Questions

- What do we know about the non-multicast case?
- Vector solutions versus instantaneous solutions.
- The issue of linearity!
- Is the non-multicast case interesting?
The algebraic setup

A linear network

Input vector: \( \mathbf{x}^T = (X(v, 1), X(v, 2), \ldots, X(v', \mu(v')) ) \)

Output vector: \( \mathbf{z}^T = (Z(u, 1), Z(u, 2), \ldots, Z(u', \nu(u')) ) \)

Transfer matrix: \( M, \mathbf{z} = M \mathbf{x} = B \cdot G \cdot A \mathbf{x} \)

\( \xi = (\xi_1, \xi_2, \ldots, ) = (\ldots, \alpha_{e,l}, \ldots, \beta_{e',e}, \ldots, \epsilon_{e',j}, \ldots) \)
\[ z = Mx = B \cdot (I - F^T)^{-1} \cdot A x \]

\[ \xi = (\xi_1, \xi_2, \ldots) = (\ldots, \alpha_{e,l}, \ldots, \beta_{e',e}, \ldots, \varepsilon_{e',j}, \ldots) \]

For acyclic networks the elements of \( G \) (and hence \( M \)) are polynomial functions in variables \( \xi = (\xi_1, \xi_2, \ldots, \) )

\[ \Rightarrow \text{an algebraic characterization of flows}.... \]
General Problems \((\mathcal{G}, \mathcal{C})\)

A general directed network with \(N\) sources and \(K\) sinks is defined as:

\[
\mathcal{C} = \{(v_i, u_j, \mathcal{K}(v_i, u_j))\}
\]

The matrix \(M\) is given by:

\[
M = \begin{pmatrix}
M_{1,1} & M_{1,2} & \cdots & M_{1,K} \\
M_{2,1} & M_{2,2} & & M_{2,K} \\
\vdots & & & \vdots \\
M_{N,1} & M_{N,2} & \cdots & M_{N,K}
\end{pmatrix}
\]

\(M_{i,j}\) corresponds to \(c_{i,j} = (v_i, u_j, \mathcal{K}(v_i, u_j))\).
**Theorem** [Generalized Min-Cut Max-Flow Condition] Let an acyclic, delay-free scalar linear network problem \((G, \mathcal{C})\) be given and let \(M = \{M_{i,j}\}\) be the corresponding transfer matrix relating the set of input nodes to the set of output nodes. The network problem is solvable if and only if there exists an assignment of numbers to \(\xi\) such that

1. \(M_{i,j} = 0\) for all pairs \((v_i, v_j)\) of vertices such that \((v_i, v_j, \mathcal{X}(v_i, v_j)) \notin \mathcal{C}\).

2. If \(\mathcal{C}\) contains the connections
   \[(v_{i_1}, v_j, \mathcal{X}(v_{i_1}, v_j)), (v_{i_2}, v_j, \mathcal{X}(v_{i_2}, v_j)), \ldots, (v_{i_\ell}, v_j, \mathcal{X}(v_{i_\ell}, v_j))\]
   the determinant of \([M_{i_1,j}^T, M_{i_2,j}^T, \ldots, M_{i_\ell,j}^T]\) is nonzero.
The ideal of \((G, C)\)

Entries in \(M_{i,j}\) that have to evaluate to zero: \(f_1(\xi), f_2(\xi), \ldots, f_L(\xi)\)

Determinants of submatrices that have to evaluate to nonzero values: \(g_1(\xi), g_2(\xi), \ldots, g_{L'}(\xi)\)

\[
\text{Ideal}((G, C)) = \langle f_1(\xi), f_2(\xi), \ldots, f_L(\xi), 1 - \xi_0 \prod_{i=1}^{L'} g_i(\xi) \rangle
\]

\[
\text{Var}((G, C)) = \{(a_1, a_2, \ldots, a_n) \in \overline{F}^n : f(a_1, a_2, \ldots, a_n) = 0 \ \forall \ f \in \text{Ideal}((G, C))\}.
\]
The central Theorem

Theorem Let a scalar linear network problem \((\mathcal{G}, \mathcal{C})\) be given. The network problem is solvable if and only if \(\text{Var}(\mathcal{G}, \mathcal{C})\) is nonempty or equivalently, the ideal \(\text{Ideal}(\mathcal{G}, \mathcal{C})\) is a proper ideal of \(\overline{F}[\xi_0, \xi]\), i.e \(\text{Ideal}(\mathcal{G}, \mathcal{C}) \subsetneq \mathbb{F}_2[\xi_0, \xi]\).
So why is the general case so much harder?

For the general case we need to find solutions to some system of polynomial equations!

For the multicast case we need to find non solutions to some system of polynomial equations!

Another way to phrase this is: In a multicast setup everybody wants everything so the issue of interference is moot!

For the general case we may have carefully balanced solutions where some unwanted information cancels out in clever ways.....
In the general problem a time sharing combination of several solutions which themselves both violate the constraints may be necessary. (This cannot happen in the multicast case!)

Doubling the bandwidth more than doubles capacity! Tripling the bandwidth does not work!
Vector solutions may help


Examples of networks that need vector length that are multiples of \( k \) for any \( k \).
By combining networks requiring vector length that are multiples of primes the following bound is derived:

**Theorem** There exist directed networks with $O(n)$ nodes such that a solution to the network coding problem requires at least an alphabet size of $2^{(e^\sqrt{n^{1/3}})}$
R. Dougherty, C. Freiling, and K. Zeger, "Insufficiency of Linear Coding in Network Information Flow", preprint, February 2004

This network is not solvable over any Galois field, including vector versions thereof!

(still the network has a distinctly linear feel to it....)
Non-multicast connections - use of cost criterion

- We propose a linear optimization problem whose minimum cost is no greater than the minimum cost of any routing solution.

- Moreover, feasible solutions correspond to network codes that perform linear operations on vectors created from the source processes.

- Main idea: create a set partition of \(\{1, \ldots, M\}\) that represents the sources that can be mixed (combined linearly) on links going into \(i\).

- Code construction steps through the nodes in topological order, examining the outgoing links and defining global coding vectors on them.
Non-multicast connections - use of cost criterion

- For any node $i$, let $T(i)$ denote the sinks that are accessible from $i$.

- Let $C(i)$ be a set partition of $\{1, \ldots, M\}$ that represents the sources that can be mixed (combined linearly) on links going into $i$. For a given $C \in C(i)$, the sinks that receive a source process in $C$ by way of link $(j, i)$ in $A$ (set of arcs) either receive all the source processes in $C$ or none at all.
Non-multicast connections - use of cost criterion

minimize \( \sum_{(i,j) \in A} a_{ij} z_{ij} \)

subject to

\( c_{ij} \geq z_{ij} = \sum_{C \in C(j)} y_{ij}^{(C)} \), \( \forall (i, j) \in A \),

\( y_{ij}^{(C)} \geq \sum_{m \in C} x_{ij}^{(t,m)} \), \( \forall (i, j) \in A, t \in T, C \in C(j) \),

\( x_{ij}^{(t,m)} \geq 0 \), \( \forall (i, j) \in A, t \in T, m = 1, \ldots, M \),

\[
\sum_{\{j|(i,j) \in A\}} x_{ij}^{(t,m)} - \sum_{\{j|(j,i) \in A\}} x_{ji}^{(t,m)} = \begin{cases} R_m & \text{if } v = s_m \text{ and } m \in D(t), \\ -R_m & \text{if } m \in D(i), \\ 0 & \text{otherwise}, \end{cases}
\]

\( \forall i \in A, t \in T, m = 1, \ldots, M, (1) \)

where we define \( D(i) := \emptyset \) for \( i \) in \( N \setminus T \). Again, the optimization problem can be easily modified to accommodate convex cost functions.
Is the non-multicast case interesting?
The non multicast scenario exhibits far more subtleties than the multicast setup. This is due to the fact that cancellations now need to be carefully arranged.

There are some generalizations to vector solutions which can be incorporated into the algebraic framework.

Not even the principle problem of linearity vs. nonlinear operation is entirely clear.

From a practical point of view a non interacting arrangement of multicast is most interesting and robust.
A principled optimization approach to match or outperform routing

- An optimization that yields a solution that is no worse than multicommodity flow
- The optimization is in effect a relaxation of multicommodity flow - akin to Steiner tree relaxation for the multicast case
- A solution of the problem implies the existence of a network code to accommodate the arbitrary demands - the types of codes subsume routing
- All decoding is performed at the receivers
- We can provide an optimization, with a linear code construction, that is guaranteed to perform as well as routing [Lun et al. 04]
Optimization

gives a set partition of \{1, \ldots, M\} that represents the sources that can be mixed (combined linearly) on links going into \(j\).

minimize \( \sum_{(i,j) \in A} a_{ij} z_{ij} \)

subject to \( c_{ij} \geq z_{ij} = \sum_{C \in \mathcal{C}(j)} y_{ij}^{(C)} \), \( \forall (i, j) \in A \),

\( y_{ij}^{(C)} \geq \sum_{m \in C} x_{ij}^{(t,m)} \), \( \forall (i, j) \in A, t \in T, C \in \mathcal{C}(j) \),

\( x_{ij}^{(t,m)} \geq 0 \), \( \forall (i, j) \in A, t \in T, m = 1, \ldots, M \),

\( \sum_{\{j| (i, j) \in A\}} x_{ij}^{(t,m)} - \sum_{\{j| (j, i) \in A\}} x_{ji}^{(t,m)} = \begin{cases} R_m & \text{if } v = s_m \text{ and } m \in D(t), \\ -R_m & \text{if } m \in D(i), \\ 0 & \text{otherwise,} \end{cases} \)

Demands of \{1, \ldots, M\} at \(t\)

\( \forall i \in A, t \in T, m = 1, \ldots, M \)

Optimization for arbitrary demands with decoding at receivers
Coding and optimization

- Sinks that receive a source process in $C$ by way of link $(j, i)$ either receive all the source processes in $C$ or none at all.
- Hence source processes in $C$ can be mixed on link $(j, i)$ as the sinks that receive the mixture will also receive the source processes (or mixtures thereof) necessary for decoding.
- We step through the nodes in topological order, examining the outgoing links and defining global coding vectors on them (akin to [Jaggi et al. 03]).
- We can build the code over an ever-expanding front.
- We can go to coding over time by considering several flows for the different times - we let the coding delay be arbitrarily large.
- The optimization and the coding are done separately as for the multicast case, but the coding is not distributed.