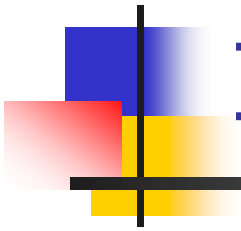
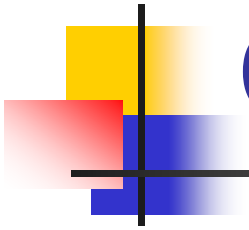


Applications and New Directions: Practical Network Coding for Internet and Wireless Applications

Philip A. Chou

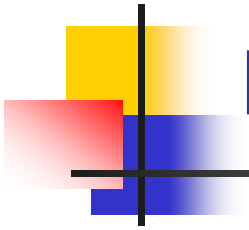
ISIT Tutorial, September 4, 2005



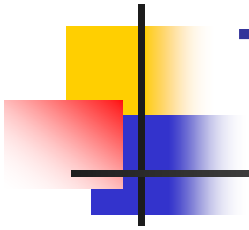
Outline

- Making Network Coding Practical
  - Packetization
  - Buffering
- Internet and Wireless Applications
  - Live Broadcasting, File Downloading, Storage, Messaging, Interactive Communication, Sensor Networks

Network Coding applicable to real networks?



- Internet
  - IP Layer
    - Routers (e.g., ISP)
  - Application Layer
    - Infrastructure (e.g., CDN)
    - Ad hoc (e.g., P2P)
- Wireless
  - Mobile multihop ad hoc wireless networks
  - Stationary wireless (residential) mesh networks
  - Sensor networks



Theory vs. Practice

- Theory:
  - Symbols flow synchronously throughout network
  - Edges have unit (or known integer) capacities
  - Centralized knowledge of topology assumed to compute encoding and decoding functions
- Practice:
  - Information travels asynchronously in packets
  - Packets subject to random delays and losses
  - Edge capacities often unknown, time-varying
  - Difficult to obtain centralized knowledge, or to arrange reliable broadcast of functions
  - Need simple technology, applicable in practice

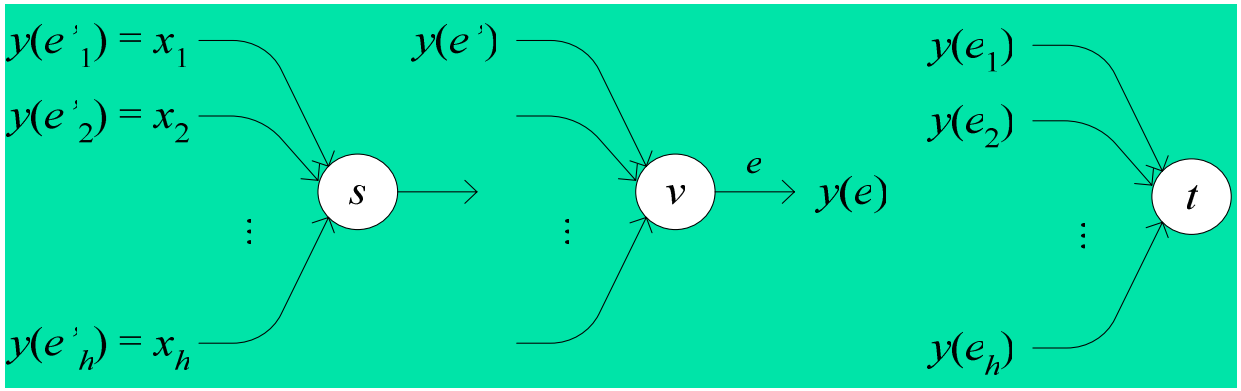
Making Network Coding Practical

- Packetization
  - Header removes need for centralized knowledge of graph topology and encoding/decoding functions
- Buffering
  - Allows asynchronous packets arrivals & departures with arbitrarily varying rates, delay, loss

[Chou, Wu, and Jain; Allerton 2003]

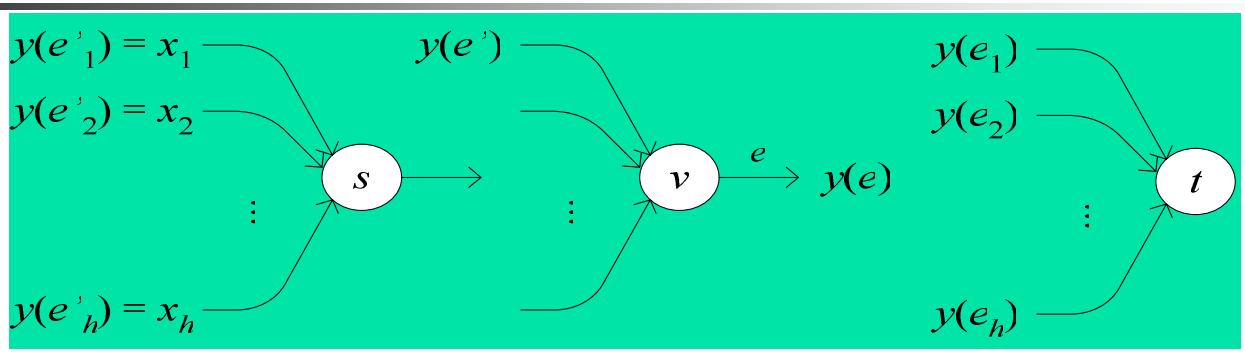
Algebraic Framework

- Graph (V,E) having unit capacity edges
- Sender s in V, set of receivers $T=\{t,\ldots\}$ in V
- Broadcast capacity $h = \min_t \text{Maxflow}(s,t)$



- $y(e) = \sum_{e'} \beta_e(e')\, y(e')$
- $\beta(e) = [\beta_e(e')]_{e'}$ is *local encoding vector*

Global Encoding Vectors



- By induction $y(e) = \sum_{i=1}^{h} g_i(e)\, x_i$
- $\mathbf{g}(e) = [g_1(e),\ldots,g_h(e)]$ is *global encoding vector*
- Receiver t can recover $x_1,\ldots,x_h$ from

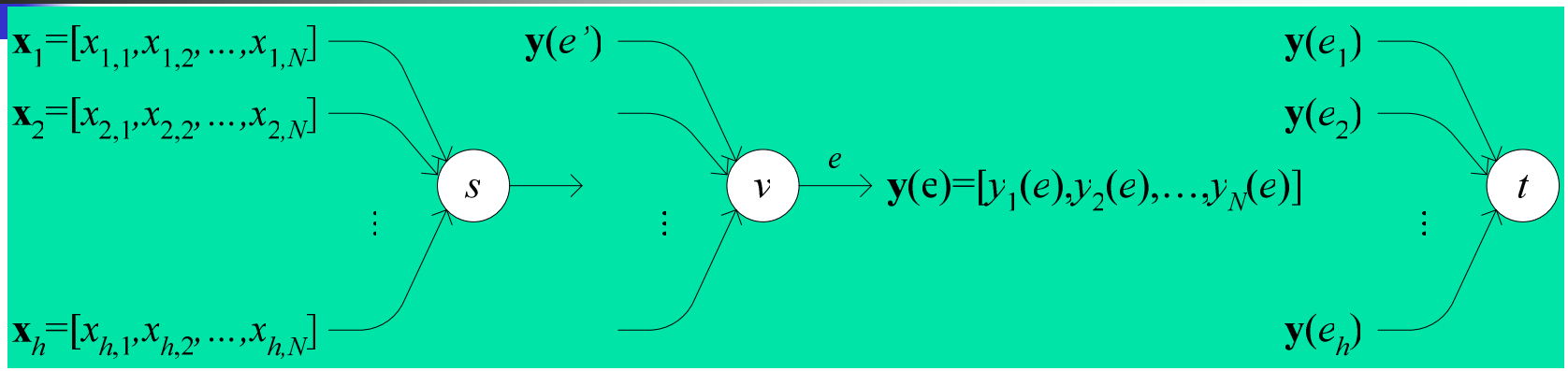
$$\begin{bmatrix} y(e_1) \\ \mathbb{M} \\ y(e_h) \end{bmatrix} = \begin{bmatrix} g_1(e_1) & \mathsf{L} & g_h(e_1) \\ \mathbb{M} & \mathsf{O} & \mathbb{M} \\ g_1(e_h) & \mathsf{L} & g_h(e_h) \end{bmatrix} \begin{bmatrix} x_1 \\ \mathbb{M} \\ x_h \end{bmatrix} = G_t \begin{bmatrix} x_1 \\ \mathbb{M} \\ x_h \end{bmatrix}$$

Invertibility of G_t

- G_t will be invertible with high probability if local encoding vectors are random and field size is sufficiently large
  - If field size = 2^{16} and |E| = 2^8 then G_t will be invertible w.p. ≥ $1-2^{-8} = 0.996$

[Ho, Koetter, Médard, Karger, and Effros; ISIT 2003]
[Jaggi, Sanders, Chou, Effros, Egner, Jain, and Tolhuizen; Trans IT 2005]

Packetization



- Internet: MTU size typically ≈ 1400+ bytes
- $\mathbf{y}(e) = \sum_{e'} \beta_e(e')\,\mathbf{y}(e') = \sum_{i=1}^{h} g_i(e)\,\mathbf{x}_i$ s.t.

$$\begin{bmatrix} \mathbf{y}(e_1) \\ \vdots \\ \mathbf{y}(e_h) \end{bmatrix} = \begin{bmatrix} y_1(e_1) & y_2(e_1) & \cdots & y_N(e_1) \\ \vdots & \vdots & & \vdots \\ y_1(e_h) & y_2(e_h) & \cdots & y_N(e_h) \end{bmatrix} = G_t \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ \vdots & \vdots & & \vdots \\ x_{h,1} & x_{h,2} & \cdots & x_{h,N} \end{bmatrix}$$

Packet Header

- Include *within each packet* on edge e
  $\mathbf{g}(e) = \sum_{e'} \beta_e(e')\, \mathbf{g}(e')$; $\mathbf{y}(e) = \sum_{e'} \beta_e(e')\, \mathbf{y}(e')$
- Can be accomplished by prefixing i th unit vector to i th source vector $\mathbf{x}_i$, $i=1,\ldots,h$

$$\begin{bmatrix} g_1(e_1) & \cdots & g_h(e_1) & y_1(e_1) & y_2(e_1) & \cdots & y_N(e_1) \\ \vdots & & \vdots & \vdots & \vdots & & \vdots \\ g_1(e_h) & \cdots & g_h(e_h) & y_1(e_h) & y_2(e_h) & \cdots & y_N(e_h) \end{bmatrix} = G_t \begin{bmatrix} 1 & & 0 & x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ & \ddots & & \vdots & \vdots & & \vdots \\ 0 & & 1 & x_{h,h} & x_{h,2} & \cdots & x_{h,N} \end{bmatrix}$$

- Then global encoding vectors needed to invert the code at any receiver can be found in the received packets themselves!

Header Cost vs. Benefit

- Cost:
  - Overhead of transmitting h extra symbols per packet; if $h = 50$ and field size $= 2^8$, then overhead $\approx 50/1400 \approx 3\%$
- Benefit:
  - Receivers can decode even if
    - Network topology & encoding functions unknown
    - Nodes & edges added & removed in ad hoc way
    - Packet loss, node & link failures w/ unknown locations
    - Local encoding vectors are time-varying & random

Erasure Protection

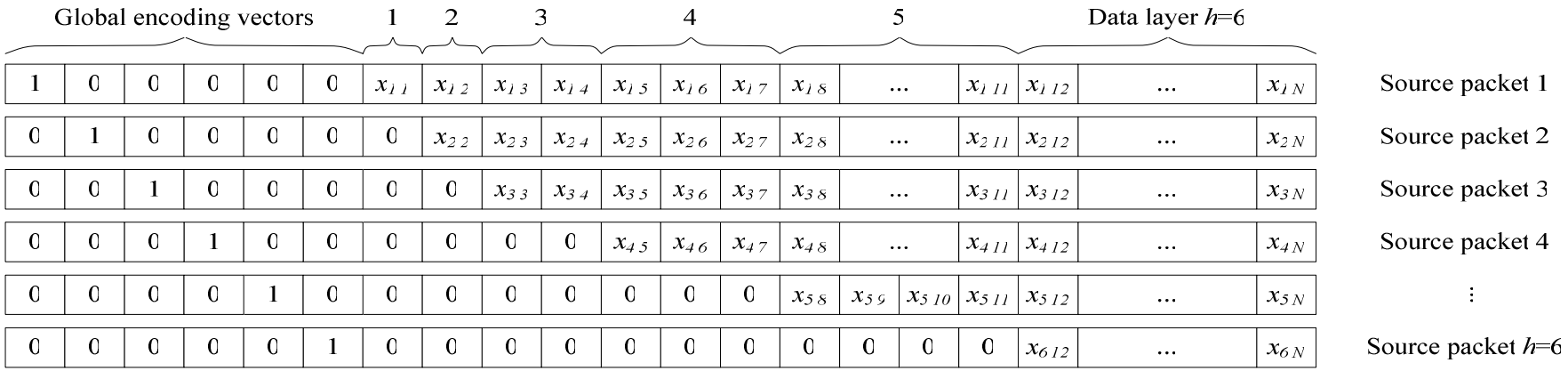
- Removals, failures, losses, poor random encoding may reduce capacity below h

$$\begin{bmatrix} g_1(e_1) & \cdots & g_h(e_1) & y_1(e_1) & y_2(e_1) & \cdots & y_N(e_1) \\ \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ g_1(e_k) & \cdots & g_h(e_k) & y_1(e_k) & y_2(e_k) & \cdots & y_N(e_k) \end{bmatrix} = G_t^{k\times h} \begin{bmatrix} 1 & & 0 & x_{1,1} & x_{1,2} & \cdots & x_{1,N} \\ & \ddots & & \vdots & \vdots & & \vdots \\ 0 & & 1 & x_{h,h} & x_{h,2} & \cdots & x_{h,N} \end{bmatrix}$$

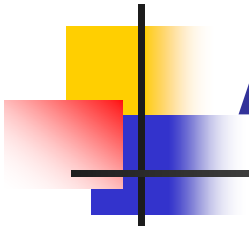
- Basic form of erasure protection: send redundant packets, e.g., last $h-k$ packets of $\mathbf{x}_1,\ldots,\mathbf{x}_h$ are known zero

Priority Encoding Transmission: unequal error protection

- More sophisticated form: partition data into layers of importance, vary redundancy by layer
- Received rank $k \rightarrow$ recover k layers



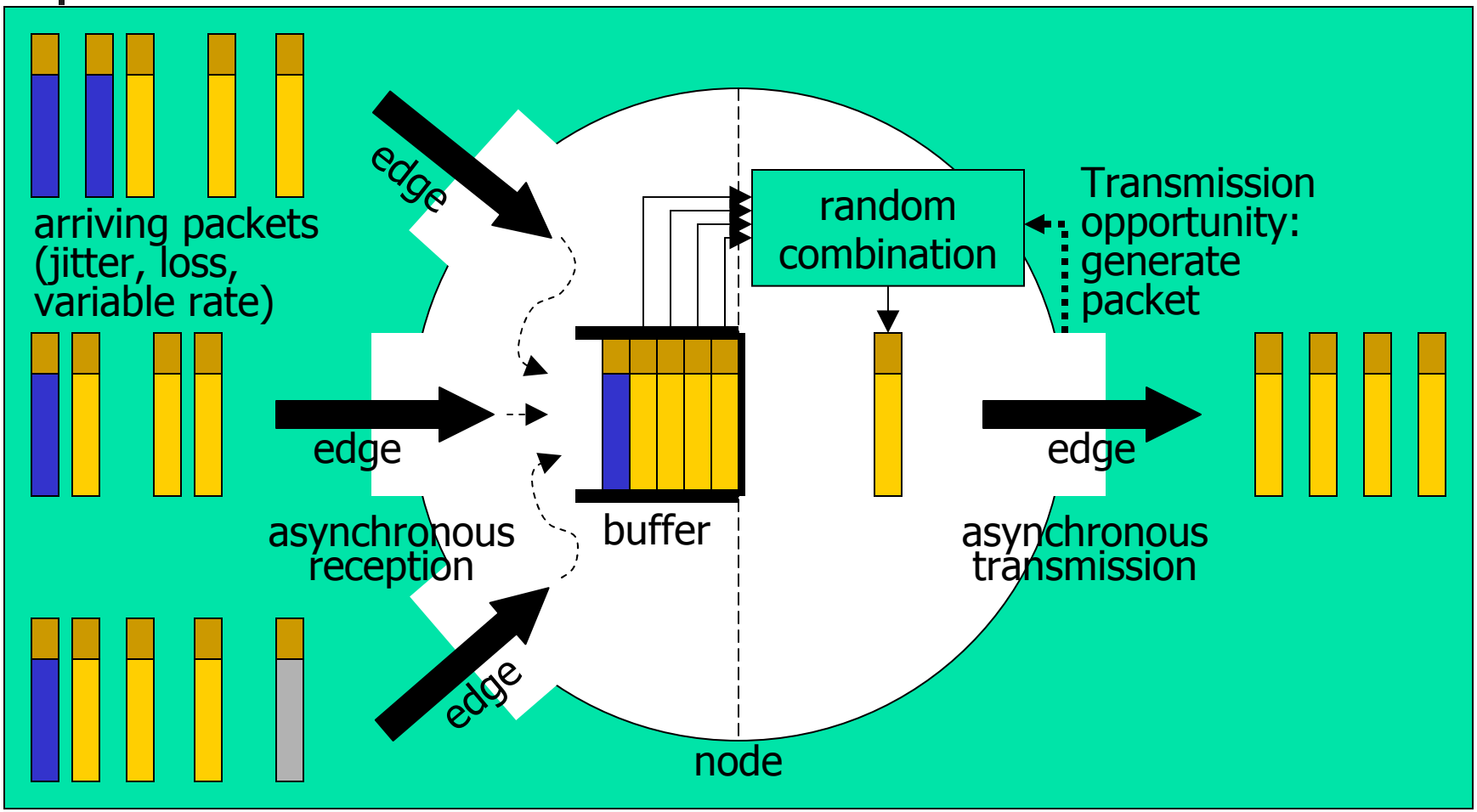
[Albanese, Blömer, Edmonds, Luby, and Sudan; Trans IT 1996]

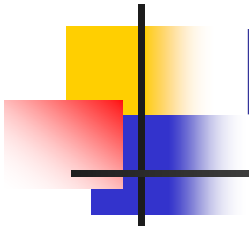


Asynchronous Communication

- In real networks
  - Packets on "unit capacity" edges between each pair of nodes are grouped and carried sequentially
  - Separate edges → separate prop & queuing delays
  - Number of packets per unit time on edge varies
    - Loss, congestion, competing traffic, rounding
- Need to synchronize
  - All packets related to same source vectors $\mathbf{x}_1, \ldots, \mathbf{x}_h$ are in same generation; h is generation size
  - All packets in same generation tagged with same generation number; one byte (mod 256) sufficient

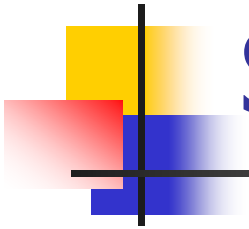
Buffering


arriving packets
(jitter, loss,
variable rate)
edge
random
combination
Transmission
opportunity:
generate
packet
edge
edge
asynchronous
reception
buffer
asynchronous
transmission
edge
node



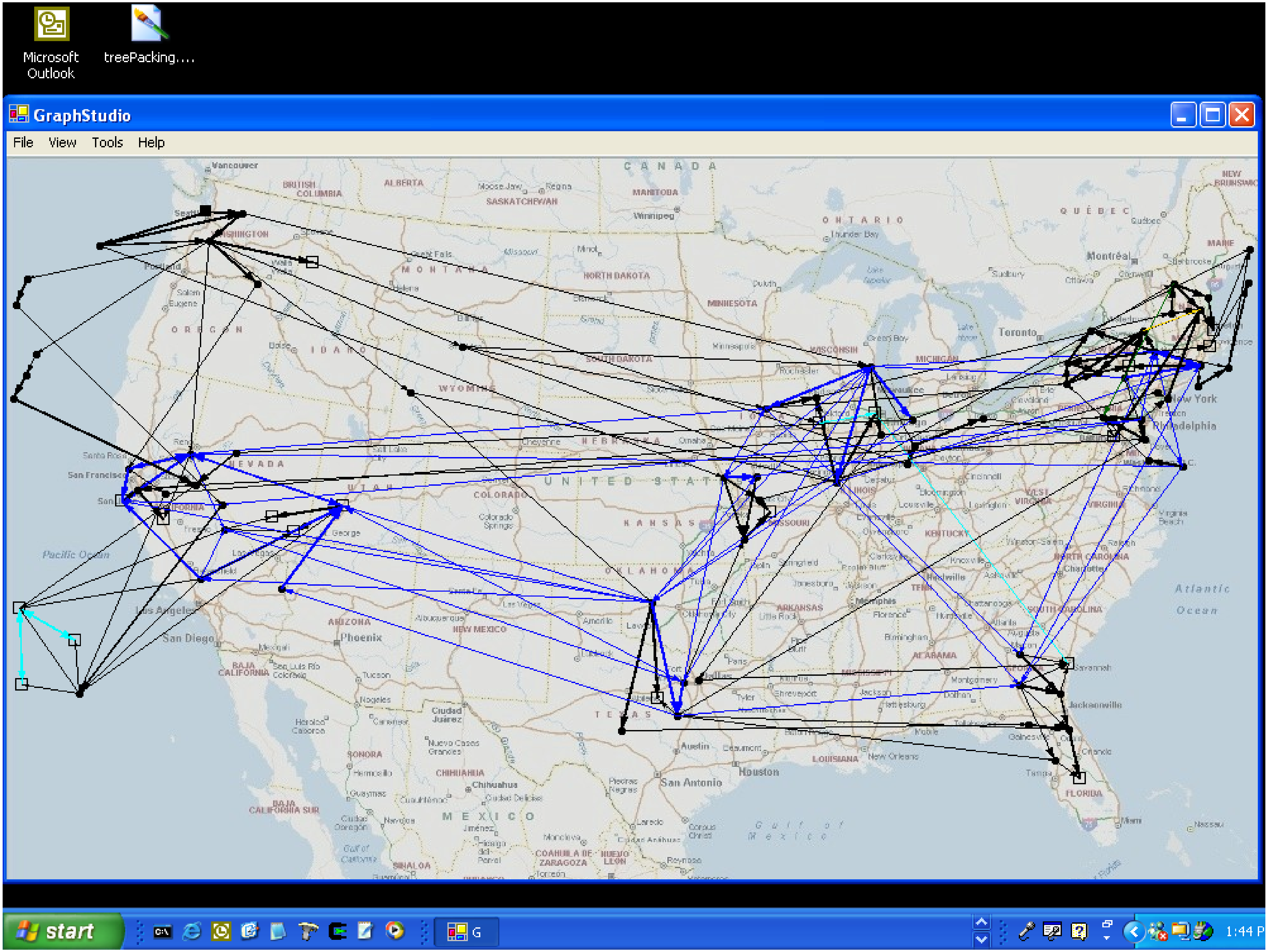
Decoding

- Block decoding:
  - Collect h or more packets, hope to invert G_t
- Earliest decoding (recommended):
  - Perform Gaussian elimination after each packet
    - At every node, detect & discard non-informative packets
  - G_t tends to be lower triangular, so can typically decode $\mathbf{x}_1,\ldots,\mathbf{x}_k$ with fewer more than k packets
  - Much lower decoding delay than block decoding
    - Approximately constant, independent of block length h

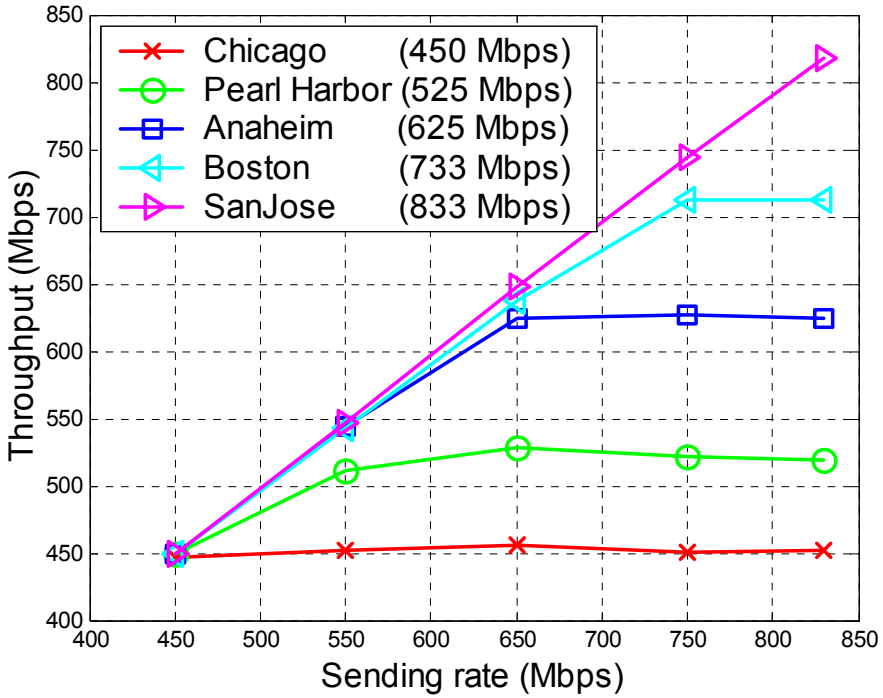


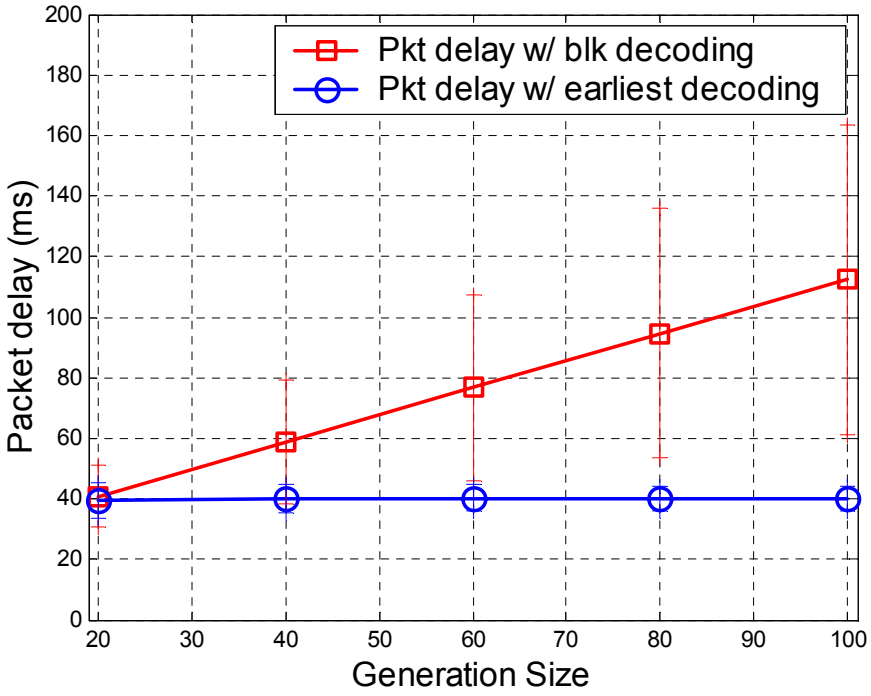
Simulations

- Implemented event-driven simulator in C++
- Six ISP graphs from Rocketfuel project (UW)
  - SprintLink: 89 nodes, 972 bidirectional edges
  - Edge capacities: scaled to 1 Gbps / "cost"
  - Edge latencies: speed of light x distance
- Sender: Seattle; Receivers: 20 arbitrary (5 shown)
  - Broadcast capacity: 450 Mbps; Max 833 Mbps
  - Union of maxflows: 89 nodes, 207 edges
- Send 20000 packets in each experiment, measure:
  - received rank, throughput, throughput loss, decoding delay vs. sendingRate(450), fieldSize(2^{16}), genSize(100), intLen(100)


Microsoft Outlook
treePacking....
GraphStudio
File
View
Tools
Help
C A N A D A
O N T A R I O
Q U É B E C
Montréal
Toronto
New York
Philadelphia
M O N T A N A
NORTH DAKOTA
SOUTH DAKOTA
MINNESOTA
WISCONSIN
MICHIGAN
O R E G O N
I D A H O
WYOMING
N E V A D A
U T A H
COLORADO
K A N S A S
MISSOURI
O K L A H O M A
ARKANSAS
KENTUCKY
NORTH CAROLINA
SOUTH CAROLINA
ALABAMA
FLORIDA
ARIZONA
NEW MEXICO
T E X A S
U N I T E D S T A T E S
Pacific Ocean
Atlantic Ocean
San Francisco
Los Angeles
San Diego
Phoenix
Houston
San Antonio
Austin
Jacksonville
M E X I C O
start
1:44 P

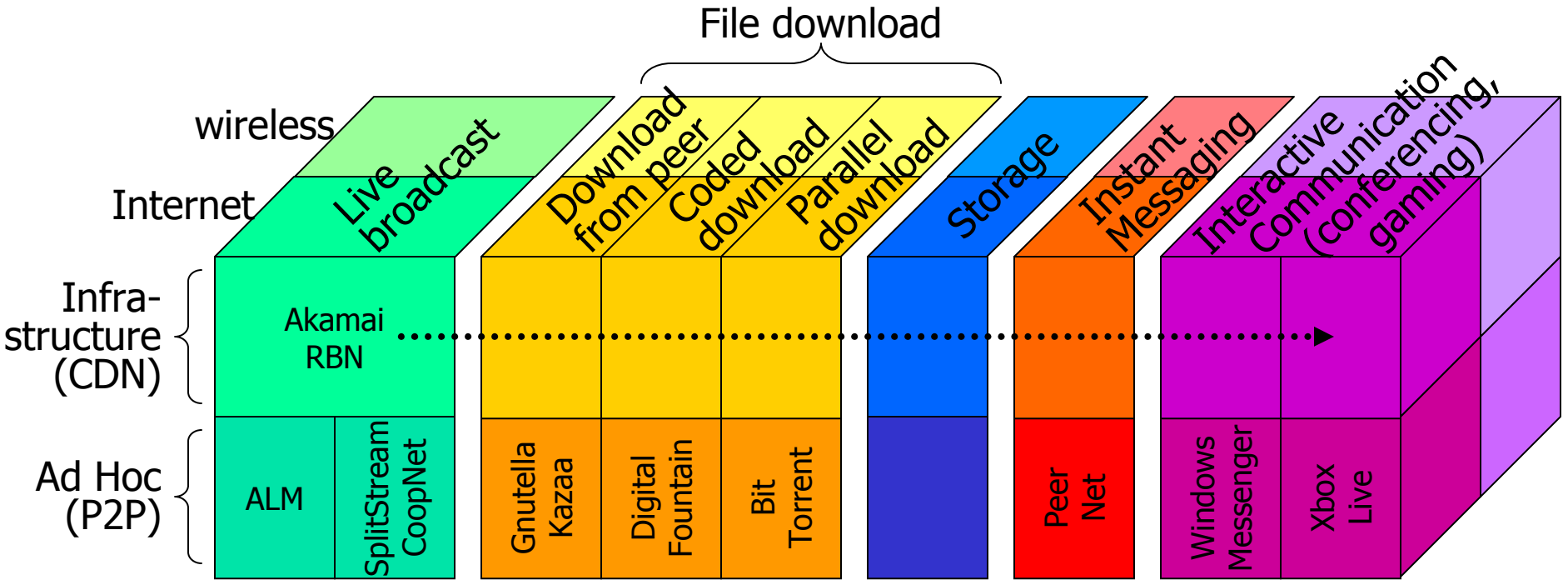
Throughput & Decoding Delay





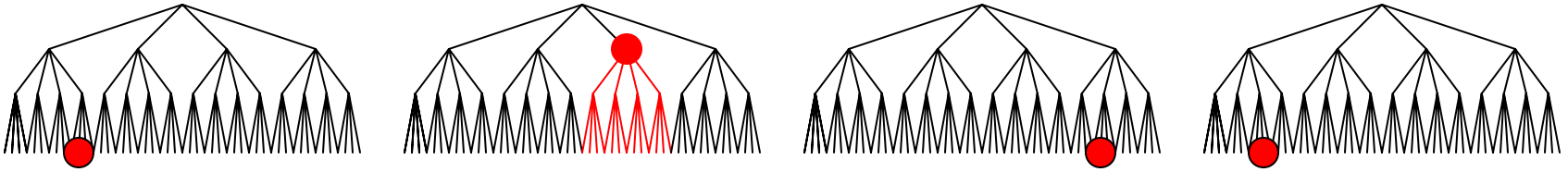
[Chou, Wu, and Jain; Allerton 2003]

Network Coding for Internet and Wireless Applications

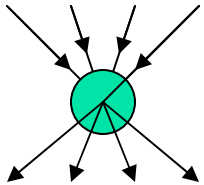


Live Broadcast

- State-of-the-art: Application Layer Multicast (ALM) trees with disjoint edges (e.g., CoopNet, SplitStream)
  - FEC/MDC striped across trees



  - Up/download bandwidths equalized

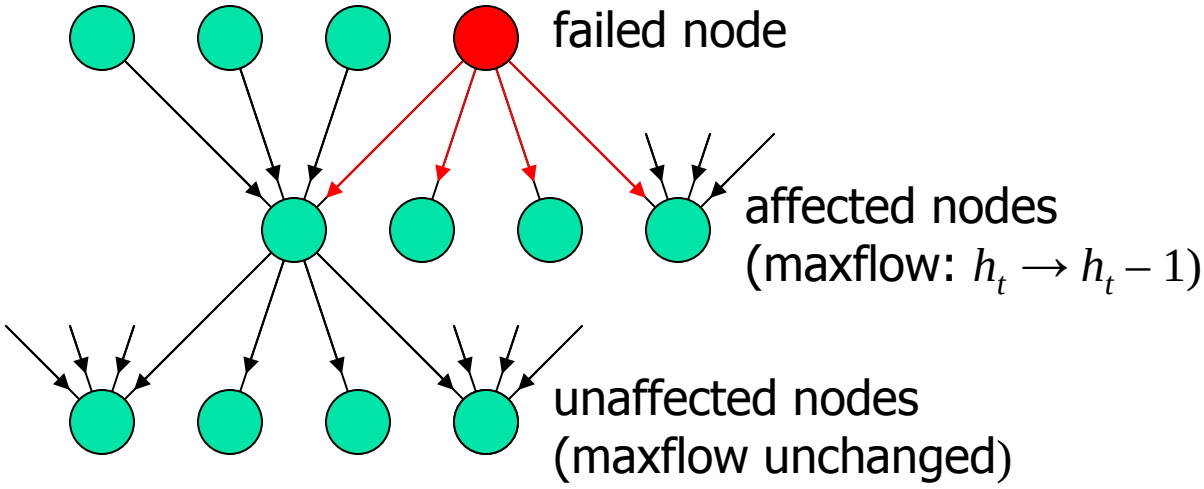


[Padmanabhan, Wang, Chou, and Sripanidkulchai; NOSSDAV 2002; ICNP 2003]

[Castro, Druschel, Kermarrec, Nandi, Rowstron, and Singh; IPTPS 2003]

Live Broadcast (2)

- Network Coding sends mix of parents to each child
  - Losses/failures not propagated beyond child

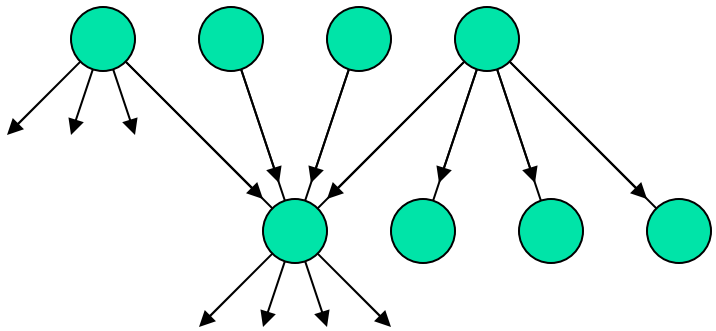


  - ALM/CoopNet average throughput: $(1-\varepsilon)^{\text{depth}}$ * sending rate
    Network Coding average throughput: $(1-\varepsilon)$ * sending rate

[Jain, Lovász, and Chou; J. Distrib. Computing 2005]

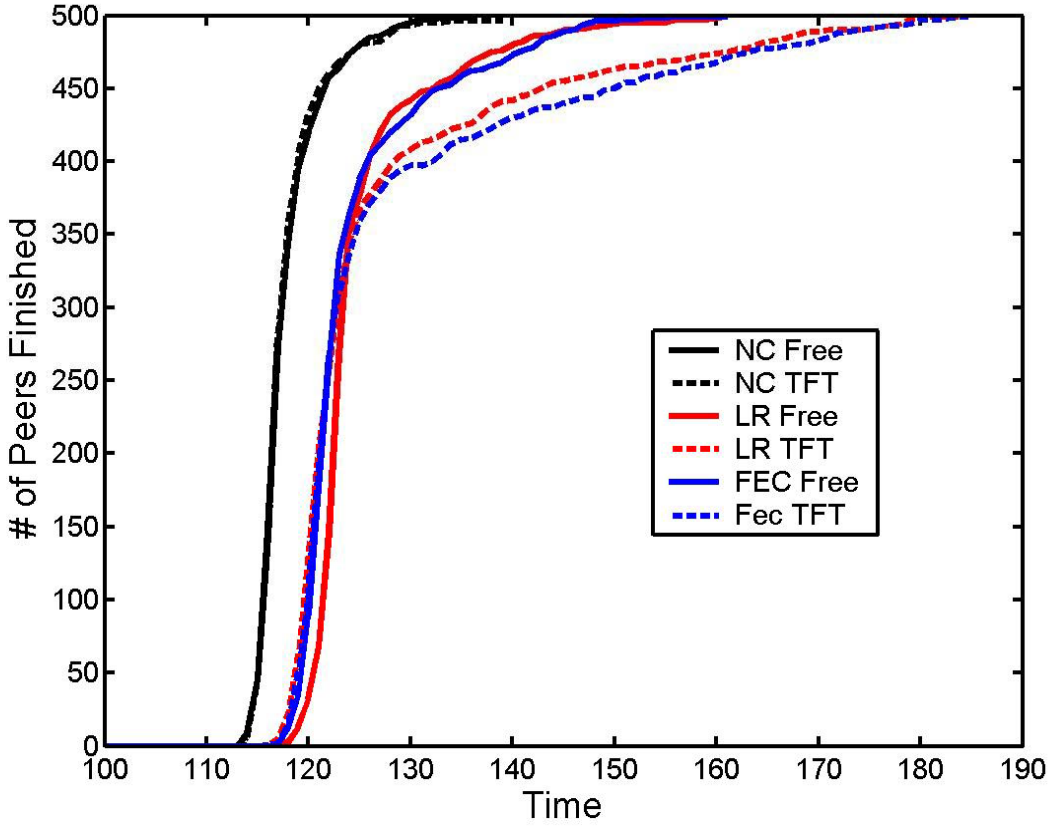
File Download

- State-of-the-Art: Parallel download (e.g., BitTorrent)
  - Selects parents at random
  - Reconciles working sets
  - Flash crowds stressful



- Network Coding:
  - Does not need to reconcile working sets
  - Handles flash crowds similarly to live broadcast
    - Throughput $\longleftrightarrow$ download time
  - Seamlessly transitions from broadcast to download mode

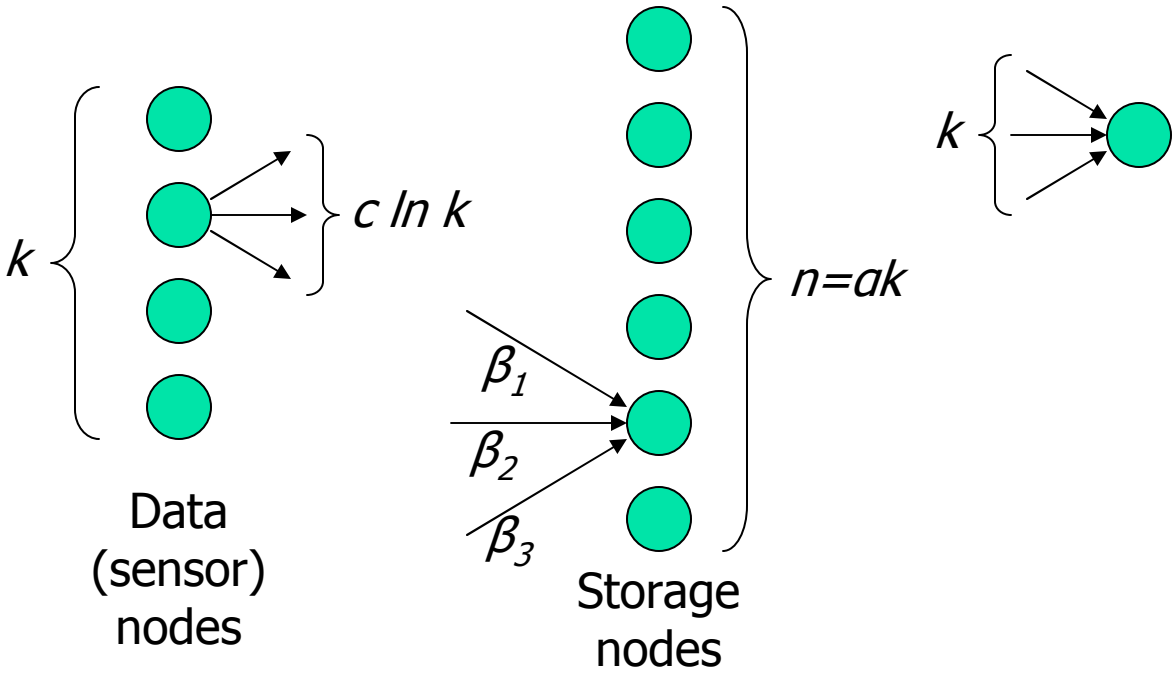
File Download (2)



	Mean	Max
LR Free	124.2	161
LR TFT	126.1	185
FEC Free	123.6	159
FEC TFT	127.1	182
NC Free	117.0	136
NC TFT	117.2	139

Courtesy of [Gkantsidis and Rodruiguez; INFOCOM 2005]

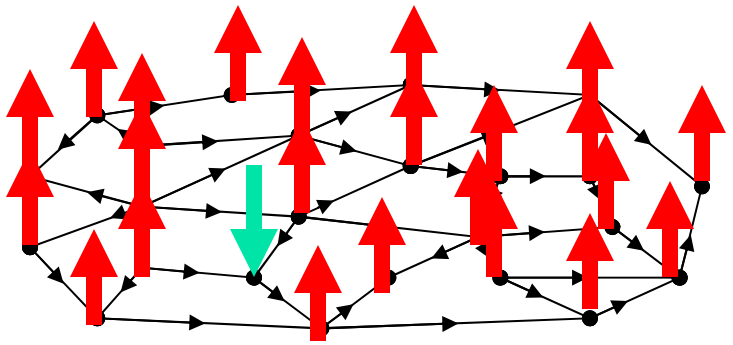
Storage



[Dimakis, Prabhakaran, and Ramchandran; Asilomar 2004; IPSN 2005]
[Acedański, Deb, Médard, and Koetter; NetCod 2005]

Instant Messaging

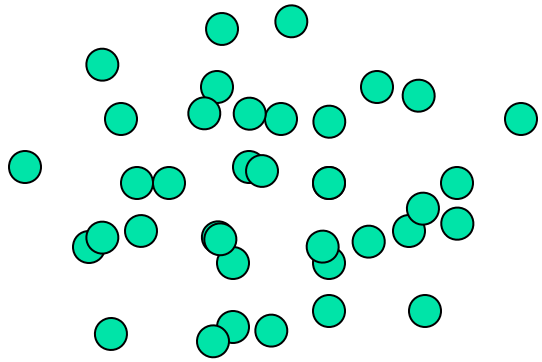
- State-of-the-Art: Flooding (e.g., PeerNet)
  - Peer Name Resolution Protocol (distributed hash table)
  - Maintains group as graph with 3-7 neighbors per node
  - Messaging service: push down at source, pops up at receivers
  - How? Flooding
    - Adaptive, reliable
    - 3-7x over-use



- Network Coding:
  - Improves network usage 3-7x (since all packets informative)
  - Scales naturally from short message to long flows

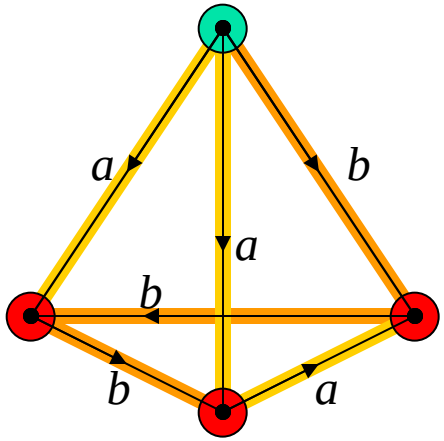
Interactive Communication in mobile ad hoc wireless networks

- State-of-the-Art: Route discovery and maintenance
  - Timeliness, reliability

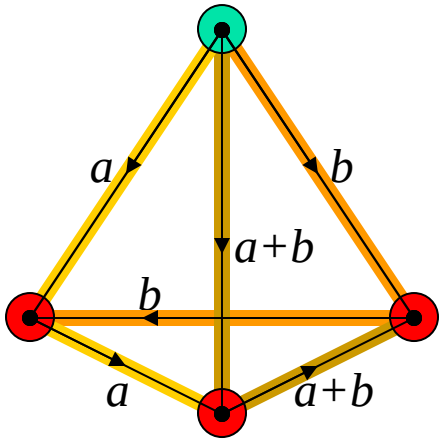


- Network Coding:
  - Is as distributed, robust, and adaptive as flooding
    - Each node becomes collector and beacon of information
  - Minimizes delay without having to find minimum delay route
  - Can also minimize energy (# transmissions)

Network Coding Minimizes Delay



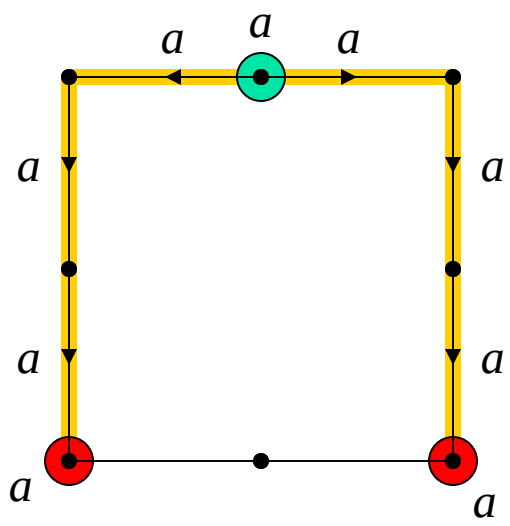
optimal uncoded multicast
delay = 3



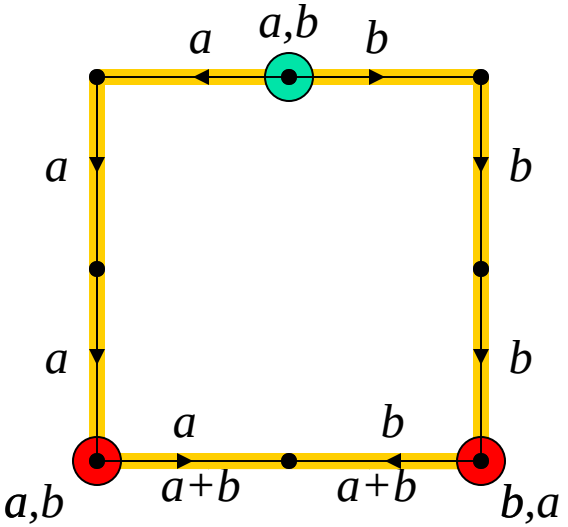
network coded multicast
delay = 2

[Jain and Chou; unpublished 2004]

Network Coding Minimizes Energy (per bit)



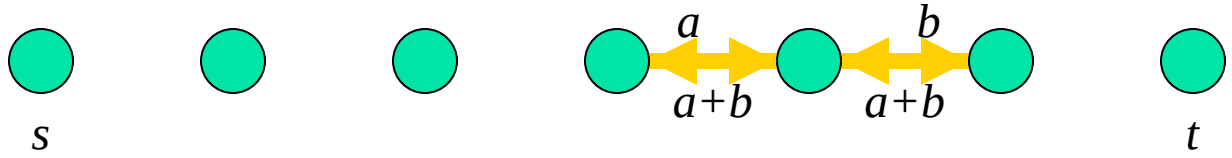
optimal uncoded multicast
transmissions per packet = 5



network coded multicast
transmissions per packet = 4.5

[Wu, Chou, Zhang, Jain, Zhu, and Kung; JSAC 2005]
[Wu, Chou, and Kung; Trans. Comm. 2005]
[Lun, Médard, Ho, and Koetter; ISITA 2004]

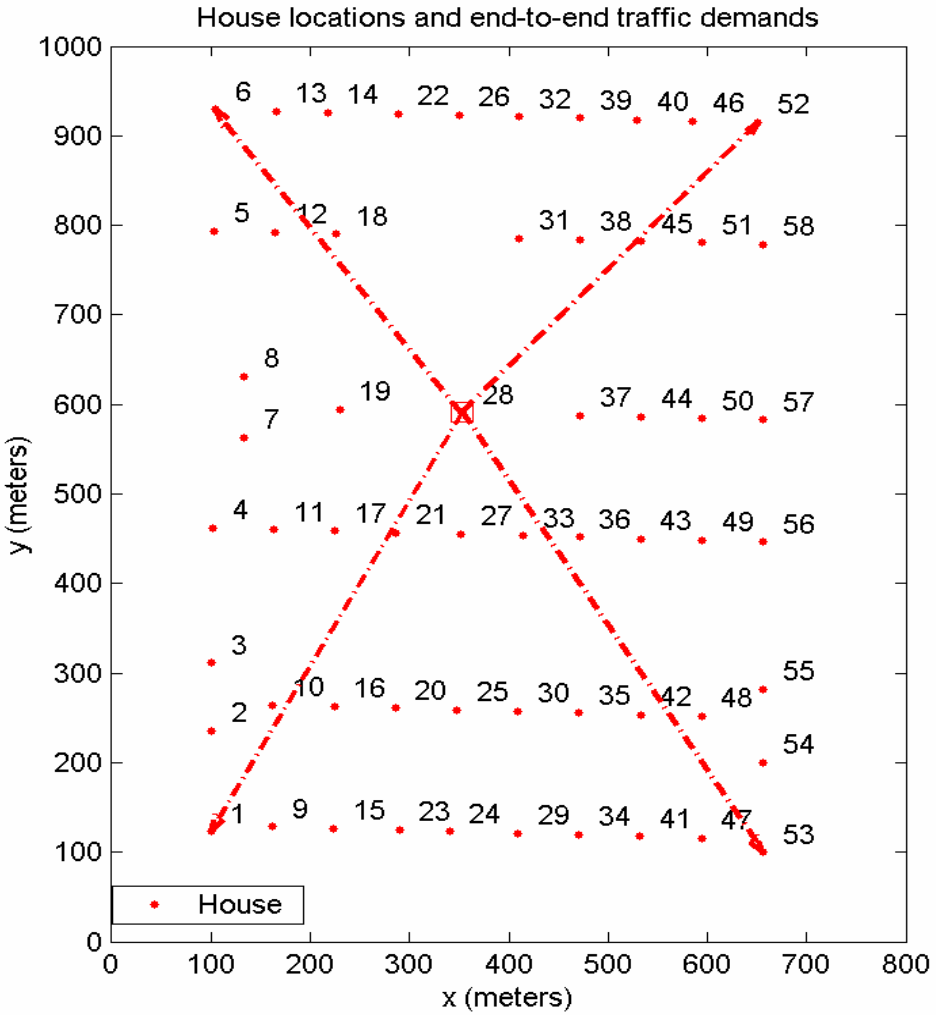
Physical Piggybacking

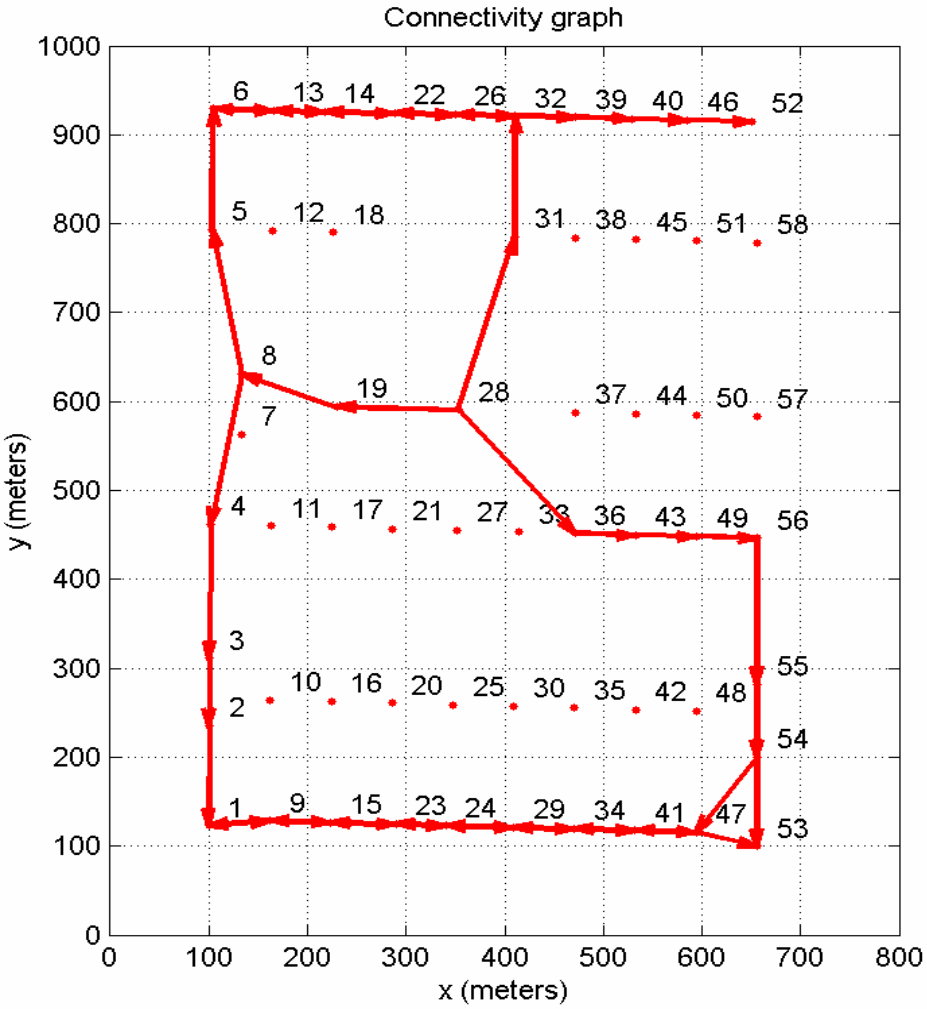


- Information sent from t to s can be piggybacked on information sent from s to t
- Network coding helps even with point-to-point interactive communication
  - throughput
  - delay
  - energy per bit

[Wu, Chou, and Kung; MSR TR 2004]

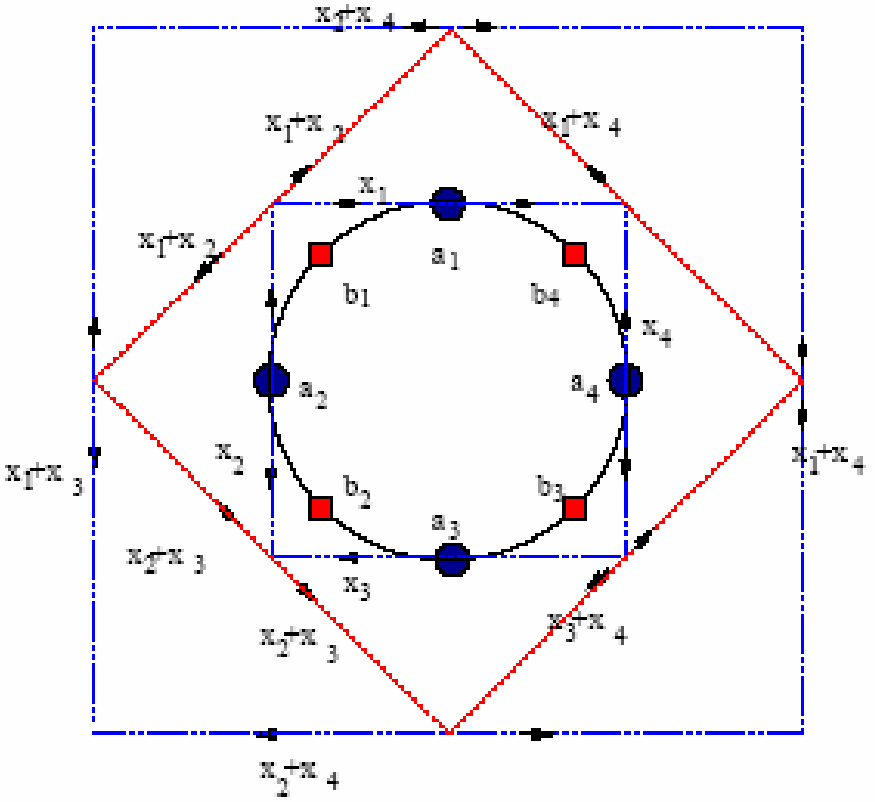
Network Coding in Residential Mesh Network (simulation)





Energy-Efficient Broadcasting in Wireless Ad-hoc Networks

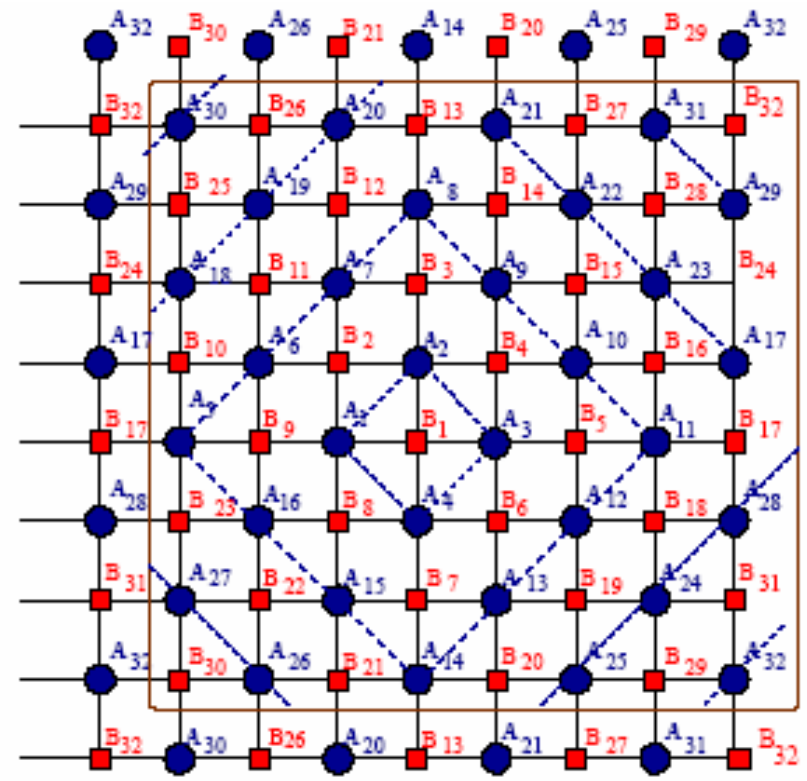
- All nodes are senders; all nodes are receivers
- T_{nc} is #transmissions needed for broadcast with network coding; T_w is #transmissions w/o network coding
- Consider ring network
- Lemma: $T_{nc}/T_w \geq ½$
- Achievable by physical piggybacking



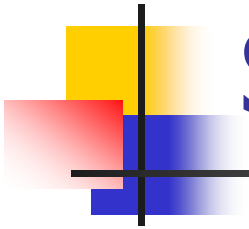
Courtesy of [Widmer, Fragouli, and Le Boudec; NetCod 2005]

Energy-Efficient Broadcasting in Wireless Ad-hoc Networks

- Consider grid network (toroidal)
- Lemma: $T_{nc}/T_w \geq 3/4$
- Achievable by physical piggybacking



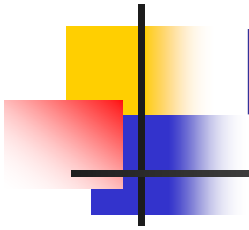
Courtesy of [Widmer, Fragouli, and Le Boudec; NetCod 2005]



Simulation Results

- 1500m x 1500m, 144 nodes randomly placed
- 250m radio range
- Idealized MAC: each time slot, create schedule: pick random node, transmit if all neighbors are idle, repeat until full
- Count #transmissions needed per node to reach certain packet delivery ratio
- Compare Network Coding, Flooding, Ideal Flooding, parametrized by *d*

[Widmer, Fragouli, and Le Boudec; NetCod 2005]

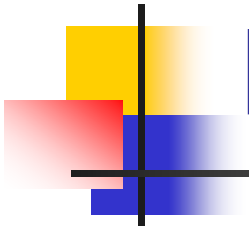


Flooding Algorithm

- Each information unit originating at node is transmitted
- A new packet received is retransmitted with probability *d*
- For "ideal" flooding, packet is not retransmitted if all neighbors have already received it (omniscient)

[Widmer, Fragouli, and Le Boudec; NetCod 2005]

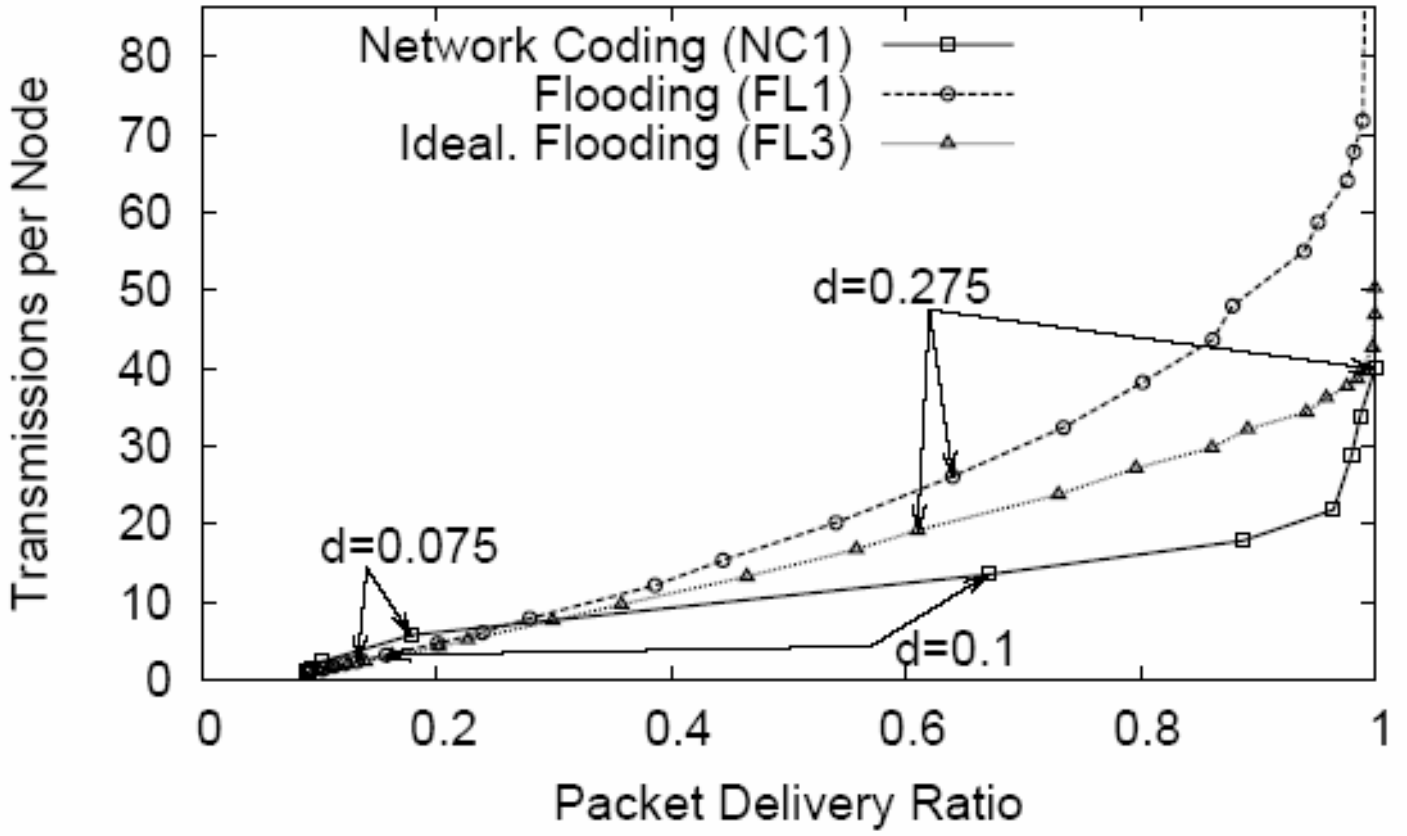
Network Coding Algorithm



- Each node maintains send counter s (#transmissions it is allowed to make)
- Initially, $s = 0$
- Each information unit originating at node increments s by 1
- Each innovative packet received increments s by fraction $d < 1$
- Each transmission decrements s by 1
- Can't transmit anything if $s < 1$

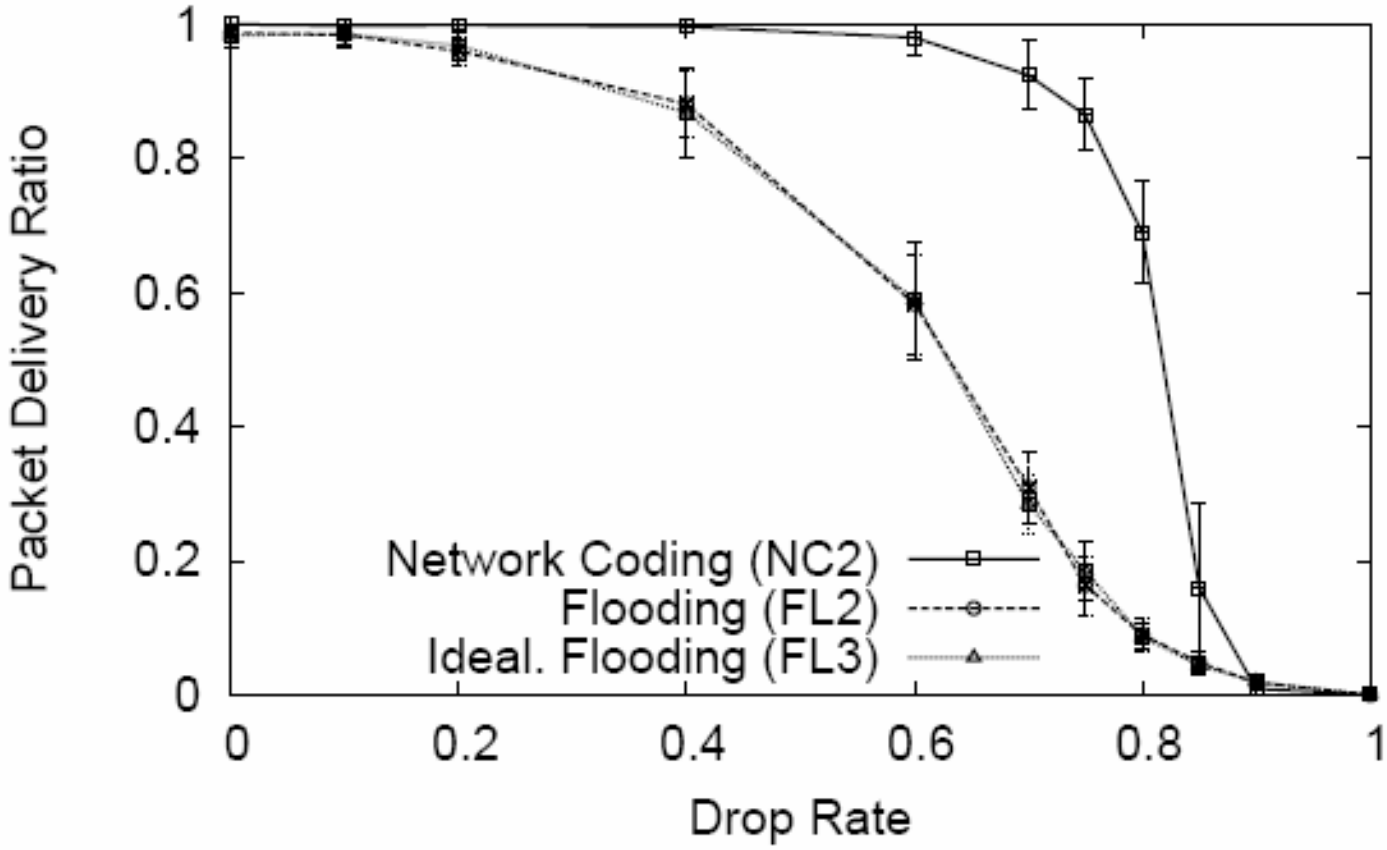
[Widmer, Fragouli, and Le Boudec; NetCod 2005]

Transmissions vs Packet delivery ratio



Courtesy of [Widmer, Fragouli, and Le Boudec; NetCod 2005]

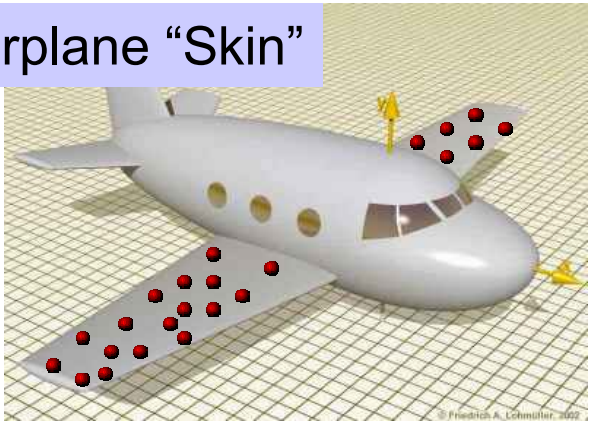
Packet delivery ratio vs Packet drop rate



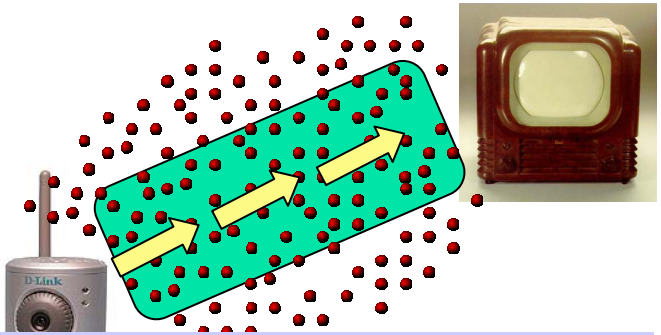
Courtesy of [Widmer, Fragouli, and Le Boudec; NetCod 2005]

Sensor Networks

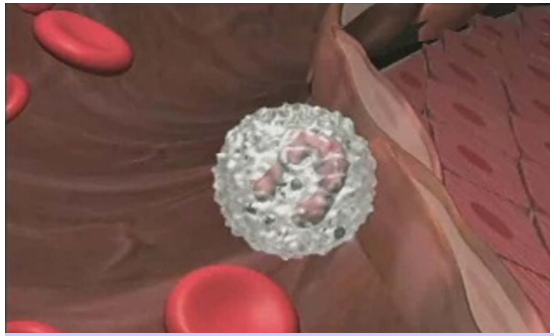
Airplane “Skin”



Smart Surfaces



Communication Backplane

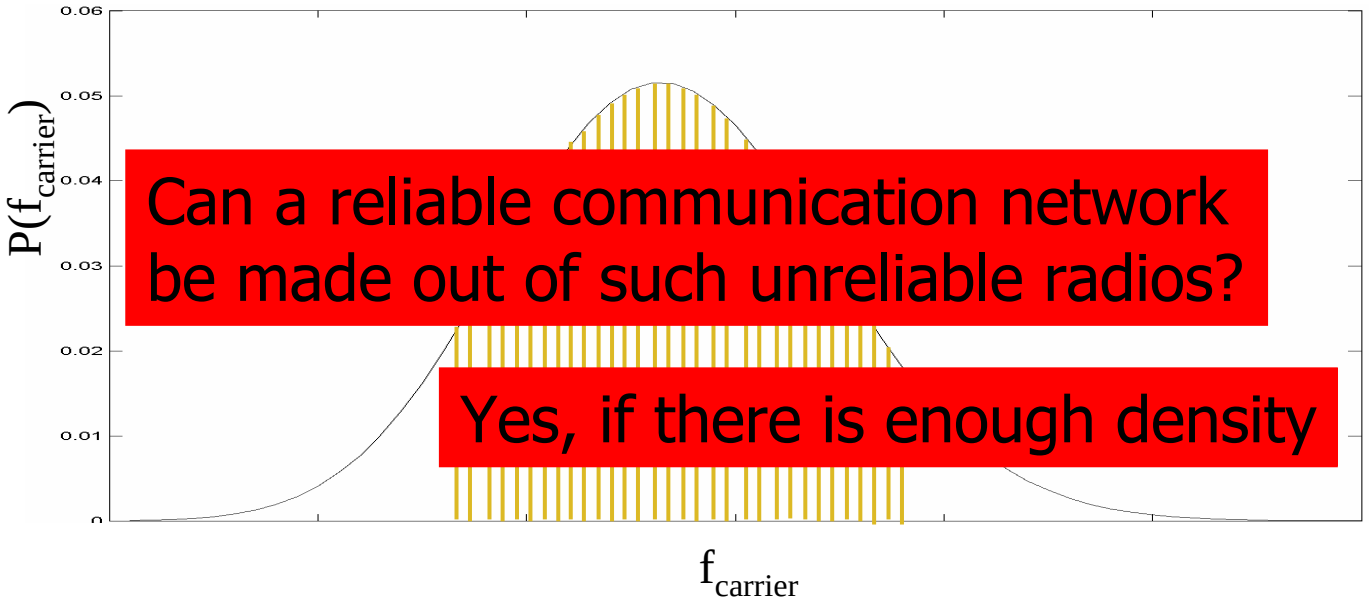


Real-time Health Monitoring

Courtesy of [Petrović, Ramchandran, and Rabaey; NetCod 2005]

Untuned Radios

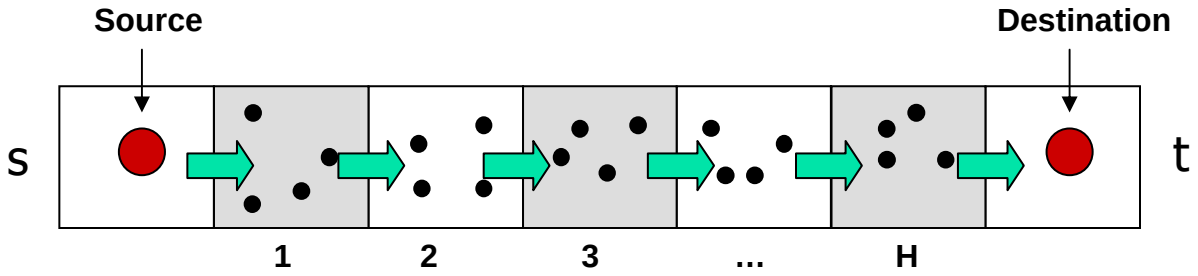
- Sensor networks use narrowband radios to maximize SNR
- Replacing crystal oscillator+PLL by on-chip LC reduces size, energy
- Manufacturing variations randomize carrier frequency



Courtesy of [Petrović, Ramchandran, and Rabaey; NetCod 2005]

Communication Method

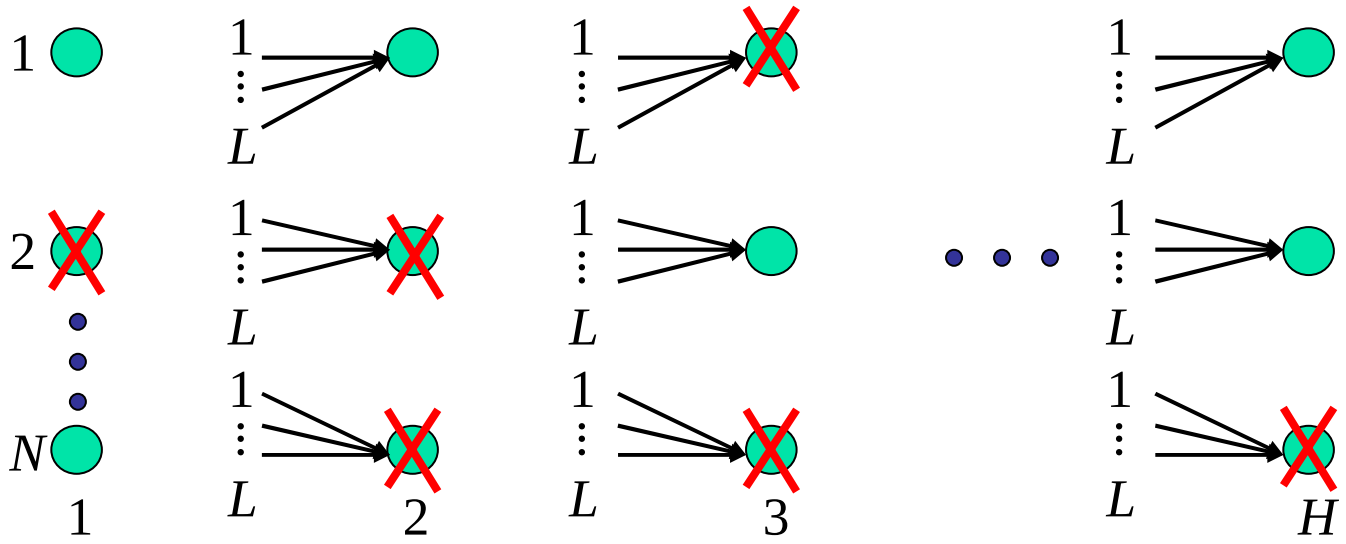
- Each Tx and Rx radio randomly chooses one of N channels
- Adjust transmission range to include N nodes
- Communicate through H stages of N nodes each



- Approx 1/e (37%) Txs communicate collision-free to some Rx radio
- Use L Rx radios per node to listen reliably to L/e nodes on average

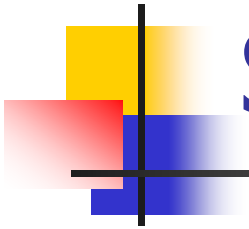
Courtesy of [Petrović, Ramchandran, and Rabaey; NetCod 2005]

Random Graph Representation



- Result 1: For all β<1/e, there exists L s.t. Maxflow(s,t) > βN with high probability as N → ∞, as long as log[H(N)]/N → 0
- Result 2: Routing gives constant throughput (does not grow with N) for H linear in N.

Courtesy of [Petrović, Ramchandran, and Rabaey; NetCod 2005]



Summary

- Network Coding can be practical
  - Packetization
  - Buffering
- Network Coding is being applied to
  - Internet
    - Live broadcast, file download, storage, messaging, …
  - Wireless ad hoc, mobile, and sensor networks