Information Exchange in Wireless Networks with Network Coding and Physical-layer Broadcast

Yunnan Wu
Dept. of Electrical Engineering,
Princeton University,
Princeton, NJ 08544
e-mail: yunnanwu@princeton.edu

Philip A. Chou
Microsoft Research,
One Microsoft Way,
Redmond, WA 98052-6399
e-mail: pachou@microsoft.com

Sun-Yuan Kung
Dept. of Electrical Engineering,
Princeton University,
Princeton, NJ 08544
e-mail: kung@princeton.edu

Abstract —

The exchange of independent information between two nodes in a wireless network can be viewed as two unicast sessions, corresponding to information transfer along one direction and the opposite direction. In this paper we show such information exchange can be efficiently performed by exploiting network coding and the physical-layer broadcast property offered by the wireless medium, which improves upon conventional solutions that separate the processing of the two unicast sessions. We propose a distributed scheme that obviates the need for synchronization and is robust to random packet loss and delay, and so on. The scheme is simple and incurs minor overhead.

I. Introduction

In this paper, we investigate the mutual exchange of independent information between two nodes in a wireless network. Let us name the two nodes in consideration a and b, respectively. Consider a packet-based communication network with all packets of equal size. The basic problem is very simple: a wants to transmit a sequence of packets $\{X_1(n)\}$ to b and b wants to transmit a sequence of packets $\{X_2(n)\}$ to a. Assume the two sequences of information packets, $\{X_1(n)\}$ and $\{X_2(n)\}$, are from two independent information sources.

Information exchange finds many useful applications. These include voice conversations, video conferencing between two participants, and instant messaging. In fact, the scope of information exchange goes much further beyond the generic two-way end-to-end communications listed above. Note that a and b do not have to be the true communication end-points for the packets $\{X_1(n)\}$ and $\{X_2(n)\}$. For example, in a wireless ad hoc network where every node can act as a router, information exchange occurs as long as there are some packets $\{X_1(n)\}$ to be routed through a to b and some other packets $\{X_2(n)\}$ to be routed through b to a. This is illustrated in Figure 1, where a and b are two wireless routers, each having some packets to be routed to the other. We can treat a and b as logical end-points for the information exchange between a and b. After all, as long as packets $\{X_1(n)\}$ and $\{X_2(n)\}$ are successfully exchanged, it does not matter which end-to-end session each packet originally belongs to.

An information exchange session between a and b is essentially two unicast sessions, one from a to b and the other from b to a. Since the two unicast sessions carry independent information, it may appear at first glance that the two sessions can be treated separately, by devoting a first route for packets $\{X_1(n)\}$ to flow from a to b and a second route for packets $\{X_2(n)\}$ to flow from b to a. However, as we will show, a joint processing of these two unicast sessions can in fact outperform a separate treatment, in a wireless network.

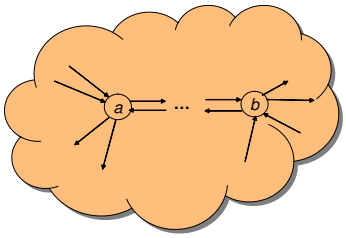

Fig. 1: An example scenario of information exchange. a and b are two wireless routers, each having packets to be routed to the other.

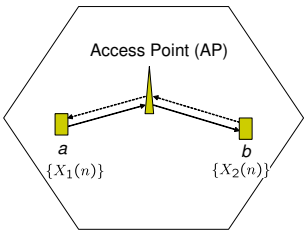

Fig. 2: An example scenario of two-way communications in a WLAN.

We now use a simple example to illustrate the basic idea. Figure 2 gives an example scenario of two-way communication in a wireless local area network (WLAN). Assume the WLAN operates in the infrastructure mode, which is similar in structure to cellular networks for voice communications. There is an access point (AP) connected to a wire-line network. All transmissions involve the access point and are classified as either down-link or up-link. Under this architecture, let us consider the problem of mutual exchange of independent information between two nodes a, b, both lying within the coverage area of the AP. Conventionally, each packet $X_1(n)$ (resp. $X_2(n)$) would be first transmitted to AP with an uplink transmission, and then transmitted from AP to b (resp. a) with a downlink transmission. Now let us show a better scheme. In the uplink periods, packets $\{X_1(n)\}$ and $\{X_2(n)\}$ are first transmitted to the access point, just as in the conventional solution. The difference occurs at the second hop. Note that the wireless medium is broadcast in nature. Hence we assume a packet sent from the access point can reach both a and b.

With the better scheme, bitwise XOR-ed results of $\{X_1(n)\}$ and $\{X_2(n)\}$, $\{X_1(n) \oplus X_2(n)\}$ are broadcast from the access point in the downlink periods. With these XOR-ed packets, it is easy to see that a and b can solve for $\{X_2(n)\}$ and $\{X_1(n)\}$, respectively. The scheme is based on two essential ingredients. First, it exploits the physical-layer broadcast property offered by the wireless medium. In other words, in a wireless network, a single transmission may successfully reach a number of neighboring nodes. Second, it utilizes *network coding*. Network coding refers to a scheme where nodes in a network are allowed to perform arbitrary operations on the data received to produce output data, rather than just *routing*, i.e., replicating and forwarding received data.

The concept of *network coding* has evolved recently as an interesting extension of the more traditional routing paradigm. Historically, network coding was proposed and studied mainly as a means to facilitate information multicast in a communication network, i.e., transmitting common information from a sender to a set of receivers. In their pioneering work [1], Ahlswede et al. demonstrated that it is in general suboptimal to restrict the network nodes to perform only routing. They showed that the *multicast capacity*, which is defined as the maximum rate that a sender can communicate common information to a set of receivers, is given by the minimum capacity of cuts separating the sender from a receiver. Moreover, they showed that while the multicast capacity cannot be achieved in general by routing, it can be achieved by network coding. Subsequently, Li, Yeung, and Cai [2] showed that it is sufficient for the encoding functions at the interior nodes to be linear. Koetter and Médard [3] gave an algebraic characterization of linear encoding schemes and proved existence of linear time-invariant codes achieving the multicast capacity.

Network coding is highly applicable to real packet networks. For example, Chou, Wu, and Jain [4] presented a prototype system for practically performing network coding in multicasting applications over packet networks, using distributed random linear network coding with buffering. The system achieves throughput close to capacity with low delay, and is robust to random packet loss and delay as well as to changes to network topology or capacity. Distributed random network coding were also investigated by Ho et al. [5, 6].

An increasingly important application domain of network coding is wireless ad hoc networks. Previous work by Wu et al. [7] considered the cross-layer optimization of wireless ad hoc networks, using network coding as the abstraction of the network layer. Previous work by Wu, Chou, and Kung [8] showed that network coding can be used to achieve the minimum energy-per-bit for information multicast in a mobile ad hoc network, under a layered model of wireless networks. A similar result was obtained in an independent work by Lun et al [9]. Regarding practical implementation, by having random mixture packets self-orchestrate multiple paths, practical network coding offers built-in error protection and adaptivity to topology changes due to joins, leaves, node or link failures, congestion, etc; by employing a flooding-type delivery, network coding can be implemented in a distributed fashion easily, whereas the creation and maintenance of distribution trees incurs notable signalling overhead, given the dynamic environment. These properties render network coding potentially useful for unicasting and multicasting in mobile ad hoc networks.

In this work, we identify information exchange in wireless networks as an additional application scenario where network coding exhibits unique advantages over conventional routing. Generalizing from the example in Figure 2, we show in Section II that network coding, combined with physical-layer broadcast, can facilitate mutual exchange of information in a wireless network by providing the same rate while consuming less network resource (power, use of channel). Specifically, for general information exchange problems, the union of a forward path from a to b and a backward path from b to a is sufficient to provide the same throughput as achievable via conventional routing. We also show how to implement network coding to realize this gain in a synchronous system. In Section III, we discuss distributed implementations without assuming synchronization.

II. Physical Piggybacking

Let us extend the example in Figure 2 from $L = 3$ nodes to $L = 4$ nodes. Figure 3 shows four nodes spaced in a line, labelled 1, 2, 3, and 4 from left to right. Suppose node 1 has a stream of packets $X_1(n)$ and node 4 has another stream of packets $X_2(n)$ and they want to exchange data. Conventionally, this would require a forward path from 1 to 4 and a backward path from 4 to 1, as shown in Figure 3.

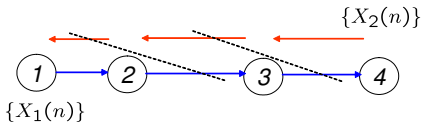

Fig. 3: An example scenario of information exchange.

Extending the earlier example, we now use a single broadcast transmission from 2 to $\{1, 3\}$ to replace two transmissions, $2 \to 3$ on the path from 1 to 4 and $2 \to 1$ on the path from 4 to 1; similarly, we use a single broadcast transmission from 3 to $\{2, 4\}$ to replace two transmissions, $3 \to 4$ and $3 \to 2$. We next show how to use network coding to achieve (asymptotically) the same rate of information exchange, with this more economic use of network resources.

First, we represent the resulting network by a graph $G = (V, E)$ with all edges having unit capacity, shown in Figure 4. In Figure 4, we model the physical layer broadcast by a tree-like structure, as proposed in [7]. For example, the broadcast transmission from 2 to $\{1, 3\}$ is represented by edges $\{22', 2'1, 2'3\}$, where $2'$ is a new node. Node $2'$ plays the role of an artificial bottleneck that constrains the rate of new information going out of the transmitter.

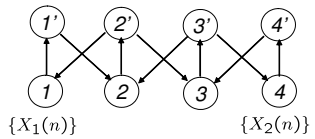

Fig. 4: Graphical representation of Figure 3. Each edge has unit capacity. Nodes $1', 2', 3'$ and $4'$ are introduced to model physical-layer broadcast.

Now we describe a network coding solution that achieves the required unit rate for information exchange between node

1 and 4. Assume the network operates as a synchronous system with a discrete time index running from 1 to $+\infty$. Assume each transmission link has unit delay. Accordingly, let edges $1'2$, $2'1$, $2'3$, $3'2$, $3'4$, and $4'3$ have unit delay and edges $11'$, $22'$, $33'$ and $44'$ have zero delay in Figure 4. Each edge has unit capacity and thus can carry one packet in each time unit. A network coding solution refers to an assignment of information packet flowing on each edge in each time unit. Let $Y_{uv}(n)$ denote the information packet assigned to edge uv in the n-th time unit. Furthermore, in order for the assignment to be realizable, $Y_{uv}(n)$ must be derived from the packets received earlier by node u. The solution for general $L \geq 3$ is given as follows:

$$Y_{11'}(n) = Y_{1'2}(n) = X_1(n), \tag{1}$$

$$Y_{L,L'}(n) = Y_{L',L-1}(n) = X_2(n), \tag{2}$$

$$\begin{aligned} Y_{l,l'}(n) &= Y_{l',l-1}(n) = Y_{l',l+1}(n) \\ &= X_1(n-(l-1)) \oplus X_2(n-(L-l)), \\ &\qquad l = 2, \ldots, L-1, \end{aligned} \tag{3, 4}$$

where for $n < 1$, we treat $X_i(n)$ as a *zero-packet*, i.e., a packet with all bits being zero.

We now verify that this network coding solution is realizable and enables node l to recover $\{X_1(n)\}$ with delay $l-1$ and $\{X_2(n)\}$ with delay $L-l$, by induction over n. At time $n = 1$, node 1 can recover $X_1(1)$ and node L can recover $X_2(1)$ since they are available initially. Hence the claim is true for $n = 1$. By inductive assumption, up to time n, node l can recover $X_1(1), \ldots, X_1(n-(l-1))$ and $X_2(1), \ldots, X_2(n-(L-l))$. At time $n+1$, node l receives

$$X_1(n-(l-1-1)) \oplus X_2(n-(L-l+1))$$

from edge $(l-1)'l$, which was sent by node $l-1$ at time n. Therefore, with this new packet, node l can recover $X_1(n+1-(l-1))$. Similarly, at time $n+1$, node l receives

$$X_1(n-(l+1-1)) \oplus X_2(n-(L-l-1))$$

from edge $(l+1)'l$, which was sent by node $l+1$ at time n. Therefore, with this new packet, node l can recover $X_1(n+1-(L-l))$. Thus the claim is established.

It is worth pointing out that $l-1$ (resp. $L-1$) time units is in fact the minimum possible delay for node l to receive $\{X_1(n)\}$ (resp. $\{X_2(n)\}$). Hence in this context, network coding achieves efficiency of resource (power, use of channel) usage without incurring any delay penalty.

To summarize, with network coding and the physical layer broadcast, a single broadcast transmission $2 \to \{1,3\}$ can now replace two transmissions, $2 \to 3$ on the path from 1 to 4 and $2 \to 1$ on the path from 4 to 1. Yet the amount of resources consumed by $2 \to \{1,3\}$ (power, use of channel) is only the maximum of that consumed by each of the two transmissions $2 \to 3$ and $2 \to 1$. It looks as if the transmission $2 \to 1$ is now *piggybacked* on the transmission $2 \to 3$ without additional cost! Consequently, we use the name *physical piggybacking with network coding* to refer to this unique advantage of network coding plus physical layer broadcast over routing.

A. Information Exchange as a Virtual Multicast Session

It should be clear by now that network coding offers unique advantages over conventional routing for information exchange in a wireless network, which involves two unicast sessions. In contrast, most previous research results about network coding have been focused on enhancing the efficiency of a single multicast session.

In fact, the problem of information exchange can be transformed into an information multicast problem via some graph transformations; this technique was used in, for example, [1]. This connection enables us to explain the gain offered by network coding in information exchange in the context of information multicast, for which a richer collection of research results is available.

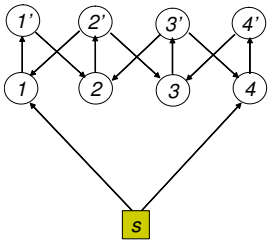

Fig. 5: Information exchange can be viewed as a multicast session.

Let us explain this using the example in Figure 3. We add to Figure 3 a virtual source node s that has a unit capacity edge entering node 1 and a unit capacity edge entering node 4; the resulting graph is shown in Figure 5. Then we consider a multicast session from sender s to receivers $\{1,4\}$ with rate 2. In this graph, the multicast capacity from s to $\{1,4\}$) is 2 since there are two edge-disjoint paths from s to node 1 and two edge-disjoint paths from s to node 4. Therefore, there exists a linear network coding scheme achieving the multicast capacity 2 [1–3]. To achieve a multicast rate of 2, distinct information has to be loaded on edge $s1$ and $s4$, which can be defined to be $\{X_1(n)\}$ and $\{X_2(n)\}$, respectively. This relates the information exchange between node 1 and node 4 with the virtual multicast session.

With this connection established, the gain of physical piggy-backing can now be explained as follows. With network coding, the union of a forward path from a to b and a backward path from b to a is sufficient to provide the same throughput as via conventional routing.

III. Distributed Implementation

In Section II, we have given a network coding solution for information exchange, assuming synchronization is available, links are lossless, and links have unit capacity and unit delay. In real networks, however, packet transmissions are subject to random delays and losses on every link, and links have essentially unknown capacities, which vary as competing communication sessions begin and end. In addition, synchronization is often regarded as difficult and costly, if not infeasible.

Since information exchange can be viewed as a virtual multicast session (Section II.A), practical network coding designs for multicasting applications [4–6] can be applied. We now briefly review the practical network coding system [4], which was developed for multicasting information in packet networks. The source packets are grouped into multiple *generations*, each containing h packets. Random linear network coding is applied separately to different generations. The packet format in the practical network coding system [4] is shown in

Figure 6. Assume that network coding is used with an operating field $GF(2^m)$. In Figure 6, the generation ID indicates the generation that this packet belongs to; the global encoding vector records the composition of this packet as a linear combination of the source packets within this generation. Denote the source packets within this generation by $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_h$, each being a row vector with elements in $GF(2^m)$. Then a packet with global encoding vector $[g_1, \ldots, g_h]$ means the payload of this packet is $\sum_{i=1}^{h} g_i \boldsymbol{x}_i$.

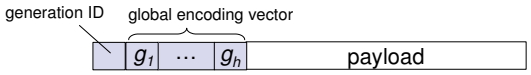

Fig. 6: The packet format in the practical network coding system [4].

Each node in the network maintains a buffer. Whenever a node receives a packet via one of its incoming links, it stores the packet into its buffer. Whenever there is a transmission opportunity available on one of its outgoing links, a node generates an output packet by linearly combining the packets in the buffer with random coefficients, $g_1, \ldots, g_h$, in $GF(2^m)$. After a destination node receives h packets with linearly independent global encoding vectors, it can recover the source packets via Gaussian elimination.

The throughput performance of the system, or from a different perspective, the efficiency in using the network resources, depends on the network topology, the generation size, the field size, etc. Generally speaking, the generation size h and/or the field size 2^m need to be sufficiently large to ensure that the proportion of wasted packet transmissions (packets that do not convey new information) is small. However, on the other hand, a large generation size and/or field size leads to a large amount of overhead in the packet header. Therefore, there is a tradeoff between these considerations.

Note though, the current setup for information exchange in wireless networks has a very special structure. Exploiting these structural properties, we propose a distributed scheme that is simpler to implement and incurs less overhead than the general practical network coding design. This proposed distributed implementation also obviates the need for synchronization and is robust to random packet loss and delay, and so on. In the following we describe this proposed implementation.

A. Basic Scheme

In the synchronous solution discussed in Section II, each packet flowing in the network is of the form $X_1(p) \oplus X_2(q)$. We now impose this restriction by requiring each packet to be of this form. We explicitly record p and q as meta-data in the packet, so as to indicate the composition of this packet. This is shown in Figure 7. Compared with Figure 6, the overhead in representing the coding operations is now reduced.

In the synchronous solution, the received packets at each node can be recovered as a sequence of "right-bound" packets $\{X_1(n)\}$ and a sequence of "left-bound" packets $\{X_2(n)\}$. We now maintain two buffers at each node, named buffer 1 and buffer 2, which hold these two types of packets respectively. This is illustrated in Figure 8. These two buffers characterize the current *knowledge* of $\{X_1(n)\}$ and $\{X_2(n)\}$ of a node. At

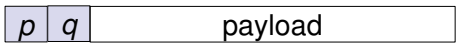

Fig. 7: The packet format. For a packet with payload being $X_1(p) \oplus X_2(q)$, p and q are explicitly recorded in the packet as metadata.

a given time, a packet $X_1(p)$ (resp. $X_2(q)$) is said to be *known* to node l if $X_1(p)$ (resp. $X_2(q)$) resides in the current buffer 1 (resp. buffer 2) of node l, and *unknown* or *new* to node l otherwise. Assume for the moment that the buffers have infinite capacity.

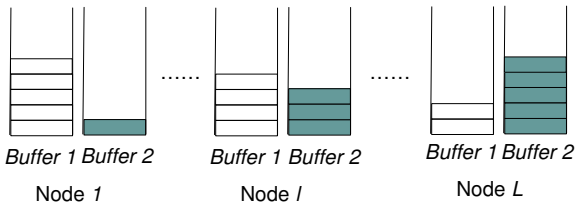

Fig. 8: The two buffers at the nodes. At any time, the content of buffer 1 in all nodes reflects the current progress of propagating source packets $\{X_1(n)\}$ from left to right; similarly, the content of buffer 2 in all nodes reflects the current progress of propagating source packets $\{X_2(n)\}$ from right to left.

Assume for now that there is an exogenous mechanism that decides when a transmission opportunity is available at a node. We present the basic scheme by describing the operations performed at a generic node l in response to events, since this is a distributed approach. Whenever there is a transmission opportunity at l, a packet is generated by taking one packet $X_1(p)$ from buffer 1 and one packet $X_2(q)$ from buffer 2 and computing the XORed result, $X_1(p) \oplus X_2(q)$. The specific rules of selecting $X_1(p)$ and $X_2(q)$ will be discussed later. Whenever a packet $X_1(p) \oplus X_2(q)$ arrives at a node l, there are four cases depending on whether $X_1(p)$ is known/unknown and $X_2(q)$ is known/unknown to l:

1. If $X_1(p)$ is known and $X_2(q)$ is unknown, then node l decodes $X_2(q)$ and stores it into buffer 2.
2. If $X_1(p)$ is unknown and $X_2(q)$ is known, then node l decodes $X_1(p)$ and stores it into buffer 1.
3. If both $X_1(p)$ and $X_2(q)$ are known, this received packet is ignored since it does not provide any new information.
4. If neither $X_1(p)$ nor $X_2(q)$ is known, this packet is also ignored since it cannot be decoded.

We now show that the last case will never happen by inductively proving an invariant property: at any time, at any node l, the content in buffer 1 is always a subset of that of node $l-1$ and the content of buffer 2 is always a subset of that of node $l+1$. This is illustrated in Figure 8. Initially, this invariant is true. When a packet $X_1(p) \oplus X_2(q)$ arrives at a node l from node $l+1$, by induction, $X_1(p)$ is known and hence $X_2(q)$ can be decoded. Similarly, when a packet $X_1(p) \oplus X_2(q)$ arrives at a node l from node $l-1$, by induction, $X_2(q)$ is known and hence $X_1(q)$ can be decoded. Hence this property continues to hold after any packet reception event.

B. Output Generation

Now let us discuss the rule of selecting $X_1(p)$ and $X_2(q)$ from the buffers to generate an output packet $X_1(p) \oplus X_2(q)$, when a transmission opportunity is available at node l. By now, it should be clear that a single packet $X_1(p) \oplus X_2(q)$ essentially provides $X_1(p)$ to node $l+1$ and $X_2(q)$ to node $l-1$. Therefore, we should try to select a packet $X_1(p)$ (resp. $X_2(q)$) that is new or most likely to be new to node $l+1$ (resp. $l-1$). Due to symmetry, in the following we will only discuss the selection of $X_1(p)$ from buffer 1.

If the wireless links can be assumed to be lossless, i.e., each transmitted packet can be successfully received by the left and right neighbors, the right-bound packets $\{X_1(n)\}$ known by node $l+1$ are just those packets that have been transmitted by node l with physical piggybacking. Consequently, node l can simply choose $X_1(p)$ as a right-bound packet that has not been transmitted, if there indeed exists one such packet. If all right-bound packets have been transmitted (and hence successfully received by node $l+1$), $X_1(p)$ can be set as a zero-packet. Furthermore, once a packet $X_1(p)$ has been transmitted, it can be removed from the buffer since its successful reception at node $l+1$ can be assured.

Now suppose links are lossy. For simplicity, suppose the packet loss probability on each transmission can be characterized by a single parameter α. Assume each node has a *transmission rate* of 1.0, which refers to the average number of transmission opportunities per unit time. Then a link from l to $l+1$ has a *reception rate* of $1-\alpha$, which refers to the average number of successfully received packets per unit time. Therefore, the maximum achievable throughput for information exchange is upper-bounded by $1-\alpha$. Achieving this bound requires that almost all packets received by $l+1$ from l provide new information to $l+1$.

In the simple scheme mentioned earlier in this section, a source packet $X_1(p)$ is transmitted at most once across any link. Hence, a source packet can arrive at the destination only if it successfully traverses all the intermediate links. Consequently, the achieved throughput by the simple scheme is at most

$$(1-\alpha)^{L-1}.$$

The efficiency loss of the simple scheme can be attributed to the (wasted) transmissions of zero-packets when all right-bound packets have been transmitted. We now outline an improvement strategy. Instead of transmitting zero-packets in this scenario, re-transmitting packets in the buffer might improve the performance since an earlier transmission might get lost. Then, a natural question to ask is which packet should be re-transmitted in such a scenario. To deal with this, we propose to associate with each packet $X_1(n)$ in the buffer a field `ProbNew`, representing the current belief (held by node l) of this packet being new to node $l+1$. We always choose one packet with the largest value of `ProbNew`; if there are more than one such packets, we always choose the one with the lowest sequence number. This field, namely the belief of a packet being new, may be altered by subsequent observations (events) according to Bayes's rule. Initially, for a packet $X_1(n)$ that has not been transmitted, we set

$$\texttt{ProbNew}[X_1(n)] = 1. \tag{5}$$

After a packet $X_1(n)$ with $p_0 \equiv \texttt{ProbNew}[X_1(n)]$ has been transmitted, the posterior belief can be set as

$$\texttt{ProbNew}[X_1(n)] = p_0\alpha. \tag{6}$$

Another interesting event is the reception of a packet from node $l+1$. After node l receives a packet $X_1(p') \oplus X_2(q')$ from node $l+1$, it can infer that $X_1(p')$ has been successfully received by node $l+1$ and hence should set

$$\texttt{ProbNew}[X_1(p')] = 0. \tag{7}$$

Thus it can be seen that a single packet $X_1(p') \oplus X_2(q')$ from node $l+1$ serves dual purposes:

- It acknowledges the successful reception of $X_1(p')$ at node $l+1$;
- It also provides $X_2(q')$ to node l.

This reveals yet another form of piggybacking facilitated by the broadcast nature of wireless medium.

However, this improved scheme by itself cannot guarantee that each packet $X_1(n)$ will eventually reach the destination. To see this, suppose a packet $X_1(p') \oplus X_2(q')$ from $l+1$ successfully reached node $l+2$ (and later acknowledged) but did not reach node l. Then, node l may never be sure whether $X_1(p')$ has successfully reached node $l+1$ since the acknowledgement was lost and not repeated later.

We now discuss how to revise the scheme to guarantee that each packet will eventually reach the destination. The fact that such guarantee was not possible with the earlier scheme can be attributed to the insufficiency of the piggybacked acknowledgement of $X_1(p')$. This problem can be solved by using a stronger form of acknowledgement. Specifically, we add two new fields, $\texttt{CACK}_1$ and $\texttt{CACK}_2$, in the packet format, as shown in Figure 9. The field $\texttt{CACK}_1$ (resp. $\texttt{CACK}_2$) serves as cumulative acknowledgement (CACK) of $\{X_1(n)\}$ (resp. $\{X_2(n)\}$) known to the node that transmits this packet. For example, if $\texttt{CACK}_1 = 3$, then $\{X_1(n), n \le 3\}$ has been received but $X_1(4)$ has not been received.

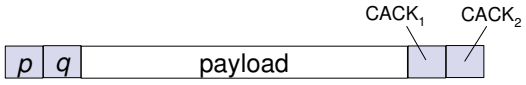

Fig. 9: The revised packet format with cumulative acknowledgement fields.

With this explict CACK, node l can better infer the content in buffer 1 of node $l+1$. After node l receives a packet $X_1(p') \oplus X_2(q')$ from node $l+1$ with $\texttt{CACK}_1 = k$, it knows that

$$\{X_1(n), n = 1, \ldots, k\} \cup \{X_1(p')\} \tag{8}$$

have been received by node $l+1$; the payload for these packets can then be eliminated from the buffer.

In addition, it also knows that $X_1(k+1)$ has not been received by the time this packet was transmitted, which gives information to update $\texttt{ProbNew}[X_1(k+1)]$ (assuming node l knows $X_1(k+1)$). Let τ denote the time when the packet $X_1(p') \oplus X_2(q')$ is received by node l. Let ϵ denote the time it takes to receive a packet. There are two cases:

1) If during $[\tau - 2\epsilon, \tau]$ node l did not transmit $X_1(k+1)$, then it can infer that the packet $X_1(k+1)$ is new to node $l+1$ and hence it should set

$$\texttt{ProbNew}[X_1(k+1)] = 1. \tag{9}$$

In this case, next time when a transmission opportunity is available at node l, $X_1(k+1)$ will be selected since it has the lowest sequence number among all packets in buffer 1 of node l with `ProbNew` being 1.

2) If during $[\tau - 2\epsilon, \tau]$ node l transmitted $X_1(k+1)$ $r > 0$ times, then it should set

$$\texttt{ProbNew}[X_1(k+1)] = \alpha^r. \tag{10}$$

When ϵ is small, the second case is rare, especially with $r > 1$. To simplify the implementation, we may just keep track of the time when each packet in the buffer was most recently transmitted. This allows to tell whether it is case 1 or case 2 above. If case 2 happens, as a simplification, just set $\texttt{ProbNew}[X_1(k+1)] = \alpha$.

Additional remarks: It is worth mentioning that with the proposed scheme, the value of `ProbNew` can only be α^k for some integer k or 0. Therefore, we just need to keep track of the power k, which indicates the number of transmissions before an acknowledgement comes back. The packet loss probability α can be regarded as a parameter introduced mainly to facilitate the analysis.

Earlier we have assumed that the buffers have infinite capacity. Now let us examine the buffer space required by the proposed scheme. Whenever node l infers that a certain packet $X_1(k)$ has been successfully received by node $l+1$, the payload of this packet can be eliminated from the buffer of node l. Therefore, node l only needs to store packets that have not been acknowledged.

As a side remark, we note that if some nodes can reach more than two neighboring nodes, the proposed scheme can still be applied, although the inference performance and the transmission efficiency may be improved further with additional knowledge about the transmission range.

C. Data-Driven Medium Access

Earlier we have assumed that there is an exogenous mechanism that decides when a transmission opportunity is available at a node and focused on deciding what data to transmit. In practice, a node also has control over how aggressive it should try to access the medium. Under the traditional routing paradigm, a node should not access the medium unless it has a packet to transmit. Extending this to the scenario with network coding, we propose a data-driven medium access control mechanism, where the aggressiveness of medium access is determined by the potential value of a transmission opportunity if granted. In fact, one way to evaluate a transmission opportunity has been given in the previous subsection. Recall that $\texttt{ProbNew}[X_1(n)]$ indicates the belief held by node l of $X_1(n)$ being new to node $l+1$. Let p_1 (resp. p_2) denote the maximum `ProbNew` over all packets in buffer 1 (resp. buffer 2). Then, $p_1 + p_2$ can be treated as the value of a transmission opportunity. To maximize the efficiency in using resources, ideally almost all transmissions should have a value of 2.

IV. Conclusion

In this paper, we identified information exchange in wireless networks as a new application scenario where network coding can offer unique advantages over conventional routing. Network coding, together with the physical layer broadcast property offered by the wireless medium, can improve the efficiency in using resources by facilitating *physical piggybacking*. To realize the advantages in practice, it is possible to make use of the practical network coding system [4], since information exchange can be treated as a virtual multicast session. Observing the special structural features of the current problem, in this paper we proposed a distributed and robust scheme that is simpler to implement and incurs less overhead than the practical network coding system [4].

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Information Theory*, IT-46(4):1204-1216, Jul. 2000.

[2] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Information Theory*, IT-49(2):371-381, Feb. 2003.

[3] R. Koetter, and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Networking*, 11(5), Oct. 2003.

[4] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," *41st Allerton Conf. Communication, Control and Computing*, Oct. 2003.

[5] T. Ho, M. Médard, J. Shi, M. Effros and D. R. Karger, "On randomized network coding," *41st Allerton Annual Conference on Communication, Control, and Computing*, 2003.

[6] T. Ho, R. Koetter, M. Médard, D. R. Karger and M. Effros, "The benefits of coding over routing in a randomized setting", *Int'l Symp. Information Theory (ISIT)*, 2003.

[7] Y. Wu, P. A. Chou, Q. Zhang, K. Jain, W. Zhu, and S.-Y. Kung, "Network planning in wireless ad hoc networks: a cross-layer approach," *IEEE J. Selected Areas in Comm.*, Jan. 2005.

[8] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," submitted to *IEEE Trans. Communications*, Mar. 2004. A conference version also appeared in *IEEE Information Theory Workshop*, Oct. 2004.

[9] D. S. Lun, M. Médard, T. Ho, and R. Koetter, "Network coding with a cost criterion," *Int'l Symp. Information Theory and its Applications (ISITA2004)*, Oct. 2004.