Performance of an Efficient Scheduling Approach to Network Coding For Wireless Local Repair

Juma Ben Saleh, Dongyu Qiu, and Ahmed K. Elhakeem

Abstract—We propose a new XOR based scheduling algorithm for network coding in cooperative local repair. The algorithm makes use of knowledge of the packets availability at neighboring nodes to improve the overall network throughput. In our proposed algorithm, we use network coding to determine which node should transmit and in which time slot (sequential MAC) that would provide the best improvement.

The proposed algorithm proceeds in three phases. First, the nodes exchange their packet's availability vectors. This is followed by a short period of distributed scheduling, during which the nodes execute the processing algorithm, developed to minimize the total transmission time. In the third phase, nodes transmit the encoded packets as per the decision of the scheduling algorithm. The upper bound on the improvement factor is also derived. In addition, we study the effects and trade-offs of file sizes, processing delays, number of users and packet availabilities. In the sequel, we show the favorable effects of file segmentation. Simulation results show improvement in the system throughput and in the processing delay of the proposed algorithm. These results also show that the improvement factor of the proposed scheduling algorithm is close to the upper bound.

***Index Terms*—Scheduling, Network Coding, Local Repair**

I. Introduction

DATA broadcast is a convenient way of one-to-many transmissions. Occasional data loss may occur due to channel impairments, etc. Forward Error Correction (FEC) [1] and Automatic Repeat Request (ARQ) [2] techniques have been utilized extensively to mitigate channel effects. Recently, network coding (NC) has been proposed [3], [4] to improve the performance of FEC and ARQ techniques. Nodes cooperation, or local repair, is another way to overcome the channel effects [5]. In local repair, the base station (BS) broadcasts the intended file to all nodes in its range. Due to channel effects, some packets of the transmitted file are probably missed at some nodes. The nodes will cooperate with each other to deliver the missing packets to each node. Local repair is a good choice in scenarios where retransmissions cannot solve problems such as poor channel quality between the BS and the subscribers. Local repair is also preferable in scenarios where a set of nodes with multiple network interfaces (i.e. Wireless Wide Area Network (WWAN) and Wireless Local Area Network (WLAN)) are looking for data transmitted by the source node in WWAN [6].

NC was initiated by Ahlsweede [7], who showed that the maximum capacity in a network can be achieved by the appropriate mixing of data in the intermediate nodes. Many studies in NC have proved that applying NC to traditional networks could provide significant improvement in overall system throughput [6-23]. Most of the studies in NC for cooperative local repair use random linear network coding (RNC) [8] to construct coded packets [6], [10-13]**.** In RNC, each intermediate node chooses its coefficients randomly and independently from a finite field. The received packets are weighted according to the chosen coefficients to create the transmitted coded packet. These coefficients are transmitted as a header in the coded packet for decoding purposes. For large files, the header size becomes significant, which wastes bandwidth. In addition, the distention node will not be able to decode the coded packet until it receives a certain number of different coded packets. These are the two most challenging problems caused by using RNC. In our algorithm, we use XOR based NC instead of RNC. In XOR NC, users utilize the availability of some packets at some nodes and their unavailability at other nodes to combine some packets and transmit the combined packet in a single transmission, which would be decodable at most nodes. The basic concept of XOR NC is that one node may combine a number of packets in its possession by XOR or use a modulo two operation of corresponding bits of the combined packets. With the proposed scheduling algorithm, the algorithm selects a set of nodes for local repair transmissions that will minimize the total required number of transmissions. Also, a node that has been selected to transmit a coded packet will be able to know its transmission time slots (sequential *MAC* determination). Furthermore, all nodes that can hear each other will be able to know the constituting packets of all the coded packets (no need to transmit any additional information to the received node about which packets were XORed together).

The outline of the paper is as follows. Related work is presented in section II, and scheduling algorithm overview is in section III. In section IV, the scheduling algorithm and

Manuscript received January 10, 2011.

Juma Ben Saleh is now a PhD candidate at Concordia University. His research interests include network coding, CDMA and cross-layer design (e-mail: ju_bens@ encs.concordia.ca).

Dongyu Qiu is currently an Assistant Professor in the Department of Electrical and Computer Engineering, Concordia University, Montreal, Quebec, Canada. His research interests are in the areas of peer-to-peer networks, TCP/IP networks, network congestion control, queuing analysis, network security, and wireless networks (e-mail: dongyu@ece.concordia.ca).

Ahmed K. Elhakeem is a Professor in the Electrical and Computer Engineering Department, Concordia University, Montreal, Quebec, Canada (e-mail: ahmed@ece.concordia.ca).

MAC procedure of the algorithm are presented, and the upper bound on the improvement factor of the proposed algorithm is presented in section V. Numerical and simulation results are presented in section VI. The conclusion is in section VII.

II. RELATED WORK

In the literature, relatively few works on NC for cooperative local repair are presented. In [10], two heuristic algorithms were designed. The first is *centralized network coding for cooperative peer-to-peer local repair* (NC-CCPR), where all nodes are assumed to have accurate information about their one-hop and two-hop neighbors. The second algorithm is *distributed network coding for cooperative peer-to-peer local repair* (NC-DCPR). The problem with this algorithm is the header size when there are large files. A structured NC is presented in [11] and [12], which is based on the concept that there are some zero coefficients of coded packets, so that, decoding is possible even when the number of received packets is small. The problem in this algorithm is the assumption that all nodes in the same ad-hoc network watch the same video. In [6], authors argue that this assumption is not practical and propose a new algorithm. The authors of [13] propose a hierarchical NC scheme wherein packets in a video stream are coded with different levels of importance. Thus, even when a small number of coded packets are received, a receiver will be able to recover the most important packets. Most of the works on NC for cooperative local repair use RNC and try to improve the data quality. Although most authors try to reduce the downsides of the RNC such as the added NC header and the ability of a receiver to decode a coded packet, the effects of RNC is still there. In our proposed algorithm, we use NC to determine which node should transmit and in which time slot (sequential MAC) that would provide the best improvement.

III. SCHEDULING ALGORITHM OVERVIEW

In this work we restrict the discussion to those algorithms that involve a selected BS. Presence of BS or some control station is justified in many applications like cellular systems even in some mesh networks. Most Previous works assume a BS (Wireless Wide Area Network (WWAN)) transmits a data file to nodes, and then these nodes use Wireless Local Area Network (WLAN) for local repair [6].

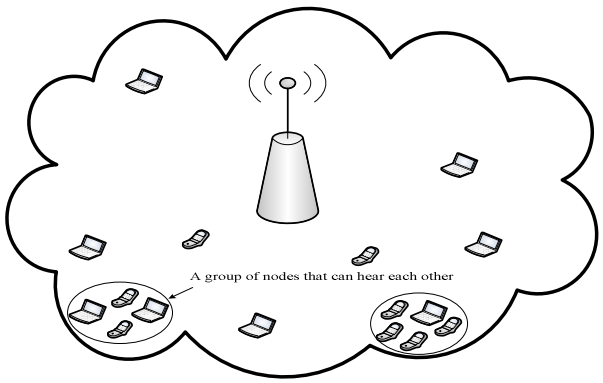

Figure 1 Network topology.

The process is initiated by the BS transmitting a file of W packets; the repair process will commence once each node receives reception reports from the other nodes. The IEEE 802.11 protocol for *MAC* channel access is utilized by all the nodes to relay the reception reports. Following this stage (reception reports transmissions), various nodes will work independently but use the same scheduling algorithm to determine the best packet combinations that will minimize the number of transmissions and thereby maximize the system throughput. The BS keeps synchronization by means of pilot signals.

Table 1 Information table at each node describing the received and missed packets of all users

Packet identity / Nodes identities	1	2	3	4	5	6	7	8	9	10
1	1	0	1	0	1	1	1	1	1	0
2	0	1	1	1	0	1	1	0	0	0
3	1	1	0	0	1	0	1	1	1	0
4	1	1	1	1	0	1	1	0	1	1

This pilot will trigger the transmissions of the scheduled combined packets one after another. The BS will allow enough time for the distributed algorithm to finish. Some nodes may take slightly less time to execute the distributed algorithm, but they will have to wait for the starting pilot of the BS as indicated above. The algorithm will also determine the identities of the nodes that will transmit the combined packets and their transmission times (sequential *MAC* determination). Thus, we do not transmit any extra data for coefficient description or any information about which packets have been combined together. As an example, assume that in Table 1, the number of nodes that can hear each other is 4 and that the file size is 10 packets. After receipt of the reception reports' transmissions, each node will have a table describing the state of all the nodes, including itself, which we call an *information table* (Table 1). Observe that in Table 1, a 'one' means that a node has received the packet correctly and a 'zero' means a node did not receive that packet. Following reception of all the reports, every node will be able to configure the information table. From Table 1, node 4 will be able to combine packets 1, 2 and 3 (1's in all the combined packets means the node has initially received those packets). Nodes 1, 2 and 3 will receive such a packet and by further XORing of the received packet, node 1, for example, will be able to find the missing bits of packet 2. Similarly, node 2 will further XOR the received combined packet with bits of packets 2 and 3 and in the process obtain bits of the missing packet 1.

We present a new XOR based scheduling algorithm for NC in cooperative local wireless repair where all nodes can hear each other, as shown in Fig. 1. We also present the associated sequential *MAC* transmission technique that results from the new cooperative repair algorithm.

IV. THE SCHEDULING ALGORITHM AND MAC PROCEDURE

The scheduling algorithm presented here was designed to benefit the maximum number of nodes. The algorithm first tries to combine the maximum number of packets that can aid the maximum number of nodes. The flow chart of the scheduling algorithm is shown in Fig. 2. We assume a sub network of N nodes that can hear each other directly (nodes within the small circuits in Fig. 1), each node s where ($s = 1, 2, 3, .., N$)has received a set of packets of the file transmitted by the BS. Obviously, the packets that are received by all N nodes will not be candidates for transmission in local repair and will be discarded from the repair window (point A on the flow chart). The window length W is thus updated by excluding the packets received by all N nodes. For example, from Table 1, packet 7 will be excluded. Also, packets that are received by only one node in the sub network, such as packet 10 in Table 1, will not be considered for NC combination. For those packets, single broadcast transmissions are more efficient since NC cannot provide any improvement (point B in the flow chart).

Table 2 Notation

N	number of nodes that can hear each other
W	is the file size
$(N-i)$	number of original packets constituting a coded packet (combination level)
$(N-j)$	number of nodes that will benefit from the coded packet
$\bar{d}_s$	a vector of the received packets identities at node s
$\bar{n}_{m,i}$	is the m^{th} set of nodes at the $(N-i)$ combination level
$\bar{l}_{m,i}$	the identities of the commonly packets received in set m
$\bar{k}_i$	is a vector of packets identities that will be considered at the $(N-i)$ combination level
$\lvert\bar{k}_i\rvert$	number of packets in $\bar{k}_i$
$\bar{C}_{s,i}$	the identities of packets received by node s that intersect with $\bar{k}_i$
$\lvert\bar{C}_{s,i}\rvert$	number of packets in $\bar{C}_{s,i}$

We denote $\bar{d}_s$ as a vector of the received packets identities at node s in the sub network, excluding the packets received by all nodes and packets received by one node only. For instance, from Table 1, $\bar{d}_3 = [1\ 2\ 5\ 8\ 9]$. Let $(N-i)$ be the *combination level* where $(i=1, 2, ...,N-2)$ (point F on the flow chart). If a node tries to combine (by XOR) the maximum number of packets, which is $(N-1)$ packets, this implies $(i=1)$. We denote the number of nodes that will benefit from the coded packet as $(N-j)$ nodes, where $(j = 1, 2, ..., N-2)$ (point E on the flow chart).

First, the scheduling algorithm tries to find combinations that would favor the maximum number of nodes ($j = 1$). The scheduling algorithm commences with the highest combination level ($i = 1$) where one combined packet will help as many nodes as possible. If there is no combination at this level that could help $(N-1)$ nodes, the scheduling algorithm decrements the combination level by one (*i becomes 2*) and tries to find a combination of packets at any node that will help $(N-1)$ nodes, and so on for all $(i \geq j)$. Motivated by efficiency considerations, we maintain that $(N-i) \leq (N-j)$, i.e. the combination level is less than or equal to the number of helped nodes. To do the contrary would be a waste of bandwidth since a higher combination level should not be used to help fewer nodes. All possible combinations of packets that will help $(N-1)$ nodes at any combination level, by any node, will be scheduled for transmission and the repairing window adjusted accordingly (point Z in the flow chart).

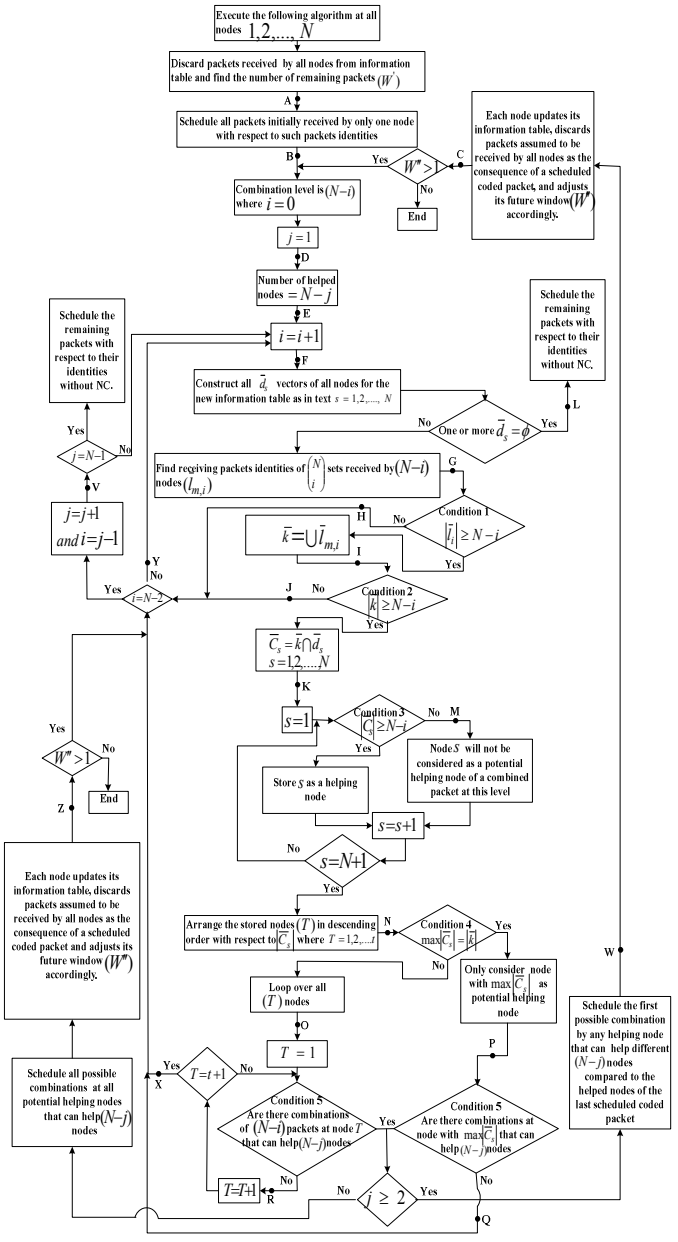

Figure 2 Flow chart of the scheduling algorithm.

If the scheduling algorithm schedules all the possible packet combinations at all the potential helping nodes, at all combination levels that can help ($N - 1$) nodes and there are still some packets that have not been scheduled (as evidenced by $W'' > 1$ in the flow chart), the scheduling algorithm increments j to 2 (point V on the flow chart) and searches for combination that will help ($N - 2$) nodes commencing from the highest combination level**.** The highest combination level for $j = 2$ is ($N - 2$) packets, since XORing ($N - 1$) packets to help ($N - 2$) nodes is inefficient in terms of process and bandwidth. The first combination that will help ($N - 2$) nodes, discovered at any combination level, will be scheduled for transmission and the repairing window adjusted accordingly (point C in the flow chart).

After scheduling a combination that can help ($N - j$) nodes ($j \geq 2$), the algorithm assumes that all the ($N - j$) nodes have received such a combined packet, and XOR decodes the packet and adjusts their $\bar{d}_s$ accordingly (point C in the flow chart). Then the scheduling algorithm tries to find other combinations that will help ($N - 1$) nodes, starting from the highest combination level. After each scheduling that helps ($N - 2$) or fewer nodes, the scheduling algorithm loops again to search for combinations that can help ($N - 1$) nodes (point D in the flow chart). If there is no combination that can help ($N - 1$) nodes after scheduling a coded packet that can help ($N - 2$) or fewer nodes, the algorithm will schedule the first combination that can help other ($N - 2$) or fewer nodes, compared to the nodes that were helped by the last scheduled coded packet (point W in the flow chart). If the scheduling algorithm schedules all the possible combinations that will help ($N - 1$) nodes and there is no possible combination that can help ($N - 2$) nodes, and there are still some packets that have not been scheduled, the algorithm searches for a combination that will help ($N - 3$) nodes, i.e. $j = 3$, starting from the highest combination level which is ($N - 3$) packets, i.e. $i = 3$ in this case, and so on. A node that has been selected to create a combined packet formed from the XORing of ($N - i$) packets will include in this XOR operation only those packets commonly received by at least ($N - i$) nodes, because the number of helped nodes ($N - j$) is always greater than or equal to the combination level ($N - i$). The main steps of the proposed scheduling algorithm can be presented as follows:

Step1: The algorithm first will find the involved packets to create the coded packets at the current combination level. We have $\binom{N}{i}$ different possible sets of nodes ($\bar{n}_{m,i}$) where the m^{th} set has ($N - i$) nodes where $m = 1, 2, \ldots, \binom{N}{i}$. As an example, if $i = 1$ in Table 1, then we have $\binom{4}{1} = 4$ sets where each set is composed of three nodes, as follows:

$\bar{n}_{1,1}$ = {node1, node2, node3}, $\bar{n}_{2,1}$ = {node1, node2, node4}

$\bar{n}_{3,1}$ = {node1, node3, node4}, $\bar{n}_{4,1}$ = {node2, node3, node4}

Then the algorithm finds the identities of the packets commonly received by all nodes in each set of nodes. Let $\bar{l}_{m,i}$ denote the identities of the commonly packets received in each set. For instance, in Table 1, for $i = 1$, $\bar{l}_{1,1} = \phi$, $\bar{l}_{2,1}$ = {3, 6}, $\bar{l}_{3,1}$ = {1, 9}, and $\bar{l}_{4,1}$ = {2}. Only these packets are involved to be XORed together to create the coded packets at this combination level which in turn minimizes the total transmission time of the subsequent repair process.

Step 2: Denote $|\bar{l}_i|$ as the number of nonempty vectors of all the $\bar{l}_{m,i}$ vectors. The algorithm checks if $|\bar{l}_i|$ is less than ($N - i$). If $|\bar{l}_i| < (N - i)$, then a packet combination at this level that helps ($N - j$) nodes is not possible, since the algorithm picks only one packet from each set to create a coded packet. The algorithm decrements the combination level by one: $i = i + 1$. Thus, the algorithm will not proceed at this combination level since it knows, from the value of $|\bar{l}_i|$, that it is not possible to create a coded packet at this combination level that can help ($N - j$) nodes. From Table 1, for $i = 1$, $|\bar{l}_i|$ is three sets which is equal to ($N - i$), and so the algorithm will proceed to try to create coded packets at this combination level.

Step 3: If $|\bar{l}_i| \geq (N - i)$, we denote $\bar{k}_i$ as the union set of all the $\bar{l}_{m,i}$ sets of packets identities.

$$\bar{k}_i = \bigcup_{m=1}^{\binom{N}{i}} \bar{l}_{m,i} \qquad (1)$$

$\bar{k}_i$ is actually a vector of packets identities that will be considered at the ($N - i$) combination level for the information table at given i (details to follow). For example, from Table 1, for $i = 1$, $\bar{k}_i$ = {1, 2, 3, 6, 9}. Denote $|\bar{k}_i|$ as the number of packets identities of vector $\bar{k}_i$. If $|\bar{k}_i| < N - i$, then it is not possible that a packet combination at this combination level can help ($N - j$) nodes. Therefore, the algorithm will decrement the combination level by one, to become ($N - i - 1$), and try for the new combination level.

Step 4: If $|\bar{k}_i| \geq N - i$ then we do find the identities of packets received by each node s that intersect with $\bar{k}_i$.

$$\bar{C}_{s,i} = \bar{k}_i \cap \bar{d}_s \qquad (2)$$

For example, from Table 1, for $i = 1$, $\overline{C}_{1,1} = \{1,3,6,9\}$, $\overline{C}_{2,1} = \{2,3,6\}$, $\overline{C}_{3,1} = \{1,2,9\}$, and $\overline{C}_{4,1} = \{1,2,3,6,9\}$. Denote $|\overline{C}_{s,i}|$ as the number of packets identities of vector $\overline{C}_{s,i}$. If $|\overline{C}_{s,i}| < N - i$, then the s^{th} node cannot combine $(N - i)$ packets that would help $(N - j)$ nodes. Nodes with $|\overline{C}_{s,i}| \geq N - i$ are sorted in descending order with respect to $|\overline{C}_{s,i}|$. For example, from Table 1, for $i = 1$, $|\overline{C}_{1,1}| = 4$, $|\overline{C}_{2,1}| = 3$, $|\overline{C}_{3,1}| = 3$ and $|\overline{C}_{4,1}| = 5$. Thus, all nodes can create coded packets. Consequently, all nodes will be considered as possible helping nodes with respect to their $|\overline{C}_{s,i}|$. Therefore, the node with the maximum $|\overline{C}_{s,i}|$ will have the highest chance to be scheduled for transmission, in order to help other nodes by its combined packet transmission.

Step 5: At the given number of nodes to be helped $(N - j)$, and the possible combination level of $(N - i)$ packets, if $max(|\overline{C}_{s,i}|) = |\overline{k}_i|$ we only consider the node with $max(|\overline{C}_{s,i}|)$ as a helping node and proceed to try to find combined packets at level $(N - i)$ as will follow later. If no $(N - j)$ nodes can benefit from the selected helping node, the algorithm will go to a lower combining level $(N - i - 1)$ without considering any other possible combined packets from any other node. The reasoning is that this node has all of the $|\overline{k}_i|$ packets that can help, and so if the combined packets of such a node cannot help $(N - j)$ nodes, no other nodes will either. For example, from Table 1, for $i = 1$, $max(|\overline{C}_{s,i}|) = |\overline{C}_{4,1}| = 5 = |\overline{k}_i|$ and as a result, only node 4 will be considered as a helping node at this combination level. However, if the $max(|\overline{C}_{s,i}|) < |\overline{k}_i|$, all nodes with $|\overline{C}_{s,i}| \geq N - i$ will be considered as helping nodes. The algorithm investigates the possibility of forming combined packets from the node with the highest $|\overline{C}_{s,i}|$ that can help $(N - j)$ nodes, and then it goes to the node with the next-highest $|\overline{C}_{s,i}|$. If, after investigating all the nodes with $|\overline{C}_{s,i}| \geq N - i$ no combined packets from any node yields benefits to $(N - j)$ nodes, then the algorithm reverts to the next lower combination level.

Step 6: Let $\overline{U}$ be the set of the identities of potentially transmitting helping nodes s, whose $|\overline{C}_{s,i}| \geq N - i$ have been sorted in descending order with respect to the $|\overline{C}_{s,i}|$ values. Denote the transmitting node under consideration to combine $(N - i)$ packets as node x. Node x will pick $(N - i)$ packets whose identities are part of $\overline{C}_{x,i}$. These packets are arranged in ascending order with respect to their identities, and the scheduling algorithm will check if these packets can help $(N - j)$ nodes. For $(j = 1)$, if the combined packet can help $(N - i)$ nodes, this specific coded packet will be scheduled and the algorithm will pick other different $(N - i)$ packets from the same helping node whose identities are part of $\overline{C}_{x,i}$, and so on. However, if the combined packet cannot help $(N - i)$ nodes, the algorithm will pick other $(N - i)$ packets from the same helping node whose identities are part of $\overline{C}_{x,i}$, and so on. After scheduling all the possible coded packets from a certain helping node, which can help $(N - 1)$ nodes, the algorithm selects another node with lower $|\overline{C}_{s,i}|$ as a potential helping node provided that $max(|\overline{C}_{s,i}|) < |\overline{k}_i|$. Packets that are part of $\overline{C}_{x,i}$ where x is the selected helping node that have not been scheduled by any other considered helping node will be involved in creating a coded packet. If $max(|\overline{C}_{s,i}|) = |\overline{k}_i|$, only a node with $max(|\overline{C}_{s,i}|)$ will be considered as a potential helping node. After scheduling all the possible coded packets at all the potential helping nodes that can help $(N - 1)$ nodes, the algorithm updates the information table and the combination level will be decremented by one. On the other hand, if $(j \geq 2)$ and the combined packet can help a group of $(N - j)$ nodes this specific coded packet is scheduled. If not, the algorithm tries to find another group of $(N - j)$ nodes that can be helped by this specific coded packet. Finally, if this specific coded packet cannot help any group of $(N - j)$ nodes, the algorithm picks another possible combined packet from the same helping node, and so on. If none of the combined packets of the particular helping node can benefit any group of $(N - j)$ nodes, the algorithm selects another node with lower $|\overline{C}_{s,i}|$ as a potential helping node provided that $max(|\overline{C}_{s,i}|) < |\overline{k}_i|$. If $max(|\overline{C}_{s,i}|) = |\overline{k}_i|$ and none of the combined packets of the node with $max(|\overline{C}_{s,i}|)$ can benefit any group of $(N - j)$ nodes, the combination level will be decremented by one and the algorithm will try to create a coded packet for this new combination level.

Step 7: To check if node x will be able to help $(N - j)$ nodes at the $(N - i)$ combination level, the algorithm will compare the $(N - i)$ chosen packets of $\overline{C}_{x,i}$ with all the $\overline{C}_{s,i}$ packets. At each node, we have to find the availability of the constituting packets of each coded packet. If the availability number is $(N - i - 1)$, this node can decode the encoded packet and find its missing packet. We also have to find the identities of the unavailable packets at each node corresponding to a certain potential combined packet. If the total number of unavailable packets at all $(N - j)$ potentially helped nodes corresponding to a certain coded packet is $(N - i)$, then this coded packet can help all the $(N - j)$ helped nodes. However, for better and faster processing we propose the following equivalent and more efficient implementation. Denote the candidate $(N - i)$ packets to be combined at node x as $p_{x,e}$ where $e = 1, 2, .., N - i$. To make sure that these $(N - i)$ packets combined by node x

will help $(N-j)$ nodes, the following three conditions must be satisfied for the $(N-j)$ nodes to be helped:

$$\sum_{s=1,s\neq x}^{N-j} p_{I(s),1}\cdot p_{I(s),2}\cdots p_{I(s),N-i}=0 \qquad (3)$$

where $I(s)$, $s = 1, 2, \ldots, N-j$ denotes the identities of the nodes potentially helped by this combined packet. The number of such nodes to be helped is $(N-j)$. This condition guaranties that each node is missing at least one packet from the packets constituting the current subject combined packet.

For example, using the information in Table 3, suppose that node 5 is trying to combine packets (1, 2, 3, and 4) to help four nodes (1, 2, 3, and 4). By applying equation (3)

$$\sum_{s=1,s\neq x}^{4} p_{I(s),1}\cdot p_{I(s),2}\cdots p_{I(s),N-i}=0+0+0+0\Rightarrow=0$$

The first condition is satisfied and the algorithm will apply the second condition.

Table 3 Information table at each node describing the received and missed packets of all users.

Packet identity / Nodes identities	1	2	3	4
1	1	0	0	1
2	1	1	0	1
3	1	1	1	0
4	0	1	1	1
5	1	1	1	1
6	1	0	0	1

The second condition is:

$$\sum_{s=1,s\neq x}^{N-j}\sum_{e=1}^{N-i} p_{I(s),e}=(N-j)(N-i-1) \qquad (4)$$

Equation (4) effectively says that each of the potentially helped $(N-j)$ nodes has already received all but one of the constituent packets of the subject packet, i.e. they have received $(N-i-1)$ packets if equation (3) is satisfied. The total number of corresponding indicators in the table for all of the $(N-j)$ potentially helped nodes would be $(N-j)(N-i-1)$, as per equation (4). The result of applying this condition at node 5, that is trying to combine four packets to help four nodes, is

$$\sum_{s=1,s\neq x}^{4}\sum_{e=1}^{4} p_{I(s),e}=3+3+2+3\Rightarrow=11\Rightarrow\neq(N-j)(N-i-1)$$

Thus, this combined packet will not be scheduled for transmission since only three nodes can benefit from this combined packet (node 2, 3 and 4). However, if node 1 in Table 3 has received packet 2 or 3, as shown in Table 4, then the second condition is satisfied and the algorithm will apply the third condition.

The third condition is:

$$\sum_{s=1,s\neq x}^{N-j} p_{I(s),e}<N-j \quad for\ all\ e \qquad (5)$$

Equation (5) guaranties that no packet of the constituting $(N-i)$ packets has been received by all of the potentially helped $(N-j)$ nodes. For example, using Table 4, apply the third condition at node 5.

$$\sum_{s=1,s\neq x}^{4} p_{I(s),1}=3<N-j\ ,\quad \sum_{s=1,s\neq x}^{4} p_{I(s),2}=4\Rightarrow=N-j$$

$$\sum_{s=1,s\neq x}^{4} p_{I(s),3}=2<N-j\ ,\quad \sum_{s=1,s\neq x}^{4} p_{I(s),4}=3<N-j$$

Table 4 Information table at each node describing the received and missed packets of all users.

Packet identity / Nodes identities	1	2	3	4
1	1	1	0	1
2	1	1	0	1
3	1	1	1	0
4	0	1	1	1
5	1	1	1	1
6	1	0	0	1

Hence, this combined packet will not be scheduled for transmission since only three packets will be received at all four helped nodes and the algorithm will combine packets 1, 3, and 4 instead. But, if node 1 in Table 4 has received packet 3 and is missing packet 2, then all the conditions are satisfied and the algorithm will schedule this combined packet.

For a specific group of $(N-j)$ nodes to be helped the computed values of $\left|\overline{C}_{s,i}\right|$, $s = 1, 2, \ldots, N$ may have two or more equal values for different s. The scheduling algorithm arranges the stored nodes that have $\left|\overline{C}_{s,i}\right|\geq(N-i)$ in descending order with respect to values of $\left|\overline{C}_{s,i}\right|$. Whenever it faces more than one node with the same $\left|\overline{C}_{s,i}\right|$, it first picks the node with the lower node identity value as the potential helping node. Finally, in the flow chart, it may happen that node misses all of the packets after some scheduling, and so NC will not help with these leftover packets and single broadcast transmission should be used.

V. PERFORMANCE ANALYSIS

In this section, the upper bound on the improvement factor of the new XOR based scheduling algorithm is derived. We

assume that packets within a file are independently received at each node. Also, the channels between the BS and the nodes are erroneous and have the same quality. On the other hand, the transmission channels between nodes that can hear each other are error free.

For a certain number of nodes (*N* nodes) that can hear each other and given the probability of packet success P_c and file size *W*, the probability that *M* nodes have received a particular packet is:

$$P_M = \binom{N}{M} P_c^M (1-P_c)^{N-M} \qquad (6)$$

where *M=0, 1, 2, .., N*

For analysis convenience each packet is assumed to have been received by at least by one node. To guarantee this, the BS will retransmit packets that were not received by any node from previous transmissions until it receives at least one acknowledgment for each packet. Such acknowledgments and repeated BS transmissions details are not discussed further here. For a certain packet, the probability that *M* nodes will receive it after the first transmission is P_M. The probability that no node will receive it after the first transmission is P_0. If no node has received the packet after the first transmission, the BS will retransmit it and the probability that *M* nodes will receive it after the second transmission is $P_0 P_M$. If no node has received the packet after the second transmission, the BS will retransmit it and the probability that *M* nodes will receive it after the third transmission is $P_0^2 P_M$, and so on. The probability of a packet eventually being received by all *M* nodes is:

$$P_M^{'} = P_M + P_0 P_M + P_0^2 P_M + P_0^3 P_M + ...$$

$$P_M^{'} = P_M (1 + P_0 + P_0^2 + P_0^3 + ...)$$

$$P_M^{'} = \frac{P_M}{(1-P_0)}$$

$$P_M^{'} = \frac{P_M}{1-(1-P_c)^N} \quad for\ M = 1, 2, \ldots, N \qquad (7)$$

Where P_0 is the probability that a certain packet was not received by any node which can be computed using (6). For a certain packet, the average number of nodes (*R*) that will receive the packet can be found as follows:

$$R = P_1^{'} + 2P_2^{'} + 3P_3^{'} + ... + NP_N^{'} \qquad (8)$$

Thus, the average number of received packets (*k*), (the total number of ones in the information table, for example from Table 1, *k* = 26 packets) can be found as follows

$$k = WR \qquad (9)$$

For a given P_c, it is possible that some packets of the file are received by all nodes with a certain probability. Packets that are received by all nodes will not be considered for the local repair process. So, we find the distribution of the window size utilized for local repair ($W^{'}$) for a given P_c as follows:

$$P(W^{'} = D) = \binom{W}{D} (1-P_N^{'})^D P_N^{'(W-D)} \qquad (10)$$

In equation (10), $P_N^{'}$ is the probability that a packet is received by all nodes. Let *T* denote the required number of transmissions for a given $W^{'}$. The maximum number of transmissions (T_{max}) is $W^{'}$, which represents the case where no NC is possible. This scenario occurs with certain probability, depending on the values of P_c, *W* and $W^{'}$. Another possible scenario is ($T = W^{'} - 1$), which also occurs with a certain probability, and so on. In the best scenario, the algorithm will be able to create coded packets for the whole repair window. Each coded packet that is created will be able to help (*N – 1*) nodes. This scenario presents the minimum required number of transmissions. The minimum number of transmissions is

$$T_{\min} = \left\lceil \frac{NW - k}{N-1} \right\rceil \qquad (11)$$

The improvement factor is defined as the ratio between the required number of transmissions without NC ($N + W^{'}$) and the required number of transmissions using NC (*N* + *T*). In order to compute the upper bound on the improvement factor (F_{up}), we assume that, the minimum number of transmissions occurs with probability equal to one. Thus, the upper bound of the improvement factor is:

$$F_{up} = \sum_{W^{'}=1}^{W} \frac{(N + W^{'})}{(N + T_{\min})} P(W^{'}) \qquad (12)$$

VI. NUMERICAL AND SIMULATION RESULTS

In this section we show the improvement factor provided by the proposed XOR based scheduling algorithm for NC in cooperative local repair, through computer simulations and upper bound analysis. We assume that the channels between nodes are error free (nodes are very close to each other). However, the channels between the nodes and the BS are error prone, which leads to the need for repairs. In addition, we assume that the channels between the BS and the nodes are of the same quality which is a reasonable assumption since the nodes are close to each other. We compare our system's (*case 1*) improvement factor with two cases where no NC is applied. In *case 2*, nodes cooperate together, i.e. exchange initial packets availability to create an information table at each node. Since no NC is used, each missing packet will be transmitted individually by one user. Thus, the total number of transmissions is ($W^{'} + N$), where $W^{'}$ is the file size after removing the packets received by all nodes, and *N* is the number of packets necessary for transmitting packets' availability by all nodes. The latter assumes that one packet is

enough to convey the IDs of the packets missing at each node and a node's identity. In *case 3*, nodes do not cooperate with each other and each node will transmit all its received packets (no packets availability exchange, no coordination and no NC). For our scheduling approach (*case 1*) we take different N, W and P_c's, and for each set of values we compute the improvement factor, processing time, etc. For *case 1*, we count the total number of transmissions until all users are able to find all of the file packets. To this number, we add the N packets necessary for the reception report transmissions, to get the total number of transmissions. Uniform random variables are used to determine the IDs of the packets initially received from the BS at every node. The number of packets initially received from the BS at each node is obtained by calling a binomially distributed random variable for a given file size and a given probability of packet success.

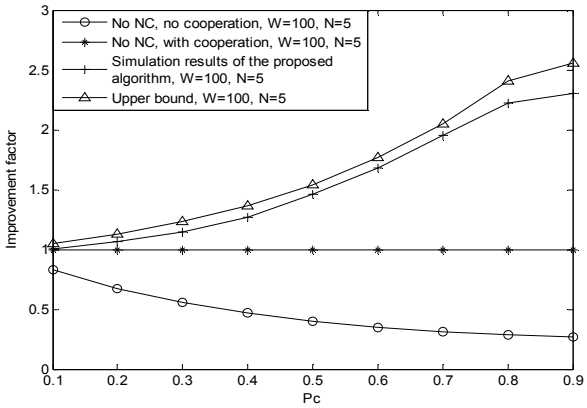

Figure 3 The improvement factors of: the new scheduling algorithm; with no NC and cooperation between nodes; and with no NC and no cooperation between nodes and the upper bound.

Figure 3 shows that, the new scheduling algorithm provides the best improvement factor compared to non-NC systems (*cases 2 and 3*). From figure 3 we can observe that, the improvement factor of the new scheduling algorithm is close to that for the upper bound. This is because the algorithm prioritizes the nodes with high packet availability to create coded packets that achieve maximum benefit. Figure 3 also shows that the higher the packets availability, the better the improvement factor, since the number of packets received by $(N-1)$ nodes increases when the packet availability increases.

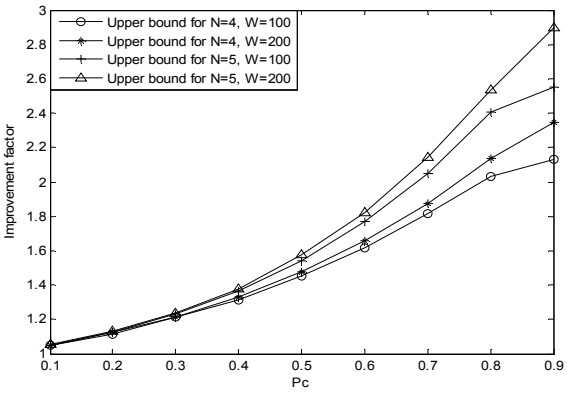

Figure 4 The upper bound on the improvement factors.

Figures 4 and 5 show the tradeoff between the number of nodes, the file size and the improvement factor. For ($P_c > 0.4$), as the file size or the number of nodes increases, the improvement factor increases. However, increasing the number of nodes provides a better improvement factor than increasing the file size. This is because as the number of nodes increases, the maximum combination level will increase. Therefore, the maximum number of packets that could be combined together increases. On the other hand, increasing the file size will lead to increasing the likelihood of having many coded packets, which will have a smaller effect as the file size becomes larger.

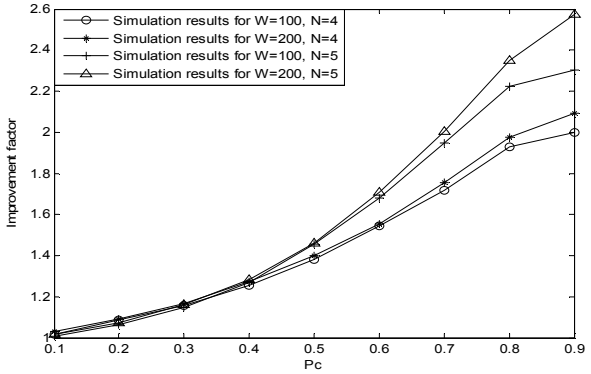

Figure 5 Simulation results.

Also from figure 4 and 5, we can observe that the effects of the file size and number of nodes for low P_c ($P_c < 0.4$) are negligible. The reason is the small number of possible generated coded packets for low P_c. The proposed scheduling algorithm always tries to find the best node that can XOR a set of packets that can help maximum number of nodes. For that reason, the performance of the proposed scheduling algorithm is close to the upper bound which can be seen in Figs. 4 and 5. The processing delay introduced by the scheduling process is an important issue to consider, since it determines the time required for local repair processes. In practice, cooperative local repair needs to be done in a certain amount of time depending on the delay tolerance of the application. The processing time was found by employing certain real time simulation indicators on a general purpose IBM PC (1.8 GH Intel(R) Core(TM)2 processor and 1 GB RAM).

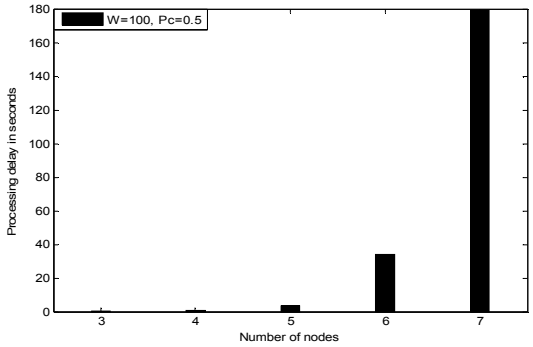

Figure 6 Processing delay (in seconds) vs. number of nodes in the new three-phase algorithm.

This is a worst case indicator in a real environment all the general functionality and processing related to performance measurement and curving would not be executed. In this sub section, we study the tradeoff between processing delay and number of nodes at certain file size and packet availability. Figure 6 shows that the less the number of nodes, the less will be the processing delay. The difference in processing delay for different number of nodes is not linear. For example, the processing delays in the cases of $N = 3$, 4 and 5 are almost the same, however it increases dramatically for $N = 7$ for these given parameters (file size and packets' availability).

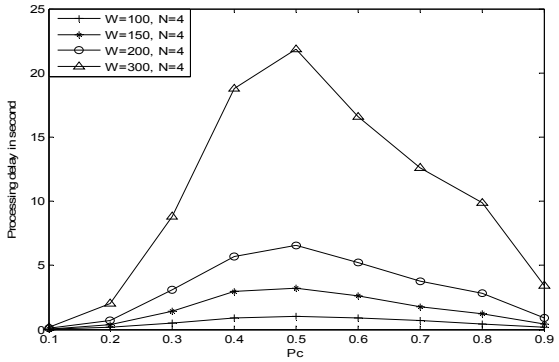

Figure 7 Processing delay (in seconds) vs. file size in the new three-phase algorithm.

Tradeoffs between processing delay, packet availability and file size are shown in figure 7. Figure 7 shows that as P_c increases, the processing delay increases up to a certain value of P_c. In addition to scheduling coded packets that will help ($N - 2$) or fewer nodes for ($0.1 \leq P_c \leq 0.5$) as P_c increases the chance of coded packets that can help ($N - 1$) nodes at different combination levels will increase which leads to more processing delay. For high P_c values (0.8 and 0.9), processing delay is reduced since NC will mostly prevail at ($N - 1$) combined packets that can help ($N - 1$) nodes. Also, the processing delay for $P_c = 0.9$ is less than the delay for $P_c = 0.8$ because the number of the packets considered for local repair operation for $P_c = 0.9$ is less than for $P_c = 0.8$.

Figure 7 shows that, for a certain number of nodes, increasing the file size will lead to increasing the processing delay. The relation between file size and processing delay is not linear. For instance, the processing delay is very small for $W = 100$ packets. However, it increases significantly for $W = 300$ packets. Also, from figure 7 we see that the required scheduling time is at a maximum when nodes have received 50% of the file.

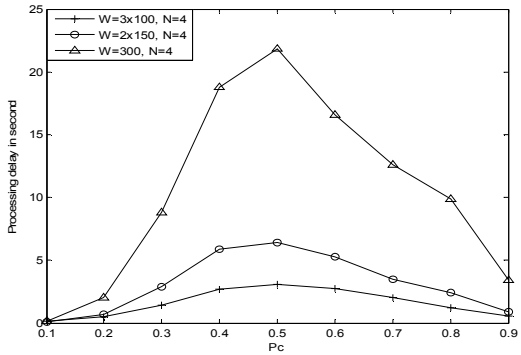

Figure 8 Segmentation effects on the processing delay in the new three-phase algorithm.

This is because most of the nodes will be involved at each combination level and the algorithm will look for the best combination with respect to the number of helped nodes ($N - j$) and the number of combined packets($N - i$). On the other hand, the processing delay is very low for $P_c = 10\%$, 20% and 90%, since most of the packets are either received by most of the nodes ($P_c = 90\%$) or received by a few nodes ($P_c = 10\%$ and 20%). Therefore, most of the scheduled transmissions will be from one combination level that can help ($N - 1$) nodes. For example, for $P_c = 90\%$, the most scheduled coded packets are combinations of ($N - 1$) packets that would help ($N - 1$) nodes.

Due to the high processing delay for the large file sizes in figure 7, file segmentation may be needed to reduce the processing delay. File segmentation will also reduce the improvement factor, as shown in figure 9, but this reduction will not be severe compared to the effects of the significant processing delays.

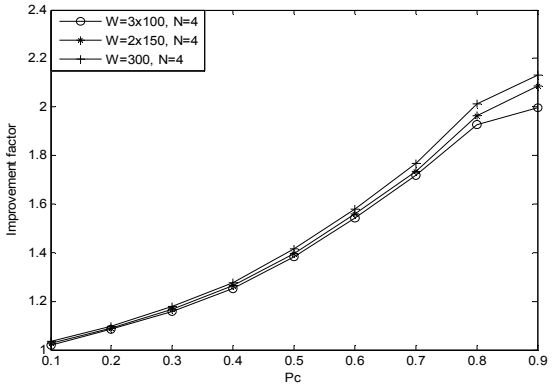

Figure 9. Segmentation effects on the improvement factor in the new three phase algorithm.

Figure 8 shows the processing delays for $W = 300$ packets and the required processing delay if we segment the same file into two or three files of sizes $W = 150$ and 100 packets. The difference in processing delay between no segmentation and with segmentation of file of size 300 packets is noticeable. At the same time, the improvement factor is reduced by a small amount, as shown in figure 9.

VII. CONCLUSION

In this paper we have presented a new XOR based scheduling algorithm for NC in cooperative local repair where all nodes can hear each other. We have found that, the improvement factor provided by the new XOR based scheduling algorithm is very close to the upper bound. We also investigated the effects of packets availability, file size and the number of nodes. We found that as the packets' availability increases, the improvement factor increases. Also, a large file requires more scheduling time than a short file, but it provides a better improvement factor. One solution for this problem is file segmentation which reduces the processing delay significantly while sacrificing a small amount of improvement factor. We also found that as the number of nodes increases, the improvement factor and the processing delay will increase.

REFERENCES

[1] S. Lin and D. J. Costello, *Error Control Coding :Fundamentals and Applications*, Second Edition ed.: Prentice Hall 2004.

[2] A. Leon-Garcia and I. Widjaja, *Communication Networks* Second Edition ed.: Elizabeth A. Jones, 2004.

[3] N. Dong, T. Tuan, N. Thinh, and B. Bose, "Wireless Broadcast Using Network Coding," *Vehicular Technology, IEEE Transactions on,* vol. 58, pp. 914-925, 2009.

[4] T. Tuan, N. Thinh, and B. Bose, "A Joint Network-Channel Coding Technique for Single-Hop Wireless Networks," in *Network Coding, Theory and Applications, 2008. NetCod 2008. Fourth Workshop on*, 2008, pp. 1-6.

[5] S. Raza, L. Danjue, C. Chen-Nee, and G. Cheung, "Cooperative Peer-to-Peer Repair for Wireless Multimedia Broadcast," in *Multimedia and Expo, 2007 IEEE International Conference on*, 2007, pp. 1075-1078.

[6] L. Xin, G. Cheung, and C. Chen-Nee, "Structured Network Coding and Cooperative Wireless Ad-Hoc Peer-to-Peer Repair for WWAN Video Broadcast," *Multimedia, IEEE Transactions on,* vol. 11, pp. 730-741, 2009.

[7] R. Ahlswede, C. Ning, S. Y. R. Li, and R. W. Yeung, "Network information flow," *Information Theory, IEEE Transactions on,* vol. 46, pp. 1204-1216, 2000.

[8] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, S. Jun, and B. Leong, "A Random Linear Network Coding Approach to Multicast," *Information Theory, IEEE Transactions on,* vol. 52, pp. 4413-4430, 2006.

[9] S. Katti, H. Rahul, H. Wenjun, D. Katabi, M. Medard, and J. Crowcroft, "XORs in the Air: Practical Wireless Network Coding," *Networking, IEEE/ACM Transactions on,* vol. 16, pp. 497-510, 2008.

[10] L. Xin, S. Raza, C. Chen-Nee, and C. Gene, "Network Coding Based Cooperative Peer-to-Peer Repair in Wireless Ad-Hoc Networks," in *Communications, 2008. ICC '08. IEEE International Conference on*, 2008, pp. 2153-2158.

[11] L. Xin, G. Cheung, and C. Chen-Nee, "Rate-distortion optimized network coding for cooperative video stream repair in wireless peer-to-peer networks," in *World of Wireless, Mobile and Multimedia Networks, 2008. WoWMoM 2008. 2008 International Symposium on a*, 2008, pp. 1-6.

[12] L. Xin, C. Gene, and C. Chen-Nee, "Structured network coding and cooperative local peer-to-peer repair for MBMS video streaming," in *Multimedia Signal Processing, 2008 IEEE 10th Workshop on*, 2008, pp. 456-461.

[13] K. Nguyen, N. Thinh, and C. Sen-ching, "Peer-to-Peer Streaming with Hierarchical Network Coding," in *Multimedia and Expo, 2007 IEEE International Conference on*, 2007, pp. 396-399.

[14] K. Chi, X. Jiang, and S. Horiguchi, "A More Efficient COPE Architecture for Network Coding in Multihop Wireless Networks," *IEICE Transactions on Communications,* vol. E92.B, pp. 766-775, 2009.

[15] P. A. Chou, Y. Wu, and K. Jain, "Practical Network Coding," in *Proceeding of 41st Annual Allerton Conference on Communication, Control and Computing*, 2003.

[16] Y. Yan, Z. Baoxian, H. T. Mouftah, and M. Jian, "Practical Coding-Aware Mechanism for Opportunistic Routing in Wireless Mesh Networks," in *Communications, 2008. ICC '08. IEEE International Conference on*, 2008, pp. 2871-2876.

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

[18] R. Koetter and M. Medard, "An algebraic approach to network coding," *Networking, IEEE/ACM Transactions on,* vol. 11, pp. 782-795, 2003.

[19] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading Structure for Randomness in Wireless Opportunistic Routing," in *Proc. ACM SIGCOMM*, 2007.

[20] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Information Theory, 2003. Proceedings. IEEE International Symposium on*, 2003, p. 442.

[21] Z. Jingyao, F. Pingyi, and K. Ben Letaief, "Network Coding for Efficient Multicast Routing in Wireless Ad-hoc Networks," *Communications, IEEE Transactions on,* vol. 56, pp. 598-607, 2008.

[22] N. Bin, N. Santhapuri, Z. Zifei, and S. Nelakuditi, "Routing with opportunistically coded exchanges in wireless mesh networks," in *Wireless Mesh Networks, 2006. WiMesh 2006. 2nd IEEE Workshop on*, 2006, pp. 157-159.

[23] J. Ben Saleh and A. K. Elhakeem, "A practical scheduling approach to network coding for wireless local repair," in *Communications (QBSC), 2010 25th Biennial Symposium on*, pp. 305-310.