國立臺灣大學電機資訊學院資訊工程研究所
碩士論文
Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Taiwan University
Master Thesis

臨機網路中基於Erasure Coding之路由與資料傳輸

Effective data and video file transfer in opportunistic networks

游鎮鴻
Chen-Hung Yu

指導教授：朱浩華 博士
Advisor: Hao-Hua Chu, Ph.D.

中華民國九十六年六月
June, 2007

Acknowledgments

首先最要感謝的是陳伶志博士，沒有他的指導，這篇論文將無法順利的完成。從一開始發現問題，到後續尋找解決的方法，陳博士都給我很大的幫助，我們一起研究、討論，我有不懂時，他會不厭其煩地教我，特別的是，他並不直接告訴我答案，而是給我可能的大方向，讓我自己去探索，如果我做錯了，他會提示我可能的錯誤，只要我有一點點的研究成果，他會鼓勵我，給我超過預期的稱讚，讓我充滿信心，有勇氣繼續嘗試，他不只是個老師，更像個朋友，除了老師對學生的嚴厲外，也有同輩間互相勉勵的情感，更難得的是，他不只是指導我論文的內容，也以身作則讓我了解"作研究"應有的態度和方法。

也要感謝我的指導教授，朱浩華老師，他讓我能毫無後顧之憂的從事我有興趣的研究，給我任何需要的資源，有他大力的支持，我的研究生涯才能如此順利，另外要感謝我的口試委員：黃寶儀老師跟藍崑展老師，你們在我的口試時，提出很多寶貴的意見。

我還要感謝我的同學們，嘉惠、芊秀、全隆，還有勇志、有德，感謝你/妳們的幫助與陪伴，還有實驗室的伙伴：創文、東昀和昊極，我們擁有革命般的情感。

最後，要感謝我的父母、家人，因爲有你們，我才有這一切。

Abstract

With the development and popularity of wireless technologies, many challenges beyond the capability of wired networks have been emerging in recent years. Although wireless data networks extend the service of Internet, they still have many constraints. Specifically, at least one complete path between the source and the destination pair should be found before transferring data. But in many scenarios, this constraint is hard to conform. In *opportunistic networks*, nodes connect to each other intermittently and opportunistically. Neither scheduled routing nor predicted routing can be utilized in such networks. If we can deal with these problems, we can greatly extend the utility of wireless data networks.

In this thesis, we propose an efficient routing approach called H-EC. It is a hybrid forwarding scheme based on erasure coding and replication methods. H-EC improves the performance of previous works in *small delay performance cases* while still keep excellent results in *worst delay performance cases*. We also propose three message scheduling algorithms of H-EC and compare their different performances. From simulation results, we find that although transferring data in opportunistic networks is difficult, our proposed approaches make it practical. Being conscious that successful delivery is very hard, we further apply Layered Multiple Description Coding (LMDC) for transferring video file in opportunistic networks in order to let end users can read incomplete video file. By LMDC based schemes, end users

can "preview" the video before completely receiving data. Moreover, LMDC based schemes decline the heavy variance in per frame quality.

摘要

隨著無線科技的發展與普及，許多超越以往有線網路能力所及的挑戰，在近幾年紛紛浮現出來。雖然無線網路擴展了Internet的服務，它們還是有許多限制。具體地說，在傳送資料前，發送端與接收端之間必須存在至少一條完整路徑，否則傳統的路由方法會宣告失敗。但在很多情況之下，這樣的限制是很難符合的。在臨機網路中，點跟點之間會間歇性且臨機性地作連結。在這種網路下，排程式或預測式的路由方法將很難被應用。如果能解決這些的問題，我們將可以大大地擴展無線資料網路的功能。

在這篇論文中，我們提出了一個有效的路由方法，稱作H-EC。它是個結合erasure coding和複製的傳送方法。H-EC比先前的方法在較小延遲的成績上更進步，同時也保留了其在最糟延遲下的傑出表現。我們也提出了在H-EC上的三種排程演算法，並比較其表現成果。從模擬結果我們發現，即使在臨機網路上傳送資料非常困難，我們提出的方法仍然可使其具有實用性。在了解到要完整的傳送資料非常困難後，我們更進一步將Layered Multiple Description Coding應用到臨機網路上作影片檔案的傳輸，以期許能讓終點端的使用者能讀取未完成的檔案。利用基於LMDC的方法，終點端的使用者可在尚未接收完資料前"預覽"影片檔。此外，基於LMDC的方法還能降低每張畫面間品質的差異。

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Although Internet is very successful, it still has many constraints. Conventional networks operate on the assumption that at least one complete path between the source and the destination must exist. If there are no such paths, current routing approaches usually assert fail. This problem also limits the capability of mobility. To support mobility, wireless technology is essential. The evolvement of wireless communication technologies extends the coverage of Internet. For instance, WiFi 802.11 provides Internet access for wireless users who are within the range of deployed access points. Another example is GPRS, which supplies Internet service through the widespread cellular infrastructure. But these two approaches do not provide as much flexibility as we envision. The WiFi access merely extends the Internet as the form of *last-hop wireless networks* and only available at a few "hotspots" without real mobility, whereas cellular networks are usually low bandwidth and expensive. Self-organized wireless networks such as ad hoc network have been proposed for ap-

plications without previous setting of infrastructure. But its applications are rarely beyond military or research community.

Benefiting from the recent development of wireless technologies, there are numerous emerging challenged network architectures. Such applications are wireless sensor networks (WSN), pocket switched networks (PSN), people networks, transportation networks, and etc. These challenged networks are different from conventional networks. An end-to-end path between the source and the destination pair does not always exist. Links between nodes are intermittent with high error rates. As a result, conventional routing protocols which need the existence of complete path can not operate properly. Routing in this kind of networks becomes taking place on a time-varying graph and utilizes the transfer opportunities of each link up. Hereafter we call the communication opportunity of a node encountering another node "contact".

Many approaches have been proposed to solve routing problem in these challenged networks. When *contacts* information can be obtained previously, one can easily design a routing scheme by scheduling transfer (e.g., linear programming routing in DTN of scheduled contacts [14]). Besides, if contacts are predictable, we can forward data based on some probability. For example, some approaches have been proposed to use contact history or node mobility information. Other approaches employ additional aids such as special nodes or robots to relay data. But what we concern is neither network with scheduled/predictable contacts nor using any assistance of special relay. Our goal is to investigate routing in *opportunistic networks*, whose contacts appear opportunistically.

We propose a routing scheme called H-EC, which is based on erasure coding and replication methods. By our proposed scheme, routing in opportunistic networks becomes practical. We will show evidence by evaluating H-EC on real network scenarios. We also discuss variances of H-EC and compare their performance results. Moreover, we apply Layered Multiple Description Coding (LMDC), which is originally proposed for multimedia stream transferring in conventional networks, to transfer video file in opportunistic networks. By using LMDC, receivers can read incomplete video data. It can also alleviate large variance in frame quality so that end users would play videos more smoothly.

The remainder of this thesis is as follows: we first introduce *Erasure Coding* and a forwarding scheme based on it in the Chapter 2. Based on knowledge of erasure coding, Chapter 3 describes our proposed schemes. Then we show detailed evaluations in Chapter 4. Finally we discuss related works in Chapter 5 and conclude in Chapter 6.

Chapter 2

Erasure Coding Background

In this section, we describe the erasure coding algorithm conceptually and a forwarding scheme based on it. This forwarding scheme can achieve better worst-case delivery delay performance by adding redundancy without the overhead of strict replication of the original data.

Erasure code operates by converting a message into a large number of coded blocks such that any sufficiently large subset of the generated blocks can be used to reconstruct the original message. The concept of erasure coding is illustrated in Fig. 2.1. The fraction of the blocks required is called the *rate*. Given n blocks and a rate R, optimal erasure codes produce $\frac{n}{R}$ coded blocks where any n blocks are sufficient to recover the original message. However, optimal erasure codes are costly (in terms of memory usage, CPU time or both) when n is large, so *near* optimal erasure codes which need slightly more than n blocks are often used.

Choosing a particular erasure coding algorithm involves trade-offs between en-

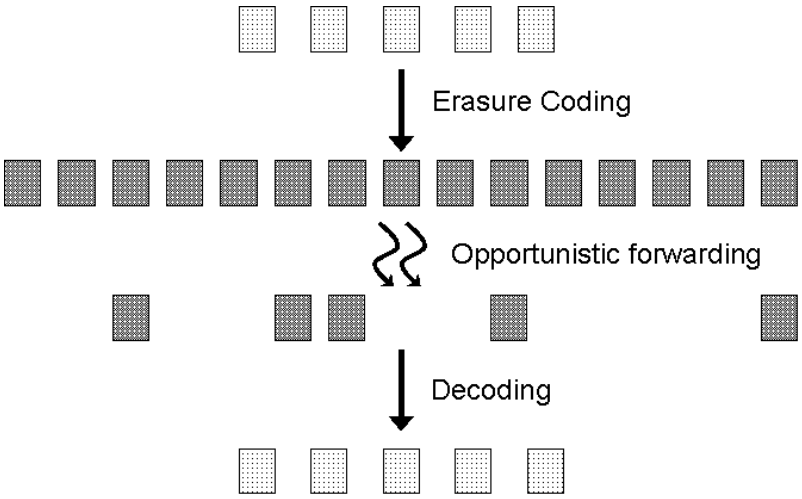


Figure 2.1: Illustration of Erasure Coding Forwarding

coding/decoding efficiency and the minimum number of coded blocks needed to reconstruct the original message. Those issues are beyond the purpose of this article. Generally speaking, given a message of size M, and a replication factor of erasure coding r, erasure codes produce $\frac{r \times M}{b}$ equal sized coded blocks of size b. After that any $\frac{(1+\epsilon) \times M}{b}$ of coded blocks which are altogether little larger then the original message can be used to reconstruct it. Here the variable ϵ is a small constant and varies depending on the certain algorithm used. The key aspect is that when using erasure coding with a *replication factor* r, only $\frac{1}{r}$ of the generated coded blocks are required to decode the message. Therefore we ignore the constant ϵ for simplicity. Constant b is the block size and is implementation dependent.

In [29], Yong Wang et al proposed a forwarding scheme (EC) based on the erasure coding, as illustrated in Fig.2.2. In that scheme, the source first encodes the message with replication factor r and generates a large number of coded blocks.

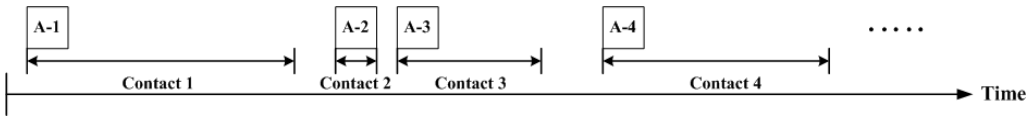


Figure 2.2: Illustration of the erasure coding based data forwarding algorithm (EC). In this figure, one erasure coded block (A) is equally split among four relays ($n = 4$).

These coded blocks are then equally split among the first n relays. Such relays are only allowed to send directly to the destination. By the property of erasure coding, once the destination receives equal or more than $\frac{1}{r}$ of the generated coded blocks, the message can be reconstructed. As reported in [29], this forwarding scheme can achieve the best worst-case delay performance with a fixed amount of overhead.

However, the drawback of this scheme is that it can not provide a good very small delay performance compared to other popular replication-based approaches. This is a consequence of that a destination has to collect enough number of coded blocks, in contrast to the simple replication scheme just one copy is enough. To reduce the number of coded blocks needed, one can increases the replication factor of erasure coding. But doing so leads to undesirable overhead. Besides, the source also spends time on spraying the coded blocks. It allocates the coded blocks to more distinct relays for more reliability. But if the network is sparse, namely the contact is rare, longer the source has to wait.

We are motivated by the above deficiency of EC scheme. Although EC scheme can achieve better worst-case delivery delay performance with a fixed amount of overhead, it can not obtain admirable delivery delay in very small delay performance cases. In next section, we will present a set of new schemes to resolve this problem.

We want to achieve good delivery delay not only in worst delay performance cases but also in very small delay performance cases.

Chapter 3

Proposed Schemes

We are going to present some forwarding schemes based on erasure coding in this section. Our purpose is to reduce the long delivery delay in very small delay performance cases and at the same time keep the good delay performance in worst-delay performance cases.

Contacts in a network are precious. When we talk about an opportunistic network, its contacts are not predictable. Especially in some sparse networks, nodes hardly contact with other nodes. For instance, in the trace collected by the ZebraNet group [31], the contact up-times are relatively short compared to the contact down-times. Therefore, it is important to efficiently utilize the available communication opportunity. In many deployed networks where nodes are battery-powered or located in volatile environments, the life time of nodes is uncertain, so that forwarding data to next custodian before it crashes becomes very important.

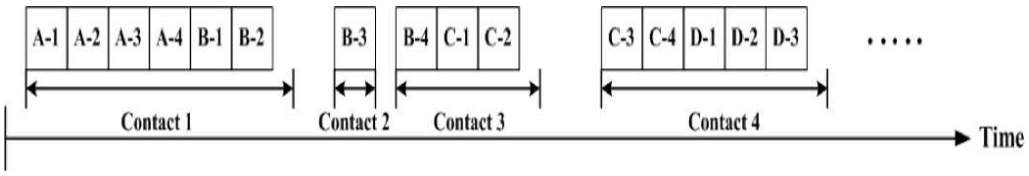


Figure 3.1: Illustration of the A-EC scheme, i.e., EC with aggressive forwarding. In this figure, four erasure coded blocks (A,B,C,D) are transmitted, and $n = 4$.

3.1 A-EC

As described in previous section, the EC scheme proposed in [29] has some defects. When using EC scheme, the source dispatches each coded block to different relay to mitigate the effects of a certain relay failure. Thus the source has to wait for enough number of contacts to distribute coded blocks rather then transmits on one contact as many as possible. In the meanwhile, it may not utilize every contact effectively. Because the number of transmitting blocks in each contact is fixed regardless of the duration of each contact. As a result, the erasure coding scheme tends to waste the residual contact period while the contacts are much longer than the required time.

To resolve this problem, we adapt the allocation method of original EC scheme and propose a scheme called A-EC (stands for Aggressive EC), as shown in Fig. 3.1. Under this scheme, the source sends as many coded blocks as possible during each contact period. As a result, this scheme is able to better utilize the network contact and thus expected to outperform EC scheme in very small delay performance cases.

However, for worst delay performance cases, A-EC yields a very large delivery delay or even poor delivery ratio when most employed relays are either unreliable

or hardly moving closer towards the destination. We call these relays *black-holes*. When these *black-holes* are present in the network, A-EC significantly degrades the overall performance of the message delivery. We will show the detail evaluation in section 4

One can easily address the problem of A-EC scheme. In order to better utilize each contact, the source usually sends more coded blocks belonging to a message to one relay. As soon as that relay reaches the destination, the destination can collect coded blocks more quickly. On the other side, if the relay can not reach the destination or delay for a long time, the destination might not be able to reconstruct the message.

3.2 H-EC

Taking advantage of the strengths of the EC and A-EC schemes, we want to achieve better message delivery performance in both worst delay performance and very small delay performance cases. We now propose a hybrid erasure coding based forwarding technique called H-EC, which is illustrated in Fig. 3.2.

In the H-EC scheme, two copies of EC blocks (constructed by the erasure coding and replication techniques described earlier) are transmitted by the sender. The first copy of EC blocks is sent in a similar way to the method used to send blocks in the original EC scheme (the white blocks in Fig. 3.2), while the second copy is sent using aggressive forwarding during the residual contact time after sending the first EC block (the gray blocks in Fig. 3.2). For general opportunistic network scenarios

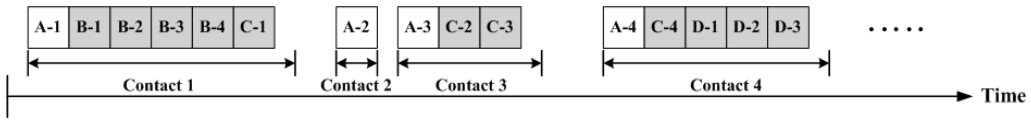


Figure 3.2: Illustration of H-EC scheme. In this figure, two copies of four erasure coded blocks (A,B,C,D) are transmitted: the first copy of EC blocks (the white blocks) is sent using EC algorithm, and the second copy (the gray ones) is sent using A-EC algorithm in the residual contact duration. Each coded block is equally split into 4 sub-blocks ($n = 4$). This is actually the HEC-SF scheme, we will elaborate more on this in the next subsection.

(i.e., without black-hole nodes), the H-EC scheme utilizes each contact opportunity better because of the aggressive forwarding feature; however, if black-hole nodes are present in the network, the scheme's performance is similar to that of the EC scheme, which achieves better forwarding in worst delay performance cases.

The performance of the H-EC scheme depends to a large extent on the message scheduling algorithm used in the aggressive forwarding phase. When the contact is longer but the associated relay crashes ultimately, the data will lose more. One obvious solution is suppressing the number of coded blocks belonging to the same message sent via a contact. Different thresholds may be diverse in performance result. Next subsections we focus on the second phase of H-EC and propose three message scheduling algorithms.

3.3 Message Scheduling in H-EC

We propose three message scheduling algorithms, namely Sequential Forwarding (SF), Full Interleaving (FI), and Block-based Interleaving (BI), for transmitting the

second copy of erasure coded blocks (i.e., using aggressive forwarding) in H-EC scheme. For simplicity, we assume the data file to be transferred in an opportunistic network is L messages in size. After applying erasure coding (i.e., adding redundancy), the size of each message is N erasure code blocks. We denote the n-th block of the l-th message as $M_{l,n}$. Moreover, we assume that each message can be reconstructed when at least B out of N blocks are successfully received by the destination node. We now describe the three algorithms in detail.

3.3.1 HEC-SF

The first is called HEC-SF representing the H-EC with Sequential Forwarding. As its name means, this algorithm sequentially sends the coded blocks in the second phase of H-EC, according with the order of the messages. There are two main advantages of this scheme. First, it is intuitive and easy to implement; second, it requires the minimal amount of buffer on the sender's side (i.e., it does not need to apply erasure coding to all messages in advance). The HEC-SF algorithm is detailed in Alg. 1.

3.3.2 HEC-FI

Sequentially forwarding coded blocks in the second phase of H-EC seems too naive and has risks when heavy-loaded relays crashes. While a contact is very long, its associated relay receives more coded blocks belonging to a certain message. This result forces a particular relay burdened with too much responsibility. In the second

Algorithm 1 The algorithm of H-EC with Sequential Forwarding (HEC-SF). The initial values of (l, n) are (0, 0).

Function HEC-SF (l, n)
if $l = L$ and $n = N$ **then**
  return $EndOfMessage$
else if $l = 0$ and $n = 0$ **then**
  $l \leftarrow 1$; $n \leftarrow 1$
  return $M_{l,n}$
else if $1 \leq l \leq L$ and $1 \leq n < N$ **then**
  $n \leftarrow n + 1$
  return $M_{l,n}$
else if $1 \leq l < L$ and $n = N$ **then**
  $l \leftarrow l + 1$; $n \leftarrow 1$
  return $M_{l,n}$
else
  return $Error$
end if

algorithm we concern more about reliability.

We now interleave the second copy of the erasure coded blocks for H-EC and propose a second scheme called HEC-FI, stands for Full Interleaving (FI). More precisely, while performing aggressive forwarding, the FI algorithm transmits the "first" coded block of all the messages at the outset, then the second block of all the messages, and so forth. The advantage of this scheme is it distributes the blocks of each message in a more diverse manner, and is thus expected to be more resilient to black-hole scenarios. The FI algorithm is detailed in Alg. 2

3.3.3 HEC-BI

The main drawback of the FI scheme is the very long response time needed to reconstruct messages when L and/or B are large (i.e., the time between sending the

Algorithm 2 The algorithm of H-EC with Full Interleaving (HEC-FI). The initial values of (l, n) are (0, 0).

Function HEC-FI (l, n)
if $l = L$ and $n = N$ **then**
  return $EndOfMessage$
else if $l = 0$ and $n = 0$ **then**
  $l \leftarrow 1$; $n \leftarrow 1$
  return $M_{l,n}$
else if $1 \leq l < L$ and $1 \leq n \leq N$ **then**
  $l \leftarrow l + 1$
  return $M_{l,n}$
else if $l = L$ and $1 \leq n < N$ **then**
  $l \leftarrow 1$; $n \leftarrow n + 1$
  return $M_{l,n}$
else
  return $Error$
end if

first block and successfully reconstructing the first message). More specifically, the response time of the FI scheme is much greater than the time required to forward the first $L \times (B-1)+1$ blocks[1]. A clever solution to this problem is to send B blocks during each contact period and interleave the sending process among L messages, instead of just sending a single block, as in the FI scheme. This is called the Block-based Interleaving (BI) scheme. Note that, the FI scheme is a specialized case of the BI scheme with B equal to 1. The algorithm for the BI scheme is detailed in Alg. 3.

[1]The response time will become even larger if we also consider data loss, data disorder, and inter-contact time.

Algorithm 3 The algorithm of H-EC with Block-based Interleaving (HEC-BI). The initial values of (l, n) are (0, 0).

Function HEC-BI (l, n)
$i \leftarrow Int(n/B)$; $j \leftarrow n \ mod \ B$
if $l = L$ and $n = N$ **then**
  return $EndOfMessage$
else if $l = 0$ and $n = 0$ **then**
  $l \leftarrow 1$; $n \leftarrow 1$
  return $M_{l,n}$
else if $j! = 0$ and $n < N$ **then**
  $n \leftarrow n + 1$
  return $M_{l,n}$
else if $(j = 0)$ or $(j \neq 0$ and $n = N)$ **then**
  if $l = L$ **then**
    $l \leftarrow 1$; $n \leftarrow (i + 1) \times B + 1$
  else
    $l \leftarrow l + 1$; $n \leftarrow i \times B + 1$
  end if
  return $M_{l,n}$
else
  return $Error$
end if

3.4 Layered Multiple Description Coding

In this section, we concentrate on the video file transfer in opportunistic networks. We first introduce the Layered Multiple Description Coding (LMDC), a scheme that combines Layered Coding and Multiple Description Coding for adapting to clients of different ability and gaining more resilience upon data loss [10]. Then we describe the detail of our scheme. With our design, users can "preview" the video without completely receiving the file. In addition, our design inherits from H-EC the good properties which are high delivery ratio and short delivery delay in both small delay performance and the worst delay performance cases.

Layered coding and multiple description coding have been proposed for peer-to-peer or multicast audio/video stream applications. MDC strips multimedia descriptions across multiple packets (or paths) and then transmits to a collection of clients. This method is used to combat both packet loss and component failure. Applications of MDC are IP level multicast [9] and application-level multicast [22, 23]. On the other hand, layered coding transmits to clients at different bit rates [21, 8]. while low-bandwidth clients would receive only a base layer, high-bandwidth clients could receive more enhancement layers thus get better quality. Combining all these characteristics, layered multiple description coding not only provides more reliable protection but also can transfer data to clients depending on their fittings. LMDC is usually implemented in conjunction with Unequal Erasure Protection (UEP) [10], which provides different levels of erasure protection to data by adding different amount of redundancy on the blocks of different importance. We propose the use of

LMDC for video file transfer in opportunistic networks, as following.

Our LMDC design is illustrated in Fig. 3.3. As the size of the collected video bit stream increases, the quality of a layered video frame increases. More specifically, suppose one of the layered video frames is S_k bits in size, one can split the video frame into k equal-sized pieces and reconstruct the video frame to Q_i quality level by using the first i out of the k pieces (i.e., the required bit stream size for reconstructing Q_i level frame is $S_i = i \times S_k/k$).

Each layered video frame is then spilt among N packets ($N \geq k$) with unequal erasure protection on each frame piece. For example, the i-th piece of the layered video frame is erasure coded with replication factor r_i and is split among N packets ; i.e., the i-th piece of the video frame can be reconstructed by any $\frac{N}{r_i}$ out of the N packets ($r_1 > r_2 > \cdots > r_{k-1} > r_k$ and $N \geq k$). The size of the i-th coded frame piece, b_i, can therefore be obtained by Eq. 3.1, and the size of the resulting N packets, b_{packet}, can be obtained by Eq. 3.2.

For simplicity, in this study, we let $r_i = \frac{N}{i}$ and $N = k$. The value of b_i and b_{packet} can thus be obtained by Eq. 3.3 and Eq. 3.4 respectively. Moreover, comparing to Layered Coding scheme, the traffic overhead of LMDC scheme, $b_{overhead}$, can be obtained by Eq. 3.5. Note that, since k is a positive integer (i.e., $k \geq 1$), one can conclude that (a) $b_{overhead} = 0$ when $N = k = 1$ (i.e., no LMDC); and (b) $b_{overhead} > 0$ otherwise.

Fig. 3.4 shows an example of an LMDC video frame at four different quality levels, i.e., Q_1, Q_3, Q_6, and Q_{10}, (k=10 and N=10). From this example illustration, it is clear that the perceived video frame quality significantly improves as the quality

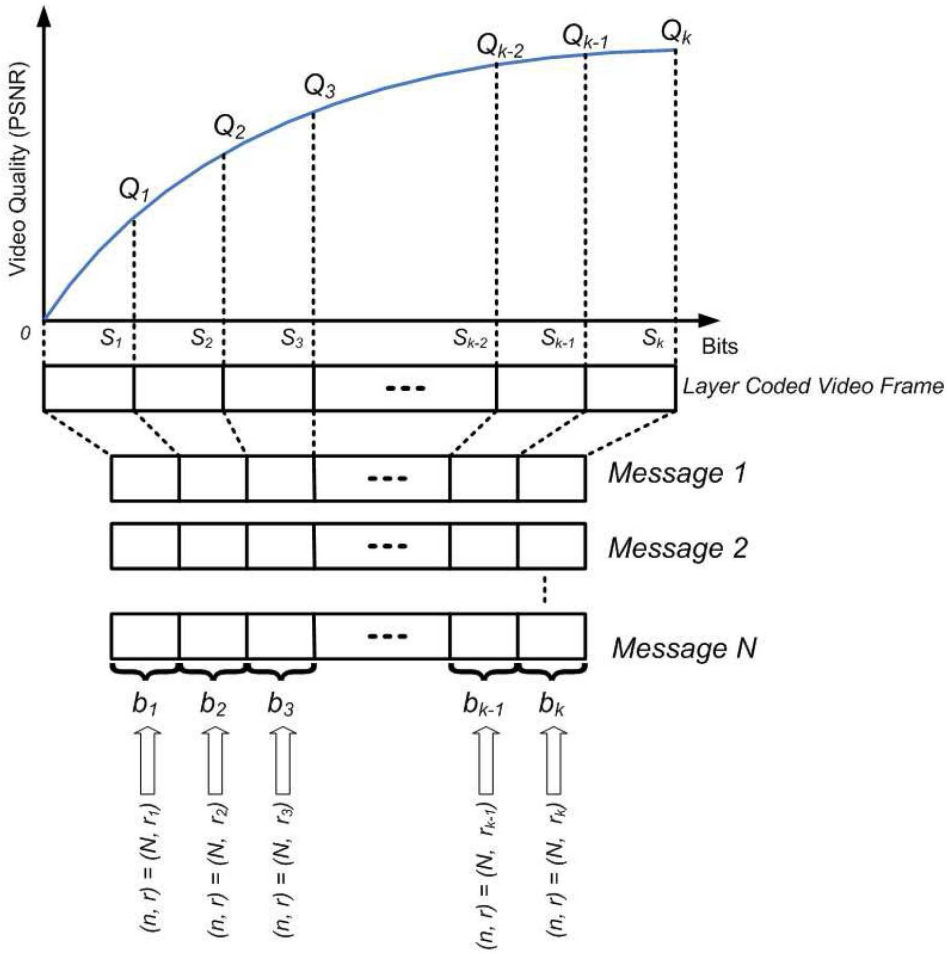


Figure 3.3: Illustration of Layered MDC scheme with unequal erasure protection: each video frame is encoded into k quality levels using layered coding, and the i-th quality level video frame is erasure protected with replication factor r_i and equally split among N relays ($r_1 > r_2 > \cdots > r_{k-1} > r_k$ and $N \geq k$).

level increases.

$$b_i = \frac{(S_i - S_{i-1}) \times r_i}{N} = \frac{S_k r_i}{kN} \tag{3.1}$$

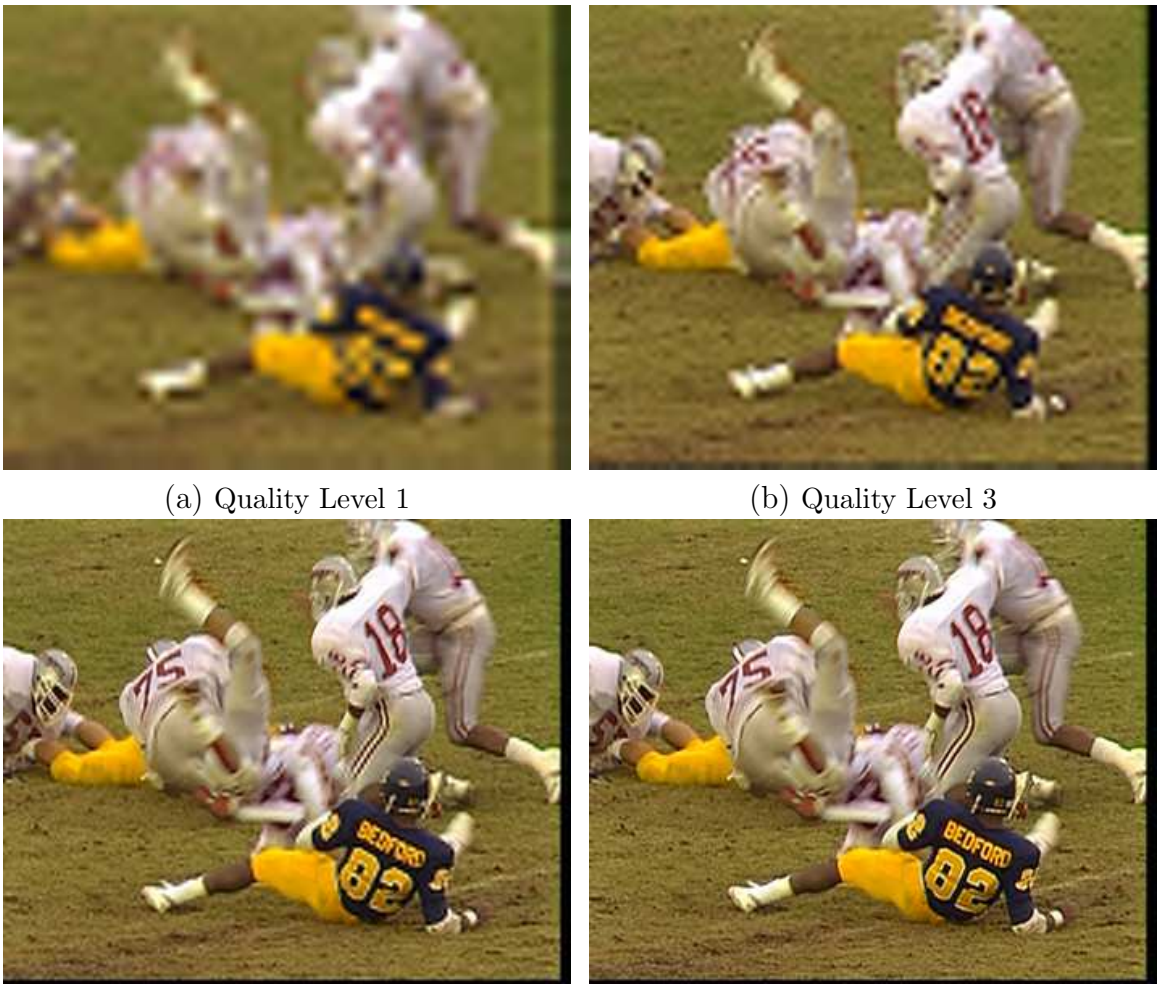
$$b_{packet} = \sum_{i=1}^{k} b_i = \frac{S_k}{kN} \sum_{i=1}^{k} r_i \tag{3.2}$$

$$b_i = \frac{S_k}{ik} \tag{3.3}$$

$$b_{packet} = \frac{S_k}{k} \sum_{i=1}^{k} \frac{1}{i} \tag{3.4}$$

$$b_{overhead} = N b_{packet} - S_k = S_k \sum_{i=2}^{k} \frac{1}{i} \tag{3.5}$$

We further extend the LMDC design above to gain more reliability with the same idea used in H-EC. The concepts of HEC-SF/FI/BI algorithms are applied to the LMDC scheme by sending the second copy of the LMDC messages during the residual contact period as in the HEC-SF/FI/BI schemes. The resulting schemes are named as LMDC-SF/FI/BI respectively. The complete evaluation will be discussed in the next section.



(a) Quality Level 1 (b) Quality Level 3

(c) Quality Level 6 (d) Quality Level 10

Figure 3.4: An example of LMDC video frame (k=10) at various quality levels.

Chapter 4

Evaluation

This section is about to evaluate our proposed H-EC schemes, and the LMDC-based schemes which are devoted to video file transfer in opportunistic networks. We implemented EC, HEC-SF, HEC-FI, and HEC-BI schemes and performed simulations in DTNSIM [2], a java based DTN simulator. We also assumed that data transmission is error-free at a fixed rate of 1Mbps. The simulation results presented in this section are all obtained by taking the average performance of 200 simulation runs, and in each run the source and the destination pair is randomly selected from all participating nodes. For all EC-based algorithms, the erasure coding parameters (r, n) are set to $(2, 16)$, which is consistent with the settings used in [29].

There are three scenarios we evaluated. One is generated according to the power-law distribution by setting both inter-contact time and contact-duration of the network with power-law distributed values of coefficient 0.6 (as reported in [6, 13]). The scenario consists of 34 nodes. The other two scenarios are all based on realistic

Table 4.1: Properties of various opportunistic network scenarios.

Trace Name	Power-Law	UCSD	Dartmouth
Device	N/A	PDA	WiFi Adapter
Network Type	N/A	WiFi	WiFi
Duration (days)	16	77	1,177
Devices participating	34	273	5,148
Number of contacts	25,959	195,364	172,308,320
Avg # Contacts/pair/day	2.89205	0.06834	0.01105

wireless network traces which are publicly available for research references. The first real trace is collected in Dartmouth campus [1], while the second is in UCSD [4]. Table 4.1 outlines the basic properties of each network scenario examined.

In more detail, the Dartmouth trace is an interface-based trace which recorded the APs that associated with a particular wireless interface during a three-year (1,177 days) period. However, it should be noted that wireless interfaces can be used by different devices at different times, and each device may use multiple wireless interfaces. For simplicity, we assume each network interface represents a single mobile user in the network.

The UCSD trace is a client-based trace that records the availability of WiFi-based access points (APs) for each participating portable device (e.g., PDAs and laptops) on the UCSD campus. The trace involves two and half months with a total of 273 participating devices. Similar to [6] [13] [7], we assume that both in UCSD and Dartmouth traces, two participating devices in ad hoc mode encounter a communication opportunity if and only if they are both associated with the same AP during the same time.

Our evaluations are in proper sequence. We first evaluate A-EC which is directly modified from EC scheme, and then investigate more delicate schemes in a traffic setting of constant bit rate. Next we focus on the file transfer performance of H-EC family and the video file transfer with LMDC based schemes.

4.1 Constant Bit Rate

In this subsection, we compare the EC, A-EC, and H-EC in the power-law scenario. Here we only exam the original version of H-EC, i.e., HEC-SF. For comparison, we also implement a simple replication based EC scheme (called R-EC), which sends EC blocks using *srep* algorithm as described in [29], with k replications for each erasure coded blocks. The total traffic amount of R-EC scheme is k times of the traffic amount of EC scheme.

In each simulation run we generate consistent data repeatedly (so-called Constant Bit Rate). We also extend the simulation duration to 16 days with 12 messages per day, and each message is 1MB in size.

4.1.1 A-EC in general network scenario

We first evaluate EC, A-EC, and R-EC (with $k = 2$ and 4) in the general network scenario. The result is depicted in Fig. 4.1 with Complementary CDF (CCDF) curves.

In fig. 4.1, A-EC obviously outperforms EC and R-EC in the left portion. This result confirms our intuition that aggressive forwarding can shorten the delivery

latency of some messages by making use of contacts more efficiently. However, the long *tail* of A-EC indicates that the last percentile has much longer delay than that of the other schemes. The reason is that A-EC forwards more coded blocks belonging to a message to an individual relay, thus the delay of that message will highly depend on the encounter time of that certain relay with the destination. If the selected relay is hard to contact with the destination, A-EC tends to fail.

As the number of replications of R-EC k increases, the delivery delay performance of R-EC improves significantly. The R-EC scheme can deliver 50% of data within 50,000 seconds when $k = 2$ and within 46,000 seconds when $k = 4$, while EC scheme delivers in 70,000 seconds. Although R-EC scheme seems to improve much as compared with EC scheme, it also brings more overhead. Note that the performance

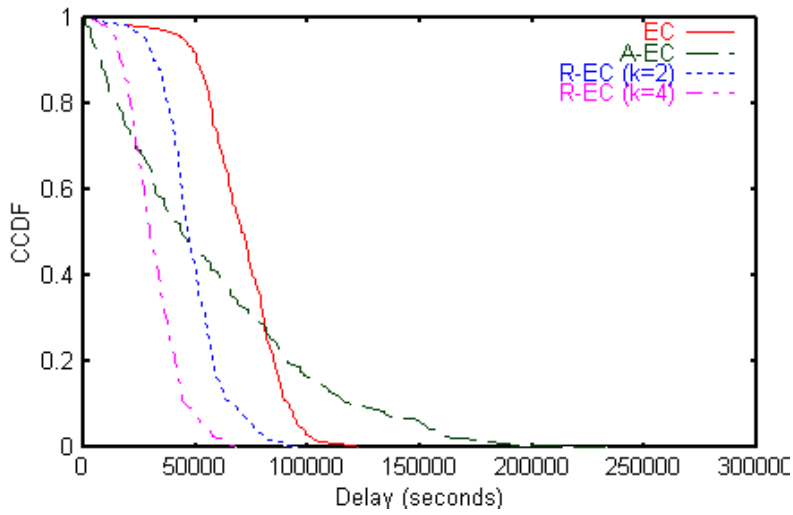


Figure 4.1: Latency distribution of EC, A-EC, and R-EC (k=2 and 4) schemes in general network scenarios. The distribution is shown in Complementary CDF (CCDF) curve.

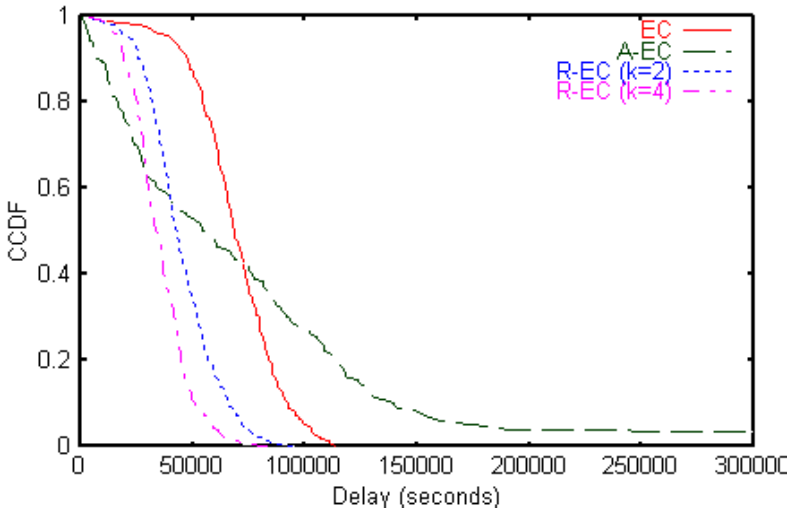


Figure 4.2: Latency distribution (CCDF) of EC, A-EC, and R-EC (k=2 and 4) schemes in *black-hole* network scenario.

improvement of R-EC is getting less as the number of replications increases. This is caused by the overhead accompanying the increase of replications. In addition, the performance of R-EC scheme converges to that of A-EC scheme for the very small delay performance cases when k increases. Because A-EC completely utilizes the contact duration, it turns out to be the performance upper bound in the very small delay performance cases. When the number of replications of R-EC scheme increases to the capacity of a contact, R-EC could completely utilize that contact as well as what A-EC does. From these observations, we can give k an upper bound, which is the maximum number of coded blocks that can be sent through a contact duration.

4.1.2 A-EC in black-hole scenario

In the second experiment, we evaluate A-EC in more difficult network scenario which we called black-hole scenario. In such network scenario, some of the relay nodes are extremely uncooperative (e.g., limited battery power, limited buffer size, or extremely large inter-contact time with the destination). Five (out of 32) relays are set as uncooperative nodes with very small buffer size (the buffer size is two-message size), while other simulation parameters are the same as the one employed in the previous subsection. Fig. 4.2 depicted the simulation results.

Although A-EC still outperforms the other schemes in good network connectivity, it suffers more danger when there are black-holes in the network. The long tail of A-EC implies it can not completely deliver data until the end of our simulation. Because A-EC sends the coded blocks aggressively, it may send a whole message to a certain relay. If that relay eventually crashes or is hard to contact with the destination, the message will be no longer recovered. In contrast, the EC based schemes spread the responsibility of forwarding over many nodes, therefore they can alleviate the impact of black-holes.

4.1.3 Evaluation of HEC

We now evaluate our proposed H-EC scheme (HEC-SF) in both general network and black-hole scenarios. Fig. 4.3 and Fig. 4.4 show the results respectively.

In the *very small delay performance* cases of both scenarios, H-EC performs as good as the A-EC scheme. While in the *worst delay performance* cases, H-EC only

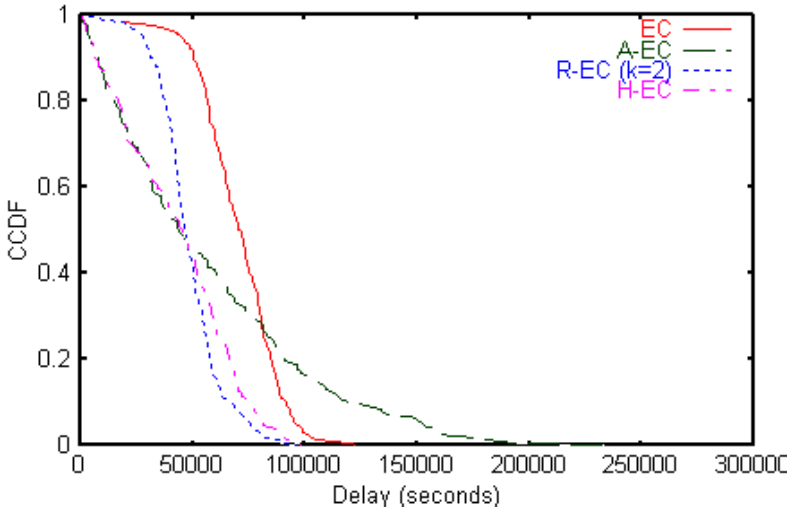


Figure 4.3: Latency distribution (CCDF) of EC, A-EC, R-EC (k=2) and H-EC schemes in general network scenarios.

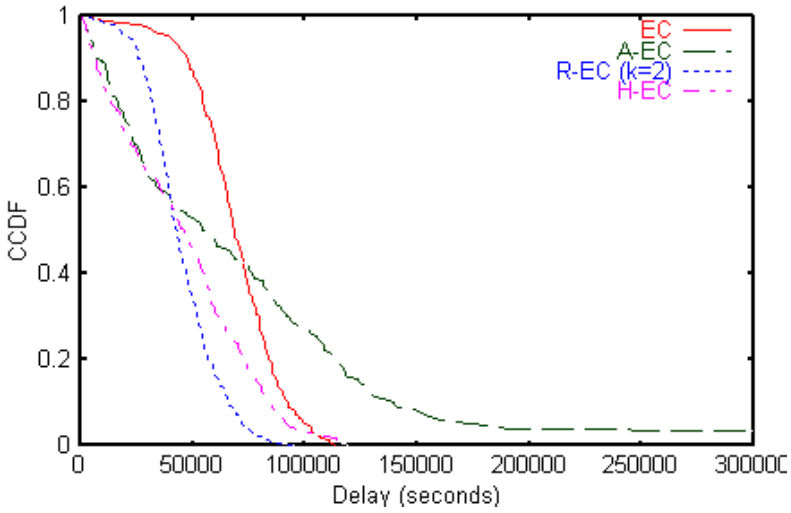


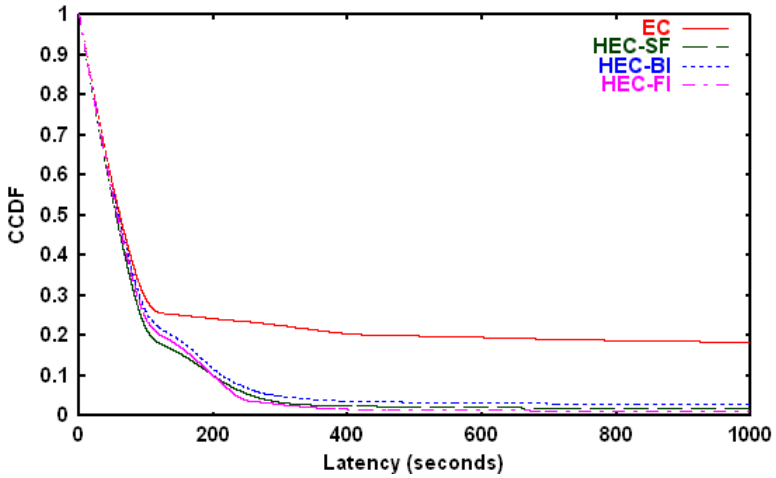
Figure 4.4: Latency distribution (CCDF) of EC, A-EC, R-EC (k=2) and H-EC schemes in *black-hole* network scenarios.

performs slightly worse than R-EC. From these results, we can conclude that H-EC takes better utilization of contact opportunities in the *very small delay performance* cases via aggressive forwarding and also preserves the reliability of EC scheme. Note that H-EC seems more subject to the impact of black-holes than R-EC for the *worst delay performance* cases, although it still outperforms the EC scheme. The reason for above problem is the forwarding strategy of the second phase of H-EC. Here H-EC is HEC-SF, which sends the second copies of coded blocks sequentially. The problem of this strategy is the same as A-EC. It is probable to send all second copies of coded blocks generated from a certain message to a single relay. As we discussed in the previous subsection, this sending strategy is more unreliable. In the next set of evaluations, we will exam the performance of different scheduling algorithms in the aggressive forwarding phase of H-EC.

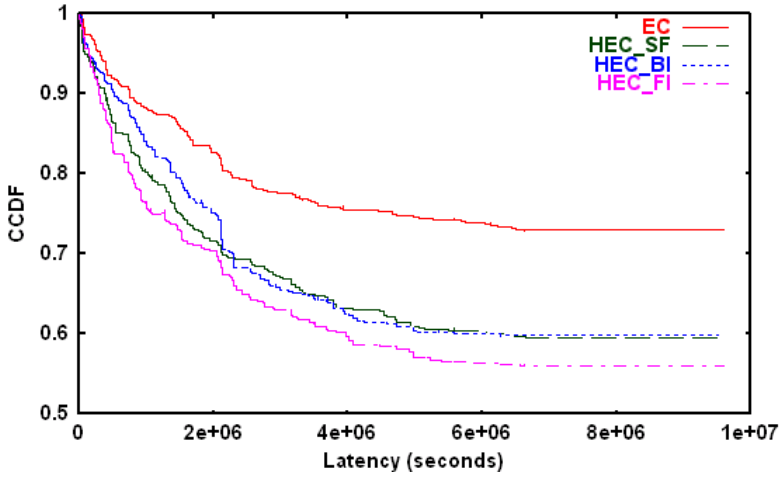
4.2 Data File Transfer

In this subsection, we evaluate the performance of our proposed schemes for data file transfer in opportunistic networks, namely HEC-SF, HEC-FI, and HEC-BI. At the beginning of each simulation run, the randomly selected source generates a huge data file (i.e., 100MB) and forwards to a randomly selected destination. The average data latency distribution results of 200 simulation runs are depicted in Fig. 4.5 by Complementary CDF (CCDF) curves.

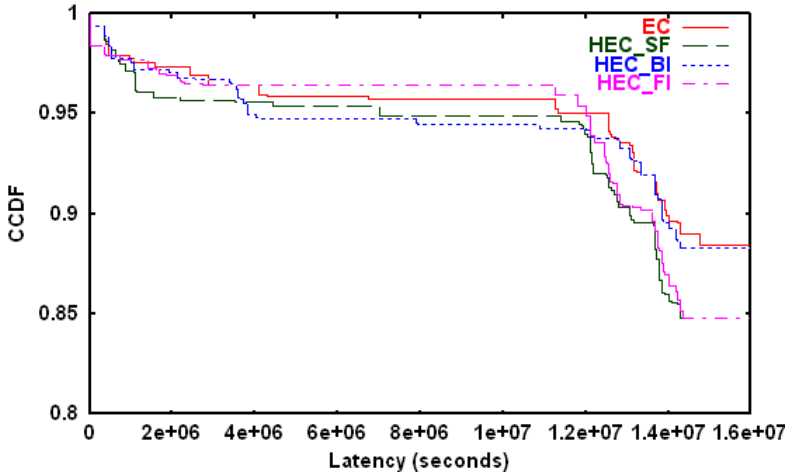
From Fig. 4.5, the H-EC based schemes (i.e., SF, BI, and FI) clearly outperform EC scheme in almost all test cases. The three variants of H-EC perform nearly



(a) Power-Law Scenario



(b) UCSD Scenario



(c) Dartmouth Scenario

Figure 4.5: Distribution (CCDF) of average latency performance of EC, HEC-SF, HEC-FI, and HEC-BI schemes ($N = 16$ and $r = 2$).

identical in the Power-Law scenario. The successfully delivery ratio is more than 95%. By contrast, the EC scheme only reaches to 80%. However, while the network connectivity (i.e., the average number of network contacts per node pair, per day) decreases, the successfully delivery ratio degrades. Compared to the high successful ratio (almost 100%) in the Power-Law scenario, all H-EC based schemes can only achieve about 15% in the Dartmouth scenario and 50% in the UCSD scenario. These results of simulation on real world trace imply that transferring complete data files in an opportunistic network is difficult, or it may require much longer transfer time than the duration employed in our simulation setting. Thus transferring data files in an opportunistic network may not be feasible from an end use's view, unless the receiver has the ability to read *partial* information from an incomplete file. This finding motivates us to investigate transferring video files with the LMDC technique. We will present the evaluation of LMDC-based schemes for video file transfer in the next subsection.

Moreover, the performances of HEC-SF and HEC-BI schemes are similar in all scenarios. Especially in the beginning of simulations, these two schemes perform better than the others (i.e., outstand in the short delay performance cases). The HEC-FI scheme, however, originally performs worse but finally always get best delivery ratio. This is caused by that the HEC-SF and HEC-BI schemes achieve better *contact efficiency*, since they allow the receiver to reconstruct the original message as soon as it encounters and receives from a certain number of coded blocks. In contrast, because the HEC-FI scheme scatters the second copies of coded blocks belonging to an identical message more widely, it can overcome more difficult sce-

narios thus triumphs in complete ratio. Nevertheless, the HEC-FI scheme requires the receiver to wait for at least a few contacts (depending on the erasure coding parameter, r) to collect sufficient coded blocks to reconstruct the message. As a result, HEC-SF and HEC-BI usually perform better or more aggressively in cases with small latency, which represents good network connectivity (i.e., the left portion of the figure) and HEC-FI usually performs better or more resiliently in cases with the worst latency, which represents poor network connectivity (i.e., the right portion in the figure).

4.3 Overhead Analysis

In this subsection, we compare routing overhead of several erasure coding and simple replication based routing approaches. The comparison results are normalized according to Direct Contact (DC) scheme [29]. For simplicity, we only consider optimal cases for A-EC and H-EC schemes (i.e., the first contact duration is long enough to forward all coded blocks). r represents the replication factor of the erasure coding algorithm employed. Erasure code blocks are split among n relays, and k is the number of copies in the replication based algorithms. Table 4.2 shows the comparison results.

In Table 4.2, the overhead of the simple replication based routing algorithm (*srep*) is proportional to the replication factor k, which indicate the number of identical copies injected into the network. On the other hand, EC and A-EC schemes have the same overhead which is proportional to the replication factor r of the

Table 4.2: Comparison of normalized routing overhead for several erasure coding and simple replication based routing schemes.

Algorithms	DC	*srep*	EC	A-EC	R-EC	H-EC
Overall Traffic	1	k	r	r	kr	$2r$
Traffic/Relay/message	1	1	$\frac{r}{n}$	r	$\frac{r}{n}$	r , $\frac{r}{n}$

employed erasure coding algorithm. For R-EC, the overhead is proportional to both r and k, since this scheme is a combination of both EC and *srep*. As to our proposed scheme, H-EC, the traffic overhead is just twice of the EC (or A-EC) scheme, because it is actually a hybrid of EC and A-EC.

Moreover, Table 4.2 also lists the amount of forwarded traffic per relay per message. This value can be interpreted as the *survivability* of a routing algorithm in extremely challenged environments. Larger value means the algorithm puts more responsibility of a message on a certain relay, thus indicates lower survivability of the corresponding algorithm. Note the largest value of A-EC explains why it is subject to black-holes. Because it tends to rely on a small set of relays to forward messages, only small number of uncooperative relays would lead to big problems. In contrast, R-EC and EC schemes distribute a message to more relays and gain more reliability. Finally, H-EC spreads the first copies of coded blocks as EC scheme, so it can preserve the reliability of EC scheme. Even if H-EC sends all its second copies of the coded blocks to a single uncooperative relay, it still has chance to recover messages via the first copies of the coded blocks.

4.4 Video File Transfer

In our simulations for video file transfer, we apply Layered Coding on a 2000-frame video clip using JPEG2000 [3] codec, and then added unequal erasure protection to each video layer in order to realize LMDC encoded on the video. In the simulation, the number of video quality levels, k, and the resulting messages for each frame, N are both set to 10. We investigate the performance of video file transfer with and without LMDC-based coding schemes in opportunistic networks in terms of Peak Signal-to-Noise Ratio (PSNR).

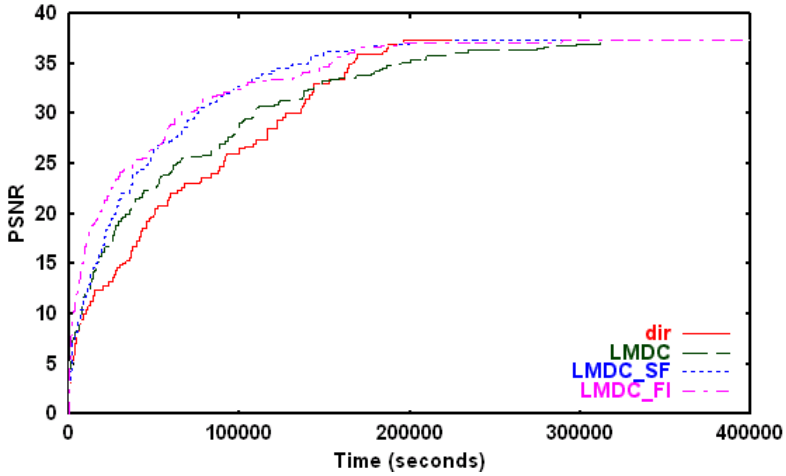
From the simulation of data file transfer in opportunistic networks, we find that receivers are difficult to receive complete files. Only if the receivers can read the incomplete files, transferring in opportunistic networks is practical. LMDC scheme can provide receivers ability to read the video via current data in hand. More data possessed by the receiver means in higher quality the video can be played. Unlike EC scheme that uses just one redundancy level for the whole message, the LMDC (with unequal erasure protection) employs multiple redundancy levels (i.e., the r parameter) for each individual video quality layer.

As mentioned in the LMDC section, the concepts of the HEC-SF/FI/BI algorithms can be applied to the LMDC scheme by sending the second copy of the LMDC messages during the residual contact period as in the HEC-SF/FI/BI, and the resulting algorithms are called LMDC-SF/FI/BI respectively. Actually we didn't implement LMDC-BI in our evaluation since the minimum number of messages required to reconstruct the original quality video is exactly the same as the number

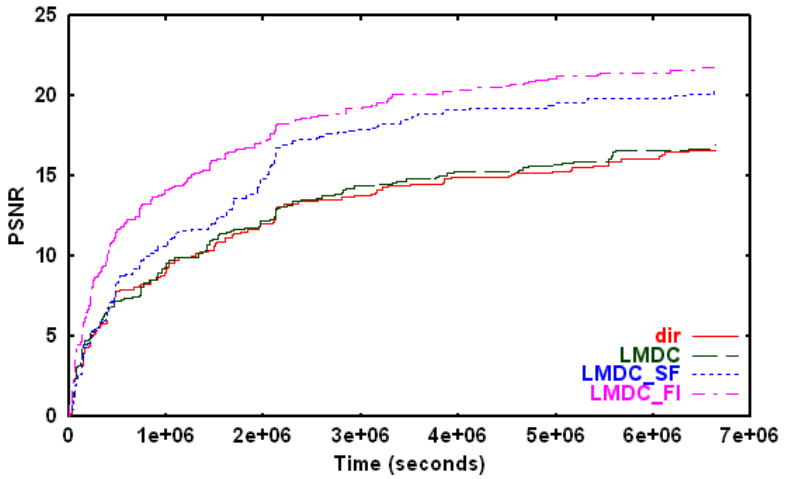
of messages transferred over the network (i.e., $k = N$). When $N > k$, one can derive the LMDC-BI scheme by simply applying the HEC-BI algorithm with the number of blocks (the variable B in the alg. 3) set to k. Fig. 4.6 depicts the average PSNR performance of the 2000-frames video file using different coding and forwarding schemes. We didn't evaluate the Dartmouth scenario since its network connectivity is very sporadic and the data delivery is very poor, as discussed in the previous subsection.

In fig. 4.6, the LMDC based schemes outperform the *dir* scheme which employs the emphdirect contact algorithm [29] to transfer the video file directly without LMDC coding. Furthermore, the LMDC schemes with HEC concepts (i.e., LMDC-SF and LMDC-FI) improve apparently. The LMDC-FI scheme even surpasses the LMDC-SF scheme in the UCSD scenario whose network connectivity is worse than that of the Power-Law scenario. Because the FI-based strategy spreads the second copy of the LMDC messages more widely, each frame quality gets promoted as soon as the receiver receives a message. On the other side, the LMDC-SF scheme operates in a more aggressive way to send the messages belonging to a single frame. But unlike the data file transfer in the EC-based schemes that a data can be reconstructed only when the receiver receives at least a number of coded blocks (depending on the replication factor, r), video files transferred via LMDC schemes can advance in quality every time the receiver receives a message. As a result, the LMDC-FI performs little better than the LMDC-SF in most of time.

In a scenario with good network connectivity like the Power-Law in Fig. 4.6(a), the average video quality of *dir* scheme is finally as good as that of LMDC-SF/FI



(a) Power-Law Scenario



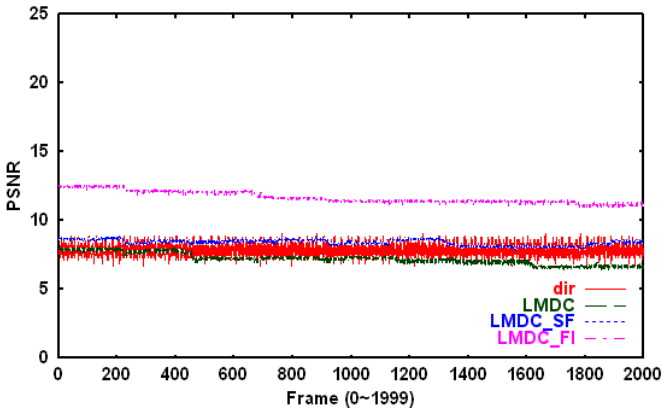
(b) UCSD Scenario

Figure 4.6: Average video quality of LMDC video transfer using direct contact and LMDC-based schemes. (Both N and k are set to 10)

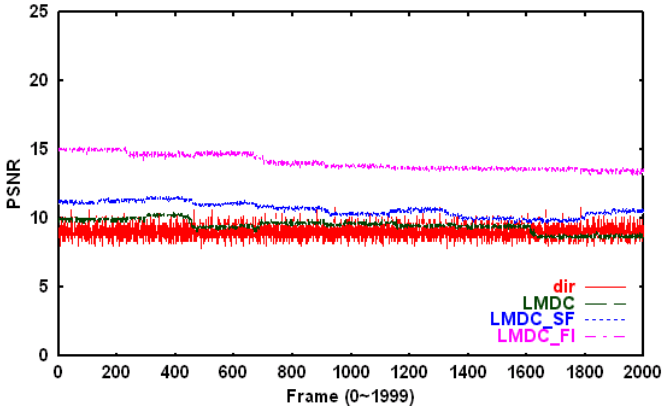
scheme, however, it is worse in the first half of all scenarios (before 200,000 seconds). It means our LMDC-SF/FI scheme can provide better quality in the beginning period of transferring. This is very important in some time-critical applications, for instance, transferring the video of a disaster venue to rescuers. Besides, the pure LMDC scheme performs only slightly better than the *dir* scheme, even worse in some time instances. This result contradicts our intuition about the LMDC scheme's added resilience to error-prone and/or poor-connected networks [10]. The similarity in performance is due to the amount of overhead carried by LMDC (i.e., layered coding and erasure protection). The overhead is too high to achieve a performance gain in extremely challenging network scenarios.

After looking at the overall average PSNR of the video, it is also important to consider the frame-by-frame PSNR performance of video file transfers, since the variance in the frame-by-frame PSNR can make a substantial impact on the perceived video quality for the end user. Fig. 4.7 shows the frame-by-frame PSNR performance of three time points (500,000, 1,000,000, and 5,000,000 seconds) in the UCSD scenario. As well as in the Fig. 4.6, the quality of each frame improves as time goes by.

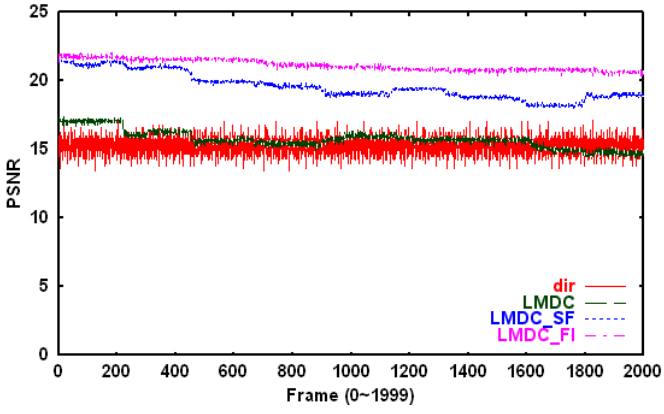
In Fig. 4.7, the LMDC-FI scheme consistently outperforms the other schemes for all time points, while the performance of LMDC-SF is similar to that of LMDC and *dir* schemes in the beginning, but becomes better after 1,000,000 seconds and approximates to LMDC-FI. This conforms to our discussions in Fig. 4.6. Moreover, the PSNR value of the *dir* scheme oscillates severely, whereas the curves of all LMDC-based schemes are much smoother (especially the curve of LMDC-FI



(a) 500,000 seconds



(b) 1,000,000 seconds



(c) 5,000,000 seconds

Figure 4.7: Comparison of average video quality (i.e., PSNR for each video frame) after 500000, 1000000, and 5000000 seconds in the UCSD scenario.

scheme). This is the most important property we want to observe from Fig. 4.7. As noted in [34], drastic PSNR variations are detrimental to the end users' perceived video quality. The smoother line in the figure means the PSNR values of each video frames are closer, which enhances end users' perceived video quality.

Compared with *dir* scheme, the performance of LMDC seems not gaining much improvement; however, one can see that the line of LMDC scheme is much smoother than that of *dir*. This is because, in the *dir* scheme, each video frame is either successfully received or completely lost; there is no intermediate quality video can be played by end users. As a result, the per frame PSNR performances of the *dir* scheme is subjected to large variations. Although the LMDC scheme performs only slightly better than the *dir* scheme, it prevents the video from oscillating. The other LMDC-based schemes (LMDC-SF, LMDC-FI) also have this property. Therefore, based on our observations above, LMDC-based schemes can indeed yield higher frame-by-frame PSNR performance for video transfer in opportunistic networks, as well as providing better perceived quality video to the end users.

Roughly speaking, the PSNR value of all LMDC-based schemes in Fig. 4.7 degrades slightly as the frame number increases. This is because the schemes all send the video frames (or LMDC coded blocks) in sequence, regardless of whether it is in the first regular EC sending phase or in the second aggressive forwarding phase. The problem can be solved by sending video frames in a uniformly random order.

Chapter 5

Related Works

Routing in DTN is a challenged issue. Different from conventional networks, static connections between nodes are unavailable. In these scenarios, round trip time varies severely, leads TCP likes protocols to fail. When the connectivity is predictable (e.g., planetary and satellite movement in Inter-Planetary Networks, IPN [5]), optimal delivery path can be computed by taking into account information about future connectivity [14, 15]. Furthermore, many proposed approaches are involved in reinforcing the connectivity by deploying auxiliary resource [33], for instance, some specialized nodes (e.g., vehicles, robots, satellites, and etc.). These specialized nodes navigate between disconnected nodes or follow a static trajectory to assist in delivering data. However, such approaches are out of our concern. We focus on networks whose connectivity is unpredictable and the contact is "opportunistic". We are also interested in approaches without the effort of reinforcing the connectivity.

Many researchers have been working on the routing algorithms in opportunistic networks. The most straightforward solution is to let the source itself carry the message all the way to the destination (Direct Transmission), or have the aid of a moving relay node (DataMule) [24]. This kind of routing schemes use node mobility (of the source or relay nodes) to assist data transfer [11]. Other single-copy schemes have also been explored in [26].

Using only one copy to transfer in the opportunistic network is rather insufficient. The delivery delay might extremely large. Thus multiple-copy schemes would send more copies to increase the successful probability as well to diminish the delivery delay. Currently, most popular design choice for opportunistic routing scheme is replication. For instance, the *Epidemic Routing* scheme, a fast way to perform routing in opportunistic network, is to flood the message throughout the network [28]. This scheme is guaranteed to find the optimal path when there is no contention for shared resource. But in realistic scenarios where bandwidth, node energy, or buffer size are usually bounded, flooding a network with duplicate data is very costly in terms of traffic overhead and energy consumption [26, 27].

To reduce the amount of overhead occurring with flooding, many approaches have been proposed [27, 16, 32, 25]. A *Controlled Flooding* scheme which is proposed in [12] employs three parameters (*willingness probability*, *Time-to-Live*, and *Kill Time*) to suppress redundant transmissions and clean up valuable buffer space after a message has been successfully delivered to the receiver. Instead of flooding, some approaches transfer message to another node with a probability smaller than one. In this kind of schemes, messages are "gossiped" rather than flooded.

A custodian doesn't broadcast to its neighboring nodes but transfer according to some conditions. For instance, the *Probabilistic Routing* scheme [20] calculates the *delivery predictability* from a node to a particular destination node based on the observed contact history. A message is then forwarded to a neighboring node if and only if the neighboring node has a higher *delivery predictability* value. In addition to contact history, Leguay et al [17] further take the *mobility pattern* into account. Namely, a message is forwarded to a node if and only if the *mobility pattern* of that node is more similar to that of the destination. [17, 18] shows that *mobility pattern* scheme is more effective than previous schemes which only consider the history of past encounters. Anyway, the concept of history-based or utility-based routing is trying to make fewer and more efficient forwarding based on more information.

A new class of opportunistic network routing schemes is based on coding technique. This kind of schemes transforms a message into a different format before transferring into networks. Widmer et al [30] proposed a scheme combining *network coding* and epidemic routing techniques to reduce the required number of transmissions in a network. Another integration of *erasure coding* and the simple replication-based routing techniques proposed in [29] improves data delivery for the worst delay performance cases in opportunistic networks. Following the concept of erasure coding-based data forwarding, an Estimation based Erasure-Coding routing scheme (EBEC) has been proposed to adapt the delivery of erasure coded blocks by estimating the Average Contact Frequency (ACF) [19].

Chapter 6

Conclusion

Although routing in opportunistic networks is hard, our proposed schemes can make it practical. Because a path between the source and the destination does not always exists, finding an optimal path in a connected graph just as conventional routing protocols does not work. Our proposed approaches efficiently utilize intermittent contacts and raises delivery ratio in an acceptable delay as high as possible.

Specifically, the evaluation shows our proposed schemes indeed work. The H-EC scheme both takes the benefit of aggressive forwarding and keeps the advantages of EC scheme. But we also find that it is difficult to receive complete data. By using LMDC, end users can "preview" the incomplete video data in hand and have a better perceived quality. We analyze overheads of our schemes and show that they are simply low. The effectiveness and robustness of our schemes prove themselves as ideal solutions for data and video file transfer in opportunistic networks.

We further publish our implementations in our web site.

http://nrl.iis.sinica.edu.tw/DTN/

One can immediately deploy our algorithms on their testbed or applications. We publish HEC-SF/FI/BI and LMDC-SF/FI codes for DTNSim. We also implement a simple codec applying unequal erasure protection on a layer coded frame. By this codec, one can encode a layer coded frame into any number of messages and recover the original frame by any subset of the generated messages. It demonstrates the LMDC scheme proposed in the Chapter 3.

Bibliography

[1] Crawdad project. http://crawdad.cs.dartmouth.edu/.

[2] Delay tolerant network simulator. http://www.dtnrg.org/code/dtnsim.tgz.

[3] Jpeg 2000. http://www.jpeg.org/jpeg2000/.

[4] Ucsd wireless topology discovery project. http://sysnet.ucsd.edu/wtd/.

[5] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss. Delay-tolerant networking: an approach to interplanetary internet. *IEEE Communications Magazine*, 41(6):128–136, 2003.

[6] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding. Technical Report UCAM-CL-TR-617, University of Cambridge, Computer Laboratory, February 2005.

[7] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on the design of opportunistic forwarding algorithms. In *IEEE Infocom*, 2006.

[8] P. A. Chou, A. E. Mohr, S. Mehrotra, and A. Wang. Error control for receiver-driven layered multicast of audio and video. *IEEE Transactions on Multimedia*, 3(1):108–122, March 2001.

[9] P. A. Chou and K. Ramchandran. Clustering source/channel rate allocations for receiver-driven multicast under a limited number of streams. In *IEEE ICME*, 2000.

[10] P. A. Chou, H. J. Wang, and V. N. Padmanabhan. Layered multiple description coding. In *IEEE Packet Video Workshop*, 2003.

[11] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. Netw.*, 10(4):477–486, 2002.

[12] K. A. Harras, K. C. Almeroth, and E. M. Belding-Royer. Delay tolerant mobile networks (dtmns): Controlled flooding in sparse mobile networks. In *IFIP Networking*, 2005.

[13] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *ACM SIGCOMM Workshop on DTN*, 2005.

[14] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *ACM SIGCOMM*, 2004.

[15] E. P. C. Jones, L. Li, and P. A. S. Ward. Practical routing in delay-tolerant networks. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[16] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *ASPLOS, San Jose, CA*, Oct. 2002.

[17] J. Leguay, T. Friedman, and V. Conan. Dtn routing in a mobility pattern space. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[18] J. Leguay, T. Friedman, and V. Conan. Evaluating mobility pattern space routing. In *Proc. IEEE Infocom*, 2006.

[19] Y. Liao, K. Tan, Z. Zhang, and L. Gao. Estimation based erasure-coding routing in delay tolerant networks. In *IWCMC '06: Proceeding of the 2006 international*

conference on Communications and mobile computing, pages 557–562, New York, NY, USA, 2006. ACM Press.

[20] A. Lindgren, A. Doria, and O. Schelen. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(3):19–20, July 2003.

[21] S. R. McCanne. *Scalable Compression and Transmission of Internet Multicast Video*. PhD thesis, University of California, Berkeley, 1996.

[22] V. N. Padmanabhan, H. J. Wang, and P. A. Chou. Resilient peer-to-peer streaming. In *IEEE ICNP*, 2003.

[23] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai. Distributing streaming media content using cooperative networking. In *IEEE NOSSDAV*, 2002.

[24] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: modeling and analysis of a three-tier architecture for sparse sensor networks. *Ad Hoc Networks*, 1(2-3):215–233, 2003.

[25] T. Small and Z. J. Haas. Resource and performance tradeoffs in delay-tolerant wireless networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-Tolerant Networking*, pages 260–267, New York, NY, USA, 2005. ACM Press.

[26] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Single-copy routing in intermittently connected mobile networks. In *IEEE SECON*, 2004.

[27] Y.-C. Tseng, S.-Y. Ni, Y.-S. Chen, and J.-P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Wirel. Netw.*, 8(2/3):153–167, 2002.

[28] A. Vahdat and D. Becker. Epidemic routing for partially-connected ad hoc networks. Technical Report CS-2000-06, Duke University, 2000.

[29] Y. Wang, S. Jain, M. Martonosi, and K. Fall. Erasure coding based routing for

opportunistic networks. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[30] J. Widmer and J.-Y. L. Boudec. Network coding for efficient communication in extreme networks. In *ACM SIGCOMM Workshop on Delay Tolerant Networks*, 2005.

[31] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi. Hardware design experiences in zebranet. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 227–238, New York, NY, USA, 2004. ACM Press.

[32] X. Zhang, G. Neglia, J. Kurose, and D. Towsley. Performance modeling of epidemic routing. *Comput. Networks*, 51(10):2867–2891, 2007.

[33] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *ACM MobiHoc*, 2004.

[34] M. Zink, O. Künzel, J. Schmitt, and R. Steinmetz. Subjective impression of variations in layer encoded videos. In *IWQoS*, 2003.

Appendix A

H-EC Codes for DTNSim

Our H-EC codes for DTNSim can be download from

http://nrl.iis.sinica.edu.tw/DTN/download/

One can employ HEC-SF/FI/BI algorithms on their simulation immediately. The Readme.txt included in the package describes how to install and run them. Note that before installing, you are recommended to backup all you files.

There are totally fifteen files. Copy *HelpFunctions.java* to your dtnsim/src/util/ directory. And copy *CbrAgent.java*, *ChewedMessages_ec.java*, *EraCodedBlk.java*, *Event.java, Main.java*, *Message.java*, *Node.java*, *SimpleNode_ec.java*, *SimpleNode_hec_sf.java*, *SimpleNode_hec_fi.java*, *SimpleNode_hec_bi.java*, *Stats.java*, *Topology.java*, and *TrAgent.java* to your dtnsim/src/simulator/.

To use H-EC algorithm, you should add one more command in your simulation script. The H-EC command is as follows:

Time HEC [SF|BI|FI] replication-factor #-of-relay

The first entry is the time at which this command will be executed. Here we usually set to 0. The second entry tells the simulator to use H-EC protocol. The third entry indicates the algorithm used to forward the second copy of H-EC. SF stands for Sequential Forwarding, FI for Full Interleaving, and BI for Block-based Interleaving. The fourth entry is the replication factor of erasure coding. Finally, the last entry is the number of relay to split the coded blocks. For example,

0 HEC SF 2 16

As usual, the time in above command is set to 0. The replication factor of erasure coding is 2. The first copies of the erasure coded blocks are sent to 16 different relays while the second copies are sent by using sequential forwarding. For more detail about the script commands, please refer to the Main.java.

Appendix B

LMDC Codes

We also publish our LMDC codes for DTNSim on the web page

http://nrl.iis.sinica.edu.tw/DTN/download/

Before installing, it would be better to backup up all your files first. In addition, you have to install H-EC codes before installing LMDC codes. Copy the six files in the LMDC package: *Main.java*, *Event.java*, *Topology.java*, *LMDCAgent.java*, *AppAgent.java*, and *Constants.java* to your dtnsim/src/simulator/ directory. Go to dtnsim/src/ directory and execute compile.sh. Go to dtnsim/test/gnuplot/ and make a directory named LMDC.

To run LMDC simulation, set the algorithm what you want. For example:

0 set -cost lmdc

The algorithms related to LMDC are:

1. lmdc: plain Layered Multiple Description Coding
2. lmdc_sf: LMDC with Sequential Forwarding
3. lmdc_fi: LMDC with Full Interleaving

Besides, the traffic agent now is different. The command to use LMDC agent is as below:

[*time*] application_agent [*LMDC type*] [*source*] [*destination*] [*port*] [*# of description*] [*# of frames*] [*message size*]

The string between [and] is a parameter. For example:

100 application_agent LMDC_FI 1 2 1234 10 1000 0.009

The message size is calculated off-line. You can observe the receiving stats at any time instant. By adding the below command:

[*time*] mdc_report

The variable [*time*] is replaced by any time instant after simulation beginning.

The simulation result are placed in dtnsim/test/gnuplot/LMDC/. The file named *.dat is the time to average PSNR of the video. The file whose name consists of time point is the PSNR of each frames of the video in the time instant set in mdc_report command.

Note that in the private class FRAME of LMDCAgent.java, there is a function named psnr() which returns the psnr of this frame. For now, the psnr value of

different layers are calculated off-line. You are expected to modify this portion to fit your simulation.

Appendix C

LMDC Codec

We further implemented a simple codec which apply unequal erasure protection on a layer coded frame. The download page is:

http://nrl.iis.sinica.edu.tw/DTN/download/

We implement the encoder and the decoder with PYTHON. The file *unequal_era.py* is the encoder, while *decoder.py* is the decoder. Another four files: *ffield.py*, *file_ecc.py*, *genericmatric.py*, and the *rs_code.py* are the libraries we used.

The usage of the encoder is:

python unequal_era.py

You will be asked to enter the number of layers and the source file that you want to encode. The source file is expected to be a layer coded frame. Then the encoder will generate messages which amount to the number of layers. These generated

messages are put in the directory named *generated*. In addition, it will further generate a file called *offset_log.txt*. This file is used as an input for the decoder to recover the original file.

The usage of the decoder is:

python decoder.py [*received message(s)*]

The arguments *received message(s)* indicates the received messages used to recover the original file. Each message name is separated by a blank. Note that the file "offset_log.txt" generated by the encoder should be put in the same directory where you run the decoder. You will be asked to enter the number of layers. The recovered file is called "recovered_frame" and put in the same directory as the decoder.