

# Optimal Solution for the Index Coding Problem Using Network Coding over $GF(2)$

Speaker: Dr. Chuan Heng FOH

Assistant Professor  
School of Computer Engineering  
Nanyang Technological University  
Singapore



# Overview

- Index Coding Problem
- Network Coding as an attractive technique for Index Coding Problem
- Our solution: “**BENEFIT**” for Index Coding with side information
- Our solution: “**Triangular Network Coding**” for Index Coding without side information
- Conclusion and Future Work

# The Index Coding Problem

Consider a transmitter carrying a series of bits

$$\mathbf{P} = \{c_1, c_2, \dots, c_M\}$$

to transmit over a noiseless channel to a set of receivers

$$\mathbf{R} = \{R_1, R_2, \dots, R_N\}$$

where each receiver wants some information,

$$\mathbf{W}(R_i) \subseteq \mathbf{P} .$$

and each of them already has some other information:

$$\mathbf{H}(R_i) \subseteq \mathbf{P} .$$

Task: To deliver the requested information to all receivers.

Problem: How to minimize the length of transmission.

# The Index Coding Problem

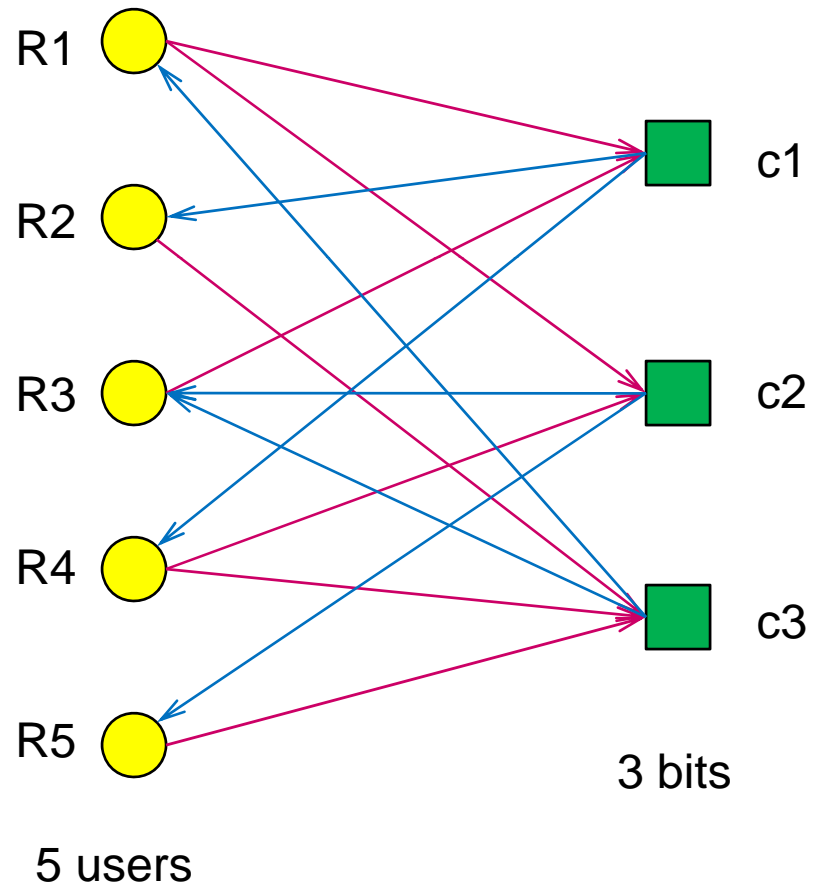
$H(R1)=\{c1,c2\}$   
 $W(R1)=\{c3\}$

$H(R2)=\{c3\}$   
 $W(R2)=\{c1\}$

$H(R3)=\{c1\}$   
 $W(R3)=\{c2,c3\}$

$H(R4)=\{c2,c3\}$   
 $W(R4)=\{c1\}$

$H(R5)=\{c3\}$   
 $W(R5)=\{c2\}$



# Some extensions of the Index Coding Problem

- The sender carries a set of packets instead of a series of bits (i.e. a packet as a basic unit).
- Broadcast transmission for each packet may not always be successful (i.e. erasure channel assumption).
- Special case:  $W(R_i) \cup H(R_i) = P, \forall i$  (i.e. every user is interested in all packets rather than some).
- Special case:  $H(R_i) = \emptyset, \forall i$  (i.e. all users have no side information at the beginning of the process).

# Applications

The index coding problem is a fundamental transmission problem which occurs in almost all multicast transmission scenarios.

- P2P file distribution
- Content distribution network
- Opportunistic routing
- Wireless one-hop multicasting ←our interest
- Satellite communication
- and others...

# The origin of the problem

in INFOCOM 1998.

## Informed-Source Coding-On-Demand (ISCOD) over Broadcast Channels

Yitzhak Birk and Tomer Kol  
Electrical Engr. Dept, Technion  
Haifa 32000, ISRAEL  
*birk@ee, tkol@psl.technion.ac.il*

*Abstract*— We present the Informed-Source Coding-On-Demand (ISCOD) approach for efficiently supplying non-identical data from a central server to multiple caching clients through a broadcast channel. The key idea underlying ISCOD is the joint exploitation of the data already cached by each client, the server's full awareness of client-cache contents and client requests, and the fact that each client only needs to be able to derive the items requested by it rather than all the items ever transmitted or even the union of the items requested by the different clients. We present a set of two-phase ISCOD algorithms. The server uses these algorithms to assemble ad-hoc error-correction sets based its knowledge of every client's cache content and of the items requested by it; next, it uses error-correction codes to construct the data that is actually transmitted. Each client uses its cached data and the received supplemental data to derive the items that it has requested. This technique achieves a reduction of up to tens of percents in the amount of data that must be transmitted in order for every client to be able to derive the data requested by it. Finally, we define  $k$ -partial cliques in a directed graph, and cast the

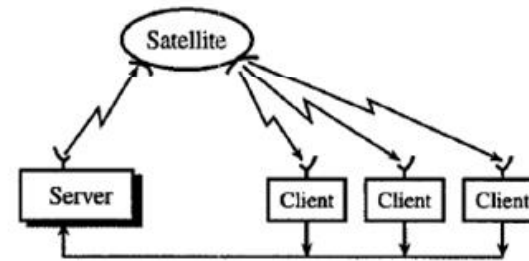


Fig. 1. Data disseminated by a server to caching clients via a broadcast channel. A separate, usually much slower, return channel, is also included.

forward broadcast channel can be utilized effectively. The cost of

# Key Development

in IEEE Symposium on Foundations of Computer Science (FOCS), October 2006

## Index Coding with Side Information

Ziv Bar-Yossef\*

Yitzhak Birk†

T. S. Jayram‡

Tomer Kol§

### Abstract

*Motivated by a problem of transmitting data over broadcast channels (Birk and Kol, INFOCOM 1998), we study the following coding problem: a sender communicates with  $n$  receivers  $R_1, \dots, R_n$ . He holds an input  $x \in \{0, 1\}^n$  and wishes to broadcast a single message so that each receiver  $R_i$  can recover the bit  $x_i$ . Each  $R_i$  has prior side information about  $x$ , induced by a directed graph  $G$  on  $n$  nodes;  $R_i$  knows the bits of  $x$  in the positions  $\{j \mid (i, j) \text{ is an edge of } G\}$ . We call encoding schemes that achieve this goal INDEX codes for  $\{0, 1\}^n$  with side information graph  $G$ .*

*side information* about  $x$ , before it is sent. Source coding with side information addresses encoding schemes that exploit the side information in order to reduce the length of the code. Classical results in this area [16, 19, 18] describe how to achieve optimal rates with respect to the joint entropy of the source and the side information.

Witsenhausen [17] initiated the study of the zero-error side information problem. For every source input  $x \in \mathcal{X}$ , the receiver gets an input  $y \in \mathcal{Y}$  that gives some information about  $x$ . This is captured by restricting the pairs  $(x, y)$  to belong to a fixed set  $\mathcal{L} \subseteq \mathcal{X} \times \mathcal{Y}$ . Both the sender and the receiver know  $\mathcal{L}$ , and thus each of them, given his own input, has information about the other's



# Research Opportunities

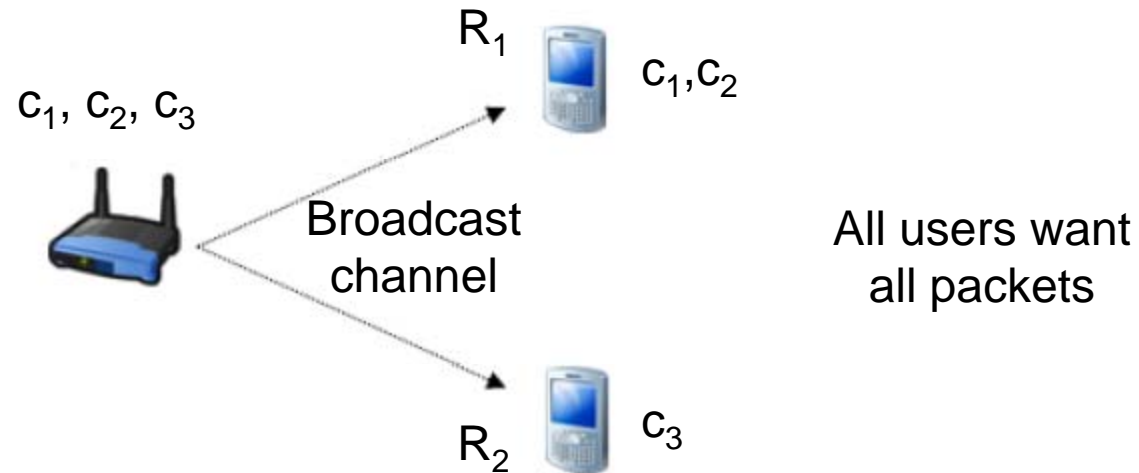
Theoretical study:

- Bounds on the optimal length of INDEX codes.
- Methods to find bounds given a particular setup.

Coding design:

- Finding an efficient (computation & bandwidth) index coding scheme over  $GF(2^q)$ 
  - Network Coding approach when  $q > 1$
  - XOR Coding approach when  $q = 1$  ←focus of this talk

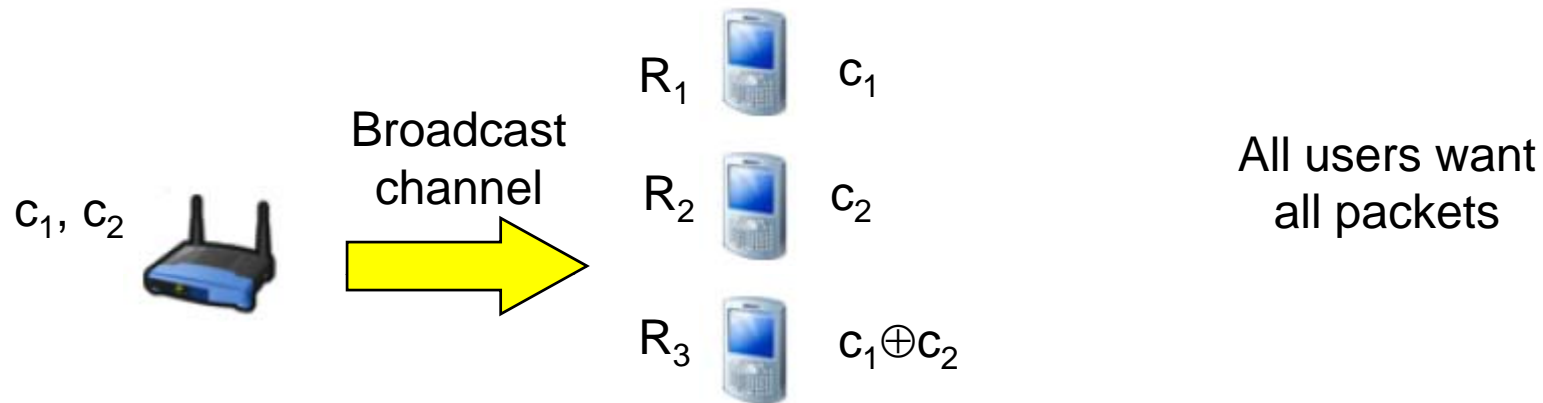
# XOR Coding Approach



To transmit  $c_3$  to  $R_1$ ,  $c_1$  and  $c_2$  to  $R_2$ , in traditional approach, the sender will transmit  $c_1$ ,  $c_2$ , and  $c_3$  separately.

With XOR coding, the sender can now transmit:  
 $c_1$  and then  $c_2 \oplus c_3$ , which reduces one transmission.

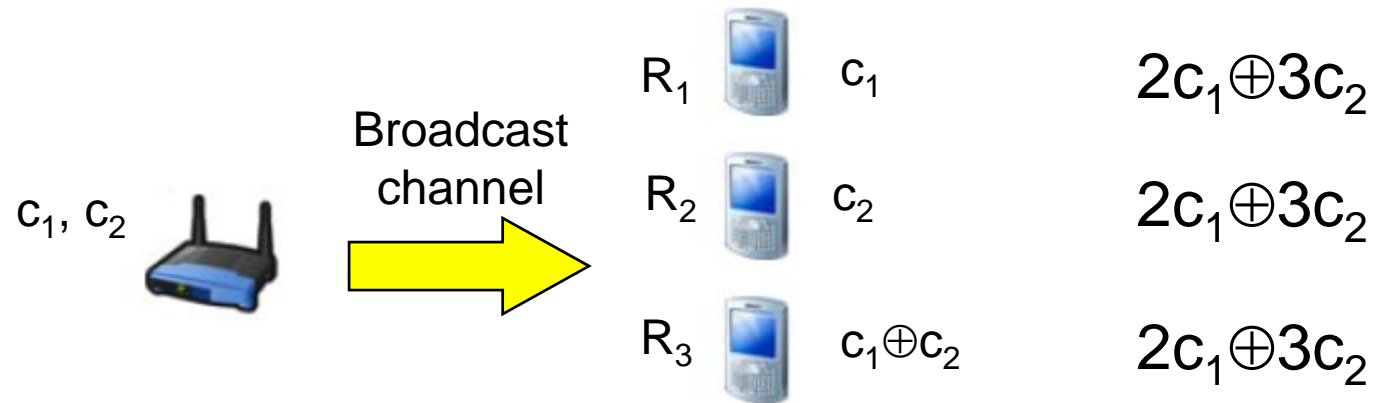
# Network Coding Approach



XOR Coding fails to provide an efficient solution. A single transmission of either  $c_1$  or  $c_2$  or  $c_1 \oplus c_2$  cannot deliver all wanted packets to all users. None of the transmission is **innovative**.

However, we can still transmit an innovative packet by using Network Coding.

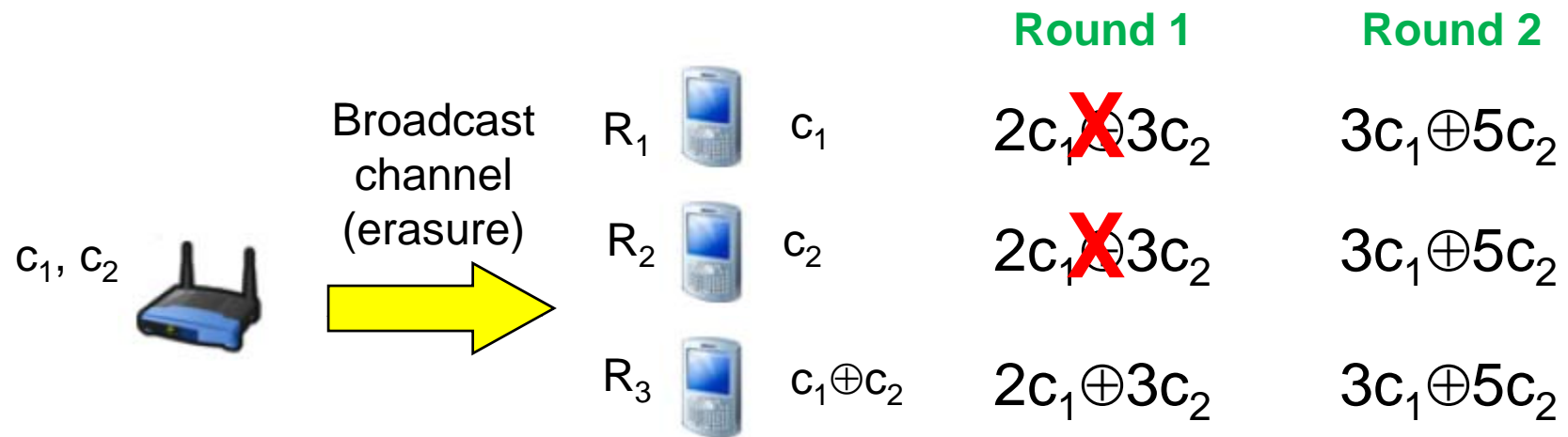
# Random Linear Network Coding (RLNC)



Let's use Random Linear Network Coding (RLNC):  
Broadcast " $2c_1 \oplus 3c_2$ "

" $2c_1 \oplus 3c_2$ " is innovative to all users.

# Erasure Channel Consideration



- After Round 1, only  $R_3$  can recover all packets.
- Based on the feedback, the source then encode another packet and broadcast to all users.
- Assuming no transmission error in Round 2, all users can now decode and recover all packets.

# Various Network Coding Techniques for Index Coding

For  $GF(2^q)$  where  $q > 1$

- Deterministic Linear Network Coding (DLNC), optimal
- Random Linear Network Coding (RLNC), suboptimal

For  $GF(2)$ , the main challenge is to find which set of packets to XOR together to achieve maximum information delivery

- Some heuristics: Sparsest set clustering, color saving, etc.
- Sort-by-Utility
- BENEFIT ← our solution
- Triangular Network Coding, optimal ← our solution  
(A new kind of Network Coding approach using both finite-field and real-field operations)

# Focus of Our Problem

- Aim: To find an efficient Index Coding solution for wireless multicasting of a set of packets
- Assumptions:
  - No side information at the beginning
  - Erasure broadcast channel
- Constraints:
  - Limited bandwidth (due to wireless channel).
  - Limited computational power (due to mobile devices).  
We focus on GF(2).

# Why GF(2)?

## Battery energy cost

- RLNC over GF(256) for iPod Touch has shown that packet encoding and decoding can account for up to **33% of the battery energy consumption**.
- Another study shows that XOR-encoding of 2 packets, each 1000 bytes long only consumes 191 nJ of energy. Given that transmission of a packet of the same length over IEEE 802.11 network on Nokia N95 consumes 2.31mJ of energy, the overall energy cost of XOR-coding has no apparent effect (**0.008%**) on the total energy cost of encoding and transmitting a XOR coded packet.




# Why GF(2)?


## Encoding and decoding throughput

- Encoding over GF(2) is approximately 8 times faster than encoding over GF(256) on iPhone 3G implementation. Similarly decoding over GF(2) is approximately 6 times faster than decoding over GF(256) on the same testbed.


# GF(2) versus GF(2<sup>q</sup>)


- Computational complexity

 GF(2) offers much lower encoding, decoding computational complexity.

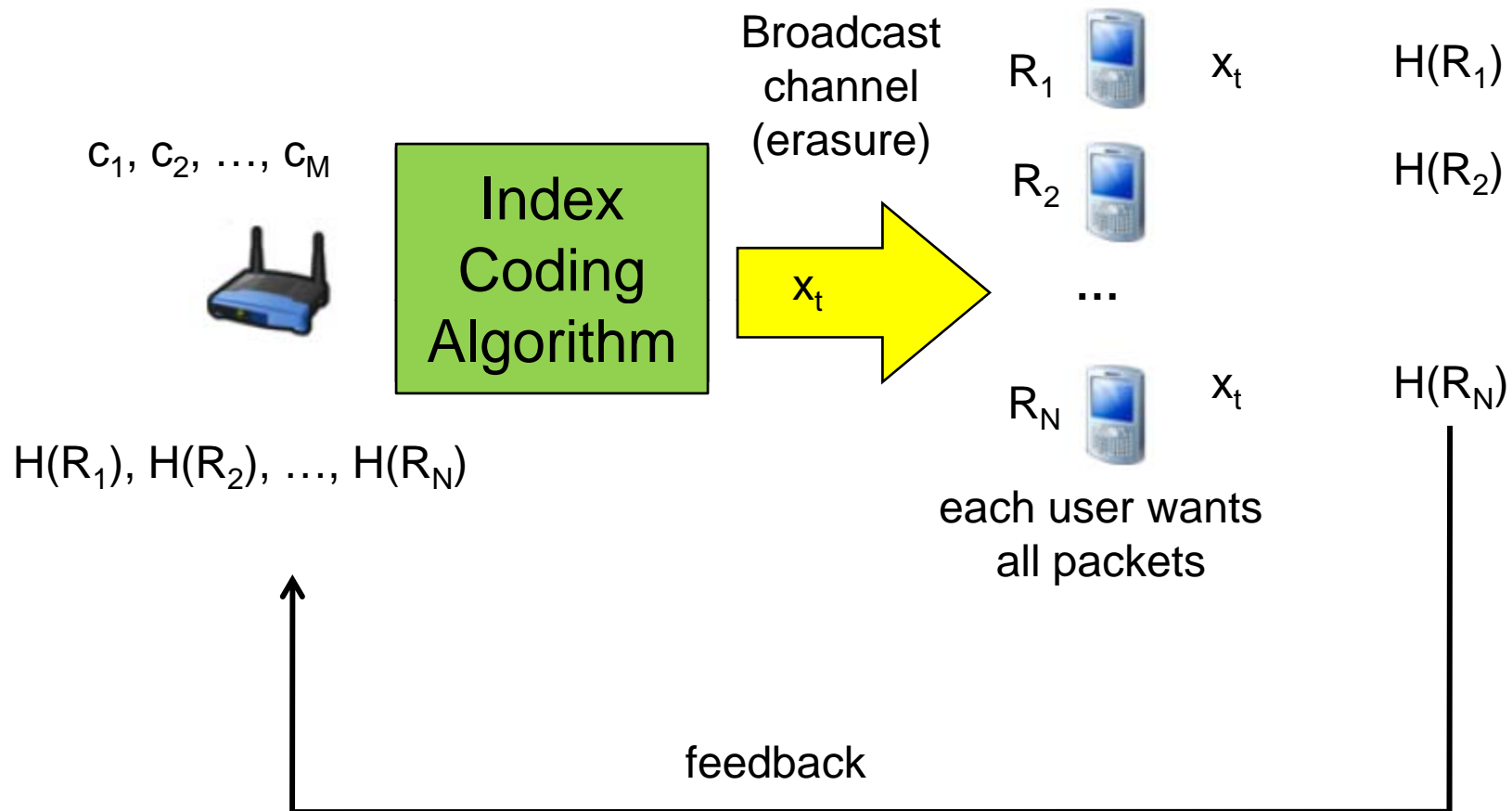
 LNC coding involves addition, multiplication and Gaussian elimination. Much higher encoding, decoding computational complexity.

- Throughput

 For GF(2), optimal solution is NP-complete. Throughput performance degrades with increasing network size and packet error probability.

 For LNC, when field size is larger than or equal to the number of receivers, an innovative packet can always be found in polynomial time.

# Our Scenario



# Index Coding Iterations

- The source maintains a matrix describing missing packets of each user ('1' indicates missing):

	c1	c2	c3	c4	c5	← packets
Users →	R1	1	1	0	0	1
	R2	0	1	0	1	0
	R3	0	1	1	0	0
	R4	1	0	0	1	1

- The source executes the Index Coding Algorithm to decide which set of packets to encode using XOR.
- After transmitting the encoded packet, the source collects feedback from all users.
- Based on the feedback, the source updates the matrix and perform Index Coding Algorithm again.

# Raw Transmissions (or no coding)

- After the systematic transmission, the source collects feedback and constructs the following matrix:

$$\begin{array}{c} \text{R1} \\ \text{R2} \\ \text{R3} \\ \text{R4} \end{array} \begin{array}{ccccc} \text{c1} & \text{c2} & \text{c3} & \text{c4} & \text{c5} \\ \left( \begin{array}{ccccc} -10 & 10 & 0 & 0 & -10 \\ 0 & 10 & 0 & -10 & 0 \\ 0 & 10 & -10 & 0 & 0 \\ -10 & 0 & 0 & -10 & -10 \end{array} \right) \end{array}$$

5 transmissions are required

# Sort-by-Utility (appeared in 2007)

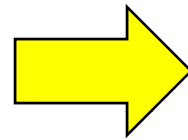
	c1	c2	c3	c4	c5	sort by utility	c2	c1	c4	c5	c3
R1	1	1	0	0	1	➔	<del>1</del> 0	<del>1</del> 0	0	<del>1</del> 0	0
R2	0	1	0	1	0		<del>1</del> 0	0	<del>1</del> 0	0	0
R3	0	1	1	0	0		<del>1</del> 0	0	0	0	<del>1</del> 0
R4	1	0	0	1	1		0	<del>1</del> 0	<del>1</del> 0	<del>1</del> 0	0

- transmit c2
- transmit  $c1 \oplus c3$
- transmit c4
- transmit c5

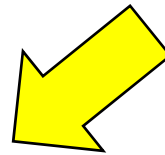
Total: 4 transmissions

# Our Solution: BENEFIT

	c1	c2	c3	c4	c5
R1	1	1	0	0	1
R2	0	-1	0	1	0
R3	0	-1	1	0	0
R4	-1	0	0	1	1



	c1	c2	c3	c4	c5
R1	-1	-1	0	0	1
R2	0	0	0	-1	0
R3	0	0	-1	0	0
R4	0	0	0	-1	1



	c1	c2	c3	c4	c5
R1	0	0	0	0	-1
R2	0	0	0	0	0
R3	0	0	0	0	0
R4	0	0	0	0	-1

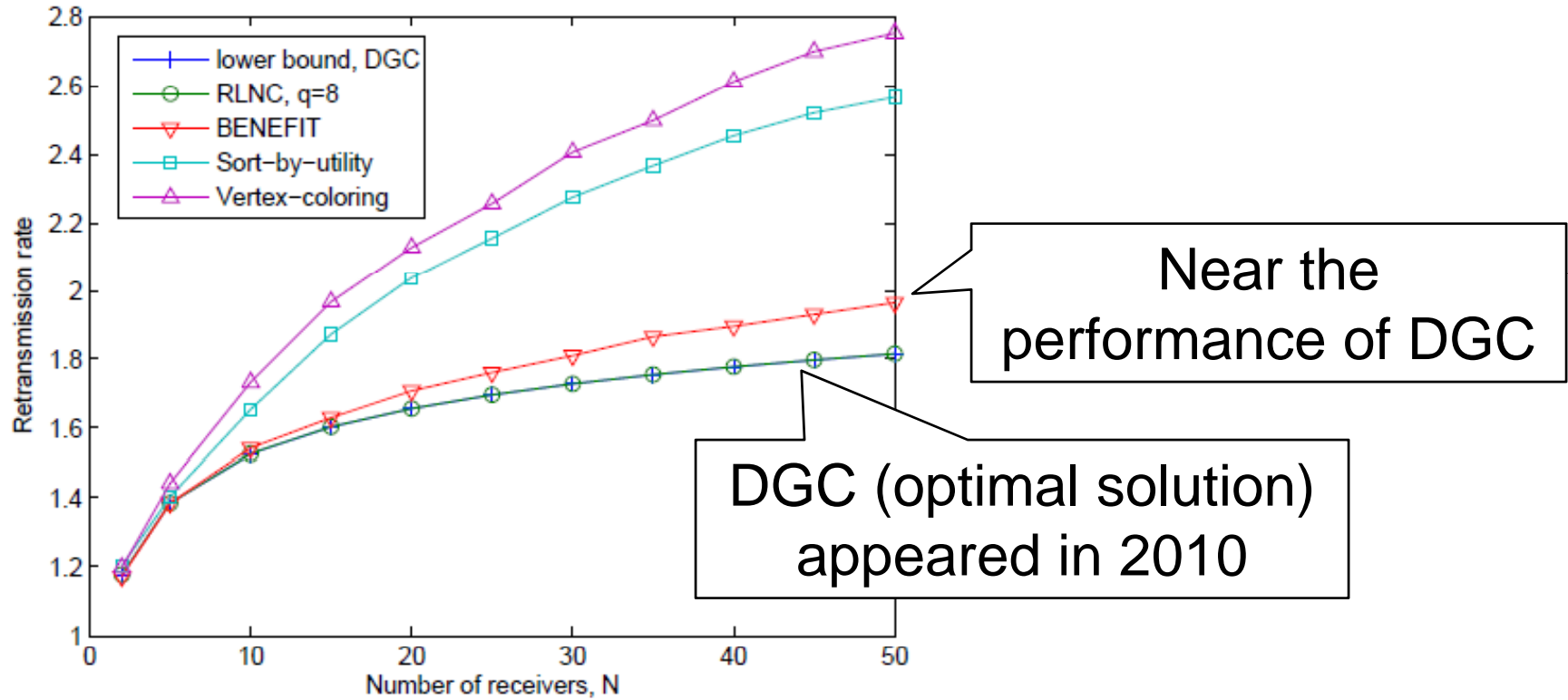
We achieve by using 3 transmissions  
(optimal for this example)

# BENEFIT

- It is a memory-based solution.
- It attempts to find a combination of packets that will be innovative to all users.
- If the coded packet cannot be decoded immediately, the packet will be buffered by the user.
- We found that many of the initially non-decodable packets can be recovered as the index coding iteration progresses.
- Due to buffering, the source does not always need to wait for the immediate feedback to optimize its encoding decision.



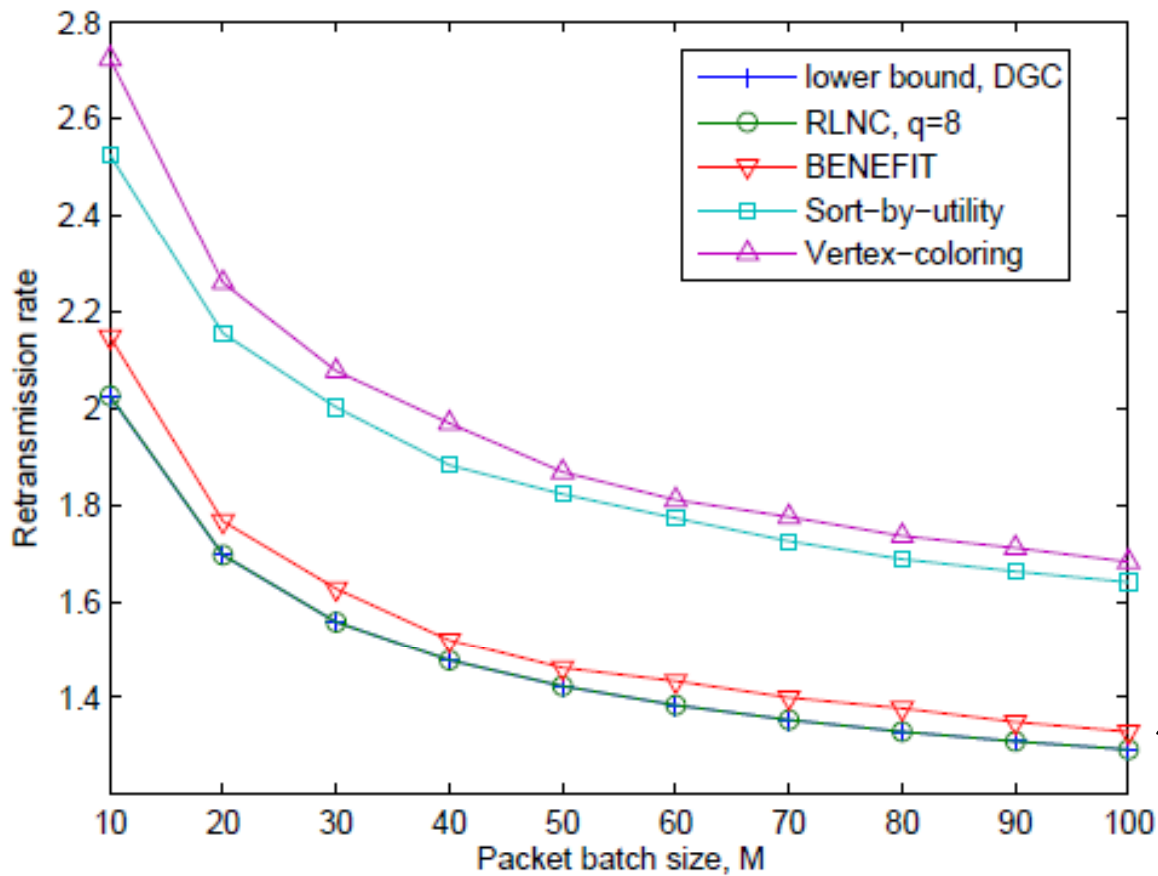
# Results: Varying number of users



(a)  $p=0.5, M=20$ .

NOTE: Dynamic General-coding based scheme (DGC) is an instance of DLNC. It proposes an optimal solution in polynomial-time using LNC where  $q \geq \lceil \log_2 N \rceil$ . <sup>25</sup>

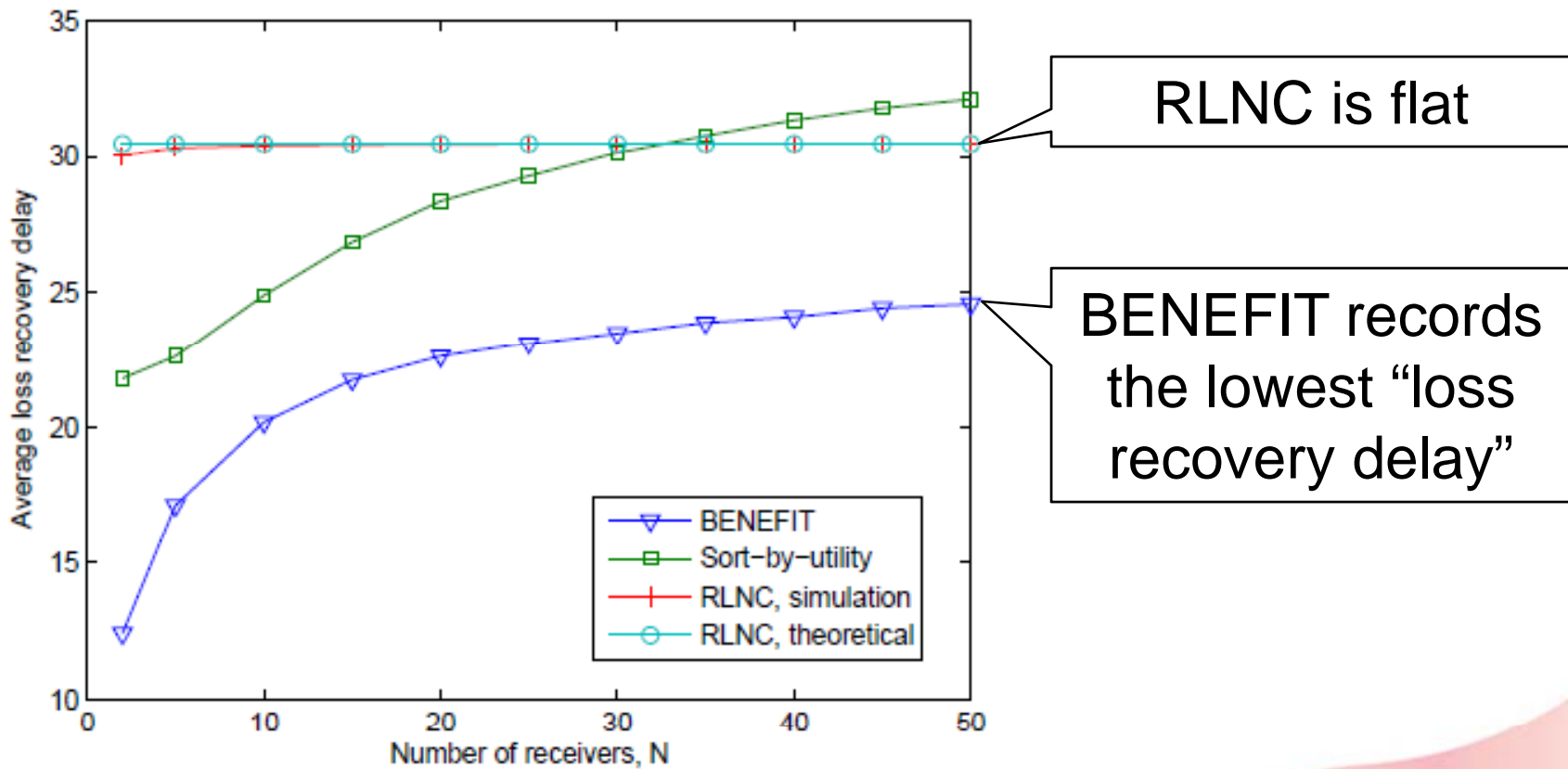
# Results: Varying packet batch size



Near the performance of DGC (optimal)

(c)  $p=0.5, N=25$ .

# Results: Delay



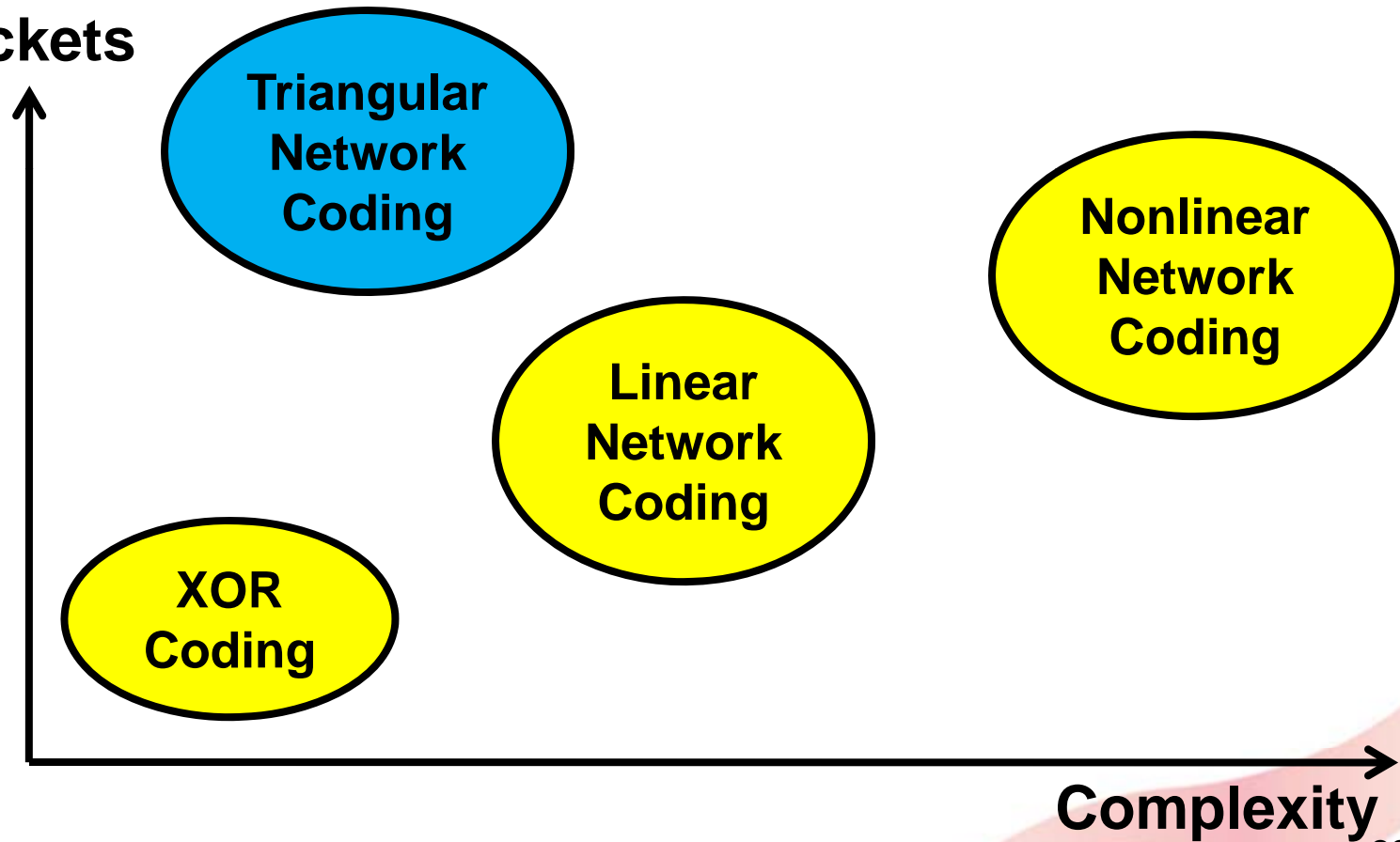
(a)  $p=0.5$ ,  $M=20$ .

## Can we do better?

- There is still a gap between BENEFIT and the optimal solution (RLNC or DGC).
- Can we narrow the gap yet maintaining low computational complexity?
- ANSWER: “**Triangular Network Coding**”
  - Its uniqueness: mixing of finite-field (XOR) and real-field (MUL by SHIFTING) operations.
  - Main benefits: offers endless supply of linearly independent encoded packets and allows simple decoding (XOR only).
  - Suitable for Index Coding with no side information.

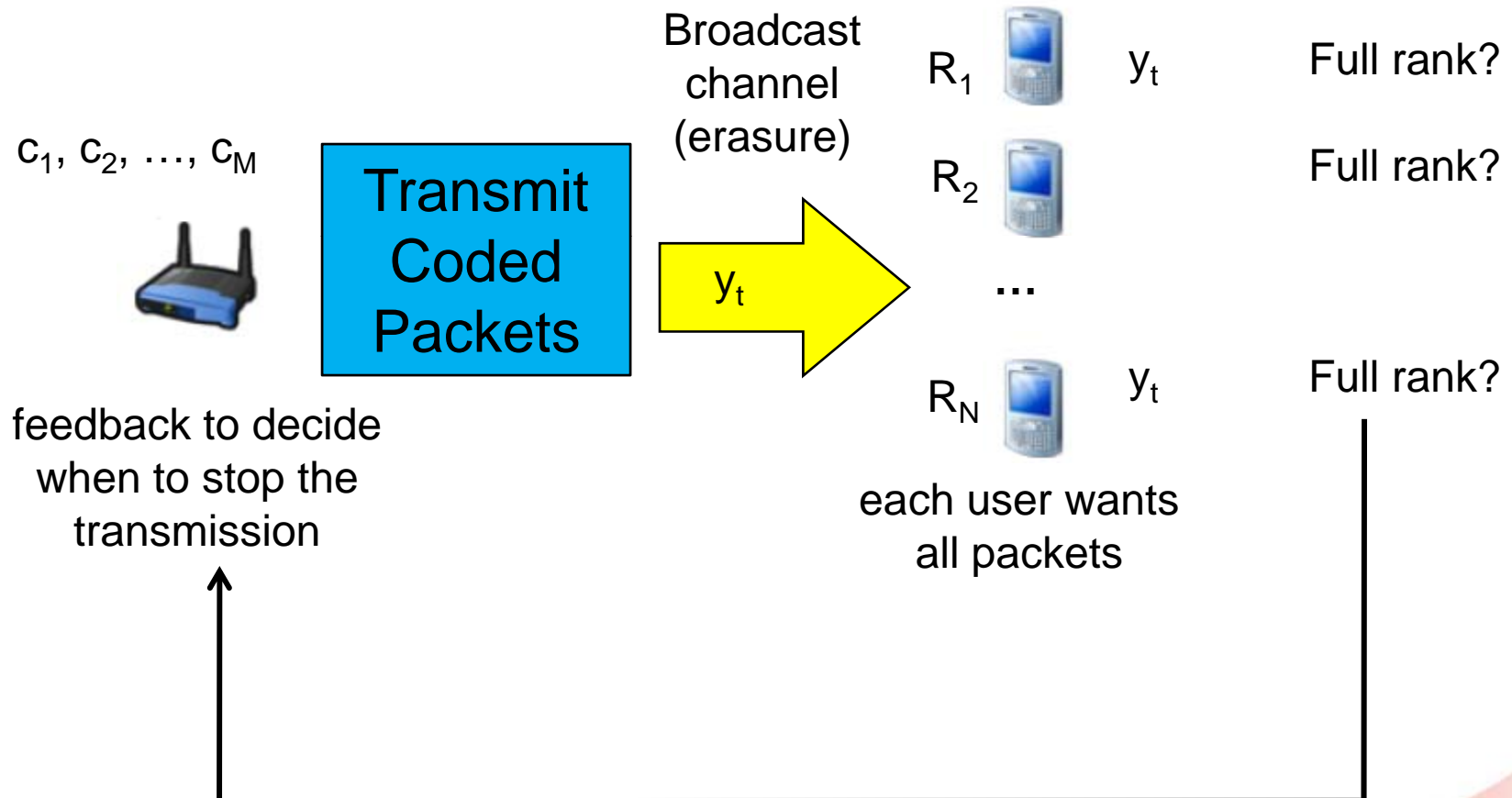
# Triangular Network Coding

Supply of linearly independent packets



Complexity

# Our Scenario: Index Coding without Side Information



# Triangular Network Coding

## Traditional XOR coding

$b_{1,1}$	$b_{2,1}$	$b_{3,1}$	$b_{4,1}$ ...
$b_{1,2}$	$b_{2,2}$	$b_{3,2}$	$b_{4,2}$ ...
$b_{1,3}$	$b_{2,3}$	$b_{3,3}$	$b_{4,3}$ ...
$b_{1,4}$	$b_{2,4}$	$b_{3,4}$	$b_{4,4}$ ...

$x_{1,a}$	$x_{2,a}$	$x_{3,a}$	$x_{4,a}$ ...
-----------	-----------	-----------	---------------

$$x_{j,a} = b_{j,1} \oplus b_{j,2} \oplus b_{j,3} \oplus b_{j,4}$$

## Triangular Network Coding

0	$b_{1,1}$	$b_{2,1}$	$b_{3,1}$	$b_{4,1}$ ...	...	$b_{B,1}$	0	0
$b_{1,2}$	$b_{2,2}$	$b_{3,2}$	$b_{4,2}$	$b_{5,2}$ ...	...	$b_{B,2}$	0	0
0	0	$b_{1,3}$	$b_{2,3}$	$b_{3,3}$	$b_{4,3}$ ...	...	$b_{B,3}$	0
0	0	0	$b_{1,4}$	$b_{2,4}$	$b_{3,4}$	$b_{4,4}$ ...	...	$b_{B,4}$

packet c1  
packet c2  
packet c3  
packet c4

$x_{1,a}$	$x_{2,a}$	$x_{3,a}$	$x_{4,a}$	$x_{5,a}$	$x_{6,a}$	$x_{7,a}$ ...
-----------	-----------	-----------	-----------	-----------	-----------	---------------

coded packet

$$x_{j,a} = b_{j-1,1} \oplus b_{j,2} \oplus b_{j-2,3} \oplus b_{j-3,4}$$

## Triangular Network Coding offers:

- **Efficient Decoding** – Back-substitution procedure and XOR operation (next slide).
- **Linear independency** – All the coded packets are always linearly independent of the previously generated coded packet.

## Back-Substitution Procedure

				0	0	c1
0					0	c2
0	0					c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0				0	0	c1
0	0			0	0	c2
0	0					c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0					c1
0					0	c2
				0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

Initial matrix. '0's filled up using information from coded packet's header.

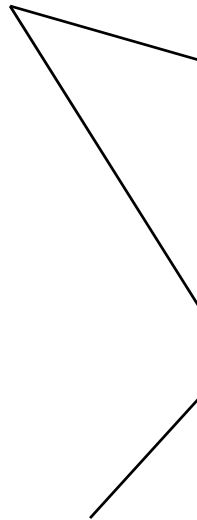


Solve

0				0	0	c1
0					0	c2
0	0					c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0				0	c1
0	0			0	0	c2
0	0					c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0				c1
0					0	c2
				0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

Substitute

Substitute



0				0	0	c1
0	1				0	c2
0	0					c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0				0	c1
1				0	0	c2
0	0					c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0				c1
0	1				0	c2
				0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

Substitute

0				0	0	c1
0	1				0	c2
0	0	0				c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0					c1
1				0	0	c2
0	0	0				c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0				c1
0	1				0	c2
0				0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

Solve

0	0			0	0	c1
0	1				0	c2
0	0	0				c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0			0	c1
1				0	0	c2
0	0	0				c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0			c1
0	1				0	c2
0				0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0			0	0	c1
0	1	0			0	c2
0	0	0				c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0			0	c1
1	0			0	0	c2
0	0	0				c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0			c1
0	1	0			0	c2
0				0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0			0	0	c1
0	1	0			0	c2
0	0	0	1			c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0			0	c1
1	0			0	0	c2
0	0	0	1			c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0			c1
0	1	0			0	c2
0	1			0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0	0		0	0	c1
0	1	0			0	c2
0	0	0	1			c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0	0		0	c1
1	0			0	0	c2
0	0	0	1			c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0	0		c1
0	1	0			0	c2
0	1			0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0	0		0	0	c1
0	1	0	0		0	c2
0	0	0	1			c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0	0		0	c1
1	0	0		0	0	c2
0	0	0	1			c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0	0		c1
0	1	0	0		0	c2
0	1			0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information



0	0	0		0	0	c1
0	1	0	0		0	c2
0	0	0	1	1		c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0	0		0	c1
1	0	0		0	0	c2
0	0	0	1	1		c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0	0		c1
0	1	0	0		0	c2
0	1	1		0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0	0	1	0	0	c1
0	1	0	0		0	c2
0	0	0	1	1		c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0	0	1	0	c1
1	0	0		0	0	c2
0	0	0	1	1		c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0	0	1	c1
0	1	0	0		0	c2
0	1	1		0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0	0	1	0	0	c1
0	1	0	0	1	0	c2
0	0	0	1	1		c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0	0	1	0	c1
1	0	0	1	0	0	c2
0	0	0	1	1		c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0	0	1	c1
0	1	0	0	1	0	c2
0	1	1		0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

0	0	0	1	0	0	c1
0	1	0	0	1	0	c2
0	0	0	1	1	1	c3
0	1	0	0	0	1	y1
					(0,1,2)	y3 header information
0	0	0	0	1	0	c1
1	0	0	1	0	0	c2
0	0	0	1	1	1	c3
1	0	0	0	0	1	y2
					(1,0,2)	y3 header information
0	0	0	0	0	1	c1
0	1	0	0	1	0	c2
0	1	1	1	0	0	c3
0	0	1	1	1	1	y3
					(2,1,0)	y3 header information

Solved

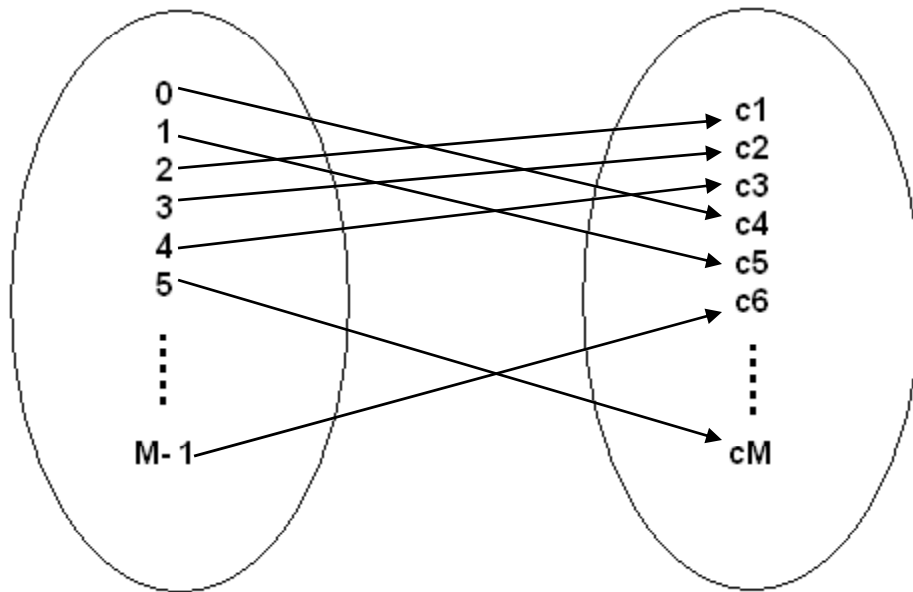
# Triangular Network Coding

Design a coding scheme such that

- The generated coded packets are **back-substitution** ready.
- The generated coded packets are **linearly independent**.
- **Endless supply** of linearly independent coded packets.
- The source does not need feedback about the state of each user.

Triangular Network Coding trades space for both the decoding computational complexity (near XOR) and performance (optimal).

# Bijjective Function Mapping: Natural Number



The number of redundant '0' bits added at the head of the packet

Corresponding packet id.

$b_{1,1}$	$b_{2,1}$	$b_{3,1}$	$b_{4,1}$	...	...	$b_{B,1}$	0	0	0
0	$b_{1,2}$	$b_{2,2}$	$b_{3,2}$	$b_{4,2}$	$b_{5,2}$	...	$b_{B,2}$	0	0
0	0	$b_{1,3}$	$b_{2,3}$	$b_{3,3}$	$b_{4,3}$	...	...	$b_{B,3}$	0
0	0	0	$b_{1,4}$	$b_{2,4}$	$b_{3,4}$	$b_{4,4}$	...	...	$b_{B,4}$

Packet id (0,1,2,3)  
or:  $8c_1 + 4c_2 + 2c_3 + c_4$

0	$b_{1,1}$	$b_{2,1}$	$b_{3,1}$	$b_{4,1}$	...	...	$b_{B,1}$	0	0
$b_{1,2}$	$b_{2,2}$	$b_{3,2}$	$b_{4,2}$	$b_{5,2}$	...	...	$b_{B,2}$	0	0
0	0	$b_{1,3}$	$b_{2,3}$	$b_{3,3}$	$b_{4,3}$	...	...	$b_{B,3}$	0
0	0	0	$b_{1,4}$	$b_{2,4}$	$b_{3,4}$	$b_{4,4}$	...	...	$b_{B,4}$

Packet id (1,0,2,3)  
or:  $4c_1 + 8c_2 + 2c_3 + c_4$

Such mapping always generates a triangular pattern, which is solvable using back-substitution.

# Back-Substitution Ready, but not always Full Rank ☹️

For 4 packets  $c_1, \dots, c_4$ , we can have:

- $8c_1 + 4c_2 + 2c_3 + c_4$      $\text{id}=(0,1,2,3)$
- $8c_1 + 4c_2 + c_3 + 2c_4$      $\text{id}=(0,1,3,2)$
- $8c_1 + 2c_2 + 4c_3 + c_4$      $\text{id}=(0,2,1,3)$
- $8c_1 + c_2 + 4c_3 + 2c_4$      $\text{id}=(0,3,1,2)$
- $8c_1 + 2c_2 + c_3 + 4c_4$      $\text{id}=(0,2,3,1)$
- $8c_1 + c_2 + 2c_3 + 4c_4$      $\text{id}=(0,3,2,1)$
- $2c_1 + c_2 + 4c_3 + 8c_4$      $\text{id}=(2,3,1,0)$
- ...

rank < 4

# Generation Function, always Full Rank

- Group 1  
(0,1,2,3), (0,3,1,2), (0,2,3,1)
- Group 2  
(1,0,2,3), (3,0,1,2), (2,0,3,1)
- Group 3  
(1,2,0,3), (3,1,0,2), (2,3,0,1)
- Group 4  
(1,2,3,0), (3,1,2,0), (2,3,1,0)

M(M-1) of coded packets.

In this example, M=4 which gives 12 linearly independent equations.

---

*Lemma 1:* Consider a system of  $M$  packets. Given an encoded packet  $y_a$  with an id of  $(0, 1, 2, 3, \dots, M - 1)_a$ , the coefficient matrix formed by all  $M - 1$  encoded packets in the same group of  $y_a$  including  $y_a$  gives a rank of  $M - 1$ .

*Lemma 2:* Consider a system of  $M$  packets. A collection of all  $M - 1$  encoded packets in the same group and an encoded packet from a different group gives a rank of  $M$ .



# More Generation Functions

- Group 2-1  
(0,2,4,6), (0,6,2,4), (0,4,6,2)
- Group 2-2  
(2,0,4,6), (6,0,2,4), (4,0,6,2)
- Group 2-3  
(2,4,0,6), (6,2,0,4), (4,6,0,2)
- Group 2-4  
(2,4,6,0), (6,2,4,0), (4,6,2,0)

} another  $M(M-1)$  of coded packets

- 
- Group 3-1  
(0,3,6,9), ...
  - and so on...

} more  $M(M-1)$  of coded packets

# Summary

- We have proposed two efficient GF(2) coding algorithms for wireless multicasting
  - BENEFIT (near optimal) for Index Coding with side information
  - Triangular Network Coding (optimal) for Index Coding without side information
- Future works
  - Triangular Network Coding assumes  $H(R_i) = \emptyset$  (no side information). We're developing an extension to lift this limitation.
  - Apply Triangular Network Coding beyond Index Coding Problem.

**Thank You**

