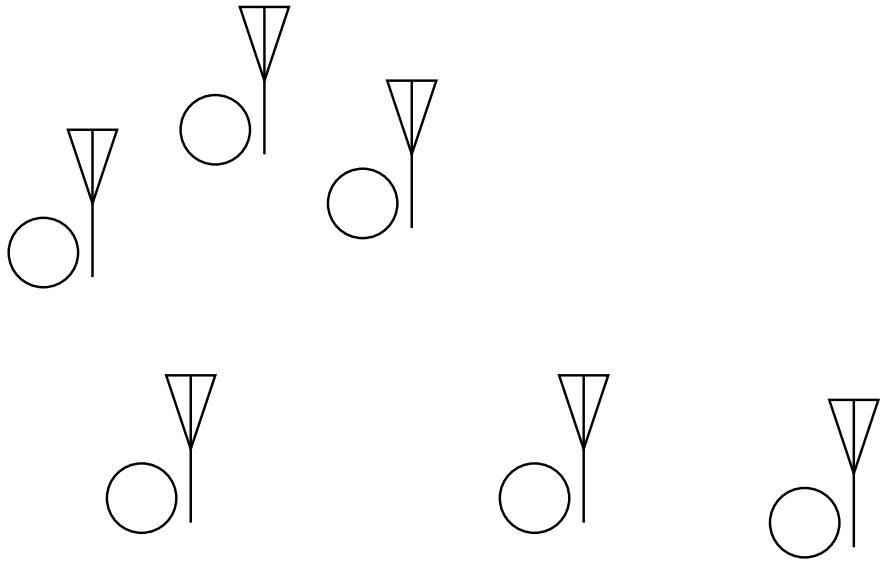
Linear Network Coding for Multicasting

6.454 presentation

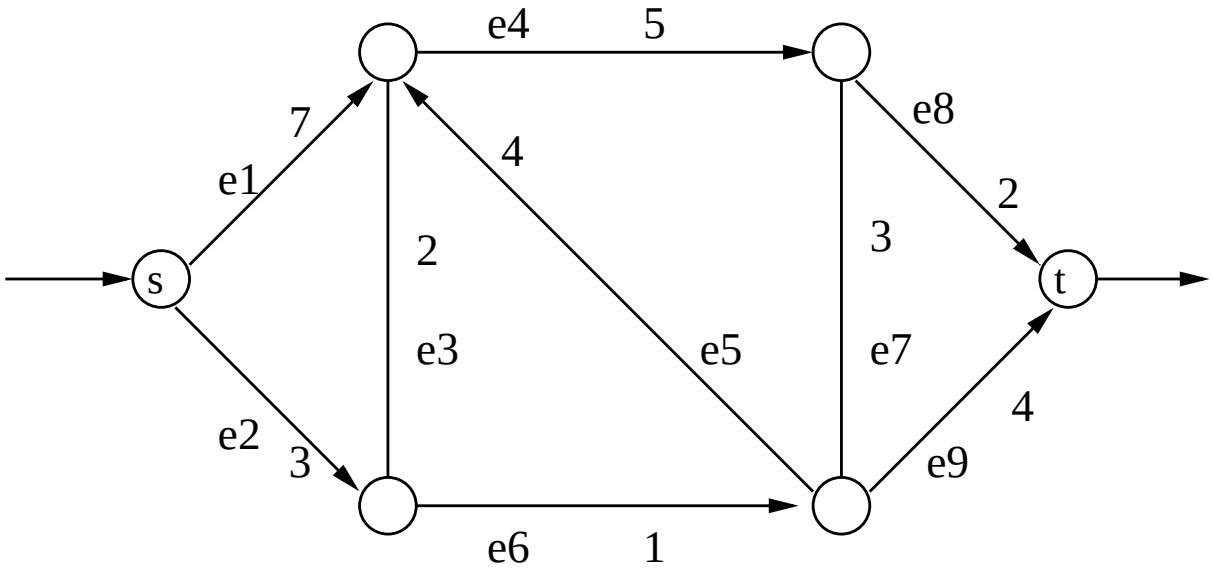
Uri Erez

The Place of Network Coding in the Grand Design



- Multiuser information theory:
  - Formally includes just about anything. Traditionally, emphasis quite different, noisy channels, correlation between signals in different channels.
- Network coding:
  - channels are reduced to bit pipes. 'Theory of smart routing".

Single-Source Single-Sink



- Shannon, Feinstein (1956) / Ford, Fulkerson, Dantzig (1956):

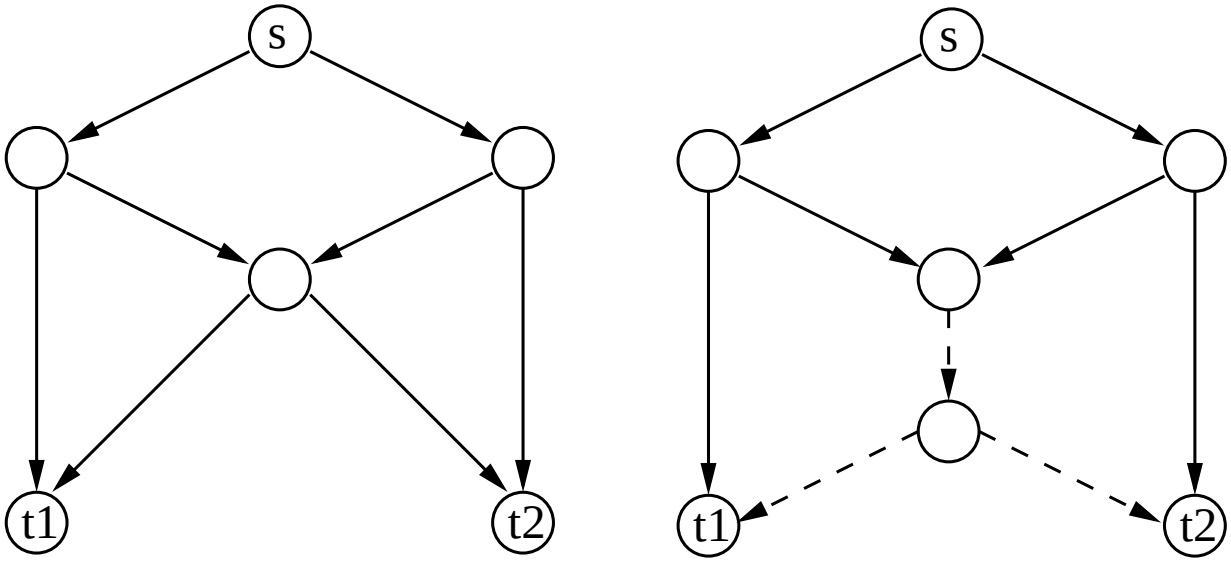
$$\text{MAX FLOW} = \text{MIN CUT}$$

- information treated as "physical commodity".
- No need for coding. Simply routing.

Scope of Presentation

- One source. Multiple sinks.
- Multicasting: Same information sent to all sinks.
- Restrict attention to directed graphs with no (directed) cycles.
- Unit capacity edges.
- Best we could hope for: send information at rate equal to minimal of single-source single-sink max flows.

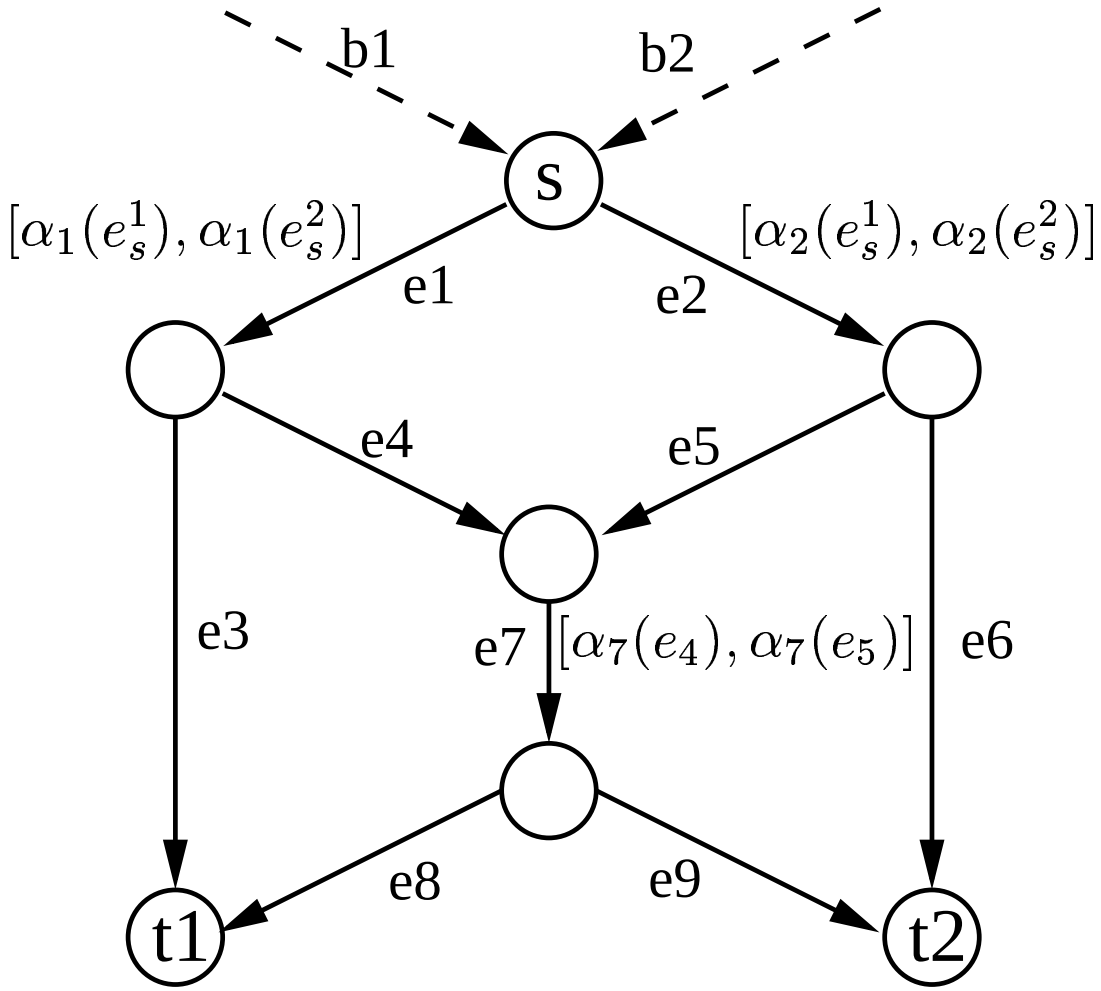
Multicasting: Classic Example



- Bottom line: No real conflict of interests between users.

Formulation as Linear Coding

- Local linear combination of incoming bits:
  $b(e) = \sum_{p \in \Gamma_I(start(e))} \alpha_e(p) b(p)$
- Global combination of source bits:
  $b(e) = \sum_{i=1}^{h} \beta_i b_i$



Formulation as Linear Coding: II

- $b(e_1) = \alpha_1(e_s^1)b_1 + \alpha_1(e_s^2)b_2$.
  $b(e_2) = \alpha_2(e_s^1)b_2 + \alpha_2(e_s^2)b_2$.
  $b(e_7) = \alpha_7(e_4)b(e_4) + \alpha_7(e_5)e_5$.
- $b(e_3) = b(e_4) = b(e_1)$. $b(e_5) = b(e_6) = b(e_2)$.
  $b(e_8) = b(e_9) = b(e_7)$.

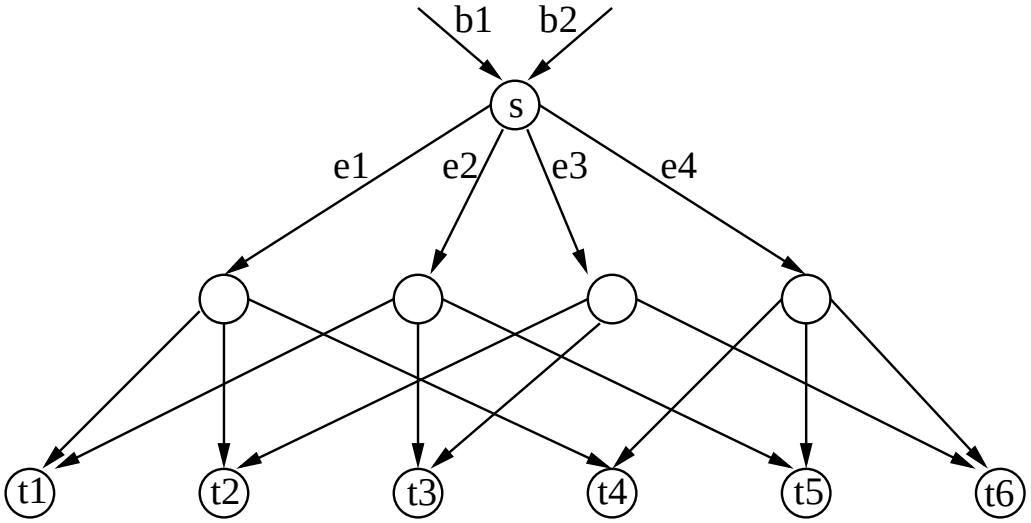
By recursion we have:

$$[b(e_3)\ b(e_8)] =$$
$$[b_1\ b_2] \begin{bmatrix} \alpha_1(e_s^1) & \alpha_7(e_4)\alpha_1(e_s^1) + \alpha_7(e_5)\alpha_2(e_s^1) \\ \alpha_1(e_s^2) & \alpha_7(e_4)\alpha_1(e_s^2) + \alpha_7(e_5)\alpha_2(e_s^2) \end{bmatrix}$$

$$[b(e_6)\ b(e_9)] =$$
$$[b_1\ b_2] \begin{bmatrix} \alpha_1(e_s^1) & \alpha_7(e_4)\alpha_1(e_s^1) + \alpha_7(e_5)\alpha_2(e_s^1) \\ \alpha_1(e_s^2) & \alpha_7(e_4)\alpha_1(e_s^2) + \alpha_7(e_5)\alpha_2(e_s^2) \end{bmatrix}$$

- Find $\alpha_{1,2,7}(\cdot)$ such that both matrices are invertible.

Example II



$$[b(e_1)\ b(e_2)\ b(e_3)\ b(e_4)] =$$

$$[b1\ b2] \begin{bmatrix} \alpha_1(e_s^1) & \alpha_2(e_s^1) & \alpha_3(e_s^1) & \alpha_4(e_s^1) \\ \alpha_1(e_s^2) & \alpha_2(e_s^2) & \alpha_3(e_s^2) & \alpha_4(e_s^2) \end{bmatrix}$$

- Can we choose $\alpha_{1,2,3,4}(\cdot)$ such that every 2×2 submatrix is invertible?

Example II Continued

- What if we group bits in blocks of 2?

$$\left[b^1(e_1)\ b^2(e_1),\ b^1(e_2)\ b^2(e_2),\ b^1(e_3)\ b^2(e_3),\ b^1(e_4)\ b^2(e_4)\right] =$$

$$[b_1^1\ b_1^2,\ b_2^1\ b_2^2] \left[\begin{array}{cc|cc|cc|cc} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \hline \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{array}\right]$$

- Can we choose $\alpha_{1,2,3,4}(\cdot)$ such that every 2×2 block submatrix is invertible?
- Use RS code...

General vs. Galois Linear Coding

In general linear coding "space and time bits on equal footing", no coupling of time bits.

- Local linear combination of incoming bits at time $i = 1, \ldots, DELAY$:
  $b_i(e) = \sum_{p \in \Gamma_I(start(e))} \sum_{j=1}^{DELAY} \alpha_e^j(p) b_j(p)$

$\mathbf{b}(\cdot)$ viewed as elements of a finite field $\Rightarrow$ can take linear combinations over field

- Local linear combination of symbols
  $\mathbf{b} \in GF(2^{DELAY})$:
  $\mathbf{b}(e) = \sum_{p \in \Gamma_I(start(e))} \alpha_e(p) \mathbf{b}(p)$

- We shall see: Galois linear coding is sufficient for multicasting (but not for general networks...).

Central Result

THEOREM:

$$MAXRATE = \min_{t \in T} MINCUT(s \to t)$$

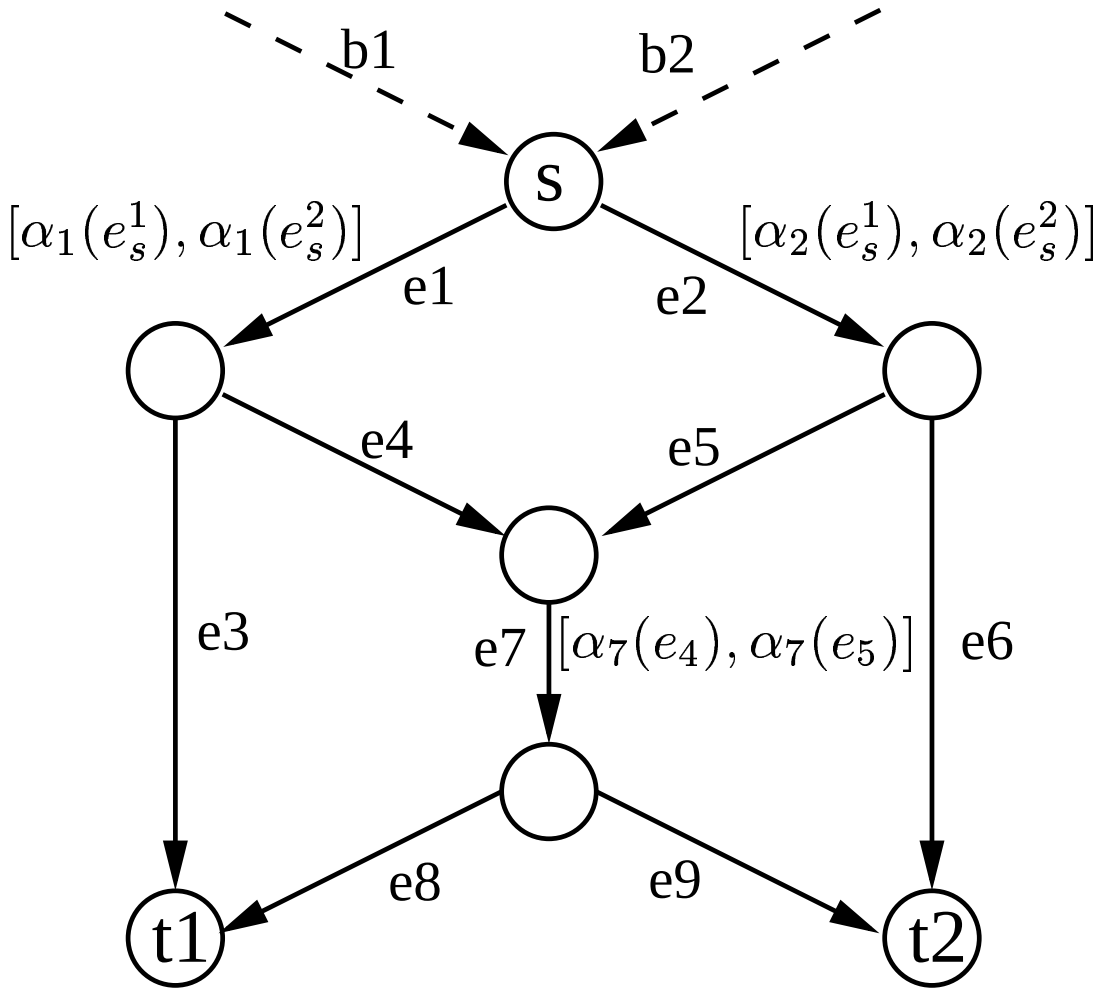
Furthermore,

- This rate is achieved by (Galois) linear coding.
- The code may be found in polynomial time.

Outline of Basic Algorithm

- For each sink, find h flow paths from source to sink.
- Denote F_t: flow associated with sink t.
- Proceed on vertices in topological order.
- $\mathcal{E}_t$: Set of edges $|\mathcal{E}_t| = h$ in flow F_t, one edge from each path (that whose coding vector was determined most recently).
- At each step: Draw $\alpha(\cdot)$ at random. Make sure that for each sink the set of symbols sent on edges of $\mathcal{E}_t$ remains linearly independent combinations of the messages.

Algorithm by Example



- at each step: Draw local linear combination at random; make sure that symbols on new $\mathcal{E}_t$ are linear independent combinations of messages.

Analysis: Existence

- Take $|\mathbb{F}| \geq 2 \cdot |T|$.
- Assume sets $\mathcal{E}_t$ are good so far for all t and we are now determining $\alpha_e(\cdot)$.
- For a fixed sink, the probability that a randomly chosen $\alpha_e(\cdot)$ is bad is $\frac{1}{|\mathbb{F}|}$.
- By union bound prob. of success per edge $\geq 1 - |T|/|\mathbb{F}| \geq 1/2$.
- $\Rightarrow \Pr\{\text{successful network}\} > 0$. QED

Monte Carlo → Vegas

- Draw random linear combination $\alpha_e(\cdot)$. probability of success $\geq 1/2$.
- Do (at most) $|T|$ linear independence tests.
- Proceed when good combination is found. Expected # of trials per edge ≤ 2.

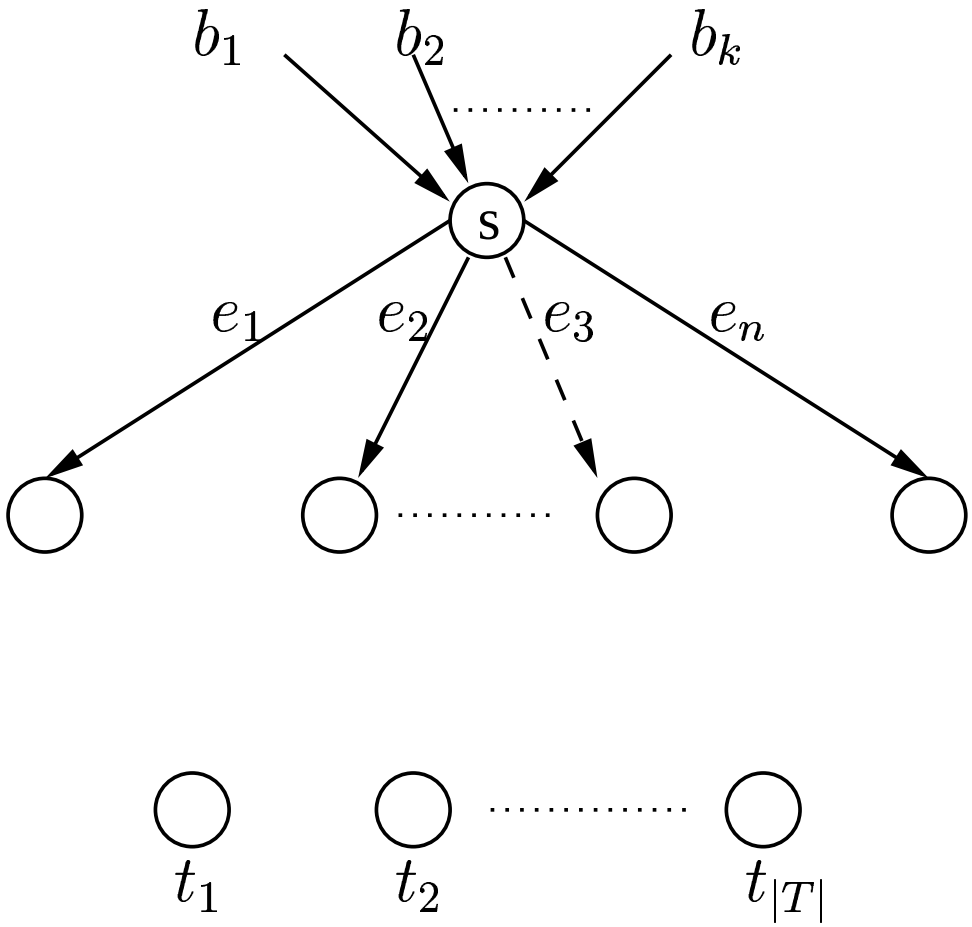
Summary of Algorithm Complexity

- randomized algorithm: expected running time $O(|E| \cdot |T| \cdot h^2)$. $|\mathbb{F}| \geq 2 \cdot |T|$ sufficient.
- node v needs time $O(\min(|\Gamma_I(v)|, |T|))$ to compute each output symbol. The source needs $O(h)$.
- Each sink needs time $O(h^2)$ for reconstruction.
- Operations are over field with 2^{DELAY} elements.
- Deterministic version of algorithm: Expected running time $O(|E| \cdot |T| \cdot h \cdot (h + |T|))$. $|\mathbb{F}| \geq |T|$ sufficient.

Comments on resulting codes

- Does the algorithm yield low complexity codes?
- Can the alphabet size be reduced (for non trivial networks..)?
- How large a gain can we get relative to an uncoded system?
- Some insight is offered by considering "MDS code networks".

MDS Codes Revisited



- We consider the cases:
  - $h = 2$
  - $h = n/2$

Lower Bound on Alphabet Size

- Is the algorithm good, obtaining codes with small delay?
- $h = 2 \Rightarrow |T| = n(n-1) = O(n^2)$.
- Lehman $\times$ 2 observe:
  - *Any* $(n, 2)$ MDS code has alphabet size $|\mathbb{F}| \geq n - 1$.
  - $\Rightarrow$ For worst case network we have:

$$|\mathbb{F}| \geq O(\sqrt{|T|})$$

  - Finding minimal $|\mathbb{F}|$ is NP-hard.
  - Approximation feasible?

Large Coding Gain

- Take $h = n/2$.
- Note that for an uncoded system

$$FLOW < 2$$

- Proof:
  - Suppose we try to send $2 \cdot DELAY$ bits.
  - U set of intermediate nodes. U_i subset receiving b_i.
  - We have $\sum_{i=1}^{2 \cdot DELAY} |U_i| \leq 2h \cdot DELAY$.
  - $\Rightarrow$ There is an i for which $|U_i| \leq h$.
  - $\Rightarrow$ For any subset of $U \setminus U_i$, the corresponding sink does not receive b_i.
- For coded system:
  - flow=$n/2$.
  - number of sinks=$\binom{n}{n/2} \approx 2^n$.
- $\Rightarrow$ coding gain=$O(\log |T|)$.

Reed and Solomon strike back

- Keep $h = n/2$, $\Rightarrow |T| = O(2^n)$.
- The algorithm uses $|\mathbb{F}| = O(|T|)$.
- An RS code requires an alphabet size of only $|\mathbb{F}| = n = O(\log |T|)$.

Some Extensions

- Can replace operations over finite field by convolution.
- Can be extended to graphs with cycles.
- Results easily generalize to multi-source multi-sink as long as all sinks demand the same information $\rightarrow$ no conflict of interests.
- Otherwise the problem is hard. Subject of current research.
- Are linear codes sufficient? Probably, but have to be general.
- Are there efficient algorithms that find approximate rate region?
- Combine with fountain coding approach?