Exploiting Interference through Structured Codes

Bobak Nazer

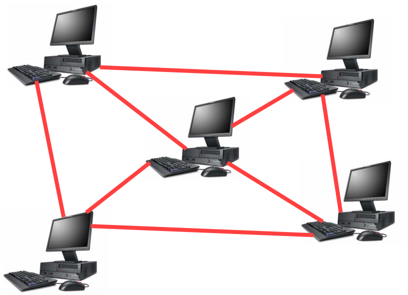
(Joint work with Michael Gastpar)

Wireless Foundations Center
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley

Dissertation Talk

May 18, 2009

Wired Networks



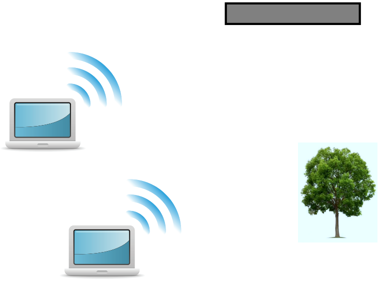
- Wired Network: users connected by point-to-point links or bit pipes.
- Interference only if two users share the same link.
- Success Story: The Internet

Success due in part to layered digital architecture.

$\cdots \implies$ Flows $\implies$ Packets $\implies$ Bits $\implies$ Signals

Physical Layer

Wireless Networks



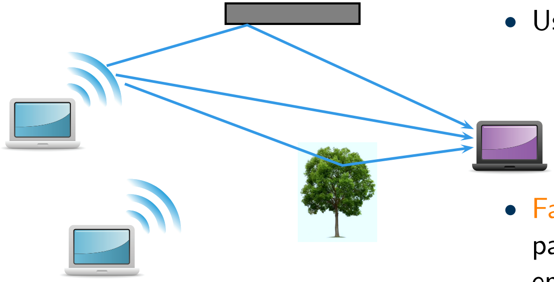
- Users share wireless medium.

- Fading due to different signal paths through the environment.

Current approach: Adapt existing wired network algorithms.

Avoid interference at all costs.

Wireless Networks

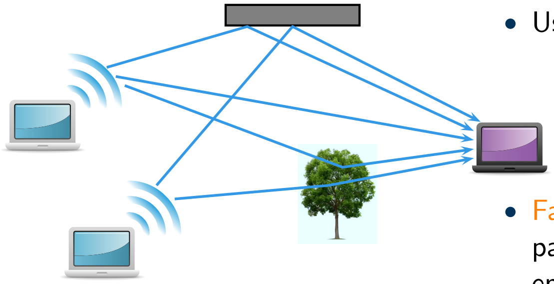


- Users share wireless medium.
- Fading due to different signal paths through the environment.

Current approach: Adapt existing wired network algorithms.

Avoid interference at all costs.

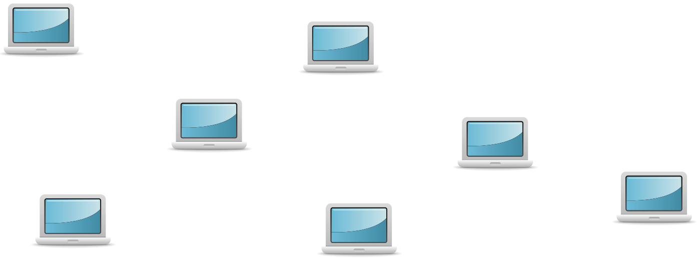
Wireless Networks



- Users share wireless medium.
- Fading due to different signal paths through the environment.

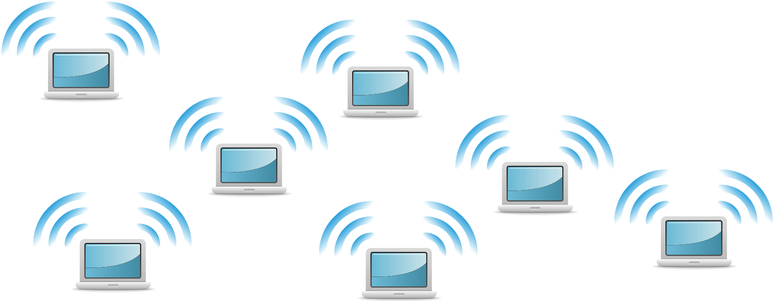
Current approach: Adapt existing wired network algorithms.

Avoid interference at all costs.



- Must cope with interference, fading, and noise.

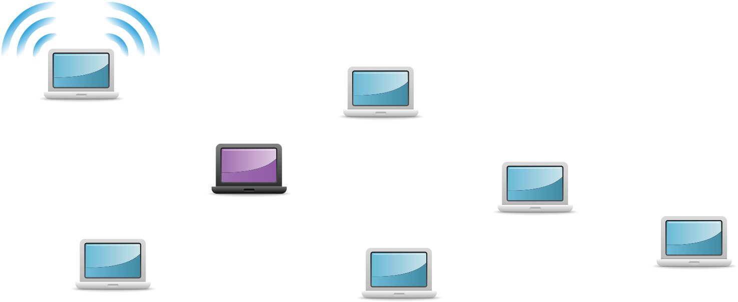
Wireless Network Model



- Must cope with interference, fading, and noise.
- Receivers observe noisy linear combinations of transmitted signals:

$$\mathbf{y} = \sum_j h_j \mathbf{x}_j + \mathbf{z}$$

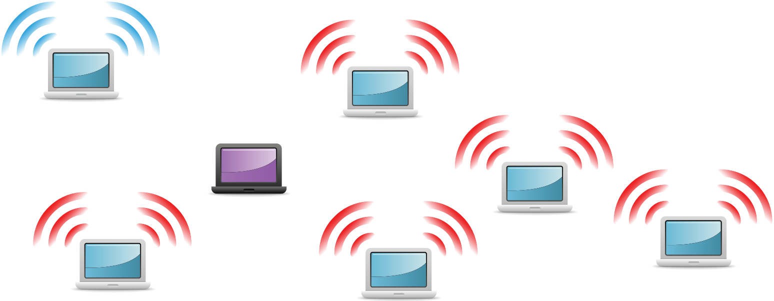
Treating Interference as Noise



- Establish connection between two users by treating other transmissions as noise:

$$\mathbf{y} = \sum_j h_j \mathbf{x}_j + \mathbf{z}$$

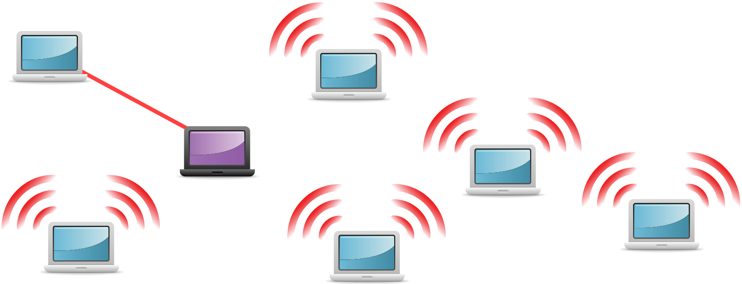
Treating Interference as Noise



- Establish connection between two users by treating other transmissions as noise:

$$\mathbf{y} = h_i \mathbf{x}_i + \sum_{j \neq i} h_j \mathbf{x}_j + \mathbf{z}$$

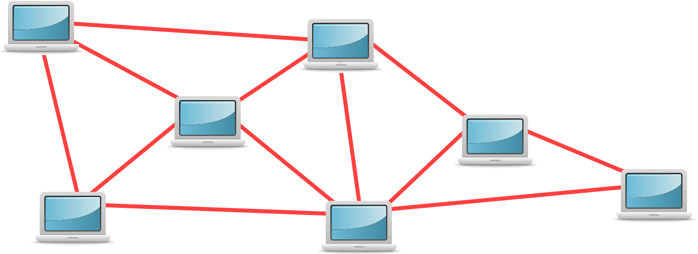
Treating Interference as Noise



- Establish connection between two users by treating other transmissions as noise:

$$\mathbf{y} = h_i \mathbf{x}_i + \sum_{j \neq i} h_j \mathbf{x}_j + \mathbf{z}$$

Treating Interference as Noise



- Establish connection between two users by treating other transmissions as noise:

$$\mathbf{y} = h_i \mathbf{x}_i + \sum_{j \neq i} h_j \mathbf{x}_j + \mathbf{z}$$

- Convert into network of reliable bit pipes.

Physical-Layer Cooperation

Lack of cooperation leads to treating other users as noise.

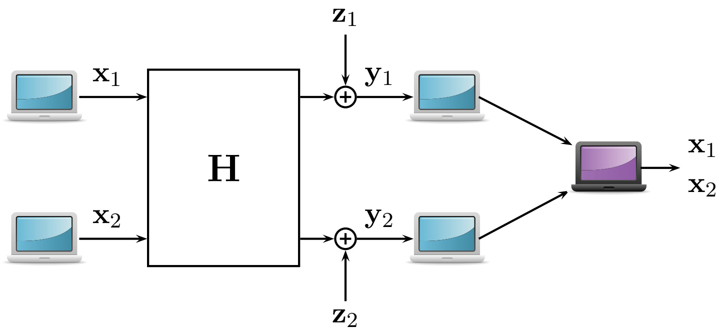
If users cooperate, we can exploit the noisy linear combinations of the wireless channel for throughput gains.

Two well-studied approaches:

- Compress-and-Forward: Send out vector-quantized received signal.
- Amplify-and-Forward: Repeat received signal.

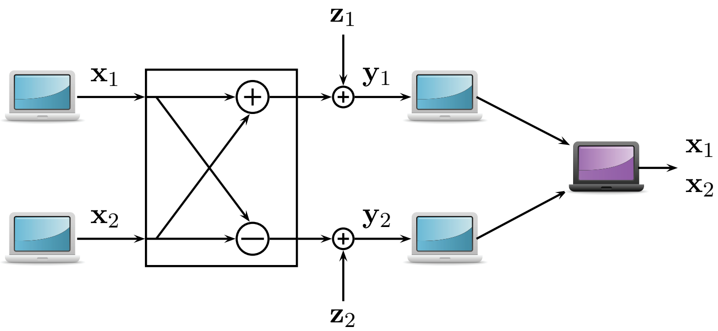
See, for instance, **Cover-El Gamal '79, Schein-Gallager '00, Sendonaris-Erkip-Aazhang '03, Laneman-Tse-Wornell '04, Kramer-Gastpar-Gupta '05, Gastpar-Vetterli '05, Özgur-Lévêque-Tse '07, Aleksic-Razaghi-Yu '07.**

Example: Cooperative Communication



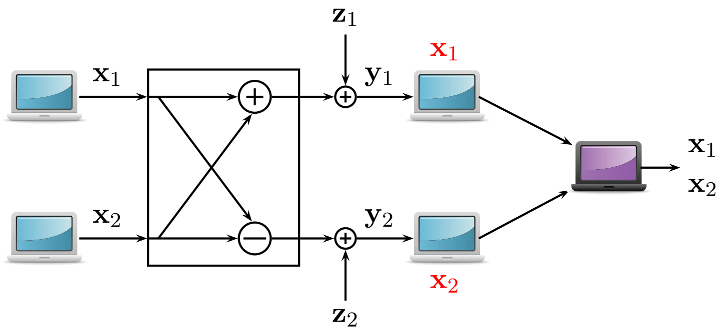
- Two users want to send messages across the network with the help of two relays.

Example: Cooperative Communication



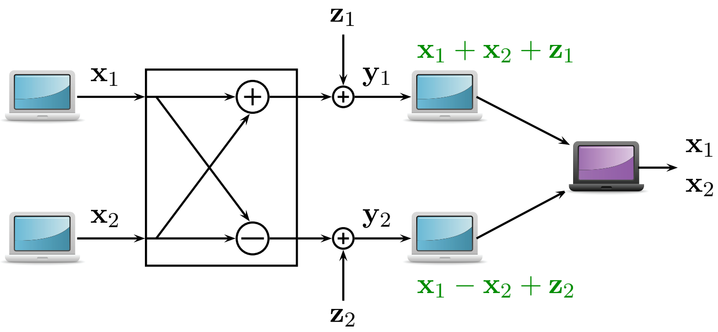
- Two users want to send messages across the network with the help of two relays.

Example: Cooperative Communication



- Two users want to send messages across the network with the help of two relays.
- Strategy 1: Each relay decodes one message.

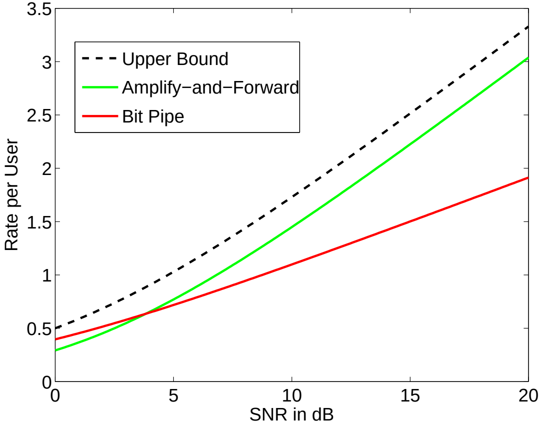
Example: Cooperative Communication



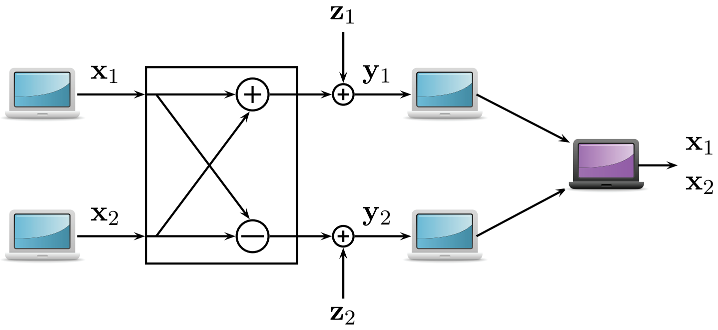
- Two users want to send messages across the network with the help of two relays.
- Strategy 1: Each relay decodes one message.
- Strategy 2: Relays send their observed signal to the destination without decoding.

Example: Cooperative Communication

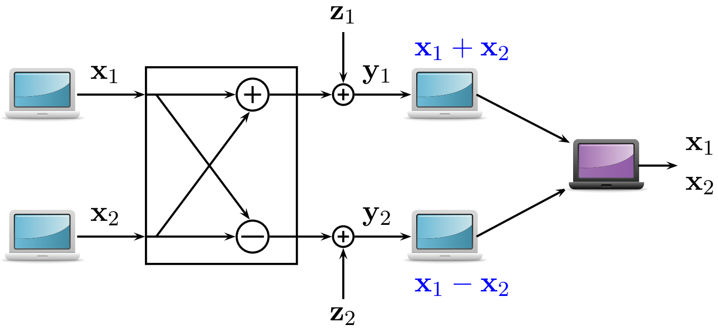
- Interference can be useful!
- Not captured by bit pipe approach.



Example: Cooperative Communication



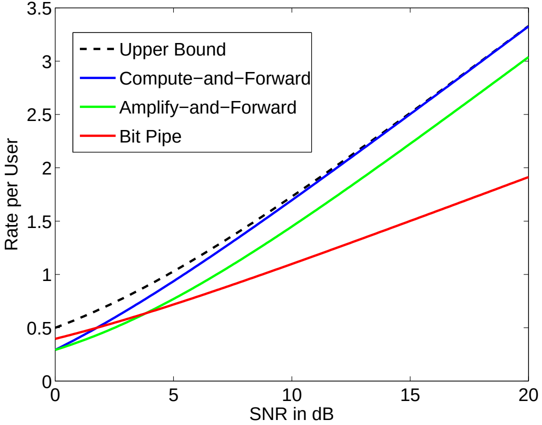
- What if each relay could decode a linear equation?



- What if each relay could decode a linear equation?
- Compute-and-Forward: One relay decodes the sum of codewords.
  Other relay decodes the difference.

Example: Cooperative Communication

- Compute-and-Forward is nearly optimal!



Talk Overview

1. How can we (reliably) compute over noisy channels?

2. What does this mean for wireless networks?

3. Beyond bits: Distributed signal processing applications.

Reliable Computation over Noisy Channels

Finite field messages:

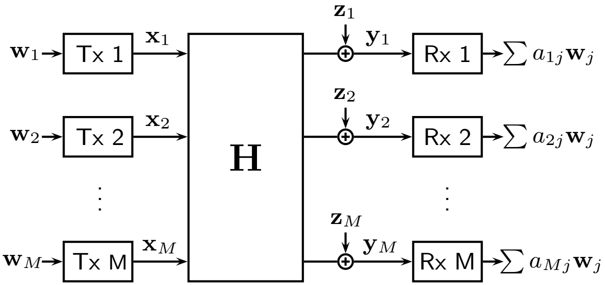
$$\mathbf{w}_j \in \mathbb{F}_p^k$$

Real-valued channel:

$$\mathbf{x}_j, \mathbf{y}_j, \mathbf{z}_j \in \mathbb{R}^n$$

Random fading:

$$h_{ij} \sim \mathcal{N}(0,1)$$



Reliable Computation over Noisy Channels

Finite field messages:

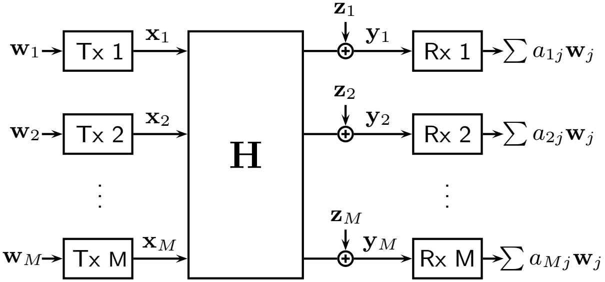
$$\mathbf{w}_j \in \mathbb{F}_p^k$$

Real-valued channel:

$$\mathbf{x}_j, \mathbf{y}_j, \mathbf{z}_j \in \mathbb{R}^n$$

Random fading:

$$h_{ij} \sim \mathcal{N}(0,1)$$



- Receivers know their fading coefficients. Transmitters do not.

Reliable Computation over Noisy Channels

Finite field messages:

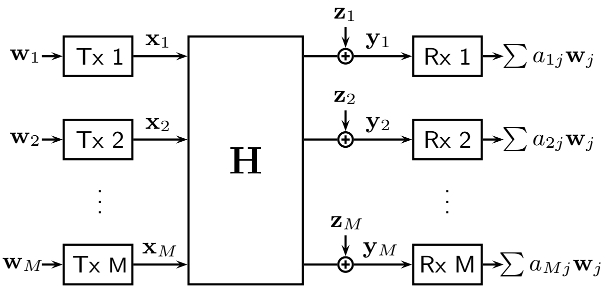
$$\mathbf{w}_j \in \mathbb{F}_p^k$$

Real-valued channel:

$$\mathbf{x}_j, \mathbf{y}_j, \mathbf{z}_j \in \mathbb{R}^n$$

Random fading:

$$h_{ij} \sim \mathcal{N}(0,1)$$

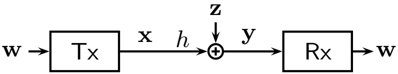


- Receivers know their fading coefficients. Transmitters do not.
- **Goal:** Recover equations reliably while maximizing rate

$$R = \frac{k}{n}\log_2 p$$

Usual Channel Coding

- Point-to-point communication: minimum distance between codewords important to protect against noise



- **Shannon '48**: Channel capacity:

$$C = \max_{p(X)} I(X;Y) = \frac{1}{2}\log\left(1 + h^2 \mathsf{SNR}\right)$$

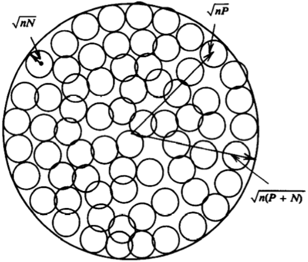
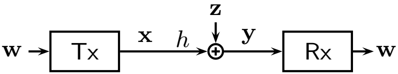


Figure 10.2. Sphere packing for the Gaussian channel.

(Cover and Thomas, *Elements of Information Theory*)

- Point-to-point communication: minimum distance between codewords important to protect against noise



- **Shannon '48**: Channel capacity:

$$C = \max_{p(X)} I(X;Y) = \frac{1}{2}\log\left(1 + h^2\mathsf{SNR}\right)$$

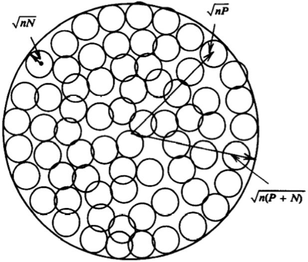
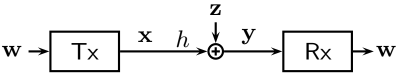


Figure 10.2. Sphere packing for the Gaussian channel.

(Cover and Thomas, *Elements of Information Theory*)

  - Many extensions: multiple-access (many-to-one), broadcast (one-to-many)

Usual Channel Coding



- Point-to-point communication: minimum distance between codewords important to protect against noise

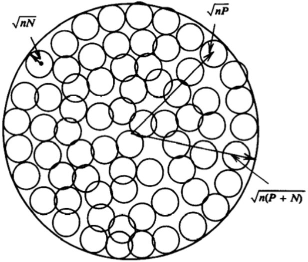


Figure 10.2. Sphere packing for the Gaussian channel.

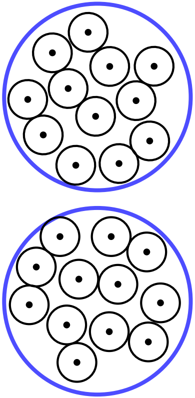
(Cover and Thomas, *Elements of Information Theory*)

- **Shannon '48**: Channel capacity:

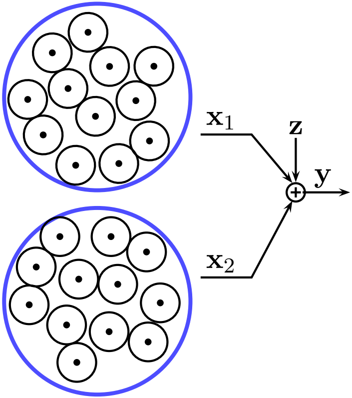
$$C = \max_{p(X)} I(X;Y) = \frac{1}{2}\log\left(1 + h^2\mathsf{SNR}\right)$$

  - Many extensions: multiple-access (many-to-one), broadcast (one-to-many)
  - Can we use these codes for efficient computation?

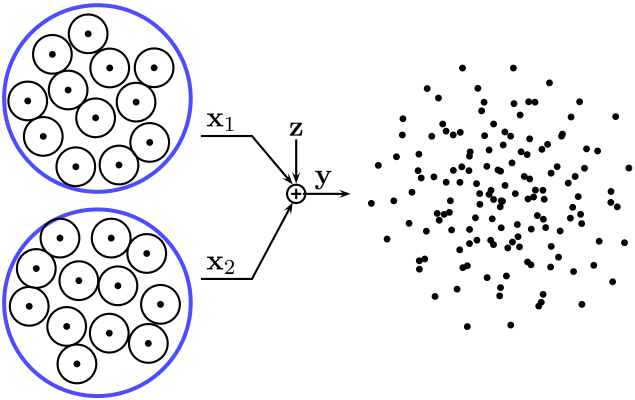
Usual Codes Not Good for Computation



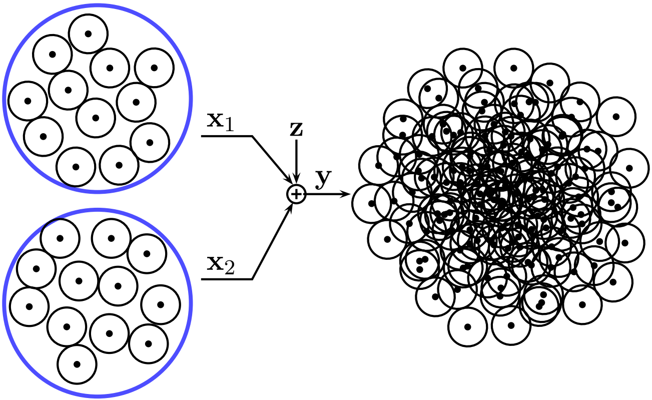
Usual Codes Not Good for Computation



Usual Codes Not Good for Computation

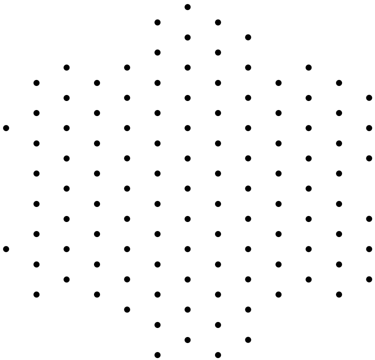


Usual Codes Not Good for Computation



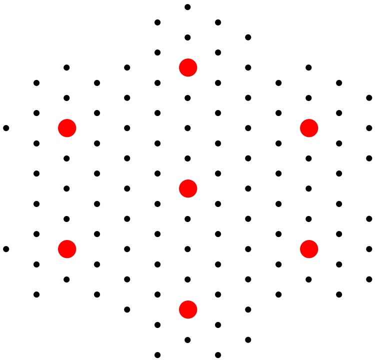
Nested Lattice Codes

- Lattice: linear tiling of $\mathbb{R}^n$.
- Λ_{FINE}: channel codewords



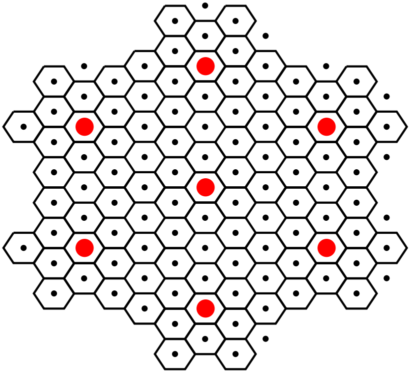
Nested Lattice Codes

- Lattice: linear tiling of $\mathbb{R}^n$.
- Λ_{FINE}: channel codewords
- $\Lambda_{\mathsf{COARSE}} \subset \Lambda_{\mathsf{FINE}}$: power constraint



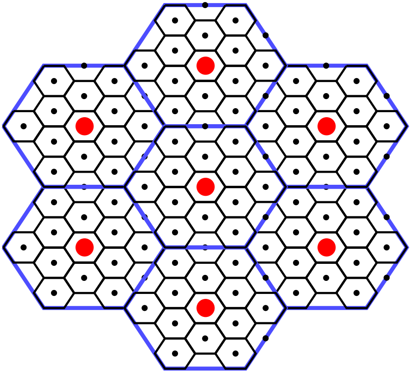
Nested Lattice Codes

- Lattice: linear tiling of $\mathbb{R}^n$.
- Λ_{FINE}: channel codewords
- $\Lambda_{\text{COARSE}} \subset \Lambda_{\text{FINE}}$: power constraint



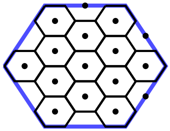
Nested Lattice Codes

- Lattice: linear tiling of $\mathbb{R}^n$.
- Λ_{FINE}: channel codewords
- $\Lambda_{\text{COARSE}} \subset \Lambda_{\text{FINE}}$: power constraint



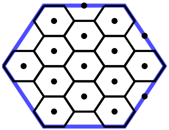
Nested Lattice Codes

- Lattice: linear tiling of $\mathbb{R}^n$.
- Λ_{FINE}: channel codewords
- $\Lambda_{\text{COARSE}} \subset \Lambda_{\text{FINE}}$: power constraint
- **Erez-Zamir '04:** Nested lattice codes achieve point-to-point AWGN capacity.



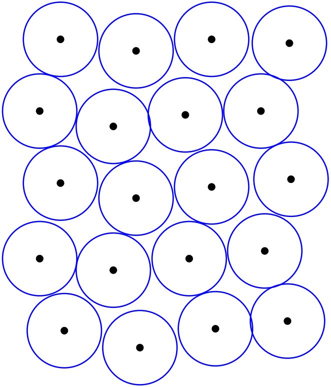
Nested Lattice Codes

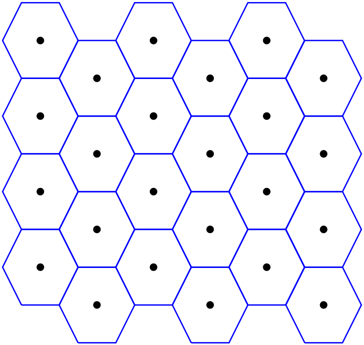
- Lattice: linear tiling of $\mathbb{R}^n$.
- Λ_{FINE}: channel codewords
- $\Lambda_{\text{COARSE}} \subset \Lambda_{\text{FINE}}$: power constraint
- **Erez-Zamir '04:** Nested lattice codes achieve point-to-point AWGN capacity.



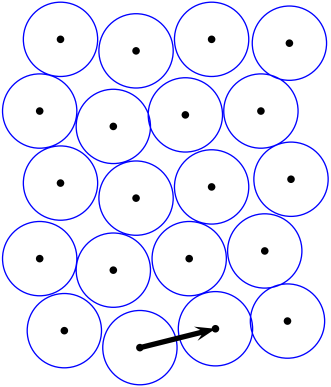
- Computation Coding Key Idea: All users employ the same nested lattice code.
- Could use any linear code instead (i.e. LDPC with QAM constellation).

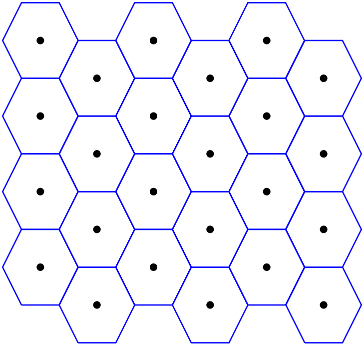
Protecting Linear Equations



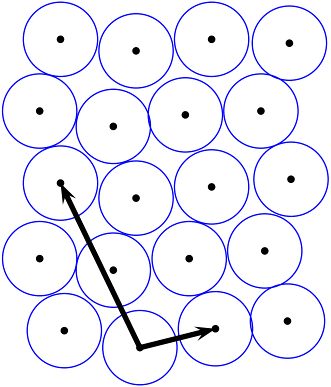


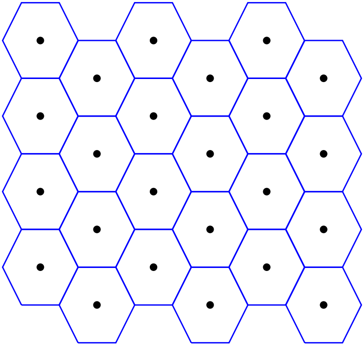
Protecting Linear Equations



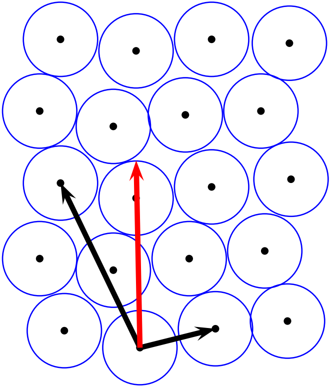


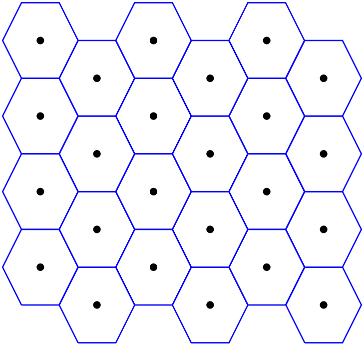
Protecting Linear Equations





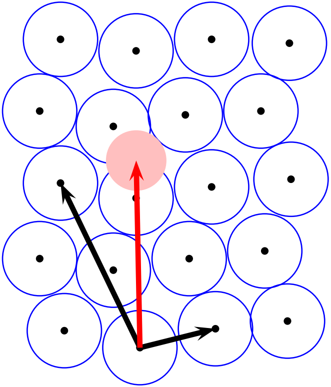
Protecting Linear Equations



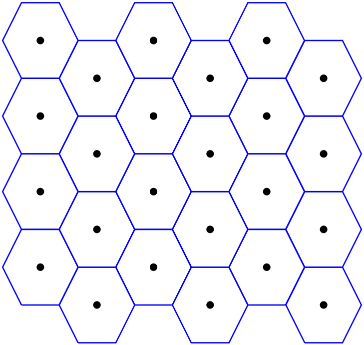


- Sum of codewords is not a codeword.
- Must decode individual messages.

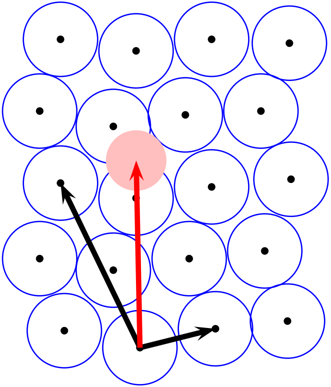
Protecting Linear Equations

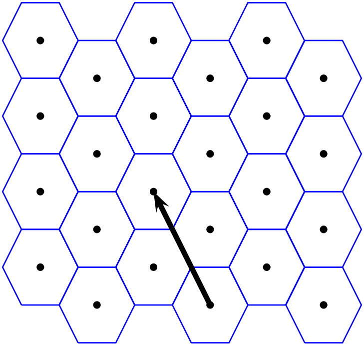


- Sum of codewords is not a codeword.
- Must decode individual messages.



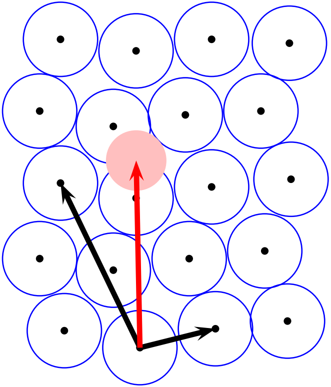
Protecting Linear Equations



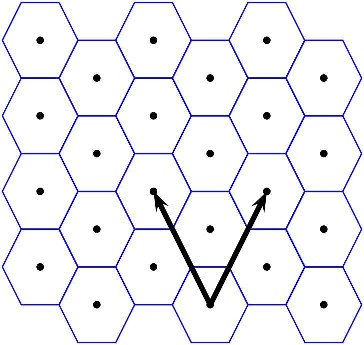


- Sum of codewords is not a codeword.
- Must decode individual messages.

Protecting Linear Equations

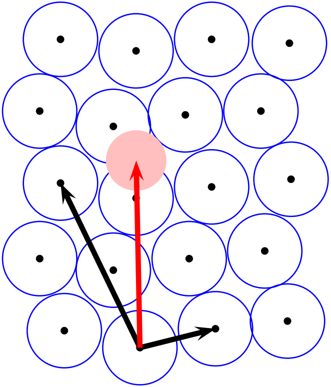


- Sum of codewords is not a codeword.
- Must decode individual messages.

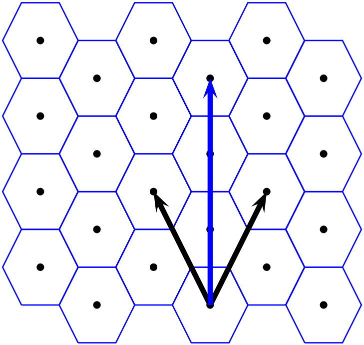


- Sum of codewords is a codeword.
- Can decode integer combinations of messages.

Protecting Linear Equations

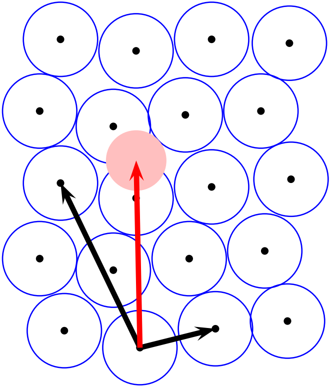


- Sum of codewords is not a codeword.
- Must decode individual messages.

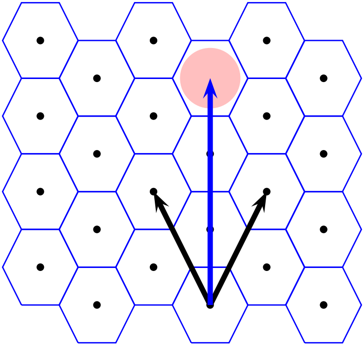


- Sum of codewords is a codeword.
- Can decode integer combinations of messages.

Protecting Linear Equations



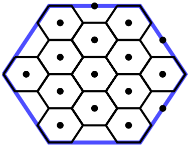
- Sum of codewords is not a codeword.
- Must decode individual messages.

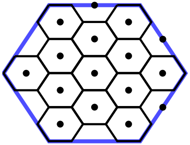


- Sum of codewords is a codeword.
- Can decode integer combinations of messages.

Computation Coding

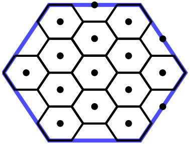
All users pick the same nested lattice code:



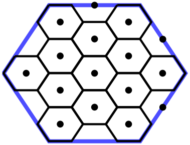


Choose messages over field $\mathbf{w}_i \in \mathbb{F}_p^k$:

$\mathbf{w}_1 \longrightarrow$

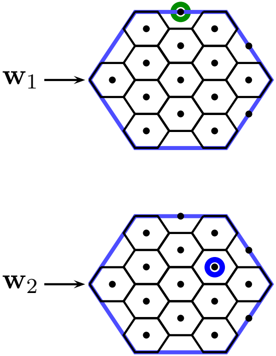


$\mathbf{w}_2 \longrightarrow$



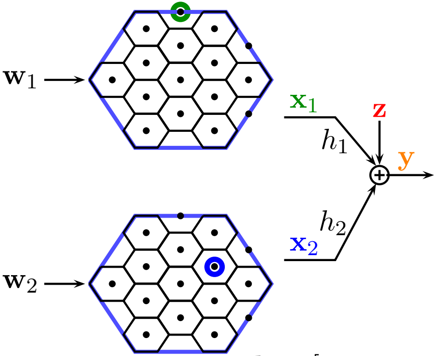
Computation Coding

Map $\mathbf{w}_i$ to lattice point in Λ_{FINE} mod Λ_{COARSE}:



Computation Coding

Transmit lattice points over the channel:

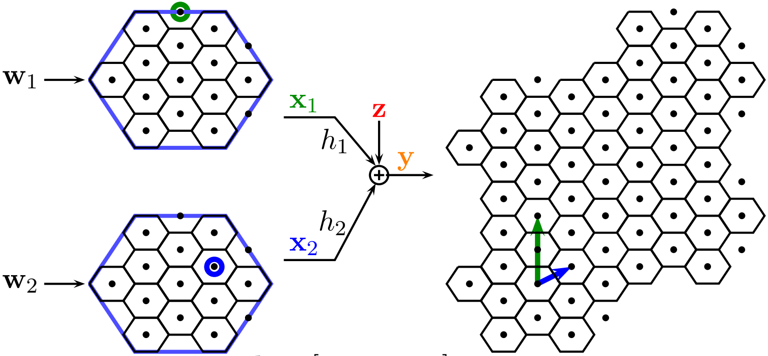


$$\mathbf{h} = [\ 1.4 \quad 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Computation Coding

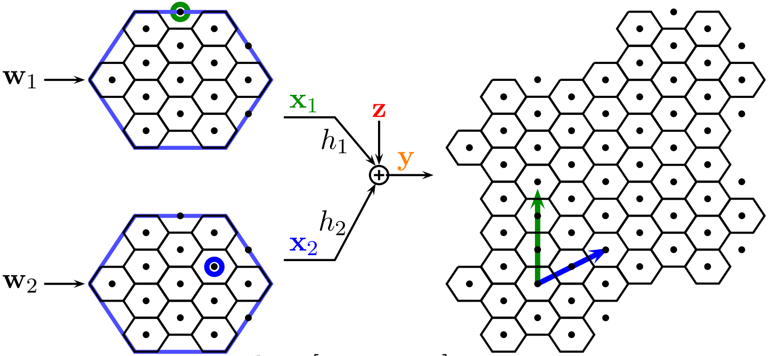
Transmit lattice points over the channel:



$$\mathbf{h} = [\ 1.4 \quad 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Lattice codewords are scaled by channel coefficients:

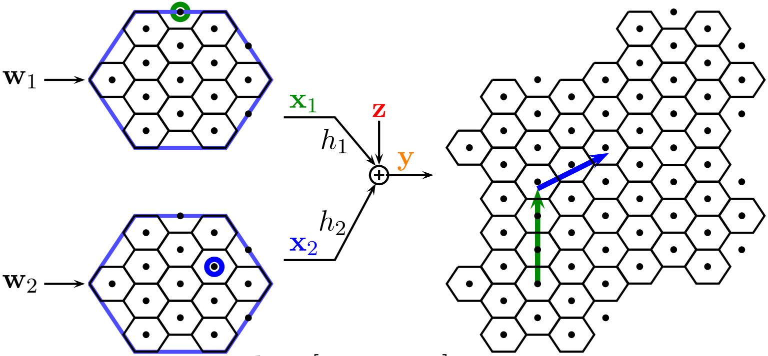


$$\mathbf{h} = [\ 1.4 \quad 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Computation Coding

Scaled codewords added together plus noise:

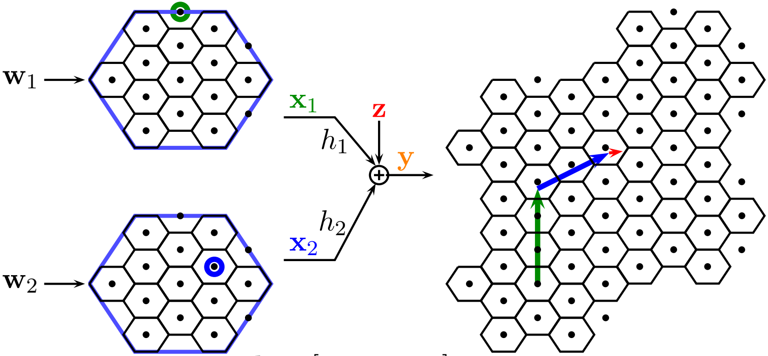


$$\mathbf{h} = [\ 1.4 \quad 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Computation Coding

Scaled codewords added together plus noise:

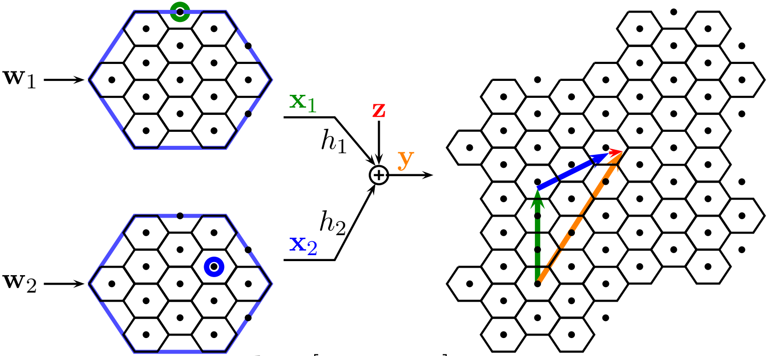


$$\mathbf{h} = [\ 1.4 \quad 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Computation Coding

Extra noise penalty for non-integer channel coefficients:

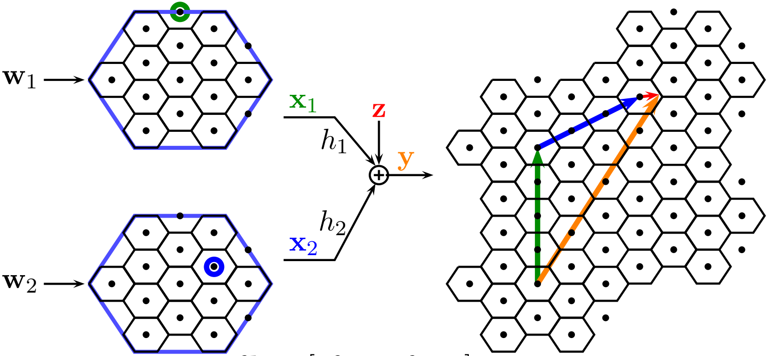


$$\mathbf{h} = [\ 1.4 \quad 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Extra noise: $\text{SNR}\|\mathbf{h} - \mathbf{a}\|^2$

Scale output by β to reduce non-integer noise penalty:

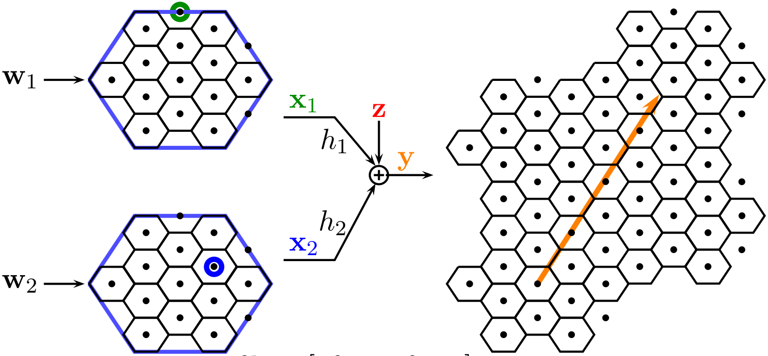


$$\beta\mathbf{h} = [\ \beta 1.4 \quad \beta 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Extra noise: $\mathsf{SNR}\|\beta\mathbf{h} - \mathbf{a}\|^2$

Scale output by β to reduce non-integer noise penalty:

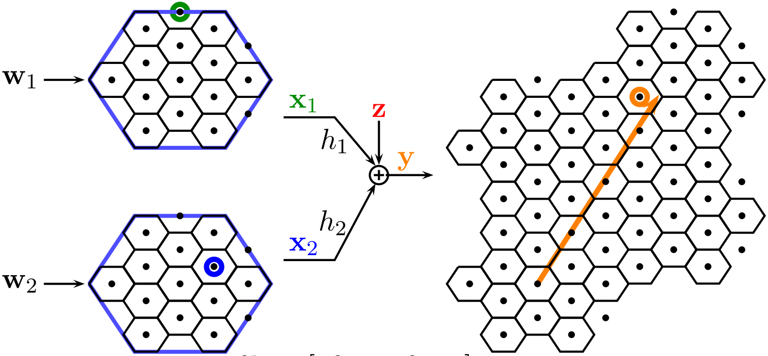


$$\beta\mathbf{h} = [\ \beta 1.4 \quad \beta 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Extra noise: $\text{SNR}\|\beta\mathbf{h} - \mathbf{a}\|^2$

Decode to closest lattice point:

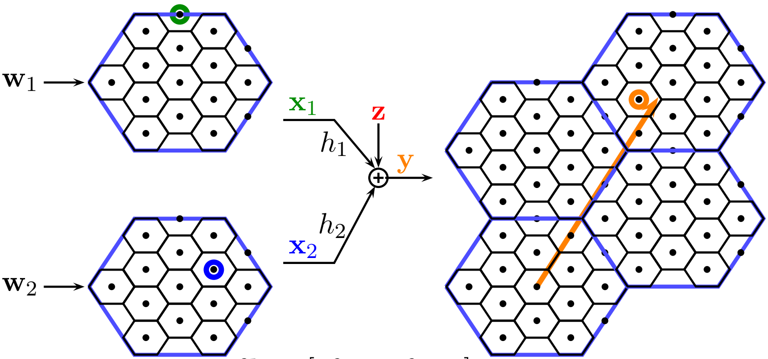


$$\beta\mathbf{h} = [\ \beta 1.4 \quad \beta 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Extra noise: $\mathsf{SNR}\|\beta\mathbf{h} - \mathbf{a}\|^2$

Compute sum of lattice points modulo the coarse lattice:



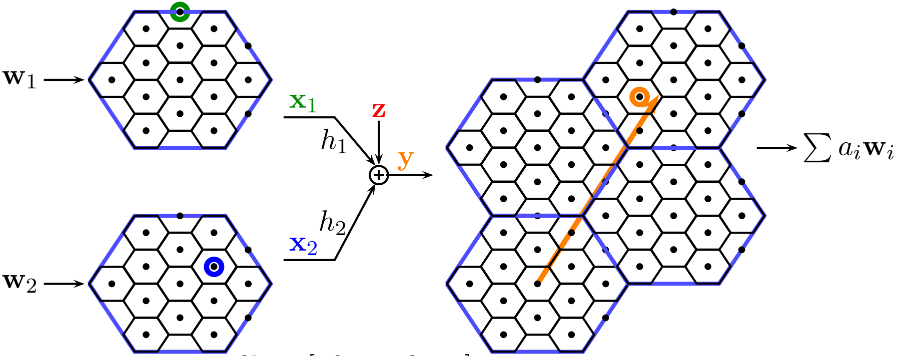
$$\beta\mathbf{h} = [\ \beta 1.4 \quad \beta 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Extra noise: $\mathsf{SNR}\|\beta\mathbf{h} - \mathbf{a}\|^2$

Computation Coding

Map back to equation of message symbols over the field:



$$\beta\mathbf{h} = [\ \beta 1.4 \quad \beta 2.1\ ]$$

$$\mathbf{a} = [\ 2 \quad 3\ ]$$

Extra noise: $\mathsf{SNR}\|\beta\mathbf{h} - \mathbf{a}\|^2$

Achievable Rates

Theorem (Nazer-Gastpar ISIT '08, Asilomar '08)

For channel coefficients $\mathbf{h}$ *and equation coefficients* $\mathbf{a}$*, a receiver can decode* $\sum a_i \mathbf{w}_i$ *at rate:*

$$R = \max_{\beta \in \mathbb{R}} \frac{1}{2} \log \left(\frac{\mathsf{SNR}}{|\beta|^2 + \mathsf{SNR} \|\beta \mathbf{h} - \mathbf{a}\|^2} \right)$$

- Plugging in $\mathbf{a} = [0 \cdots 0 \ 1 \ 0 \cdots 0]$ recovers bit pipe rates.

Achievable Rates

Theorem (Nazer-Gastpar ISIT '08, Asilomar '08)

For channel coefficients $\mathbf{h}$ *and equation coefficients* $\mathbf{a}$*, a receiver can decode* $\sum a_i \mathbf{w}_i$ *at rate:*

$$\begin{aligned} R &= \max_{\beta \in \mathbb{R}} \frac{1}{2} \log \left(\frac{\mathsf{SNR}}{|\beta|^2 + \mathsf{SNR} \|\beta \mathbf{h} - \mathbf{a}\|^2} \right) \\ &= \frac{1}{2} \log \left(\frac{1}{\|\mathbf{a}\|^2 - \beta_{MMSE} < \mathbf{a}, \mathbf{h} >} \right) \end{aligned}$$

- The optimal choice of β is always given by the MMSE coefficient:

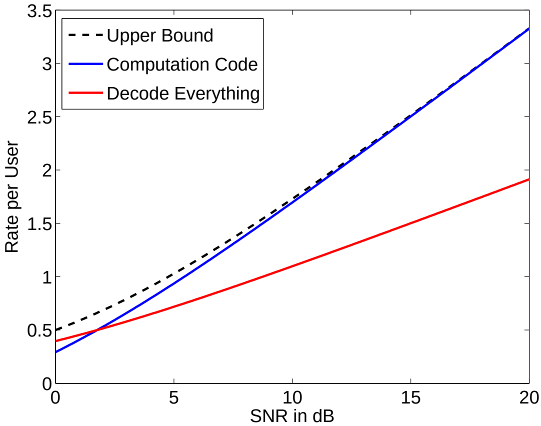
$$\beta_{\mathsf{MMSE}} = \frac{\mathsf{SNR} < \mathbf{h}, \mathbf{a} >}{1 + \mathsf{SNR} \|\mathbf{h}\|^2}$$

- Plugging in $\mathbf{a} = [0 \cdots 0 \ 1 \ 0 \cdots 0]$ recovers bit pipe rates.

Example: Recovering the Sum

- Want sum of messages $\sum_{i=1}^{M} \mathbf{w}_i$
- Channel is perfectly matched $\mathbf{y} = \sum_{i=1}^{M} \mathbf{x}_i + \mathbf{z}$

$$M = 2$$



$$R_{\mathsf{UPPER}} = \frac{1}{2}\log\left(1 + \mathsf{SNR}\right)$$

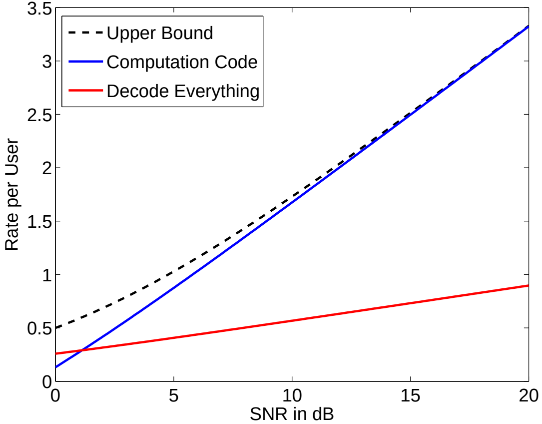
$$R_{\mathsf{COMP}} = \frac{1}{2}\log\left(\frac{1}{M} + M\mathsf{SNR}\right)$$

$$R_{\mathsf{RAND}} = \frac{1}{2M}\log\left(1 + M\mathsf{SNR}\right)$$

Example: Recovering the Sum

- Want sum of messages $\sum_{i=1}^{M} \mathbf{w}_i$
- Channel is perfectly matched $\mathbf{y} = \sum_{i=1}^{M} \mathbf{x}_i + \mathbf{z}$

$$M = 5$$



$$R_{\mathsf{UPPER}} = \frac{1}{2} \log\left(1 + \mathsf{SNR}\right)$$

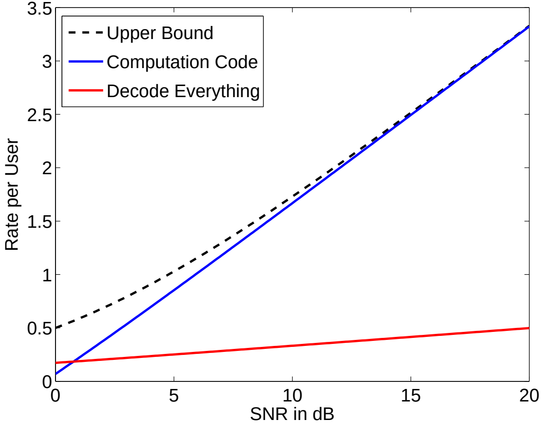
$$R_{\mathsf{COMP}} = \frac{1}{2} \log\left(\frac{1}{M} + M\mathsf{SNR}\right)$$

$$R_{\mathsf{RAND}} = \frac{1}{2M} \log\left(1 + M\mathsf{SNR}\right)$$

Example: Recovering the Sum

- Want sum of messages $\sum_{i=1}^{M} \mathbf{w}_i$
- Channel is perfectly matched $\mathbf{y} = \sum_{i=1}^{M} \mathbf{x}_i + \mathbf{z}$

$$M = 10$$



$$R_{\text{UPPER}} = \frac{1}{2}\log\left(1 + \text{SNR}\right)$$

$$R_{\text{COMP}} = \frac{1}{2}\log\left(\frac{1}{M} + M\text{SNR}\right)$$

$$R_{\text{RAND}} = \frac{1}{2M}\log\left(1 + M\text{SNR}\right)$$

Decode the Best Equation

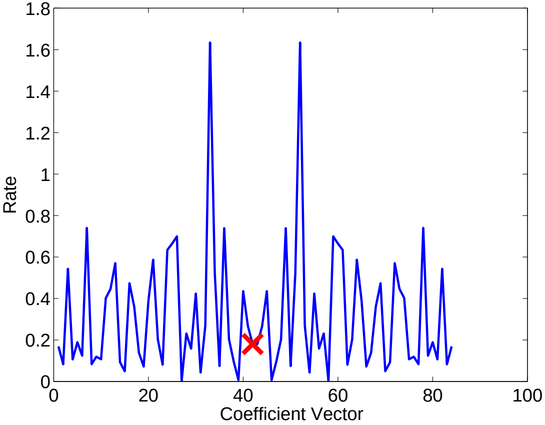
- Receiver chooses equation coefficients $\mathbf{a}$ that maximize the rate:

$$R = \max_{\mathbf{a} \in \mathbb{Z}^M} \frac{1}{2} \log \left(\frac{1}{\|\mathbf{a}\|^2 - \beta_{\mathsf{MMSE}} < \mathbf{a}, \mathbf{h} >} \right)$$

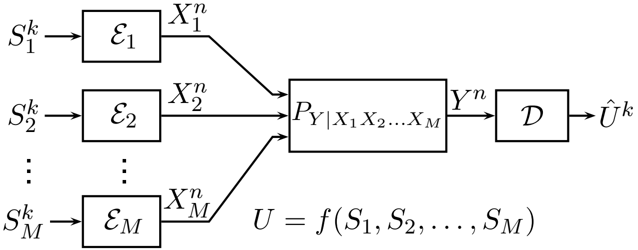
- Only need to check $\mathbf{a}$ satisfying $\|\mathbf{a}\|^2 \leq 1 + \|\mathbf{h}\|^2 \mathsf{SNR}$

Example: Equation Rates

- 4 users
- Bit Pipe Rate $= 0.1807$
- Maximum Computation Rate $= 1.6343$
- Equation Coefficients $= [1 \quad -1 \quad 0 \quad 1]$
- Channel $=$ $[0.74 \;\; -1.06 \;\; -0.16 \;\; 0.88]$



Computation over Multiple-Access Channels

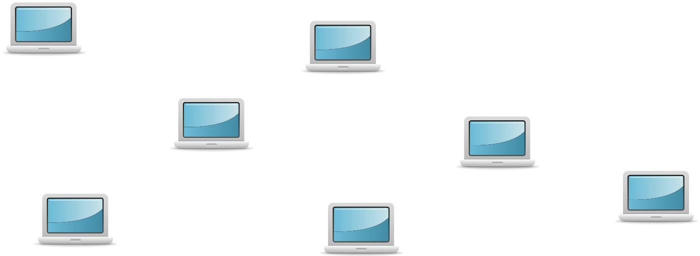


- Goal: maximize computation rate, functions evaluated per channel use
- **Nazer-Gastpar IT '07**: partial results for general functions and channels, computation capacity for finite field models

Talk Overview

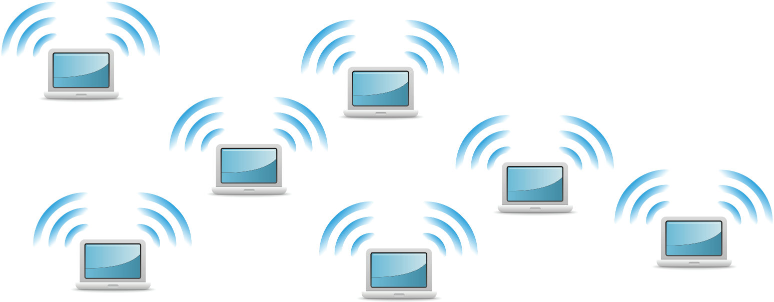
1. How can we (reliably) compute over noisy channels?
2. What does this mean for wireless networks?
3. Beyond bits: Distributed signal processing applications.

Wireless Network



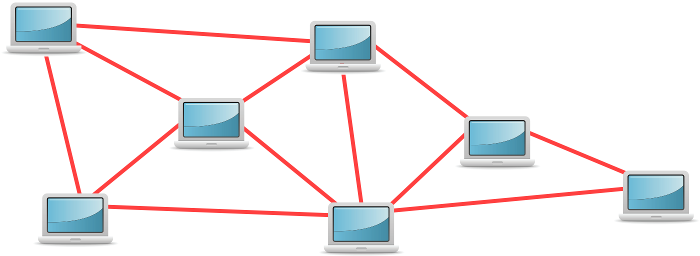
- Usually fight interference and convert to network of bit pipes.

Wireless Network



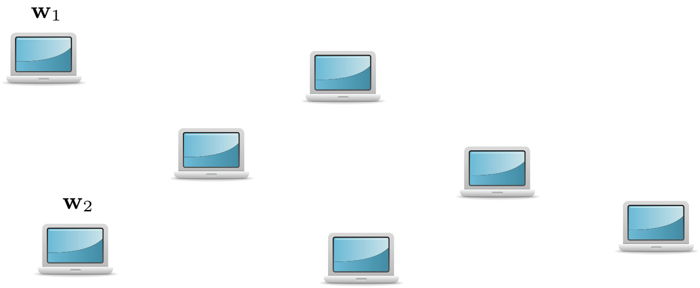
- Usually fight interference and convert to network of bit pipes.

Wireless Network



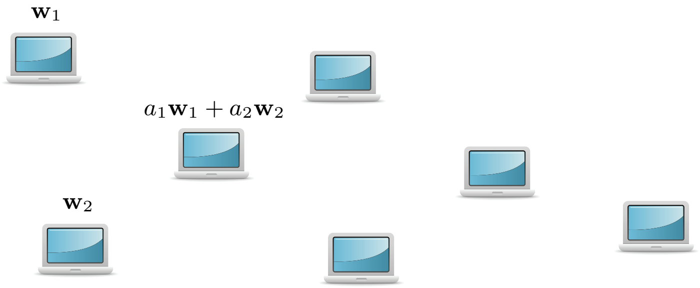
- Usually fight interference and convert to network of bit pipes.

Wireless Network



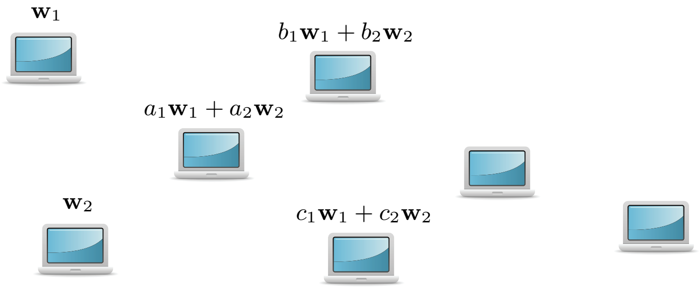
- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.

Wireless Network



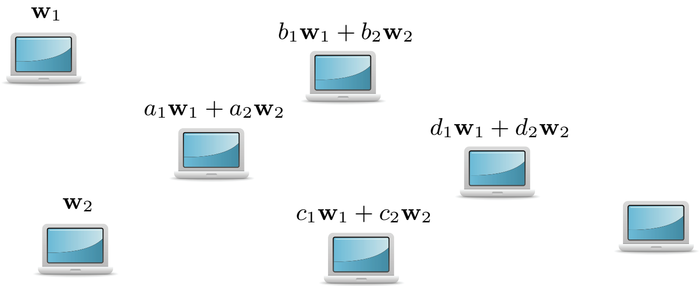
- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.

Wireless Network



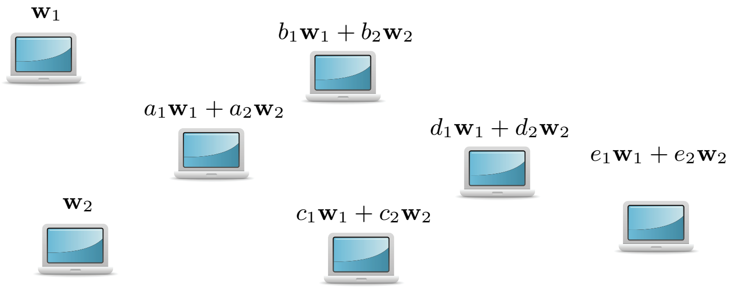
- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.

Wireless Network



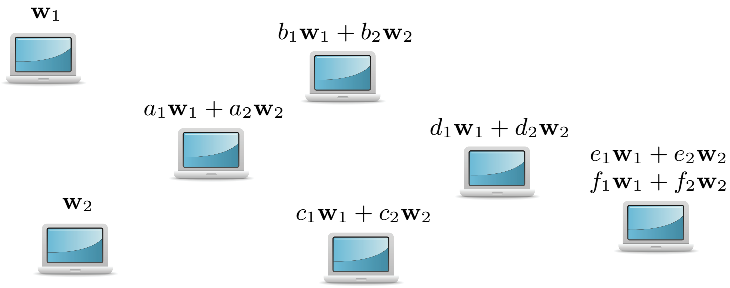
- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.

Wireless Network



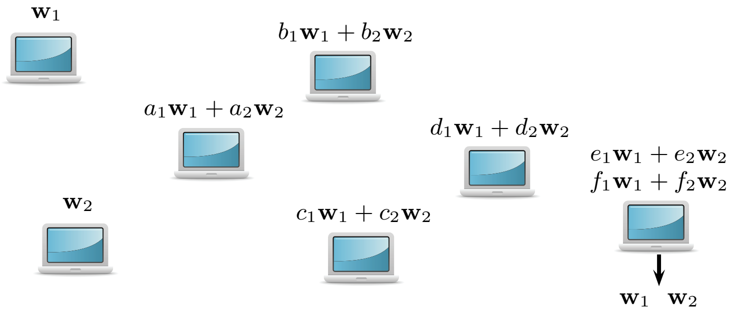
- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.
- Physical-layer interface for network coding: collect equations and solve for desired messages.

Wireless Network



- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.
- Physical-layer interface for network coding: collect equations and solve for desired messages.

Wireless Network

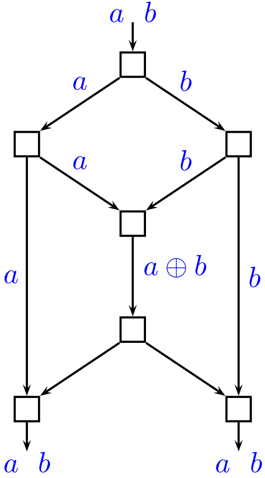


- Usually fight interference and convert to network of bit pipes.
- Compute-and-Forward: Users decode linear combinations of messages according to fading coefficients.
- Physical-layer interface for network coding: collect equations and solve for desired messages.

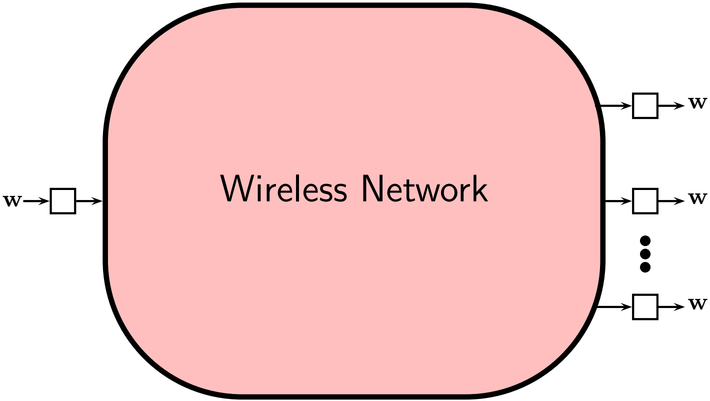
Network Coding for Wired Networks

Ahlswede-Cai-Li-Yeung '00: Network coding achieves the multicast capacity of wired networks. Routing is suboptimal.

- Example: Butterfly Network
- Want to multicast two packets: a and b.
- Mixing packets (network coding) is optimal.
- Send mod-2 sum down center path.

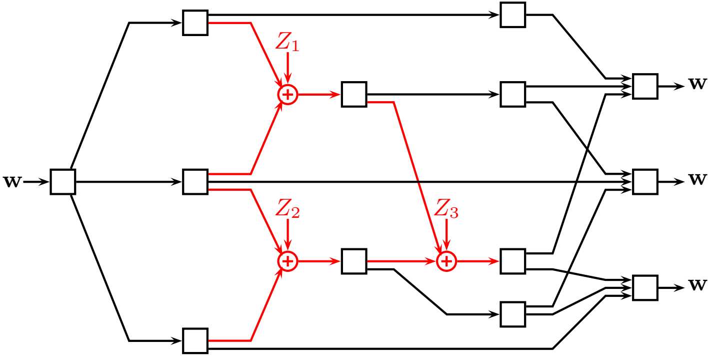


Network Coding for Wireless Networks



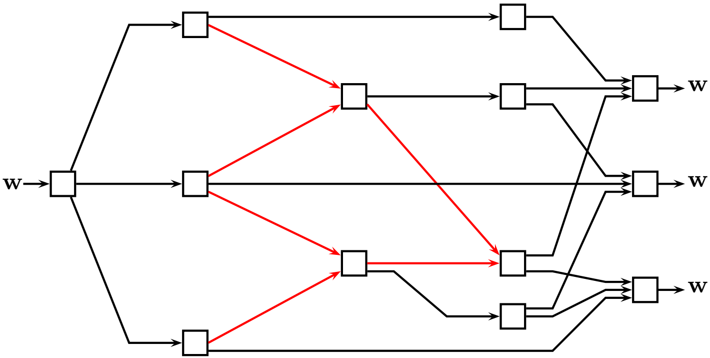
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



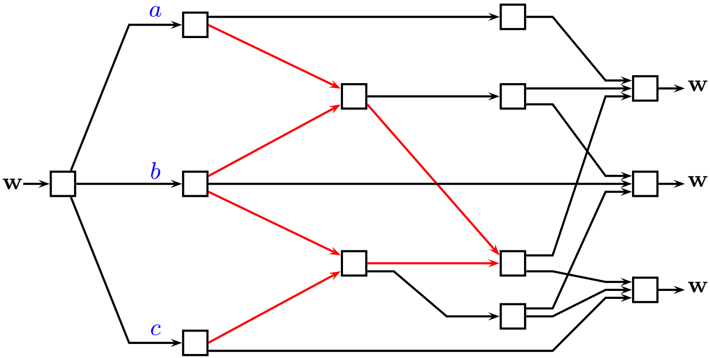
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



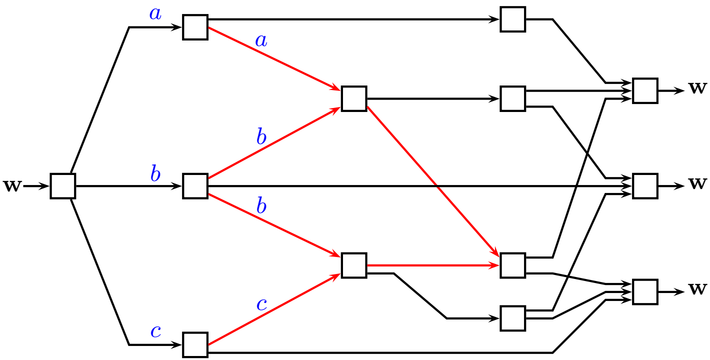
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



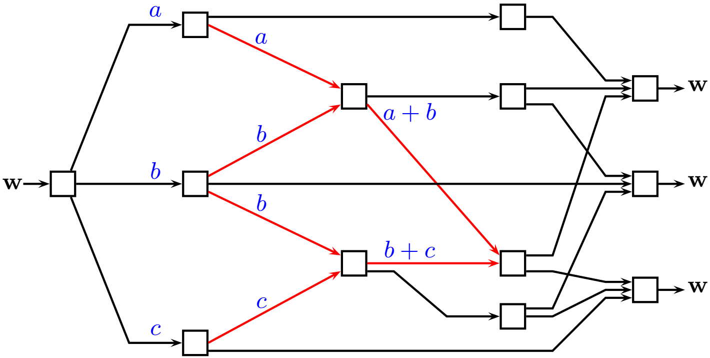
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



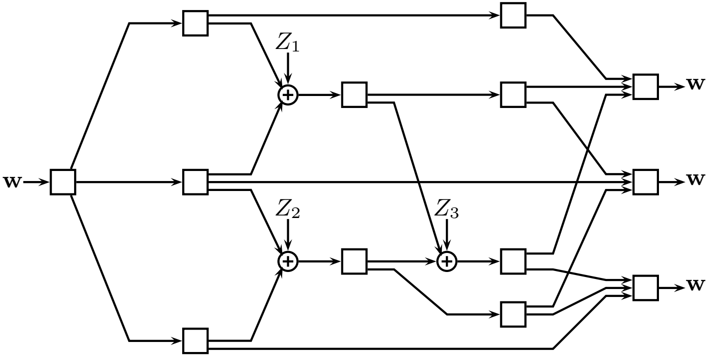
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



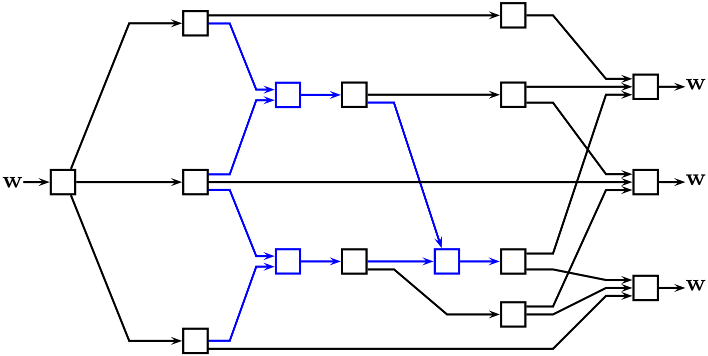
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Network Coding for Wireless Networks



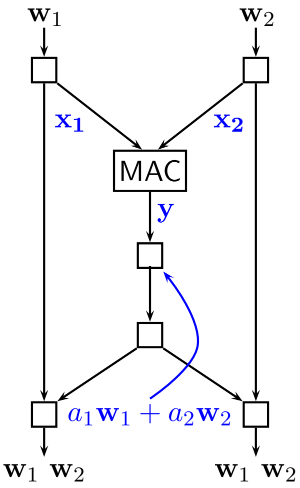
- Bit Pipe Approach: Decode incoming packets. Transmit random linear combination.
- Compute-and-Forward: Channel does network coding for us.

Example: Wireless Network Coding

Butterfly with a multiple-access channel in the middle:

$$\mathbf{y} = h_1\mathbf{x}_1 + h_2\mathbf{x}_2 + \mathbf{z}$$

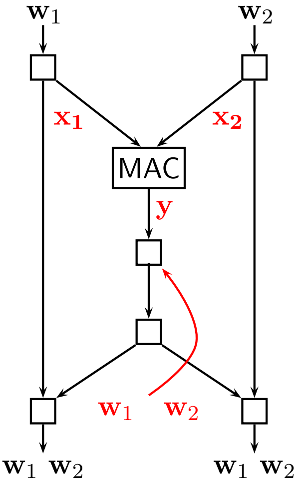
- Compute-and-Forward: Decode any equation with non-zero coefficients.



Butterfly with a multiple-access channel in the middle:

$$\mathbf{y} = h_1\mathbf{x}_1 + h_2\mathbf{x}_2 + \mathbf{z}$$

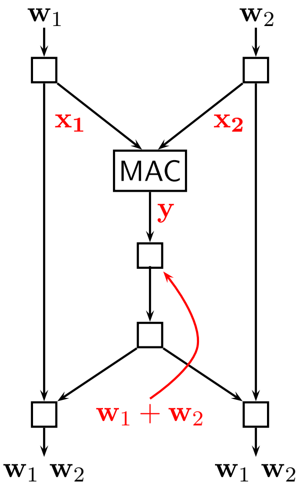
- Compute-and-Forward: Decode any equation with non-zero coefficients.
- Bit Pipe: Decode *both messages* then compute the sum.



Butterfly with a multiple-access channel in the middle:

$$\mathbf{y} = h_1\mathbf{x}_1 + h_2\mathbf{x}_2 + \mathbf{z}$$

- Compute-and-Forward: Decode any equation with non-zero coefficients.
- Bit Pipe: Decode *both messages* then compute the sum.

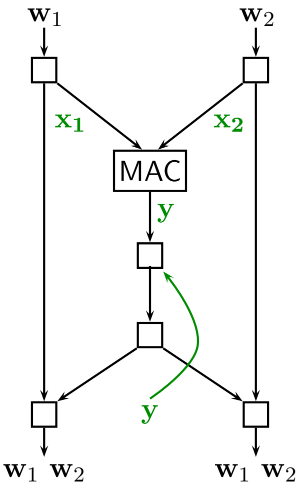


Example: Wireless Network Coding

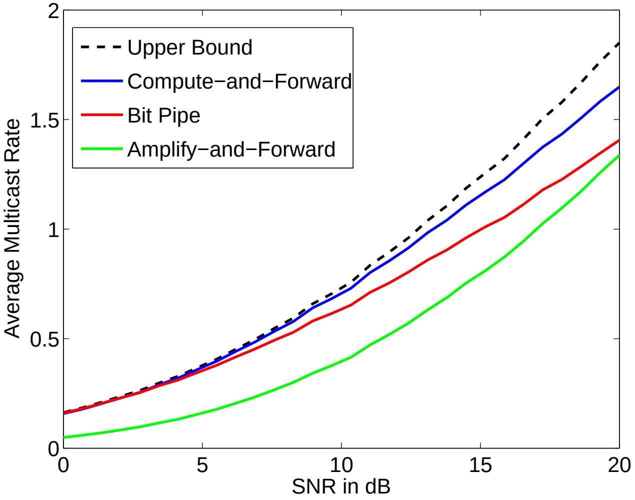
Butterfly with a multiple-access channel in the middle:

$$\mathbf{y} = h_1\mathbf{x}_1 + h_2\mathbf{x}_2 + \mathbf{z}$$

- Compute-and-Forward: Decode any equation with non-zero coefficients.
- Bit Pipe: Decode *both messages* then compute the sum.
- Amplify-and-Forward: Retransmit channel observation.
  (**Katti-Gollakota-Katabi '07**)



Example: Wireless Network Coding



Beyond Multicast

- Multicasting requires every user to recover every message. Need full rank set of equations.

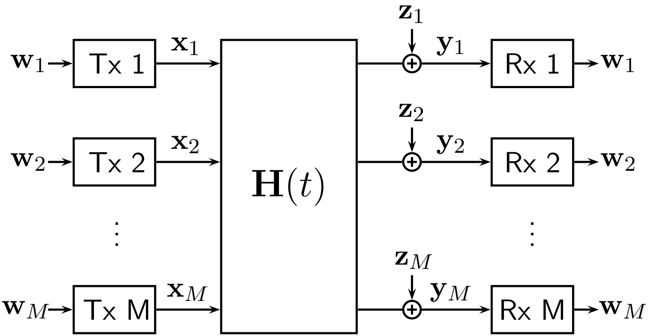
$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1M} \\ a_{21} & a_{22} & \cdots & a_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ a_{M1} & a_{M2} & \cdots & a_{MM} \end{bmatrix}$$

- May just want one message at a destination. Need fewer equations.

$$\begin{bmatrix} b_1 & b_2 & \cdots & b_M \\ b_1 & -b_2 & \cdots & -b_M \end{bmatrix}$$

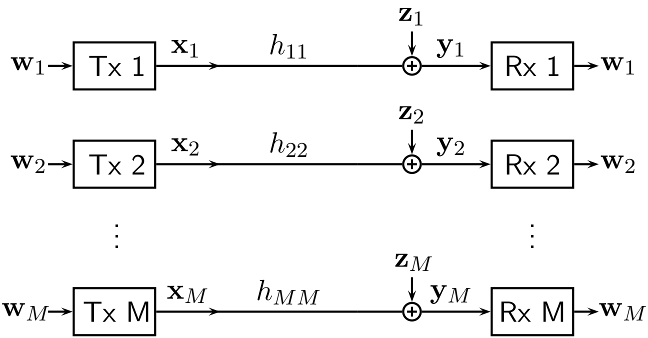
M-User Fast Fading Interference Channel

- Time-varying fading with i.i.d. uniform phases.
- Transmitters know $\mathbf{H}(t)$ before time t.



M-User Fast Fading Interference Channel

- Time-varying fading with i.i.d. uniform phases.
- Transmitters know $\mathbf{H}(t)$ before time t.

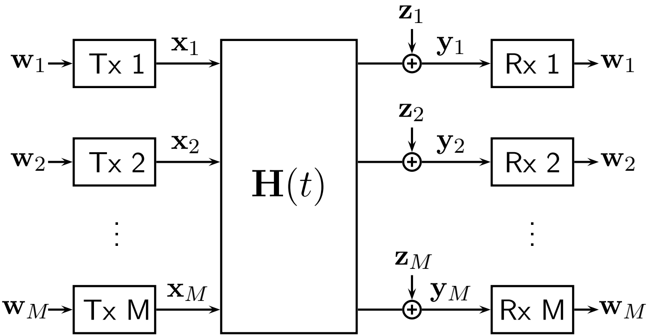


- Interference-free rate:

$$R_{\text{FREE}} = E\left[\log\left(1 + |h_{mm}|^2 \mathsf{SNR}_m\right)\right]$$

M-User Fast Fading Interference Channel

- Time-varying fading with i.i.d. uniform phases.
- Transmitters know $\mathbf{H}(t)$ before time t.



- Interference-free rate:

$$R_{\text{FREE}} = E\left[\log\left(1 + |h_{mm}|^2 \text{SNR}_m\right)\right]$$

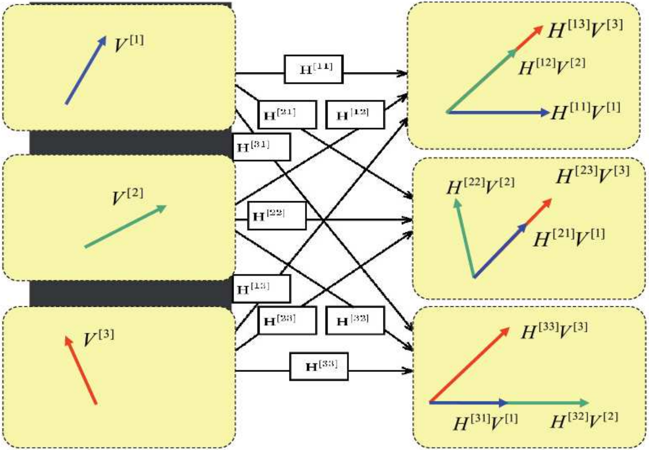
- Time-division:

$$R_{\text{TDMA}} = \frac{1}{M} E\left[\log\left(1 + M|h_{mm}|^2 \text{SNR}_m\right)\right]$$

Interference Alignment

Cadambe-Jafar '08: With careful choice of precoding matrices, each user can get "half the cake" (at high SNR):

$$\lim_{\mathsf{SNR}\to\infty} \frac{R_{IA}}{\log(1+\mathsf{SNR})} = \frac{1}{2}$$



Ergodic Interference Alignment

Nazer-Gastpar-Jafar-Vishwanath ISIT '09:

1. Send chunk of bits at time t with channel matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ h_{21} & h_{22} & \cdots & h_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ h_{M1} & h_{M2} & \cdots & h_{MM} \end{bmatrix}$$

Ergodic Interference Alignment

Nazer-Gastpar-Jafar-Vishwanath ISIT '09:

1. Send chunk of bits at time t with channel matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ h_{21} & h_{22} & \cdots & h_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ h_{M1} & h_{M2} & \cdots & h_{MM} \end{bmatrix}$$

2. Send same chunk of bits when complementary matrix $\mathbf{H}_C$ occurs:

$$\mathbf{H}_C = \begin{bmatrix} h_{11} & -h_{12} & \cdots & -h_{1M} \\ -h_{21} & h_{22} & \cdots & -h_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ -h_{M1} & -h_{M2} & \cdots & h_{MM} \end{bmatrix} \pm \delta$$

Ergodic Interference Alignment

Nazer-Gastpar-Jafar-Vishwanath ISIT '09:

1. Send chunk of bits at time t with channel matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1M} \\ h_{21} & h_{22} & \cdots & h_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ h_{M1} & h_{M2} & \cdots & h_{MM} \end{bmatrix}$$

2. Send same chunk of bits when complementary matrix $\mathbf{H}_C$ occurs:

$$\mathbf{H}_C = \begin{bmatrix} h_{11} & -h_{12} & \cdots & -h_{1M} \\ -h_{21} & h_{22} & \cdots & -h_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ -h_{M1} & -h_{M2} & \cdots & h_{MM} \end{bmatrix} \pm \delta$$

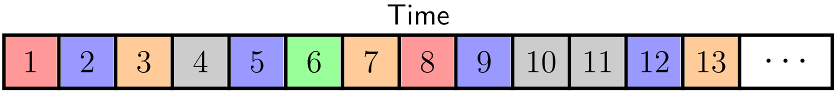
3. Otherwise, send a different chunk of bits (and wait for its $\mathbf{H}_C$ too).

Ergodic Interference Alignment

Sum of channel observations is interference-free:

$$\mathbf{H} + \mathbf{H}_C = \begin{bmatrix} 2h_{11} & & 0 \\ & \ddots & \\ 0 & & 2h_{MM} \end{bmatrix} \pm \delta$$

Choose block length large enough so that sequence of channel matrices converges in type. Then, most channel matrices will have a match.



Interference Channel Ergodic Capacity

Theorem (Nazer-Gastpar-Jafar-Vishwanath ISIT '09)

Each user can achieve at least half its interference-free capacity at **any signal-to-noise ratio***:*

$$R = \frac{1}{2} E\left[\log\left(1 + 2|h_{mm}|^2 \mathsf{SNR}_m\right)\right] > \frac{1}{2} R_{\textit{FREE}}$$

- **Jafar '09:** For uniform phase fading and a large number of users, scheme achieves the ergodic capacity.
- Can also show this approach achieves the ergodic capacity region for finite field channel models.
- Does not meet outer bound in general. For example, can do slightly better in Rayleigh channels by including a "strong interference" mode.

Structured Codes Help in Networks

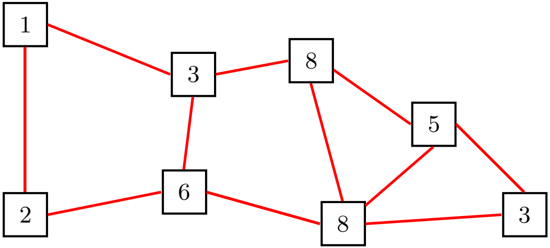
Recent research shows that structured codes are needed to approach the capacity of networks.

- Distributed MIMO (**Nazer-Gastpar '08**)
- Distributed Source Coding (**Krithivasan-Pradhan '08**)
- Distributed Function Compression (**Körner-Marton '79, Krithivasan-Pradhan '07, Wagner '08**)
- Two-Way Relay Channel (**Wilson-Narayanan-Pfister-Sprintson '07, '08, Nam-Chung-Lee '08**)
- Dirty Multiple-Access Channel (**Philosof-Khisti-Erez-Zamir '07)**
- Interference Channels (**Bresler-Parekh-Tse '07, Sridharan-Jafarian-Vishwanath-Jafar-Shamai '08**)
- Secrecy (**He-Yener '08, '09**)

Talk Overview

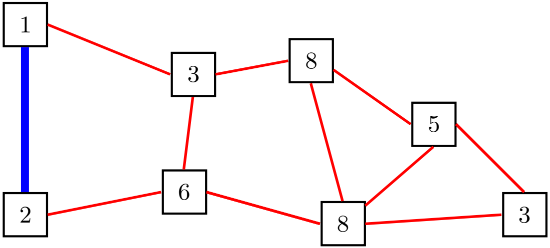
1. How can we (reliably) compute over noisy channels?
2. What does this mean for wireless networks?
3. Beyond bits: Distributed signal processing applications.

Gossip Algorithms



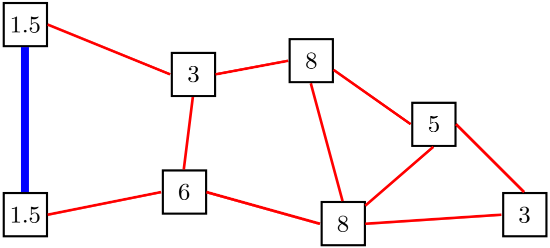
- A node randomly wakes up and computes a local average with a neighbor.

Gossip Algorithms



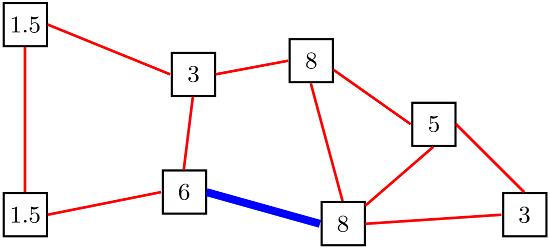
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.

Gossip Algorithms



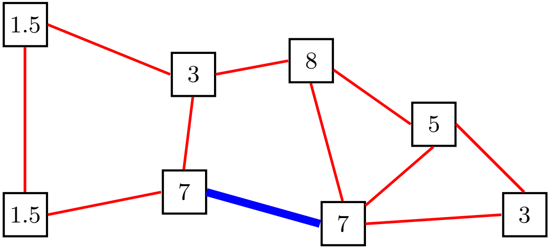
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.
- Pairwise gossip well-studied for general graphs by **Boyd-Ghosh-Prabhakar-Shah '06**.

Gossip Algorithms



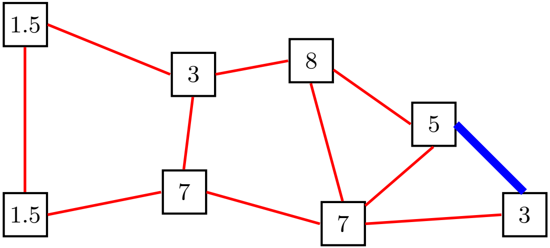
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.
- Pairwise gossip well-studied for general graphs by **Boyd-Ghosh-Prabhakar-Shah '06**.

Gossip Algorithms



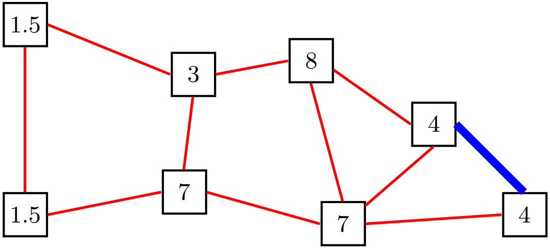
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.
- Pairwise gossip well-studied for general graphs by **Boyd-Ghosh-Prabhakar-Shah '06**.

Gossip Algorithms



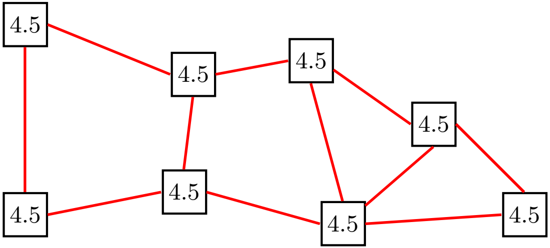
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.
- Pairwise gossip well-studied for general graphs by **Boyd-Ghosh-Prabhakar-Shah '06**.

Gossip Algorithms



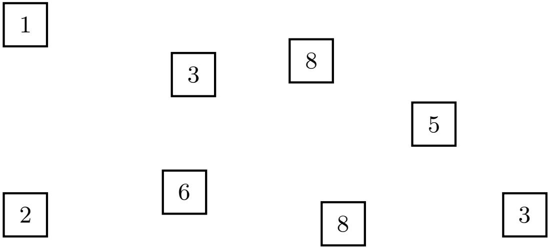
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.
- Pairwise gossip well-studied for general graphs by **Boyd-Ghosh-Prabhakar-Shah '06**.

Gossip Algorithms



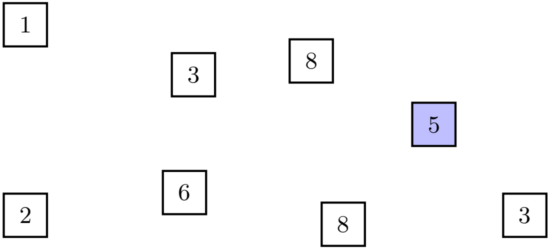
- A node randomly wakes up and computes a local average with a neighbor.
- After enough rounds, every node knows the global average.
- Pairwise gossip well-studied for general graphs by **Boyd-Ghosh-Prabhakar-Shah '06**.

Gossip Algorithms



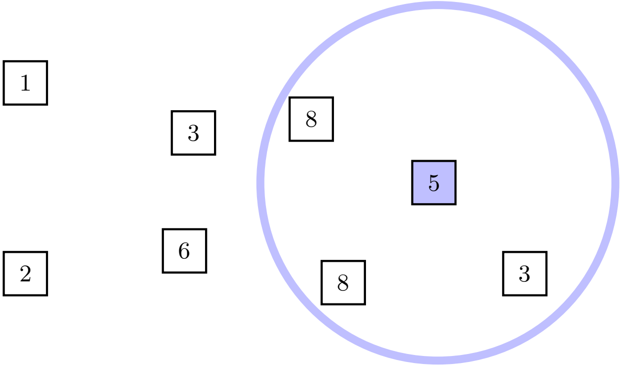
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



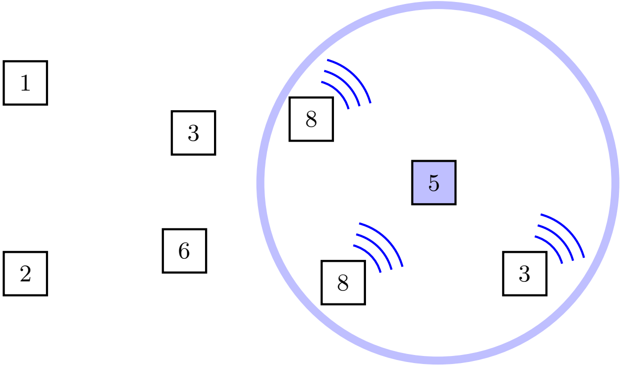
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



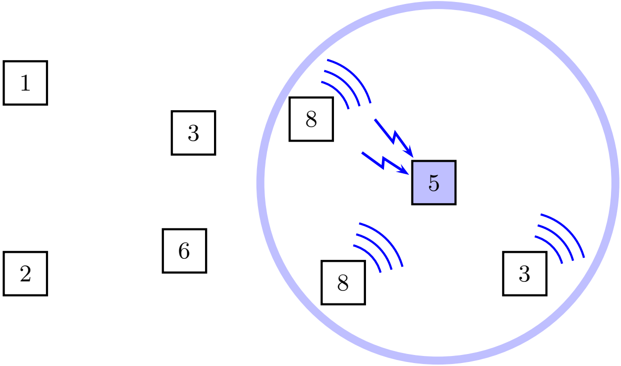
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



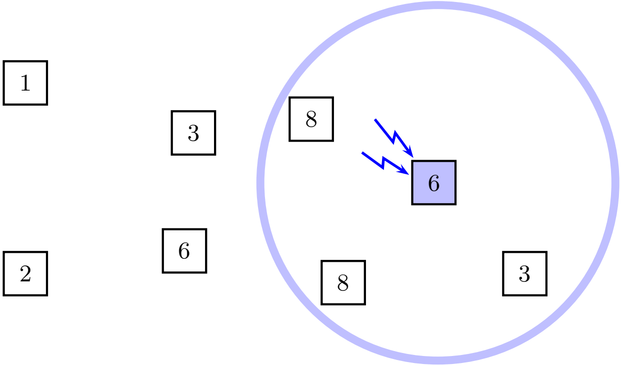
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



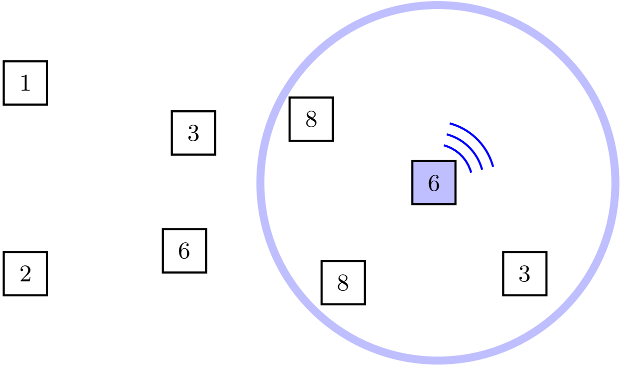
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



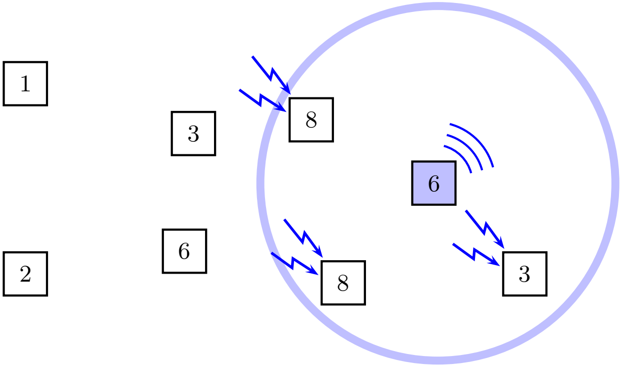
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



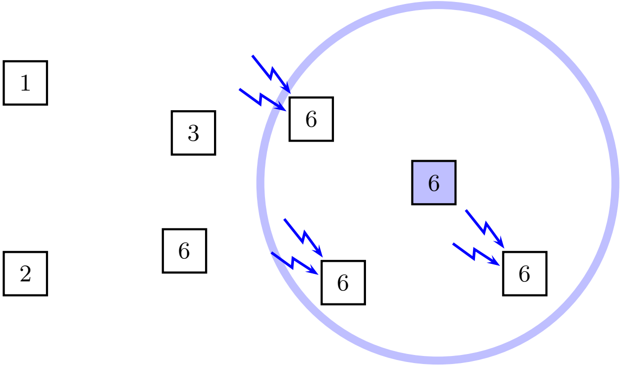
- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms



- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Gossip Algorithms

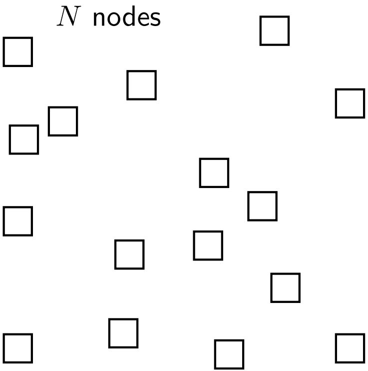


- Usually, we design algorithms to minimize the number of bits transmitted.
- With computation codes, we can instead minimize the number of equations computed.

Problem Statement

- N nodes randomly placed in a square
- Each node has a real-valued measurement
- AWGN multiple-access channel model:

$$Y_\ell = \sum r_{m\ell}^{-\alpha/2} \phi_{m\ell} X_m + Z_\ell$$

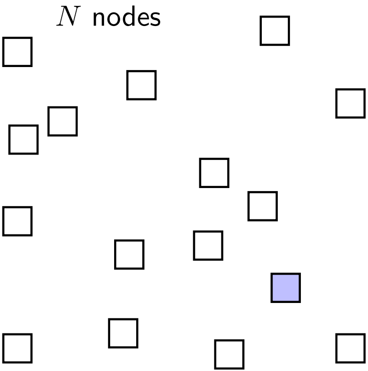


- Want all nodes to learn the global average.

Problem Statement

- N nodes randomly placed in a square
- Each node has a real-valued measurement
- AWGN multiple-access channel model:

$$Y_\ell = \sum r_{m\ell}^{-\alpha/2} \phi_{m\ell} X_m + Z_\ell$$

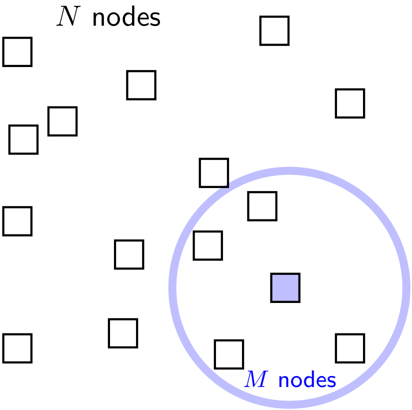


- Want all nodes to learn the global average.

Problem Statement

- N nodes randomly placed in a square
- Each node has a real-valued measurement
- AWGN multiple-access channel model:

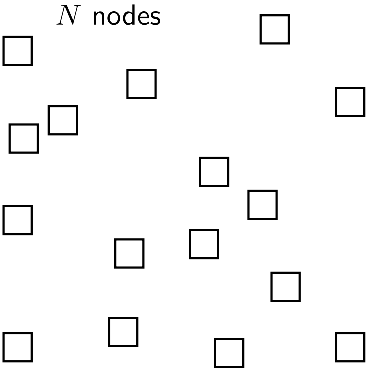
$$Y_\ell = \sum r_{m\ell}^{-\alpha/2} \phi_{m\ell} X_m + Z_\ell$$



- Want all nodes to learn the global average.
- Channel knowledge, $r_{m\ell}, \phi_{m\ell}$, available about size M local neighborhood.

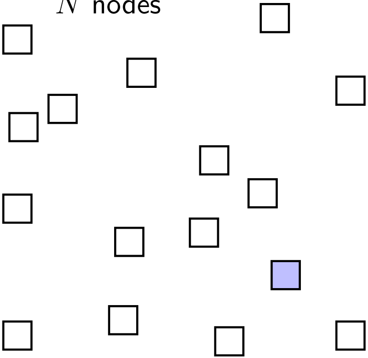
Pairwise Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with a random neighbor.
- Random geometric graph
- **Boyd-Ghosh-Prabhakar-Shah '06**: Converges in $\Theta(N^2)$ rounds.
- Comes from spectral gap of $\bar{W}$, the averaging matrix.



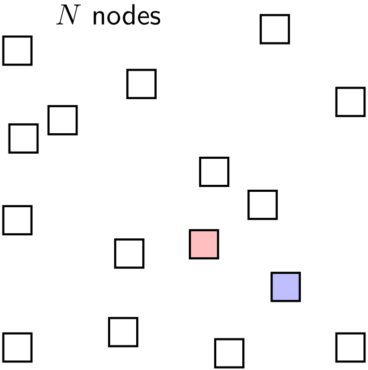
Pairwise Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with a random neighbor.
- Random geometric graph
- **Boyd-Ghosh-Prabhakar-Shah '06**: Converges in $\Theta(N^2)$ rounds.
- Comes from spectral gap of $\bar{W}$, the averaging matrix.



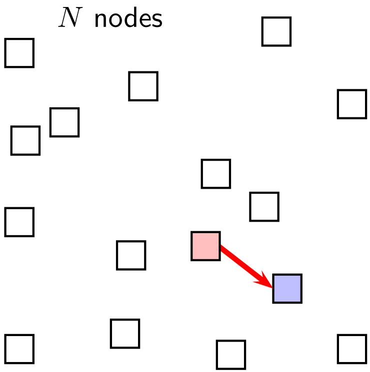
Pairwise Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with a random neighbor.
- Random geometric graph
- **Boyd-Ghosh-Prabhakar-Shah '06**: Converges in $\Theta(N^2)$ rounds.
- Comes from spectral gap of $\bar{W}$, the averaging matrix.



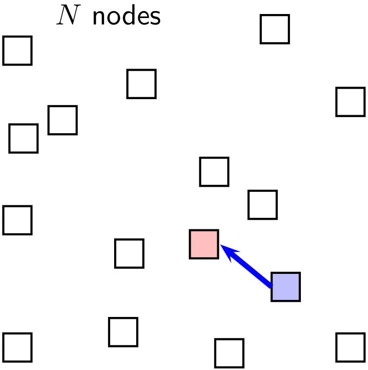
Pairwise Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with a random neighbor.
- Random geometric graph
- **Boyd-Ghosh-Prabhakar-Shah '06**: Converges in $\Theta(N^2)$ rounds.
- Comes from spectral gap of $\bar{W}$, the averaging matrix.



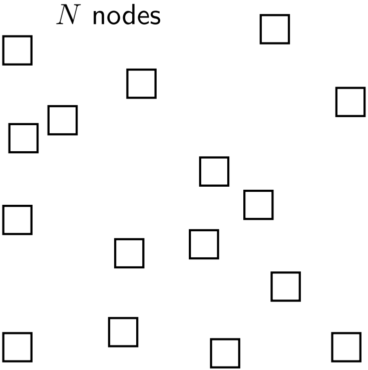
Pairwise Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with a random neighbor.
- Random geometric graph
- **Boyd-Ghosh-Prabhakar-Shah '06**: Converges in $\Theta(N^2)$ rounds.
- Comes from spectral gap of $\bar{W}$, the averaging matrix.



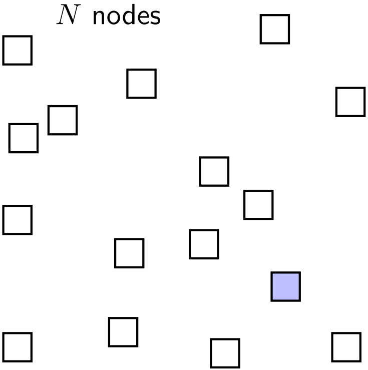
Neighborhood Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with its entire neighborhood.
- Averaging matrix $\bar{W}$ difficult to compute.
- Idea: Lower bound conductance of a related Markov chain.



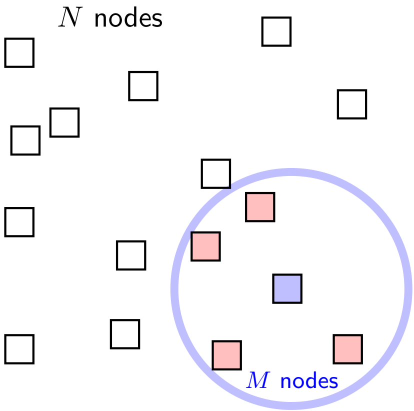
Neighborhood Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with its entire neighborhood.
- Averaging matrix $\bar{W}$ difficult to compute.
- Idea: Lower bound conductance of a related Markov chain.



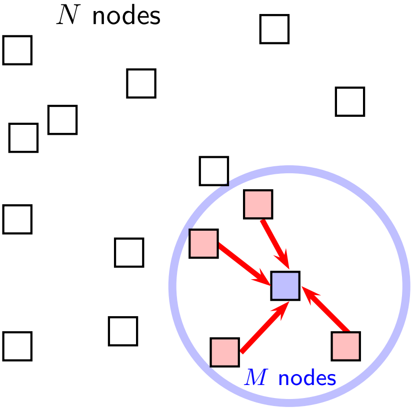
Neighborhood Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with its entire neighborhood.
- Averaging matrix $\bar{W}$ difficult to compute.
- Idea: Lower bound conductance of a related Markov chain.



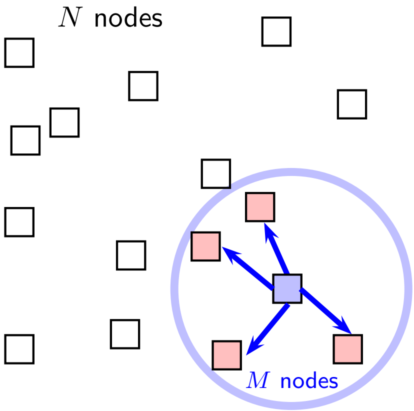
Neighborhood Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with its entire neighborhood.
- Averaging matrix $\bar{W}$ difficult to compute.
- Idea: Lower bound conductance of a related Markov chain.



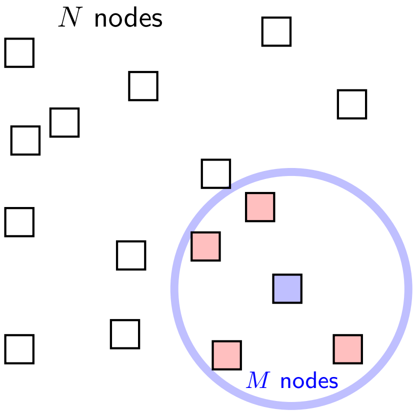
Neighborhood Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with its entire neighborhood.
- Averaging matrix $\bar{W}$ difficult to compute.
- Idea: Lower bound conductance of a related Markov chain.



Neighborhood Gossip: Convergence

- Gossip Round: a node randomly wakes up and averages with its entire neighborhood.
- Averaging matrix $\bar{W}$ difficult to compute.
- Idea: Lower bound conductance of a related Markov chain.



Theorem (Nazer-Dimakis-Gastpar ICASSP '09)

All nodes converge to the global average in $O\left(\frac{N^2}{M^2}\right)$ *rounds.*

Energy Analysis

- Want to compare how much energy each scheme uses to converge in time T.
- Measured in total transmit energy:

$$\text{Total Energy} = \sum_{i=1}^{n} \sum_{t=1}^{T} |x_i(t)|^2$$

- Need to analyze how much energy used for communication per gossip round in pairwise and neighborhood gossip.

Time-Energy Tradeoff

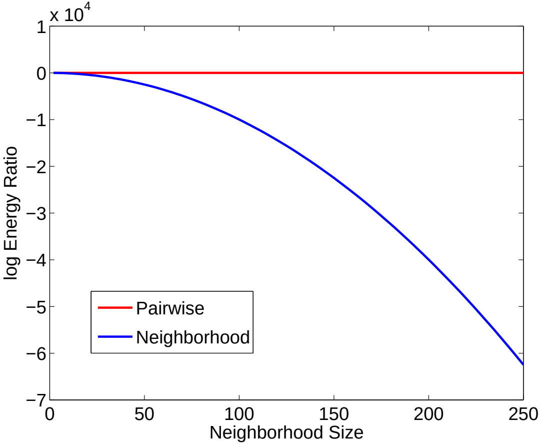
- $N =$ number of nodes
- $M =$ neighborhood size
- $\alpha =$ power path-loss coefficient
- $\tau =$ speed-up factor $= \frac{\text{Pairwise Convergence Time}}{\text{Neighborhood Convergence Time}}$

$$\log \frac{\text{Pairwise Energy}}{\text{Neighborhood Energy}} = \left(\frac{\alpha}{2} + 2\right) \log M - \log \tau - \frac{M^2}{\tau}$$

Exponential energy savings possible if the neighborhood size scales with the network size!

Exponential Energy Savings

- Energy savings increase as the neighborhood size increases.
- Is this a fair comparison?



Making a Fair Comparison

- **Issue 1:** Multiple gossip rounds could happen at once.

Making a Fair Comparison

- **Issue 1:** Multiple gossip rounds could happen at once.
- **Solution 1:** Assume all nodes can pairwise gossip at once.
  Only one neighborhood gossip round at at time.

Making a Fair Comparison

- **Issue 1:** Multiple gossip rounds could happen at once.
- **Solution 1:** Assume all nodes can pairwise gossip at once.
  Only one neighborhood gossip round at at time.
- Penalty 1: N factor slow-down

Making a Fair Comparison

- **Issue 1:** Multiple gossip rounds could happen at once.
- **Solution 1:** Assume all nodes can pairwise gossip at once.
  Only one neighborhood gossip round at at time.
- Penalty 1: N factor slow-down

- **Issue 2:** Quantization error could build up.

Quantization references: **Nedic et al. '07, Frasca et al. '08, Aysal et al. '08, Kar et al. '09**

Making a Fair Comparison

- **Issue 1:** Multiple gossip rounds could happen at once.
- **Solution 1:** Assume all nodes can pairwise gossip at once.
  Only one neighborhood gossip round at at time.
- Penalty 1: N factor slow-down

- **Issue 2:** Quantization error could build up.
- **Solution 2:** Assume no build up for pairwise gossip.
  Assume worst-case build up for neighborhood gossip.

Quantization references: **Nedic et al. '07, Frasca et al. '08, Aysal et al. '08, Kar et al. '09**

Making a Fair Comparison

- **Issue 1:** Multiple gossip rounds could happen at once.
- **Solution 1:** Assume all nodes can pairwise gossip at once.
  Only one neighborhood gossip round at at time.
- Penalty 1: N factor slow-down

- **Issue 2:** Quantization error could build up.
- **Solution 2:** Assume no build up for pairwise gossip.
  Assume worst-case build up for neighborhood gossip.
- Penalty 2: $\log N$ extra quantization bits

Quantization references: **Nedic et al. '07, Frasca et al. '08, Aysal et al. '08, Kar et al. '09**

Time-Energy Tradeoff Revised

- $N =$ number of nodes
- $M =$ neighborhood size
- $\alpha =$ power path-loss coefficient
- $\tau =$ speed-up factor $= \dfrac{\text{Pairwise Convergence Time}}{\text{Neighborhood Convergence Time}}$

$$\log \frac{\text{Pairwise Energy}}{\text{Neighborhood Energy}} = \left(\frac{\alpha}{2} + 2\right) \log M - \log \tau - \frac{M^2}{\tau}$$

Exponential energy savings still possible if the neighborhood is large enough!

Time-Energy Tradeoff Revised

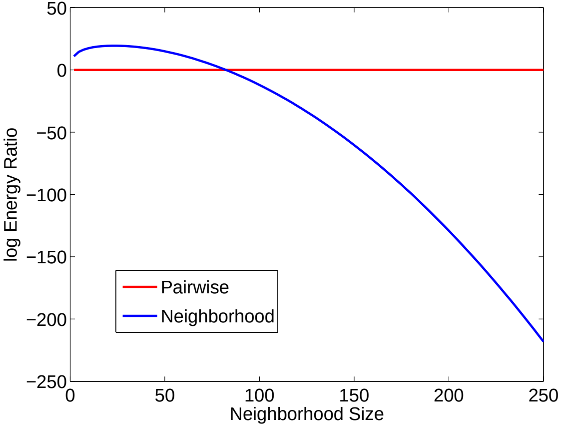
- $N =$ number of nodes
- $M =$ neighborhood size
- $\alpha =$ power path-loss coefficient
- $\tau =$ speed-up factor $= \dfrac{\text{Pairwise Convergence Time}}{\text{Neighborhood Convergence Time}}$

$$\log \frac{\text{Pairwise Energy}}{\text{Neighborhood Energy}} = \left(\frac{\alpha}{2} + 2\right) \log M + \log N - \log \tau - \frac{M^2}{\tau N}$$

Exponential energy savings still possible if the neighborhood is large enough!

Critical Neighborhood Size

Exponential energy savings if the neighborhood is larger than a critical value that depends on the power path-loss coefficient and the speed-up factor.



Conclusions

- Compute-and-Forward: new communication architecture based on equations of bits instead of bits.
- Significant gains are possible since it exploits the noisy linear combinations of the wireless channel.
- Optimal network communication requires both statistical and algebraic considerations.