Network Coding for the Internet and Wireless Networks

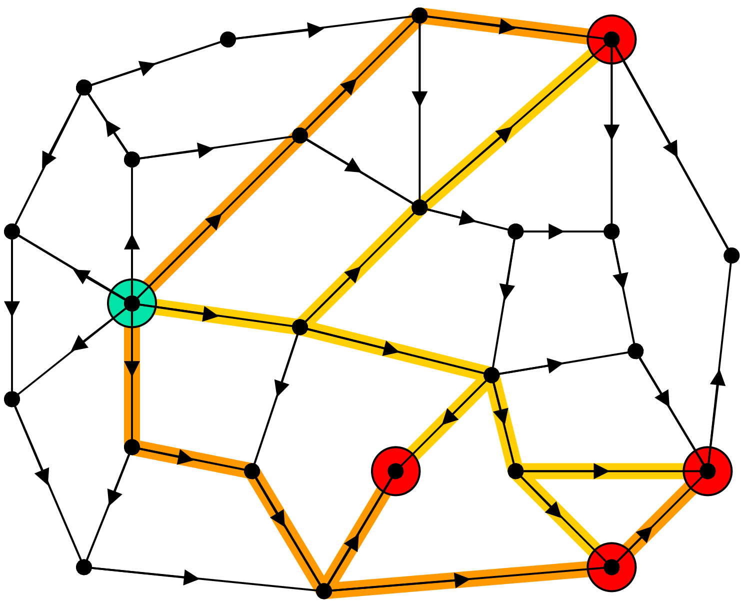
Philip A. Chou
with thanks to Yunnan Wu, Kamal Jain, and Pablo Rodruiguez
Microsoft Research

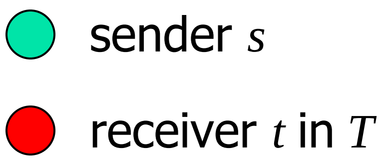
Banff International Research Station
July 23-28, 2005

Outline

- Introduction to Network Coding
- Practical Network Coding
  - Packet format
  - Buffering
- Internet and Wireless Applications
  - Live Broadcasting, File Downloading, Messaging, Interactive Communication

Network Coding Introduction

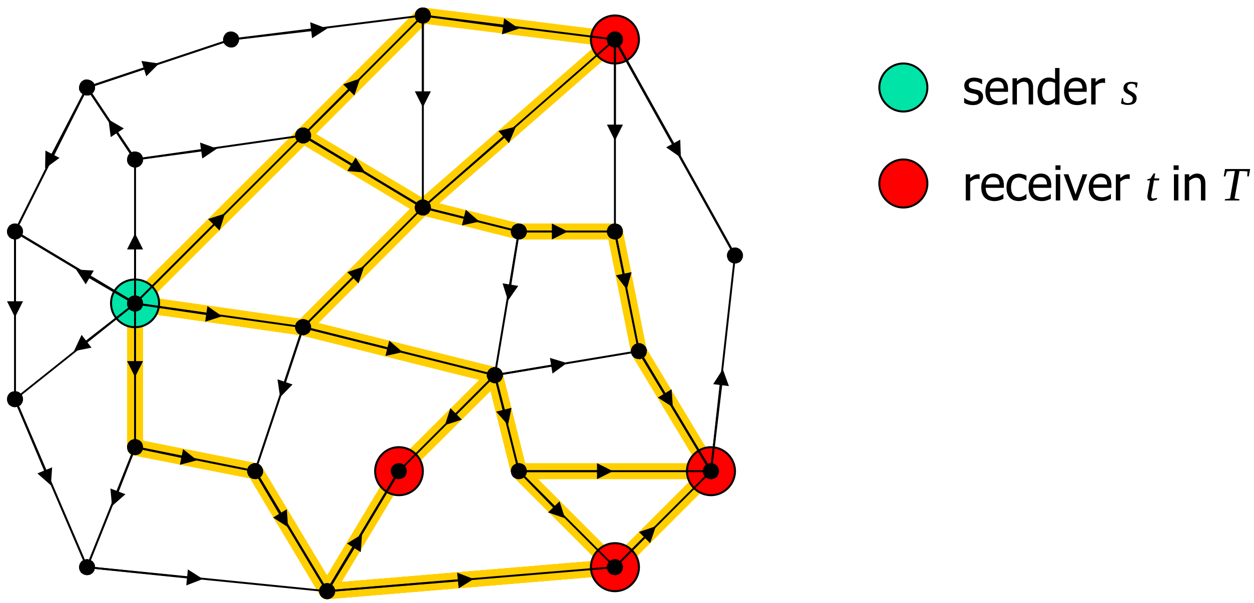




- Directed graph with edge capacities
- Sender s, set of receivers T
- Ask: Maximum rate to multicast info from s to T ?

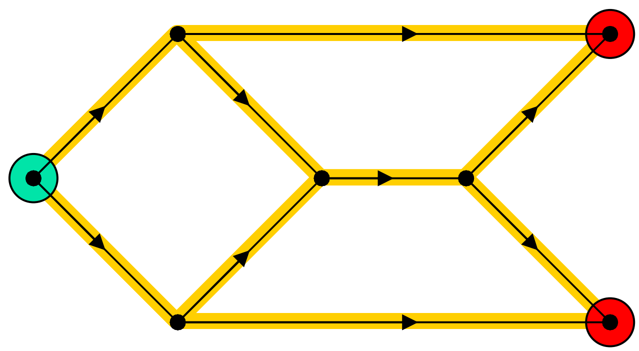
  (the "multicast capacity" from s to T)

Maximum Flow

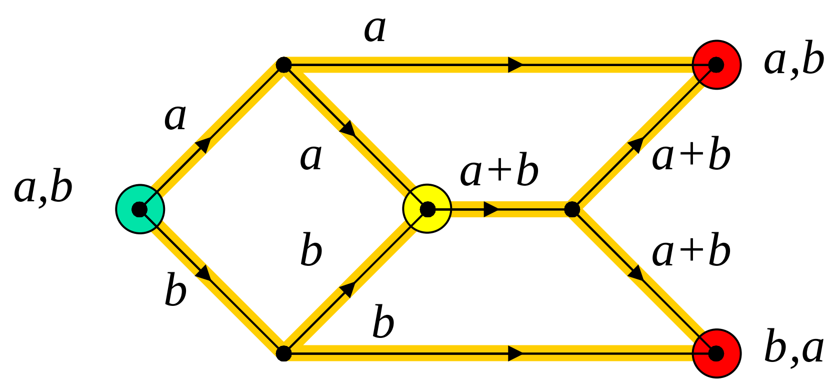


- Menger (1927) – single receiver
  - Maxflow(s,t) ≤ Mincut(s,t) ≡ h_t achievable
- Edmonds (1972) – all nodes are receivers
  - Maxflow(s,T) ≤ $\min_t h_t$ ≡ h achievable

Network Coding Maximizes Throughput



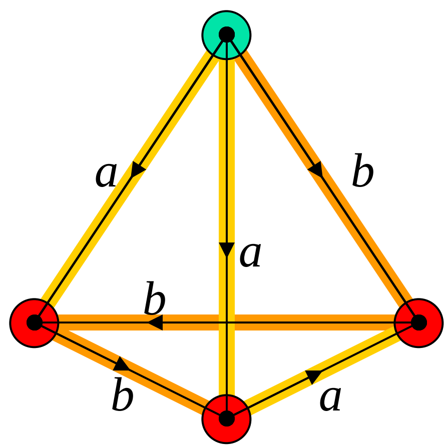
optimal uncoded multicast
throughput = 1.5



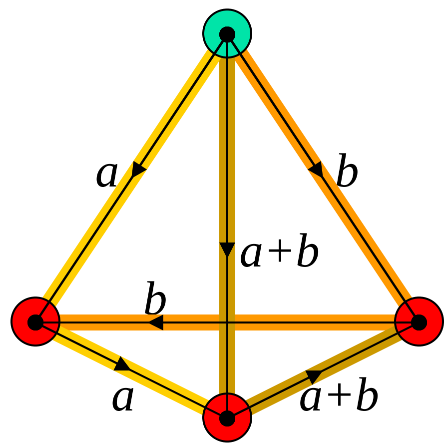
network coding
throughput = 2

- Alswede, Cai, Li, Yeung (2000)
  - NC always achieves $h = \min_t h_t$
- Li, Yeung, Cai (2003)
- Koetter and Médard (2003)
- Jaggi, Sanders, et al. (2005)

Network Coding Minimizes Delay



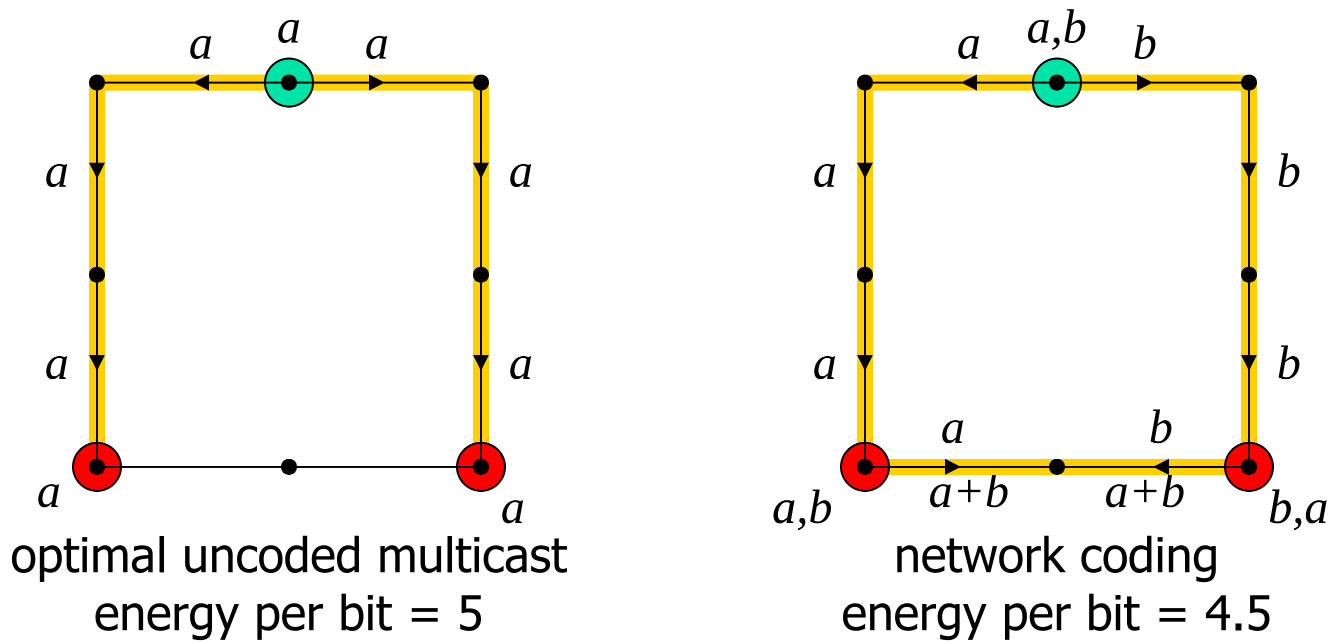
optimal uncoded multicast
delay = 3



network coding
delay = 2

- Jain and Chou (2004)

Network Coding Minimizes Energy (per bit)



- Wu et al. (2003); Wu, Chou, Kung (2004)
- Lun, Médard, Ho, Koetter (2004)

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
  - Sensor networks
  - Stationary wireless (residential) mesh 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

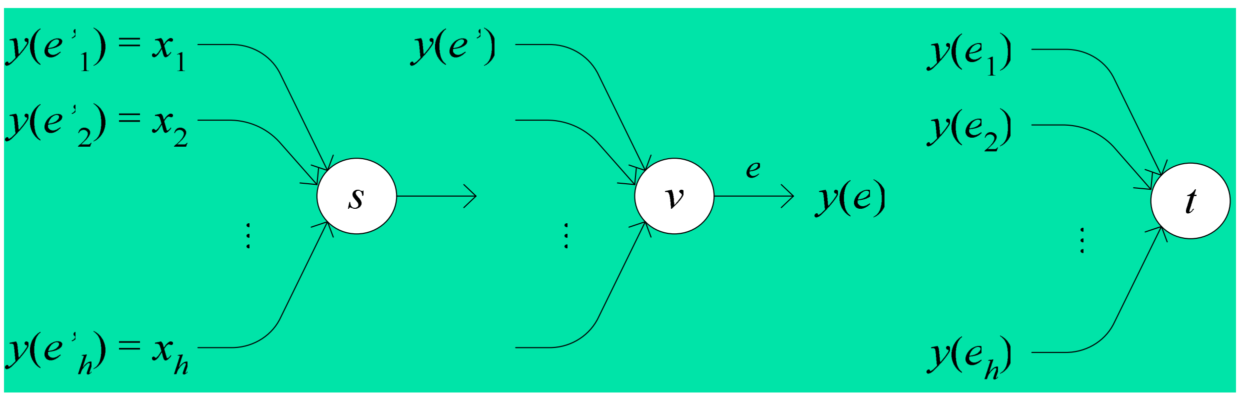
Approach

- Packet Format
  - Removes need for centralized knowledge of graph topology and encoding/decoding functions
- Buffer Model
  - Allows asynchronous packets arrivals & departures with arbitrarily varying rates, delay, loss

[Chou, Wu, and Jain, Allerton 2003]
[Ho, Koetter, Médard, Karger, and Effros, ISIT 2003]

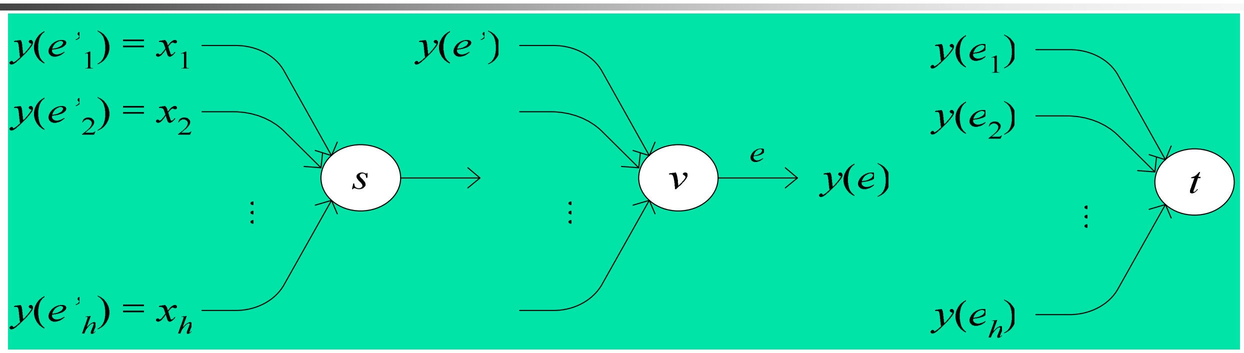
Algebraic Formulation

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



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

Global Encoding Vectors



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

$$\begin{bmatrix} y(e_1) \\ \vdots \\ y(e_h) \end{bmatrix} = \begin{bmatrix} g_1(e_1) & \cdots & g_h(e_1) \\ \vdots & \ddots & \vdots \\ g_1(e_h) & \cdots & g_h(e_h) \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix} = G_t \begin{bmatrix} x_1 \\ \vdots \\ x_h \end{bmatrix}$$

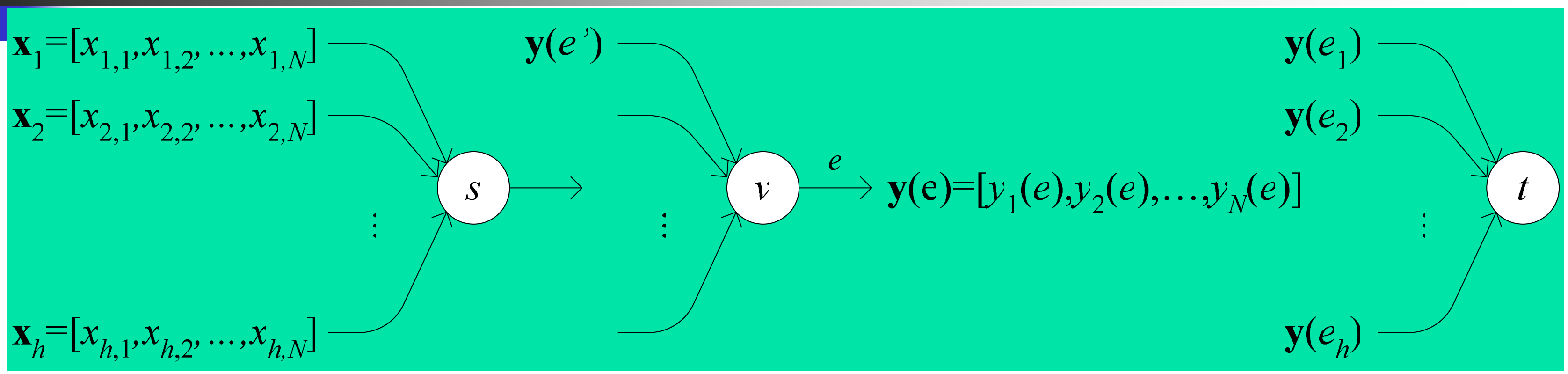
Invertibility of G_t

- 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 et al., 2003]

[Jaggi, Sanders, et al., 2003]

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 Format

- 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 $\mathbf{x}_i$, $i=1,\ldots,h$

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

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

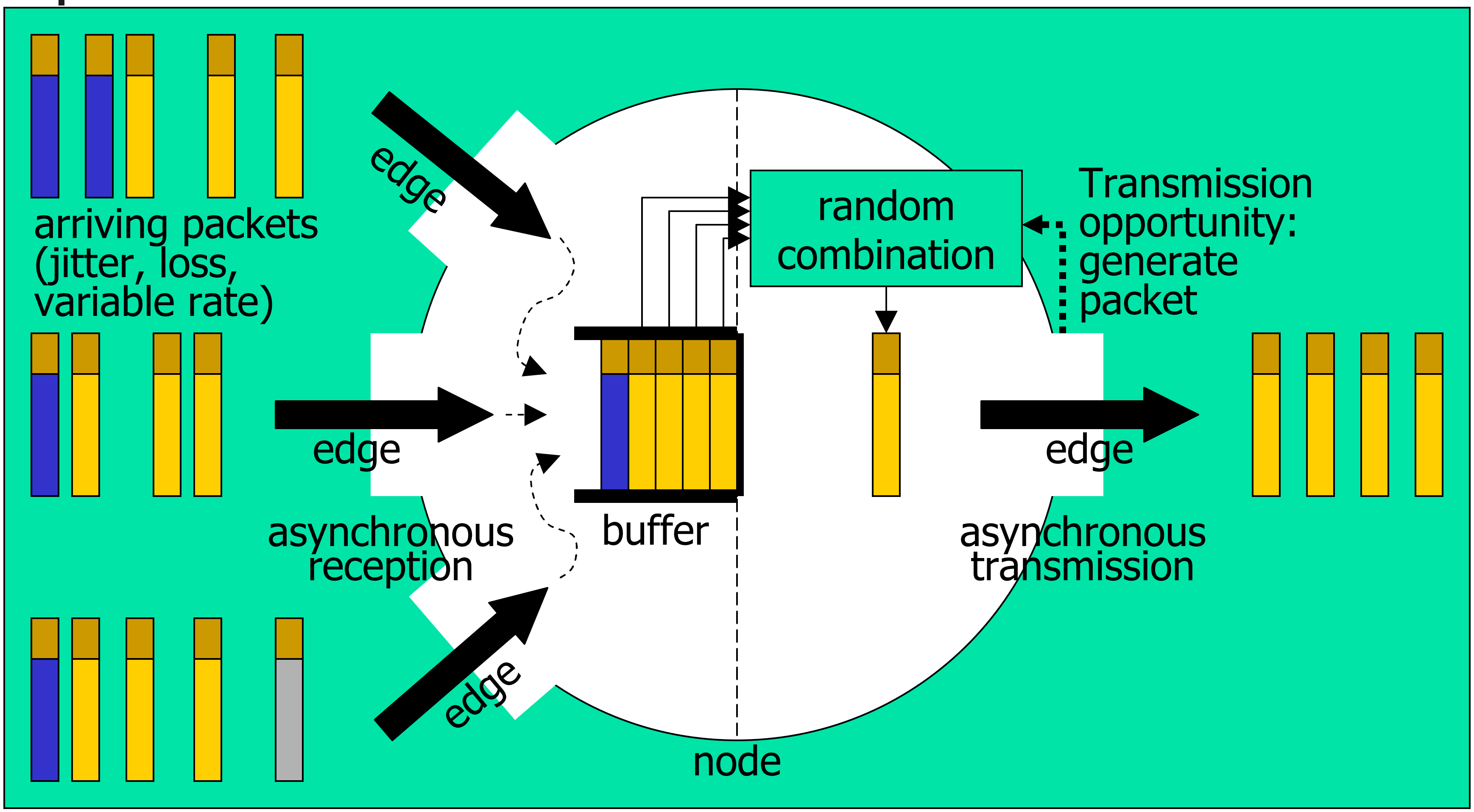
Cost vs. Benefit

- 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

Asynchronous Communication

- In real networks
  - Packets on "unit capacity" edges between each pair of nodes are grouped and carried sequentially
  - Separate edges $\rightarrow$ 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



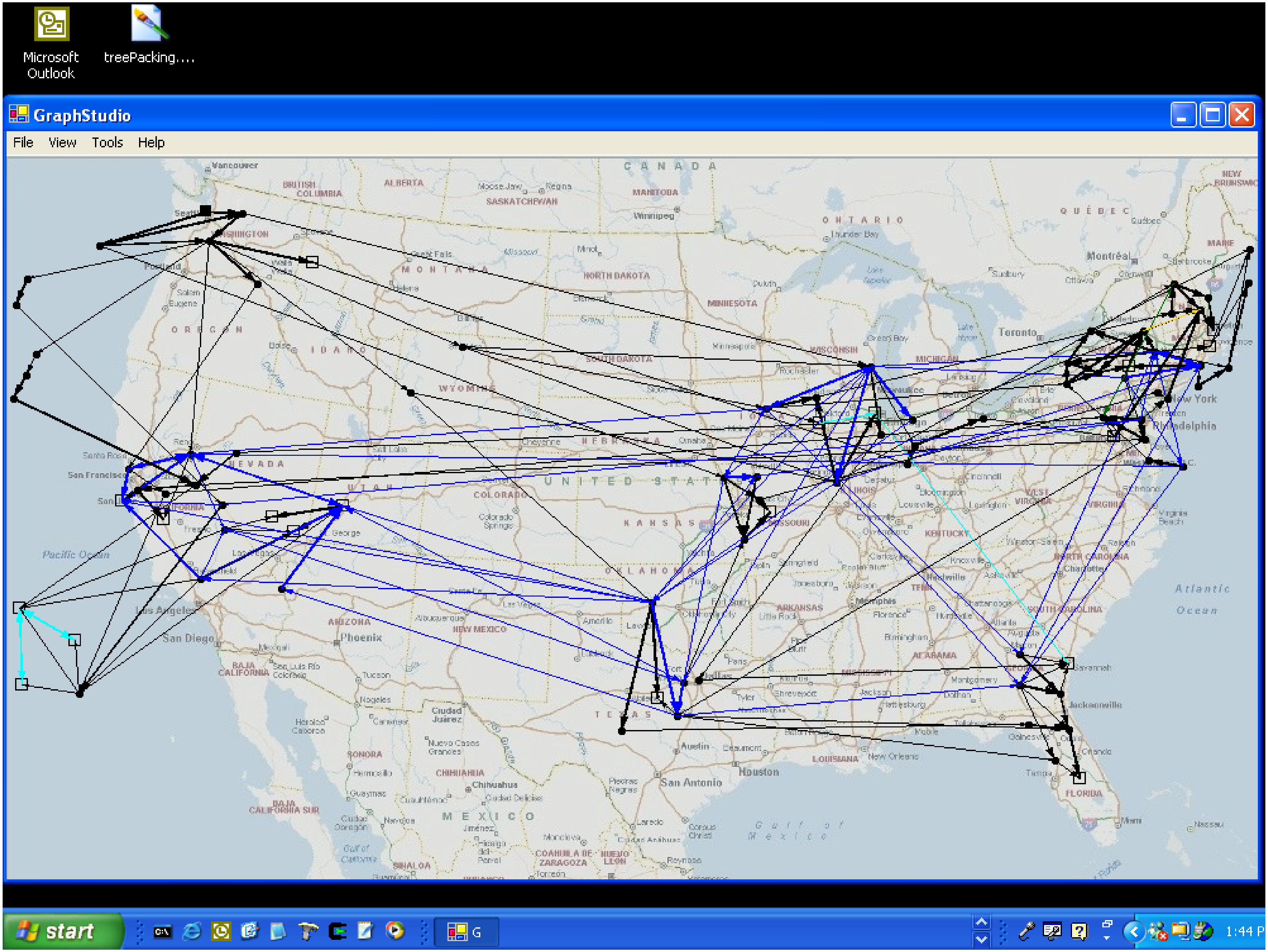
arriving packets
(jitter, loss,
variable rate)
edge
edge
edge
random
combination
Transmission
opportunity:
generate
packet
edge
asynchronous
reception
buffer
asynchronous
transmission
node

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

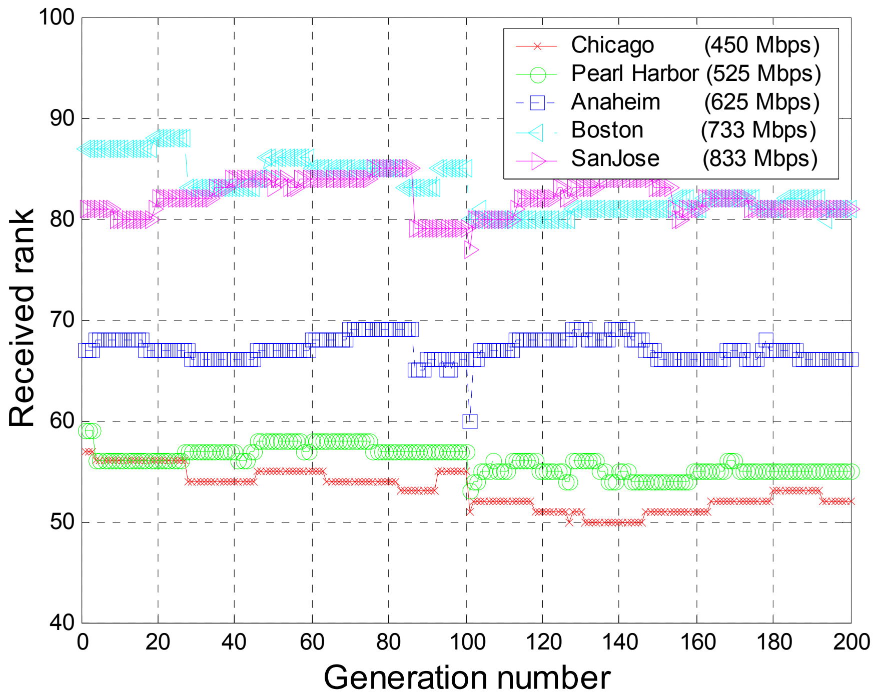
Simulations

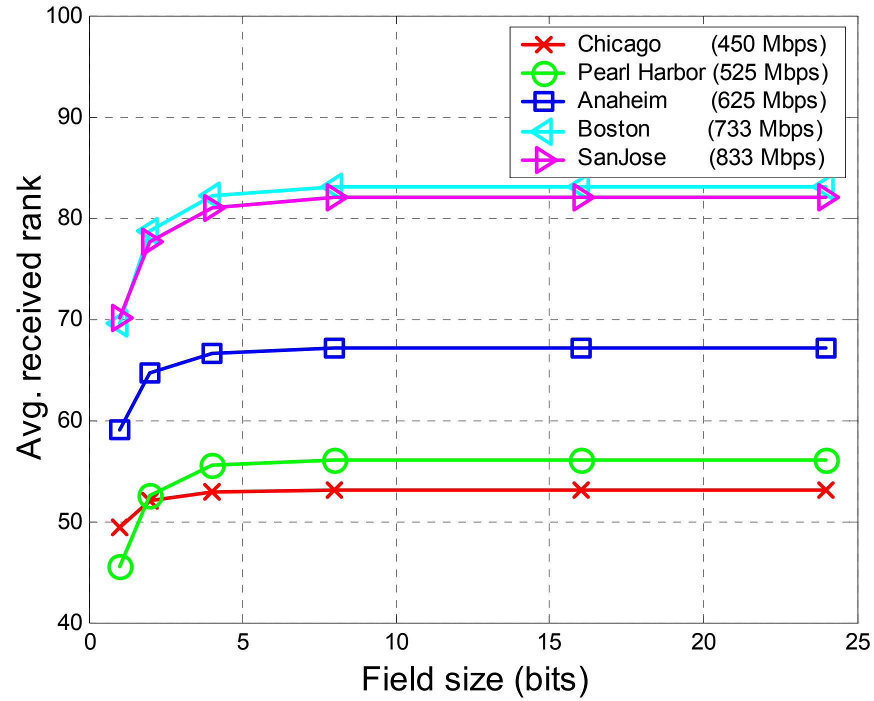
- Implemented event-driven simulator in C++
- Six ISP graphs from Rocketfuel project (UW)
  - SprintLink: 89 nodes, 972 bidirectional edges
  - Edge capacities: scaled to 1 Gbps / "cost"
  - Edge latencies: speed of light x distance
- Sender: Seattle; Receivers: 20 arbitrary (5 shown)
  - Mincut: 450 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)



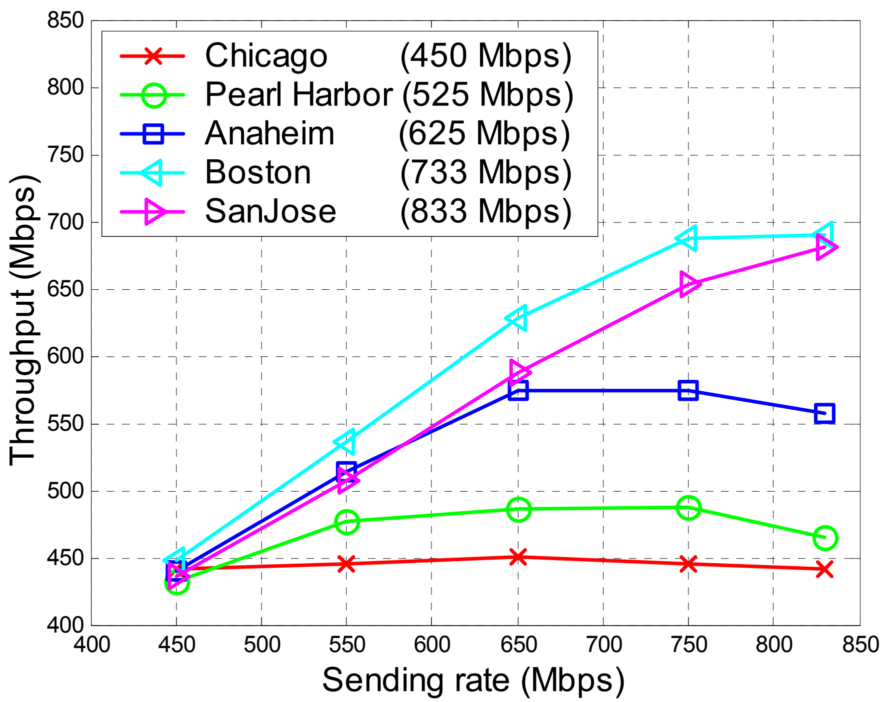
Microsoft Outlook
treePacking....
GraphStudio
File
View
Tools
Help
CANADA
BRITISH COLUMBIA
ALBERTA
MANITOBA
ONTARIO
QUÉBEC
Montréal
Toronto
New York
Philadelphia
WASHINGTON
OREGON
IDAHO
MONTANA
WYOMING
NORTH DAKOTA
SOUTH DAKOTA
MINNESOTA
WISCONSIN
MICHIGAN
NEVADA
UTAH
COLORADO
KANSAS
MISSOURI
KENTUCKY
OKLAHOMA
ARKANSAS
TEXAS
ARIZONA
NEW MEXICO
ALABAMA
FLORIDA
NORTH CAROLINA
SOUTH CAROLINA
UNITED STATES
San Francisco
Los Angeles
San Diego
Phoenix
Dallas
Austin
San Antonio
Houston
Jacksonville
Pacific Ocean
Atlantic Ocean
MEXICO
start
1:44 P

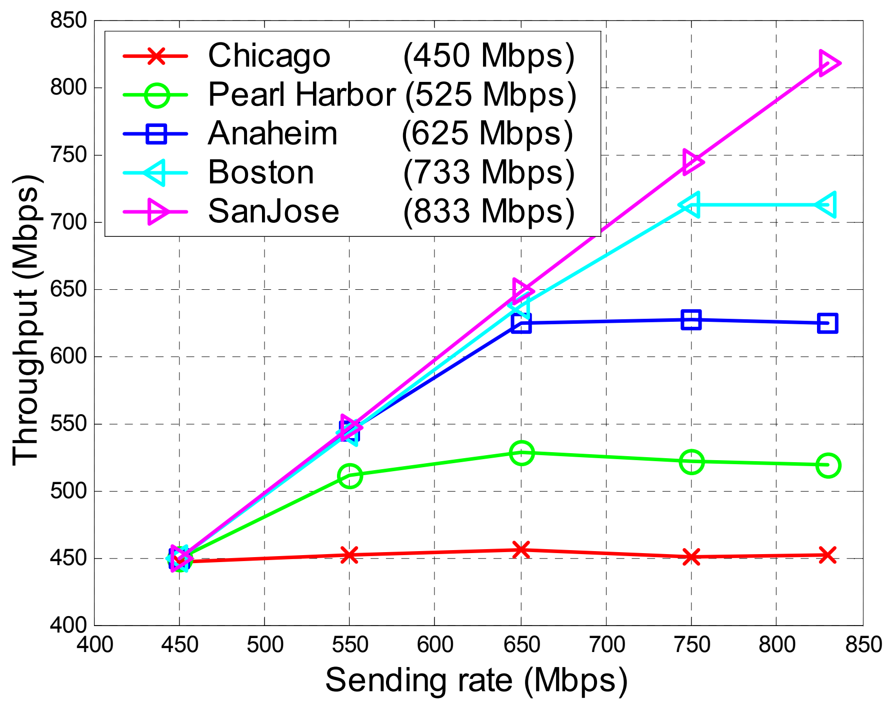
Received Rank



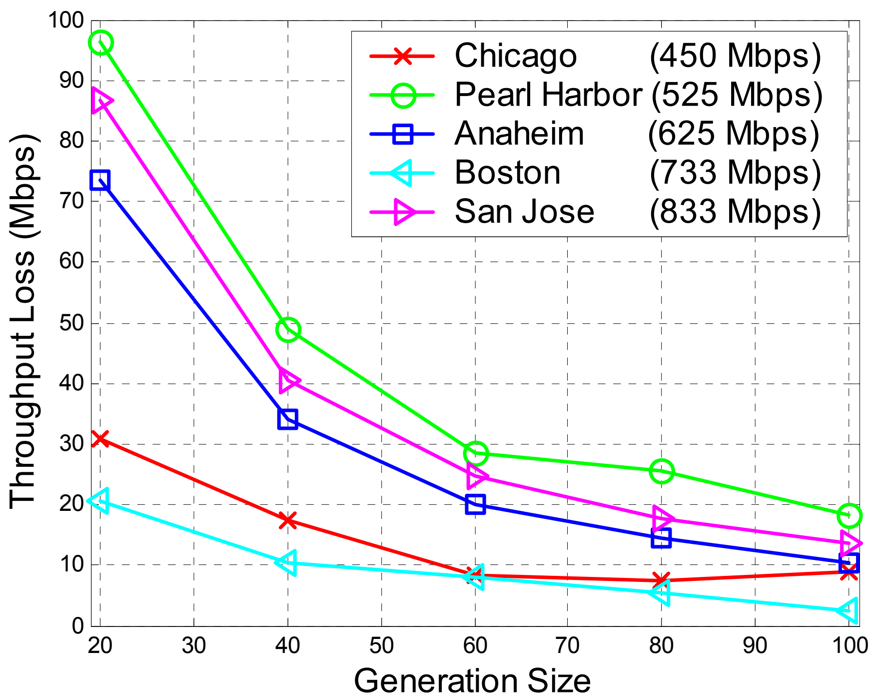


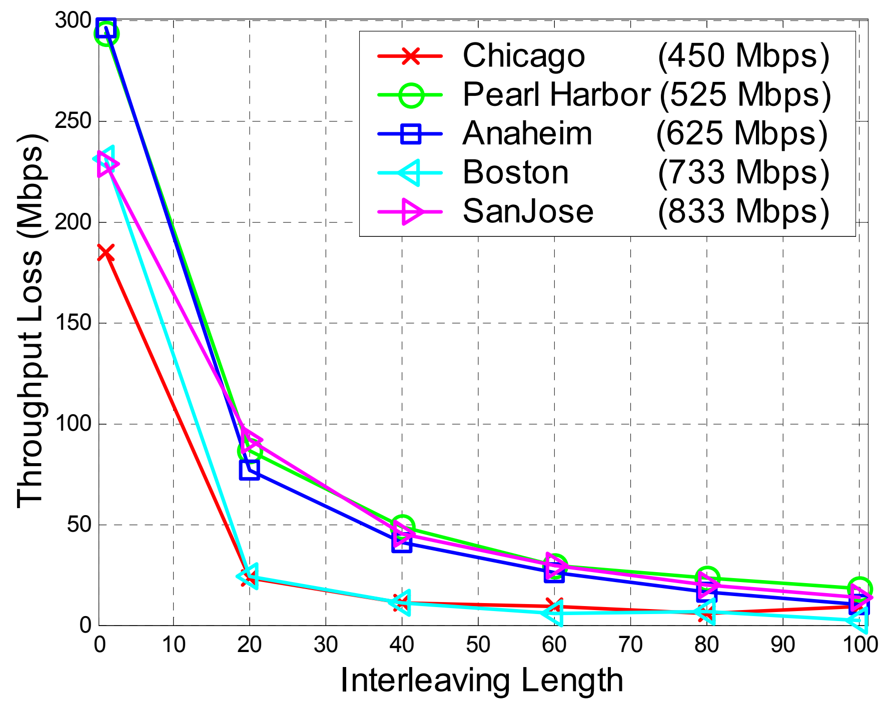
Throughput


Chicago (450 Mbps)
Pearl Harbor (525 Mbps)
Anaheim (625 Mbps)
Boston (733 Mbps)
SanJose (833 Mbps)
Throughput (Mbps)
Sending rate (Mbps)

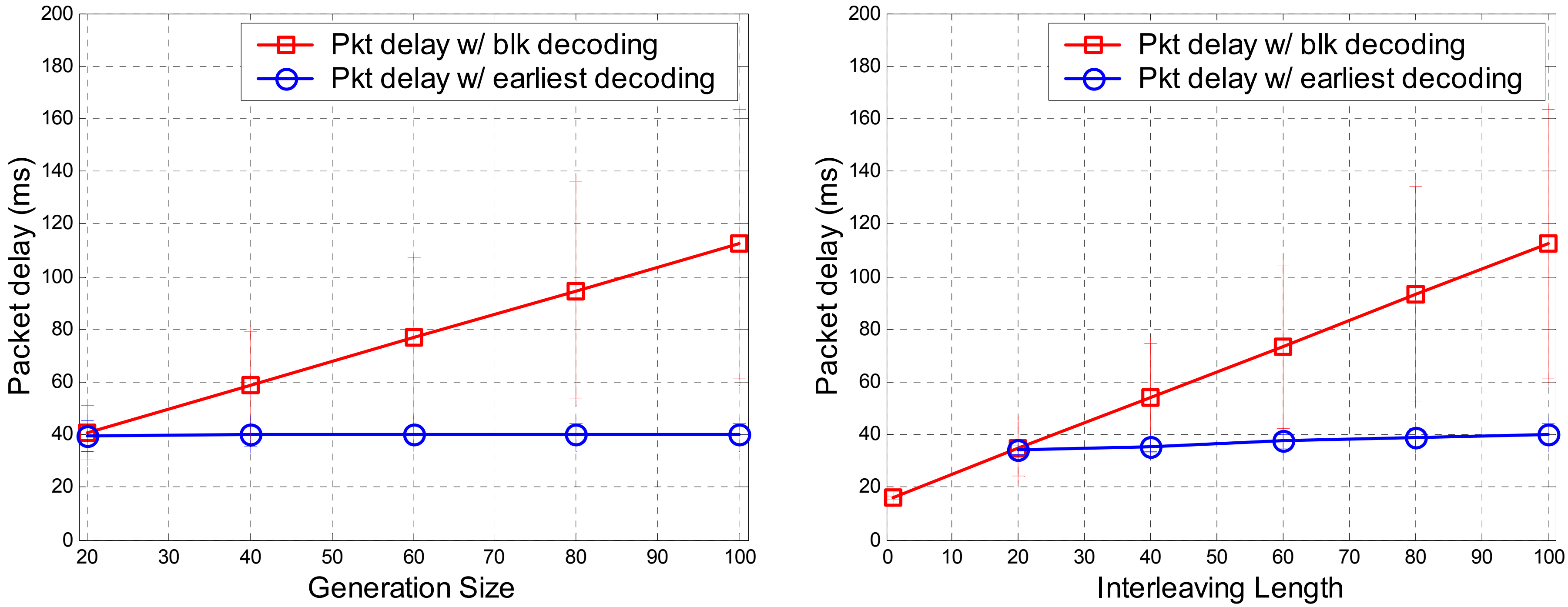

Chicago (450 Mbps)
Pearl Harbor (525 Mbps)
Anaheim (625 Mbps)
Boston (733 Mbps)
SanJose (833 Mbps)
Throughput (Mbps)
Sending rate (Mbps)

Throughput Loss

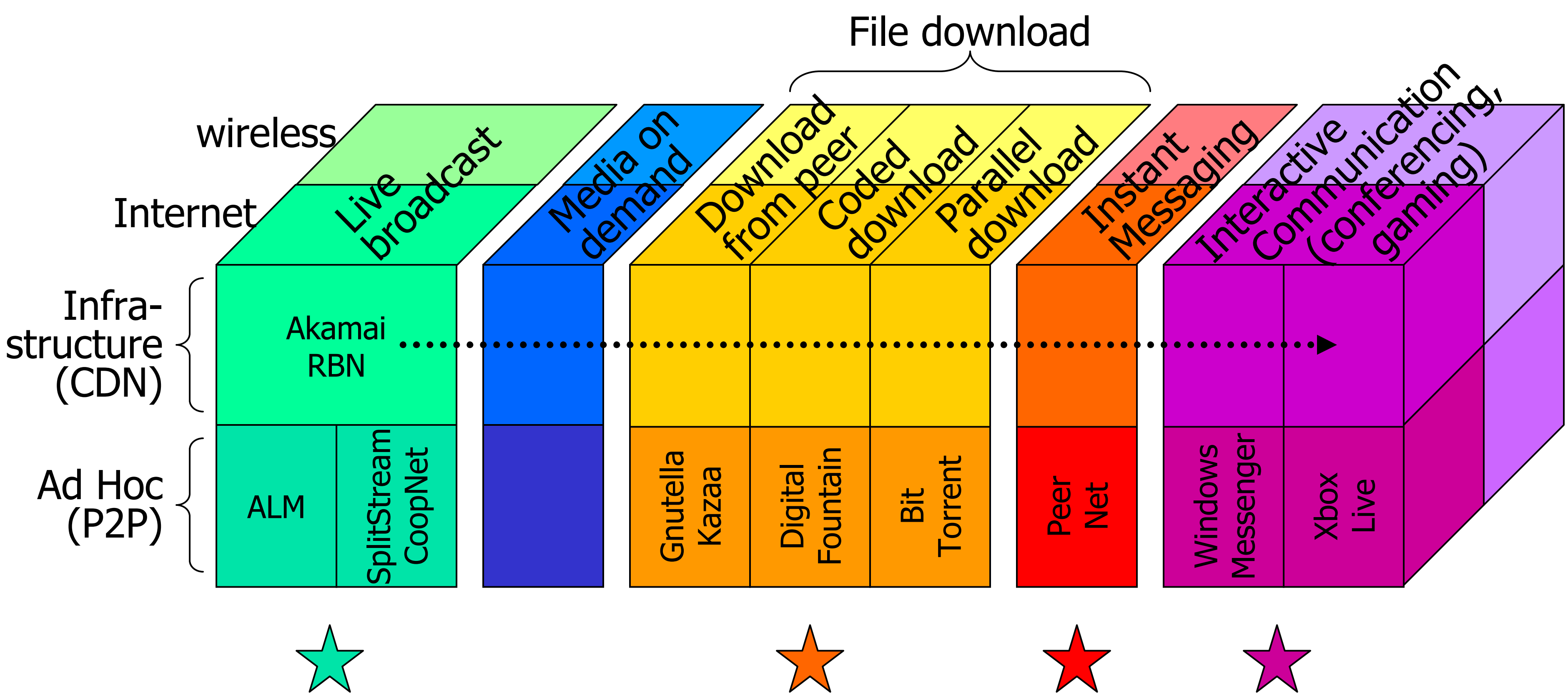




Decoding Delay

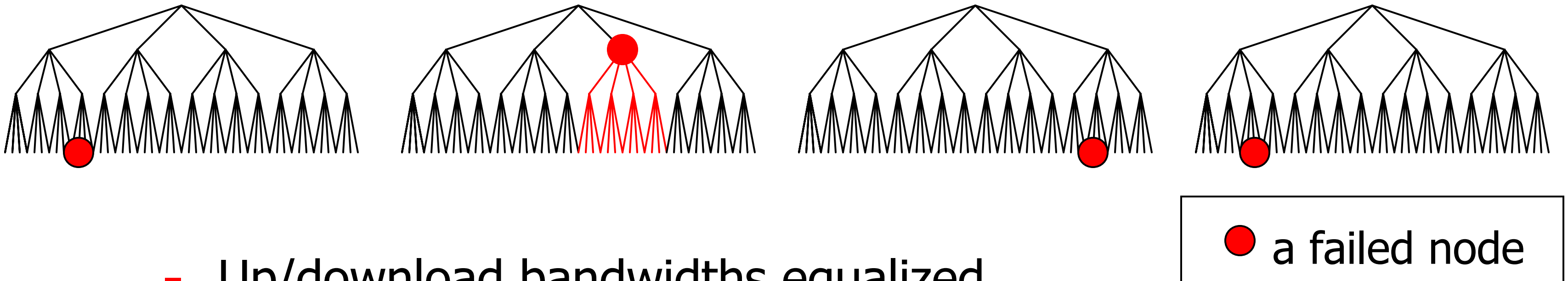


Network Coding for Internet and Wireless Applications

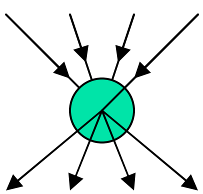


Live Broadcast

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



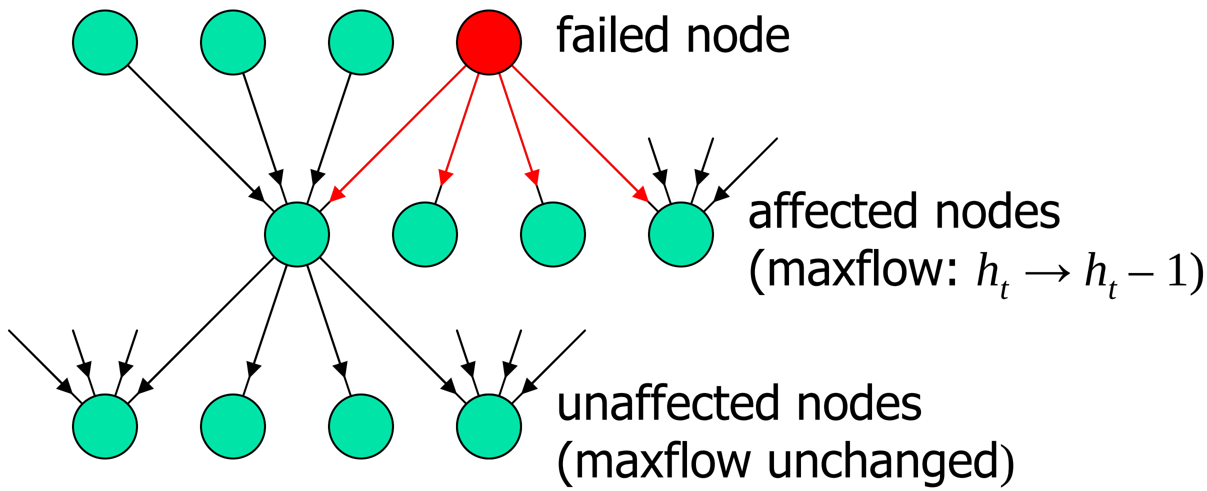
  - Up/download bandwidths equalized



[Padmanabhan, Wang, and Chou, 2003]

Live Broadcast (2)

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

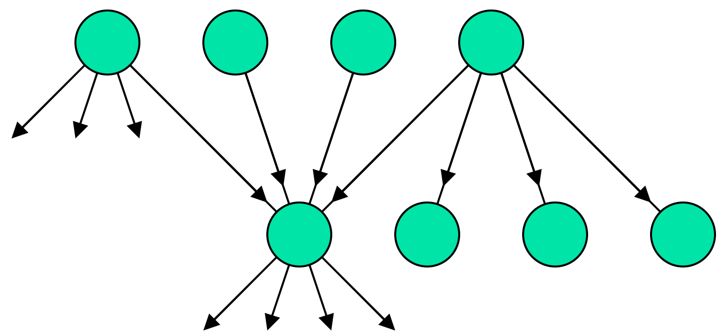


  - ALM/CoopNet average throughput: $(1\text{–}\varepsilon)^{\text{depth}}$ * sending rate
    Network Coding average throughput: $(1\text{–}\varepsilon)$ * sending rate

[Jain, Lovász, and Chou, 2004]

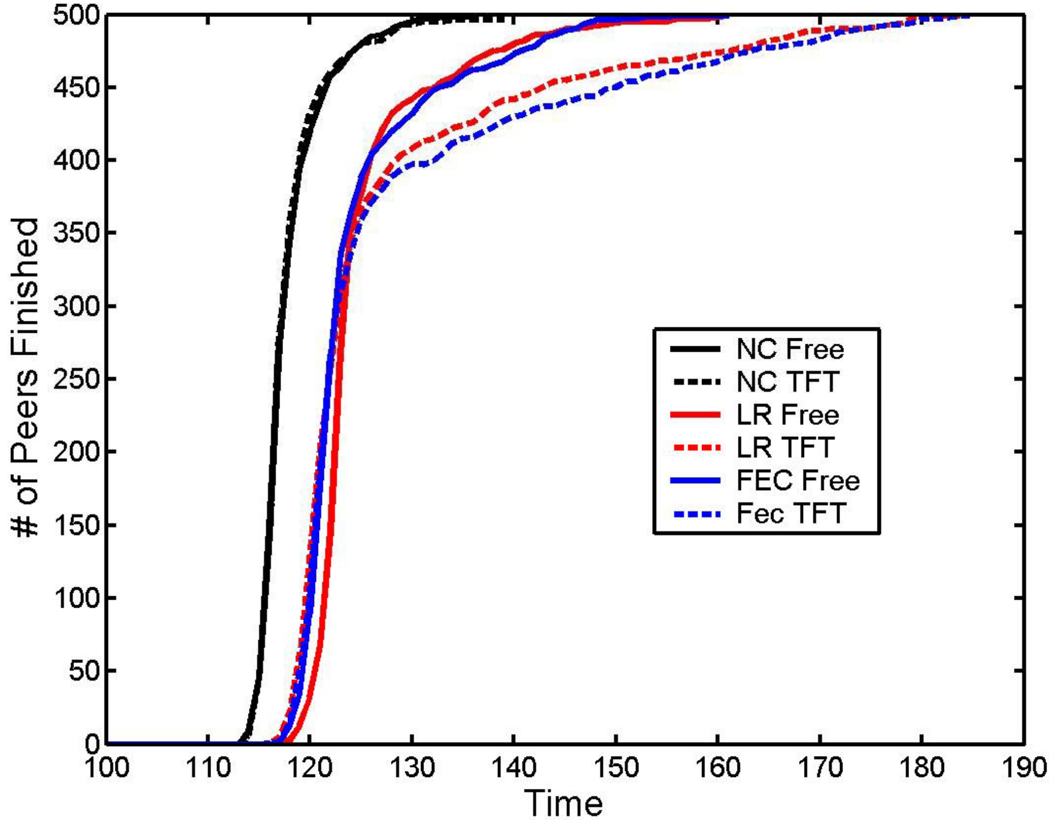
File Download

- 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 ⟷ download time
  - Seamlessly transitions from broadcast to download mode

File Download (2)

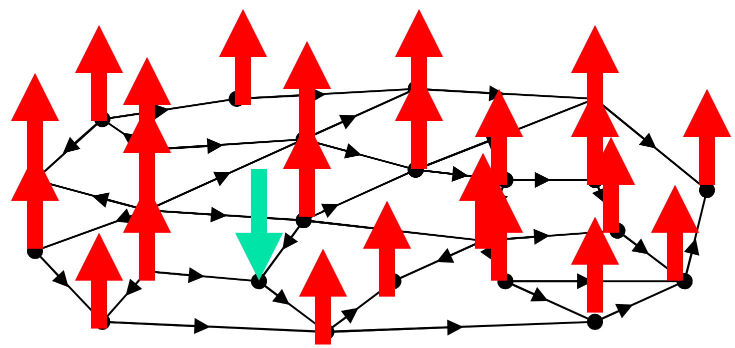


	Mean	Max
LR Free	124.2	161
LR TFT	126.1	185
FEC Free	123.6	159
FEC TFT	127.1	182
NC Free	117.0	136
NC TFT	117.2	139

C. Gkantsidis and P. Rodriguez Rodruiguez, *Network Coding for large scale content distribution,* INFOCOM 2005, reprinted with permission.

Instant Messaging

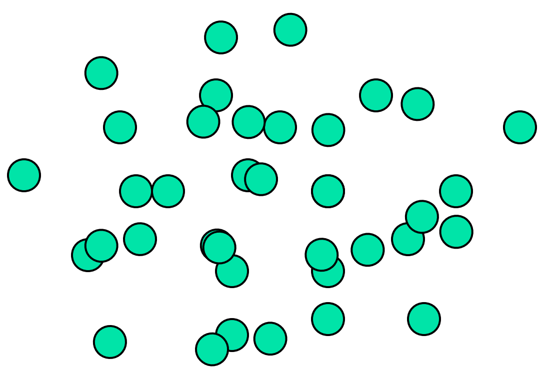
- State-of-the-Art: Flooding (e.g., PeerNet)
  - Peer Name Resolution Protocol (distributed hash table)
  - Maintains group as graph with 3-7 neighbors per node
  - Messaging service: push down at source, pops up at receivers
  - How? Flooding
    - Adaptive, reliable
    - 3-7x over-use



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

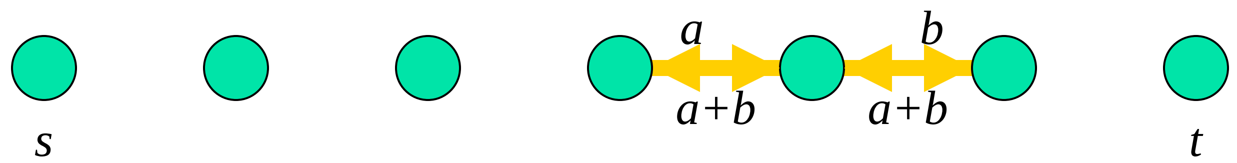
Interactive Communication in mobile ad hoc wireless networks

- 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)

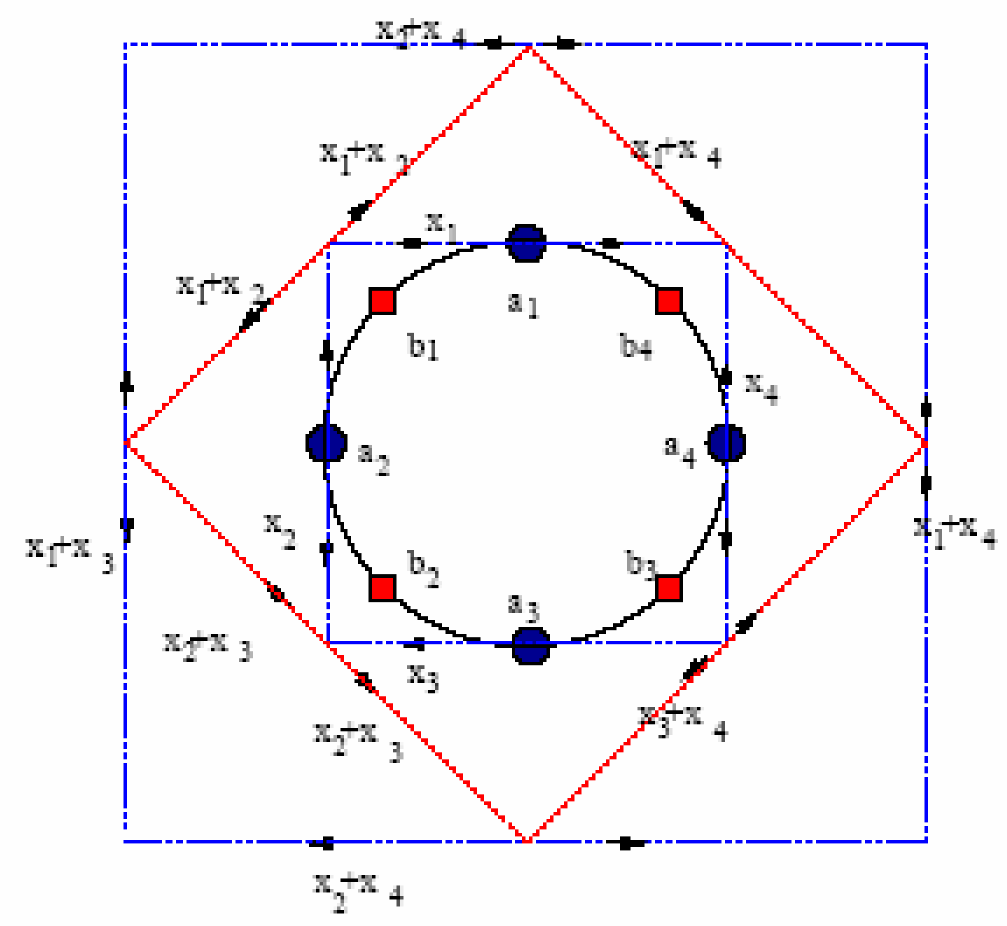
Physical Piggybacking



- 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
  - energy per bit
  - delay

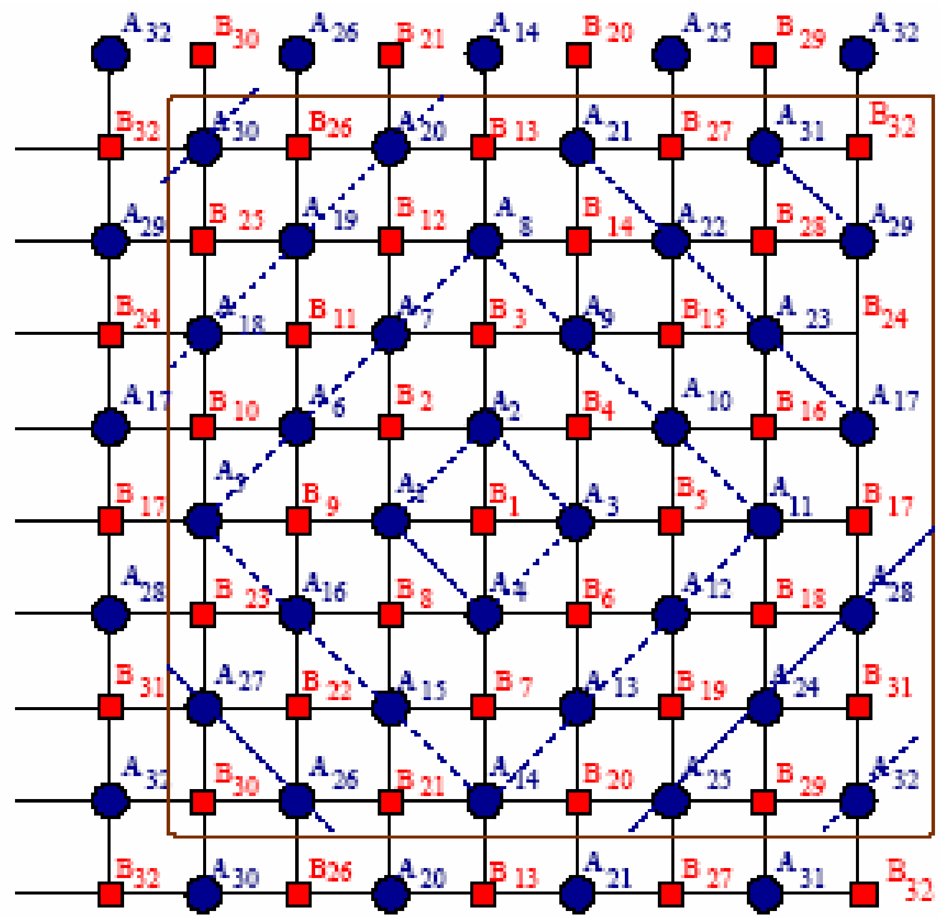
Energy-Efficient Broadcasting in Wireless Ad-hoc Networks

- By Widmer, Fragouli, Le Boudec (NetCod'05)
- 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



Energy-Efficient Broadcasting in Wireless Ad-hoc Networks

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



Simulation Results

- Widmer, Fragouli, Le Boudec
- 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

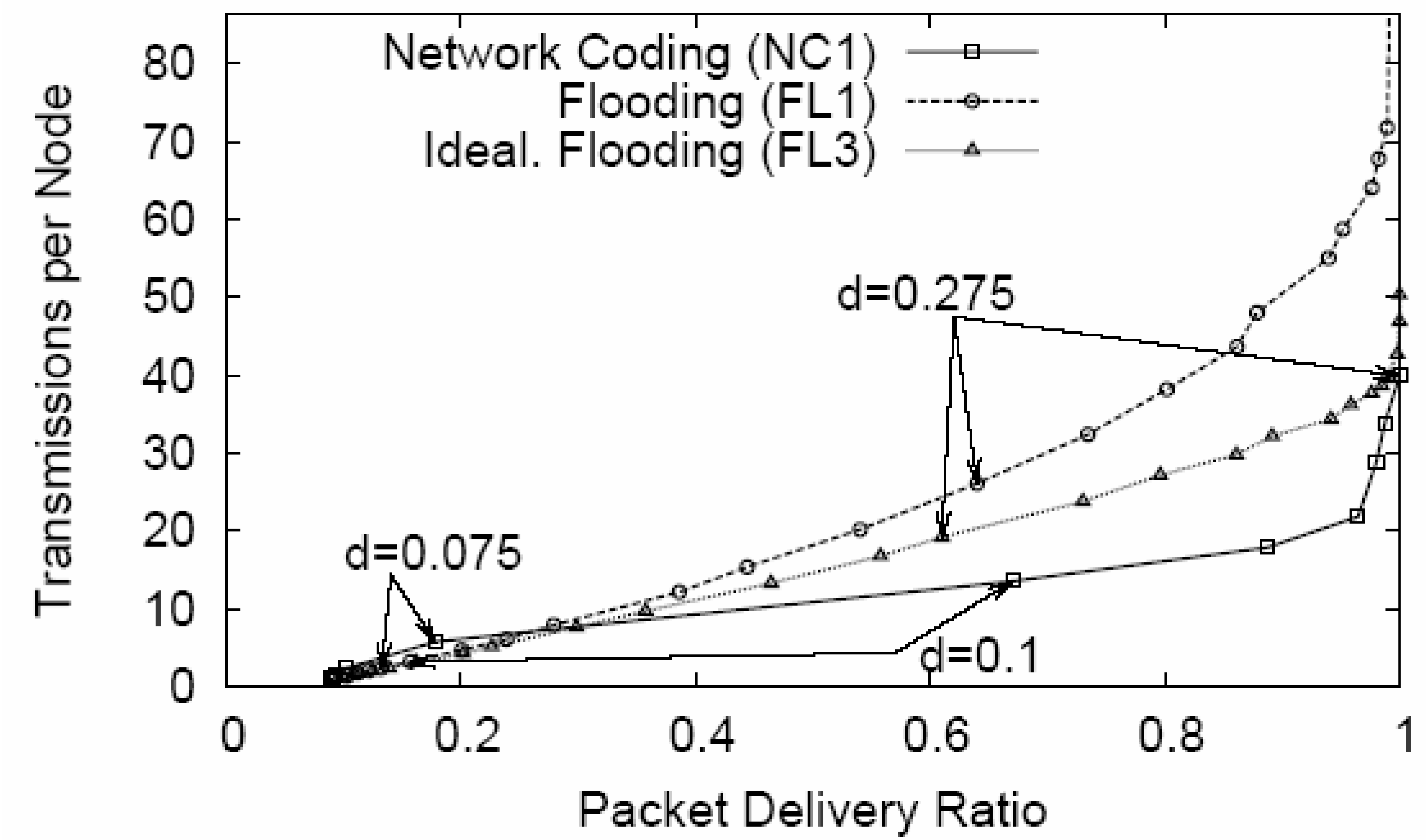
Flooding Algorithm

- 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)

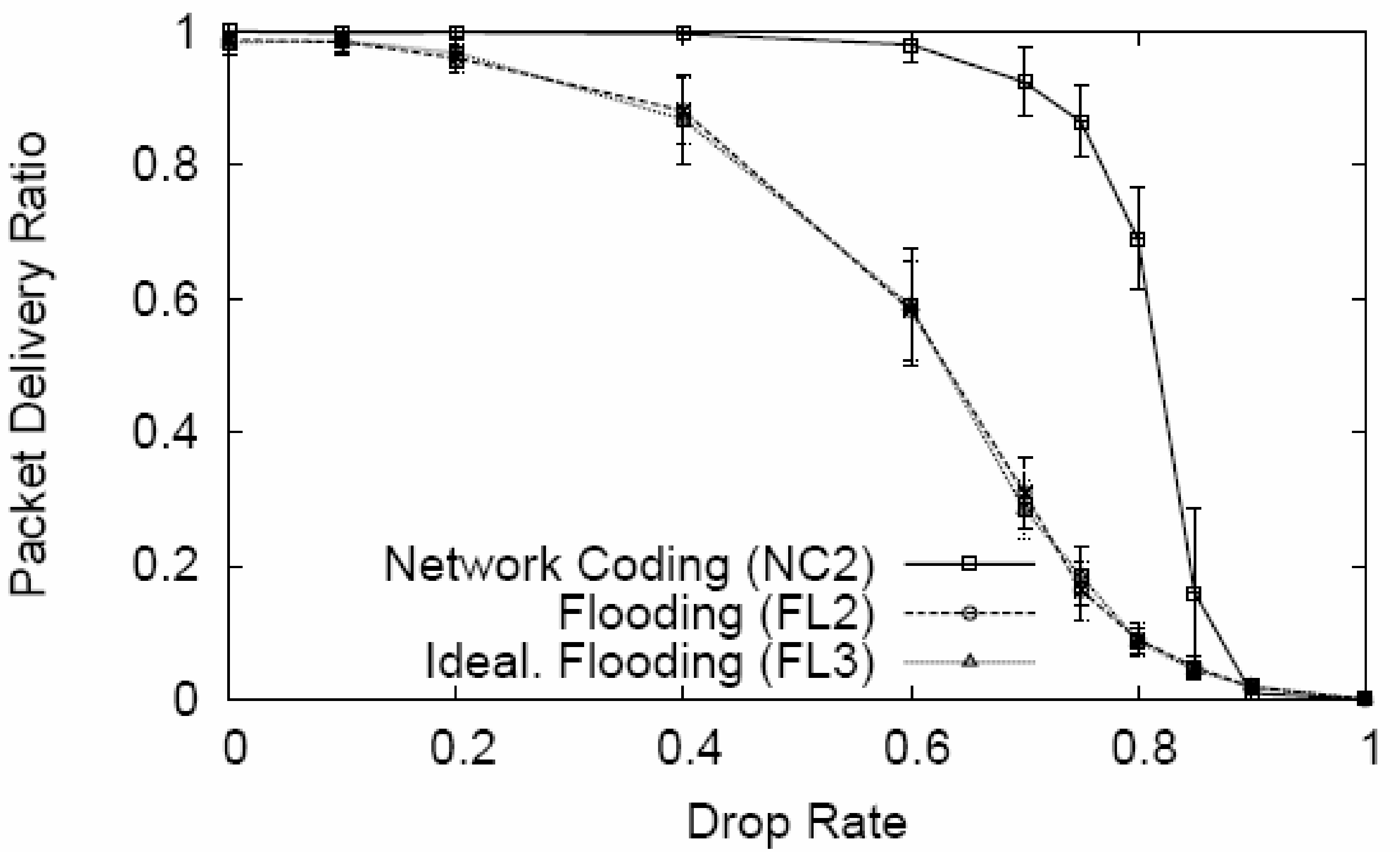
Network Coding Algorithm

- 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 $s < 1$

Transmissions vs Packet delivery ratio



Packet delivery ratio vs Packet drop rate (w/ d=0.5)



Summary

- Network Coding is Practical
  - Packet Format
  - Buffering
- Network Coding can improve performance
  - in IP or wireless networks
  - for live broadcast, file download, messaging, interactive communication
  - by improving throughput, robustness, delay, energy consumption, manageability
  - even if all nodes are receivers, even for point-to-point communication