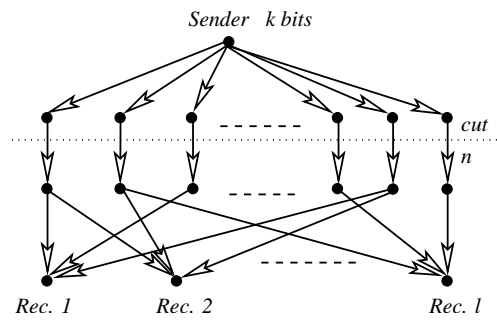


III — More about the multicast



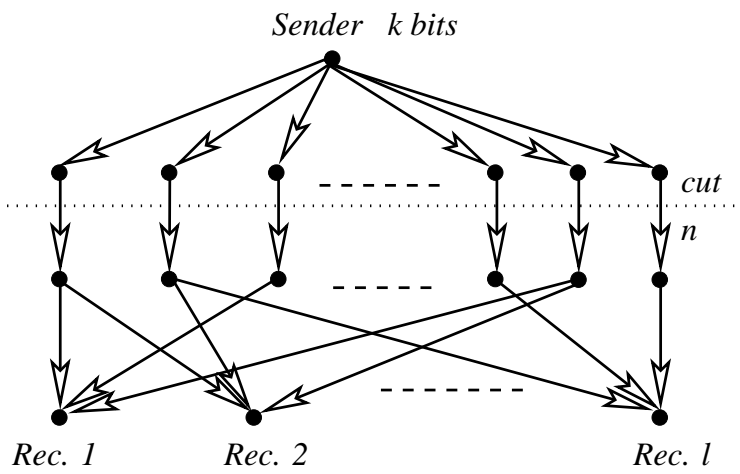
Questions

- Do we really need codes?
- What alphabet sizes do we need?
- How do we find solutions?
- Bidirectional links?

Do we really need codes?

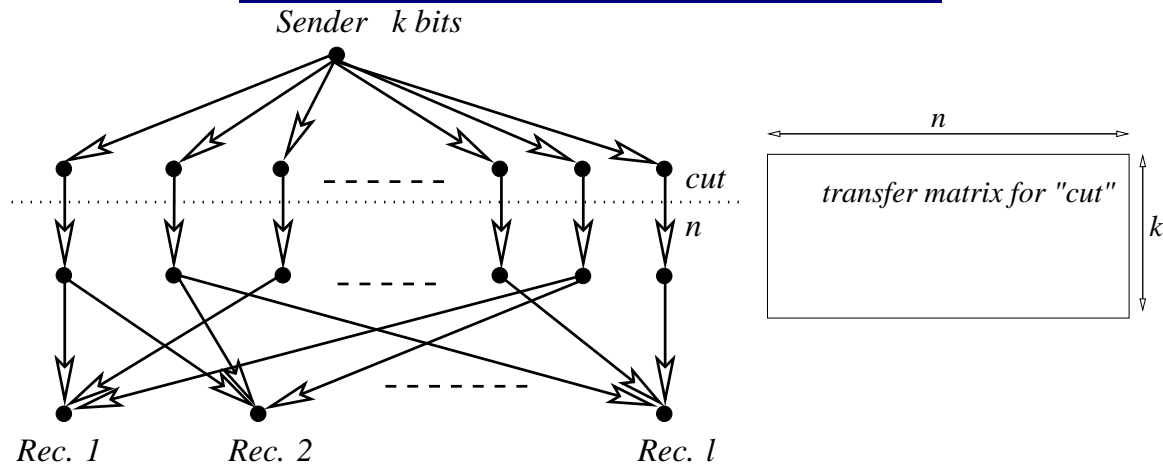
Is the performance gap between the network coding solution and routing, i.e. packing directed Steiner trees bounded?

A simple example:



Each receiver picks out one of $\binom{n}{k}$ possible middle layer links

Do we really need codes?



The transfer matrix from the transmitter to the "cut" has to satisfy that $k \times k$ submatrix has full rank!

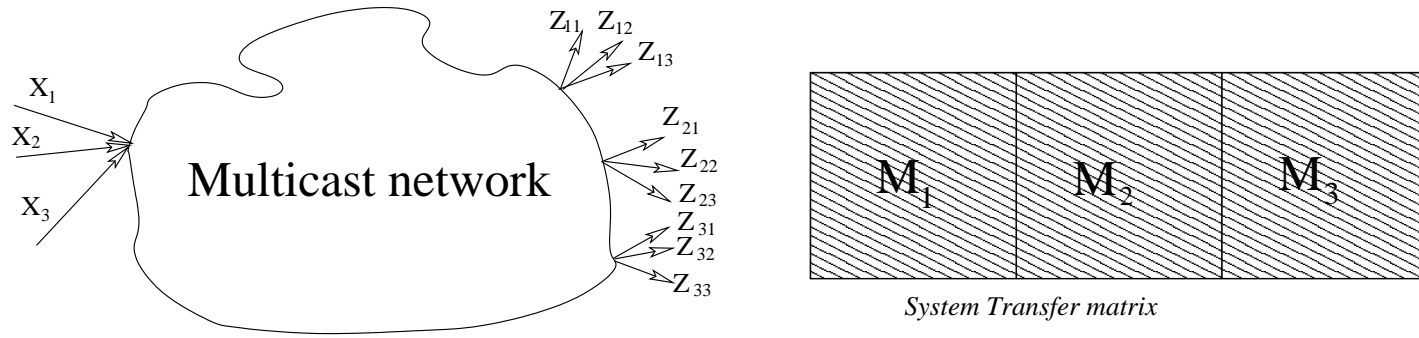
\Rightarrow The field size is at least in the same order as n
(the MDS conjecture)

A lower bound

Theorem There exist multicast problems with T receivers such that the minimum field size required for a solution grows as $O(\sqrt{T})$.

Theorem There exist multicast problems such that the gap between routing and network coded strategies is arbitrarily large.

How do we find solutions for the Multicast?



$$\mathcal{C} = \{(v, u_1, \mathcal{X}(v)), (v, u_2, \mathcal{X}(v)), \dots, (v, u_K, \mathcal{X}(v))\}$$

M is a $|\mathcal{X}(v)| \times K|\mathcal{X}(v)|$ matrix.

$$m_i(\underline{\xi}) = \det(M_i(\underline{\xi}))$$

Choose the coefficients in $\overline{\mathbb{F}}$ so that all $m_i(\underline{\xi})$ are unequal to zero.

The degree of each $m_i(\underline{\xi})$ is at most $|\mathcal{X}(v)|$

Multicast:

An algorithm to find a vector \underline{a} such that $F(\underline{a}) \neq 0$ for a polynomial F .

Input: A polynomial F in indeterminates $\xi_1, \xi_2, \dots, \xi_n$, integers: $i = 1, t = 1$

Iteration:

1. Find the maximal degree δ of F in any variable ξ_j and let i be the smallest number such that $2^i > \delta$.

2. Find an element a_t in \mathbb{F}_{2^i} such that $F(\underline{\xi})|_{\xi_t=a_t} \neq 0$ and let $F \leftarrow F(\underline{\xi})|_{\xi_t=a_t}$.
3. If $t = n$ then halt, else $t \leftarrow t + 1$, goto 2).

Output: (a_1, a_2, \dots, a_n) .

The crucial step is 2) which is successful if the fieldsize is larger than the degree of F .

Multicast:

Let $(\mathcal{G}, \mathcal{C})$ be a multicast network coding problem with T receivers and R symbols transmitted per time unit. There exists a solution for $(\mathcal{G}, \mathcal{C})$ over a finite field \mathbb{F}_{2^m} with

$$m \leq \lceil \log_2(TR + 1) \rceil.$$

(A more careful analysis shows that a field \mathbb{F}_{2^m} with $m \leq \lceil \log_2(T) \rceil$ or $\mathbb{F} \geq T$)

Multicast:

For any multicast networking problem with T receivers there always exists a solution over an alphabet which is at least as large as T .

Conversely:

There exist multicast networking problems with T receivers such that the minimum alphabet size is bounded below by $\sqrt{T} - o(1)$.

(In practice - just try the random approach...)

A different approach...

S.-Y. R. Li, R. W. Yeung, and N. Cai. "Linear network coding". IEEE Transactions on Information Theory , February, 2003

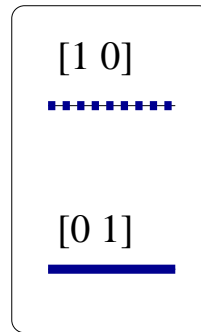
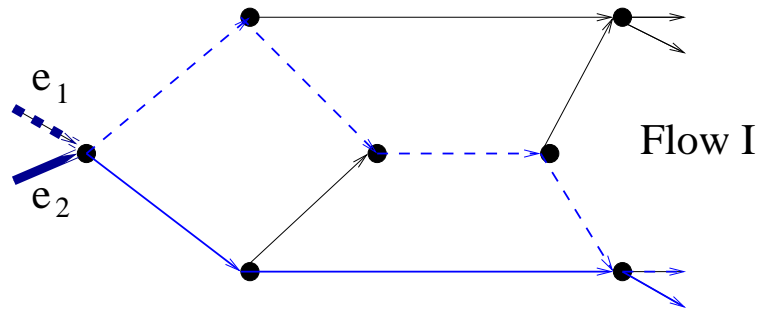
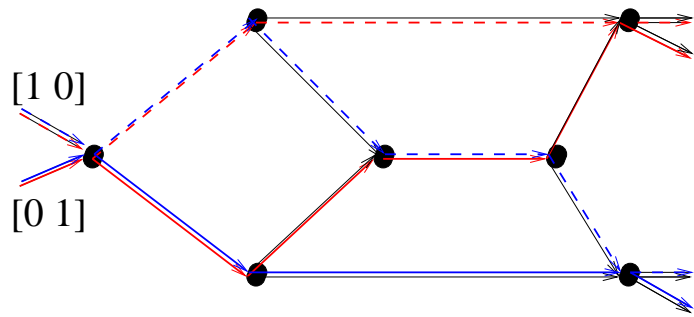
S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egnér, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," IEEE Transactions on Information Theory. Submitted July 2003.

A flow based approach that carefully constructs a solution in polynomial time.

A different approach...

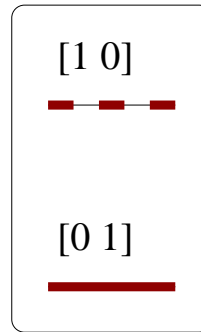
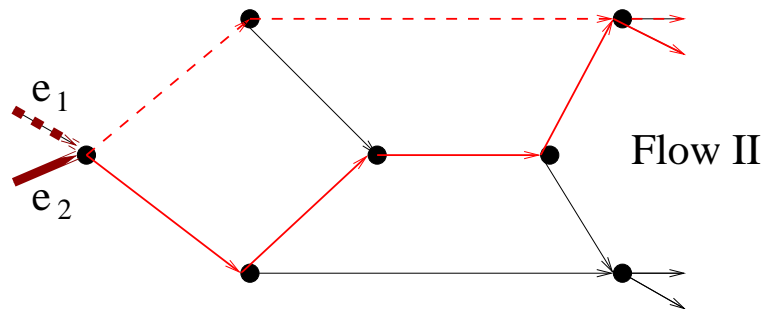
A solution for acyclic networks is constructed “one link at a time” starting at the source.

Each flow to a receiver is being treated as a set of disjoint paths with the set of edges that was processed last (the frontier set) having to form a full rank matrix

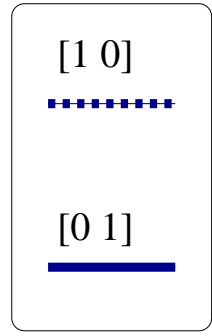
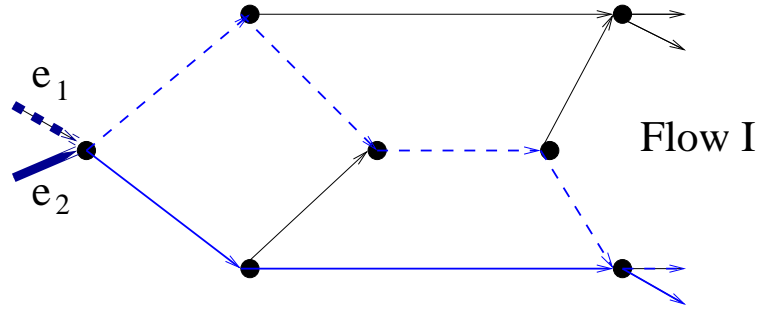
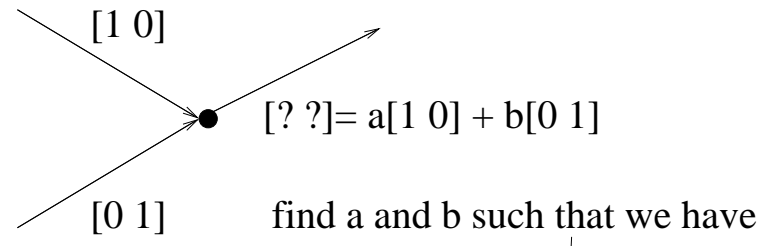
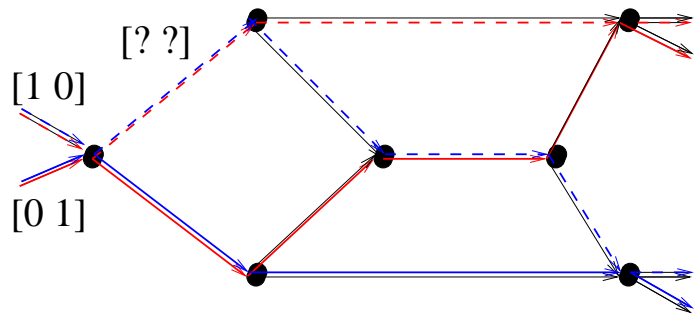


Frontier Sets

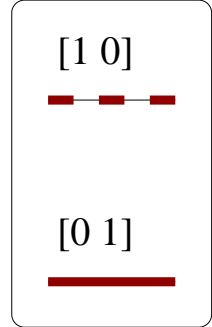
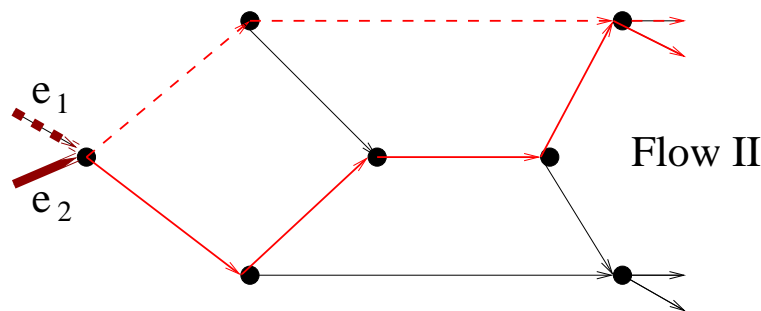
full rank at all times



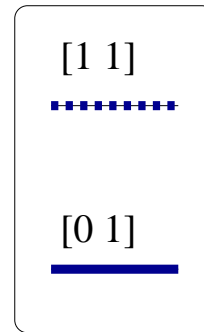
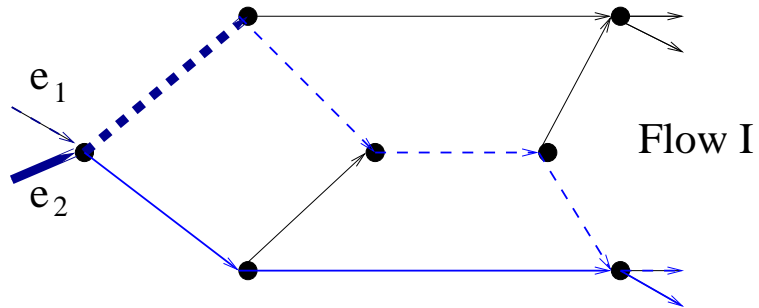
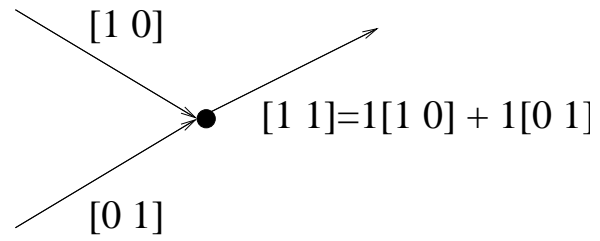
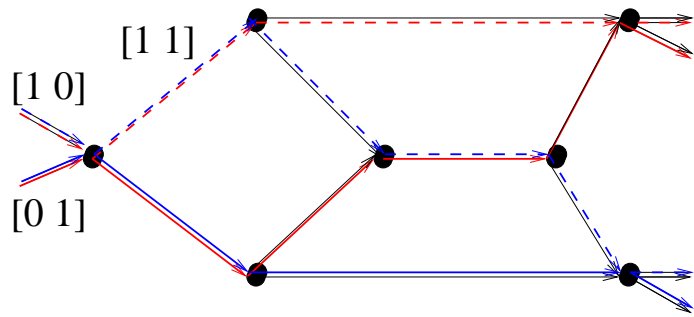
Frontier Sets



Frontier Sets
full rank at all times

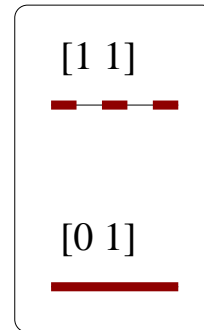
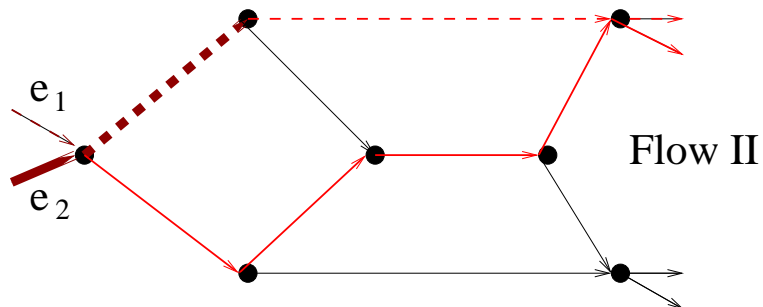


Frontier Sets
full rank at all times



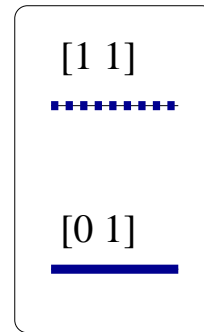
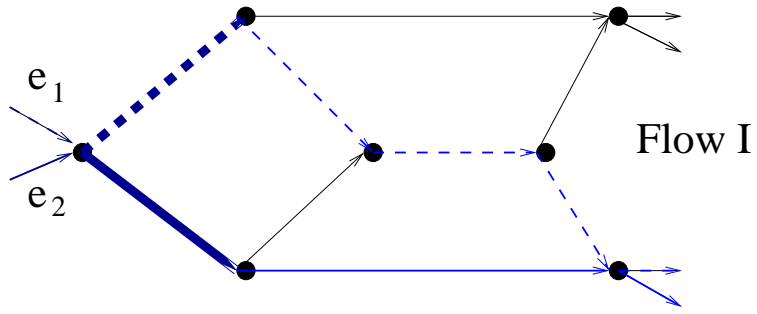
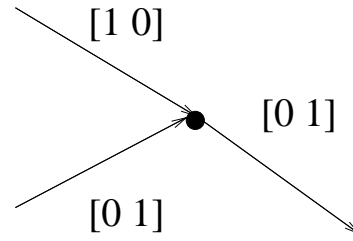
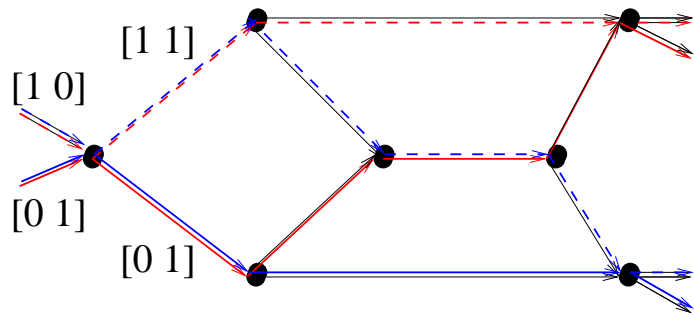
Frontier Sets

full rank at all times



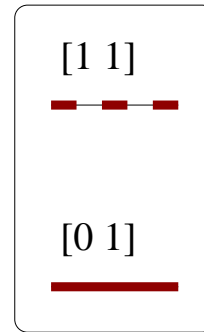
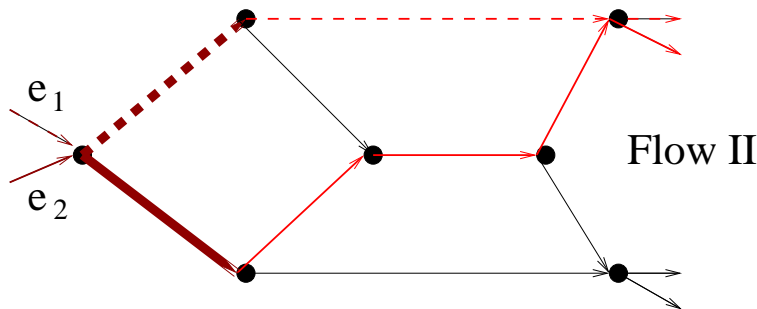
Frontier Sets

full rank at all times



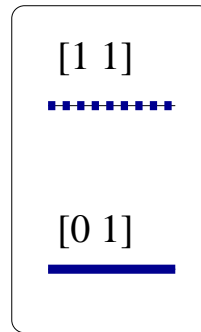
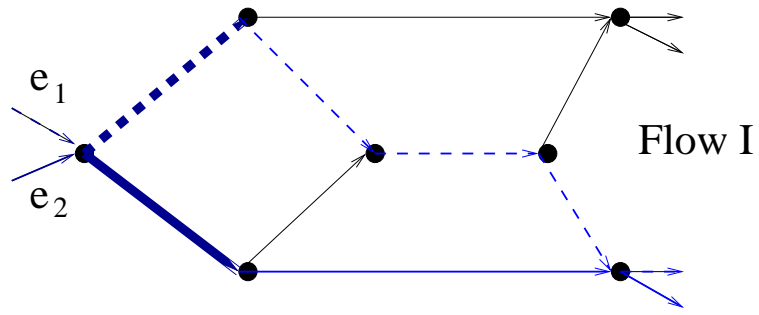
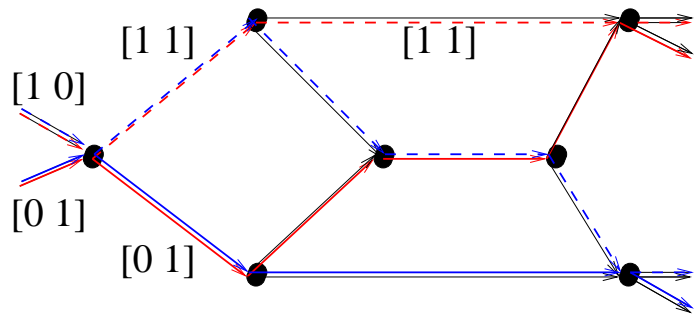
Frontier Sets

full rank at all times



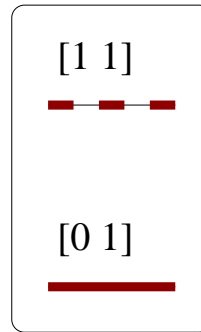
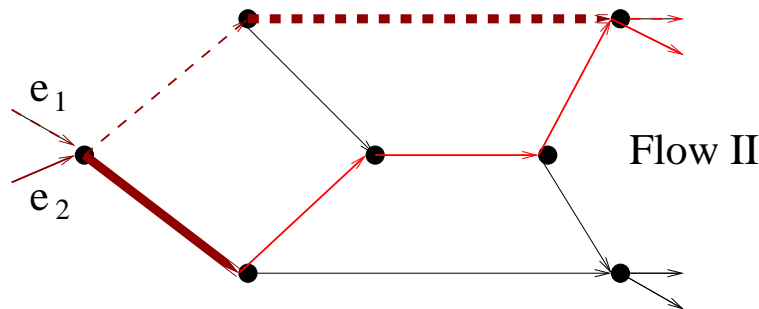
Frontier Sets

full rank at all times



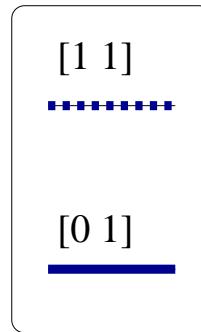
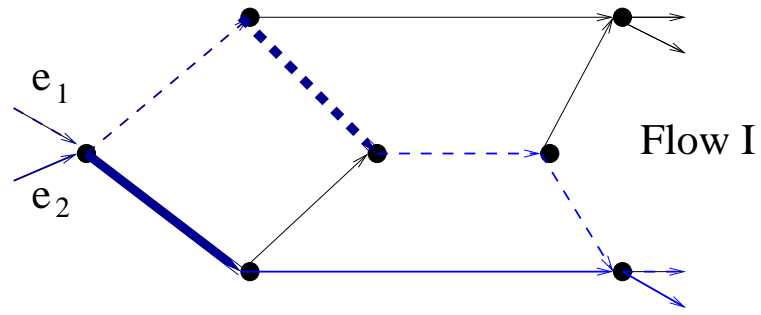
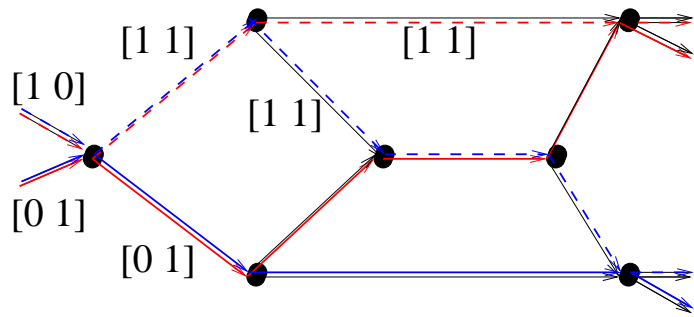
Frontier Sets

full rank at all times

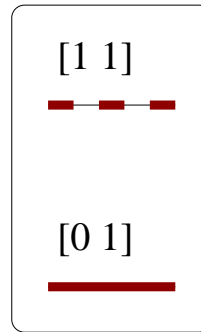
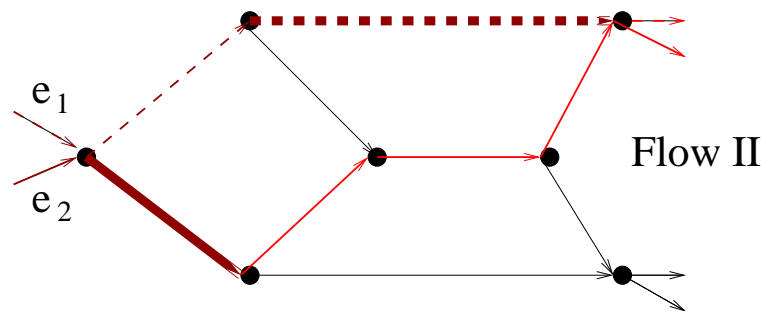


Frontier Sets

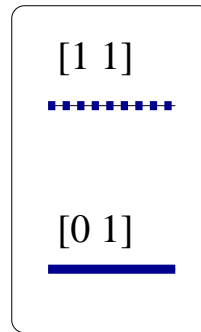
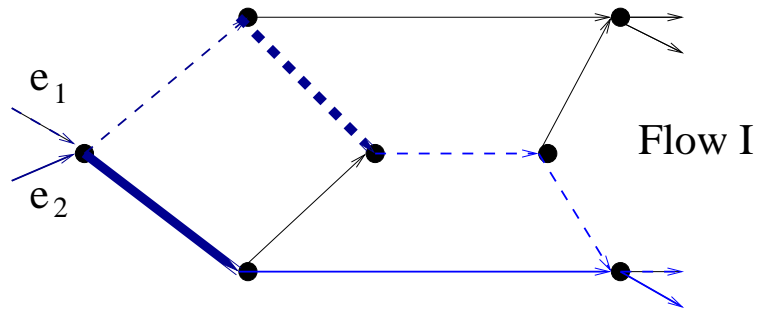
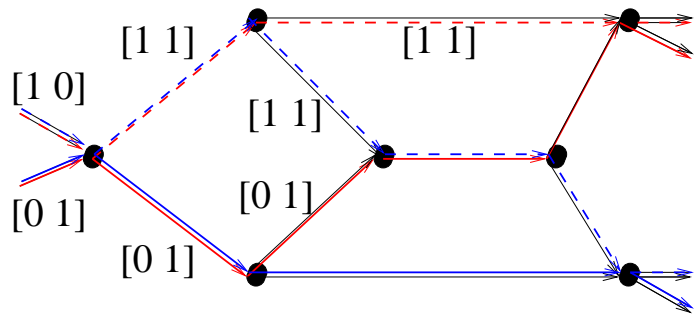
full rank at all times



Frontier Sets
 full rank at all times

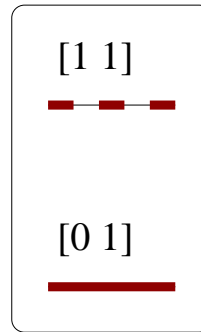
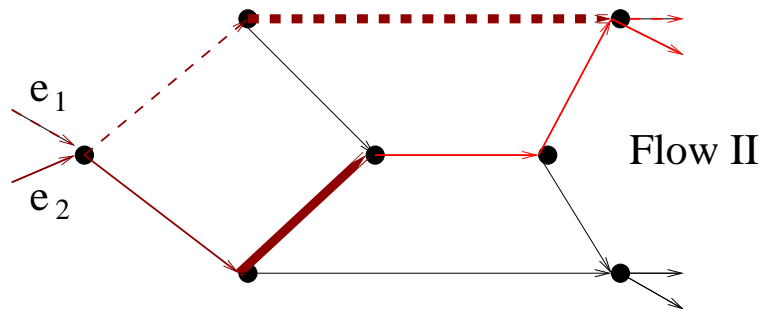


Frontier Sets
 full rank at all times



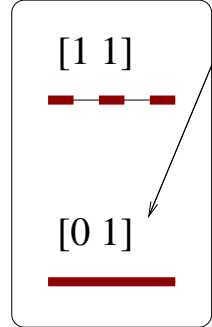
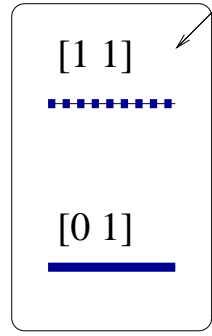
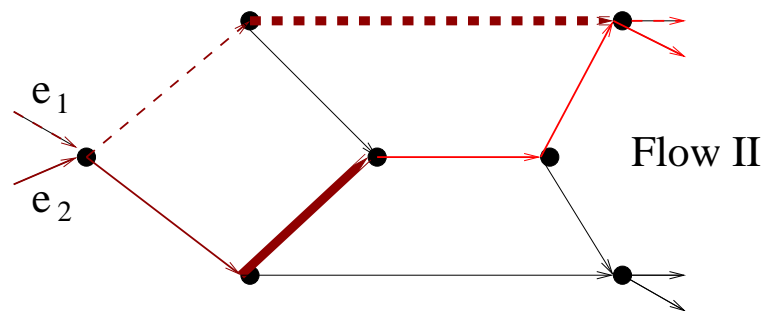
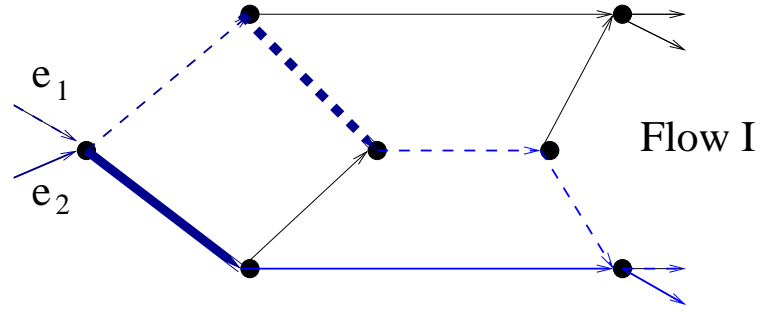
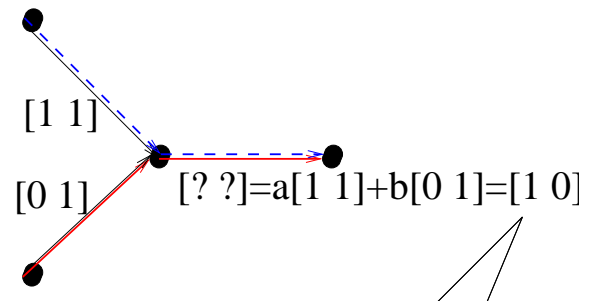
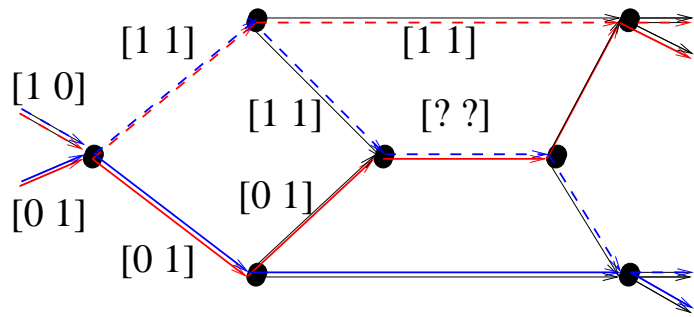
Frontier Sets

full rank at all times

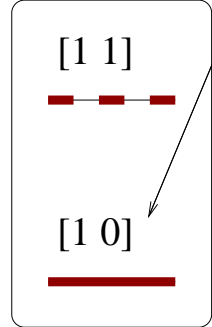
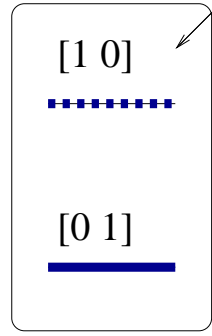
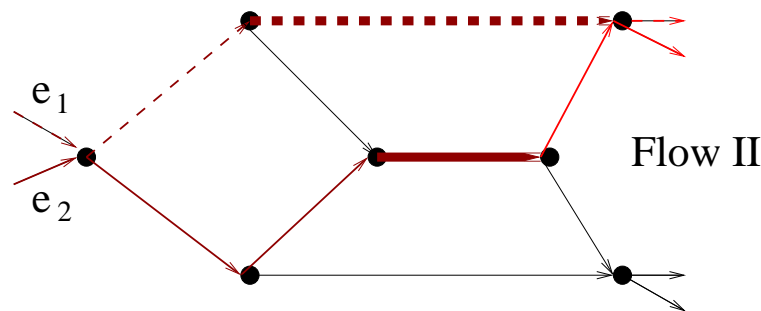
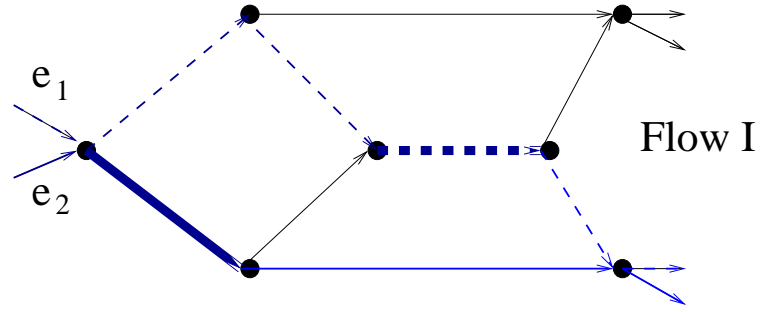
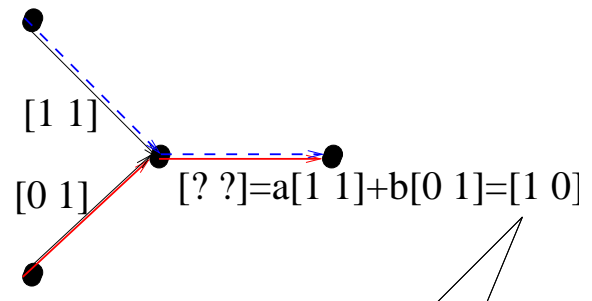
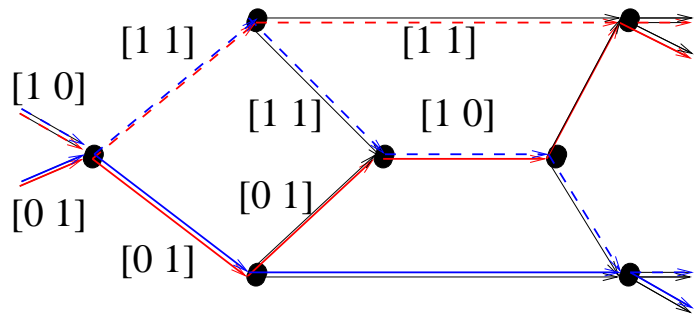


Frontier Sets

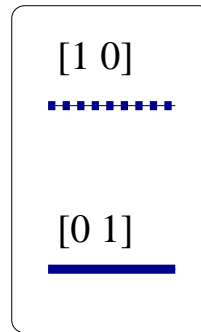
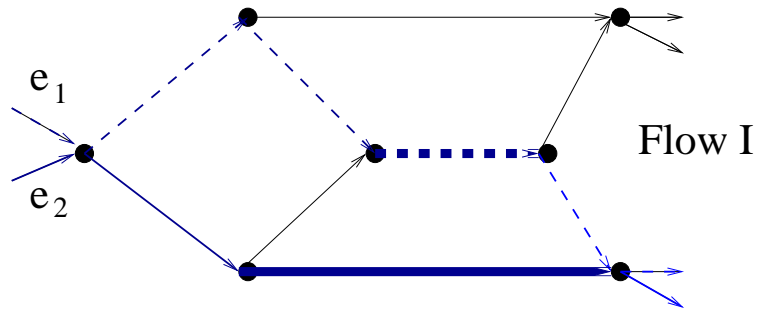
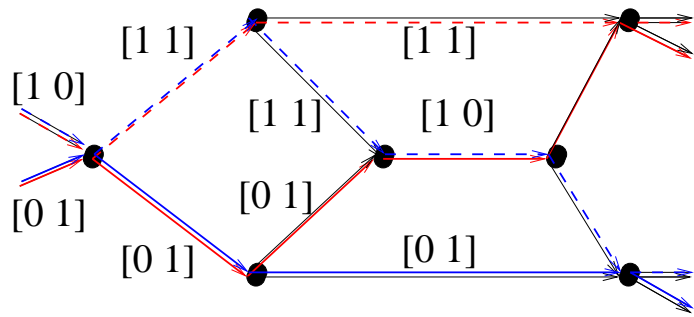
full rank at all times



Frontier Sets
full rank at all times

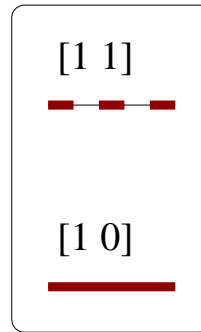
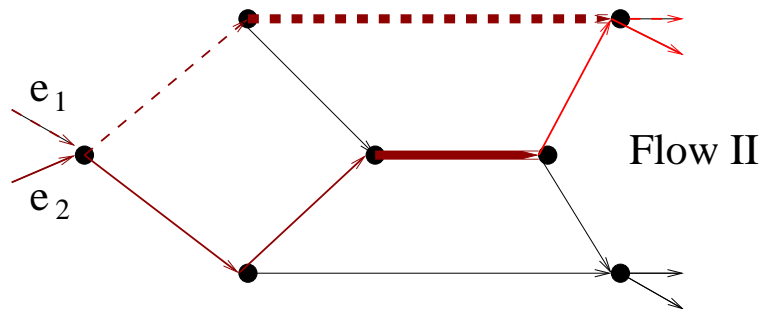


Frontier Sets
full rank at all times

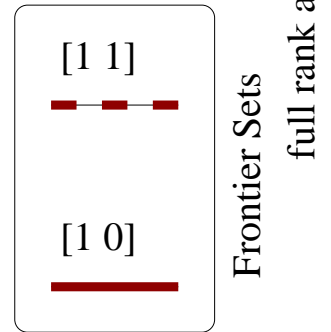
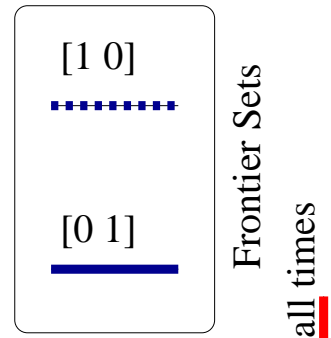
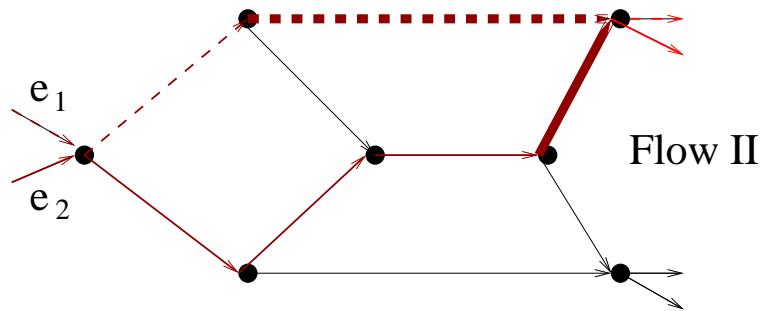
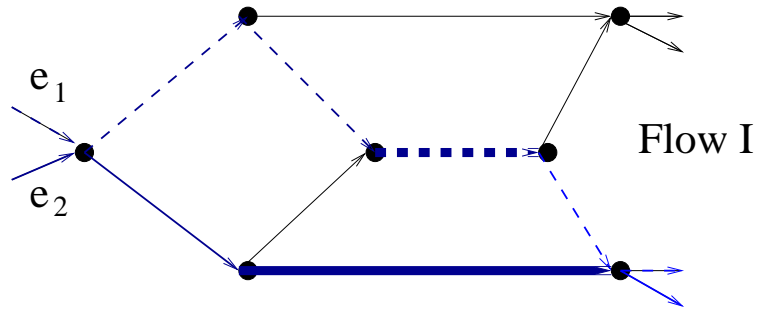
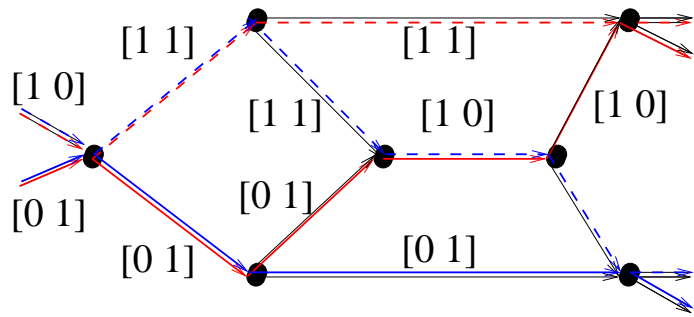


Frontier Sets

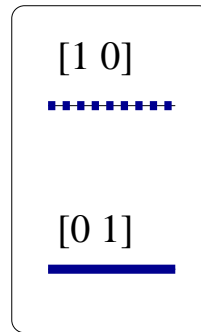
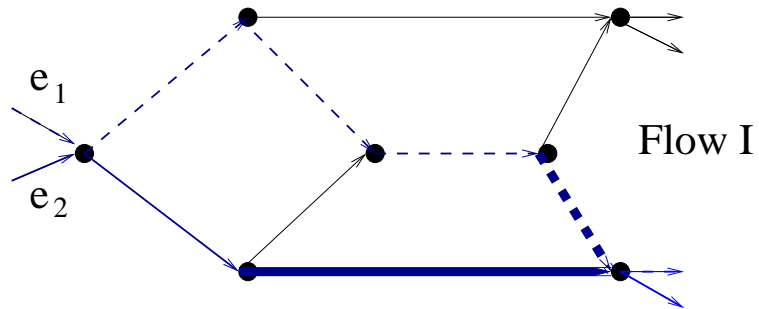
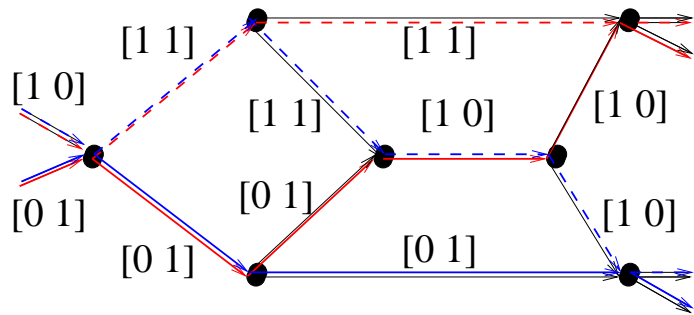
full rank at all times



Frontier Sets

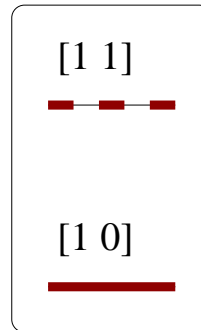
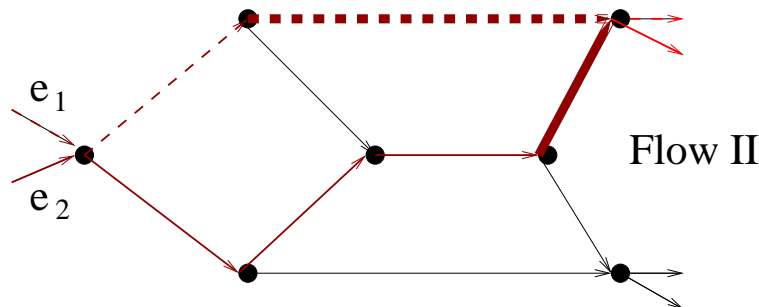


Frontier Sets
full rank at all times



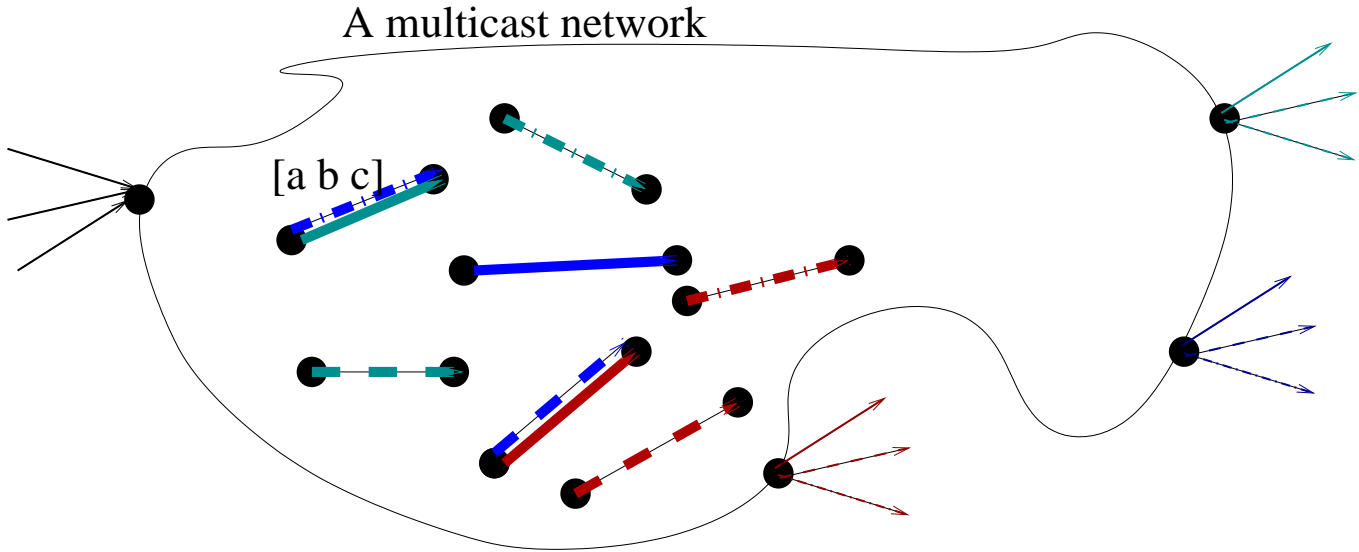
Frontier Sets

full rank at all times





Frontier Sets

The algorithm of Jaggi, Sanders et al.



The frontier sets of a multicast to three receivers

-  [a b c]
-  [a' b' c']
-  [a'' b'' c'']

$\begin{bmatrix} a & b & c \\ a' & b' & c' \\ a'' & b'' & c'' \end{bmatrix}$
 has full rank
 for all colors

The algorithm of Jaggi, Sanders et al.

Theorem [Jaggi, Sanders, et al] Let $(\mathcal{G}, \mathcal{C})$ be a multicast network coding problem with E edges, R symbols to be transmitted simultaneously and T receivers. There exists a linear network coding solution for $(\mathcal{G}, \mathcal{C})$ over a finite field \mathbb{F} if $|\mathbb{F}| > T$. Moreover this solution can be found in time $O(E \cdot T \cdot R(R + T))$.

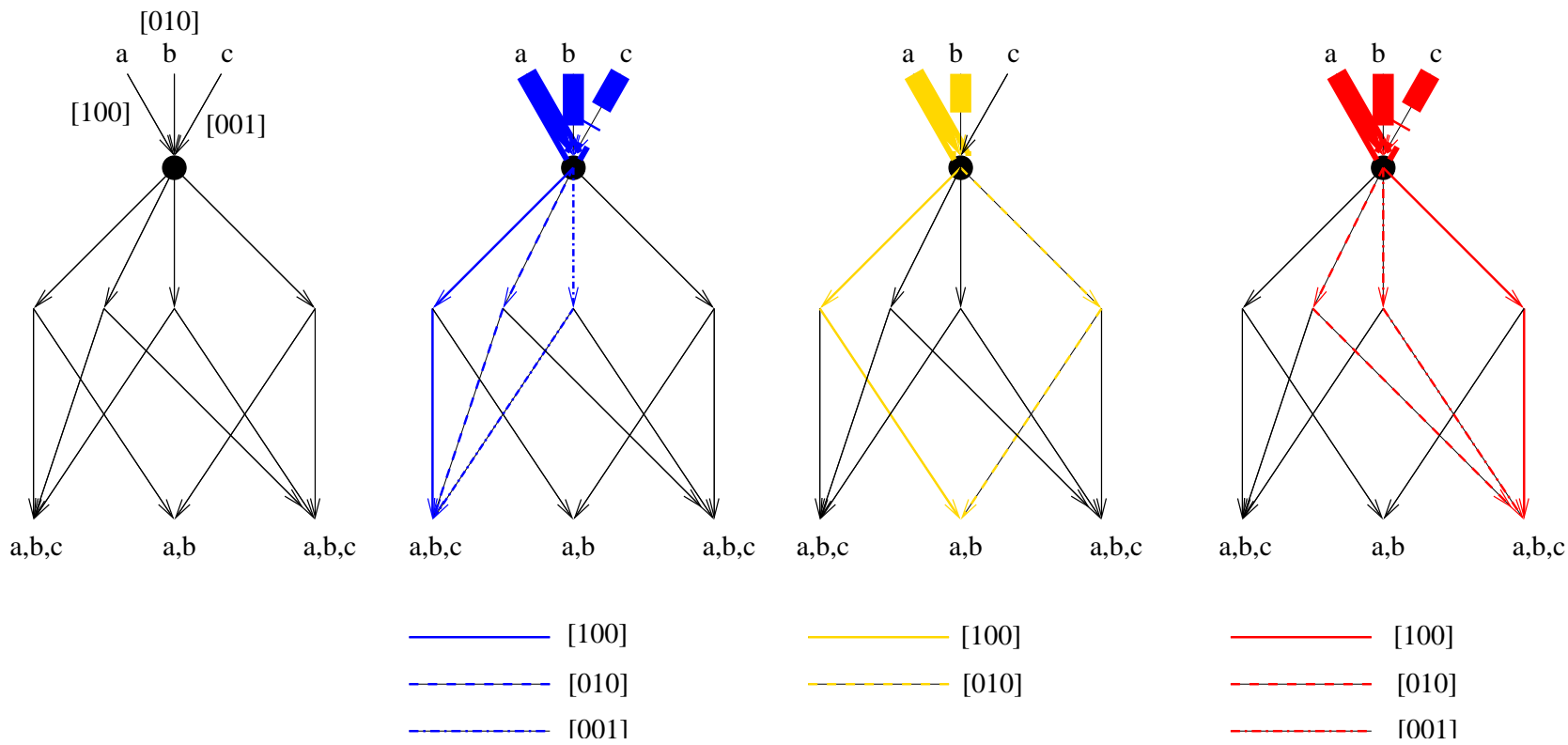
The algorithm of Jaggi, Sanders et al.

Theorem [Jaggi, Sanders, et al] Let $(\mathcal{G}, \mathcal{C})$ be a multicast network coding problem with E edges, R symbols to be transmitted simultaneously and T receivers. There exists a linear network coding solution for $(\mathcal{G}, \mathcal{C})$ over a finite field \mathbb{F} if $|\mathbb{F}| > T$. Moreover this solution can be found in time $O(E \cdot T \cdot R(R + T))$.

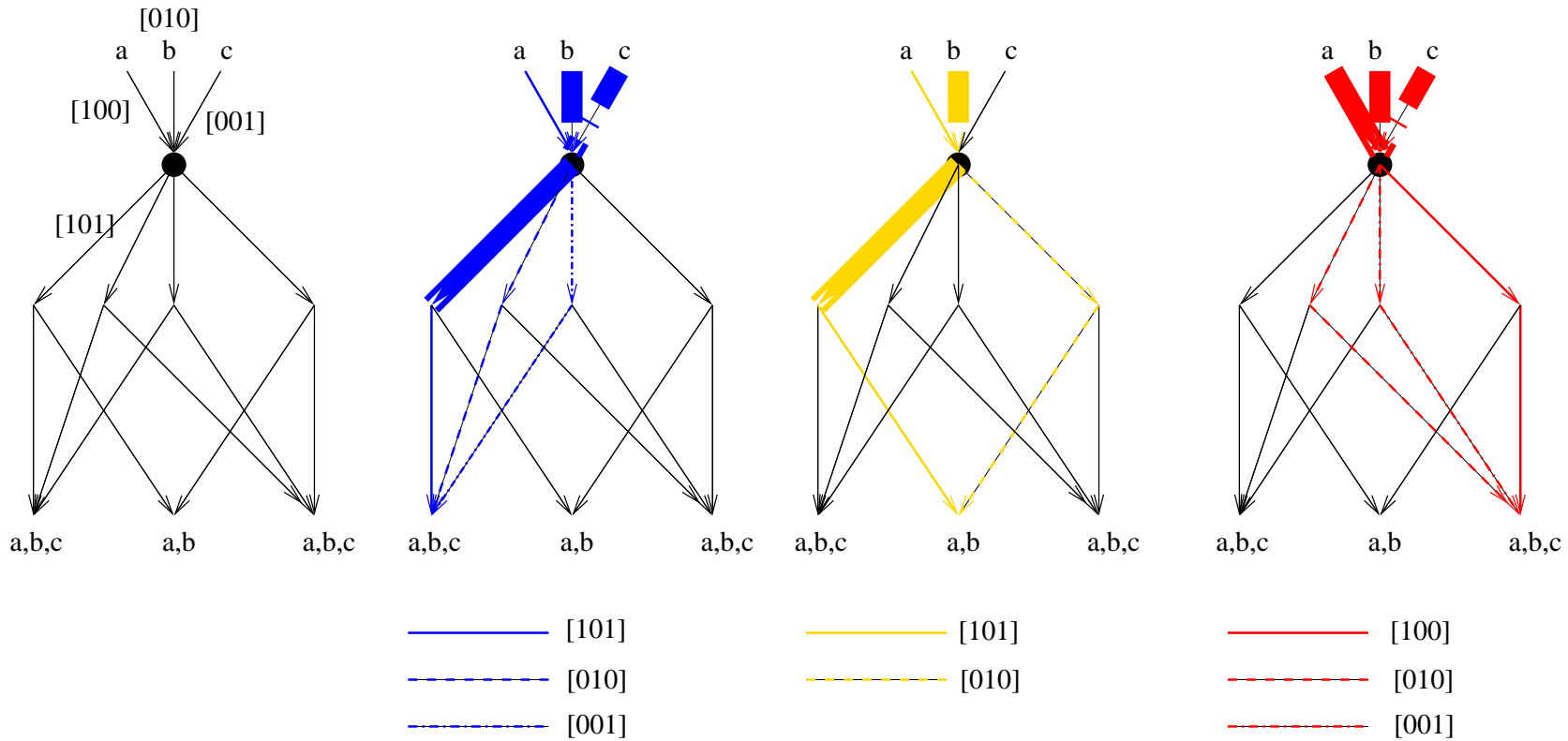
(In practice - still just try the random approach...)

The “link growth” algorithm can be modified such that it is applicable to all the generalizations of the previous session. Example: “Two-Level Multicast”

Two-Level Multicast

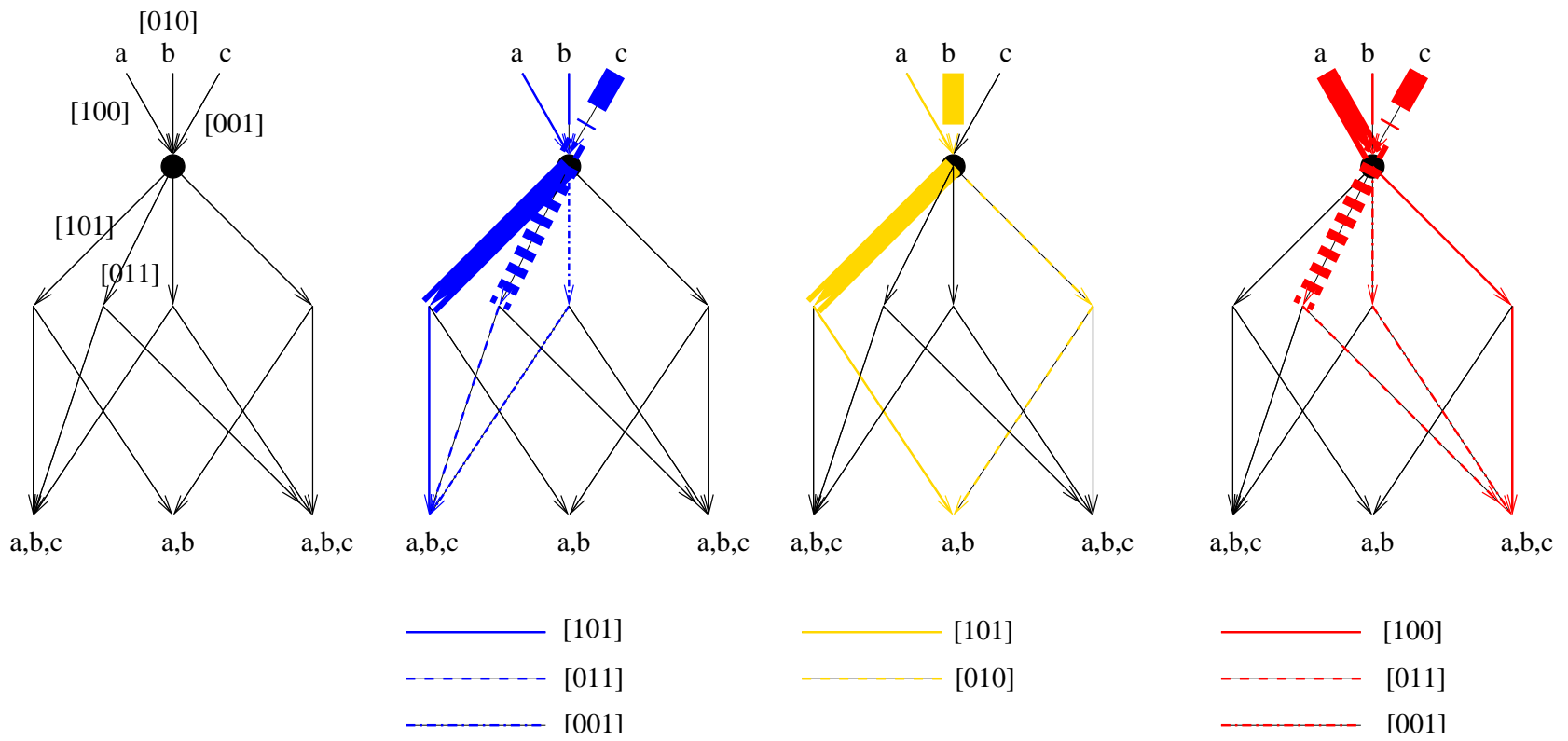


Two-Level Multicast

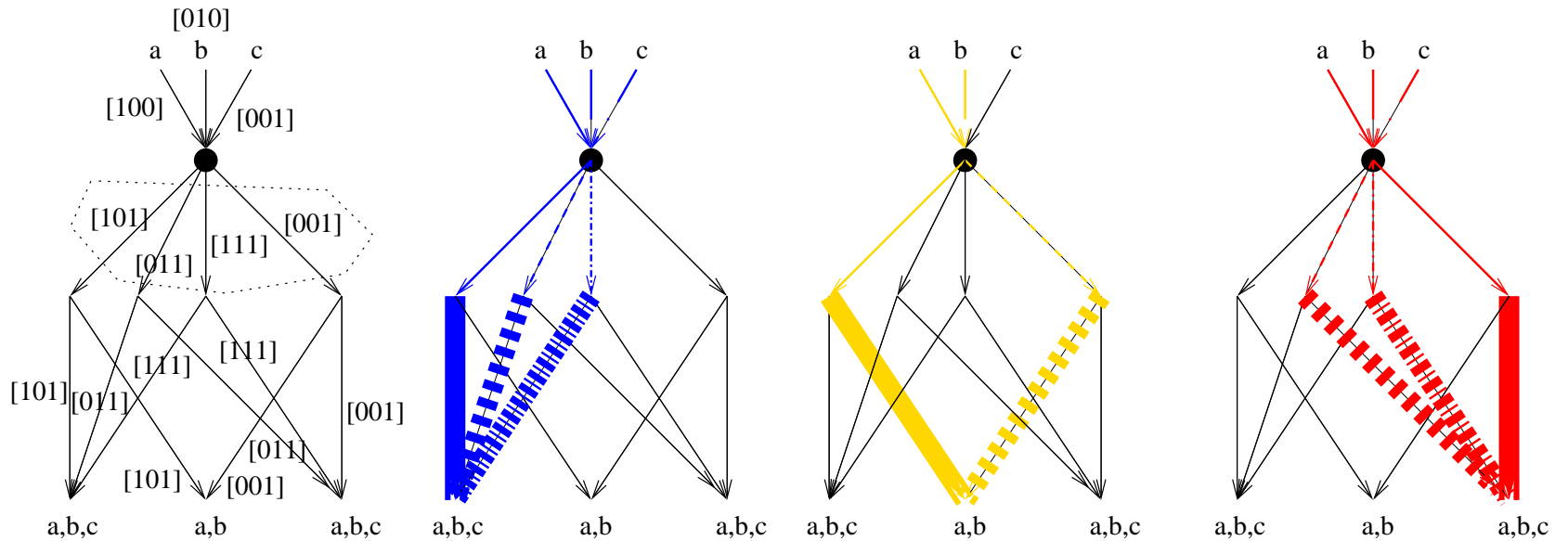


full rank for all frontier sets

Two-Level Multicast



Two-Level Multicast



[101]
[011]
[111]
[001]

———— [101]
- - - - [011]
- · - · [111]

———— [101]
- · - · [001]

———— [001]
- · - · [011]
- · - · 111]

Reencode $[a,b,c]$ as $A[abc]^T$ such that $\begin{bmatrix} [101] \\ [011] \\ [001] \end{bmatrix} A = \begin{bmatrix} [100] \\ [010] \end{bmatrix}$

An analysis of random assignments is done in the next session.

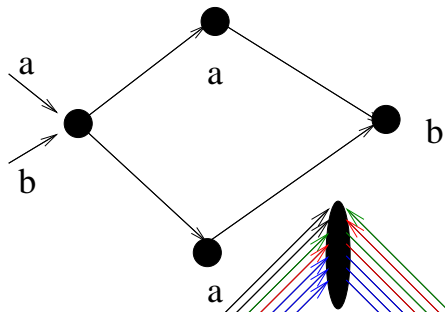
The flow based algorithm is inherently more efficient than a pure random assignment.

How do we pack flows with as much overlap as possible?

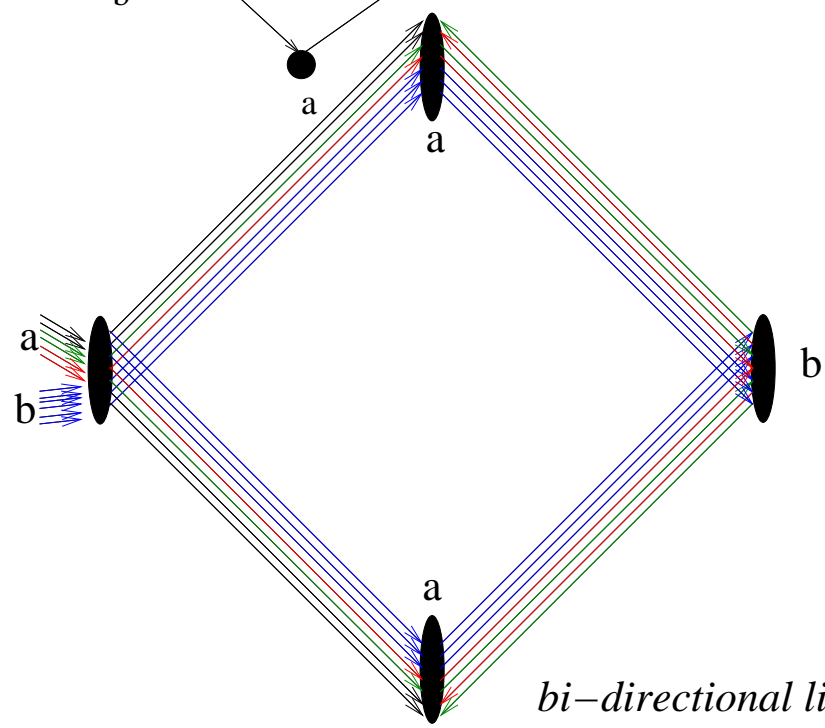
But first: The case of bidirectional links!

(Zongpeng Li, Baochun Li, Dan Jiang, Lap Chi Lau. "On Achieving Optimal End-to-End Throughput in Data Networks: Theoretical and Empirical Studies," Technical Report, University of Toronto, May 2004)

Bidirectional links — A case where network coding does not help



directional links: rate of transmission 0.5 symbols per time unit



bi-directional links: Rate of transmission is bounded by 6/7

Bidirectional links

Steiner Tree Packing for Multicast Problems:

Find the set of all Steiner trees \mathcal{T} , i.e. trees connecting all receivers with a source in a multicast group.

For a link e and $T \in \mathcal{T}$:

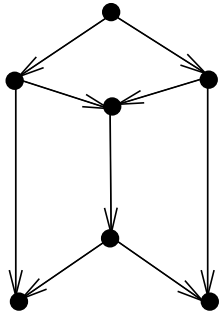
$$I(e, T) = \begin{cases} 1 & e \text{ is part of } T \\ 0 & \text{otherwise} \end{cases}$$

The central problem: Find $\lambda(T) \in \mathbb{R}_+$ maximizing

$$\sum_{T \in \mathcal{T}} \lambda(T) \quad \text{such that} \quad \sum_{T \in \mathcal{T}} \lambda(T) I(e, T) \leq C(e)$$

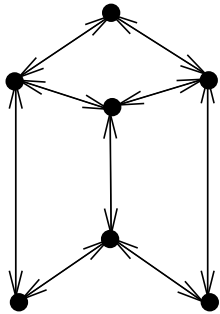
for all links e .

Bidirectional links



Without network coding: One symbol per time unit

With network coding: Two symbols per time unit

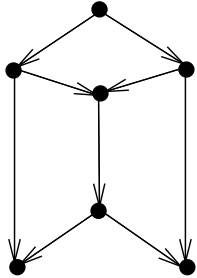


Without network coding: ?

With network coding: ?

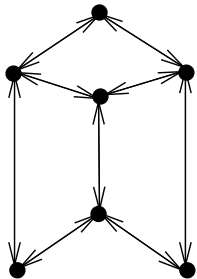
A case where network coding does help (even though it's not much)

Bidirectional links



Without network coding: One symbol per time unit

With network coding: Two symbols per time unit

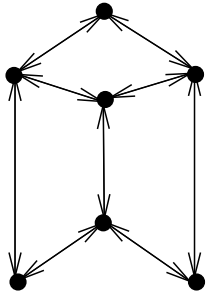


Without network coding: ?

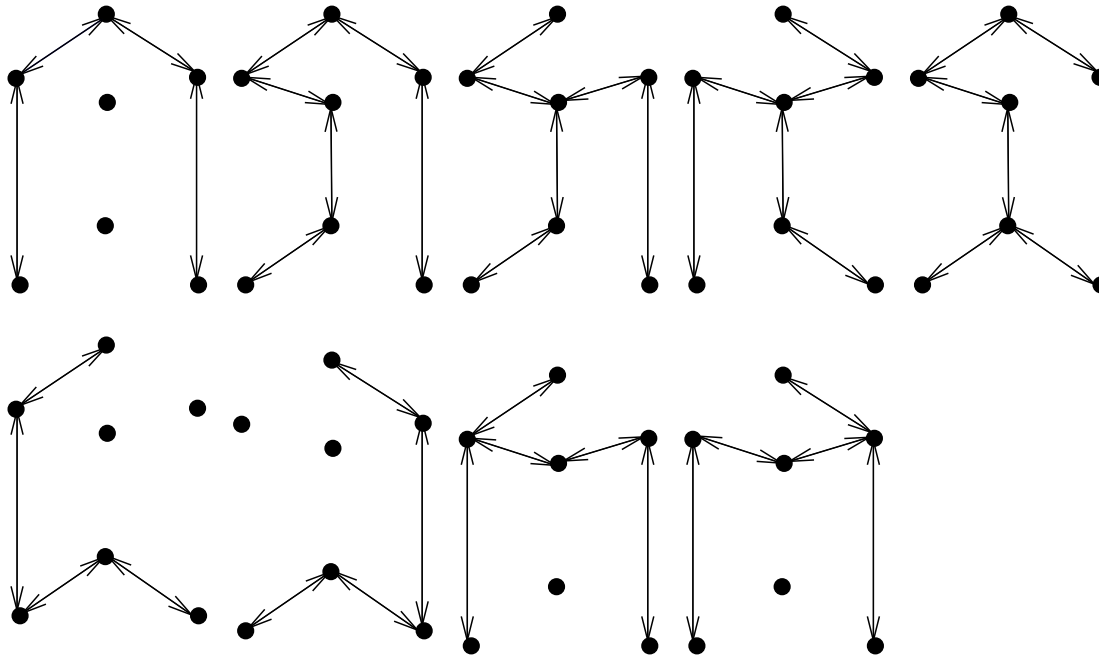
With network coding: Two symbols per time unit (min cut)

A case where network coding does help (even though it's not much)

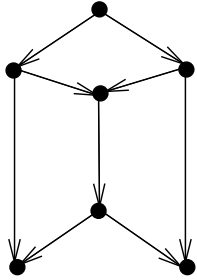
Bidirectional links



Packing the below trees yields
a rate of 1.5 symbols per time unit
(1.875 optimal [Li,Li,Lau])

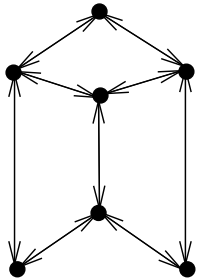


Bidirectional links



Without network coding: One symbol per time unit

With network coding: Two symbols per time unit



Without network coding: 1.875 symbols per time unit

With network coding: Two symbols per time unit (min cut)

A case where network coding does help (even though it's not much)

Bidirectional links

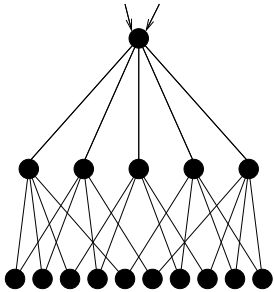
[Li,Li,Lau] The ratio between the multicast rates achievable with or without network coding in bidirectional networks is bounded by a factor of two.

Bidirectional links

[Li,Li,Lau] The ratio between the multicast rates achievable with or without network coding in bidirectional networks is bounded by a factor of two.

(The point of network coding here is really complexity!)

Bidirectional links - An Example



With network coding we achieve a capacity of 2 symbols per time unit

Without network coding we achieve a throughput of 1.786 symbols per unit time

This comes at a cost of optimizing over 119104 Steiner trees [Li,Li,Lau]

Bidirectional links

The crucial step in a network coding solution for the multicast problem in bidirectional links is to find the best (bidirectional) flows corresponding to each receiver. To this end we formulate a linear program:

Each link e carries two flows (direction $+$ and $-$) $f_+^{(\ell)}(e)$ and $f_-^{(\ell)}(e)$ due to receiver ℓ .

Maximize: f

Constraints for all ℓ

$$f_+^{(\ell)}(e) + f_-^{(\ell)}(e) \leq c(e)$$

$$\sum_{f^{(\ell)} \text{ flowing into receiver } \ell} f^{(\ell)} = f$$

$$\sum_{f^{(\ell)} \text{ flowing out of the source}} f^{(\ell)} = f$$

$$\sum_{f^{(\ell)} \text{ flowing into node } i} f^{(\ell)} = \sum_{f^{(\ell)} \text{ flowing out of node } i} f^{(\ell)}$$

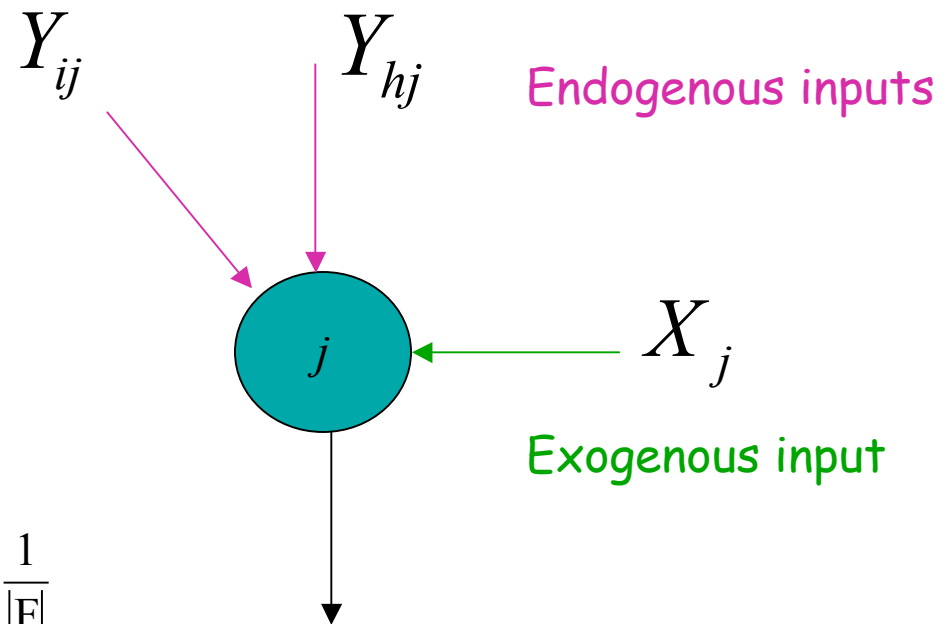
Summary:

- For directed networks the “coding gain” is unbounded
- We “really” need codes
- The necessary multicast fieldsize is bounded as $\sqrt{T} \leq |\mathbb{F}| \leq T$
- Two basic methods to find solutions: algebraic and recursively assigning edges
- A natural method: “random assignment” (more about this shortly)

- For bidirectional link the coding gain is bounded by 2
- The main advantage of network coding in complexity.
- More about linear programs shortly!

Randomized network coding

- The effect of the network is that of a transfer matrix from sources to receivers
- To recover symbols at the receivers, we require sufficient degrees of freedom - an invertible matrix in the coefficients of all nodes
- The realization of the determinant of the matrix will be non-zero with high probability if the coefficients are chosen independently and randomly
- Probability of success over field $F \approx 1 - \frac{1}{|F|}$
- Randomized network coding can use any multicast subgraph which satisfies min-cut max-flow bound for each receiver [HKMKE03, HMSEK03, WCJ03] for any number of sources, even when correlated [HMEK04]



$$Y_{jm} = \alpha_{jm}^{ij} Y_{ij} + \alpha_{jm}^{hj} Y_{hj} + \alpha_{jm}^j X_j$$

Randomized network coding

- [HKMKE03, HMSEK03] For a feasible d -receiver multicast connection problem on a network with
- independent or linearly correlated sources
- a network code in which code coefficients for η links are chosen independently and uniformly over \mathbb{F}_q
- the success probability is at least

$$(1 - d/q)^\eta$$

- Error bound is of the order of the inverse of the field size, so error probability decreases exponentially with codeword length

Proof outline

- Recall transfer matrix for each receiver must be non-singular
- We show an equivalent condition connected with bipartite matching: the Edmonds matrices

singular

$$\begin{bmatrix} A & 0 \\ I - F & B_{\beta}^T \end{bmatrix}$$

$$\begin{bmatrix} A & 0 \\ I - DF & B_{\beta}^T \end{bmatrix}$$

Proof outline

- This shows that if η links have random coefficients, the determinant polynomial has maximum degree η in the random variables and is linear in each of these variables
- We want the product of the d receivers' determinant polynomials to be nonzero
- We can show inductively, using the Schwartz-Zippel Theorem, that for any polynomial

$$P \in \mathbb{F}[\xi_1, \xi_2, \dots]$$

of degree $\leq d\eta$

- in which each ξ_i has exponent at most d if ξ_1, ξ_2, \dots are chosen independently and uniformly at random from $\mathbb{F}_q \subseteq \mathbb{F}$ then the polynomial evaluates to 0 with probability at most $1 - (1 - d/q)^\eta$
- Particular form of the determinant polynomials gives rise to a tighter bound than the Schwartz-Zippel bound for general polynomials of the same total degree

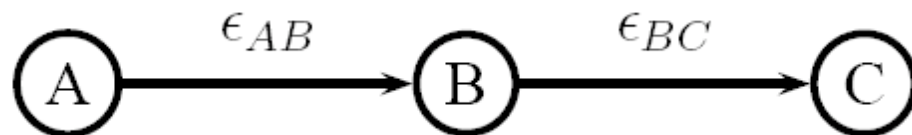
Erasure reliability

- Packet losses in networks result from
 - congestion,
 - buffer overflows,
 - (in wireless) outage due to fading or change in topology
- Prevailing approach for reliability: Request retransmission
- Not suitable for
 - high-loss environments,
 - multicast,
 - real-time applications.

Erasure reliability

- Alternative approach: Forward Error Correction (FEC)
 - Multiple description codes
 - Erasure-correcting codes (e.g. Reed-Solomon, Tornado, LT, Raptor)
- End-to-end: Connection as a whole is viewed as a single channel; coding is performed only at the source node.

Erasure reliability - single flow



End-to-end erasure coding: Capacity is $(1 - \epsilon_{AB})(1 - \epsilon_{BC})$ packets per unit time.

As two separate channels: Capacity is $\min(1 - \epsilon_{AB}, 1 - \epsilon_{BC})$ packets per unit time.
- Can use block erasure coding on each channel. But delay is a problem.

Network coding: minimum cut is capacity

- For erasures, correlated or not, we can in the multicast case deal with *average* flows uniquely [Lun et al. 04, 05], [Dana et al. 04]:
 - Nodes store received packets in memory
 - Random linear combinations of memory contents sent out
 - Delay expressions generalize Jackson networks to the innovative packets
 - Can be used in a rateless fashion

Erasure reliability

- For erasures, correlated or not, we can in the multicast case deal with *average* flows uniquely [LME04], [LMK05], [DGPHE04]
- We consider a scheme [LME04] where
 - nodes store received packets in memory;
 - random linear combinations of memory contents sent out at every transmission opportunity (without waiting for full block).
- Scheme gets to capacity under arbitrary coding at every node for
 - unicast and multicast connections
 - networks with point-to-point and broadcast links.

Scheme for erasure reliability

- We have k message packets w_1, w_2, \dots, w_k (fixed-length vectors over F_q) at the source.
- (Uniformly-)random linear combinations of w_1, w_2, \dots, w_k injected into source's memory according process with rate R_0 .
- At every node, (uniformly-)random linear combinations of memory contents sent out;
 - received packets stored into memory.
 - in every packet, store length- k vector over F_q representing the transformation it is of w_1, w_2, \dots, w_k — *global encoding vector*.

Coding scheme

- Since all coding is linear, can write any packet x as a linear combination

$$x = \sum_{m=1}^k \gamma_m w_m.$$

of w_1, w_2, \dots, w_k :

- The vector γ is the *global encoding vector* of x .
- We send the *global encoding vector* along with x , in its header, incurring a constant overhead.
- The side information provided by γ is very important to the functioning of the scheme.

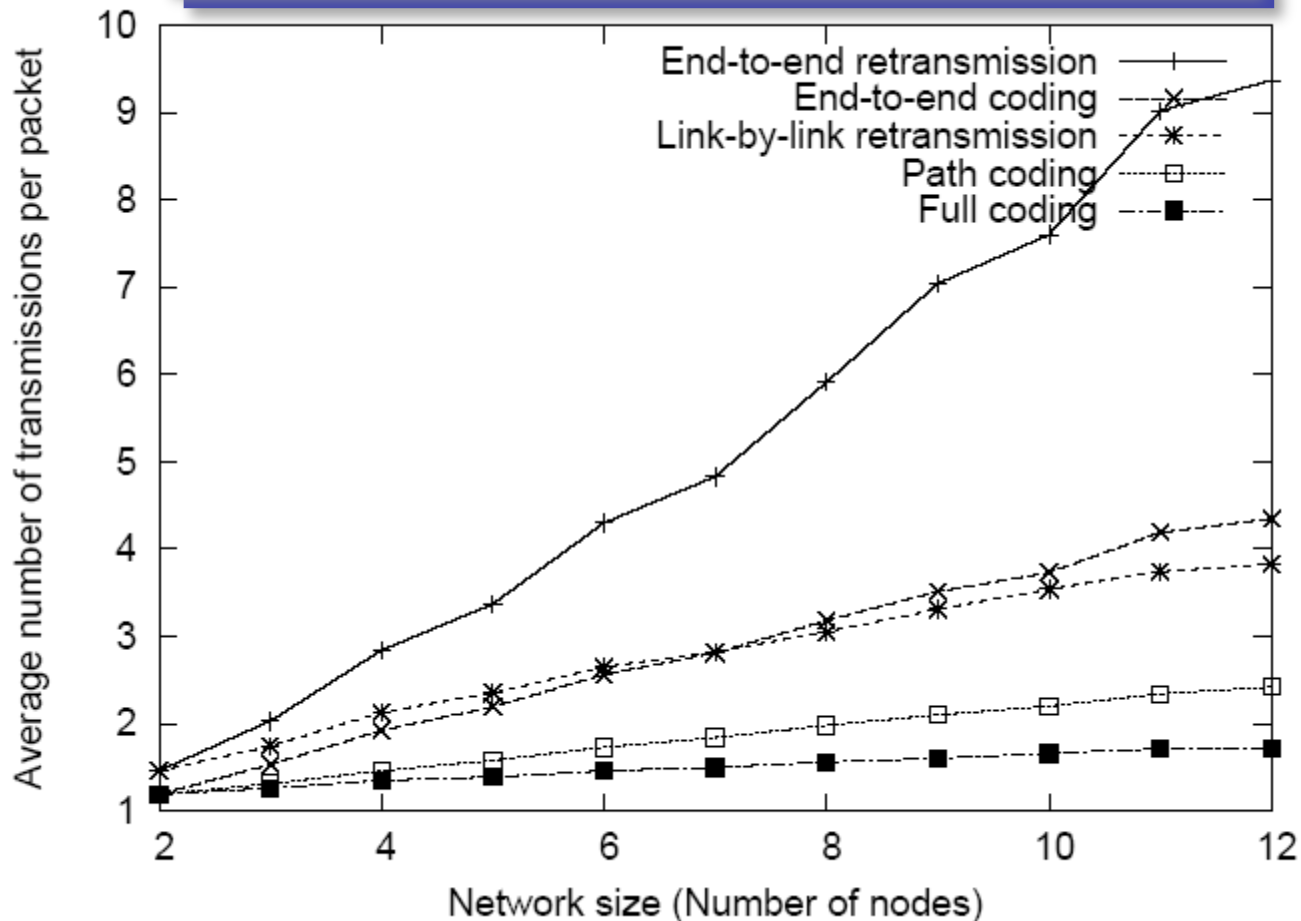
Outline of proof

- Keep track of the propagation of *innovative packets* - packets whose *auxiliary encoding vectors* (transformation with respect to the n packets injected into the source's memory) are linearly independent across particular cuts.
- Can show that, if R_0 less than capacity and input process is Poisson, then propagation of innovative packets through any node forms a stable $M/M/1$ queueing system in steady-state.
- So, N_i , the number of innovative packets in the network is a time-invariant random variable with finite mean.
- We obtain delay expressions using in effect a generalization of Jackson networks for the innovative packets

Comments for erasure reliability

- Particularly suitable for
 - overlay networks using UDP, and
 - wireless packet networks (have erasures and can perform coding at all nodes).
- Code construction is completely decentralized.
- Scheme can be operated *ratelessly* - can be run indefinitely until successful reception.

Coding for packet losses - unicast



Average number of transmissions required per packet in random networks of varying size. Sources and sinks were chosen randomly according to a uniform distribution. Paths or subgraphs were chosen in each random instance to minimize the total number of transmissions required, except in the cases of end-to-end retransmission and end-to-end coding, where they were chosen to minimize the number of transmissions required by the source node.

Further Results on Coding for Reliable Communication over Packet Networks

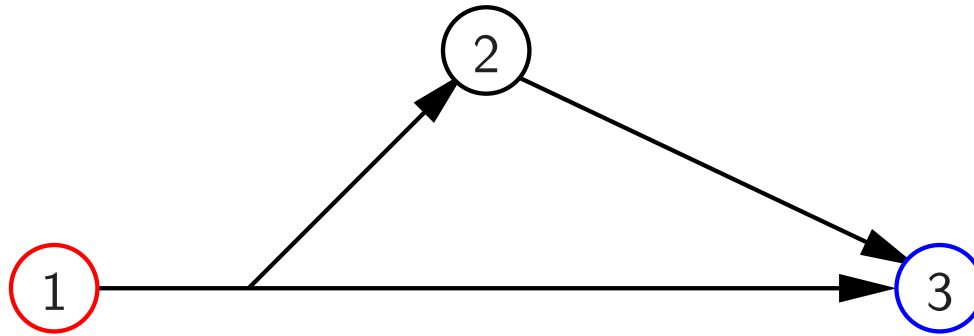
Desmond Lun, MIT

Muriel Médard, MIT
Michelle Effros, CalTech

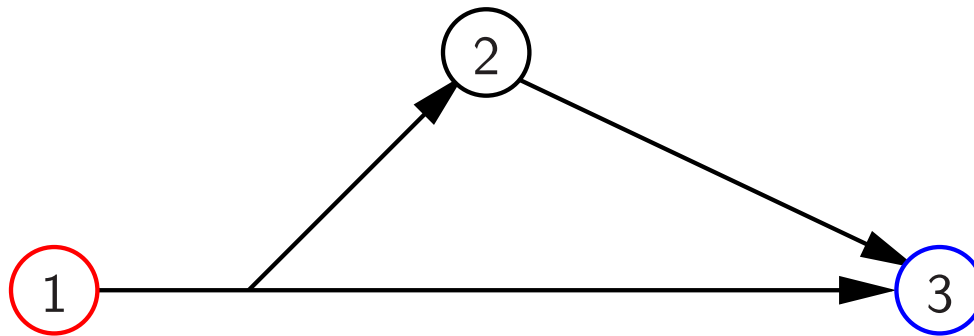
Ralf Koetter, UIUC

9 September 2005

An example: Slotted Aloha wireless network

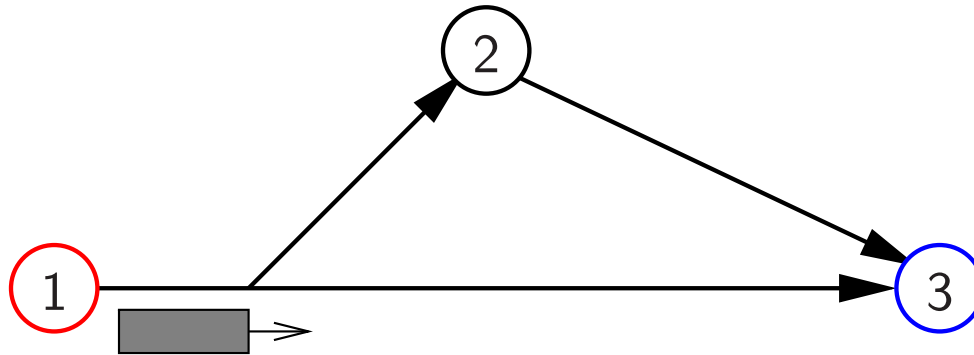


An example: Slotted Aloha wireless network



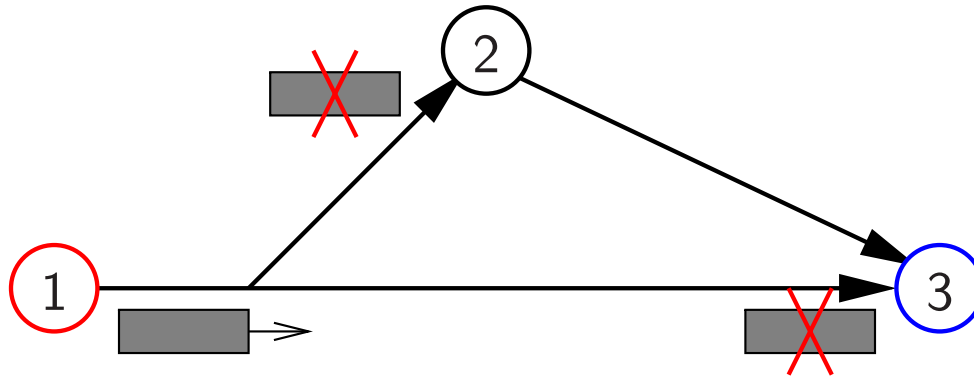
- Prob. q_0 : neither 1 nor 2 transmits

An example: Slotted Aloha wireless network



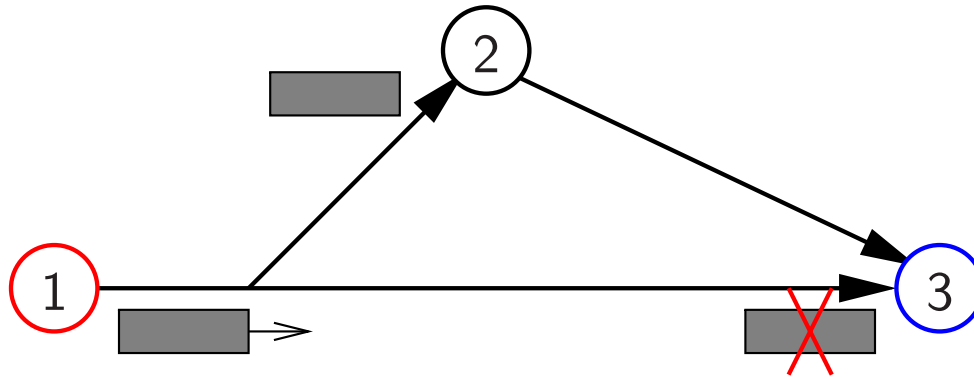
- Prob. q_1 : 1 transmits, 2 does not

An example: Slotted Aloha wireless network



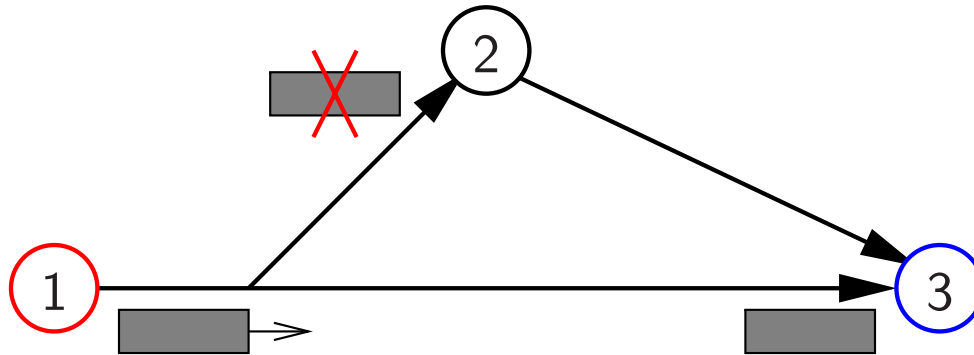
- Prob. q_1 : 1 transmits, 2 does not
 - Prob. p_{10} : packet is received by neither 2 nor 3

An example: Slotted Aloha wireless network



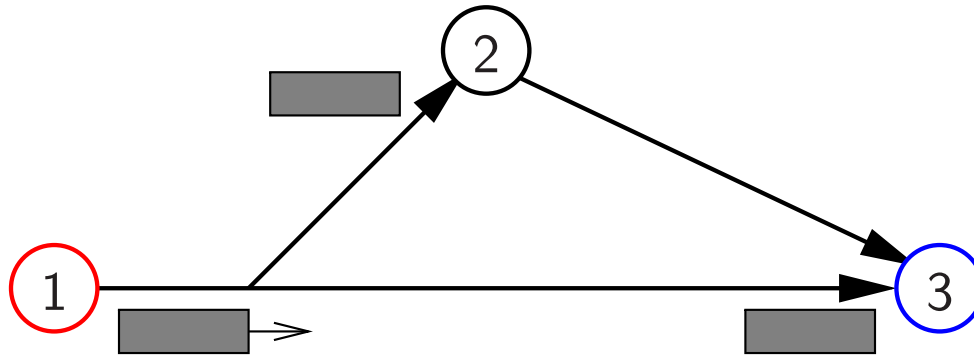
- Prob. q_1 : 1 transmits, 2 does not
 - Prob. p_{12} : packet is received by 2 but not 3

An example: Slotted Aloha wireless network



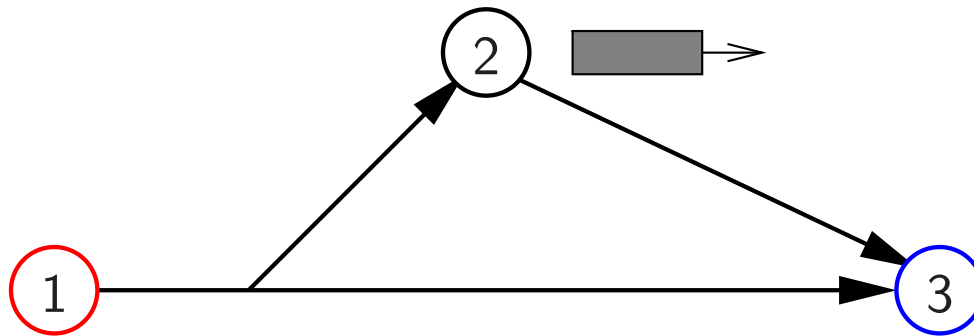
- Prob. q_1 : 1 transmits, 2 does not
 - Prob. p_{13} : packet is received by 3 but not 2

An example: Slotted Aloha wireless network



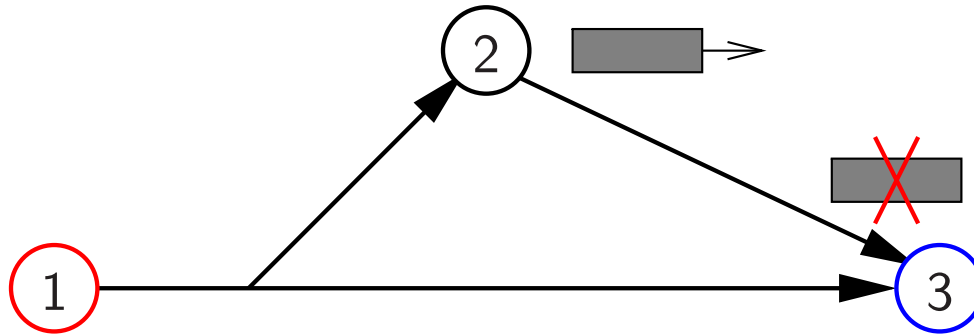
- Prob. q_1 : 1 transmits, 2 does not
 - Prob. p_{123} : packet is received by 2 and 3

An example: Slotted Aloha wireless network



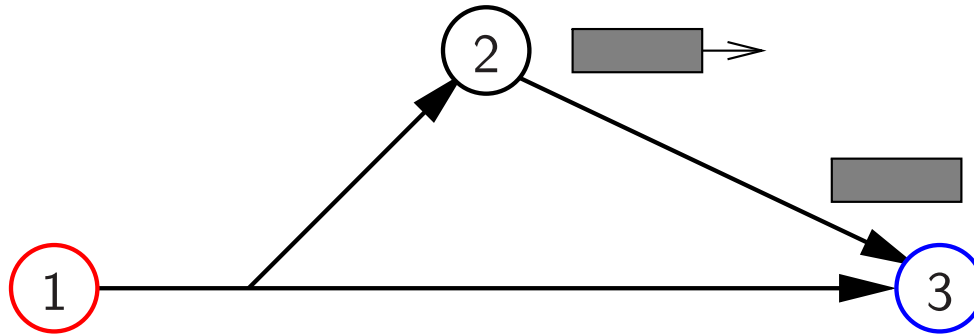
- Prob. q_2 : 2 transmits, 1 does not

An example: Slotted Aloha wireless network



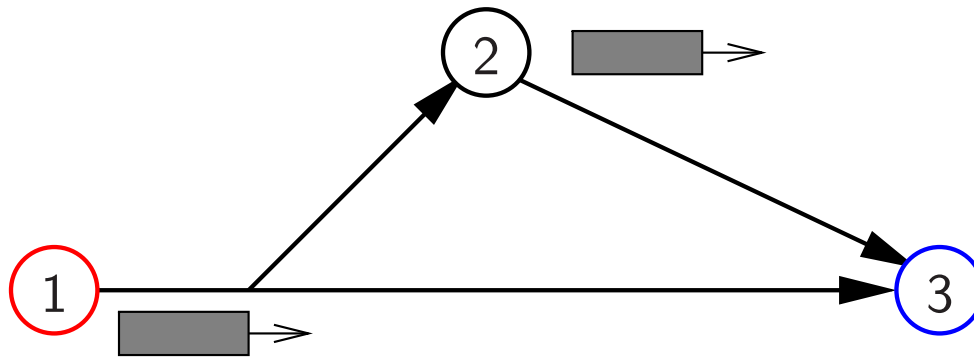
- Prob. q_2 : 2 transmits, 1 does not
 - Prob. p_{20} : packet is not received by 3

An example: Slotted Aloha wireless network



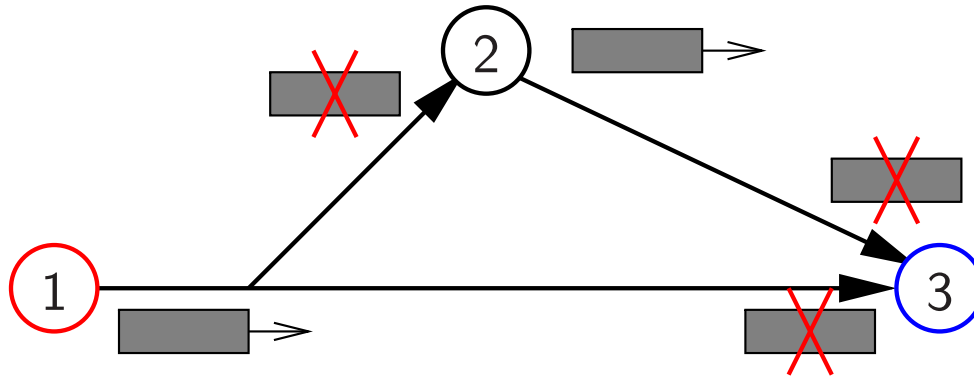
- Prob. q_2 : 2 transmits, 1 does not
 - Prob. p_{20} : packet is received by 3

An example: Slotted Aloha wireless network



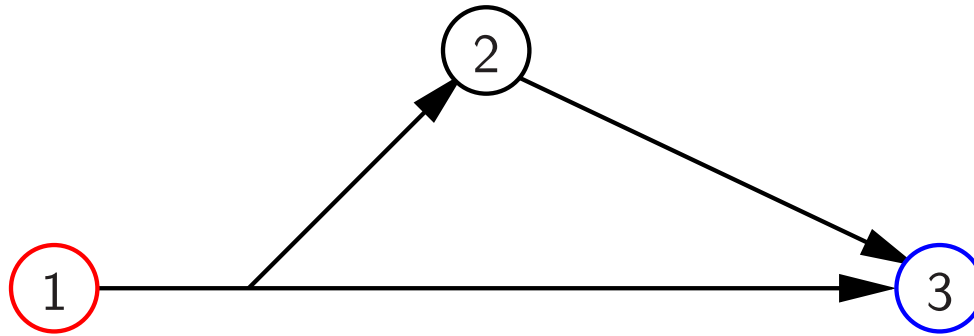
- Prob. q_{12} : both 1 and 2 transmit

An example: Slotted Aloha wireless network



- Prob. q_{12} : both 1 and 2 transmit \rightarrow collision
- Behavior of links is correlated

An example: Slotted Aloha wireless network



- What is the capacity?
- How is it achieved?

Results

- We consider the following operation:
 - nodes store received packets in memory
 - send random linear combinations of memory at every opportunity
- Scheme is capacity achieving for
 - unicast or multicast connections
 - lossy wireline or wireless packet networks
- Traffic model is very general
- Error exponents for Poisson traffic with i.i.d. losses

Coding over packets

1. Code only for resilience against erasures, not for errors
2. Side-information can be included in headers
3. Transmissions often not synchronized

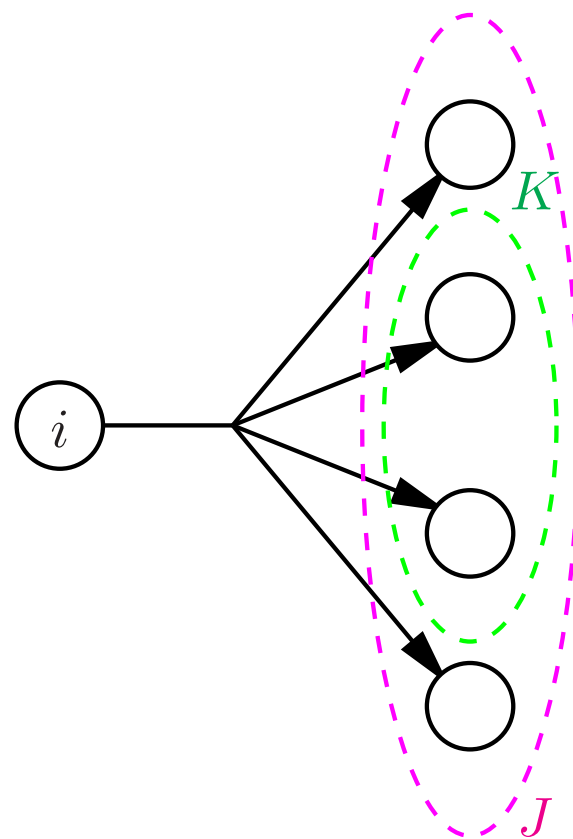
Model: Wireline networks

- Directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$
- Arc (i, j) : lossy point-to-point link
 - packets injected at i according to some process
 - packets received at j with average rate z_{ij}
- Meaning of average rate:
 - $A_{ij}(\tau)$: number of packets received between time 0 and time τ on arc (i, j)

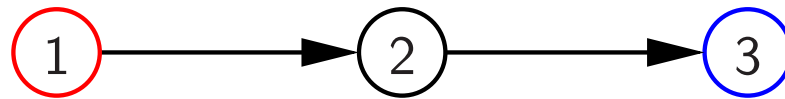
$$\lim_{\tau \rightarrow \infty} \frac{A_{ij}(\tau)}{\tau} = z_{ij} \quad \text{a.s.}$$

Model: Wireless networks

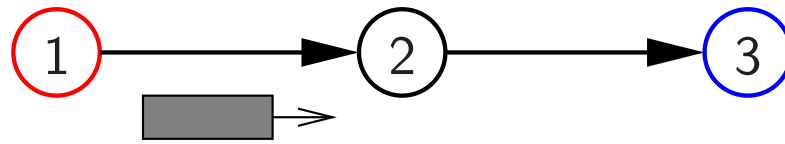
- Directed hypergraph $\mathcal{H} = (\mathcal{N}, \mathcal{A})$
- Hyperarc (i, J) : lossy broadcast link
 - packets injected at i according to some process
 - packets received by set of nodes $K \subset J$ with average rate z_{iJK}



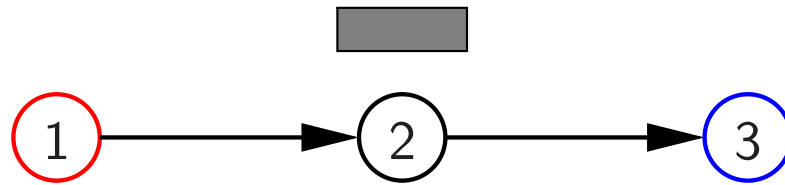
Idea of proof



Idea of proof

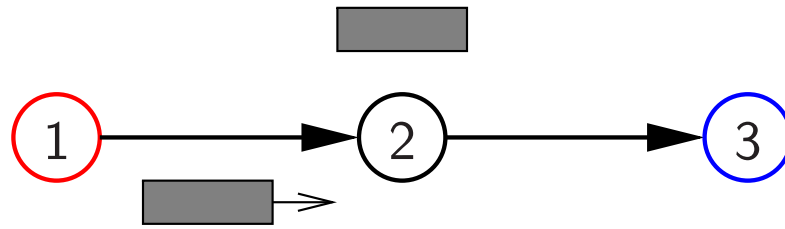


Idea of proof



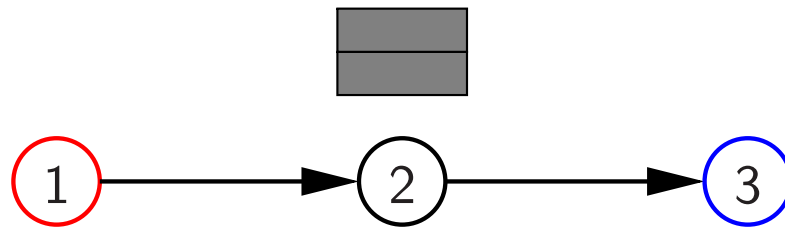
- Node 2: x_1

Idea of proof



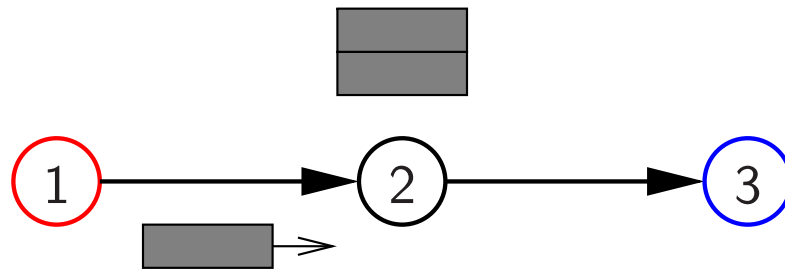
- Node 2: x_1

Idea of proof



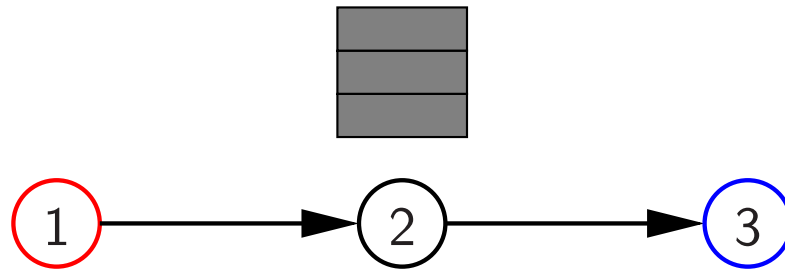
- Node 2: x_1, x_2

Idea of proof



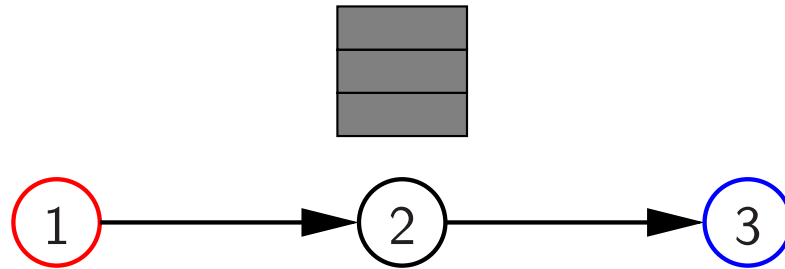
- Node 2: x_1, x_2

Idea of proof



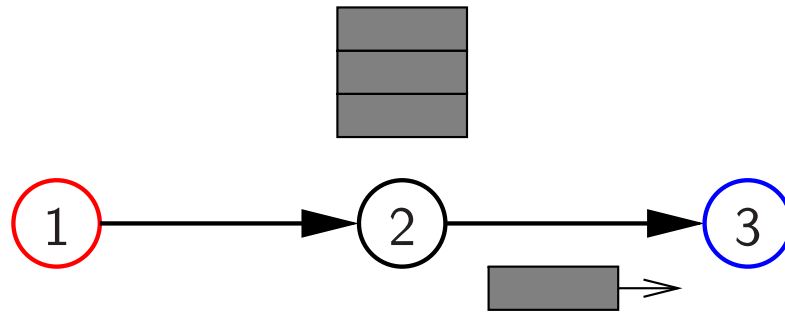
- Node 2: x_1, x_2, x_3

Idea of proof



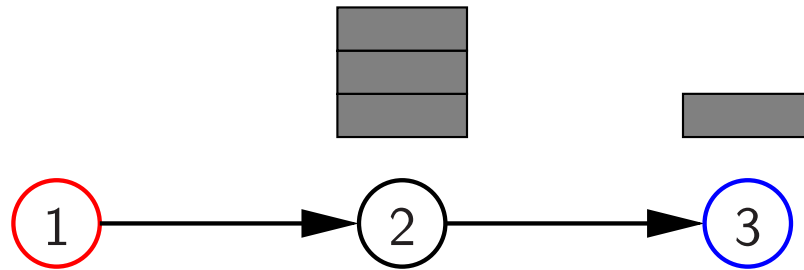
- Node 2: $x_1 \rightarrow (1, 0, 0)$, $x_2 \rightarrow (0, 1, 0)$, $x_3 \rightarrow (0, 0, 1)$

Idea of proof



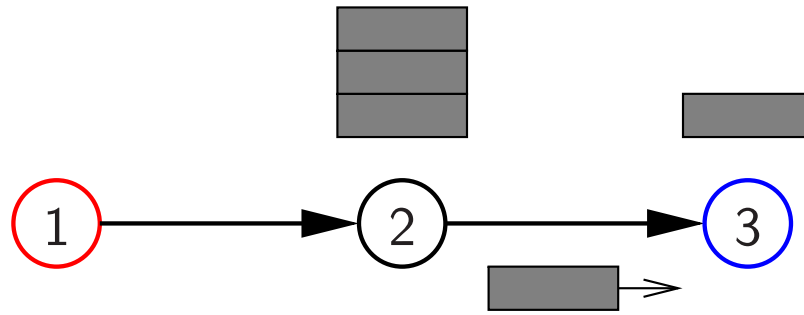
- Node 2: $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$

Idea of proof



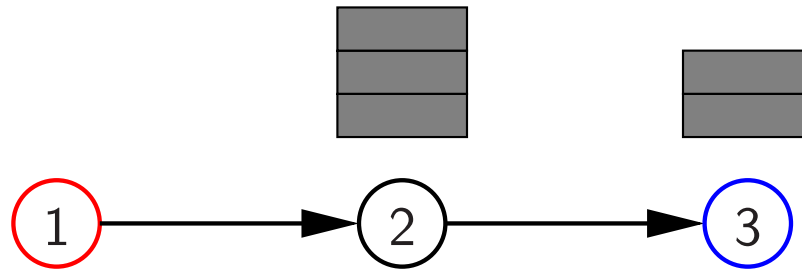
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13})$

Idea of proof



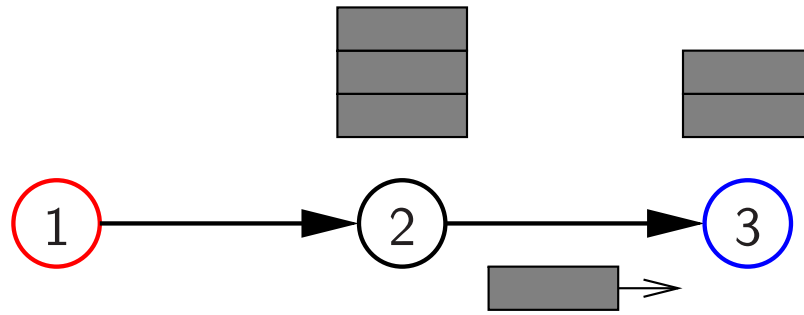
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13})$

Idea of proof



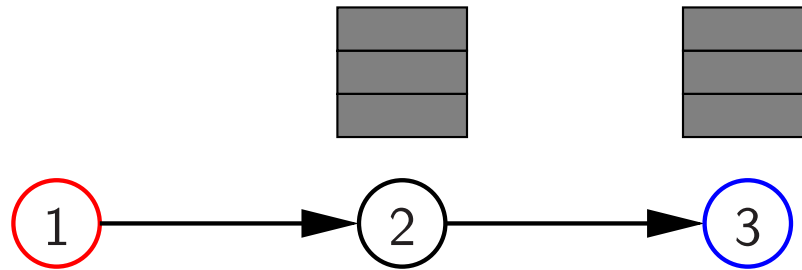
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23})$

Idea of proof



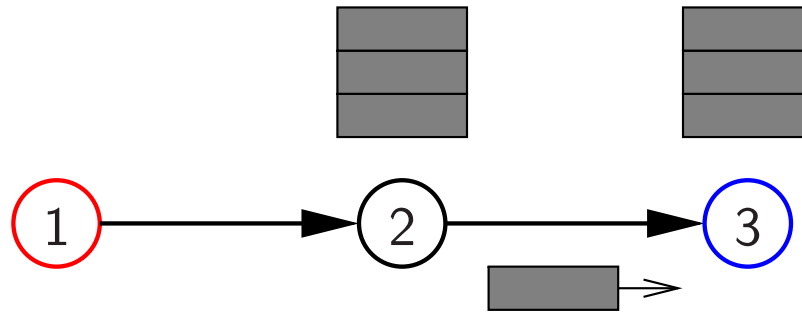
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23})$

Idea of proof



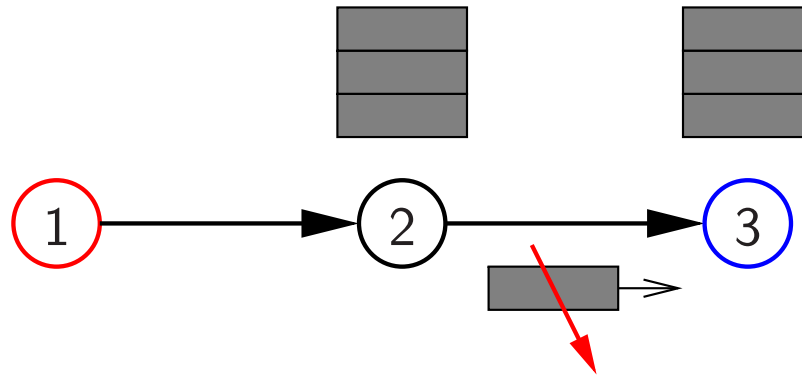
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23}), (\alpha_{31}, \alpha_{32}, \alpha_{33})$

Idea of proof



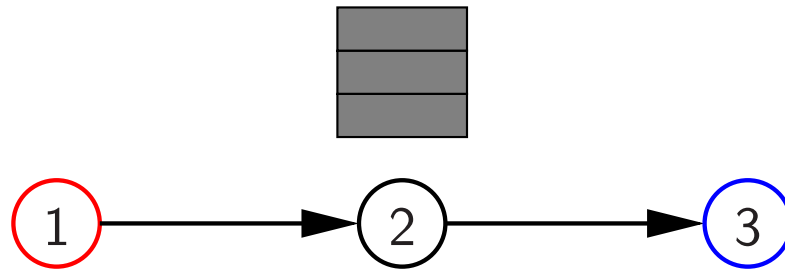
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23}), (\alpha_{31}, \alpha_{32}, \alpha_{33})$

Idea of proof



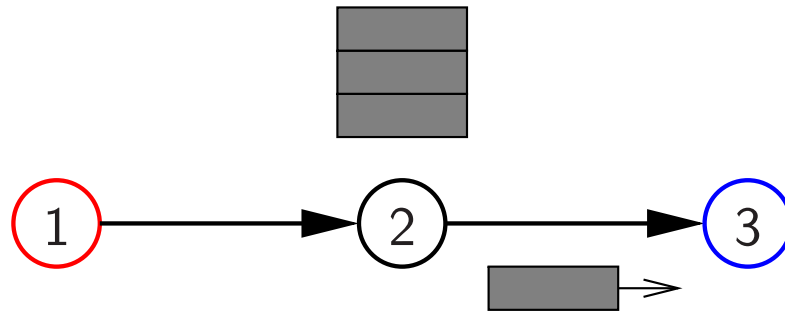
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23}), (\alpha_{31}, \alpha_{32}, \alpha_{33})$

Idea of proof



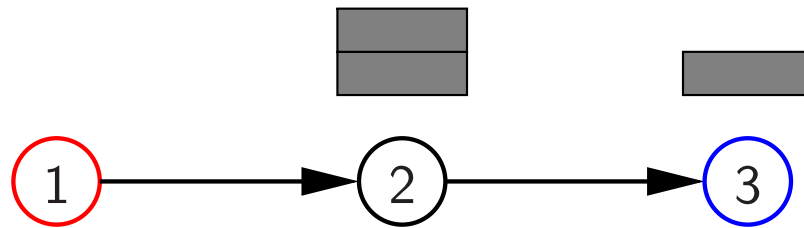
- Node 2: $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$

Idea of proof



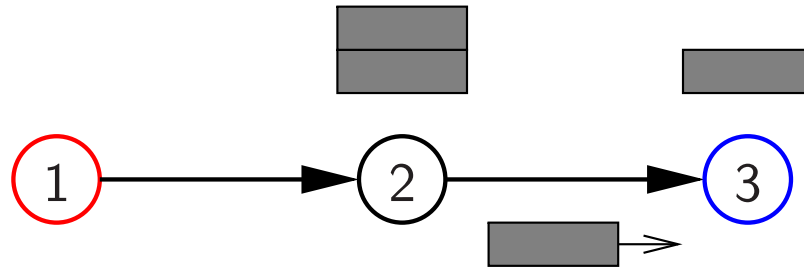
- Node 2: $(1, 0, 0)$, $(0, 1, 0)$, $(0, 0, 1)$

Idea of proof



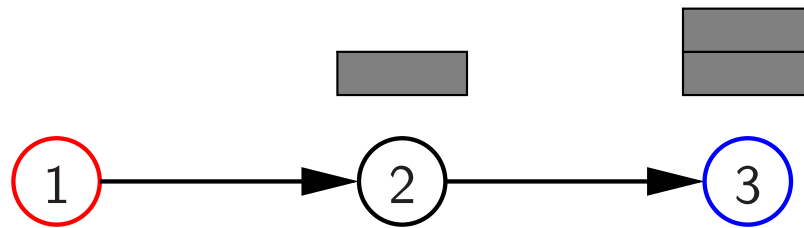
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13})$

Idea of proof



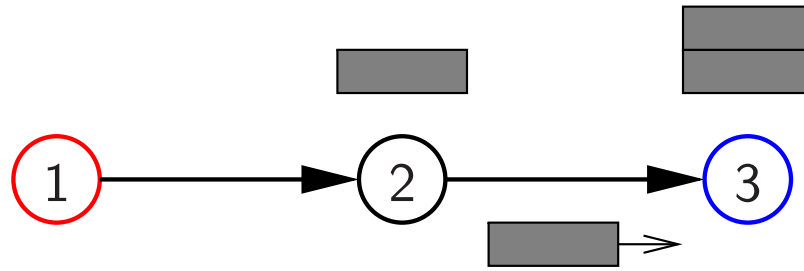
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13})$

Idea of proof



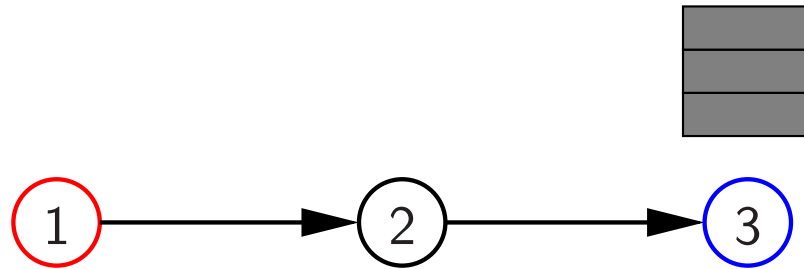
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23})$

Idea of proof



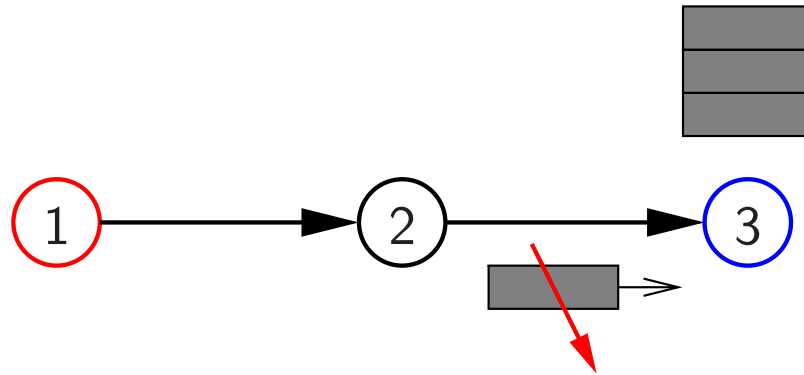
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23})$

Idea of proof



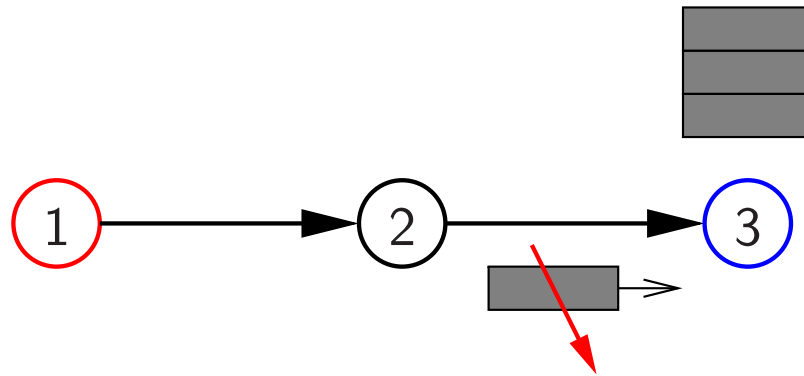
- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23}), (\alpha_{31}, \alpha_{32}, \alpha_{33})$

Idea of proof



- Node 2: $(1, 0, 0), (0, 1, 0), (0, 0, 1)$
- Node 3: $(\alpha_{11}, \alpha_{12}, \alpha_{13}), (\alpha_{21}, \alpha_{22}, \alpha_{23}), (\alpha_{31}, \alpha_{32}, \alpha_{33})$

Idea of proof

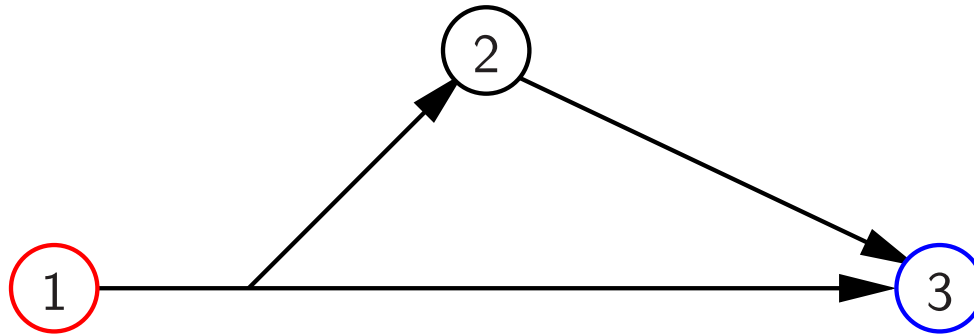


- Node 2 behaves like a queue
 - If $(1, 2)$ faster than $(2, 3)$, queue unstable, $(2, 3)$ is bottleneck
 - If $(2, 3)$ faster than $(1, 2)$, queue stable, $(1, 2)$ is bottleneck

Idea of proof

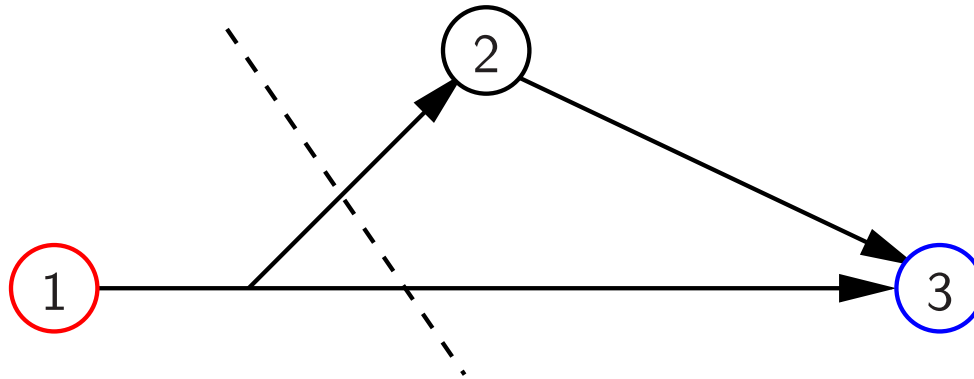
- Two links \rightarrow one path of arbitrarily many links
- One path \rightarrow flow (superposition of paths)
- Flow (wireline networks) \rightarrow hypergraph flow (wireless networks)
- Single flow (unicast) \rightarrow superposition, or union, of flows (multicast)
- Some points:
 - Packet transformation must be described to decoder (side information)
 - Decoding equates to matrix inversion
 - Rateless operation is possible

An example: Slotted Aloha wireless network



- What is the capacity?
- How is it achieved?

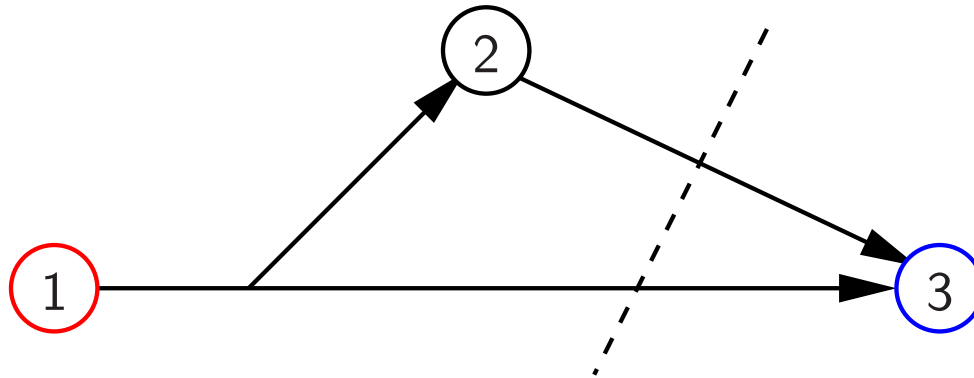
An example: Slotted Aloha wireless network



- Rate of packets over cut:

$$q_1(p_{12} + p_{13} + p_{123})$$

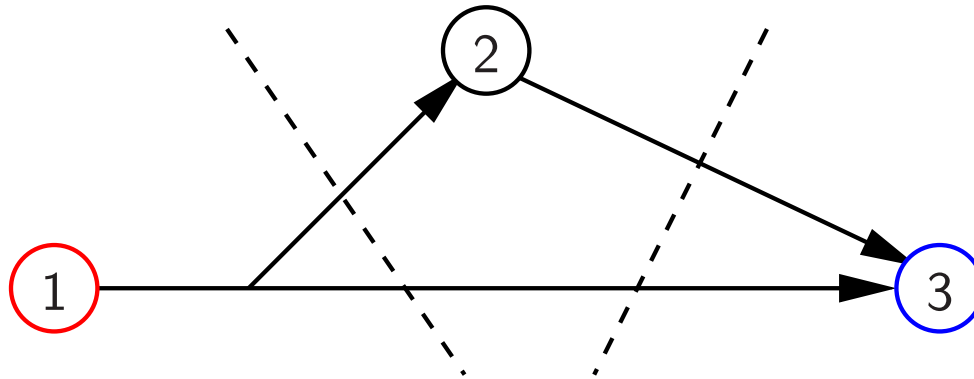
An example: Slotted Aloha wireless network



- Rate of packets over cut:

$$q_1(p_{13} + p_{123}) + q_2p_{23}$$

An example: Slotted Aloha wireless network



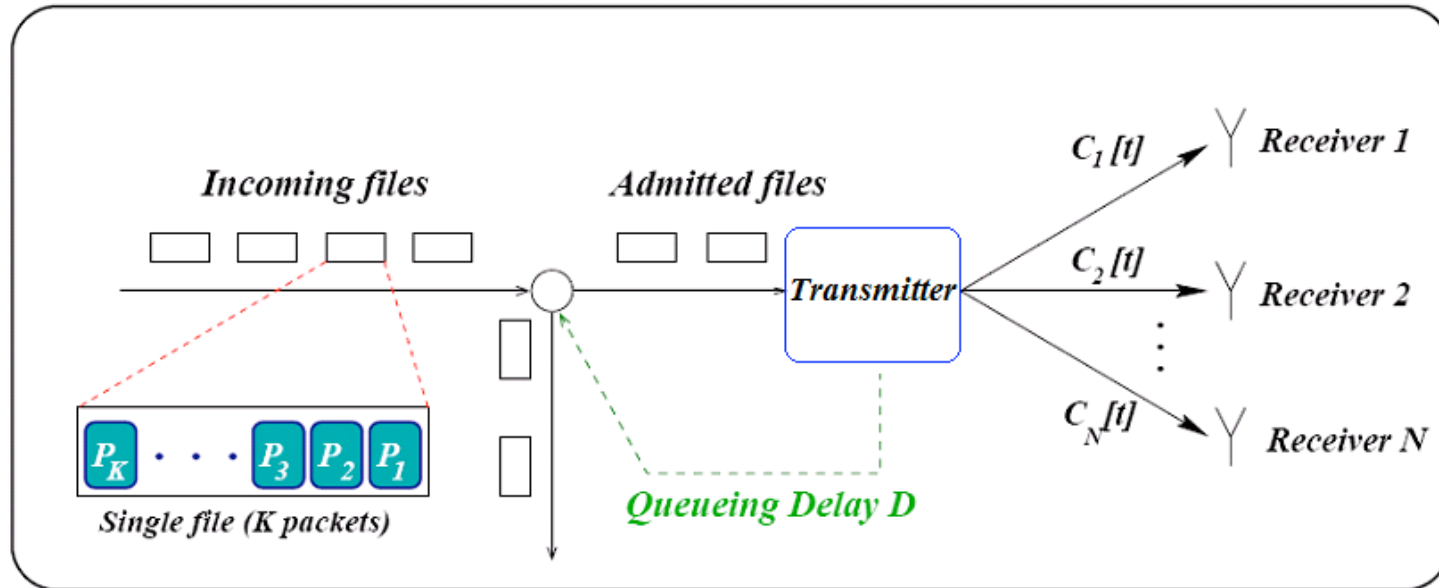
- Our scheme achieves rates less than minimum cut:

$$R < C = \min(q_1(p_{12} + p_{13} + p_{123}), q_1(p_{13} + p_{123}) + q_2p_{23})$$

Conclusion

- Considered simple random linear coding scheme
- Showed rate optimality under very general conditions
- Derived error exponents for Poisson traffic with i.i.d. losses
- Main challenge: Improve on simple scheme
 - memory requirements
 - decoding complexity
 - side-information overhead

File downloads



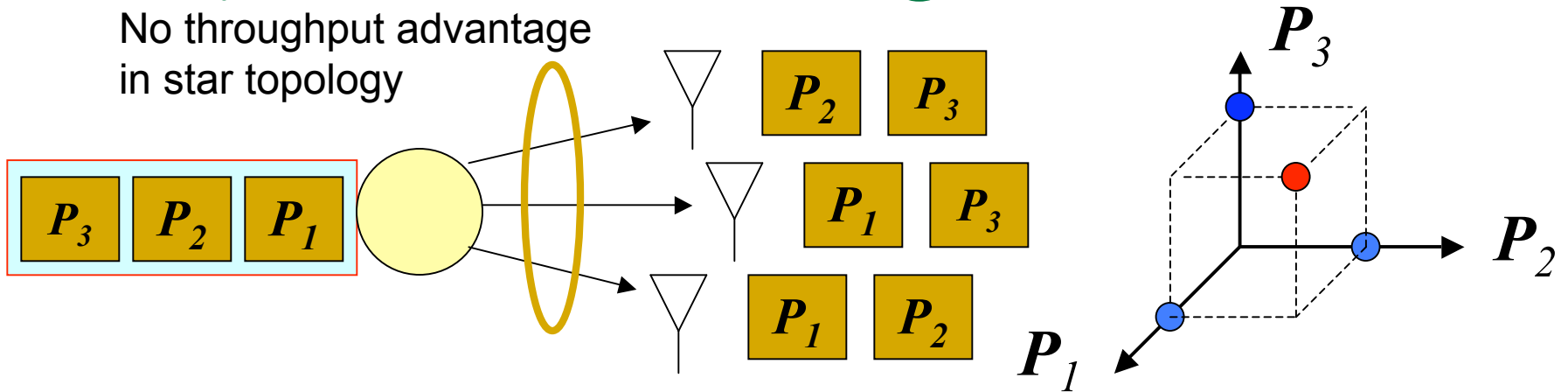
- **Files (Users)** arrive according to a **Poisson process** with rate γ
- Each **file** contains **K packets** to be broadcast to all receivers
- A single packet can be transmitted in one **time slot**
- Users are admitted/rejected based on their delay constraints.
- $C_i[t] \in \{0,1\}$ are i.i.d. **Bernoulli(c)** distributed, and **unknown**

File download

- Files are broadcast in a FIFO fashion:
 - Transmission of the next file starts only after the transmission of the current file is complete
 - Each receiver sends an ACK once it can reconstruct all the packets in the file
 - **Elastic Traffic**: Users have no delay constraint, hence every incoming user is admitted for service
 - **Inelastic Traffic**: Each user has a delay constraint associated with it and is admitted only if the mean waiting time is lower than its constraint
 - **Random Network Coding (RNC)** is the optimal coding strategy
 - **Round Robin (RR) Scheduling** is the optimal scheduling strategy.
-

Delay issues and coding

No throughput advantage
in star topology



$t = 1$	$t = 2$	$t = 3$	$t = 4$
X	P_2	P_3	$P_1 + P_2 + P_3$
P_1	X	P_3	$P_1 + P_2 + P_3$
P_1	P_2	X	$P_1 + P_2 + P_3$

Delays can be improved in cases, such as star networks,
where throughput is not improved

[Eryilmaz et al. 06]

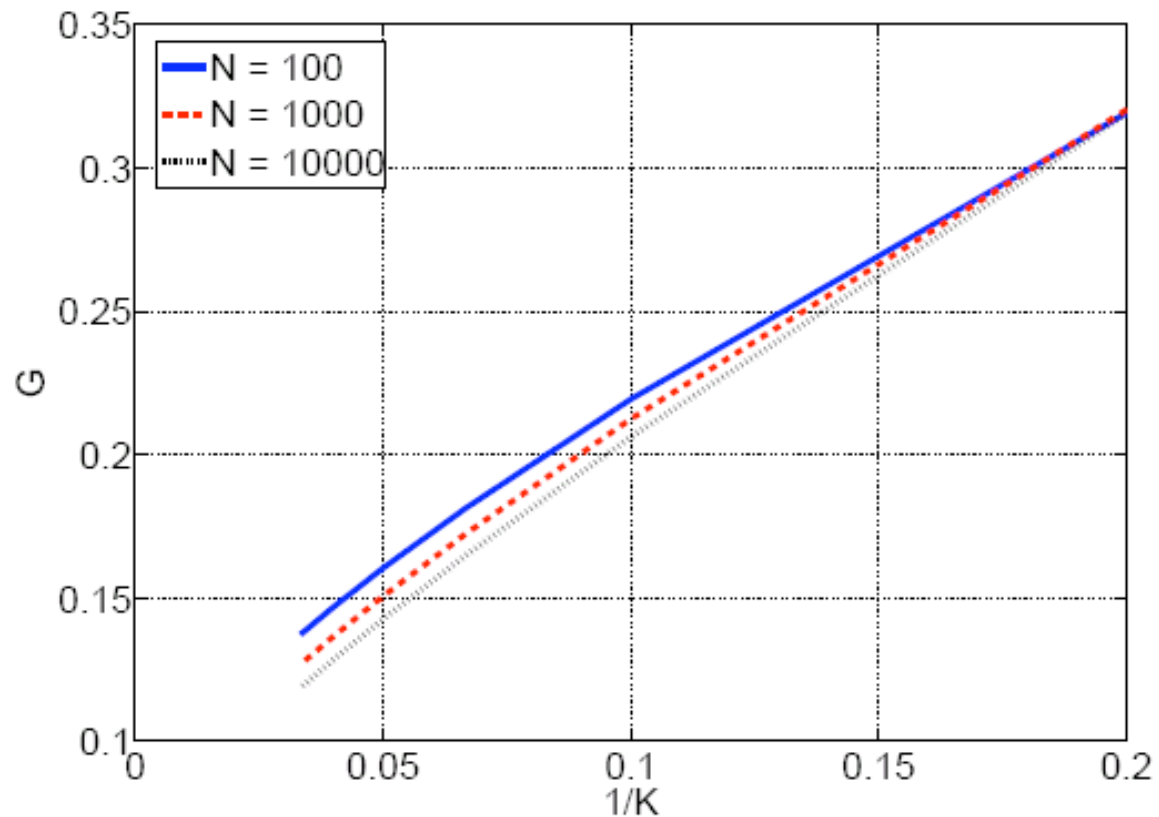
Results – Elastic Traffic

- $m_I^{RNC, RR}(N, K)$ = Mean service time under RNC and RR policies
- $G(N, K) = m_I^{RNC}/m_I^{RR}$ is the relative gain of using RNC as opposed to RR
- We show [Ahmed et al. 07] that

$$\lim_{N \rightarrow \infty} G(N, K) = \frac{1}{K}$$

- There is no scaling gain in N
 - The relative gain of RNC can be made arbitrarily large for a dense network by setting the file size K large enough.
-

Numerical Results – Elastic Traffic



- The numerical computations confirm the predictions of the asymptotic result.

Results – Inelastic Traffic

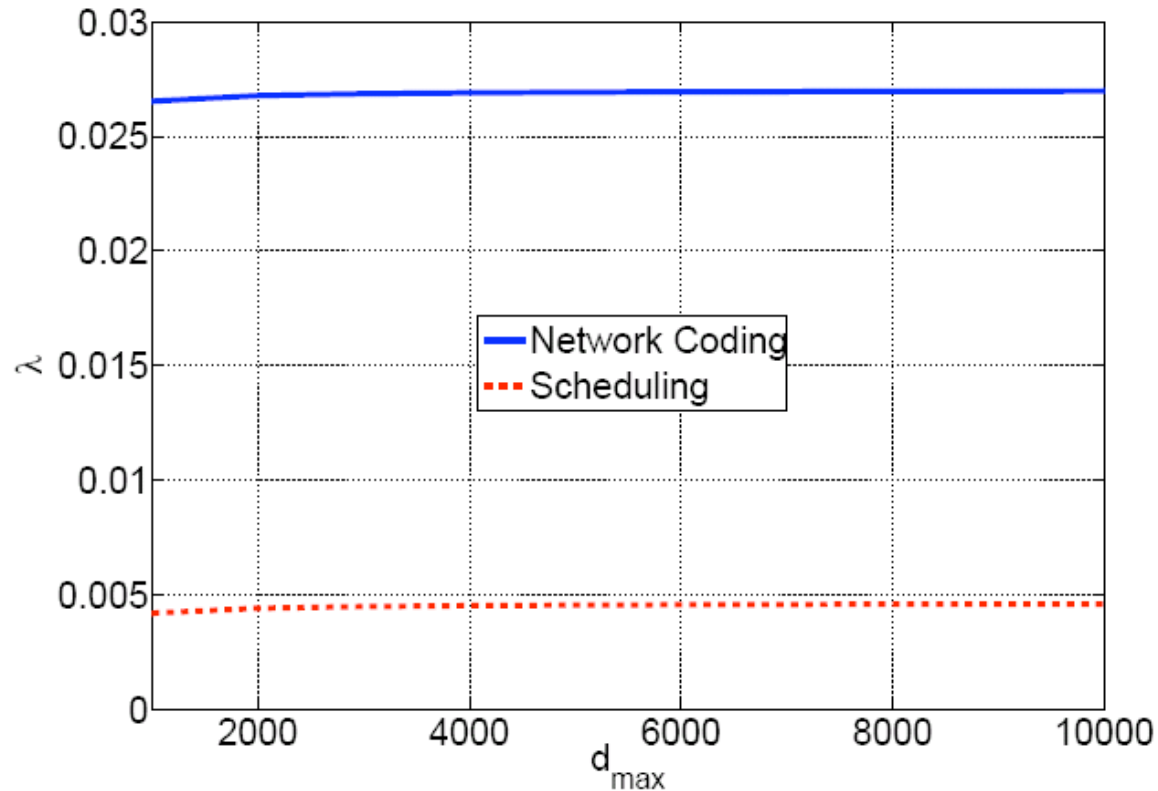
- Scaling delay with respect to N is no longer the right measure
- System becomes an M/G/1 system, where the service time distribution depends on the transmission strategy being employed
- For an admitted traffic rate of λ , mean waiting time is given by

$$E[D] = \frac{\lambda m_2}{2(1 - \lambda m_1)},$$

where m_i is the i^{th} moment of the service time distribution

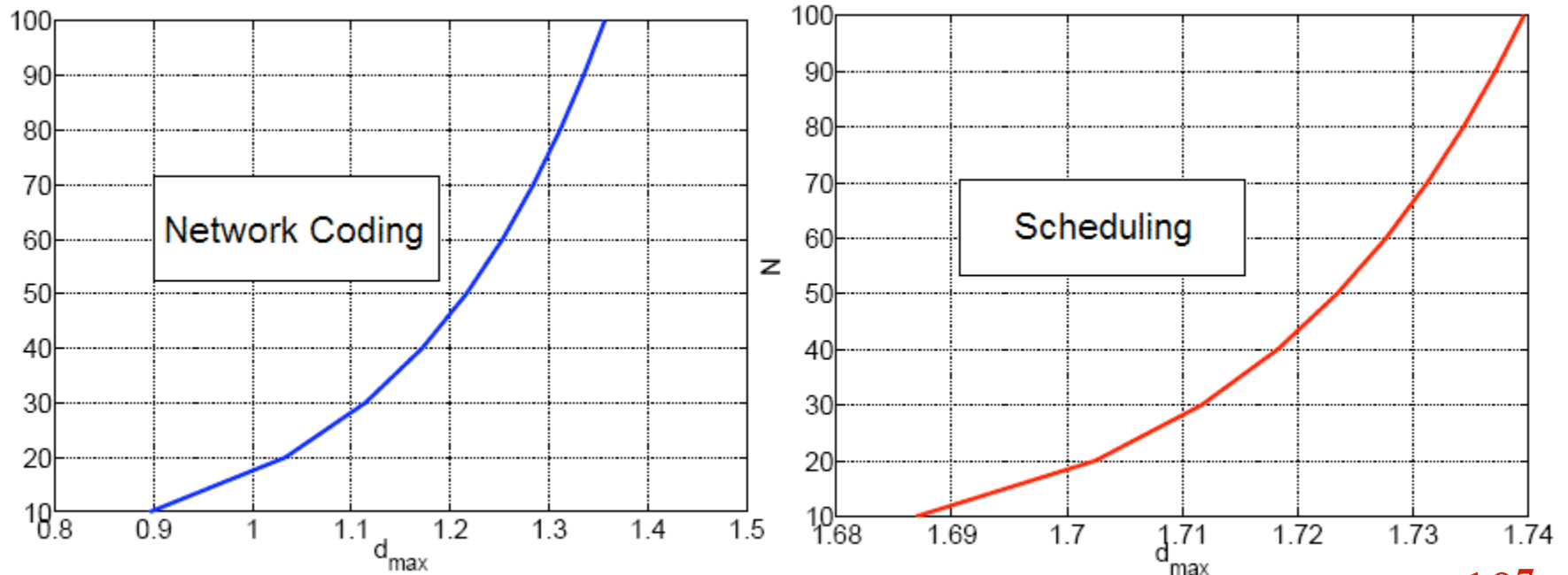
- Each user has a delay constraint Θ that is distributed uniformly between θ and d_{max}
 - Only users that observe a mean delay $E[D] \cdot \Theta$ enter the system
-

Numerical Results – Inelastic Traffic



Admitted traffic rate λ with varying delay constraints for $N=50$,
 $K=20$

Numerical Results – Inelastic Traffic



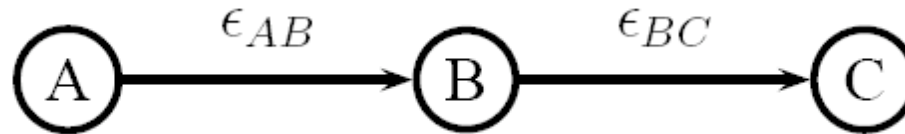
Number of supportable users as a function of delay $\times 10^7$ constraints for RNC and RR

- RNC can serve significantly more delay constrained traffic compared to RR under the same network setting

Main Results

- Elastic Traffic:
 - No scaling gains in N
 - Gains can be made arbitrarily high by picking large K
 - Inelastic Traffic:
 - For a fixed d_{max} , scaling laws do not make sense: we always have $\lambda(N) \neq 0$.
 - Need to look at scaling with d_{max}
 - With scheduling, the traffic is essentially restricted to being elastic
 - Coding allows us to serve significant inelastic traffic
 - We have only used feedback to indicate file completion – is that the missing piece for delay?
-

Feedback for reliability



Parameters we consider:

- delay incurred at B: excess time, relative to the theoretical minimum, that it takes for k packets to be communicated, disregarding any delay due to the use of the feedback channel
 - block size
 - feedback: number of feedback packets used
(feedback rate $R_f = \text{number of feedback messages} / \text{number of received packets}$)
 - memory requirement at B
 - achievable rate from A to C
-

Feedback for reliability

	Schemes	Rate	Delay	Feedback	Memory	Blocksize
I	FEC	min-cut: $(1 - \epsilon_{AB})$	$O(\sqrt{k \log k})$	0	$O(k)$	k
II	end-to-end FEC	$(1 - \epsilon_{AB})(1 - \epsilon_{BC})$	$O(k)$	0	0	k
III	FEC-m	$(1 - \epsilon_{AB})(1 - \pi_m)$	$O(k)$	0	m	k
IV	ARQ ($R_f = 1$)	min-cut: $(1 - \epsilon_{AB})$	$O(\sqrt{k})$	k	$O(\sqrt{k})$	1
V	FEC+ARQ (R_f)	min-cut: $(1 - \epsilon_{AB})$	$O(\sqrt{k})$	kR_f	$O(\sqrt{k})$	k

Follow the approach of [Pakzad et al. 05], [Lun et al. 06]

Scheme V allows us to achieve the min-cut rate, while keeping the average memory requirements at node B finite

note that the feedback delay for Scheme V is smaller than the usual ARQ (with $R_f = 1$) by a factor of R_f

feedback is required only on link BC