), AND (III) CONTENT- AND LOSS-UNAWARE IDNC (HEURISTIC)

Video	(I)	(II)	(III)
Akiyo	16.1%	11.5%	25.7%
Grandma	13.5%	7.5%	18.5%

Table IV shows the results for the total distortion improvement of Content- and Loss-Aware IDNC-P_2 (Heuristic) over Content-Aware and Loss-Unaware IDNC-P_2 (Heuristic), Content-Unaware and Loss-Aware IDNC (Heuristic) and Content- and Loss-Unaware IDNC (Heuristic). As seen, Content- and Loss-Aware IDNC-P_2 (Heuristic) improves by 14.8% over Content-Aware and Loss-Unaware IDNC (Heuristic), 9.5% over Content-Unaware and Loss-Aware IDNC (Heuristic), and around 22.1% over Content- and Loss-Unaware IDNC (Heuristic).

Note that among the four methods of Content- and Loss-Aware IDNC, Content-Aware and Loss-Unaware IDNC, Content-Unaware and Loss-Aware IDNC, and Content- and Loss-Unaware IDNC, the performance of Content- and Loss-Aware IDNC is the best and the performance of Content- and Loss-Unaware IDNC is the worst. The reason is that content-awareness and loss-awareness are the two components that we consider in our method, Content- and Loss-Aware IDNC, to improve the performance of IDNC. Since Content- and Loss-Unaware IDNC does not consider either of these components, it has degraded performance. The two methods Content-Aware and Loss-Unaware IDNC and Content-Unaware and Loss-Aware IDNC considers only one of these components; Content-Aware and Loss-Unaware IDNC considers only content-awareness and Content-Unaware and Loss-Aware IDNC considers only loss-awareness. The wider the range of variation for probabilities of channel losses for D2D links in the local area, the more improvement is obtained by using loss-aware methods. On the other hand, the wider the range of variation for the importances of packets, the more improvement is obtained by using content-aware methods.

3.5.3 Complexity

Complexity of Content- and Loss-Aware IDNC: Our optimization algorithms, Content- and Loss-Aware IDNC, rely on finding optimum cliques with maximum weights to determine the best network codes. While the exact solution to the clique-finding problem is NP-complete, this is not a bottleneck in our practical system setup, due to the fact that (i) the content is divided into blocks of packets, and we run our algorithms over these blocks, and (ii) we are interested in a micro-setup with a small number of devices cooperating to exchange packets.

Complexity of Content- and Loss-Aware IDNC (Heuristic): The complexity of Algorithm 3, a heuristic approach to find maximum weighted clique at each transmission slot, is computed as follows.

The complexity of checking the connectivity of each vertex with the other vertices and updating its assigned weight is $O(MN)$, where M is the number of packets and N is the number of devices. The reason is that, each vertex can only be connected to the vertices with the same local graph $\mathscr{G}_t$, which consists of at most $M(N-1)$ vertices. The maximum number of vertices in $\mathscr{G}$ is equal to $MN(N-1)$ (because $\mathscr{G}$ is the union of all $\mathscr{G}_t, t \in \mathscr{N}$ and each $\mathscr{G}_t$ consists of at most $M(N-1)$ vertices). Therefore, the weight assignments to all vertices of the IDNC graph $\mathscr{G}$, which is required by the first step of Algorithm 3, has the complexity of the order of $O(M^2N^3)$. Then, the vertex with the maximum weight among all vertices is selected with the complexity of $O(MN(N-1))$. At the end of the first step, the graph is updated by determining the vertices that are connected to the selected vertex with the complexity of $O(MN)$. As seen, the complexity of the first step is equal to $O(M^2N^3)$.

The maximum size of the updated graph $\widetilde{\mathscr{G}}$, which is used in the second step, is $M(N-1)$. Each vertex in this graph is assigned a weight with the complexity of $O(MN)$. Therefore, the weight assignment in the second step is done with the complexity of $O(M^2N^2)$. Then, the vertex with the maximum assigned weight is selected with the complexity of $O(MN)$ and the graph is updated with the complexity of $O(MN)$. Thus, the complexity of the second step is equal to $O(M^2N^2)$. Similarly, the complexity of each step after the second step is equal to $O(M^2N^2)$.

The maximum number of required steps is N, because the maximum size of the selected clique is N. The overall complexity, for each transmission slot, is equal to the complexity of the first step summed with the complexity of all other steps, which is $O(M^2N^3)$.

3.6 Summary

In this chapter, we proposed a novel framework to improve the performance of instantly decodable network coding by exploring content-awareness. We considered a setup in which a group of mobile devices are interested in the same content, but each device has a partial content. Then, the devices cooperate by exploiting their D2D connections to receive the missing content. IDNC has been used to reduce the completion time when all devices receive the complete content. In practical applications, such as video streaming, not all packets have the same importance. In such applications, each device is interested in receiving a high quality content, instead of the complete content. We proposed Content- and Loss-Aware IDNC that delivers a high quality content to each device by taking advantage of the contributions different parts have to the content. Simulation results showed significant improvement over baselines.

CHAPTER 4

NETWORK CODING FOR COOPERATIVE MOBILE DEVICES WITH MULTIPLE INTERFACES

The contents of this chapters are based on our work that is under review in IEEE/ACM Trans. on Networking (Keshtkarjahromi et al., 2016b) and published in the proceedings of 2015 IEEE Military Communication (Milcom) conference (Keshtkarjahromi et al., 2015b)

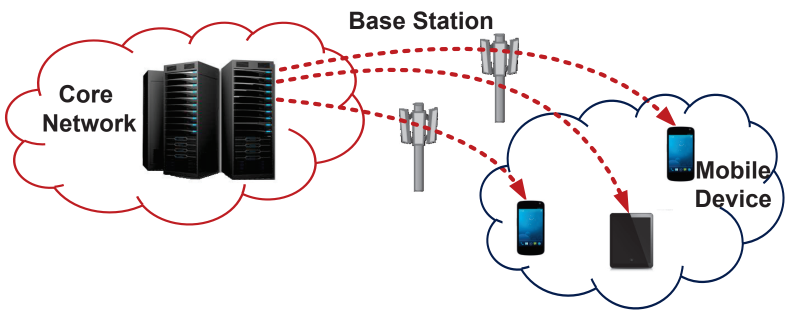
Utilizing device-to-device (D2D) connections among mobile devices is promising to meet the increasing throughput demand over cellular links. In particular, when mobile devices are in close proximity of each other and are interested in the same content, D2D connections such as Wi-Fi Direct can be opportunistically used to construct a cooperative (and jointly operating) cellular and D2D networking system. However, it is crucial to understand, quantify, and exploit the potential of network coding for cooperating mobile devices in the joint cellular and D2D setup. In this chaper, we consider this problem, and (i) develop a network coding framework, namely NCMI, for cooperative mobile devices in the joint cellular and D2D setup, and (ii) characterize the performance of the proposed network coding framework, where we use packet completion time, which is the number of transmission slots to recover all packets, as a performance metric. We demonstrate the benefits of our network coding framework through simulations.

4.1 Background

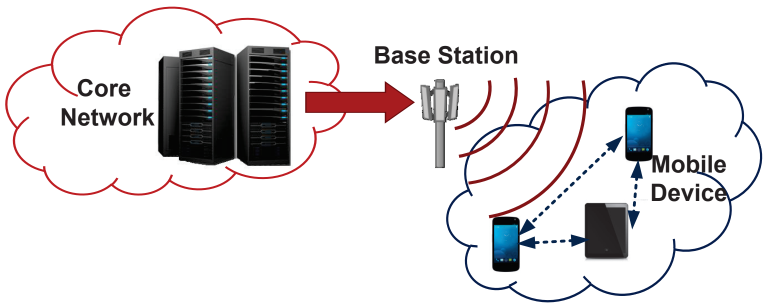
Utilizing device-to-device (D2D) connections among mobile devices simultaneously with the cellular connections is promising to meet the increasing throughput demand (Deng et al., 2014), (Nikravesh et al., 2016), (Tsao and Sivakumar, 2009), (Barua et al., 2016), (Stiemerling and Kiesel, 2009), (Khan et al., 2016), (Keller et al., 2012), (Seferoglu et al., 2011). In this work, our goal is to understand the potential of the system when D2D networking meets cellular, and develop a network coding framework to exploit this potential. We consider a scenario that a group of mobile devices are in the same transmission range and thus can hear each other. These cooperative mobile devices, that are interested in the same content, *e.g.,* video, exploit both cellular and D2D connections. In this setup, a common content is broadcast over cellular links[1], Fig. 13(a). However, mobile devices may receive only a partial content due to packet losses over cellular links, Fig. 13(b). The remaining missing content can then be recovered by utilizing both cellular and D2D connections simultaneously in a cooperative manner. In this setup, thanks to using different parts of the spectrum, cellular links and D2D connections (namely, we consider Wi-Fi Direct) operate concurrently. Thus, a mobile device can receive two packets simultaneously; one via cellular, and the other via D2D connections. The fundamental question in this context, and the focus of this work, is to design and analysis of efficient network coding algorithms that take into account (i) concurrent operation of cellular and D2D connections, and (ii) the cooperative nature of mobile devices.

The performance of network coding in cellular only and D2D only systems has been considered in previous work, (Katti et al., 2008),(Wang, 2010),(El Rouayheb et al., 2007),(El Rouayheb et al.,

[1] Note that broadcasting over cellular links is part of LTE (cis, 2010 2015), and getting increasing interest in practice, so we consider broadcast scenario instead of unicast.



(a) The default operation to connect to the Internet



(b) Using both cellular and D2D interfaces simultaneously

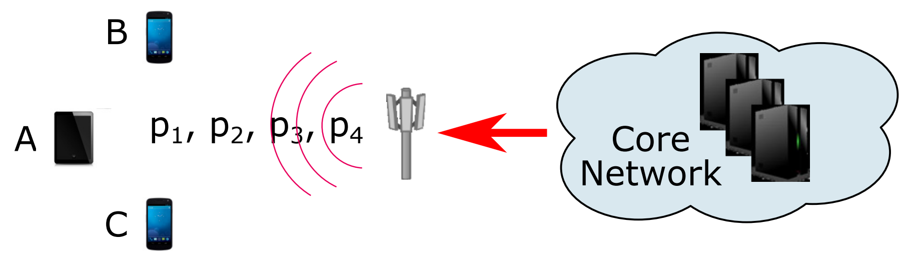
Figure 12: (a) The default operation in today's cellular systems: Each mobile device receives its data via unicast transmission over a cellular link. (b) Cooperative mobile devices with multiple interfaces: Mobile devices can cooperate and use both cellular and D2D interfaces concurrently to efficiently utilize available resources.

2010),(Sprintson et al., 2010),(A. Sprintson, 2010),(Tajbakhsh et al., 2014),(Yu et al., 2014), in the context of broadcasting a common content over cellular links, and repairing the missing content via retransmissions over cellular links, or by exploiting D2D connections. The following example demonstrates the potential of network coding in cellular and D2D only systems.

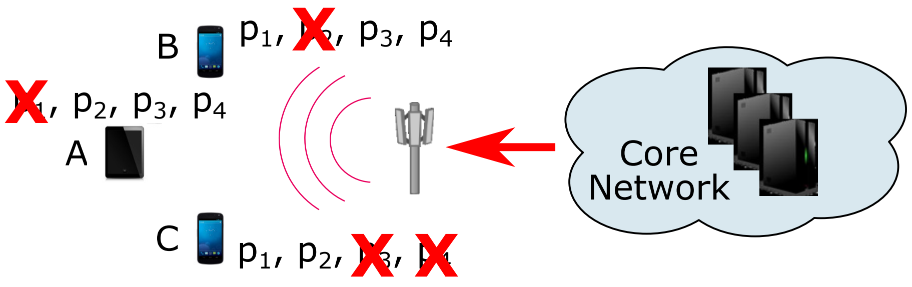
Example 4 Cellular only setup: *Let us consider Fig. 13(a), where four packets, p_1, p_2, p_3, p_4 are broadcast from the base station. Assume that after the broadcast, p_1 is missing at mobile device A, p_2 is missing at B, and p_3 and p_4 are missing at C, Fig. 13(b). The missing packets can be recovered via re-transmissions (broadcasts) in a cellular only setup (D2D connections are not used for recovery). Without network coding, four transmissions are required so that each mobile device receives all the packets. With network coding, two transmissions from the base station are sufficient: $p_1 + p_2 + p_3$ and p_4. After these two transmissions, all mobile devices have the complete set of packets. As seen, network coding reduces four transmissions to two, which shows the benefit of network coding in* cellular only *setup.*

D2D only setup: *Now let us consider packet recovery by exploiting only D2D connections (cellular connections are not used for recovery). Assume that p_1 is missing at mobile device A, p_2 is missing at B, and p_3 and p_4 are missing at C. Without network coding, four transmissions are required to recover all missing packets in all mobile devices. With network coding by exploiting D2D connections, two transmissions are sufficient: (i) mobile device B broadcasts $p_1 + p_3$, and (ii) A broadcasts $p_2 + p_4$. After these two transmissions, all mobile devices have all the packets. In this example, by taking advantage of network coding, the number of transmissions are reduced from four to two transmissions.* $\square$

The above example demonstrates the benefit of network coding in *cellular only* and *D2D only* setups. Yet, mobile devices are not limited to operate in *cellular only* and *D2D only* setups. Indeed, mobile devices can exploit multiple interfaces simultaneously including cellular and D2D connections. The following example demonstrates the potential of network coding in the *joint cellular and D2D* setup.



(a) Broadcasting four packets; p_1, p_2, p_3, p_4



(b) Missing packets after broadcast

Figure 13: Example scenario with packet losses. (a) Packets p_1, p_2, p_3, and p_4 are broadcast from the base station. (b) After the broadcast, p_1 is missing at mobile device A, p_2 is missing at B, and p_3 and p_4 are missing at C.

Example 5 Joint cellular and D2D setup: *Let us consider Fig. 13(b) again, and assume that after the broadcast, p_1 is missing at A, p_2 is missing at B, and p_3 and p_4 are missing at C. In order to recover the missing packets, we exploit both cellular and D2D links; each mobile device can receive two simultaneous packets, one of which is transmitted through the cellular links from the source and the other is transmitted through a D2D link from one of the mobile devices. For this example, the following* *transmissions are simultaneously made to recover the missing packets: (i) the base station broadcasts* $p_1 + p_3$ *via cellular links, and (ii) mobile device A broadcasts* $p_2 + p_4$ *via D2D links. As seen, the number of transmission slots is reduced to one from two as compared to Example 4.* □

As seen, mobile devices that use their cellular and D2D connections simultaneously and cooperatively have a potential of improving throughput significantly. However, it is crucial to understand and quantify the potential of network coding for cooperating mobile devices in joint cellular and D2D setup. In this work, we consider this problem, and (i) develop a network coding framework, called *network coding for multiple interfaces (i.e., jointly operating cellular and D2D interfaces) (NCMI)*, for cooperative mobile devices, and (ii) characterize the performance of the proposed network coding framework, where we use packet completion time, which is the number of transmission slots to recover all packets, as a performance metric. The following are the key contributions of this work:

- We propose a network coding algorithm; `NCMI-Batch`, where packets are network coded as a batch (of packets) to improve the throughput of cooperative mobile devices in a joint cellular and D2D setup. By taking into account the number of packets that each mobile device would like to receive for packet recovery, we develop an upper bound on the packet completion time of `NCMI-Batch`.
- For the same joint cellular and D2D setup, we develop a network coding algorithm; `NCMI-Instant`, where packets are network coded in a way that they can be decoded immediately after they are received. `NCMI-Instant` is crucial for applications with deadline constraints. Furthermore, we characterize the performance of `NCMI-Instant`, and develop an upper bound on its packet completion time.

- We develop a lower bound on the packet completion time when any network coding algorithm is employed in the joint cellular and D2D setup.
- We evaluate `NCMI-Batch` and `NCMI-Instant` for different numbers of devices and packets, and different loss probabilities. The simulation results show that our algorithms significantly improve completion time as compared to baselines, and the upper bounds we developed for `NCMI-Batch` and `NCMI-Instant` are tight.

4.2 Preliminaries & Problem Statement

We consider a setup with N cooperative mobile devices, where $\mathscr{N}$ is the set of devices in our system with $N = |\mathscr{N}|$. These devices are within close proximity of each other, so they are in the same transmission range. The mobile devices in $\mathscr{N}$ are interested in receiving packets p_m from set $\mathscr{M}$, *i.e.,* $p_m \in \mathscr{M}$ and $M = |\mathscr{M}|$.

The mobile devices in our system model are able to use cellular and D2D interfaces simultaneously to receive data. In particular, we consider a two-stage model for our joint cellular and D2D setup. In the first stage, all packets are broadcast to all devices via cellular links, while in the second stage, missing packets are recovered by utilizing cellular and D2D links jointly. This two-stage model fits well for error correction, because the cellular only operation in the first stage saves energy (as keeping multiple interfaces open increases energy consumption), and the joint cellular and D2D operation in the second stage helps quickly recover missing packets while relieving the load on cellular links. In our setup, D2D links are only needed if some packets are lost over the cellular broadcast link. Thus, at time slots when there are no packet losses, which is common scenario for practical loss rates, D2D interfaces of devices would remain idle if we keep both cellular and D2D interfaces open in the first stage. However, idle interfaces, *i.e.,* even if no packet is transmitted or received, still consume energy (G. P. Perrucci and Widmer, 2011), (Geng et al., 2015). Thus, our two-stage model is a good approach for mobile devices operating on batteries. Next, we further explain the operation of our two-stage model.

In the first stage, all packets are broadcast to all devices via cellular links. During the first stage, mobile devices may receive partial content due to packet losses over the cellular broadcast link. Thus, after the first stage, the set of packets that mobile device $n \in \mathscr{N}$ has successfully received is $\mathscr{H}_n$, and is referred to as *Has* set of device n. The set of packets that is lost in the first stage at mobile device n is referred to as *Wants* set of device n and denoted by $\mathscr{W}_n$; $\mathscr{W}_n = \mathscr{M} \setminus \mathscr{H}_n$. Furthermore, we define the set $\mathscr{M}_c$ as $\mathscr{M}_c = \bigcap_{n \in \mathscr{N}} \mathscr{W}_n$. Note that the packets in $\mathscr{M}_c$ are not received by any devices during the first stage.

In the second stage, missing packets are recovered by utilizing cellular and D2D links jointly. In particular, a mobile device may receive two recovery packets; one from cellular link and another from D2D link, simultaneously. Exploiting joint cellular and D2D links has the potential of improving throughput (Seferoglu et al., 2011). In order to use the available resources more efficiently, we need to determine the best possible network coded packets to be transmitted over cellular and D2D links at each transmission slot. This is an open problem and the focus of this chapter.

In particular, in this chapter, we develop a network coding framework for multiple interfaces (NCMI) in a joint cellular and D2D setup to recover missing packets in the second stage. Namely, we develop `NCMI-Batch` based on batch network coding and `NCMI-Instant` based on instantly decodable network coding. In `NCMI-Batch`, each transmitted packet to device $n \in \mathscr{N}$ is a linear combination of the missing packets in that device and thus it carries information about all missing packets. Therefore, all missing packets can be decoded at device n once enough number of packets are received successfully by device n. In `NCMI-Instant`, each transmitted packet to device $n \in \mathcal{N}$ carries information about only one of the missing packets at that device. Therefore, one missing packet can be decoded each time the transmitted packet is received successfully by device n.

The integral part of our work is to analyze the throughput performance of `NCMI-Batch` and `NCMI-Instant`. The amount of time required to recover the missing packets is an indicator of resource (such as energy, time, and bandwidth) consumption. Therefore, we consider the packet completion time as the performance metric, which is defined as follows:

Definition 1 *Packet completion time T is the number of transmission slots in the second stage that are required for all mobile devices to receive and decode all packets in their* Wants *sets.*

In our joint cellular and D2D setup, $\eta_n, n \in \mathcal{N}$ denotes the loss probability over the cellular link towards device n and $\varepsilon_{k,l}$ denotes the loss probability over the D2D link when device k transmits a packet to device l. We assume that η_n and $\varepsilon_{k,l}$ are i.i.d. according to a uniform distribution.

Assumptions: We assume, without loss of generality, that for each packet $p_m \in \mathcal{M}$, there is at least one mobile device that wants packet p_m. In other words, $\forall p_m \in \mathcal{M}, \exists n \in \mathcal{N}$ such that $p_m \in \mathcal{W}_n$. This assumption does not violate generality, because packets that are not wanted by any of the devices could be removed from $\mathcal{M}$.

4.3 NCMI and Upper Bounds on T

In this section, we develop network coding algorithms with multiple interfaces (NCMI) for the joint cellular and D2D setup, and provide upper bounds on their packet completion time. In particular, we develop two network coding algorithms; `NCMI-Batch`, which uses random linear network coding, and

NCMI-Instant, which provides instant decodability guarantee. We first consider the case of no loss in the second stage, while there is loss in the first stage, and analyze NCMI-Batch and NCMI-Instant. The analysis with no loss provides us insight while designing network coding algorithms for the lossy scenario in the second stage.[1]

4.3.1 No Loss in the Second stage

In this section, we assume that cellular connections in the first stage are lossy, but both the cellular and D2D connections are lossless in the second stage. Thus, all the transmitted packets in the second stage are received correctly. We develop network coding algorithms NCMI-Batch and NCMI-Instant, and develop upper bounds on the completion time.

4.3.1.1 NCMI-Batch

In this section, we explain and analyze NCMI-Batch.

4.3.1.1.1 Algorithm Description

As we mentioned earlier in Section 4.2, our system model consists of two stages. In the first stage, all packets are broadcast to all devices via cellular links without network coding. In the second stage, both cellular and D2D links are utilized simultaneously and network coding is employed. In particular, both the source and one of the devices in the local area transmit network coded packets simultaneously at each transmission slot until there is no missing packet in the local area. In NCMI-Batch the transmitted packets are formed as a linear combination of the missing packets and thus carry information about

[1]We also note that computational complexity and signaling overhead of lossless NCMIs are lower as compared to their lossy versions as we will demonstrate later in this section. Thus, if D2D links are lossless, we can directly use lossless NCMI to enjoy lower computational complexity and signaling overhead.

all packets in the set $\mathscr{M}$. Therefore, to minimize the completion time, the network coded packets to be transmitted from the source (through the cellular links) and in the local area among the mobile devices (through D2D links) are selected such that they contain as much information as possible about all missing packets. Next, we explain the details on how network coding is performed by the source and the local area devices at each transmission slot.

The source (i) determines the missing packets in all mobile devices, and (ii) transmits linear combinations of these packets (using random linear network coding over a sufficiently large field) through cellular links. These network coded packets are innovative and beneficial for any device n for which $|\mathscr{H}_n| \leq M$, because these network coded packets carry information about all missing packets in the local area. After each transmission, if the received packet is innovative for device n, it is inserted into $\mathscr{H}_n$ set. The procedure continues until each device n receives $|\mathscr{W}_n|$ innovative packets.

On the other hand, in the local area, the mobile device $n_{\max}$ with the largest *Has* set has the most information about the missing packets among all cooperating devices. Therefore, at each transmission slot, one of the devices is selected randomly as the controller; the controller selects $n_{\max} = \arg\max_{n \in \mathscr{N}} |\mathscr{H}_n|$ as the transmitter. If there are multiple of such devices, one of them is selected randomly. The transmitter linearly combines all packets in its *Has* set, $\mathscr{H}_{n_{\max}}$, and broadcasts the network coded packet to all other mobile devices via D2D links. This network coded packet is beneficial to any device $k \neq n_{max}$ (any device except for the transmitter) as long as $\mathscr{W}_k \setminus \mathscr{M}_c$ is not an empty set. The reason is that device n_{max} has the most number of packets from the set $\mathscr{M}$ among all devices. Therefore, n_{max} is the only device that has innovative information about the missing packets at all devices $k \in (\mathscr{N} \setminus n_{max})$. After each transmission, if the received packet has innovative information for device n, the received packet is inserted into the *Has* set of device n. Note that the network coded packets that include packets from $\mathscr{M}_c = \bigcap_{n\in\mathscr{N}} \mathscr{W}_n$ can only be transmitted from the source, since these packets do not exist in any of the devices. Therefore, the devices stop transmitting network coded packets through D2D links if each device n receives (i) $|\mathscr{W}_n| - |\mathscr{M}_c|$ innovative packets from D2D links, or (ii) $|\mathscr{W}_n|$ innovative packets from both the cellular and D2D links.

Next, we characterize how long it takes until all missing packets are recovered and calculate the completion time, T.

4.3.1.1.2 Upper Bound on T

In order to characterize the performance of proposed `NCMI-Batch`, we consider the worst case scenario and develop an upper bound on the completion time of `NCMI-Batch`. We note that `NCMI-Batch` is guaranteed to outperform the upper bound, as proved here and shown in the simulation results.

Theorem 1 *Packet completion time; T when* `NCMI-Batch` *is employed by cooperative mobile devices on a joint cellular and D2D setup is upper bounded by*

$$T \leq \left\lceil \max\left(|\mathscr{M}_c|, \frac{1}{3}\big(\max_{n\in\mathscr{N}} |\mathscr{W}_n| + \min_{n\in\mathscr{N}} |\mathscr{W}_n|\big), \frac{1}{2}\max_{n\in\mathscr{N}} |\mathscr{W}_n| \right) \right\rceil. \tag{4.1}$$

Proof: In `NCMI-Batch`, at each transmission slot, a linear combination of all missing packets, that is innovative for all mobile devices, is broadcast from the source to all devices. Meanwhile, in the local area, the device with the largest *Has* set is selected as the transmitter; the transmitter broadcasts a linear combination of the packets in its *Has* set, that is innovative for all other devices. Therefore, the size of *Has* set for all devices except for the device with the maximum size of *Has* set (the transmitter) is increased by two and the size of *Has* set for the transmitter is increased by one. In the first transmission slot, $x = \arg\max_{n \in \mathscr{N}} |\mathscr{H}_n|$ is selected as the transmitter so the size of $\mathscr{H}_n, n \in (\mathscr{N} \setminus x)$ is increased by two and the the size of $\mathscr{H}_x$ is increased by one. For the next transmission slots, x remains the transmitter until the size of *Has* set for one of the other devices is equal to the size of the *Has* set for x. It takes at most $|\mathscr{H}_x| - |\mathscr{H}_n|$ transmission slots for device n to have the same size of *Has* set as device x and thus to be selected as the transmitter. Therefore, by considering $r = \arg\min_{n \in (\mathscr{N} \setminus x)} |\mathscr{H}_n| = \arg\max_{n \in (\mathscr{N} \setminus x)} |\mathscr{W}_n|$, it takes at most $|\mathscr{H}_t| - |\mathscr{H}_r|$ transmission slots for the transmitter to be reselected. On the other hand, it takes at most $M - |\mathscr{H}_x|$ transmission slots that the size of $\mathscr{H}_x$ becomes equal to M. We consider two cases:

1. $(M - |\mathscr{H}_x|) \leq |\mathscr{H}_x| - |\mathscr{H}_r|$:

   After at most $M - |\mathscr{H}_x|$ transmission slots, the size of $\mathscr{H}_x$ becomes equal to M and the size of $\mathscr{H}_r$ is still less than the size of $\mathscr{H}_x$. Therefore, in the rest of the transmission slots, x remains as the transmitter and thus, the completion time is upper bounded by the required number of transmission slots to satisfy device r, which is equal to $\frac{1}{2} \max_{n \in \mathscr{N}} |\mathscr{W}_n|$.

2. $(M - |\mathscr{H}_x|) \geq |\mathscr{H}_x| - |\mathscr{H}_r|$:

   After at most $|\mathscr{H}_x| - |\mathscr{H}_r|$ transmission slots, the size of $\mathscr{H}_x$ becomes equal to $\mathscr{H}_r$. Therefore, in the rest of the transmission slots, r and x are selected as the transmitter, alternatively; *i.e.,* in one of the transmission slots x is selected as the transmitter and in the consecutive transmission slot, r is selected as the transmitter and thus in every two consecutive transmission slots, the size of $\mathscr{H}_r$ is increased by three. The completion time is upper bounded by the required number of transmission slots to satisfy device r, which is equal to $\frac{1}{3}(|\mathscr{W}_r| + |\mathscr{W}_x|)$.

In addition, the number of transmission slots cannot be less than $|\mathscr{M}_c|$. By considering this fact and the results from cases (i) and (ii), the upper bound in Theorem 1 is obtained. This concludes the proof. ■

Example 6 *Let us consider three mobile devices with the* Wants *sets;* $\mathscr{W}_A = \{p_1, p_2, p_3\}$, $\mathscr{W}_B = \{p_1, p_4, p_5\}$, $\mathscr{W}_C = \{p_1, p_6, p_7\}$. *Using* `NCMI-Batch`*, in the first transmission slot, the source transmits a linear combination of the packets* $p_1, p_2, ..., p_7$*, which is innovative for all devices A, B, and C, as it carries information about all missing packets. Meanwhile, in the local area, the device with the largest* Has *set is selected. Since there is equality in this example (*$|\mathscr{H}_A| = |\mathscr{H}_B| = |\mathscr{H}_C| = 4$*), one device is selected randomly, let us say device A. Device A transmits a linear combinations of* p_4, p_5, p_6, p_7 *via D2D links. Note that this network coded packet is beneficial to both devices B and C, as it carries information about their missing packets. Therefore, at the end of the first transmission slot, device A receives one innovative packet and thus the size of its* Has *set is increased by one and devices B and C receive two innovative packets and thus the sizes of the* Has *sets for these devices are increased by two;* i.e., $|\mathscr{H}_A| = 5$, $|\mathscr{H}_B| = |\mathscr{H}_C| = 6$. *In the second transmission slot, the source transmits a linear combination of* $p_1, p_2, ..., p_7$*, which is innovative for all devices A, B, and C and at the same time B or C (with the larger size of* Has *set than A) transmits a linear combination of the packets in its* Has *set, which is innovative for A. Therefore, at the end of the second transmission slot, device A receives two innovative packets and thus the size of its* Has *set is increased by two and devices B and C receive one innovative packet (they only need one innovative packet to be satisfied) and thus the sizes of the*

Has *sets for these devices are increased by one;* i.e., $|\mathscr{H}_A| = 7$, $|\mathscr{H}_B| = |\mathscr{H}_C| = 7$. *As seen, each device receives three innovative packets after two transmission slots and thus the completion time is equal to* $T = 2$. *On the other hand, from Theorem 1, the upper bound for the completion time is equal to* $\left\lceil \max\left(1, \frac{1}{3}(3+3), \frac{3}{2}\right)\right\rceil = 2$. *As seen, the inequality* $T \leq 2$ *(Eq. 4.1) is satisfied, in this example. It can also be seen that the upper bound is tight, in this example.*

4.3.1.1.3 Computational Complexity

In `NCMI-Batch`, at each transmission slot, the source creates a linear combination of all packets with the complexity of $O(M)$. Meanwhile, among all devices, the one with the maximum size of *Has* set is selected as the transmitter with the complexity of $O(N)$. Then, the transmitter creates a linear combination of the packets in its *Has* with the complexity of $O(M)$. Therefore, the computational complexity at each transmission slot is $O(M+N+M) = O(M+N)$. As the maximum number of transmission slots is equal to the number of missing packet, M, the complexity of `NCMI-Batch` (when the channels are lossless in the second stage), is polynomial with the complexity of $O(M(M+N)) = O(M^2+N)$.

4.3.1.2 `NCMI-Instant`

In this section, we develop and analyze `NCMI-Instant`, where packets are network coded in a way that they can be decoded immediately after they are received by the mobile devices. `NCMI-Instant` is crucial for applications with deadline constraints.

4.3.1.2.1 Algorithm Description

`NCMI-Instant` determines the IDNC packets to be transmitted at each transmission slot through the cellular and D2D links with the goal of minimizing the completion time. To reach this goal, the optimum way is exhaustively creating all combinations of IDNC packets that can be transmitted from the source and the mobile devices in all transmission slots and selecting the sequence of packets that results in the minimum average completion time. However, the complexity of exhaustive search is high and thus in this chapter, we proposed a heuristic method `NCMI-Instant` with linear complexity with respect to the number of devices and quadratic complexity with respect to the number of packets. `NCMI-Instant` consists of three steps: (i) creating IDNC packets (Algorithm 1), (ii) grouping the created IDNC packets into the sets $\mathscr{M}_c$, $\mathscr{M}_l$ and $\mathscr{M}_d$ (Algorithm 1), and (iii) determining the IDNC packets to be transmitted from the two interfaces based on the the created groups. In the following, we explain these steps.

Step 1: Creating IDNC packets: Algorithm 1 is a greedy algorithm that creates IDNC packets, sequentially, from the packets in the set of missing packets in all devices. The main idea behind this algorithm is to check uncoded packets sequentially and try to merge them into IDNC packets. We note that similar ideas have been considered in network coding literature, *e.g.,* a greedy algorithm for creating network coding packets over wireless mesh networks is developed using a similar approach in (Katti et al., 2008). According to Algorithm 1, the first IDNC packet is created with the first uncoded packet. Then for each remaining uncoded packets, the algorithm checks all of the previously created IDNC packets; if the uncoded packet can be merged with the IDNC packet (*i.e.,* the merged IDNC packet is instantly decodable for all devices that want one of the uncoded packets in the IDNC packet), the IDNC packet would be updated by adding the uncoded packet. Otherwise, a new IDNC packet containing the uncoded packet is created. This algorithm creates IDNC packets with complexity of $O(M^2+N)$.

Algorithm 4 Grouping the packets in the *Wants* Sets

```
1: for any packet $p_m$ in $\mathscr{M}$ do
2:     Define vector $v_m$ with size $N$. Each element of the vector $v_m$ is initially set to $NULL$; i.e., $v_m[n] = NULL, \forall n \in \mathscr{N}$.
3:     for any device $n$ in $\mathscr{N}$ do
4:         if $p_m$ is wanted by device $n$ then
5:             $v_m[n] = p_m$ and $p_m$ is removed from the Wants set $\mathscr{W}_n$.
6:     if there exists a vector $v_{m'}$, $m' < m$ satisfying for any $n$, for which $v_m[n] = p_m$ then $v_{m'}[n] = NULL$ then
7:         Replace $v_{m'}$ with $v_{m'} + v_m$ and delete $v_m$. (Note that $p_m + NULL = p_m$ for any $m$)
8: Each element of $\mathscr{M}_c$ is constructed by network coding all packets in a vector $v_m$ if all elements of $v_m$ are the same and not equal to $NULL$; i.e., $v_m[1] \neq NULL$ and $v_m[n] = v_m[1], \forall n \in \mathscr{N}$.
9: Each element of $\mathscr{M}_d$ is constructed by network coding all packets in a vector $v_m$ if $v_m$ contains at least one element equal to $NULL$; i.e., $\exists x \in \mathscr{N} \mid v_m[x] = NULL$.
10: Construct $\mathscr{M}_l$ using the remaining vectors. In other words, each element of $\mathscr{M}_l$ is constructed by network coding all packets in a vector $v_m$ if $v_m$ does not contain any $NULL$ element and contains at least two different elements; i.e., $v_m[n] \neq NULL, \forall n \in \mathscr{N}$ and $\exists n, x \mid v_m[n] \neq v_m[x]$.
```

Note that the complexity of exhaustive search to create all possible IDNC packets is NP-hard. Next, we describe the details of Algorithm 1 on Step 1, creating IDNC packets.

In Algorithm 4, we define vectors with length N, whose elements are either $NULL$ or $p_m \in \mathscr{M}$. A network coded packet is associated to each vector and constructed as a linear combination of all elements of the vector that are not equal to $NULL$. We describe how each vector is defined by Algorithm 4 in the following. First, the vector v_m with the length of N is defined for each packet $p_m \in \mathscr{M}$. The nth element of this vector, where n is any device that wants that packet, is initially set to p_m. The value of this vector for the remaining elements, which correspond to the devices that have that packet, is initially set to $NULL$ (lines 1-5). A network coded packet is instantly decodable for device n if it contains information about one and only one of the packets in the *Wants* set of device n. Let us consider packet p_m with its corresponding vector v_m and the instantly decodable network coded packet corresponding to vector $v_{m'}$. These two packets can be combined and merged as a new instantly decodable network coding packet (line 7) if the value of vector $v_{m'}$ for all devices that want packet p_m (*i.e.,* any device n for which $v_m[n]$ is p_m) is $NULL$ (the condition of *if* statement at line 6).

Step 2: Grouping the created IDNC packets: After creating the IDNC packets, we group them into the sets, $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$, based on their differences from the view point of cellular and D2D links. The packets in $\mathscr{M}_c$ can only be transmitted via cellular links. The packets in $\mathscr{M}_l$ can be transmitted via cellular or D2D links, but it may take more time slots if they are transmitted via D2D links. The packets in $\mathscr{M}_d$ can be transmitted either via cellular or D2D links, and the number of time slots for transmitting a packet is the same in both cellular and D2D. The sets $\mathscr{M}_c$, $\mathscr{M}_l$ and $\mathscr{M}_d$ are created as the following. $\mathscr{M}_c$ is the set of packets that are not available in the local area and should only be transmitted from the base station (source). These are the packets that are wanted by all devices. $\mathscr{M}_d$ is the set of IDNC packets that can be transmitted by a single transmission from either the source or a mobile device. These are the packets that can be formed by combining a subset of the packets in the *Has* set of one of the devices (note that any combination of packets can be created in the source as it has all packets). $\mathscr{M}_l$ is the set of remaining IDNC packets that can be transmitted by a single transmission from the source or by two transmissions (according to Lemma 2) in the local area. These are the packets that contain one and only one uncoded packet from the *Wants* set of each device and thus they are not available in one single device. Therefore, in order to send these packets through D2D links, they need to be divided into two IDNC packets and each part can be transmitted from the device that has all the corresponding uncoded packets.

Lemma 2 *Each network coded packet in $\mathscr{M}_l$ can be transmitted by exactly two transmission slots using D2D links by splitting the packet into two (network coded) packets.*

Proof: Each network coded packet in $\mathscr{M}_l$ contains one and only one packet from each $\mathscr{W}_n, \forall n \in \mathscr{N}$ set. Therefore, there is no single device among the cooperating devices that has all uncoded packets of the network coded packet, and thus one transmission slot is not sufficient to transmit the network coded packet in the local area. On the other hand, since there are at least two different uncoded packets in the network coded packet, we can split the network coded packet into two (network coded) packets; each device wants one uncoded packet from one of the splitted packets and has all the uncoded packets in the other splitted packet. Therefore, each of the two splitted packets can be transmitted from at least one device among the cooperating devices and thus two transmissions are necessary and sufficient to transmit the content of the network coded packet via D2D links. ■

The properties of the sets $\mathscr{M}_c, \mathscr{M}_l$, and $\mathscr{M}_d$ are summarized in Table V.

Next, we describe the details of Algorithm 1 on Step 2, grouping the created IDNC packets.

TABLE V: THE REQUIRED NUMBER OF TRANSMISSIONS TO TRANSMIT EACH PACKET IN $\mathscr{M}_c, \mathscr{M}_l$, AND $\mathscr{M}_d$ VIA CELLULAR AND D2D LINKS.

Set	Cellular Link	D2D Link
$\mathscr{M}_c$	1	N/A
$\mathscr{M}_l$	1	2
$\mathscr{M}_d$	1	1

The set $\mathscr{M}_c$ consists of all packets that are wanted by all devices; *i.e.,* by using any vector v_m whose elements are the same and not equal to $NULL$ (line 8). The set $\mathscr{M}_d$ consists of the network coded packets that can be formed by combining a subset of the packets in the *Has* set of one of the devices, called x; *i.e.,* by using the packets in the vector v_m whose xth element is $NULL$ (line 9). The set $\mathscr{M}_l$ consists of the rest of network coded packets. In other words, each network coded packet in $\mathscr{M}_l$ contains one and only one packet from the *Wants* set of each device; *i.e.,* by using any vector v_m whose elements are not equal to $NULL$ and it contains at least two different elements (line 10).

Next, we give an example on how Algorithm 4 works.

Example 7 *Let us assume that there are three mobile devices with the* Wants *sets;* $\mathscr{W}_A = \{p_1, p_4\}$, $\mathscr{W}_B = \{p_1, p_2, p_3\}$, $\mathscr{W}_C = \{p_1, p_4, p_5\}$. *Note that* $\mathscr{M} = \bigcup_{n \in \mathscr{N}} \mathscr{W}_n = \{p_1, p_2, p_3, p_4, p_5\}$ *and* $\mathscr{H}_n = \mathscr{M} \setminus \mathscr{W}_n$ *for* $n \in \{A, B, C\}$; $\mathscr{H}_A = \{p_2, p_3, p_5\}$, $\mathscr{H}_B = \{p_4, p_5\}$, $\mathscr{H}_C = \{p_2, p_3\}$. *According to Algorithm 4,* v_1 *is equal to* $[p_1, p_1, p_1]$, *because it is wanted by all three devices, as shown in Table VI. Then, for packet* p_2

TABLE VI: GROUPING THE PACKETS IN THE *WANTS* SETS.

Set		$\mathscr{W}_A$	$\mathscr{W}_B$	$\mathscr{W}_C$
$\mathscr{M}_c$	$v_1:$	p_1	p_1	p_1
$\mathscr{M}_l$	$v_2:$	p_4	p_2	p_4
$\mathscr{M}_d$	$v_3:$	$NULL$	p_3	p_5

we define vector $v_2 = [NULL, p_2, NULL]$, because p_2 is wanted by device B only. Similarly, for packet p_3, we define vector $v_3 = [NULL, p_3, NULL]$. v_3 can not be merged with v_1 or v_2, because the second element of v_3 is p_3, while the second elements of v_1 and v_2 are not NULL. In the next step, we define vector $v_4 = [p_4, NULL, p_4]$. v_4 can be merged with v_2, because the first and third elements of v_4 is p_4 and the first and the second elements of v_2 is NULL. Therefore, v_2 is updated as $v_2 + v_4 = [p_4, p_2, p_4]$ and v_4 is deleted, as shown in Table VI. Similarly, v_5 is defined as $[NULL, NULL, p_5]$. v_5 can be merged with v_3, because the third element of v_5 is p_5 and the third element of v_3 is NULL. Therefore, v_3 is updated as $v_3 + v_5 = [NULL, p_3, p_5]$ and v_5 is deleted, as shown in Table VI.

All elements of v_1 are the same and not equal to NULL. Therefore, $\mathscr{M}_c$ is constructed from v_1; $\mathscr{M}_c = \{p_1\}$, as shown in Table VI. v_3 has a NULL element. Therefore, $\mathscr{M}_d$ is constructed from v_3; $\mathscr{M}_d = \{p_3 + p_5\}$. v_2 does not have any NULL element and its first and second elements are different. Therefore, $\mathscr{M}_l$ is constructed from vector v_2; $\mathscr{M}_l = \{p_2 + p_4\}$.

$\square$

Step 3: Determining the IDNC packets to be transmitted from the two interfaces: At each transmission slot, two packets are selected from the sets $\mathscr{M}_c \cup \mathscr{M}_l \cup \mathscr{M}_d$; one to be transmitted from the base station and another one to be transmitted from one of the mobile devices. The idea behind packet selection in `NCMI-Instant` is that the selected packet can be received by possibly largest number of devices successfully as it would deliver more information in one transmission and thus would eventually reduce the completion time. Note that the decisions of which network coded packet is transmitted and which device is selected as the transmitter in the local area, are made by the controller (which can be selected randomly among mobile devices). **Packet Selection by the Source:** The packets in $\mathscr{M}_c$ should only be transmitted from the source. On the other hand, a packet transmission from the set $\mathscr{M}_c$ targets all devices. Therefore, the source selects a packet from $\mathscr{M}_c$ to transmit, if this set is not empty. Among the sets $\mathscr{M}_l$ and $\mathscr{M}_d$, the packets in $\mathscr{M}_l$ targets all devices, while the packets in $\mathscr{M}_d$ targets a subset of devices. On the other hand, packets in $\mathscr{M}_l$ require less number of transmissions if transmitted from the source than the mobile devices. Therefore, the source selects a packet from $\mathscr{M}_l$ to transmit if $\mathscr{M}_c$ is empty. At last, if both $\mathscr{M}_c$ and $\mathscr{M}_l$ are empty, a packet in $\mathscr{M}_d$, is selected to be transmitted from the source. **Packet Selection by the Mobile Devices:** Any packet from the set $\mathscr{M}_d$ can be transmitted from the local area by a single transmission, however only a partial of a packet from the set $\mathscr{M}_l$ can be transmitted by a single transmission from the local area. Therefore, the order of transmitting the packets in the local area is (i) $\mathscr{M}_d$ and (ii) $\mathscr{M}_l$.

4.3.1.2.2 Upper Bound on T

Now, we analyze the completion time performance of `NCMI-Instant` by developing an upper bound on the completion time.

Theorem 3 *Packet completion time when* `NCMI-Instant` *is employed by cooperative mobile devices on a joint cellular and D2D setup is upper bounded by:*

$$T \leq \lceil \min\Big(\max\big(\frac{M}{2}, |\mathscr{M}_c|\big), \max\big(|\mathscr{M}_c|, \frac{1}{3}\big(2\min_{n\in\mathscr{N}}|\mathscr{W}_n| + |\mathscr{M}_d|\big), \frac{1}{2}\big(\min_{n\in\mathscr{N}}|\mathscr{W}_n| + |\mathscr{M}_d|\big)\big)\Big)\rceil.$$

Proof: We consider three conditions based on the relative sizes of the sets $\mathscr{M}_c$, $\mathscr{M}_l$ and $\mathscr{M}_d$ and then calculate the maximum completion time obtained from each of the conditions.

1. $|\mathscr{M}_c| \geq (|\mathscr{M}_d| + 2|\mathscr{M}_l|)$

   Under this condition, the base station starts transmitting the packets in $\mathscr{M}_c$; meanwhile, in the local area a network coded packet in $\mathscr{M}_d$ and $\mathscr{M}_l$ with the order of (i) $\mathscr{M}_d$ and (ii) $\mathscr{M}_l$ is selected to be transmitted from one of the mobile devices. After $|\mathscr{M}_d| + 2|\mathscr{M}_l|$ transmission slots, all the packets in $\mathscr{M}_d$ and $\mathscr{M}_l$ are transmitted by the cooperating devices and $|\mathscr{M}_c| - (|\mathscr{M}_d| + 2|\mathscr{M}_l|)$ packets are left from $\mathscr{M}_c$; it takes $|\mathscr{M}_c| - (|\mathscr{M}_d| + 2|\mathscr{M}_l|)$ transmission slots for the base station to transmit these remaining packets. By summing the required number of transmission slots, the completion time under condition (1) is equal to:

$$T_{(1)} = |\mathscr{M}_c| \tag{4.2}$$

Example 8 *Let us consider three mobile devices with the* Wants *sets of* $\mathscr{W}_A = \{p_1, p_2, p_3, p_4, p_5\}$, $\mathscr{W}_B = \{p_1, p_2, p_3, p_4, p_6, p_7\}$, $\mathscr{W}_C = \{p_1, p_2, p_3, p_4, p_6, p_8\}$*. By using Algorithm 4,* $\mathscr{M}_c = \{p_1, p_2, p_3, p_4\}$, $\mathscr{M}_l = \{p_5 + p_6\}$, $\mathscr{M}_d = \{p_7 + p_8\}$*. For this example, condition (1) is met;* $|\mathscr{M}_c| = 4 > 3 = (|\mathscr{M}_d| + 2|\mathscr{M}_l|)$*. Accordingly,* 4 *transmission slots are required; in the first transmission slot,*

p_1 is transmitted from the base station and at the same time $p_7 + p_8$ is transmitted from device A. In the second transmission slot, p_2 is transmitted from the base station and p_6 is transmitted from device A. In the third transmission slot, p_3 is transmitted from the base station and p_5 is transmitted from device B (or device C). In the forth transmission slot, p_4 is transmitted from the the base station. $\square$

2. $(|\mathscr{M}_d| + 2|\mathscr{M}_l|) \geq |\mathscr{M}_c| \geq (|\mathscr{M}_d| - |\mathscr{M}_l|)$

   For this condition, we consider two cases of (i) $|\mathscr{M}_c| \leq |\mathscr{M}_d|$ and (ii) $|\mathscr{M}_c| \geq |\mathscr{M}_d|$.

   In case (i), the base station starts transmitting the packets in $\mathscr{M}_c$; meanwhile in the local area the packets in $\mathscr{M}_d$ are transmitted from x (the device that has all packets in $\mathscr{M}_d$). Since $|\mathscr{M}_c| \leq |\mathscr{M}_d|$, after $|\mathscr{M}_c|$ transmission slots, all the packets in $\mathscr{M}_c$ have been transmitted by the base station and $|\mathscr{M}_d| - |\mathscr{M}_c|$ packets are left from $\mathscr{M}_d$. According to condition (2), $|\mathscr{M}_c|$ is greater than $(|\mathscr{M}_d| - |\mathscr{M}_l|)$ and thus $(|\mathscr{M}_d| - |\mathscr{M}_c|)$ is smaller than $|\mathscr{M}_l|$. Therefore, in the next $|\mathscr{M}_d| - |\mathscr{M}_c|$ transmission slots, the remaining packets in $\mathscr{M}_d$ is transmitted by the cooperative devices in the local area and the base station transmits the network coded packets from $\mathscr{M}_l$. At last, $|\mathscr{M}_l| - (|\mathscr{M}_d| - |\mathscr{M}_c|)$ packets are left from $\mathscr{M}_l$; it takes $2/3(|\mathscr{M}_l| - (|\mathscr{M}_d| - |\mathscr{M}_c|))$ transmission slots by the source and the cooperating devices, jointly to transmit these remaining packets. By summing the required number of transmission slots, the completion time for case (i) is equal to $\frac{1}{3}(2|\mathscr{M}_l| + 2|\mathscr{M}_c| + |\mathscr{M}_d|)$.

   In case (ii), the base station starts transmitting the packets in $\mathscr{M}_c$; meanwhile in the local area the packets in $\mathscr{M}_d$ are transmitted from x (the device that has all packets in $\mathscr{M}_d$). Since $|\mathscr{M}_c| \geq |\mathscr{M}_d|$, after $|\mathscr{M}_d|$ transmission slots, all the packets in $\mathscr{M}_d$ have been transmitted by the cooperating devices in the local area and $|\mathscr{M}_c|-|\mathscr{M}_d|$ packets are left from $\mathscr{M}_c$. According to condition (2), $|\mathscr{M}_c|$ is smaller than $(|\mathscr{M}_d|+2|\mathscr{M}_l|)$ and thus $(|\mathscr{M}_c|-|\mathscr{M}_l|)$ is smaller than $2|\mathscr{M}_l|$. Therefore, in the next $|\mathscr{M}_c|-|\mathscr{M}_d|$ transmission slots, the base station transmits the remaining packets in $\mathscr{M}_c$ and the cooperating devices transmit $\frac{|\mathscr{M}_c|-|\mathscr{M}_d|}{2}$ packets from $\mathscr{M}_l$. At last, $|\mathscr{M}_l|-\frac{|\mathscr{M}_c|-|\mathscr{M}_d|}{2}$ packets are left from $|\mathscr{M}_l|$; it takes $\frac{2}{3}(|\mathscr{M}_l|-\frac{|\mathscr{M}_c|-|\mathscr{M}_d|}{2})$ transmission slots by the source and the cooperating devices, jointly to transmit these remaining packets. By summing the required number of transmission slots, the completion time for case (ii) is equal to $\frac{1}{3}(2|\mathscr{M}_l|+2|\mathscr{M}_c|+|\mathscr{M}_d|)$.

In addition, any packet in $\mathscr{M}_c$ contains one packet from the *Wants* set of each device by definition. On the other hand, any network coded packet in $\mathscr{M}_l$ contains one and only one packet from the *Wants* set of each device. Therefore, the inequality $|\mathscr{W}_n| \geq (|\mathscr{M}_c|+|\mathscr{M}_l|)$ holds for each device $n \in \mathscr{N}$ including the device with the minimum size of *Wants* set. Therefore, the following inequality, holds:

$$|\mathscr{M}_l| \leq (\min_{n\in\mathscr{N}} |\mathscr{W}_n| - |\mathscr{M}_c|). \tag{4.3}$$

By using the above discussion, the maximum completion time under condition (2) is upper bounded by:

$$\begin{aligned} T_{(2)} &= \frac{1}{3}(2|\mathscr{M}_l| + 2|\mathscr{M}_c| + |\mathscr{M}_d|) \\ &\leq \frac{1}{3}(2 \min_{n \in \mathscr{N}} |\mathscr{W}_n| + |\mathscr{M}_d|). \end{aligned} \tag{4.4}$$

Example 9 *Let us consider three mobile devices with the* Wants *sets of* $\mathscr{W}_A = \{p_1, p_2, p_5, p_8\}$, $\mathscr{W}_B = \{p_1, p_3, p_6, p_9, p_{11}\}$, $\mathscr{W}_C = \{p_1, p_4, p_7, p_{10}, p_{11}\}$. *By using Algorithm 4,* $\mathscr{M}_c = \{p_1\}, \mathscr{M}_l = \{p_2 + p_3 + p_4, p_5 + p_6 + p_7, p_8 + p_9 + p_{10}\}, \mathscr{M}_d = \{p_{11}\}$. *For this example, condition (2) is met;* $(|\mathscr{M}_d| - |\mathscr{M}_l|) < |\mathscr{M}_c| < (|\mathscr{M}_d| + 2|\mathscr{M}_l|)$. *Accordingly,* 3 *transmission slots are required; in the first transmission slot,* p_1 *is transmitted from the base station and at the same time* p_{11} *is transmitted from device A. In the second transmission slot,* $p_2 + p_3 + p_4$ *is transmitted from the base station and* $p_9 + p_{10}$ *is transmitted from device A. In the third transmission slot,* $p_5 + p_6 + p_7$ *is transmitted from the base station and* p_8 *is transmitted from device B.* □

3. $|\mathscr{M}_c| \leq (|\mathscr{M}_d| - |\mathscr{M}_l|)$

Under this condition, the base station starts transmitting the packets in $\mathscr{M}_c$; meanwhile in the local area the packets in $\mathscr{M}_d$ are transmitted from x (the device that has all packets in $\mathscr{M}_d$). After $|\mathscr{M}_c|$ transmission slots, all packets in $\mathscr{M}_c$ have been transmitted by the base station and $|\mathscr{M}_d| - |\mathscr{M}_c|$ packets are left from $\mathscr{M}_d$. In the next $|\mathscr{M}_l|$ transmission slots, the base station transmits all packets in $\mathscr{M}_l$ and the cooperating devices transmit $|\mathscr{M}_l|$ packets from $\mathscr{M}_d$. At last, $|\mathscr{M}_d| - |\mathscr{M}_c| - |\mathscr{M}_l|$ packets are left from $\mathscr{M}_d$; it takes $\frac{|\mathscr{M}_d| - |\mathscr{M}_c| - |\mathscr{M}_l|}{2}$ transmission slots by the source and the cooperating devices, jointly to transmit these remaining packets. By summing the required number of transmission slots and from 4.3, the maximum completion time under condition (3) is upper bounded by:

$$
\begin{aligned}
T_{(3)} &= \frac{|\mathscr{M}_d| + |\mathscr{M}_c| + |\mathscr{M}_l|}{2} \\
&\leq \frac{|\mathscr{M}_d| + \min_{n \in \mathscr{N}} |\mathscr{W}_n|}{2}
\end{aligned}
\tag{4.5}
$$

Example 10 *Let us consider three mobile devices with the* Wants *sets of* $\mathscr{W}_A = \{p_1, p_2\}$, $\mathscr{W}_B = \{p_1, p_3, p_5, p_6, p_9\}$, $\mathscr{W}_C = \{p_1, p_4, p_5, p_7, p_8, p_{10}\}$. *By using Algorithm 4,* $\mathscr{M}_c = \{p_1\}$, $\mathscr{M}_l = \{p_2 + p_3 + p_4\}$, $\mathscr{M}_d = \{p_5, p_6 + p_7, p_8 + p_9, p_{10}\}$. *For this example, condition (3) is met;* $|\mathscr{M}_c| < (|\mathscr{M}_d| - |\mathscr{M}_l|)$. *Accordingly,* 3 *transmission slots are required; in the first transmission slot,* p_1 *is transmitted from the base station and at the same time* p_{10} *is transmitted from device A. In the second transmission slot,* $p_2 + p_3 + p_4$ *is transmitted from the base station and* $p_8 + p_9$ *is transmitted from device A. In the third transmission slot,* p_5 *is transmitted from the base station and* $p_6 + p_7$ *is transmitted from device A.* □

By combining the completion time obtained from conditions (1), (2), and (3), the upper bound of $T_{upper,1} = \lceil \max(|\mathscr{M}_c|, \frac{1}{3}(2\min_{n \in \mathscr{N}} |\mathscr{W}_n| + |\mathscr{M}_d|), \frac{1}{2}(\min_{n \in \mathscr{N}} |\mathscr{W}_n| + |\mathscr{M}_d|)) \rceil$ is achieved.

In addition, according to Algorithm 4, first vector $v_m[n]$ is defined for each packet $p_m \in \mathscr{M}$. Then, different vectors may be combined as a network coded packet that can be transmitted from the source or the cooperating devices. In the worst case scenario, the vectors can not be combined as network coded packets. Therefore, we have at most $M = |\mathscr{M}|$ vectors, each representing one of the packets from set $\mathscr{M}$. In this case, each vector can be transmitted by a single transmission from the source or by a single transmission from the cooperating devices (if the packet represented by the vector is available in one of the devices). Under this worst case scenario, the upper bound of $T_{upper,2} = \lceil \max(|\mathscr{M}_c|, \frac{M}{2}) \rceil$ is achieved.

By combining $T_{upper,1}$ and $T_{upper,2}$, the completion time in Theorem 3 is obtained. This concludes the proof. ■

Example 11 *Let us assume that there are three mobile devices with the* Wants *sets;* $\mathscr{W}_A = \{p_1, p_2, p_4, p_7, p_9\}$, $\mathscr{W}_B = \{p_1, p_2, p_5, p_7, p_{10}\}$, $\mathscr{W}_C = \{p_1, p_3, p_6, p_8\}$. *Note that* $\mathscr{M} = \bigcup_{n \in \mathscr{N}} \mathscr{W}_n = \{p_1, \ldots, p_{10}\}$ *and* $\mathscr{H}_n = \mathscr{M} \setminus \mathscr{W}_n$ *for* $n \in \{A, B, C\}$; $\mathscr{H}_A = \{p_3, p_5, p_6, p_8, p_{10}\}$, $\mathscr{H}_B = \{p_3, p_4, p_6, p_8, p_9\}$, $\mathscr{H}_C = \{p_2, p_4, p_5, p_7, p_9, p_{10}\}$. *According to Algorithm 4,* $\mathscr{M}_c = \{p_1\}$, $\mathscr{M}_d = \{p_9 + p_{10}\}$, *and* $\mathscr{M}_l = \{p_2 + p_3, p_4 + p_5 + p_6, p_7 + p_8\}$ *in this example. Using* `NCMI-Instant`*, in the first transmission slot packet* p_1 *(in set* $\mathscr{M}_c$*) is transmitted from the source and packet* $p_9 + p_{10}$ *(in set* $\mathscr{M}_d$*) is transmitted from device C. In the second transmission slot, packet* $p_2 + p_3$ *(in set* $\mathscr{M}_l$*) is transmitted from the source and packet* p_7 *(the first splitted packet of the third network coded packet in set* $\mathscr{M}_l$ *available at device C) is transmitted from device C. In the third transmission slot, packet* $p_4 + p_5 + p_6$ *(in set* $\mathscr{M}_l$*) is transmitted from source and packet* p_8 *(the second splitted packet of the third network coded packet in set* $\mathscr{M}_l$ *available at device A and B) is transmitted from device A or B. Therefore, in total three transmission slots are required by* `NCMI-Instant`*;* $T = 3$. *On the other hand, from Theorem 3, the upper bound for the completion time is equal to* $\lceil \min\big(5, \max(1, 3, 2.5)\big) \rceil = 3$. *As seen, the inequality* $T \leq 3$ *(Eq. 4.2) is satisfied, in this example. It can also be seen that the upper bound is tight, in this example.*

□

4.3.1.2.3 Computational Complexity

Algorithm 4 constructs vectors by checking all packets and devices with complexity of $O(MN)$. Each constructed vector is checked whether it can be merged with the other vectors; since the maximum number of vectors is M, the complexity of merging the vectors is $O(M^2)$. In `NCMI-Instant`, at each transmission slot, the source creates a linear combination of all packets in one of the vectors with the complexity of $O(M)$. Meanwhile, in the local area, one vector is chosen, the device that has all the uncoded packets of the vector in its *Has* set is selected as the transmitter with the complexity of $O(N)$, and a linear combination of the packets corresponding to the vector is created with the complexity of $O(M)$ to be transmitted from the transmitter device. Therefore, the complexity of creating and transmitting the packets is $O(M+N+M) = O(M+N)$ for one transmission slot and thus $O(M^2+MN) = O(M^2+N)$ (with at most M transmission slots) for all transmission slots. This complexity is added to the complexity of Algorithm 4, $O(M^2+N)$. Thus, the complexity of `NCMI-Instant` when the channels are lossless in the second stage is polynomial with $O(M^2+N)$.

4.3.2 Lossy Links in the Second Stage

In the previous section, we developed the network coding algorithms `NCMI-Batch` and `NCMI-Instant` for the case that packets in the second stage are received successfully in the receiver devices without any loss (Fig. 14(a)). In this section, we consider a more realistic scenario, where the channels are lossy in the second stage (Fig. 14(b)).

We assume that each packet transmitted from the source to each device $n \in \mathcal{N}$ is lost with probability of η_n. Similarly, each packet transmitted from the transmitter device k to the receiver device l is lost with probability of $\varepsilon_{k,l}$. We also assume that probability of all cellular channel losses, $\eta_n, \forall n \in \mathcal{N}$, and D2D channel losses, $\varepsilon_{k,l}, \forall k,l \in \mathcal{N}$, are i.i.d. and uniformly distributed.

The completion time for the lossy links in the second stage, depends on the packets that are received successfully in addition to the transmitted packets at each transmission slot. Therefore, for each transmitted packet, we define *targeted receivers*, $\mathcal{N}_r$, as the set of devices for which the transmitted packet is innovative. We also define *successful receivers* as the set of devices for which the transmitted packet is innovative and is received successfully. We consider the average number of successful receivers for each transmitted packet, which is calculated as follows.

Average number of successful receivers, for a packet transmitted from the source to the set of targeted receivers $\mathcal{N}_r$, is equal to the average number of packets that are received successfully at the devices in the set $\mathcal{N}_r$ and calculated as $\sum_{n\in\mathcal{N}_r}(1-\eta_n)$. Similarly, average number of successful receivers for a packet transmitted from the transmitter device t to the set of targeted receivers $\mathcal{N}_r$ is equal to $\sum_{n\in\mathcal{N}_r}(1-\varepsilon_{t,n})$.

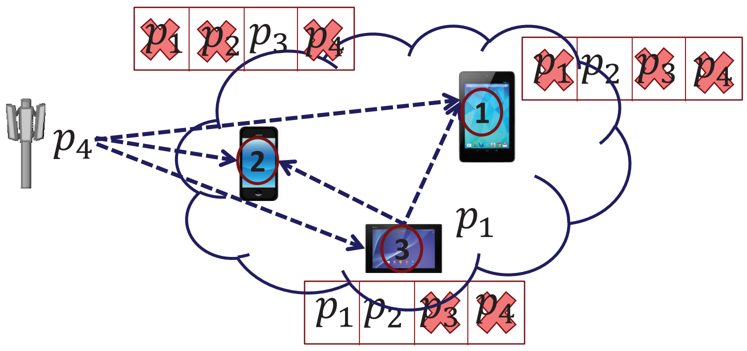
Note that, in the case of no loss, the average number of successful receivers with the set of targeted receivers, $\mathcal{N}_r$, is equal to the size of this set; $|\mathcal{N}_r|$.

4.3.2.1 NCMI-Batch

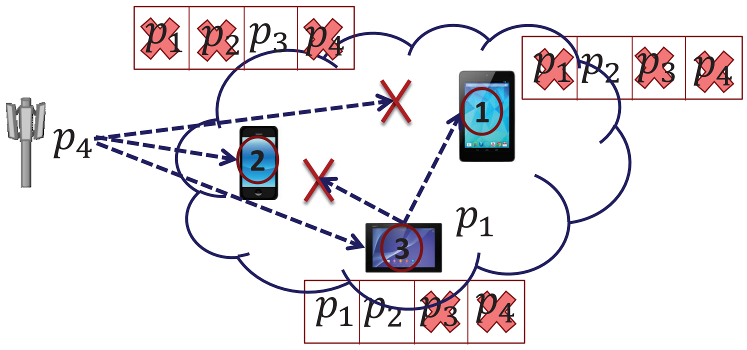
In this section, we describe and analyze NCMI-Batch.

4.3.2.1.1 Algorithm Description

As described in Section 4.3.1.1, for the case of lossless channels in the second stage, network coding decisions by NCMI-Batch are made such that the transmitted packets contain as much information as possible about missing packets in all devices. For the case of lossy channels, the transmitted packets are


$\cancel{p_1}$ $\cancel{p_2}$ p_3 $\cancel{p_4}$
$\cancel{p_1}$ p_2 $\cancel{p_3}$ $\cancel{p_4}$
p_4
1
2
3
p_1
p_1 p_2 $\cancel{p_3}$ $\cancel{p_4}$

(a) No loss in the second stage


$\cancel{p_1}$ $\cancel{p_2}$ p_3 $\cancel{p_4}$
$\cancel{p_1}$ p_2 $\cancel{p_3}$ $\cancel{p_4}$
p_4
1
2
3
p_1
p_1 p_2 $\cancel{p_3}$ $\cancel{p_4}$

(b) Loss in the second stage

Figure 14: To recover the missing packets in the second stage, p_1 is selected to be transmitted from device 3 and p_4 is selected to be transmitted from the source. (a) Lossless links in the second stage: All packets are received successfully by the receivers. (b) Lossy links in the second stage: The packet that is transmitted from device 3 is lost in device 2 and received successfully in device 1. Packet transmitted from the source is lost in device 1 and received successfully in devices 2 and 3.

Algorithm 5 `NCMI-Batch` for Lossy Channels

```
1: $\mathscr{M}$: the set of the missing packets in all devices.
   **Packet Selection by the Source:**
2: A network coded packet is generated as a linear combination of all packets in $\mathscr{M}$.
3: The source broadcasts this packet to all mobile devices.
   **Packet Selection by the Mobile Devices:**
4: for any device $n$ in $\mathscr{N}$ do
5:     $\mathscr{N}_r = \{k \mid (\exists p \in \mathscr{H}_n \mid p \perp\!\!\!\perp \mathscr{H}_k)\}$.
6:     $Ave_n = \sum_{k \in \mathscr{N}_r}(1 - \varepsilon_{n,k})$.
7: $x = \arg\max_{n \in \mathscr{N}} Ave_n$.
8: A network coded packet is generated as a linear combination of all packets in $\mathscr{H}_x$.
9: $x$ broadcasts this packet to all other mobile devices..
```

selected such that the *successfully delivered* packets contain as much information as possible about the missing packets in all devices. In other words, the transmitted packets are selected such that the average number of successful receivers are maximized. Packet selection algorithm by `NCMI-Batch` at each transmission slot for the lossy channels is provided in Algorithm 5. According to this algorithm, two packets are selected; one to be transmitted from the source and the other one to be transmitted in the local area. The details of packet selection in Algorithm 5 are described next.

Packet Selection by the Source (lines 1-2): The average number of successful receivers, for a transmitted packet from the source with the set of targeted receivers, $\mathscr{N}_r$, is equal to $\sum_{n \in \mathscr{N}_r}(1 - \eta_n)$.

On the other hand, any device n with $|\mathscr{W}_n| > 0$ or equivalently $|\mathscr{H}_n| < M$, is interested in receiving an innovative packet and can be a member of $\mathscr{N}_r$ for the transmitted packet from the source. Therefore, in order to maximize the average number of successful receivers, we need to enlarge the set $\mathscr{N}_r$ to all devices that need an innovative packet; *i.e.,* $\mathscr{N}_r = \{n \mid (|\mathscr{H}_n| < M)\}$. According to Algorithm 5, in `NCMI-Batch`, the source (i) determines the missing packets in all mobile devices, and (ii) transmits linear combinations of these packets (using random linear network coding over a sufficiently large field) as the network coded packet through cellular links. This network coded packet carry information about all missing packets in the local area, and thus is innovative and beneficial for any device n for which $|\mathscr{H}_n| < M$; *i.e.,* $\mathscr{N}_r = \{n \mid (|\mathscr{H}_n| < M)\}$. Therefore, the packet selected to be transmitted from the source in `NCMI-Batch` maximizes the average number of successful receivers. After each transmission, if the received packet is innovative for device n, it is inserted into $\mathscr{H}_n$ set. The procedure continues until each device n receives $|\mathscr{W}_n|$ innovative packets.

Packet Selection by the Mobile Devices (lines 3-7): In `NCMI-Batch`, the controller (which can be randomly selected among mobile devices) selects one of the devices as the transmitter (according to line 7). Then, the transmitter transmits a linear combination of the packets in its *Has* set to all other devices through D2D links. The set of targeted receivers, $\mathscr{N}_r$, for a packet transmitted from device n, is the set of devices, for which a random linear combination of the packets in $\mathscr{H}_n$ is innovative. In other words, if there is at least one packet from $\mathscr{H}_n$ that is linearly independent of all packets in $\mathscr{H}_k$, then k is a member of $\mathscr{N}_r$; $\mathscr{N}_r = \{k \mid (\exists p \in \mathscr{H}_n \mid p \perp\!\!\!\perp \mathscr{H}_k)\}$. Accordingly, the average number of successful receivers for a packet transmitted from n is equal to $\sum_{k \in \mathscr{N}_r} (1 - \varepsilon_{n,k})$. In `NCMI-Batch`, the mobile device x with the largest average number of successful receivers is selected as the transmitter among all devices at each transmission slot; $x = \arg\max_{n\in\mathcal{N}} \sum_{k\in\mathcal{N}_r}(1-\varepsilon_{n,k})$. If there are multiple of such devices, one of them is selected randomly. The transmitter linearly combines all packets in its *Has* set, $\mathcal{H}_x$, and broadcasts the network coded packet to all other mobile devices. This network coded packet is beneficial to all devices in the set of targeted receivers (by the transmitter) that receive the packet successfully.

After each transmission, if the successfully received packet has innovative information for device k, it is inserted into the *Has* set of device k. Similar to the case of lossless channels, the cooperating devices stop transmitting network coded packets if each device n successfully receives (i) $|\mathcal{W}_n| - |\mathcal{M}_c|$ innovative packets from the cooperating devices, or (ii) $|\mathcal{W}_n|$ innovative packets from both the source and cooperating devices. Note that selecting the device with the largest average number of successful receivers for the case of no loss, $\varepsilon_{n,k} = 0, \forall n,k \in \mathcal{N}$, is equivalent to selecting the device with the largest number of targeted receivers and thus the device with the largest *Has* set. This is aligned with our strategy, presented in Section 4.3.1.1, for packet selection in `NCMI-Batch` when the channels are lossless in the second stage.

Next, we give an example on `NCMI-Batch` for lossy channels.

Example 12 *Let us consider three mobile devices with the* Wants *sets;* $\mathcal{W}_A = \{p_1, p_2, p_3\}, \mathcal{W}_B = \{p_1, p_4, p_5\}, \mathcal{W}_C = \{p_1, p_6, p_7\}$*, the* Has *sets* $\mathcal{H}_A = \{p_4, p_5, p_6, p_7\}, \mathcal{H}_B = \{p_2, p_3, p_6, p_7\}, \mathcal{H}_C = \{p_2, p_3, p_4, p_5\}$ *and probabilities of channel losses;* $\eta_A = 0.35, \eta_B = 0.4, \eta_C = 0.45, \varepsilon_{A,B} = 0.1, \varepsilon_{A,C} = 0.3, \varepsilon_{B,A} = 0.1, \varepsilon_{B,C} = 0.2, \varepsilon_{C,A} = 0.3, \varepsilon_{C,B} = 0.2$. *By using* `NCMI-Batch`*,* $p_1, \ldots, p_7$ *are combined as a network coded packet in the source and transmitted to all devices in the first slot. This transmitted packet can be beneficial to all devices (*$\mathcal{N}_r = \{A, B, C\}$*) as it carries information about all missing packets. Note that the average number of successful receivers for this transmitted packet is equal to* $(1-0.35)+(1-0.4)+$

$(1-0.45) = 1.8$. *Meanwhile, in the local area, the device with the largest average number of receivers is selected as the transmitter. The set of targeted receivers for a random linear network coded packet transmitted from A is* $\{B,C\}$, *as* $p_4 \in \mathscr{H}_A$ *(or* $p_5 \in \mathscr{H}_A$*) is linearly independent of* $\mathscr{H}_B = \{p_2, p_3, p_6, p_7\}$ *and* $p_6 \in \mathscr{H}_A$ *(or* $p_7 \in \mathscr{H}_A$*) is linearly independent of* $\mathscr{H}_C = \{p_2, p_3, p_4, p_5\}$ *and thus the average number of successful receivers is equal to* $(1-\varepsilon_{A,B}) + (1-\varepsilon_{A,C}) = 1.6$. *Similarly, the average number of successful receivers for a random linear network coded packet transmitted from B and C is equal to* 1.7 *and* 1.5, *respectively. Therefore, device B with the maximum average number of successful receivers is selected as the transmitter and transmits a random linear combination of* p_2, p_3, p_6, p_7. *Note that this network coded packet can be received successfully at devices A and C with probabilities of* 0.9 *and* 0.8, *respectively. Thus, in the first slot, the source transmits linear combination of* $p_1, \ldots, p_7$ *and device B transmits a random linear combination of* p_2, p_3, p_6, p_7, *simultaneously. The procedure is repeated at every slot until each device receives* 3 *innovative packets.* □

Next, we characterize how long it takes until all missing packets are recovered; *i.e.,* the completion time; T.

4.3.2.1.2 Upper Bound on T

In order to characterize the performance of proposed `NCMI-Batch`, we provide an upper bound on the completion time obtained from `NCMI-Batch` in the following theorem.

Theorem 4 *The average of completion time; T when* `NCMI-Batch` *is employed by cooperative mobile devices on a joint cellular and D2D setup when the channel links are lossy is upper bounded by*

$$T \leq \left\lceil \max\left(\frac{|\mathscr{M}_c|}{1 - \prod_{n \in \mathscr{N}} \eta_n}, T_j \right) \right\rceil, \tag{4.6}$$

where,

$$T_j = \begin{cases} \max(T_x, T_r), & \text{if } \frac{|\mathscr{W}_r| - |\mathscr{W}_x|}{1-\eta_r-\varepsilon_{x,r}+\eta_x} \leq \frac{|\mathscr{W}_x|}{1-\eta_x} \\ \frac{|\mathscr{W}_r|}{2-\eta_r-\varepsilon_{x,r}}, & \text{otherwise,} \end{cases} \tag{4.7}$$

$$T_x = \frac{|\mathscr{W}_r|(1-\varepsilon_{r,x}) + |\mathscr{W}_x|(1+2\eta_x - 2\eta_r + \varepsilon_{r,x} - 2\varepsilon_{x,r})}{(1-\eta_r-\varepsilon_{x,r}+\eta_x)(3-2\eta_x-\varepsilon_{r,x})}, \tag{4.8}$$

$$T_r = \frac{|\mathscr{W}_r|(1-2\eta_r-\varepsilon_{x,r}+2\eta_x) + |\mathscr{W}_x|(1-\varepsilon_{x,r})}{(1-\eta_r-\varepsilon_{x,r}+\eta_x)(3-2\eta_r-\varepsilon_{x,r})}, \tag{4.9}$$

$$x = \arg\max_{n\in\mathscr{N}} |\mathscr{H}_n| = \arg\min_{n\in\mathscr{N}} |\mathscr{W}_n|, \tag{4.10}$$

$$r = \arg\max_{n\in(\mathscr{N}\backslash x)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}}, \tag{4.11}$$

Proof: In `NCMI-Batch` the packets are selected such that the delivered packets carry the most information about the missing packets in the local area. Therefore, the packets with the maximum average number of successful receivers are selected to be transmitted from the source and in the local area. Any other packet selection method with less average number of successful receivers delivers less information about the missing packets to devices and thus requires larger completion time. To find an upper bound on `NCMI-Batch`, we consider one of such methods, where the packet selection at each transmission slot is as follows: (i) the source transmits a random linear combination of the missing packets in all devices and (ii) in the local area, the device with the maximum size of *Has* set is selected as the transmitter and transmits a linear combination of the packets in its *Has* set. If there are multiple of such devices, they are selected alternatively as the transmitter for the rest of the transmission slots until the end of the completion time. As seen, the packet selection in the source for this method is the same as `NCMI-Batch`. But the packet selection in the local area differs from `NCMI-Batch`; in `NCMI-Batch` the packet with the maximum average number of successful receivers is selected to be transmitted, while in the mentioned method, there is no guarantee that the selected packet is associated with the maximum average number of successful receivers. Therefore, the completion time obtained from this mentioned method is an upper bound on the completion time for `NCMI-Batch`. In the following, we consider the worst case scenario for the considered method and calculate the average completion time for this scenario. The calculated completion time is an upper bound on the completion time achieved by `NCMI-Batch`.

At the beginning, device $x = \arg\max_{n \in \mathscr{N}} |\mathscr{H}_n|$ is selected as the transmitter in the local area. Therefore, any other device $n \in (\mathscr{N} \setminus x)$ receives two transmissions; one from the source with the probability of channel loss of η_n and another from x with the probability of channel loss of $\varepsilon_{x,n}$. Device x receives only one transmission which is transmitted from the source with the probability of channel loss of η_x. Therefore, in average, the size of $\mathscr{W}_x$ is reduced by $(1-\eta_x)$ and the size of $\mathscr{W}_n, n \in (\mathscr{N} \setminus x)$ is reduced by $(1-\eta_n+1-\varepsilon_{x,n}) = (2-\eta_n-\varepsilon_{x,n})$ at each transmission slot until one of the devices other than x is selected as the transmitter. According to the method, a device is selected as the transmitter if it has the largest *Has* set (or smallest *Wants* set) among all devices. Therefore,

x remains the transmitter until the size of *Wants* set for one of the other devices is equal to the size of *Wants* set for x. After the average of k transmission slot, the size of $\mathscr{W}_n, n \in (\mathscr{N} \setminus x)$ is equal to $|\mathscr{W}_n| - k(2 - \eta_n - \varepsilon_{x,n})$ and the size of $\mathscr{W}_x$ is equal to $|\mathscr{W}_x| - k(1 - \eta_x)$. Therefore, by considering device r as $\arg\max_{n \in (\mathscr{N} \setminus x)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}}$, it takes at most $k = \frac{|\mathscr{W}_r|-|\mathscr{W}_x|}{(2-\eta_r-\varepsilon_{x,r})-(1-\eta_x)}$ transmission slots (k is obtained by solving the equation $|\mathscr{W}_x| - k(1 - \eta_x) = |\mathscr{W}_r| - k(2 - \eta_r - \varepsilon_{x,r})$) for another device (which is device r) to be selected as the transmitter in average. On the other hand, it takes at most $\frac{|\mathscr{W}_x|}{1-\eta_x}$ transmission slots for device x to receive all required packets and the size of its *Has* set becomes equal to M. We consider two cases :

1. $k \geq \frac{|\mathscr{W}_x|}{1-\eta_x}$

   After at most $\frac{|\mathscr{W}_x|}{1-\eta_x}$ transmission slots, the size of *Has* set for device x is equal to M and the size of *Has* set for device r is less than M. Thus x remains the transmitter for the next transmission slots until the end of completion time. Therefore, the completion time is at most equal to the number of transmission slots required by r to be satisfied; $T \leq \lceil \frac{|\mathscr{W}_r|}{2-\eta_r-\varepsilon_{x,r}} \rceil$.

2. $k \leq \frac{|\mathscr{W}_x|}{1-\eta_x}$

   After at most k transmission slots, the size of *Has* set for device r is equal to the size of *Has* set for device x. Therefore, according to the method, these devices are selected as the transmitter alternatively for the rest of the transmission slots until the end of the completion time. In this way, the sizes of $\mathscr{W}_r$ and $\mathscr{W}_x$ are reduced by $(1 - \eta_r) + (2 - \eta_r - \varepsilon_{x,r})$ and $(1 - \eta_x) + (2 - \eta_x - \varepsilon_{r,x})$ in every two consecutive transmission slots after the first k transmission slots, respectively and thus it takes at most $T_r = k + 2\frac{|\mathscr{W}_r|-k(2-\eta_r-\varepsilon_{x,r})}{3-2\eta_r-\varepsilon_{x,r}}$ and $T_x = k + 2\frac{|\mathscr{W}_x|-k(1-\eta_x)}{3-2\eta_x-\varepsilon_{r,x}}$ transmission slots for devices

r and x, respectively to be satisfied. By replacing $k = \frac{|\mathscr{W}_r| - |\mathscr{W}_x|}{(2-\eta_r-\varepsilon_{x,r})-(1-\eta_x)}$ in the obtained expression for T_x and T_r, equations 4.8 and 4.9 is obtained. The total required number of transmission slots is equal to maximum of T_x and T_r.

In addition, each packet in $\mathscr{M}_c$ can only be transmitted from the source until at least one of the devices can receive it successfully. Each packet that is transmitted from the source is received successfully by at least one of the devices with probability of $1 - \prod_{n \in \mathscr{N}} \eta_n$. Therefore, it takes $\frac{1}{1-\prod_{n \in \mathscr{N}} \eta_n}$ transmissions for each packet in $\mathscr{M}_c$ to be received successfully by at least one of the devices and thus the number of transmission slots cannot be less than $\lceil \frac{|\mathscr{M}_c|}{1-\prod_{n \in \mathscr{N}} \eta_n} \rceil$. By considering this fact and the results from cases (1) and (2), the upper bound in Theorem 1 is obtained. This concludes the proof. ■

4.3.2.1.3 Computational Complexity

In `NCMI-Batch` the source creates a linear combination of all packets with the complexity of $O(M)$ at each transmission slot. Meanwhile, in the local area, for each device the average number of successful receivers is calculated with the complexity of $O(N-1)$. The complexity of calculating the average number of successful receivers for all devices is $O(N^2)$. Then, the device with the maximum average number of successful receivers is selected as the transmitter with the complexity of $O(N)$ and a linear combination of the packets in its *Has* set is created with the complexity of $O(M)$ at each transmission slot. By considering the maximum of M transmission slots, the complexity of `NCMI-Batch` is polynomial with the complexity of $O(M^2 + N^2)$. This computational complexity, by also taking additional steps such as dividing a file into smaller sets of M packets, makes `NCMI-Batch` applicable for practical deployment.

4.3.2.2 `NCMI-Instant`

In section 4.3.1.2, we described and analyzed `NCMI-Instant` for the case that the channel links are lossless. In this section, we consider a more generalized case where the channel links are lossy and present `NCMI-Instant` for this generalized case.

4.3.2.2.1 Algorithm Description

`NCMI-Instant` algorithm, described in Section 4.3.1.2.1, groups the packets that are wanted by the devices into sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ when there is no loss in the second stage. Then, packets from these sets are network coded and transmitted from the cellular and D2D links. Yet, when the links are lossy in the second stage, this approach should be revised for the following reasons.

First, each network coded packet is transmitted successfully when there is no loss. This makes creating fixed $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ sets possible and reasonable before transmitting the packets in the second stage. However, when the channels are lossy in the second stage, the transmitted network coded packet at each transmission slot may be received successfully by some of the targeted receivers (known as successful receivers) and may be lost by the others. Thus, fixed $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ sets are not appropriate in this scenario, and they should be updated after each transmission depending on successful packet transmissions.

Also, when the channel links are lossy, different network coded packets even if they are in the same set (such as in $\mathscr{M}_l$) have different priorities for transmission. In particular, the packets that can be received by possibly larger number of devices successfully should be prioritized as it would deliver more information in one transmission, which would eventually reduce the completion time.

Algorithm 6 `NCMI-Instant` for Lossy Channels

```
Group the packets in the Wants sets of all devices into the sets $\mathscr{M}_c$, $\mathscr{M}_l$ and $\mathscr{M}_d$ using Algorithm 4
and keep all network coded packets, $p \in (\mathscr{M}_c \cup \mathscr{M}_l \cup \mathscr{M}_d)$, along with their corresponding vectors,
$v_p$, in these sets.
Packet Selection by the Source:
if $\mathscr{M}_c$ is not empty then
  The first element of $\mathscr{M}_c$ is selected to be transmitted from the source.
else if $\mathscr{M}_l$ is not empty then
  The first element of $\mathscr{M}_l$ is selected to be transmitted from the source.
else if $\mathscr{M}_d$ is not empty then
  The packet with the maximum average number of successful receivers among all packets in $\mathscr{M}_d$,
  which is equal to $\arg\max_{p \in \mathscr{M}_d} \sum_{n|(v_p[n] \neq NULL)} (1 - \eta_n)$, is selected to be transmitted from the
  source.
Packet Selection by the Mobile Devices:
Consider packet $p$ with its corresponding vector $v_p$ in $\mathscr{M}_l$; any device in $x \in \mathscr{N}$ can transmit
partial of packet $p$ which is constructed by network coding all packets in the vector $v_p \setminus v_p[x]$. The
average number of successful receivers for this packet is equal to $\sum_{n|(v_p[n] \neq v_p[x])} (1 - \varepsilon_{x,n})$. Determine
all possible transmitted packets with their corresponding transmitters and calculate their average
number of successful receivers.
Consider packet $p$ with its corresponding vector $v_p$ in $\mathscr{M}_d$; any device that has all uncoded packets in
$v_p$ can transmit $p$. The average number of successful receivers for packet $p \in \mathscr{M}_d$ to be transmitted
```

from device $x \in \{i \mid v_p[i] = NULL\}$ is equal to $\sum_{n|(v_p[n] \neq NULL)}(1-\varepsilon_{x,n})$. Determine all possible transmitted packets with their corresponding transmitters and calculate their average number of successful receivers.

From all packets determined in 8 and 9, select the packet with the maximum average number of successful receivers to be transmitted from its corresponding transmitter through D2D links.

By taking into account these two points, we updated `NCMI-Instant` for lossy channels. The new algorithm is provided in Algorithm 6. In this algorithm, we first determine the sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ using Algorithm 4 at each slot. Then two packets are selected from these sets; one to be transmitted from the source and the other one to be transmitted by a mobile device using D2D connections. The idea behind packet selection is that the selected packet can be received by possibly largest number of devices successfully as it would deliver more information in one transmission and thus would eventually reduce the completion time. The details of packet selection in Algorithm 6 are described below.

Packet Selection by the Source (lines 2-7): The average number of successful receivers for any packet in the set $\mathscr{M}_c \cup \mathscr{M}_l$ that is transmitted from the source is equal to $\sum_{n\in\mathscr{N}}(1-\eta_n)$, because this packet targets all devices. On the other hand, the average number of successful receivers for any packet p in the set $\mathscr{M}_d$ with its corresponding vector v_p is equal to $\sum_{n|(v_p[n]\neq NULL)}(1-\eta_n)$, because this packet is innovative for any device $n \in \mathscr{N}$ for which $v_p[n]$ is not $NULL$ and thus the set of targeted receivers is $\mathscr{N}_r = \{n \mid (v_p[n] \neq NULL)\}$. Since $\mathscr{N}_r$ is a subset of $\mathscr{N}$, $\sum_{n\in\mathscr{N}}(1-\eta_n)$, the average number of successful receivers for any packet in $\mathscr{M}_c$ or $\mathscr{M}_l$, is greater than $\sum_{n\in\mathscr{N}_r}(1-\eta_n)$, the average number of successful receivers for any packet in $\mathscr{M}_d$ and thus, a packet from the set $\mathscr{M}_c$ or $\mathscr{M}_l$ is preferred to be transmitted from the source than a packet from $\mathscr{M}_d$. Between the sets $\mathscr{M}_c$ and $\mathscr{M}_l$, a packet from the set

$\mathscr{M}_c$ is preferred to be selected (lines 2-3), because the packets in $\mathscr{M}_c$ are not available to be transmitted through the other interface. Therefore, the order of transmitting the packets from the source is (i) $\mathscr{M}_c$, (ii) $\mathscr{M}_l$, and (iii) $\mathscr{M}_d$. There is no priority among the packets of the same set for the sets $\mathscr{M}_c$ and $\mathscr{M}_l$, because all packets have the same average number of successful receivers (lines 4-5). For the set $\mathscr{M}_d$, the packet with the maximum average number of successful receivers is preferred to be transmitted (lines 6-7).

Packet Selection by the Mobile Devices (lines 8-10): We first consider packet p from set $\mathscr{M}_l$ with its vector v_p. Each device $x \in \mathscr{N}$ can transmit a partial of this packet that is available in its *Has* set. $v_p[x]$ is the uncoded packet in the network coded packet p that is wanted by device x. Therefore, $v_p[x]$ is the only uncoded packet in v_p that is not available in the *Has* set of x. With that being said, x can transmit the partial of packet p which is constructed by network coding all packets in the vector $v_p \setminus v_p[x]$ with the targeted receivers $\mathscr{N}_r = \{i \mid (v_p[i] \neq v_p[x])\}$. Therefore, the average number of successful receivers for the partial packet transmitted from x is equal to $\sum_{n|(v_p[n]\neq v_p[x])}(1-\varepsilon_{x,n})$. We determine all possible transmitted packets from the set $\mathscr{M}_l$ along with their corresponding transmitters and calculate their average number of successful receivers (line 8). Then, we consider packet p from set $\mathscr{M}_d$ with its vector v_p. Each device x for which $v_p[x] = NULL$ can transmit packet p, because it has all uncoded packets of p in its *Has* set and the set of its targeted receivers is $\mathscr{N}_r = \{i \mid (v_p[i] \neq NULL)\}$. With that being said, the average number of successful receivers for packet $p \in \mathscr{M}_d$ transmitted from x is equal to $\sum_{n|(v_p[n]\neq NULL)}(1-\varepsilon_{x,n})$. We determine all possible transmitters for each packet from set $\mathscr{M}_d$ and calculate their average number of successful receivers (line 9). At last, we select the packet with the maximum average number of successful receivers among all packets from the sets $\mathscr{M}_l$ and $\mathscr{M}_d$ to be transmitted from its corresponding transmitter through D2D links (line 10).

4.3.2.2.2 Upper Bound on T

In order to characterize the performance of proposed `NCMI-Instant`, we develop the following upper bound for the completion time obtained from `NCMI-Instant` for the lossy channels in the second stage.

Theorem 5 *The average of completion time; T when `NCMI-Instant` is employed by cooperative mobile devices on a joint cellular and D2D setup when the channel links are lossy is upper bounded by*

$$T \leq \left\lceil \max\left(T_{s,c}, \frac{(T_{l,d}+T_{l,l})(T_{s,c}+T_{s,d}+T_{s,l})}{T_{l,d}+T_{l,l}+T_{s,d}+T_{s,l}}\right)\right\rceil, \tag{4.12}$$

where $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ are the sets that are constructed by Algorithm 4 for the first transmission slot and $T_{s,c}$, $T_{s,l}$, and $T_{s,d}$ are the average completion times for the source to transmit the packets in the sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$, respectively. $T_{l,l}$ and $T_{l,d}$ are the average completion times for the cooperating devices in the local area to transmit the packets in the sets $\mathscr{M}_l$ and $\mathscr{M}_d$, as calculated in the following:

$$T_{s,c} = \frac{|\mathscr{M}_c|}{1-\max\limits_{n\in\mathscr{N}}\eta_n}, \tag{4.13}$$

$$T_{s,l} = \frac{|\mathscr{M}_l|}{1-\max\limits_{n\in\mathscr{N}}\eta_n}, \tag{4.14}$$

$$T_{s,d} = \sum_{p \in \mathscr{M}_d} \frac{1}{1 - \max\limits_{n|(v_p[n] \neq NULL)} \eta_n}, \tag{4.15}$$

where v_p is the vector associated with the network coded packet p.

$$T_{l,l} = \sum_{p \in \mathscr{M}_l} \left(\frac{1}{1 - \max\limits_{n|(v_p[n] \neq v_p[x])} \varepsilon_{x,n}} + \frac{1}{1 - \max\limits_{n|(v_p[n] = v_p[x])} \varepsilon_{x',n}} \right), \tag{4.16}$$

where x is the device to be selected to transmit the partial of packet $p \in \mathscr{M}_l$. According to Algorithm 6, x is selected such that the average number of successful receivers is maximized; $x = \arg\max_{i \in \mathscr{N}} \sum_{n|(v_p[n] \neq v_p[i])} (1 - \varepsilon_{i,n})$. x' is the device to be selected to transmit the residual of packet p, which includes the uncoded packet $v_p[x]$. x' is selected such that the average number of successful receivers is maximized; $x' = \arg\max_{i|(v_p[i] \neq v_p[x])} \sum_{n|(v_p[n] = v_p[x])} (1 - \varepsilon_{i,n})$.

$$T_{l,d} = \sum_{p \in \mathscr{M}_d} \frac{1}{1 - \max\limits_{n|(v_p[n] \neq NULL)} \varepsilon_{x,n}}, \tag{4.17}$$

where x is the device to be selected to transmit packet $p \in \mathscr{M}_d$. According to Algorithm 6, x is selected such that the average number of successful receivers is maximized; $x = \arg\max_{i|(v_p[i] = NULL)} \sum_{n|(v_p[n] \neq NULL)} (1 - \varepsilon_{i,n})$.

Proof: We consider the initial sets of $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ constructed for the first transmission slot of `NCMI-Instant`. The calculated number of transmission slots to transmit all the packets in these sets gives an upper bound on the completion time by `NCMI-Instant`. The reason is that in `NCMI-Instant`, the completion time is improved by updating the sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ at each transmission slot and thus the resulted completion time from `NCMI-Instant` is less than the completion time resulted from NCMI-Instant without updating the sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ at each transmission slot (*i.e.,* these sets are set to their initializations at the beginning of the first transmission slot). In the following, we first derive the expressions for the average completion times for the source to transmit the packets in $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$, denoted by $T_{s,c}$ (Eq. 4.13), $T_{s,l}$ (Eq. 4.14), and $T_{s,d}$ (Eq. 4.15), respectively and the expressions for the average completion times for the devices in the local area to transmit the packets in $\mathscr{M}_l$, and $\mathscr{M}_d$, denoted by $T_{l,l}$ (Eq. 4.16), and $T_{l,d}$ (Eq. 4.17), respectively. Then, we calculate the average completion time when NCMI-Instant, without updating the sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ at each transmission slot, is used to recover the missing packets in all devices.

The average number of transmissions from the transmitter (or the source) required to satisfy a device is equal to $1/(1-\varepsilon)$, where ε is the link loss between the transmitter (or the source) and the device. The average number of transmissions required to satisfy a set of targeted receivers is restricted by the required number of transmissions for the receiver with the maximum channel loss. This proves Eqs. 4.13, 4.14, 4.15, and 4.17. To transmit packet p in $\mathscr{M}_l$ from the cooperating devices in the local area, first device t (with the maximum average number of successful receivers) is selected among all devices to transmit a partial of the packet. The set of targeted receivers for this transmission is $\{n \mid (v_p[n] \neq v_p[x])\}$. Therefore, it takes an average of $\frac{1}{1-max_{n|(v_p[n]\neq v_p[x])}\varepsilon_{x,n}}$ transmissions to transmit partial of packet p (the first term in Eq. 4.16). Then, the residual part of p, which includes the uncoded packet in p that is wanted by device x ($v_p[x]$), is transmitted. This packet is transmitted from, x', the device with the maximum number of successful receivers, and targets the devices in the set $\{n \mid (v_p[n] = v_p[x])\}$). Therefore, it takes an average of $\frac{1}{1-max_{n|(v_p[n]=v_p[x])}\varepsilon_{x',n}}$ transmissions to transmit residual of packet p (the second term in Eq. 4.16). This proves Eq. 4.16.

To prove Eq. 4.6, we first consider two cases based on the relative completion time for transmitting the packets in the sets $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ and then calculate the average completion time obtained from each of these cases.

1. $T_{s,c} \geq (T_{l,l} + T_{l,d})$

   Under this condition, the source starts transmitting the packets in $\mathscr{M}_c$, which takes the average of $T_{s,c}$ transmissions. Meanwhile the packets in the set $\mathscr{M}_n = \mathscr{M}_l \cup \mathscr{M}_d$ are transmitted in the local area, which takes the average of $T_{l,l} + T_{l,d}$ transmissions. After $T_{l,l} + T_{l,d}$ transmission slots, all the packets in $\mathscr{M}_d$ and $\mathscr{M}_l$ are transmitted in the local area and the average of $|\mathscr{M}_c| - \frac{|\mathscr{M}_c|(T_{l,l}+T_{l,d})}{T_{s,c}} = \frac{|\mathscr{M}_c|(T_{s,c}-T_{l,l}-T_{l,d})}{T_{s,c}}$ packets are left from $\mathscr{M}_c$; it takes an average of $\frac{|\mathscr{M}_c|(T_{s,c}-T_{l,l}-T_{l,d})}{T_{s,c}} \times \frac{T_{s,c}}{|\mathscr{M}_c|} = T_{s,c} - T_{l,l} - T_{l,d}$ transmissions for the source to transmit these packets. By summing the required number of transmission slots, the average completion time under condition (1) is equal to:

$$T_{(1)} = T_{s,c} \tag{4.18}$$

2. $T_{s,c} \leq (T_{l,l} + T_{l,d})$

   Under this condition, the base station starts transmitting the packets in $\mathscr{M}_c$; meanwhile the packets in the set $\mathscr{M}_n = \mathscr{M}_l \cup \mathscr{M}_d$ are transmitted in the local area. Since $T_{s,c} \leq (T_{l,l} + T_{l,d})$, after $T_{s,c}$ transmission slots, all the packets in $\mathscr{M}_c$ have been transmitted by the source and the average of $|\mathscr{M}_d| + |\mathscr{M}_l| - \frac{T_{s,c}(|\mathscr{M}_l|+|\mathscr{M}_d|)}{T_{l,l}+T_{l,d}} = \frac{(T_{l,l}+T_{l,d}-T_{s,c})(|\mathscr{M}_l|+|\mathscr{M}_d|)}{T_{l,l}+T_{l,d}}$ packets are left from $\mathscr{M}_n$; It takes the average of $\frac{(T_{l,l}+T_{l,d}-T_{s,c})(|\mathscr{M}_l|+|\mathscr{M}_d|)}{T_{l,l}+T_{l,d}} \times \frac{(T_{l,l}+T_{l,d})(T_{s,l}+T_{s,d})}{(T_{l,l}+T_{l,d})+(T_{s,l}+T_{s,d})} \times \frac{1}{|\mathscr{M}_l|+|\mathscr{M}_d|} = \frac{(T_{s,l}+T_{s,d})(T_{l,l}+T_{l,d}-T_{s,c})}{(T_{l,l}+T_{l,d}+T_{s,l}+T_{s,d})}$ transmission slots to transmit these packets by using both the source and the cooperating devices in the local area. By summing the required number of transmission slots, the completion time under condition (2) is equal to:

$$
\begin{aligned}
T_{(2)} &= T_{s,c} + \frac{(T_{s,l}+T_{s,d})(T_{l,l}+T_{l,d}-T_{s,c})}{T_{l,l}+T_{l,d}+T_{s,l}+T_{s,d}} \\
&= \frac{T_{s,c}(T_{s,l}+T_{s,d})+T_{s,c}(T_{l,l}+T_{l,d})}{T_{l,l}+T_{l,d}+T_{s,l}+T_{s,d}} + \\
&\quad \frac{(T_{s,l}+T_{s,d})(T_{l,l}+T_{l,d})-T_{s,c}(T_{s,l}+T_{s,d})}{T_{l,l}+T_{l,d}+T_{s,l}+T_{s,d}} \\
&= \frac{T_{s,c}(T_{l,l}+T_{l,d})+(T_{s,l}+T_{s,d})(T_{l,l}+T_{l,d})}{T_{l,l}+T_{l,d}+T_{s,l}+T_{s,d}} \\
&= \frac{(T_{l,l}+T_{l,d})(T_{s,c}+T_{s,l}+T_{s,d})}{T_{l,l}+T_{l,d}+T_{s,l}+T_{s,d}}.
\end{aligned}
$$

By combining the completion time obtained from conditions (1) and (2), the upper bound presented in Theorem 5 is achieved. This concludes the proof. ■

4.3.2.2.3 Computational Complexity

In `NCMI-Instant` Algorithm 4 is run with the complexity of $O(M^2+N)$, at each transmission slot. Then, the source calculates the average number of successful receivers (if it is required to transmit a packet from the set $\mathscr{M}_d$) with the complexity of $O(MN)$, selects the packet with maximum average number of successful receivers with the complexity of $O(M)$, and creates a network coded packet with the complexity of $O(M)$ at each transmission slot. Meanwhile, in the local area, the average number of successful receivers is calculated for all devices as the transmitter and all packets with the complexity of $O(MN^2)$, the packet with the maximum average number of successful receivers with its corresponding transmitter is selected with the complexity of $O(MN)$, and a network coded packet with the complexity of $O(M)$ is created, at each transmission slot. Therefore, the complexity of `NCMI-Instant` at each transmission slot is $O(M^2+N^2)$. By considering the maximum of M transmission slots, the complexity of `NCMI-Instant` is polynomial with the complexity of $O(M^3+N^2)$. This computational complexity, by also taking additional steps such as dividing a file into smaller sets of M packets, makes both `NCMI-Instant` applicable for practical deployment.

4.4 Lower Bound on T

In this section, we develop a lower bound on the packet completion time when any network coding algorithm is employed by cooperative mobile devices on a joint cellular and D2D setup. The effectiveness of a network coding algorithm is evaluated by comparing the completion time obtained from the algorithm with the lower bound; the closer the completion time obtained from a network coding algorithm is to the lower bound, the more effective is the algorithm.

Theorem 6 *The packet completion time when network coding is employed by cooperative mobile devices on a joint cellular and D2D setup is lower bounded by:*

$$T \geq \left\lceil \max\left(\frac{|\mathscr{M}_c|}{1-\prod_{n\in\mathscr{N}}\eta_n}, \max_{n\in\mathscr{N}} \Big(\min_{x\in(\mathscr{N}\backslash n)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}} \Big) \right) \right\rceil . \tag{4.19}$$

Proof: To prove Eq. 4.19, we first derive a lower bound on the completion time to recover the missing packets in device n, when the cellular and D2D links are used, jointly. This term is denoted by T_n. In the best case scenario, device n receives two simultaneous transmissions at each transmission slot, one from the source with the loss probability of η_n and another one from the transmitter device x

with the loss probability of $\varepsilon_{x,n}$. Therefore, the average number of packets that device n receives at each transmission slot, is equal to $(1-\eta_n)+(1-\varepsilon_{x,n}) = 2-\eta_n-\varepsilon_{x,n}$ and thus, in the best case scenario, it takes an average of $\frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}}$ transmission slots to satisfy device n with the transmitter device x for the transmission in the local area. Again, in the best case scenario, x is selected such that T_n is minimized. Therefore, we have:

$$\begin{aligned} T_n &\geq \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}} \\ &\geq \min_{x\in(\mathscr{N}\setminus n)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}}. \end{aligned} \tag{4.20}$$

By using network coding algorithms, the completion time to satisfy all devices is equal to maximum completion time required to satisfy each device. In other words, we have:

$$\begin{aligned} T &= \max_{n\in\mathscr{N}} T_n \\ &\geq \max_{n\in\mathscr{N}}(\min_{x\in(\mathscr{N}\setminus n)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}}). \end{aligned} \tag{4.21}$$

On the other hand, the packets in $\mathscr{M}_c$ can only be sent through the cellular link, because they are not available in any of the devices and thus cannot be transmitted through D2D links. The average of required number of transmission slots for a packet in $\mathscr{M}_c$ to be received successfully by at least one of the devices (so that it will be available to be transmitted through D2D links) is equal to $\frac{1}{1-\prod_{n\in\mathscr{N}}\eta_n}$. Therefore, the minimum completion time should be larger than $\frac{|\mathscr{M}_c|}{1-\prod_{n\in\mathscr{N}}\eta_n}$. Thus, the completion time is bounded by $T \geq \max(\frac{|\mathscr{M}_c|}{1-\prod_{n\in\mathscr{N}}\eta_n}, \max_{n\in\mathscr{N}}(\min_{x\in(\mathscr{N}\setminus n)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}})$. Furthermore, since the completion time can only have an integer value, the completion time is lower bounded by $\lceil \max\,(\frac{|\mathscr{M}_c|}{1-\prod_{n\in\mathscr{N}}\eta_n}, \max_{n\in\mathscr{N}}$ $(\min_{x\in(\mathscr{N}\setminus n)} \frac{|\mathscr{W}_n|}{2-\eta_n-\varepsilon_{x,n}})\rceil$. This concludes the proof. ■

Corollary 7 *The packet completion time when network coding is employed by cooperative mobile devices on a joint cellular and D2D setup when the channel links are lossless in the second stage, is lower bounded by:*

$$T \geq \left\lceil \max\left(|\mathscr{M}_c|, \frac{1}{2}\max_{n\in\mathscr{N}} |\mathscr{W}_n|\right)\right\rceil. \tag{4.22}$$

Proof: By substituting $\eta_n = \varepsilon_{k,l} = 0, \forall n,k,l \in \mathscr{N}$ in (4.19), the lower bound provided in (4.22) is obtained. ■

As shown in the simulation results, our proposed `NCMI-Batch` and `NCMI-Instant` methods perform closer to the lower bound as compared to the baselines.

4.5 Signalling Overhead Analysis

In our proposed algorithms, at each transmission slot, two network coded packets are selected to be transmitted from the two interfaces of the cellular and D2D. The decision of which network coded packet to be transmitted from the cellular links is made by the source. Meanwhile, the decisions of which network coded packet and which device to be selected as the transmitter in the local area, are made by the controller (which is selected randomly among mobile devices). In order for the source and the controller to make these decisions, the information about the packets that each device has received successfully, needs to be transferred to the source and the controller. Therefore, the signaling overhead at each transmission slot is the sum of (i) O_d bits for sending the controller's decision to the selected transmitter device (the controller needs to inform the transmitter device about its decision of packet transmission through D2D), (ii) O_{nc} bits as network coding overhead (the coefficients of uncoded packets in a network coded packet should be included as header in the transmitted network coded packet), and (iii) O_{ack} bits for sending acknowledgment packets from the targeted receivers at the end of the transmission to the source and controller indicating successful/unsuccessful transmissions. The signaling overhead O_d is related to the packet transmission through D2D links and the signaling overheads related to O_{nc} and O_{ack} are related to the packet transmissions through both cellular and D2D links. We analyze the signaling overhead for our proposed methods, lossy and lossless `NCMI-Batch` and `NCMI-Instant`, in detail in the following.

Stage One: In stage one, we use the single interface of cellular links for packet transmissions. Therefore, there is no packet transmissions in the local area, so $O_d = 0$. O_{nc} is also equal to 0 as the uncoded packets are transmitted to all devices without network coding in stage one. At the end of each packet transmission, acknowledgment packets need to be transmitted from each device to both source and controller to provide them with the required information for making packet transmission decisions in stage two. Therefore, O_{ack} for each packet transmission and each user is equal to 2 bits (1 bit for sending the acknowledgment packet to the source and one more bit for sending the acknowledgment packet to the controller). Thus, the overhead fraction per packet transmission to each user is equal to $\frac{O_d+O_{nc}+O_{ack}}{P} = \frac{2}{P}$, where P is the size of each uncoded packet in bits.

Stage Two, Lossless NCMI-Batch: For lossless `NCMI-Batch`, at each transmission slot, the controller selects one of the devices as the transmitter to send a random linear combination of the packets in its *Has* set; a packet with the size of one bit is sufficient to be transmitted from the controller to the selected transmitter to inform the transmitter to send a packet; $O_d = 1$ bit. Then, two random linear network coded packets need to be transmitted; one from the source and another from the selected transmitter. For each random linear network coded packet, we need to include the coefficients of each uncoded packet as the packet header; e.g. a random linear combination of the packets $p_1, p_2, ..., p_M$ is equal to the network coded packet $p = a_1 p_1 + a_2 p_2 + ... + a_M p_M$. For transmitting the network coded packet p, all the coefficients of $a_i, i = 1, 2, ..., M$ are added to the data packet p as a packet header. The number of bits required for displaying each coefficient is equal to $log(F)$, where F is the size of the field from which the coefficients are selected. Therefore, O_{nc} is equal to $Mlog(F)$ bits for sending each network coded packet and $2Mlog(F)$ bits for sending the two network coded packets via the two interfaces of cellular and D2D. Finally, as the channels are lossless, all packets will be received successfully at the targeted receivers and there is no need to send acknowledgment packets at the end of each packet transmissions; $O_{ack} = 0$. Therefore, the overhead fraction per transmission slot is equal to $\frac{O_d+O_{nc}+O_{ack}}{2P} = \frac{1+2Mlog(F)}{2P}$ bits for lossless `NCMI-Batch`. Note that the multiplier 2 in the denominator is due to sending two network coded packets at each transmission slots over the two interfaces, each with the size of P.

Stage Two, Lossless NCMI-Instant: For lossless `NCMI-Instant`, at each transmission slot, the controller selects one of the devices as the transmitter and determines the set of packets to be XORed as the instantly decodable network coded packet and transmitted from the transmitter. In order for the controller to inform the transmitter which packets should be XORed, a packet with the size of M bits (M is the number of the missing packets in all devices) is required; if the ith bit is 1, p_i should be included in the IDNC packet and if it is equal to 0, p_i should not included in the IDNC packet. For example, assume that the controller determines the IDNC packet $p = p_1 + p_3 + p_4$ for the set of missing packets $\{p_1, p_2, p_3, p_4, p_5\}$, with $M = 5$ packets, to be transmitted from the selected transmitter. Then, the controller needs to send the packet containing $O_d = 5$ bits of $[10110]$ to the transmitter to inform it about its decision. Therefore, $O_d = M$ bits for `NCMI-Instant`. Then, the transmitter and the source need to send IDNC packets, in which the coefficients of each uncoded packet should be included as the header. As an IDNC packet is created by XORing the uncoded packets, 1 bit is sufficient to represent the coefficient of each uncoded packet, so M bits are required for the coefficients of all uncoded packets. Therefore, the total overhead due to sending network coding coefficients of the two IDNC packets, transmitted over cellular and D2D, is equal to $O_{nc} = 2M$ bits, at each transmission slot. Finally, since the channels are lossless, $O_{ack} = 0$, as discussed in the previous paragraph. Therefore, the overhead fraction per transmission slot is equal to $\frac{O_d+O_{nc}+O_{ack}}{2P} = \frac{M+2M}{2P} = \frac{3M}{2P}$ bits for lossless `NCMI-Instant`.

Stage Two, Lossy NCMI-Batch: The analysis of O_d and O_{nc} for lossy `NCMI-Batch`, is the same as lossless `NCMI-Batch`; $O_d = 1$ bit and $O_{nc} = 2Mlog(F)$ bits. However, since the channels are lossy here, at the end of each packet transmission, acknowledgment packets need to be transmitted from the targeted receivers to both source and controller to provide them with the information for making future packet transmission decisions. Therefore, for each packet transmission, the size of acknowledgment packet is equal to 2 bits (1 bit for sending the acknowledgment packet to the source and one more bit for sending the acknowledgment packet to the controller) for each targeted receiver. Since, at each transmission slot, two packets are transmitted, the size of overhead due to sending acknowledgement packets is 4 bits per each targeted receiver. Considering N_t as the number of targeted receivers, the total size of overhead due to sending acknowledgment packets, is equal to $O_{ack} = 4N_t$ bits, at each transmission slot. Therefore, the overhead fraction at each transmission slot is equal to

$\frac{O_d+O_{nc}+O_{ack}}{2P} = \frac{1+2Mlog(F)+4N_t}{2P} \leq \frac{1+2Mlog(F)+4N}{2P}$ for lossy `NCMI-Batch`. Note that here we assume that the acknowledgement packets are not lost, for the sake of simplicity. If an acknowledgement packet is lost, it will be retransmitted and thus the overhead will be slightly increased.

Stage Two, Lossy NCMI-Instant: The analysis of O_d and O_{nc} for lossy `NCMI-Instant`is the same as lossless `NCMI-Instant`; $O_d = M$ bits and $O_{nc} = 2M$ bits. Also, the analysis O_{ack} for lossy `NCMI-Instant` is the same as lossy `NCMI-Batch`; $O_{ack} = 4N_t$ bits. Therefore, the overhead fraction at each transmission slot is equal to $\frac{O_d+O_{nc}+O_{ack}}{2P} = \frac{M+2M+4N_t}{2P} = \frac{3M+4N_t}{2P} \leq \frac{3M+4N}{2P}$ for lossy `NCMI-Instant`.

In the following, we give an example on the calculation of overhead fraction for stage one and stage two of `NCMI-Batch` and `NCMI-Instant`.

Example 13 *Assume that the packet size of each transmitted packet is $P = 1000$ Bytes $= 8000$ bits, the field size is $F = 256$ for `NCMI-Batch`, the number of missing packets is $M = 30$ and the number of devices is $N = 5$. The overhead fraction for each packet transmission in stage one is equal to $\frac{2}{P} = 0.00025$. The overhead fraction at each transmission slot for lossless `NCMI-Batch`, lossless `NCMI-Instant`, lossy `NCMI-Batch`, and lossy `NCMI-Instant` at stage two is equal to $\frac{1+2Mlog(F)}{2P} = 0.03006$, $\frac{3M}{2P} = 0.00562$, $\frac{1+2Mlog(F)+4N_t}{2P} \leq 0.03131$, and $\frac{3M+4N_t}{2P} \leq 0.00687$, respectively."* □

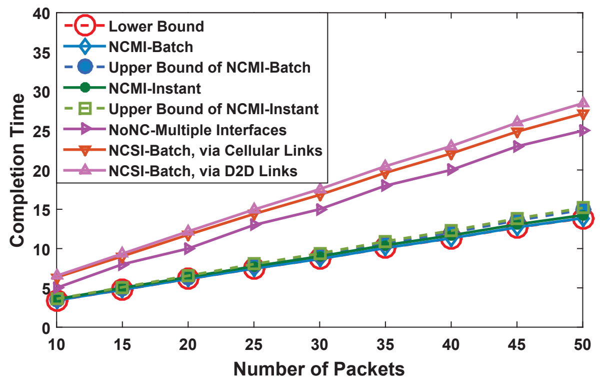
4.6 Simulation Results

We considered a topology shown in Fig. 12(b) for different number of devices, packets, and loss probabilities. Then we implemented our proposed methods, `NCMI-Batch` and `NCMI-Instant` for this topology and compared their performances with the lower bound and their upper bounds as well as the baselines. For each simulated point, we used 500 random events. In our simulation results, bounds are plotted using dashed lines, while the real simulation results are plotted using the solid lines. We first consider the case where the channels are lossless in the second stage. Then, we present the simulation results for lossy channels in the second stage. Finally, we investigate the effect of subpacketization on both lossless and lossy NCMI.

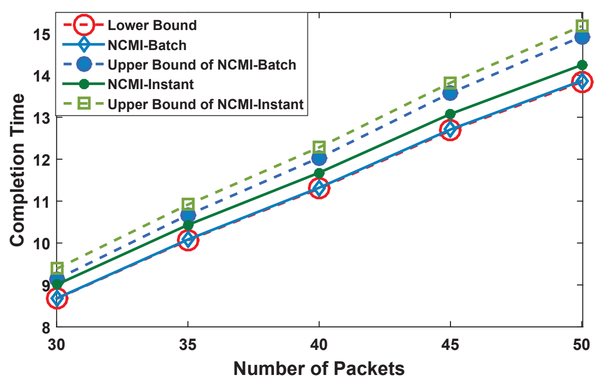
4.6.1 Lossless Channels in Stage Two

In this section, we consider a setup, in which each device misses the packets transmitted from the source in stage one with a specific loss probability. The loss probability for each device is selected uniformly from $[0.3, 0.5]$ for Figs. 15, 16, and 18. Note that the number of lost packets is equal to $M = |\bigcup_{n \in \mathcal{N}} W_n|$. All packet transmissions in stage two are assumed to be lossless. We implemented our proposed schemes in this setup, `NCMI-Batch` and `NCMI-Instant`, and compared their completion time performances with: (i) the *Lower Bound*, in Eq. 4.22, (ii) the *Upper Bounds* provided in Eq. 4.1 for `NCMI-Batch` and Eq. 4.2 for `NCMI-Instant`, (iii) *NoNC-Multiple Interfaces*, which is a no network coding scheme, but using cellular and D2D links jointly, (iv) *NCSI-Batch, via Cellular Links*, which uses batch-based network coding via the single interface of cellular links, (v) *NCSI-Batch, via D2D Links*, which uses batch-based network coding via the single interface of D2D links. Note that packets in $\mathcal{M}_c$ are requested from the source device via the cellular links in *NCSI-Batch, via D2D Links* scheme.

Completion time vs. number of packets: Fig. 15 shows the completion time for different number of packets and $N = 5$ devices. As seen, `NCMI-Instant` and `NCMI-Batch` improve the completion time significantly as compared to the single-interface systems and No-NC. This shows the effective-



(a) Completion time vs. number of packets



(b) Zoomed version of (a)

Figure 15: (a) Completion time vs. number of packets for `NCMI-Instant` and `NCMI-Batch` as compared to the lower bound and their upper bounds as well as baselines, when the channels are lossless in stage two. (b) Zoomed version of (a).

ness of (i) using multiple interfaces as compared to the single-interface systems, and (ii) employing network coding. `NCMI-Batch` and `NCMI-Instant` are slightly better than their upper bounds for

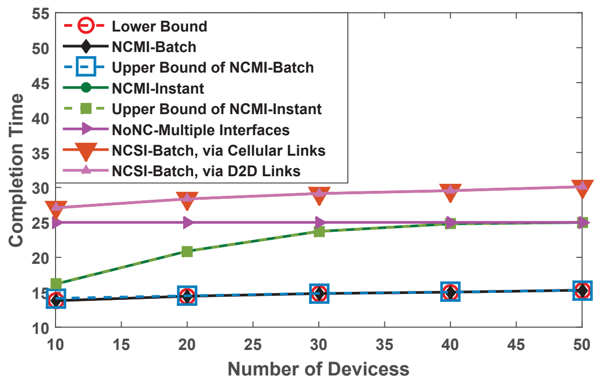


Figure 16: Completion time vs. number of devices for `NCMI-Instant` and `NCMI-Batch` as compared to the lower bound and their upper bounds as well as baselines, when the channels are lossless in stage two.

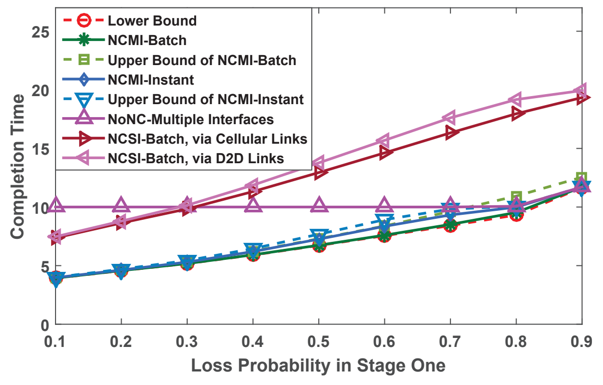


Figure 17: Completion time vs. loss probabilities of cellular links in stage one for `NCMI-Instant` and `NCMI-Batch` as compared to the lower bound and their upper bounds as well as baselines, when the channels are lossless in stage two.

larger number of lost packets, because the upper bounds give the worst case performance guarantee for `NCMI-Batch` and `NCMI-Instant`. Yet, as seen in the zoomed version of the figure (*i.e.,* Fig. 15(b)), the upper and lower bounds are very close to the actual performance of `NCMI-Batch` and `NCMI-Instant`. This shows the tightness of our upper and lower bound analysis.

Also, as seen in Fig. 15(s), single interface cellular system has less completion time than single interface D2D system. The reason is that any network coded packet can be created in the source and transmitted through the cellular link, while in the local area, only the network coded packets that can be created in one of the devices can be selected to be transmitted. Therefore, the network coded packet transmitted through the cellular links may target larger number of devices than the packets that are transmitted through D2D links and thus results in less completion time.

Completion time vs. number of devices: Fig. 16 shows the completion time for different number of devices and $M = 50$ packets. As shown in the figure, `NCMI-Batch` performs very close to the lower bound and upper bounds, and better than the single-interface systems and No-NC. `NCMI-Instant` performs better than single-interface systems and No-NC and worse than `NCMI-Batch`. The performance of `NCMI-Instant` gets closer to No-NC method for the larger number of devices. The reason is that the network coding opportunities of NCMI-Instant decrease in this lossless scenario when the number of devices increases, so NCMI-Instant gets closer to NoNC.

Completion time vs. loss probability: Fig. 17 presents the completion time for different loss probabilities in stage one when $M = 20$ and $N = 5$. In this setup, the loss probabilities of cellular links are the same for all mobile devices. As seen, the performances of NCMI schemes and the lower bound get closer to the no network coding scheme by increasing the channel losses of cellular links. In other words, when the channel losses are large, the no network coding scheme (with lower complexity than network coding schemes) has the performance close to the optimal scheme, which is defined by the lower bound.

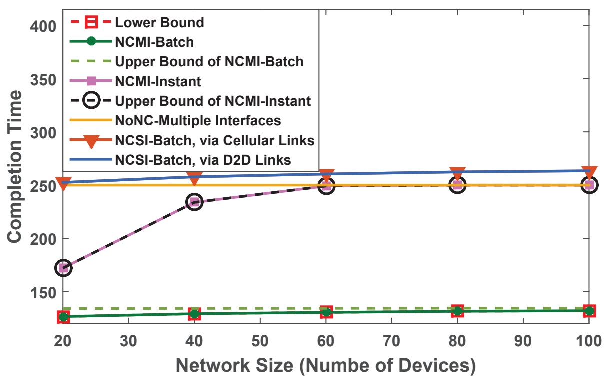


Figure 18: Scalability of `NCMI-Instant` and `NCMI-Batch` with the network size as compared to the lower bound and their upper bounds as well as baselines, when the channels are lossless in stage two.

Scalability with the Network Size: Fig. 18 shows the completion time for different network sizes with $M = 500$ packets. As seen, `NCMI-Batch` scales well with increasing the network size, but `NCMI-Instant` performs close to No-NC scheme for larger networks and fixed number of packets. In order for `NCMI-Instant` to perform close to its best performance, the number of packets should also be increased by increasing the size of the network.

4.6.2 Lossy Channels in Stage Two

In this section, we consider a setup in which the packet transmissions in both stages are lossy. In stage one, all packets are transmitted from the source; each transmitted packet is lost with the loss probability of η_n at each device n. In stage two, two packets are broadcast simultaneously at each transmission slot; one packet is broadcast from the source and is lost with the loss probability of ε_n at each device n and the other packet is broadcast from the transmitter device t (to all other devices) and is lost with the loss probability of $\varepsilon_{t,n}$ at each device n.

We implemented our proposed schemes, `NCMI-Batch` and `NCMI-Instant` in this setup and compared their completion time performances with the following: (i) *Lower Bound*, which is derived in (4.19). (ii) *Upper Bounds*, provided in (4.6) for `NCMI-Batch` and (4.12) for `NCMI-Instant`. (iii) *NoNC-Multiple Interfaces*, which is a no network coding scheme, but using cellular and D2D links jointly. At each transmission slot, each interface selects a missing packet with the maximum average number of receivers to be broadcast to all devices. (iv) *NCSI-Batch, via D2D Links*, which uses instantly decodable network coding via the single interface of D2D links. At each transmission slot, the device with the maximum size of *Has* set is selected as the transmitter and broadcast a random combination of all packets in its *Has* set to all other devices. (v) *NCSI-Instant, via D2D Links*, which uses instantly decodable network coding via the single interface of D2D links. At each transmission slot, packets are grouped to $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ sets according to Algorithm 4. Then, if $\mathscr{M}_c$ is not empty, the packets in this set are requested to be transmitted from the source until they are received successfully by at least one of the devices. If $\mathscr{M}_c$ is empty, the order of transmitting the packets is from the sets $\mathscr{M}_d$ and $\mathscr{M}_l$; at each transmission slot, the packet with the maximum average number of successful receiver is selected among all packets of the same set. (vi) *NCSI-Batch, via Cellular Links*, which uses batch-based network coding via the single interface of cellular links. At each transmission slot, a random combination of all packets is broadcast to all devices from the source. (vii) *NCSI-Instant, via Cellular Links*, which uses instantly decodable network coding via the single interface of cellular links. At each transmission slot, the union of the missing packets in all devices are grouped into $\mathscr{M}_c$, $\mathscr{M}_l$, and $\mathscr{M}_d$ sets according to Algorithm 4. The order of transmitting the packets is from the sets $\mathscr{M}_c$, $\mathscr{M}_l$ and $\mathscr{M}_d$.

Completion time vs. number of packets/number of devices: Fig. 19(a) shows the completion time for different number of packets and $N = 5$ devices and Fig. 19(b) shows the completion time for different number of devices and $M = 50$ packets. The channel loss probabilities $\eta_n, n \in \mathscr{N}$ and $\varepsilon_{t,n}, t, n \in \mathscr{N}$ are selected uniformly from $[0.15, 0.35]$ for these figures. As seen, `NCMI-Batch` and `NCMI-Instant` improve the completion time significantly as compared to the single-interface systems and No-NC. Note that `NCMI-Instant` significantly improves over NoNC when the number of devices increases. This is different than what we observed in Fig. 16 (where `NCMI-Instant` gets closer to NoNC with increasing number of devices). The reason is that network coding opportunities of `NCMI-Instant` still exist in this scenario due to losses in stage two. Also, network coding schemes using the single interface of D2D links outperform the network coding schemes using single interface of cellular links, despite the fact that more various network coded packet can be transmitted from the cellular link than the D2D link. The reason is that the packets to satisfy each device n using D2D links can be transmitted from a variety of channel links with the loss probabilities of $\varepsilon_{t,n}, t \in (\mathscr{N} \setminus n)$ from which the most reliable link is selected. While the packets to satisfy each device n using the cellular links can only be transmitted from the cellular link with the fixed loss probability of η_n and thus if η_n is high, there is no other more reliable link to be selected. Finally, the performance of `NCMI-Instant` and `NCMI-Batch` are close to the lower bound and their upper bounds.

4.6.3 Effect of Subpacketization on Lossless and Lossy NCMI

We investigated the impact of subpacketization for a file with the size of $M = 100$ packets and $N = 5$ cooperating devices. We divided the file into several subfiles with equal size and then applied different methods on each subfile. As seen in Fig. 20, the completion time decreases with increasing the size of subfile. This means that it would be more efficient to apply proposed methods on the whole file instead of dividing it into subfiles and apply the proposed methods on each subfile. As seen, our NCMI algorithms still significantly improve the completion time as compared to the baselines.

4.7 Related Work

Network Coding for Single-Interface Systems: The performance of network coding has been evaluated for single-interface systems in literature. The problem of minimizing the number of broadcast transmissions required to satisfy all devices is considered in (El Rouayheb et al., 2007), and the bounds for completion time are developed. A deterministic linear network coding algorithm that minimizes the number of broadcast transmissions is considered in (Yu et al., 2014). Minimization of the completion delay while broadcasting instantly decodable network coding packets has been considered in (Sorour and Valaee, 2010b). The problem of recovering the missing content using cooperative data exchange utilizing local area connections is considered in (El Rouayheb et al., 2010) and (Sprintson et al., 2010), and the lower and upper bounds on the minimum number of transmissions are developed. Deterministic algorithms for the cooperative data exchange problem with polynomial time are designed in (Milosavljevic et al., 2011) and (A. Sprintson, 2010). In our previous work, (Keshtkarjahromi et al., 2015a), we considered data exchange problem for multimedia applications and proposed content-aware network coding schemes that improves content quality and reduces the completion time using D2D links only. As compared to (Keshtkarjahromi et al., 2015a), we consider cooperative mobile devices in the joint cellular and D2D setup, and develop a network coding scheme for this setup in this chapter.

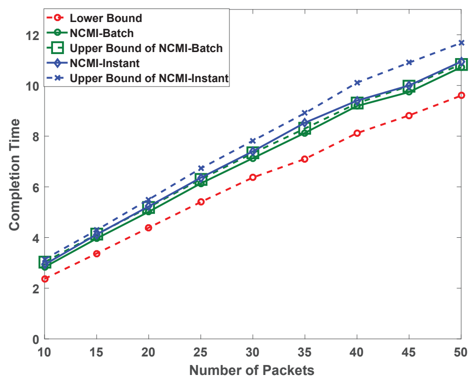
Multiple-Interface Systems: The performance of WiFi-only, cellular-only, and multiple-interface (WiFi plus cellular) systems are studied and compared in (Deng et al., 2014). A flexible software architecture is developed in (Nikravesh et al., 2016) by adaptively selecting available interfaces at mobile devices with the goal of improving Quality of Experience (QoE) while minimizing the energy overhead. The heterogeneity of cellular and Wi-Fi interfaces are effectively utilized to deliver data to mobile devices in (Tsao and Sivakumar, 2009). Cooperative content delivery to mobile devices is developed in (Barua et al., 2016) by taking into account the link quality of both cellular and WiFi interfaces, where a device with the best link quality is preferred in the cooperative system. The scenario of delivering same content to a group of users by using unicast cellular and D2D links is considered in (Stiemerling and Kiesel, 2009). A comprehensive survey on exploiting multiple available wireless interfaces to deliver content to cooperative mobile devices is provided in (Khan et al., 2016). This line of work demonstrates the practicality of simultaneous operation of multiple interfaces including cellular and short-range D2D links. As compared to this work, we consider how efficient network coding algorithms can be developed with provable performance guarantees for cooperative mobile devices in the joint broadcast cellular and broadcast D2D setup.

Network Coding for Multiple-Interface Systems: Network coding has been employed in the previous work for devices with multiple interfaces. Wireless video broadcasting with P2P error recovery is proposed by Li and Chan (Li and Chan, 2007). An efficient scheduling approach with network coding for wireless local repair is introduced by Saleh et al. (Saleh et al., 2011). Another body of work (Pedersen and Fitzek, 2008; Pedersen et al., 2009; Vingelmann et al., 2011) proposes systems where there are a base station broadcasting packets and a group of smartphone users helping each other to correct errors. Compared to prior work (Li and Chan, 2007; Saleh et al., 2011; Pedersen and Fitzek, 2008; Pedersen et al., 2009; Vingelmann et al., 2011), where each phone downloads all the data, and D2D links are used for error recovery, our scheme jointly utilizes cellular and D2D links and analyzes the performance of network coding in such a setup.

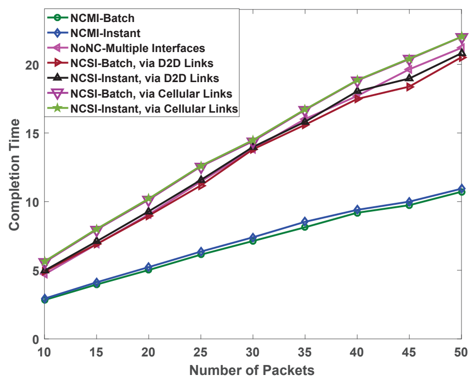
Simultaneous operation of multiple interfaces and employing network coding for this setup has also been considered in the previous work; (Keller et al., 2012; Seferoglu et al., 2011; ParandehGheibi et al., 2011). As compared to (Keller et al., 2012) and (Seferoglu et al., 2011), we consider broadcast cellular links simultaneously with D2D links for error recovery purposes, while (Keller et al., 2012) and (Seferoglu et al., 2011) use unicast cellular links simultaneously with D2D links for throughput improvement purposes. Multimedia streaming to a single user is considered in [25], where multiple interfaces are used at the single user. As compared to this work, we use cooperation among mobile devices using D2D links in conjunction with cellular links. Also, in this chapter, we consider how efficient network coding algorithms can be developed with provable performance guarantees for cooperative mobile devices in the joint cellular and D2D setup, instead of using existing network coding algorithms.

4.8 Summary

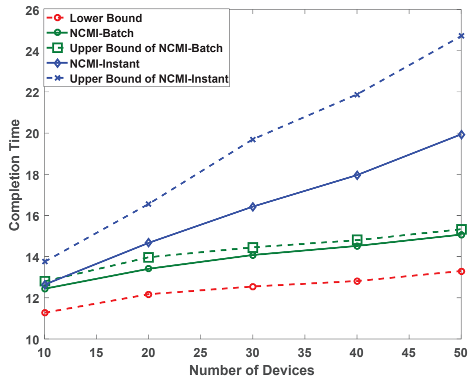
In this chapter, we considered a scenario where a group of mobile devices is interested in the same content, but each device has a partial content due to packet losses over links. In this setup, mobile devices cooperate and exploit the cellular and D2D links jointly to recover the missing content. We developed network coding schemes; `NCMI-Batch` and `NCMI-Instant` for this setup, and analyzed their completion time. Simulation results confirm that `NCMI-Batch` and `NCMI-Instant` significantly reduce the completion time.



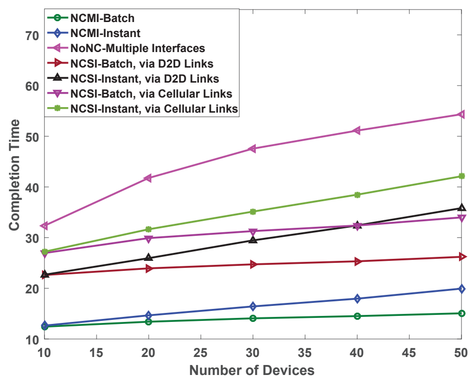
(a) Completion time vs. number of packets



(b) Completion time vs. number of packets

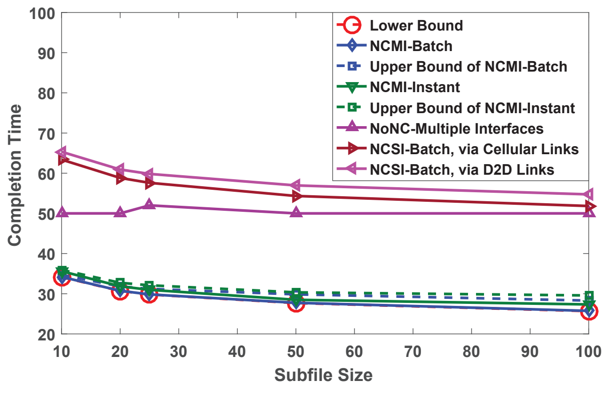


(c) Completion time vs. number of devices

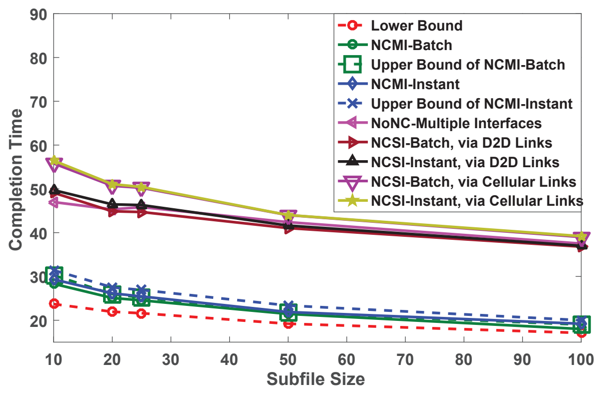


(d) Completion time vs. number of devices

Figure 19: The completion time performance of `NCMI-Instant` and `NCMI-Batch` as compared to the lower bound and the baselines, when the channels are lossy in stage two.



(a) Completion time vs. subfile size for lossless channels



(b) Completion time vs. subfile size for lossy channels

Figure 20: The impact of subpacketization on the completion time performance.

CHAPTER 5

CODED COOPERATIVE COMPUTATION FOR INTERNET OF THINGS

Computationally intensive tasks are continuously emerging in different practical applications including health, entertainment, and transportation systems over mobile devices and Internet of Things (IoT). It is crucial to handle such computationally intensive tasks by taking into account delay and energy consumption requirements. In this chapter, our goal is to exploit cooperation among mobile devices and coding to reduce the delay of the computationally intensive tasks, energy consumption of mobile devices, and bandwidth efficiency of the cooperative setup.

In particular, we consider a setup where a mobile device, which we name as collector, would like to perform a computationally intensive task. The task is partitioned into subtasks, and each subtask is offloaded to another device, which we name as helper. Each helper processes the subtask assigned to it, and sends the processed data back to the collector device. In this setup, coded computation is promising to improve delay and energy. However, existing work on coded computation does not consider the heterogeneity of helpers devices, which leads to waste of resources at the helper devices.

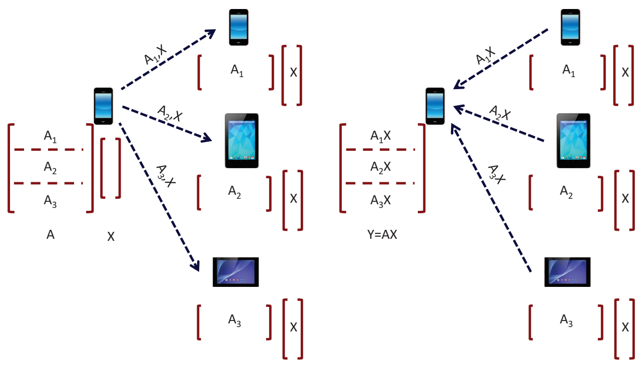
In this chapter, we consider coded computation problem with heterogeneous helpers, and (i) find the optimum task allocation to helper devices, and (ii) develop a dynamic task allocation algorithm that performs very close to the optimum solution.

5.1 Background

We consider a setup where a mobile device, which we name as collector, would like to perform a computationally intensive task. The task is partitioned into subtasks, and each subtask is offloaded to another device, which we name as helper using device-to-device (D2D) links. Each helper processes the subtask assigned to it, and sends the processed data back to the collector device via D2D links. This distributed computing approach is inspired by the well known MapReduce framework (Dean and Ghemawat, 2004).

In this work, our focus is on matrix-vector multiplication as it is the basis of many computationally intensive tasks. For this purpose, we consider matrix A, that needs to be multiplied with the vector X at the collector device. In cooperative computation setup, matrix A is divided into multiple submatrices with equal size and each submatrix along with the vector X is transmitted from the collector to the helpers Fig. 21(a). Then, each helper multiplies its received submatrix with vector X and sends back the result to the collector Fig. 21(b). At the collector side, the results that are received from the helpers are concatenated to obtain the final result of AX. It is shown that coding can reduce the computation delay for cooperative matrix-vector multiplication (Lee et al., 2016), (Li et al., 2015). In the following example, we explain how coded cooperative computation can reduce delay for the matrix-vector multiplication as compared to uncoded cooperation.

Example 14 *Let us consider that we have three helper devices and one collector device, which would like to calculate the matrix-vector multiplication of AX, where the number of rows of A is $R = 6$. In an uncoded cooperative system, the matrix A is divided into three submatrices with equal number of rows,* i.e., *each submatrices A_1, A_2, and A_3 have* 2 *rows. Now let us assume that it takes $CT_1 = 1$ time unit*



(a) Sending the subtasks from the collector to helpers

(b) Sending back the results of subtasks from the helpers to the collector

Figure 21: Example scenario of uncoded cooperative computation for multiplication of matrix A by vector X. (a) Matrix A is partitioned into A_1, A_2, and A_3; each partition along with the vector X is transmitted from the collector to one of the helpers. (b) Each helper multiplies its partition with vector X and sends the computed value back to the collector.

to compute one row at helper 1, $CT_2 = 2$ at helper 2, and $CT_3 = 10$ at helper 3. Thus, in the case of uncoded computation, computation delay is $D_1 = 1*2 = 2$ at helper 1, $D_2 = 2*2 = 4$ at helper 2, and $D_3 = 10*2 = 20$ at helper 3. As the collector should receive the computed values from from all three helpers, the overall delay is equal to $D = max(2,4,20) = 20$.

Now let us assume that we use coding. We can partition A into two submatrices of A_1 and A_2 with three rows each. Then, we send A_1 to helper 1, A_2 to helper 2, and $A_1 + A_2$ to helper 3. In this coded *setup, the collector should receive response from just any two of the helpers. Since each of the assigned submatrix to each helper contains three rows, the delays at the helpers are equal to* $D_1 = 3$, $D_2 = 6$ *and* $D_3 = 30$ *and thus the overall delay is equal to* $D = 6$. *As seen, by using coding, the delay is reduced from* 20 *to* 6, *which is significant.*

The above example demonstrates the benefit of coding for cooperative computation. However, fixed allocation of subtasks to helpers, without considering their heterogeneous nature is inefficient as demonstrated by the next example.

Example 15 *Let us assume that there are 3 helpers to compute AX, where A has* $R = 12$ *rows. Also, assume that the computation runtime for calculating one row are* $CT_1 = 1$, $CT_2 = 2$, *and* $CT_3 = 3$ *at helper 1, 2, and 3, respectively. If coding is not used, and fixed number of rows are allocated to each helper, the total delay becomes* $D = 18$. *If coding is used, but still fixed number of rows are allocated to each helper, the total delay is* $D = 16$. *On the other hand, if we allocate 6 rows to helper 1, 4 rows to helper 2, and 2 rows to helper 3, then the overall delay becomes* $D = 12$, *which is the minimum possible delay in this setup. Our goal in this work is to develop a task allocation framework that minimized the total delay by taking into account the heterogeneity of devices.*

As shown in the above example, in order to minimize delay, the number of rows that is assigned to each helper device, should be proportional to its computation capabilities. However, these capabilities are not known at the collector device a priori and they may vary over time. In this work, we consider this problem and propose a dynamic solution that estimates the capabilities of the helpers and assign the rows to the helpers accordingly.

5.2 System Model

We consider a mobile device, called the collector, that uses N other mobile devices, called helpers, in parallel for computing the matrix-vector multiplication of $Y = AX$, Fig. 22, where the number of rows of A is equal to R. For this purpose, r_n rows from matrix A along with vector X, are transmitted to each helper n. Then, each helper n starts computing the multiplication of each received row with vector X and sends it back to the controller. The controller generate the final result once it receives all computed rows from all helpers. Note that the summation of rows allocated to all helpers should be equal to the number of rows in matrix, R. In the following, we formulate provide theoretical analysis on how to determine the optimum row allocation to helpers by taking into account their computational capabilities. In the following, we first define the delay optimization problem and find the optimum row allocation analytically. Then, we propose a dynamic algorithm that performs close to the theoretical bounds.

5.3 Delay Optimization Problem

The overall delay for each helper n consists of three parts: (i) transmission delay for transmitting the first row from the collector to the helper n equal to $\frac{B_x}{C_n^u}$, where B_x is the size of each row (bits) and C_n^u is the capacity of uplink channel (bits/sec), (ii) computation time equal to $\frac{W r_n B_x}{C_n^{cpu}}$, where W is the required cycles per bit for computing one bit and C_n^{cpu} is the capacity of helper n (cycles/sec), and (iii) transmission delay for transmitting the last computed row from the helper to the collector equal to $\frac{B_r}{C_n^d}$, where B_r is the size of each computed row (bits) and C_n^d is the capacity of downlink channel (bits/sec). Note that we assume the propagation delay is negligible. Therefore, the overall delay for helper n is equal to:

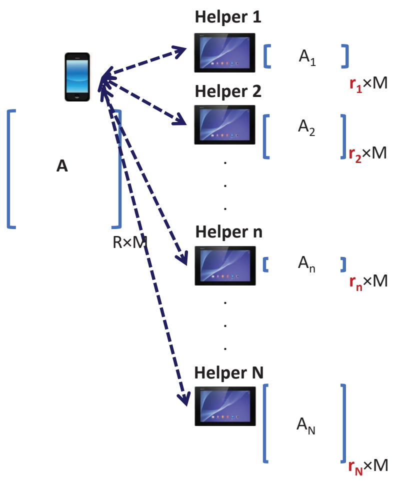


Figure 22: Computing matrix-vector multiplication of AX at the collector mobile device by using N mobile devices as helpers in parallel.

$$D_n = \frac{B_x}{C_n^u} + \frac{W r_n B_x}{C_n^{cpu}} + \frac{B_r}{C_n^d} \tag{5.1}$$

$$= \beta_n r_n + \alpha_n \tag{5.2}$$

$$\simeq \beta_n r_n, \tag{5.3}$$

where, $\beta_n = \frac{W B_x}{C_n^{cpu}}$ and $\alpha_n = \frac{B_x}{C_n^u} + \frac{B_r}{C_n^d}$. In the last line, we assume that transmission delay for sending one row is negligible as compared to computation time. Then, the overall delay is equal to the maximum delay over all helpers:

$$D = \max_{n \in \mathcal{N}} D_n. \tag{5.4}$$

Here, we want to find the optimum row allocation that minimizes the overall delay:

$$\text{minimize } \max_{n \in \mathcal{N}} \beta_n r_n \tag{5.5}$$

$$\text{subject to } \sum_{n \in \mathcal{N}} r_n = R \tag{5.6}$$

$$\& \ r_n \text{ is a non-negative integer } \forall n \in \mathcal{N}. \tag{5.7}$$

In the following, we provide the solution to this optimization problem.

5.4 Optimum Row Allocation to Helper Devices in Deterministic Case

I this section, we assume that the computation time per each row is fixed and known a priori at each helper, and solve the optimization problem of row assignment to each helper under this deterministic condition.

The first constraint of the above optimization problem, Eq. 5.6, states that the summation of the assigned rows to all helpers should be equal to the number of rows in the original matrix. Obviously, the number of assigned rows to each helper should be an integer, which is greater than or equal to zero, Eq. 5.7. Without considering the integer constraint, the minimum delay is achieved when D_n is equal for all helpers; in other words, $D = D_n = \beta_n r_n, \forall n \in \mathcal{N}$. Therefore, the optimum r_n is equal to $r_n = \frac{D_n}{\beta_n}$. On the other hand, we have $\sum_{n \in \mathcal{N}} r_n = R$, which results that the optimum r_n is equal to

Algorithm 7 Optimum row assignment when the channel conditions and hardware capability of each helper is constant and known as a priori

```
1: $r_n = 0, \forall n \in \mathcal{N}$.
2: while $\sum_{n \in \mathcal{N}} r_n < R$ do
3:     $D_n = \beta_n (r_n + 1), \forall n \in \mathcal{N}$.
4:     $n^* = \arg\min D_n$.
5:     $r_{n^*} = r_{n^*} + 1$.
```

$r_n = \frac{R}{\beta_n \sum_{n \in \mathcal{N}} 1/\beta_n}, \forall n \in \mathcal{N}$. However, this cannot be the optimum solution when integer constraint is considered. Next, we explain our proposed algorithm to find the optimum r_n as the solution to the optimization problem in Eq. 5.5 by considering all constraints of Eqs. 5.6 and 5.7.

The key idea of the non-integer solution is that the optimum rows assigned to each helper should be arranged in a way that the delay of all helpers should be roughly equal. We use the same idea for integer solutions. Our proposed solution is described in Algorithm 7. In this algorithm, first we initialize the number of rows assigned to each helper as 0, as stated in line 1. Then, we assign the rows one by one to each helper. For each row, we compute the resulted delay for each helper n if it is assigned to that helper; *i.e.,* if r_n is increased by one (line 3). Then, the row is assigned to the helper that has the minimum resulted delay D_n (lines 4 and 5). This continues until all rows are assigned (line 2). This algorithm makes sure that the overall delay for all helpers is roughly equal.

As seen, algorithm 7 finds the optimum row assignment to the helpers with the assumption that $\beta_n, \forall n \in \mathcal{N}$, which is the indicator of hardware capability of each helper n, is constant and known as a priori. However, in practice these parameters are not known as a priori. In addition, they may vary over time. In the next section, we consider a probabilistic case, where $\beta_n, n = 1,2,...,N$ is random with known distribution.

5.5 Optimum Row Allocation to Helper Devices in Probabilistic Case

In this section, we assume that β_n, the probability distribution of computation delay for computing one row for each helper n, is shifted exponential:

$$f_n(t) = \mu_n e^{-\mu_n(t-1)}. \tag{5.8}$$

Therefore, the delay for each helper n is equal to:

$$f_{D_n}(d_n) = \frac{\mu_n}{r_n} e^{-\mu_n(d_n/r_n-1)}. \tag{5.9}$$

Here, we consider the optimization problem of minimizing the average delay. In order to solve this optimization problem, $E(D)$, the mean of delay for all helpers should be equal, $E(D_1) = E(D_2) = ... = E(D_n) = ... = E(D_N)$. Therefore, the optimum row allocation is equal to:

$$r_n = \frac{R}{(1/\mu_n + 1)\sum_{n\in\mathscr{N}} \frac{\mu_n}{1+/\mu_n}}, \forall n \in \mathscr{N} \tag{5.10}$$

In this section, we characterized the computation delay of heterogeneous helpers and their variations over time by modeling the computation delay with a shifted exponential distribution. We assumed that the computation delay for each helper n is a random variable from a shifted exponential distribution. The heterogeneity of mobile devices is taken into account by selecting various distribution parameters for different helpers. Then, we found the optimal row allocation to each device to minimize the average overall delay. The assumption that the computational delay of helpers is random with a shifted exponential distribution, is not practical. Therefore, it is necessary to create a dynamic algorithm that adapts to the variations of computational delay over helpers and also over time. In the next section, we propose CCP, Computation Control Protocol, that assigns rows to each helper by estimating the computation delays and updating the estimation over time.

5.6 Computation Control Protocol (CCP)

In this section, we propose an algorithm that dynamically determines the the number of packets (matrix rows) to be offloaded to each helper for computation. In the best case scenario, the worker should not be underutilized waiting for a packet to compute, nor congested with too many packets waiting in its buffer to be computed. The ideal case is that each packet is arrived at a helper as soon as the computation of the ongoing packet is finished. We describe the details of the (i) ideal case, (ii) underutilized case, and (iii) congested case in the following. We explain what should be the control policies in the the underutilized and congested cases to push the system to the ideal case.

5.6.1 Ideal Case

The ideal case occurs when packet $i+1$ is received at the worker at the very same time computation of packet i is finished. This scenario is shown in Fig. 23(a). As seen, the time interval between sending the two consecutive packets i and $i+1$ from the collector to the helper should be equal to CT_i to have the ideal case scenario. The crucial question in this setup is how to determine CT_i. Calculating CT_i is not easy because (i) it varies depending on the hardware capabilities (*e.g.,* memory, energy, etc) of the helper and (ii) we are at the collector side and do not have access to the helper's capabilities instantaneously.

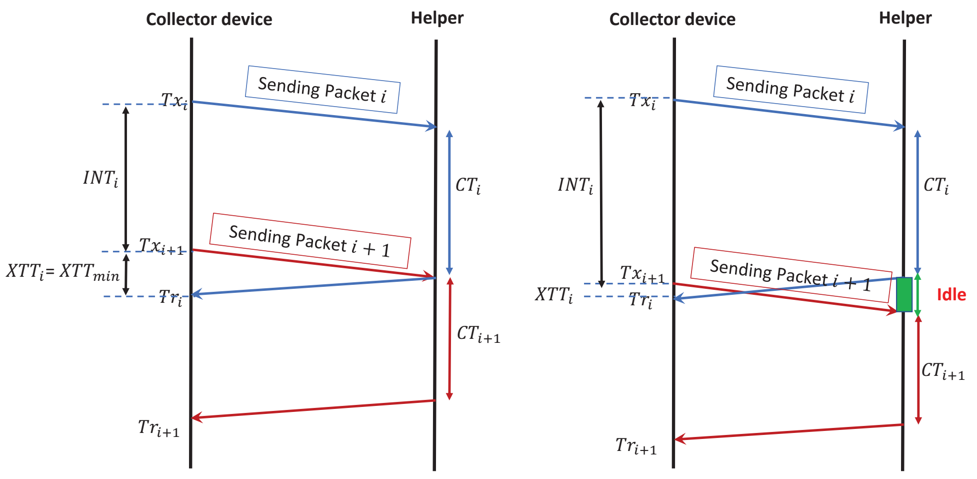
Therefore, we should develop practical and efficient mechanisms to determine CT_i. Towards this goal, we define XTT as the time interval between sending packet $i+1$ and receiving computed packet i; $XTT = Tr_i - Tx_{i+1}$. We call the required XTT in the ideal case as XTT_{min} (Fig. 23(a)). We discuss how to determine XTT_{min} from the variables available at the collector next.

5.6.2 Underutilized Case

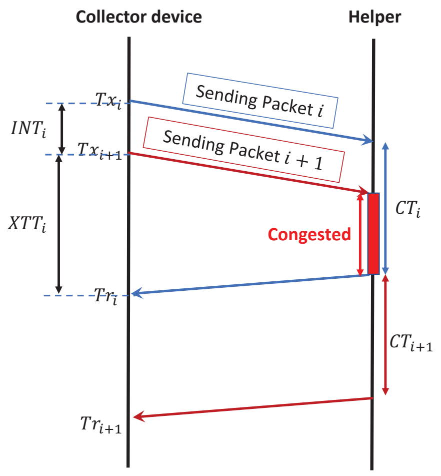
The underutilized case occurs when packet $i+1$ is received at the worker after a while that the computation of packet i is finished. In this case, the helper is underutilized for the time interval between finishing computing packet i and receiving packet $i+1$ from the collector for computing. This case scenario is shown in Fig. 23(b). As seen, the time interval between sending the two consecutive packets i and $i+1$ from the collector to the helper is longer than CT_i in the underutilized case, and XTT is less than XTT_{min}. Therefore, to push the system from the underutilized case to the ideal case, we should decrease the time interval between sending two consecutive packets from the collector by $XTT_{min} - XTT$ as stated in line 3 of Algorithm 8.

5.6.3 Congested Case

The congested case occurs when packet $i+1$ is received at the worker meanwhile the helper is busy with computing packet i. In this case, the received packet i should be queued in the worker buffer and wait for the worker to finish computing packet i. Therefore, the helper is congested for the time interval between receiving the packet $i+1$ from the helper and finishing computing packet i. This case scenario is shown in Fig. 23(c). As seen, the time interval between sending the two consecutive packets i and $i+1$ from the collector to the helper is shorter than CT_i in the congested case, and XTT is larger than XTT_{min}. Therefore, to push the system from the congested case to the ideal case, we should increase the time interval between sending two consecutive packets from the collector by $XTT - XTT_{min}$ as stated in line 6 of Algorithm 8.



(a) Ideal case scenario (b) Underutilized case scenario



(c) Congested case scenario

Figure 23: Different scenarios of sending packets from the collector to helper for computation.

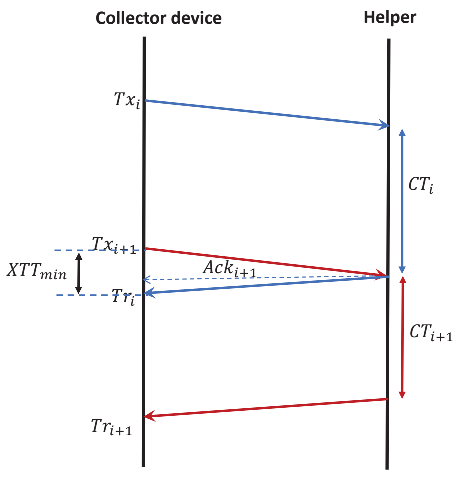


Figure 24: Calculating XTT_{min}

5.6.4 Calculating XTT_{min}

As seen in Fig. 23(a), XTT_{min} is the time interval between sending a packet (*e.g.,* packet $i+1$) from the collector to the helper and receiving a computed packet (*e.g.,* packet i) in the ideal case scenario. Assuming that we are using ACK based transport protocol (such as TCP) below CCP, then XTT_{min} becomes on the order of RTT. In particular, each helper sends an acknowledgment packet to the collector as soon as it receives a packet from the collector, where RTT is equal to the time interval between sending the packet from the collector and receiving its acknowledgement packet from the helper, Fig. 24. In this setup, since the size of acknowledgment packet is different from the size of the computed packet, RTT_{i+1} is not equal to XTT_{min}. However, by knowing the size of the computed packet and the size of the acknowledgement packet, we can calculate XTT_{min} from RTT_{i+1}, as shown in the following.

$$RTT_{i+1}(t) = \frac{B_x}{C_n^u} + \frac{B_r}{C_n^d} \tag{5.11}$$

$$XTT_{min}(t) = \frac{B_x}{C_n^u} + \frac{B_{ack}}{C_n^d}, \tag{5.12}$$

where, B_x is the size of the transmitted packet, B_r is the size of the computed packet, and B_{ack} is the size of the acknowledgment packet for the transmitted packet. Here, we assume that the capacity of downlink channel is equal to the capacity of uplink channel. Therefore, we can calculate $XTT_{min}(t)$ as the following:

$$XTT_{min}(t) = \frac{B_x + B_r}{B_x + B_{ack}} RTT_{i+1}. \tag{5.13}$$

Since, RTT (hence $XTT_{min}(t)$) varies over time, we approximate XTT_{min} as a weighted average of $XTT_{min}(t)$ and the calculate XTT_{min}, as stated in line 11 of Algorithm 8.

5.6.5 CCP Algorithm

In this section, we describe Algorithm 8, that updates the system parameters whenever a packet is received at the collector from the helper. If the received packet from the helper is a computed packet, then we need to update the time interval between sending consecutive packets from the collector to the helper based on the received time of the packet. This is described in sections 5.6.2 and 5.6.3 and stated in lines 1-6 of Algorithm 8. If the received packet from the helper is an acknowledgment packet, then we need to update XTT_{min} based on the round trip time of sending the packet to be computed to the

Algorithm 8 CCP algorithm for each helper n

```
1: if a computed packet $i$ is received at the collector before timeout; $(Tr_i - Tx_i) \leq TO$ then
2:     Calculate $XTT = Tr_i - Tx_{i+1}$ after receiving computed packet $i+1$ at the collector.
3:     if $XTT \leq XTT_{min}$ then
4:         Decrease the time interval between sending two consecutive packets by $XTT_{min} - XTT$; $Int = Int - (XTT_{min} - XTT)$.
5:         Update timeout as $TO = XTT_{min} + Int$.
6:     else if $XTT \geq XTT_{min}$ then
7:         Increase the time interval between sending two consecutive packets by $XTT - XTT_{min}$; $Int = Int + (XTT - XTT_{min})$.
8:         Update timeout as $TO = 2(XTT_{min} + Int)$.
9: else if an acknowledgement packet for the transmitted packet $i$ is received at the collector then
10:    Calculate $RTT_i$.
11:    Update $XTT_{min} = \alpha(RTT_i \frac{B_x+B_r}{B_x+B_{ack}}) + (1-\alpha)XTT_{min}$.
12:    Update timeout as $TO = 2(XTT_{min} + Int)$.
13: if the computed packet $i$ is timeout; $(Tr_i - Tx_i) > TO$ then
14:    Triple the time interval between sending two consecutive packets; $INT = 3INT$.
```

helper and receiving its acknowledgment at the collector. This is described in section 5.6.4 and stated in lines 7-12 of Algorithm 8.

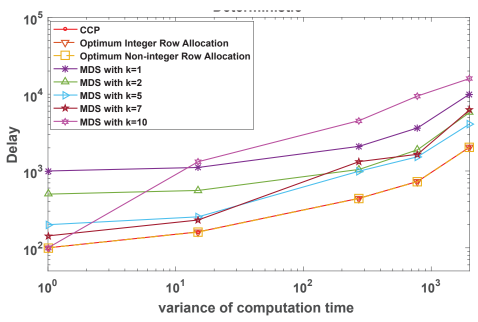
In CCP, the packets are transmitted orderly to the helpers based on the CCP algorithm described for each helper. If a helper becomes free, the delayed packets are transmitted to it. The procedure continues until all computed rows are received at the collector device. Also, we use fountain codes for creating packets from the rows, because if some packets are lost or delayed, the impact to overall delay is minimal thanks to coding (randomization of packets). In addition, fountain codes play well with our dynamic CCP algorithm due to their rateless property, had computationally efficient encoding and decoding algorithms and requires small overhead (*i.e.,* redundancy).

5.7 Simulation Results

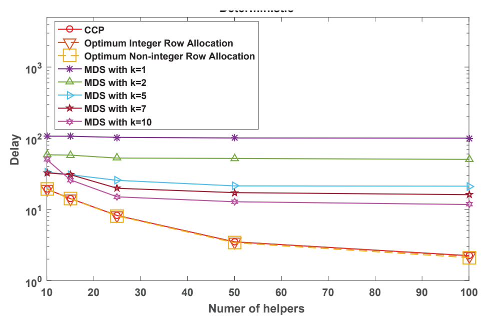
We simulated our proposed method and compared it with the theoretical bounds and MDS codes. The system parameters for these simulations are set to $R = 1000$ rows, $B_x = 8000$ bits, $B_{Ack} = 1$ bit, and $B_r = 8$ bits. The capacity of downlink channel is equal to capacity of uplink channel $C_n^d = C_n^u$ and is selected from a uniform distribution between 1 Mbps and 5 Mbps.

Fig. 25 considers the deterministic case. Fig. 25(a) shows delay versus variance of computation time over helpers for $N = 10$ helpers. As seen, CCP outperforms MDS coding by increasing the variance, which is an indicator of increasing heterogeneity among mobile devices. Fig. 25(b) shows delay versus number of helpers; as seen delay decreases by increasing the number of helpers for CCP, because CCP effectively uses the available resources.

Fig. 26 considers the probabilistic case. Here, the completion time for each helper is selected from a shifted exponential distribution. Fig. 26(a) shows delay versus average of computation time mean over all helpers for $N = 10$ helpers. As seen, CCP outperforms MDS coding and delay increases by increasing the mean computation time. Fig. 26(b) shows delay versus number of helpers; as seen delay



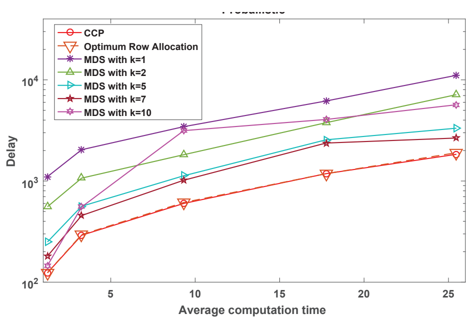
(a)



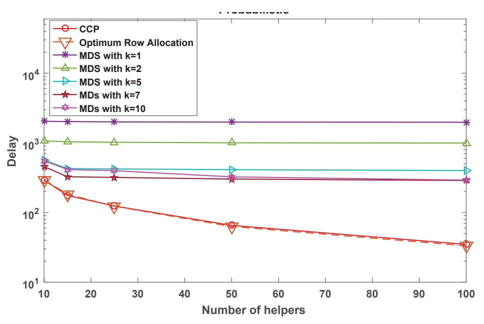
(b)

Figure 25: Deterministic scenario.

decreases by increasing the number of helpers for CCP, because CCP effectively uses the available resources.



(a)



(b)

Figure 26: Probabilistic scenario.

5.8 Summary

Coded cooperative computation is shown to improve the performance of computationally intensive tasks, which are continuously emerging in many diverse applications. However, it is crucial to take into account the heterogeneity of mobile devices used as helpers for assigning subtasks to these devices in a coded cooperative setup. In this chapter, we considered coded computation problem with heterogeneous helpers, and (i) found the optimum task allocation to helper devices, and (ii) developed a dynamic task allocation algorithm that performs very close to the optimum solution.

CHAPTER 6

CONCLUSION

In this thesis, we proposed schemes to improve throughput in wireless networks by taking advantage of cooperation among the nodes. In particular:

1. We proposed a multi-hop cooperative configuration for decentralized detection that maximizes the network lifetime and minimizes the probability of error detection as compared to the previous configurations.
2. We considered broadcasting a common content to a group of wireless devices. The whole content may once be broadcast from the source. The remaining missing content can then be recovered by using cooperation among the nodes. To recover the missing content, we proposed content-aware network coding schemes that improve quality of the content in multimedia applications with deadline constraint.
3. We considered the same scenario as 2. To recover the missing content at each node, we proposed network coding schemes that use multiple interfaces such as cellular and local area links simultaneously. Using multiple interfaces at the same time reduces the required amount of time to recover the missing packets compared to a single interface system.
4. We considered coded cooperative computation in distributed systems and proposed a dynamic algorithm that assigns subtasks to each helper devices by taking into account the heterogeneity of devices.

APPENDIX

COPYRIGHT PERMISSIONS

In this appendix, we present the copyright permissions for the articles, whose contents were used in this thesis. The list of the articles include WD'13 (Keshtkarjahromi et al., 2013), ICNC'15 (Keshtkarjahromi et al., 2015a) and Milcom'15 (Keshtkarjahromi et al., 2015b) conference papers following by an article in IEEE Transactions on Mobile Computing (Keshtkarjahromi et al., 2016a).

In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to `http://www.ieee.org/publications_standards/publications/rights/rights_link.html` to learn how to obtain a License from RightsLink.

CITED LITERATURE

[h26, 2006] H.264/avc reference software version jm 8.6 http://iphome.hhi.de/suehring/tml/download/, 2006.

[cis, 2010 2015] Cisco visual networking index: Global mobile data traffic forecast update, 2010-2015.

[A. Sprintson, 2010] A. Sprintson, P. Sadeghi, G. B. S. E. R.: Deterministic algorithm for coded cooperative data exchange. In QShine, Nov. 2010.

[Aboutorab and Sadeghi, 2016] Aboutorab, N. and Sadeghi, P.: Instantly decodable network coding for completion time or decoding delay reduction in cooperative data exchange systems. IEEE Transactions on Vehicular Technology, 65(3):1212–1228, March 2016.

[Ahlswede et al. , 2000] Ahlswede, R., Cai, N., Li, S.-Y., and Yeung, R.: Network information flow. Information Theory, IEEE Transactions on, 46(4):1204–1216, Jul 2000.

[Barua et al. , 2016] Barua, B., Khan, Z., Han, Z., Abouzeid, A. A., and Latva-aho, M.: Incentivizing selected devices to perform cooperative content delivery: A carrier aggregation-based approach. IEEE Transactions on Wireless Communications, 15(7):5030–5045, July 2016.

[Blum et al. , 1997] Blum, R., Kassam, S., and Poor, H.: Distributed detection with multiple sensors i. advanced topics. Proceedings of the IEEE, 85(1):64 –79, Jan 1997.

[Brahma and Fragouli, 2012] Brahma, S. and Fragouli, C.: Pliable index coding. In Information Theory Proceedings (ISIT), 2012 IEEE International Symposium on, pages 2251–2255, July 2012.

[Chamberland and Veeravalli, 2003] Chamberland, J.-F. and Veeravalli, V.: Decentralized detection in sensor networks. Signal Processing, IEEE Transactions on, 51(2):407 – 416, Feb 2003.

[Chamberland and Veeravalli, 2007] Chamberland, J.-F. and Veeravalli, V.: Wireless sensors in distributed detection applications. Signal Processing Magazine, IEEE, 24(3):16 –25, May 2007.

[Cover and Thomas, 1991] Cover, T. and Thomas, J.: Elements of Information Theory. New York, Wiley, 1991.

[Dean and Ghemawat, 2004] Dean, J. and Ghemawat, S.: Mapreduce: Simplified data processing on large clusters. In 2004 ACM Conference on Symposium on Opearting Systems Design & Implementation, Dec 2004.

[Deng et al. , 2014] Deng, S., Netravali, R., Sivaraman, A., and Balakrishnan, H.: Wifi, lte, or both?: Measuring multi-homed wireless internet performance. In Proceedings of the 2014 Conference on Internet Measurement Conference, IMC '14, pages 181–194, New York, NY, USA, 2014. ACM.

[Draves et al. , 2004] Draves, R., Padhye, J., and Zill, B.: Routing in multi-radio, multi-hop wireless mesh networks. In Proceedings of the 10th Annual International Conference on Mobile Computing and Networking, MobiCom '04, pages 114–128, New York, NY, USA, 2004. ACM.

[El Rouayheb et al. , 2010] El Rouayheb, S., Sprintson, A., and Sadeghi, P.: On coding for cooperative data exchange. In Information Theory Workshop (ITW), 2010 IEEE, pages 1–5, Jan 2010.

[El Rouayheb et al. , 2007] El Rouayheb, S. Y., Chaudhry, M., and Sprintson, A.: On the minimum number of transmissions in single-hop wireless coding networks. In Information Theory Workshop, 2007. ITW '07. IEEE, pages 120–125, Sept 2007.

[G. P. Perrucci and Widmer, 2011] G. P. Perrucci, F. H. P. F. and Widmer, J.: Survey on energy consumption entities on the smartphone platform. In Proceedings of IEEE 73rd Vehicular Technology Conference (VTC Spring), 2011.

[Geng et al. , 2015] Geng, Y., Hu, W., Yang, Y., Gao, W., and Cao, G.: Energy-efficient computation offloading in cellular networks. In 2015 IEEE 23rd International Conference on Network Protocols (ICNP), pages 145–155, Nov 2015.

[Hassanalieragh et al. , 2015] Hassanalieragh, M., Page, A., Soyata, T., Sharma, G., Aktas, M., Mateos, G., Kantarci, B., and Andreescu, S.: Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Opportunities and challenges. In 2015 IEEE International Conference on Services Computing, pages 285–292, June 2015.

[Ho et al. , 2006] Ho, T., Medard, M., Koetter, R., Karger, D., Effros, M., Shi, J., and Leong, B.: A random linear network coding approach to multicast. Information Theory, IEEE Transactions on, 52(10):4413–4430, Oct 2006.

[Jaggi et al. , 2005] Jaggi, S., Sanders, P., Chou, P., Effros, M., Egner, S., Jain, K., and Tolhuizen, L.: Polynomial time algorithms for multicast network code construction. Information Theory, IEEE Transactions on, 51(6):1973–1982, June 2005.

[Katti et al. , 2008] Katti, S., Rahul, H., Hu, W., Katabi, D., Medard, M., and Crowcroft, J.: Xors in the air: Practical wireless network coding. Networking, IEEE/ACM Transactions on, 16(3):497–510, June 2008.

[Keller et al. , 2012] Keller, L., Le, A., Cici, B., Seferoglu, H., Fragouli, C., and Markopoulou, A.: Microcast: Cooperative video streaming on smartphones. In ACM MobiSys, June 2012.

[Keshtkarjahromi et al. , 2013] Keshtkarjahromi, Y., Ansari, R., and Khokhar, A.: Energy efficient decentralized detection based on bit-optimal multi-hop transmission in one-dimensional wireless sensor networks. In Wireless Days (WD), 2013 IFIP, pages 1–8, Nov 2013.

[Keshtkarjahromi et al. , 2015a] Keshtkarjahromi, Y., Seferoglu, H., Ansari, R., and Khokhar, A.: Content-aware instantly decodable network coding over wireless networks. In Computing, Networking and Communications (ICNC), 2015 International Conference on, pages 803–809, Feb 2015.

[Keshtkarjahromi et al. , 2015b] Keshtkarjahromi, Y., Seferoglu, H., Ansari, R., and Khokhar, A.: Network coding for cooperative mobile devices with multiple interfaces. In MILCOM 2015 - 2015 IEEE Military Communications Conference, pages 701–707, Oct 2015.

[Keshtkarjahromi et al. , 2016a] Keshtkarjahromi, Y., Seferoglu, H., Ansari, R., and Khokhar, A.: Content-aware network coding over device-to-device networks. IEEE Transactions on Mobile Computing, PP(99):1–1, 2016.

[Keshtkarjahromi et al. , 2016b] Keshtkarjahromi, Y., Seferoglu, H., Ansari, R., and Khokhar, A.: Device-to-device networking meets cellular via network coding. In http://ykesht2.people.uic.edu/D2DMeetsCellular.pdf, 2016.

[Khan et al. , 2016] Khan, Z., Vasilakos, A. V., Barua, B., Shahabuddin, S., and Ahmadi, H.: Cooperative content delivery exploiting multiple wireless interfaces: Methods, new technological developments, open research issues and a case study. Wirel. Netw., 22(6):1961–1983, August 2016.

[Kramer et al. , 2007] Kramer, G. K., Berry, R., El Gamal, A., El Gamal, H., Franceschetti, M., Gastpar, M., and Laneman, J.: Introduction to the special issue on models, theory, and codes for relaying and cooperation in communication networks [guest editorial]. Information Theory, IEEE Transactions on, 53(10):3297–3301, Oct 2007.

[Kyasanur and Vaidya, 2006] Kyasanur, P. and Vaidya, N. H.: Routing and link-layer protocols for multi-channel multi-interface ad hoc wireless networks. SIGMOBILE Mob. Comput. Commun. Rev., 10(1):31–43, January 2006.

[Le et al. , 2013] Le, A., Tehrani, A., Dimakis, A., and Markopoulou, A.: Instantly decodable network codes for real-time applications. In Network Coding (NetCod), 2013 International Symposium on, pages 1–6, June 2013.

[Lee et al. , 2016] Lee, K., Lam, M., Pedarsani, R., Papailiopoulos, D., and Ramchandran, K.: Speeding up distributed machine learning using codes. In 2016 IEEE International Symposium on Information Theory (ISIT), pages 1143–1147, July 2016.

[Li et al. , 2015] Li, S., Maddah-Ali, M. A., and Avestimehr, A. S.: Coded mapreduce. In 2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton), pages 964–971, Sept 2015.

[Li et al. , 2003] Li, S.-Y., Yeung, R., and Cai, N.: Linear network coding. Information Theory, IEEE Transactions on, 49(2):371–381, Feb 2003.

[Li and Chan, 2007] Li, S. and Chan, S.-H.: Bopper: Wireless video broadcasting with peer-to-peer error recovery. In Multimedia and Expo, 2007 IEEE International Conference on, pages 392–395, July 2007.

[Li and Dai, 2008] Li, W. and Dai, H.: Energy-efficient distributed detection via multihop transmission in sensor networks. Signal Processing Letters, IEEE, 15:265–268, 2008.

[Li et al. , 2011] Li, X., Wang, C.-C., and Lin, X.: On the capacity of immediately-decodable coding schemes for wireless stored-video broadcast with hard deadline constraints. Selected Areas in Communications, IEEE Journal on, 29(5):1094–1105, May 2011.

[Masazade et al. , 2010] Masazade, E., Rajagopalan, R., Varshney, P., Mohan, C., Sendur, G., and Keskinoz, M.: A multiobjective optimization approach to obtain decision thresholds for distributed detection in wireless sensor networks. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on, 40(2):444–457, 2010.

[Milosavljevic et al. , 2011] Milosavljevic, N., Pawar, S., El Rouayheb, S., Gastpar, M., and Ramchandran, K.: Deterministic algorithm for the cooperative data exchange problem. In Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on, pages 410–414, July 2011.

[Muhammad et al. , 2013] Muhammad, M., Berioli, M., Liva, G., and Giambene, G.: Instantly decodable network coding protocols with unequal error protection. In Communications (ICC), 2013 IEEE International Conference on, pages 5120–5125, June 2013.

[Nguyen et al. , 2007] Nguyen, D., Nguyen, T., and Yang, X.: Multimedia wireless transmission with network coding. In Packet Video 2007, pages 326–335, Nov 2007.

[Nikravesh et al. , 2016] Nikravesh, A., Guo, Y., Qian, F., Mao, Z. M., and Sen, S.: An in-depth understanding of multipath tcp on mobile devices: Measurement and system design. In Proceedings of the 22Nd Annual International Conference on Mobile Computing and Networking, MobiCom '16, pages 189–201, New York, NY, USA, 2016. ACM.

[Ortega and Ramchandran, 1998] Ortega, A. and Ramchandran, K.: Rate-distortion methods for image and video compression. IEEE Signal Processing Magazine, 15(6):23–50, Nov 1998.

[P. Sadeghi, 2010] P. Sadeghi, R. Shams, D. T.: An optimal adaptive network coding scheme for minimizing decoding delay in broadcast erasure channels. EURASIP Journal of Wireless Communications and Networking, pages 1–14, April 2010.

[ParandehGheibi et al. , 2011] ParandehGheibi, A., Medard, M., Ozdaglar, A., and Shakkottai, S.: Avoiding interruptions a qoe reliability function for streaming media applications. Selected Areas in Communications, IEEE Journal on, 29(5):1064–1074, May 2011.

[Pedersen and Fitzek, 2008] Pedersen, M. and Fitzek, F.: Implementation and performance evaluation of network coding for cooperative mobile devices. In Communications Workshops, 2008. ICC Workshops '08. IEEE International Conference on, pages 91–96, May 2008.

[Pedersen et al. , 2009] Pedersen, M., Heide, J., Fitzek, F., and Larsen, T.: Pictureviewer - a mobile application using network coding. In Wireless Conference, 2009. EW 2009. European, pages 151–156, May 2009.

[Pelillo, 2009] Pelillo, M.: Heuristics for Maximum Clique and Independent Set. Boston, MA, Springer, 2009.

[Rec., 2005] Rec., I.-T.: H.264 : Advanced video coding for generic audiovisual services, 2005.

[Sadeghi et al. , 2009] Sadeghi, P., Traskov, D., and Koetter, R.: Adaptive network coding for broadcast channels. In Network Coding, Theory, and Applications, 2009. NetCod '09. Workshop on, pages 80–85, June 2009.

[Saleh et al. , 2011] Saleh, J. B., Qiu, D., and Elhakeem, A. K.: Performance of an efficient scheduling approach to network coding for wireless local repair. Cyber Journals: JSAT, 2(1):49–58, January 2011.

[Seferoglu et al. , 2011] Seferoglu, H., Keller, L., Cici, B., Le, A., and Markopoulou, A.: Cooperative video streaming on smartphones. In Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on, pages 220–227, Sept 2011.

[Seferoglu and Markopoulou, 2007] Seferoglu, H. and Markopoulou, A.: Opportunistic network coding for video streaming over wireless. In Packet Video 2007, pages 191–200, Nov 2007.

[Sendonaris et al. , 2003] Sendonaris, A., Erkip, E., and Aazhang, B.: User cooperation diversity. part i. system description. Communications, IEEE Transactions on, 51(11):1927–1938, Nov 2003.

[Sorour et al. , 2013] Sorour, S., Aboutorab, N., Sadeghi, P., Karim, M., Al-Naffouri, T., and Alouini, M.-S.: Delay reduction in persistent erasure channels for generalized instantly decodable network coding. In Vehicular Technology Conference (VTC Spring), 2013 IEEE 77th, pages 1–5, June 2013.

[Sorour and Valaee, 2010a] Sorour, S. and Valaee, S.: Minimum broadcast decoding delay for generalized instantly decodable network coding. In Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE, pages 1–5, Dec 2010.

[Sorour and Valaee, 2010b] Sorour, S. and Valaee, S.: On minimizing broadcast completion delay for instantly decodable network coding. In Communications (ICC), 2010 IEEE International Conference on, pages 1–5, May 2010.

[Sorour and Valaee, 2011a] Sorour, S. and Valaee, S.: Completion delay minimization for instantly decodable network coding with limited feedback. In Communications (ICC), 2011 IEEE International Conference on, pages 1–5, June 2011.

[Sorour and Valaee, 2011b] Sorour, S. and Valaee, S.: Completion delay reduction in lossy feedback scenarios for instantly decodable network coding. In Personal Indoor and Mobile Radio Communications (PIMRC), 2011 IEEE 22nd International Symposium on, pages 2025–2029, Sept 2011.

[Sorour and Valaee, 2014] Sorour, S. and Valaee, S.: Completion delay minimization for instantly decodable network codes. Networking, IEEE/ACM Transactions on, PP(99):1–1, 2014.

[Sprintson et al. , 2010] Sprintson, A., Sadeghi, P., Booker, G., and El Rouayheb, S.: A randomized algorithm and performance bounds for coded cooperative data exchange. In Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on, pages 1888–1892, June 2010.

[Stiemerling and Kiesel, 2009] Stiemerling, M. and Kiesel, S.: A system for peer-to-peer video streaming in resource constrained mobile environments. In Proceedings of the 1st ACM Workshop on

User-provided Networking: Challenges and Opportunities, U-NET ’09, pages 25–30, New York, NY, USA, 2009. ACM.

[Tajbakhsh et al. , 2014] Tajbakhsh, S., Sadeghi, P., and Aboutorab, N.: Instantly decodable network codes for cooperative index coding problem over general topologies. In Communications Theory Workshop (AusCTW), 2014 Australian, pages 84–89, Feb 2014.

[Tajbakhsh et al. , 2011] Tajbakhsh, S., Sadeghi, P., and Shams, R.: A generalized model for cost and fairness analysis in coded cooperative data exchange. In Network Coding (NetCod), 2011 International Symposium on, pages 1–6, July 2011.

[Tenney and Sandell, 1980] Tenney, R. R. and Sandell, N. R.: Detection with distributed sensors. In Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on, volume 19, pages 433 –437, Dec. 1980.

[Tsao and Sivakumar, 2009] Tsao, C.-L. and Sivakumar, R.: On effectively exploiting multiple wireless interfaces in mobile hosts. In Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies, CoNEXT ’09, pages 337–348, New York, NY, USA, 2009. ACM.

[Tsitsiklis, 1993] Tsitsiklis, J.: Decentralized detection. In Advances in Statistical Signal Processing, volume 2, pages 297–344, 1993.

[Varshney, 1996] Varshney, P.: Distributed Detection and Data Fusion. New York, Springer-Verlag, 1996.

[Vingelmann et al. , 2011] Vingelmann, P., Pedersen, M., Fitzek, F., and Heide, J.: On-the-fly packet error recovery in a cooperative cluster of mobile devices. In Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE, pages 1–6, Dec 2011.

[Viswanathan and Varshney, 1997] Viswanathan, R. and Varshney, P.: Distributed detection with multiple sensors i. fundamentals. Proceedings of the IEEE, 85(1):54 –63, Jan 1997.

[Wang, 2010] Wang, C.-C.: Capacity of 1-to-k broadcast packet erasure channels with channel output feedback. In Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on, pages 1347–1354, Sept 2010.

[Yu et al. , 2014] Yu, M., Sadeghi, P., and Aboutorab, N.: On deterministic linear network coded broadcast and its relation to matroid theory. In Information Theory Workshop (ITW), 2014 IEEE, pages 536–540, Nov 2014.

VITA

YASAMAN KESHTKARJAHROMI

Education

Ph.D. Electrical and Computer Engineering 2010 – 2017
University of Illinois at Chicago

M.S. Electrical Engineering 2007 – 2010
University of Tehran

B.S. Electrical Engineering (Electronics) 2003 – 2007
Shiraz University

Publications

Yasaman Keshtkarjahromi, Hulya Seferoglu, Rashid Ansari, Ashfaq Khokhar, "Content-Aware Network Coding over Device-to-Device Networks," *in IEEE Transactions on Mobile Computing*, 2016.

Yasaman Keshtkarjahromi, Hulya Seferoglu, Rashid Ansari, Ashfaq Khokhar, "D2D Meets Cellular Using Network Coding," *in ACM CoNext Student Workshop*, Irvine, CA, Dec. 2016.

Yasaman Keshtkarjahromi, Hulya Seferoglu, Rashid Ansari, Ashfaq Khokhar, "Network Coding for Cooperative Mobile Devices with Multiple Interfaces," *in Proc. of IEEE Milcom*, Tampa, FL, Oct. 2015.

Yasaman Keshtkarjahromi, Hulya Seferoglu, Rashid Ansari, Ashfaq Khokhar, "Content-Aware Instantly Decodable Network Coding over Wireless Networks," *in Proc. of IEEE International Conference on Computing, Networking and Communications (ICNC)*, Anaheim, CA, Feb. 2015.

Yasaman Keshtkarjahromi, Mehrdad Valipour, Farshad Lahouti, "Multi-Level Distributed Arithmetic Coding with Nested Lattice Quantization," *in Proc. of IEEE Data Compression Conference (DCC)*, Snowbird, UT, Mar. 2014.

Yasaman Keshtkarjahromi, Rashid Ansari, Ashfaq Khokhar, "Energy Efficient Decentralized Detection Based on Bit-optimal Multi-hop Transmission in One-dimensional Wireless Sensor Network," *in Proc. of IEEE Wireless Days (WD)*,

	Valencia, Spain, Nov. 2013.
	Daniella Tuninetti, Natasha Devroye, Yasaman Keshtkarjahromi, "On Cognitive Channels with an Oblivion Constraint," *in Proc. of IEEE International ICST Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM)*, Osaka, Japan, June 2011.
	Mehran Yazdi, Zahra Adelpour, Batoul Bahraini, Yasaman Keshtkarjahromi, "Novel Ridge Orientation Based Approach for Fingerprint Identification Using Co-occurrence Matrix," *in Proceedings of World Academy of Science, Engineering and Technology*, vol. 26, pp. 371-375, Dec. 2007.
Awards	Provost Award for Graduate Research (Univ. of Illinois at Chicago, 2016)
	Best Poster Award in N^{2}Women Workshop, SIGCOMM Conference (Chicago, IL, 2014)
	Student Travel Grant (CoNext 2016, Milcom 2015, ICNP 2014)
	Student Travel Presenter's Award, DCC Conference (Snowbird, UT, 2014)
Presentations	Invited Talk at CRAB Workshop, ICNP, Raleigh, NC, Oct. 2014 on "Quality Improvement in Instantly Decodable Network Coding"
	Poster Presentation at N^{2}Women Workshop, SIGCOMM, Chicago, IL, Aug. 2014 on "Content-Aware Instantly Decodable Network Coding over Wireless Networks"
Experience	Summer Intern at Huawei R&D (July 2016 - Sep. 2016)
	Summer Intern at Nokia Bell-Labs (June 2015 - Aug. 2015)