Distributed Computation in Networks, Part II

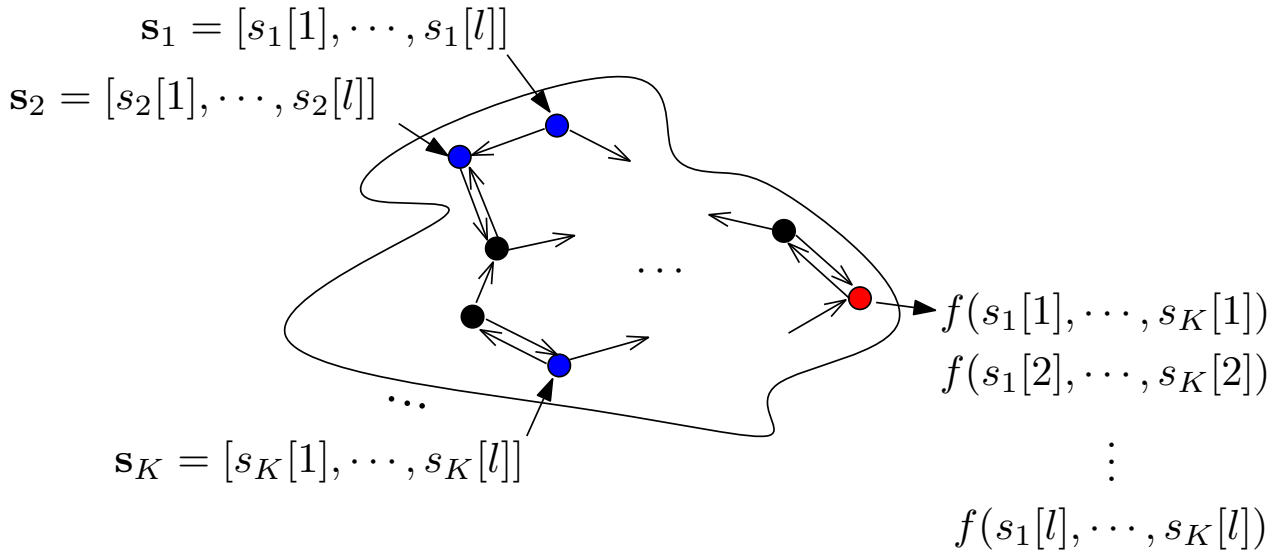
Function Computation in Networks

Michael Gastpar*

EPFL

Based on joint work with Chien-Yi Wang (EPFL), Sang-Woon Jeon (Andong National U.), Changho Suh (KAIST), Naveen Goela (Berkeley/Qualcomm), Bobak Nazer (Boston U.)

Function Computation in Networks



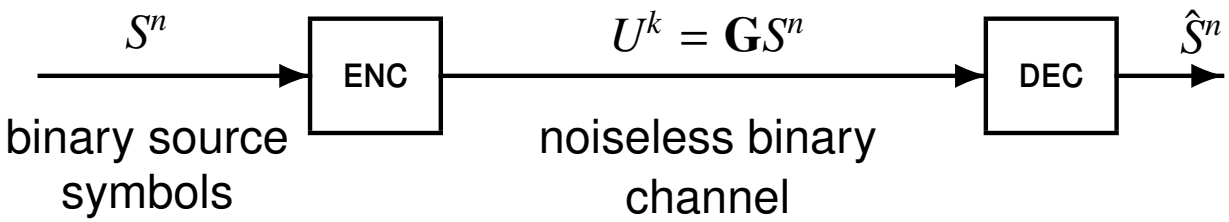
- A directed graph (V, E).
  - There are K source vertices. W.l.o.g., they are the vertices numbered $1, 2, \ldots, K$.
  - Every vertex $v \in V$ has a set of incoming edges $\Gamma_{in}(v)$ and a set of outgoing edges $\Gamma_{out}(v)$.
- Computation rate: $R_{comp} = \frac{l}{n}$ [functions/channel use]
- Generally, it is a difficult problem.
- The answer depends on:
  - Structure of the function $f(\cdots)$
  - Structure of the network.

Function Computation in Networks

Desired function	Networks of point-to-point channels	Networks with multi-access components	Networks with all-to-all interference	...
Type Functions	✓	✓		... ...
Arithmetic sum	✓	✓		... ...
⋮	⋮	⋮	⋮	⋱

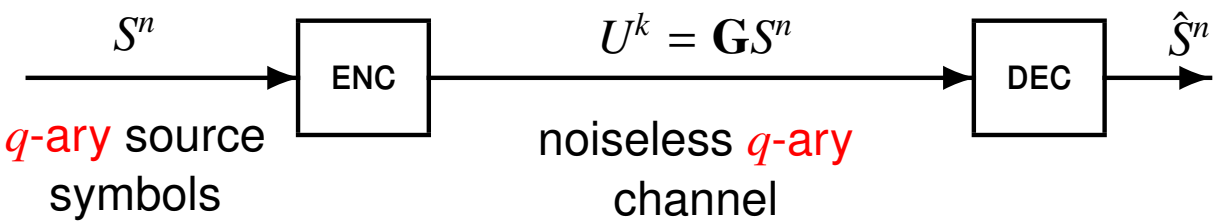
Outline

A Basic Building Block: Linear Binning



$$\text{Rate} = \frac{1}{H(S)} \quad \frac{\text{Source Symbols}}{\text{Channel Use}}$$

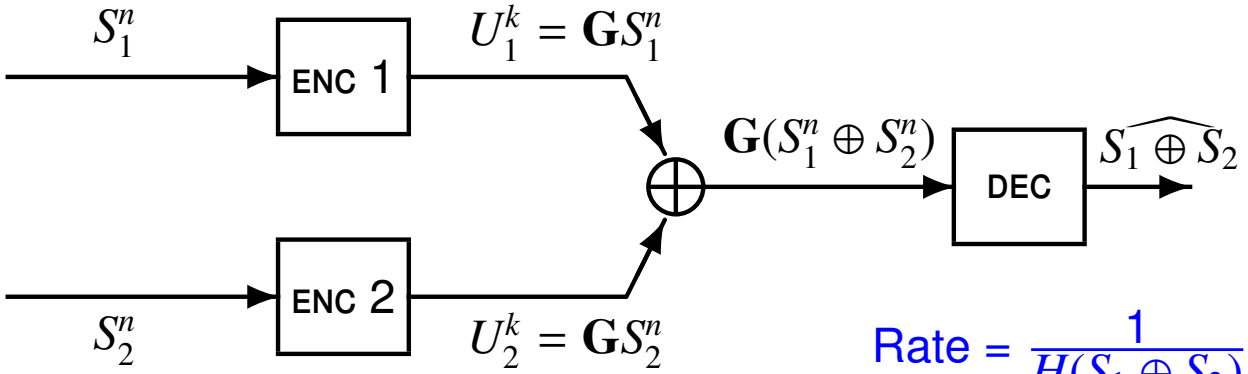
A Basic Building Block: Linear Binning



$$\text{Rate} = \frac{\log_2 q}{H(S)} \quad \frac{\text{Source Symbols}}{\text{Channel Use}}$$

where q is prime

A Basic Building Block: Linear Binning

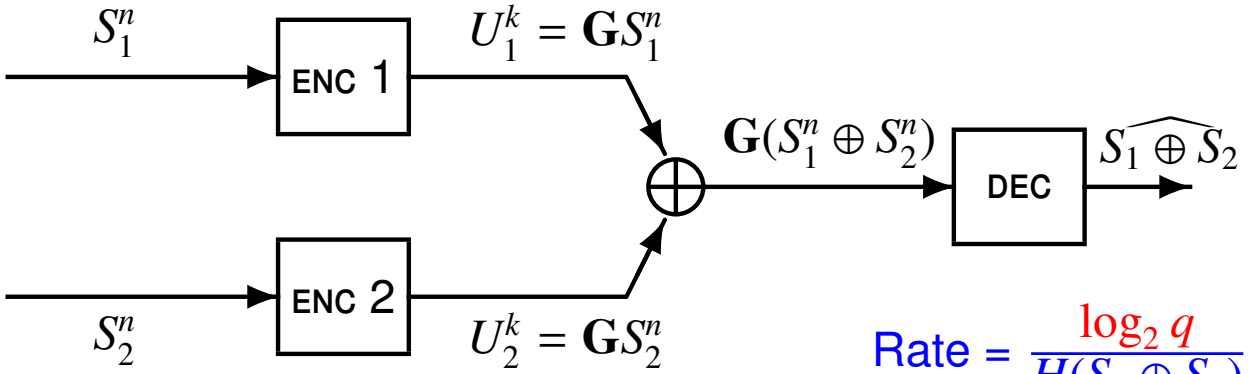


binary source symbols

noiseless binary interference

$$\text{Rate} = \frac{1}{H(S_1 \oplus S_2)} \quad \frac{\text{Computed Symbols}}{\text{Channel Use}}$$

A Basic Building Block: Linear Binning



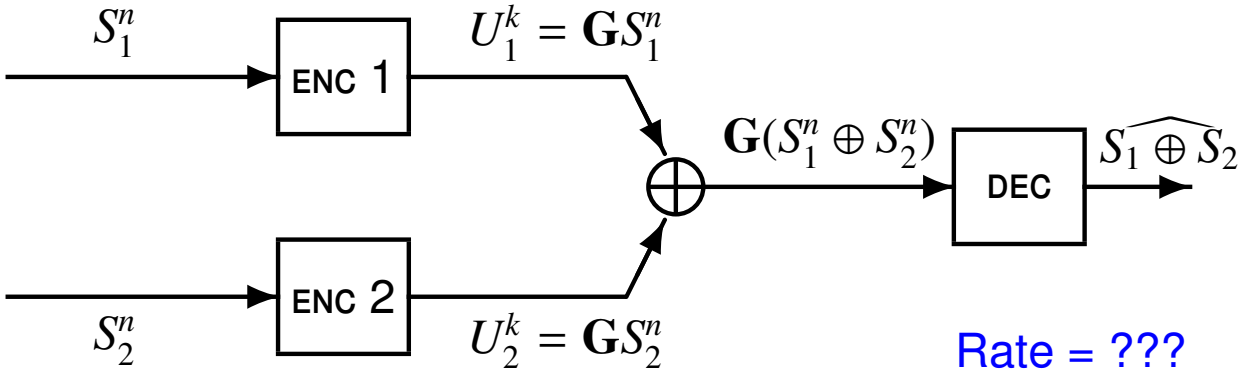
$$\text{Rate} = \frac{\log_2 q}{H(S_1 \oplus S_2)} \quad \frac{\text{Computed Symbols}}{\text{Channel Use}}$$

q-ary source symbols

noiseless q-ary interference

where q is prime

A Basic Building Block: Linear Binning



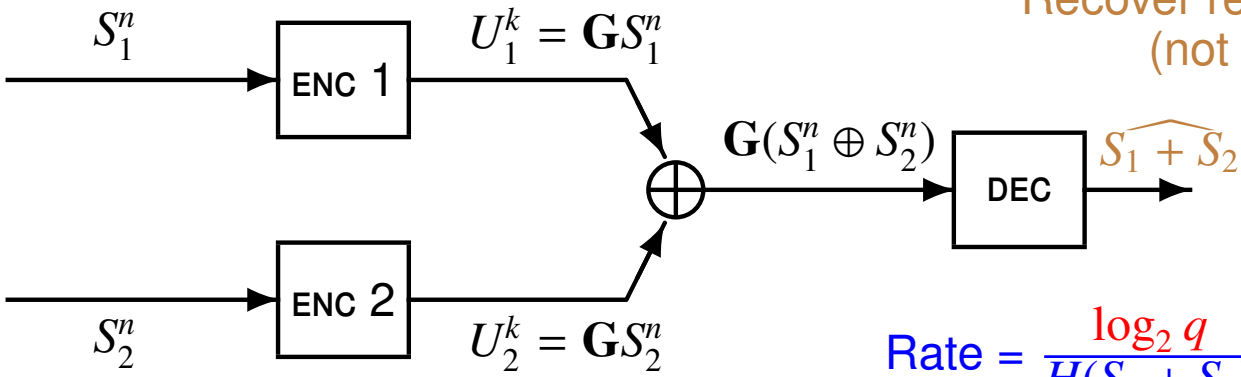
Rate = ???

$$\frac{\text{Computed Symbols}}{\text{Channel Use}}$$

p-ary source symbols

noiseless q-ary interference

A Basic Building Block: Linear Binning



Recover real-valued sum (not modulo)

Rate = $\dfrac{\log_2 q}{H(S_1 + S_2)}$ $\dfrac{\text{Computed Symbols}}{\text{Channel Use}}$

p-ary source symbols

noiseless q-ary interference

where q is prime **and** $q > 2(p-1)$

Represent sources over q-ary alphabet

But is this optimal?

Outline

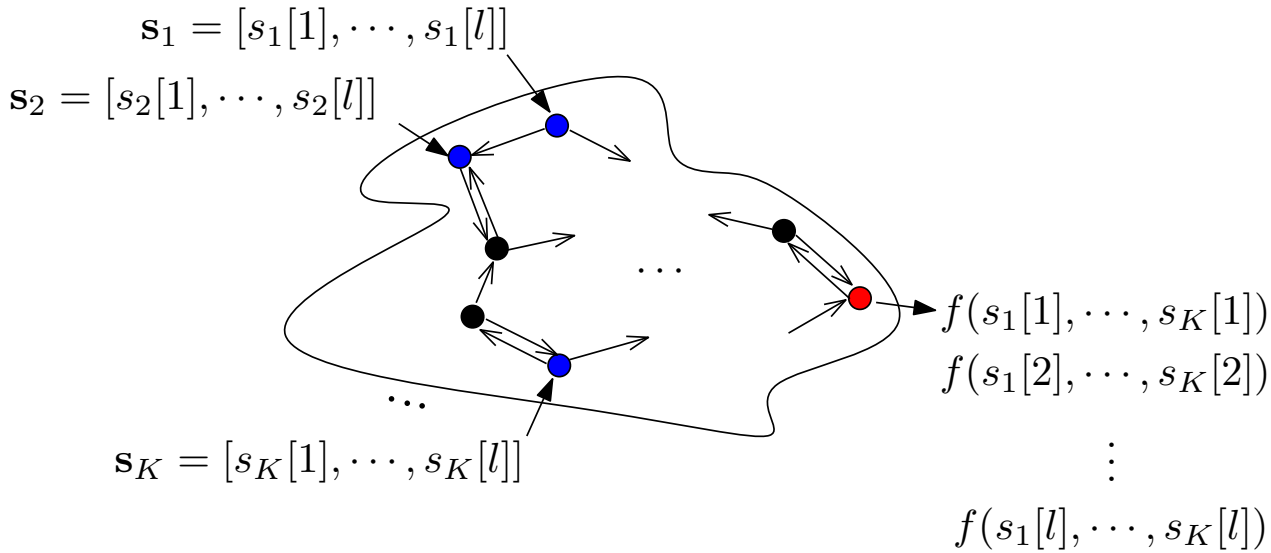
I. A Basic Building Block

II. Type Functions and Arithmetic Sums

III. Lossy Computation

IV. Multiple Receivers

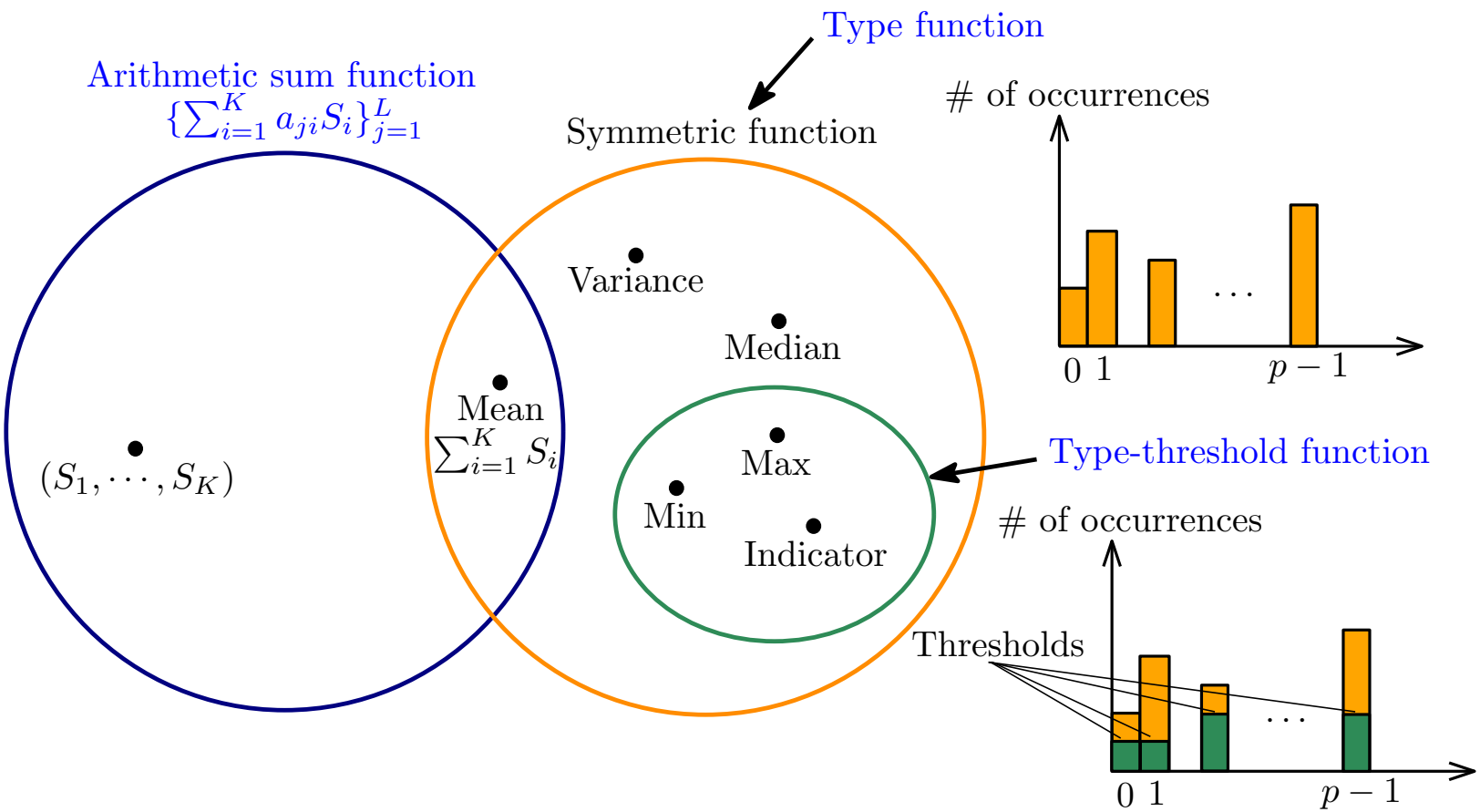
Type Functions and Arithmetic Sums



- $\mathbf{s}[t] := [s_1[t], \cdots, s_K[t]] \in [0 : p-1]^K$ is drawn from some joint pmf
- Computation rate: $R_{comp} = \frac{l}{n}$ [functions/channel use]

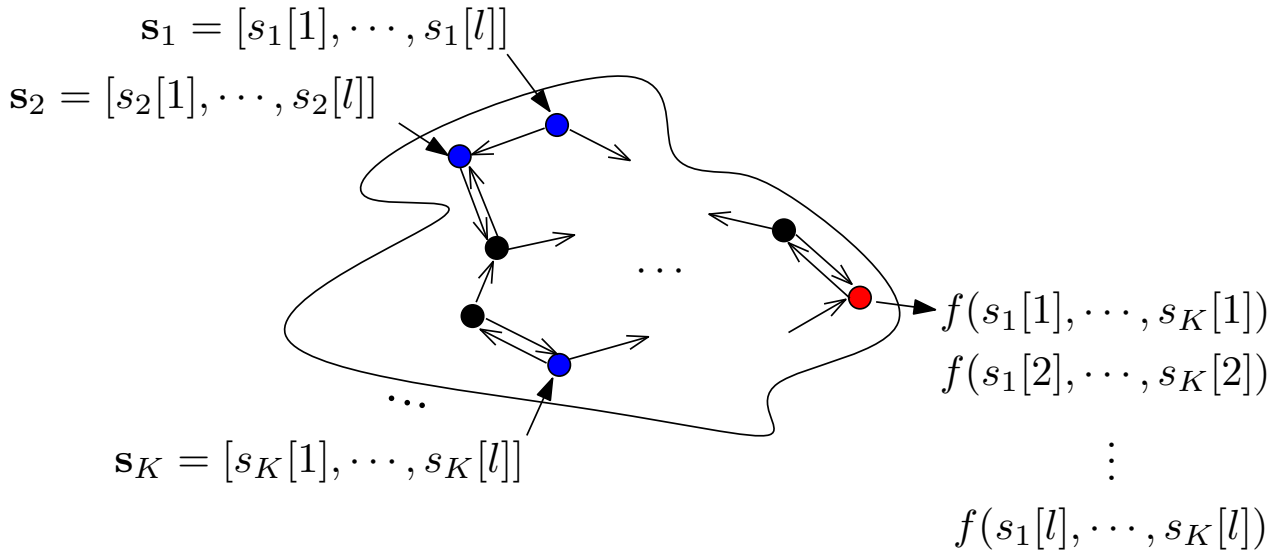
Type Functions and Arithmetic Sums

- Sources: $\mathbf{S} = [S_1, \cdots, S_K] \in [0 : p-1]^K$



- $H(f(\mathbf{S})) \leq H(\mathbf{S})$

Networks of Point-to-point Channels



- $\mathbf{s}[t] := [s_1[t], \cdots, s_K[t]] \in [0 : p-1]^K$ is drawn from some joint pmf
- Networks of Point-to-point Channels
  - Nodes u and v are connected by a channel of capacity $C_{u,v}$.
- Computation rate: $\frac{l}{n}$ [functions/channel use]

Networks of Point-to-point Channels

Theorem (Networks of Point-to-point Channels)

Consider an orthogonal Gaussian network. Then the computation rate of

$$R = \frac{\min_{i\in[1:K]} \bar{C}(\{i\})}{H(f(\mathbf{S}))}$$

is achievable, where

$$\bar{C}(\Sigma) = \min_{\Omega\subseteq V\,:\,\Sigma\subseteq\Omega} \sum_{\substack{u\in\Omega, v\in\Omega^c \\ s.t.(u,v)\in E(\Sigma)}} C_{u,v},$$

for $\Sigma \subseteq [1:K]$ (i.e., Σ is a subset of the source nodes).

- From the cut-set bound

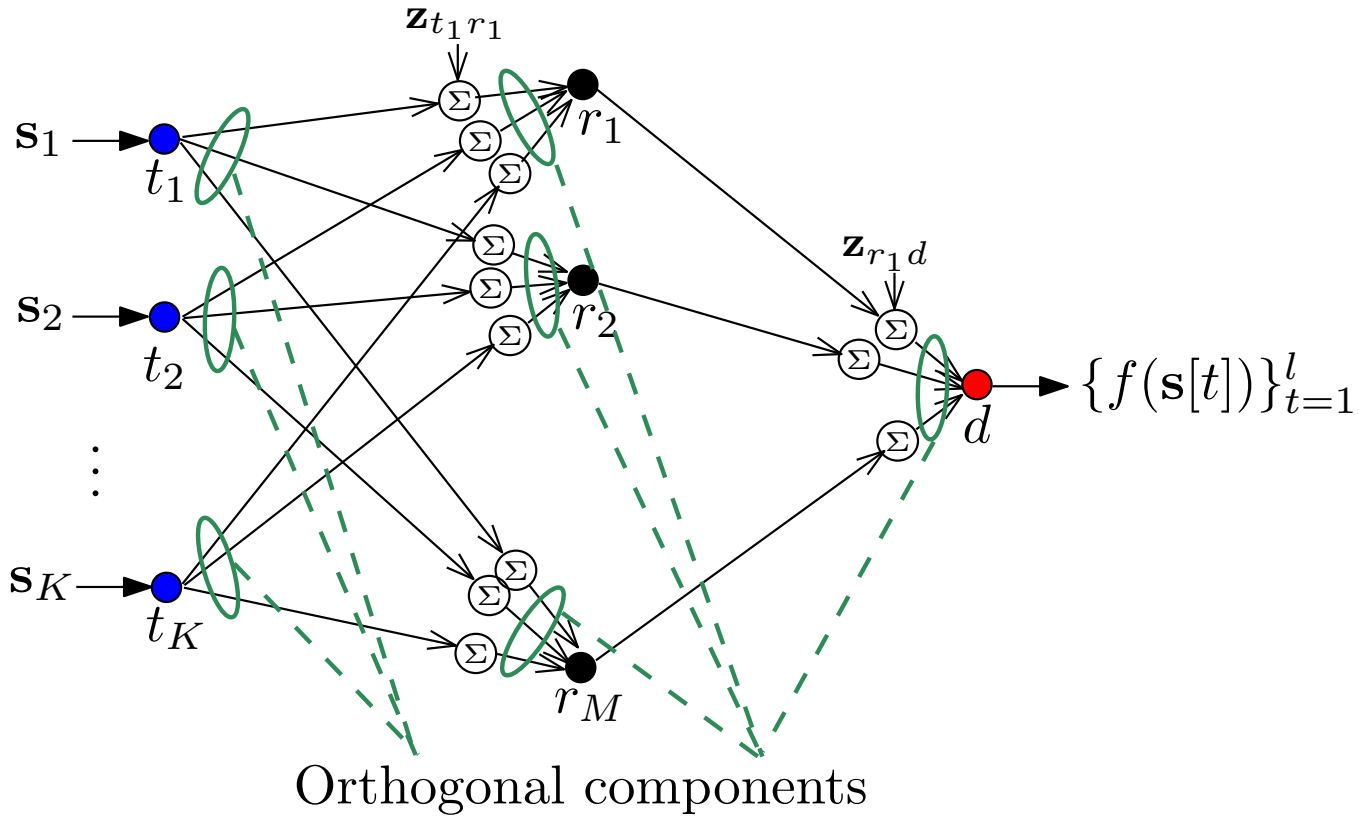
$$R \le \frac{\bar{C}([1:K])}{H(f(\mathbf{S}))}$$

- Computation capacity if

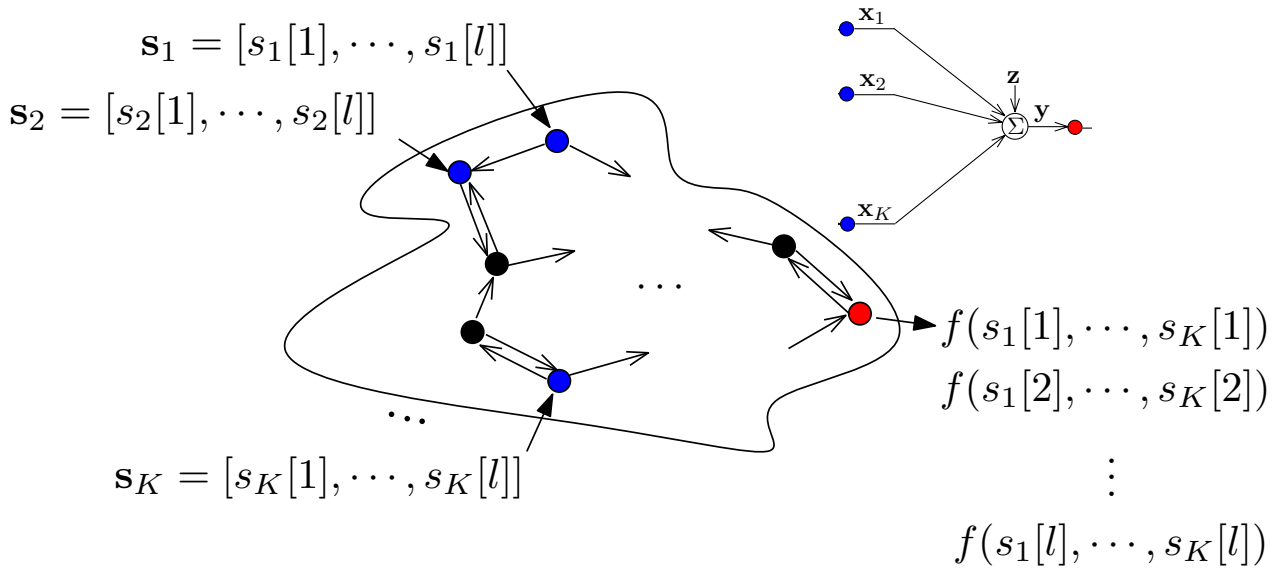
$$\min_{i\in[1:K]} \bar{C}(\{i\}) = \bar{C}([1:K])$$

Networks of Point-to-point Channels

- Example network that achieves computation capacity
- Jeon–Wang–Gastpar ISIT '13



Gaussian Networks With MAC Components



- $\mathbf{s}[t] := [s_1[t], \cdots, s_K[t]] \in [0 : p-1]^K$ is drawn from some joint pmf
- Gaussian networks with multiple-access

$$\text{Length-}n \text{ input–output: } \mathbf{y}_v = \sum_{u \in \Gamma_{in}(v)} h_{u,v}\mathbf{x}_{u,v} + \mathbf{z}_v \text{ for each } v \in V$$

- Computation rate: $\frac{l}{n}$ [functions/channel use]

Gaussian Networks With MAC Components

Theorem (Gaussian Networks With Multiple-Access Components)

Consider a Gaussian network with multiple-access. Then the computation rate of

$$R = \frac{\min_{i\in[1:K]} C^+(\{i\})}{H(f(\mathbf{S}))}$$

is achievable, where

$$C^+(\{i\}) = \min_{\Omega\subseteq V\,:\, i\subseteq\Omega} \sum_{v\in\Omega^c} \left(\mathbf{1}_{\{u\in\Omega\cap\Gamma_{in}(v)\}\neq\emptyset} \max\left\{ \frac{1}{2}\log\left(\frac{1}{|\Gamma_{in}(v)|} + \min_{u\in\Gamma_{in}(v)} h_{u,v}^2 P \right), 0 \right\} \right).$$

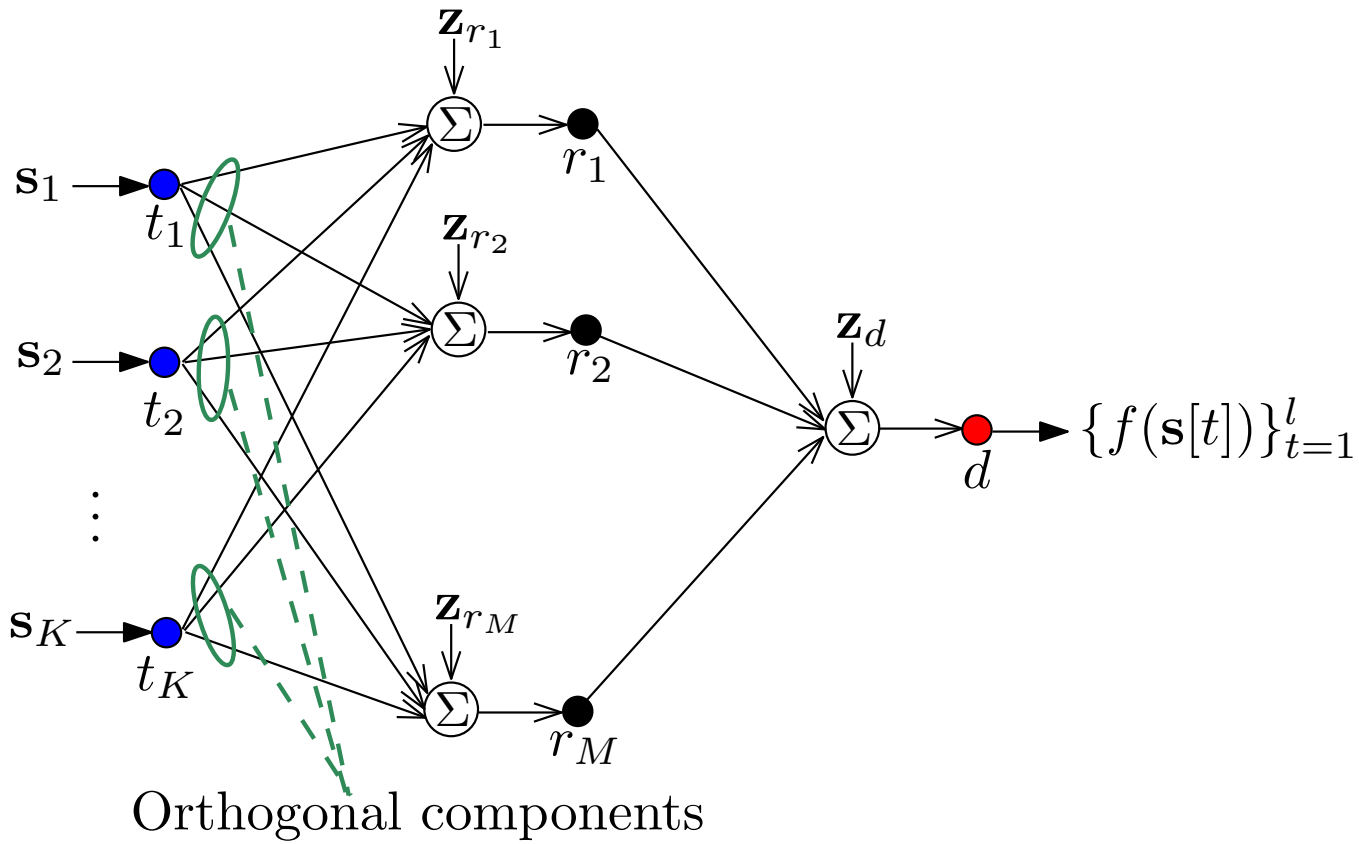
- From the cut-set bound

$$R \leq \frac{\bar{C}([1:K])}{H(f(\mathbf{S}))}$$

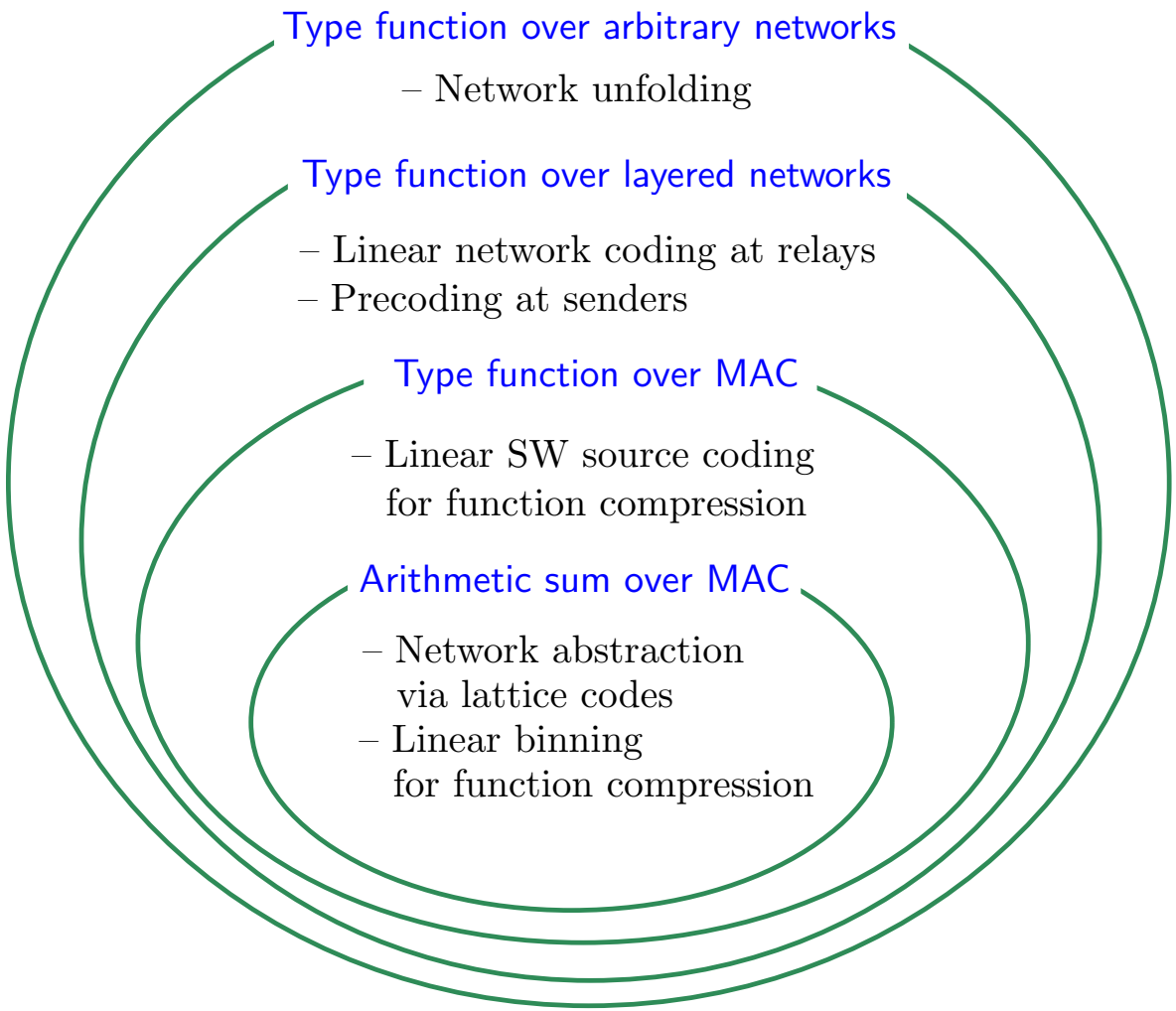
$$\bar{C}(\Sigma) = \min_{\Omega\subseteq V\,:\,\Sigma\subseteq\Omega} \sum_{v\in\Omega^c} \frac{1}{2}\log\left(1 + \left(\sum_{u\in\Omega\cap\Gamma_{in}(v)} |h_{u,v}| \right)^2 P \right) \text{ for } \Sigma\subseteq[1:K]$$

Gaussian Networks With MAC Components

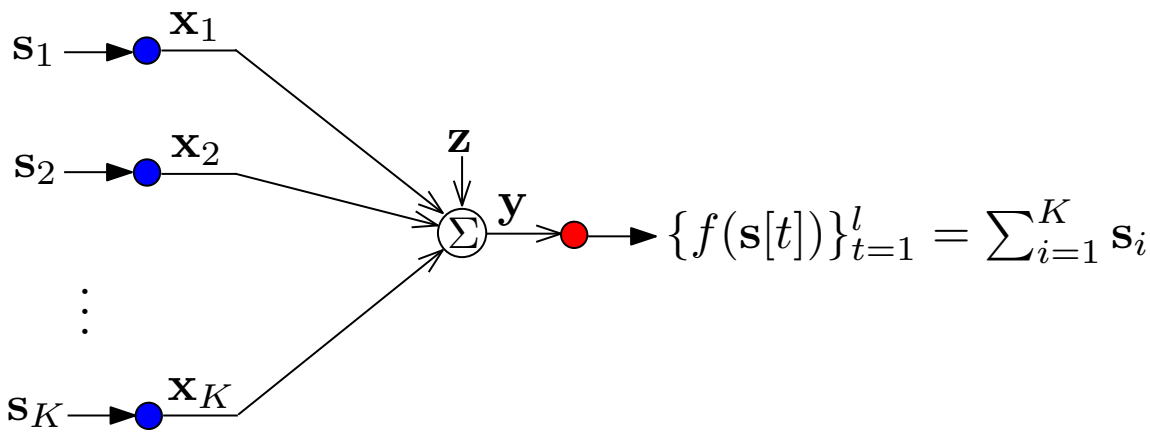
- Example network that achieves computation capacity to within $c\frac{|V|\log|V|}{H(f(\mathbf{S}))}$ for any power P, where $c > 0$ is some constant
- Jeon–Wang–Gastpar ISIT '13



Key Ingredients



Arithmetic Sum Over Gaussian MAC



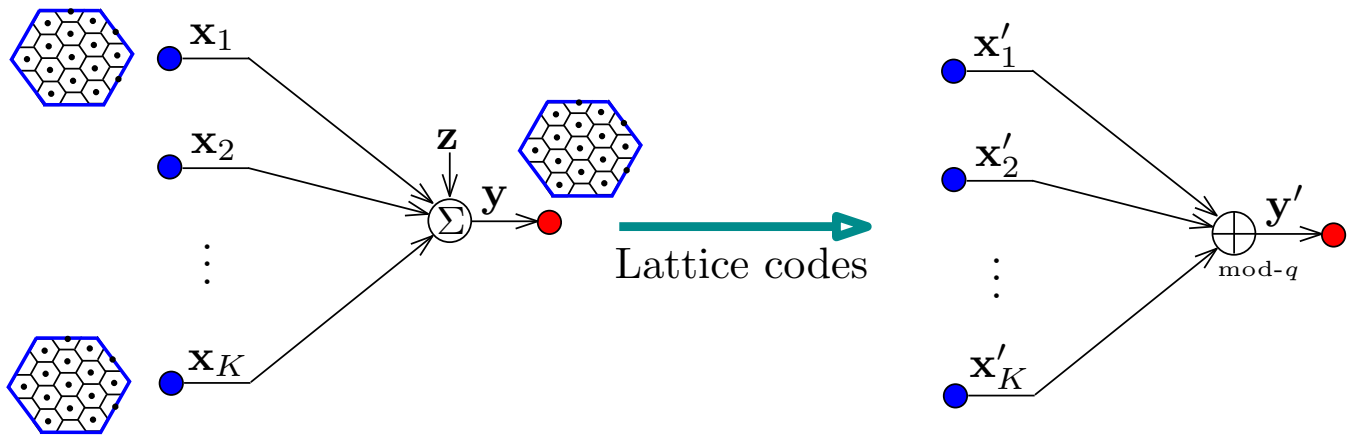
- Length-n input–output

$$\mathbf{y} = \sum_{i=1}^{K} \mathbf{x}_i + \mathbf{z}$$

- Arithmetic sum

$$f(\mathbf{s}[t]) = \sum_{i=1}^{K} s_i[t] \text{ for } t \in [1 : l]$$

Transformation Into Modulo-q Sum Channel



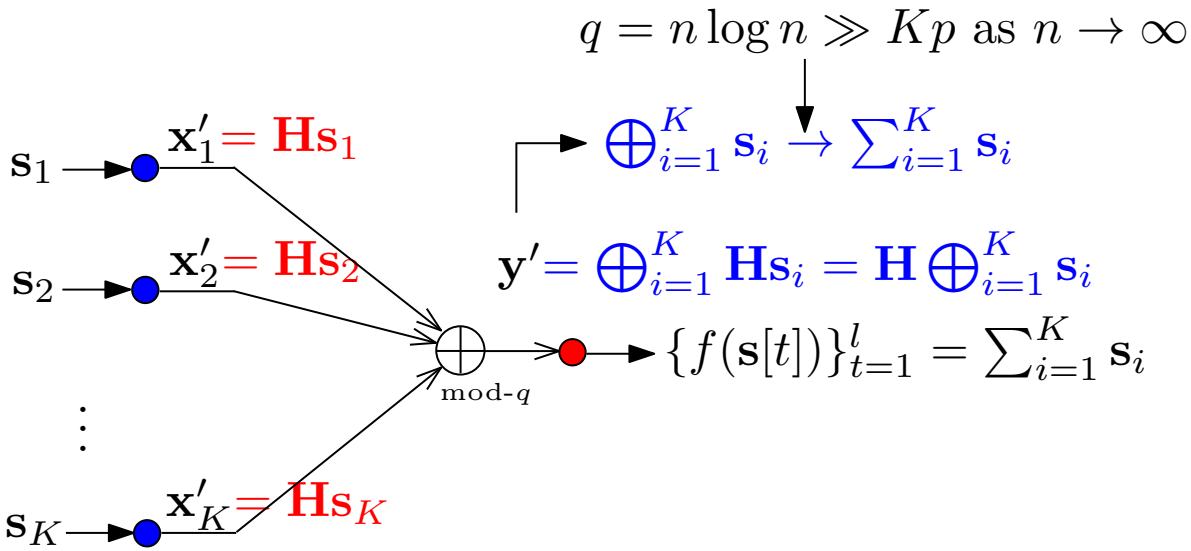
Length-n Gaussian MAC Length-m modulo-q sum channel

- Network abstraction via lattice codes
  - Decode the sum of lattice points
  - Erez–Zamir IT '04, Erez-Zamir IT '08, Zamir IT '09, Nazer–Gastpar IT '11
- Transformed modulo-q sum channel

$$\mathbf{y}' = \bigoplus_{i=1}^{K} \mathbf{x}'_i, \text{ where } \mathbf{x}'_i \in \mathbb{F}_q{}^m$$

  - $q = n \log n$
  - $m = n \max\left\{\frac{1}{2}\log\left(\frac{1}{K} + P\right), 0\right\} (\log q)^{-1}$

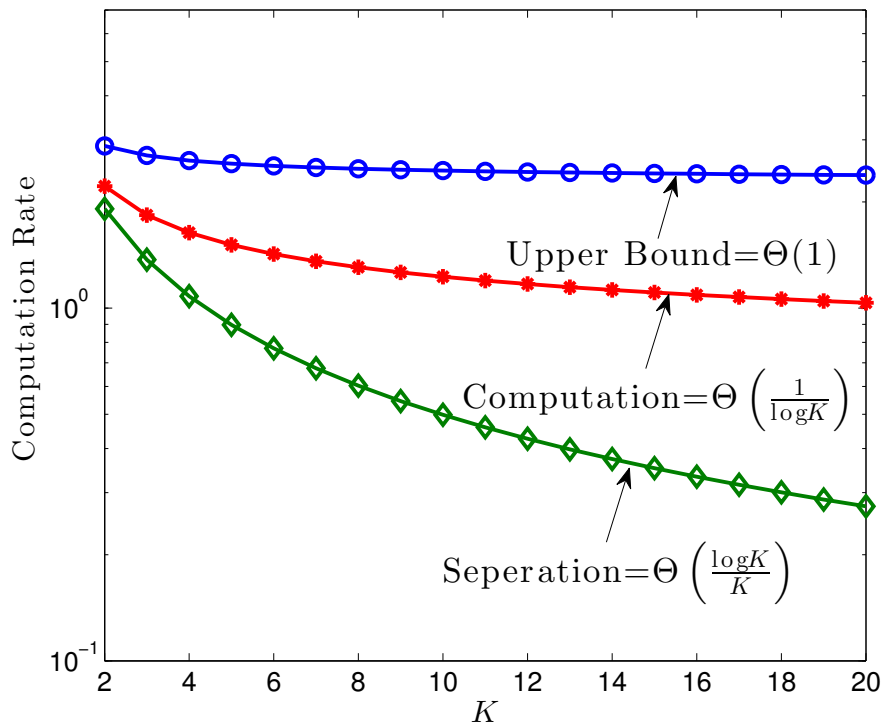
Computation Over Transformed Modulo-q Sum Channel



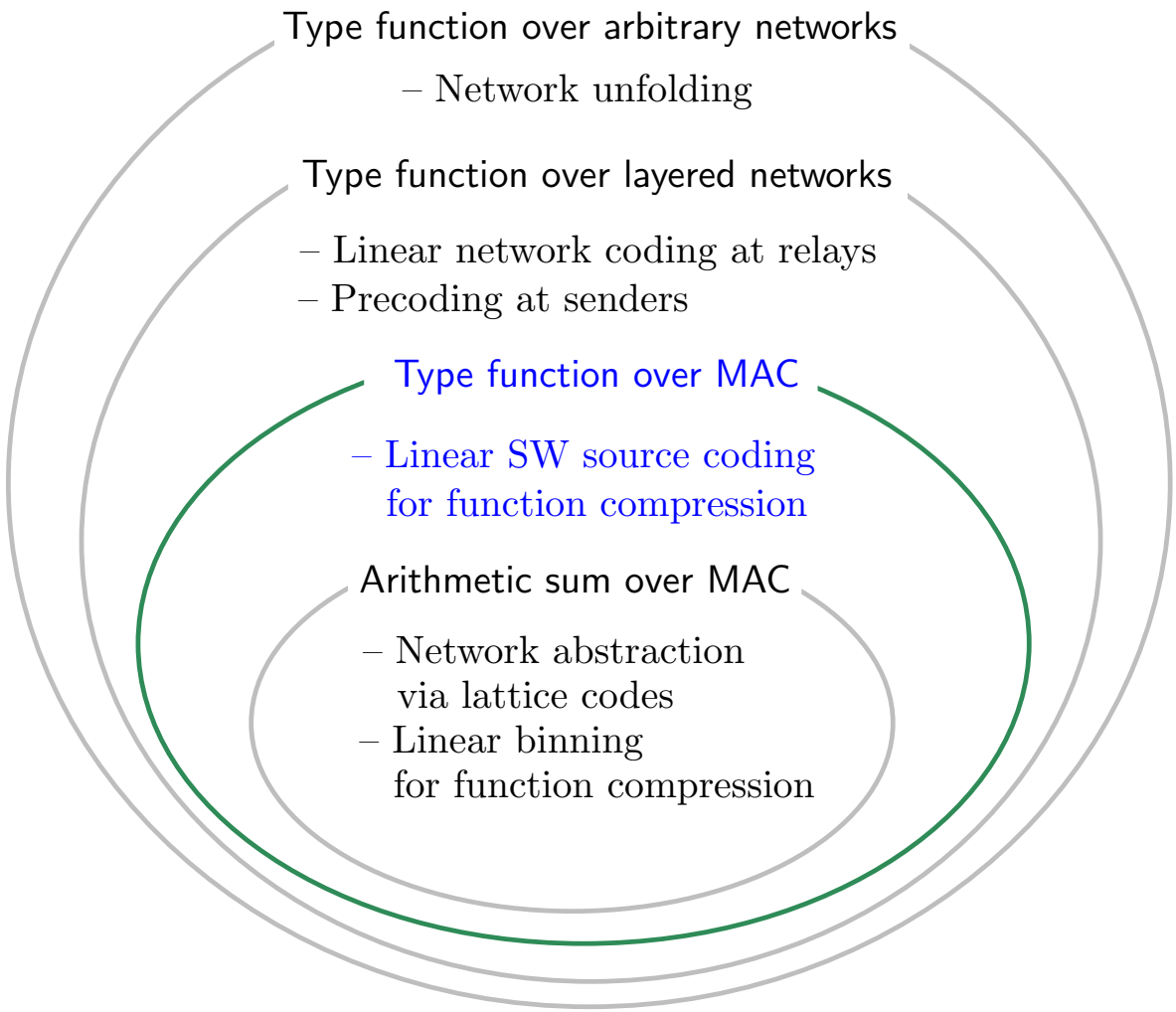
- Compression via linear binning
  - Csiszár IT '82, Nazer–Gastpar IT '07
  - Compensate the inefficiency of uncoded transmission
- Optimal computation over the transformed channel !

Computation Rates

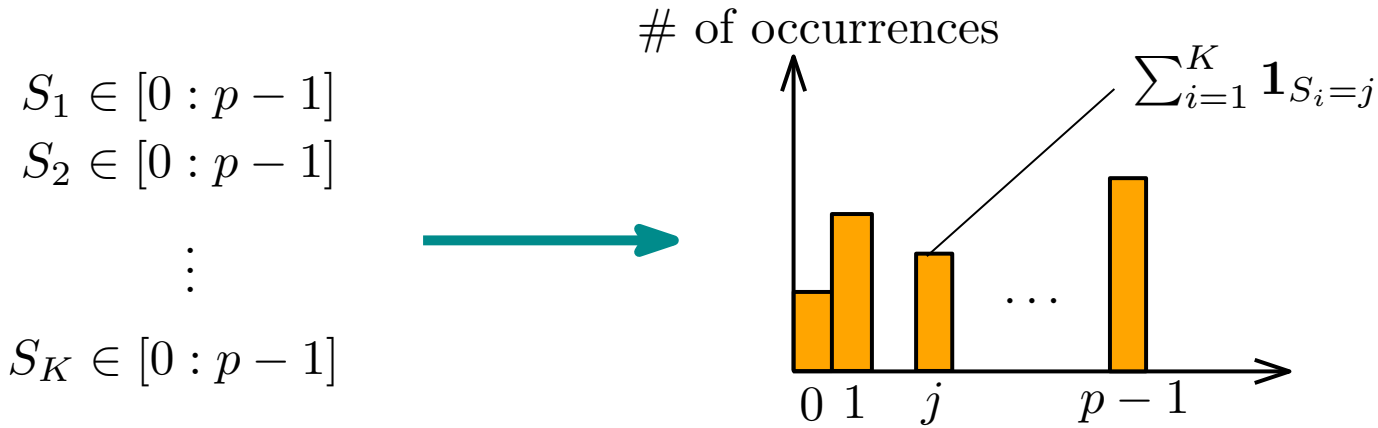
- $R_{\text{comp}} = \frac{\max\left\{\frac{1}{2}\log\left(\frac{1}{K}+P\right),0\right\}}{H\left(\sum_{i=1}^{K} S_i\right)}$
- $R_{\text{upper}} = \frac{\frac{1}{2}\log(1+K^2P)}{H\left(\sum_{i=1}^{K} S_i\right)}$
- $R_{\text{sep}} = \frac{\frac{1}{2}\log(1+KP)}{H(S_1,\cdots,S_K)}$
- Example: $\{S_i\}_{i=1}^{K}$ are i.i.d. binary sources



Key Ingredients



Type Computation

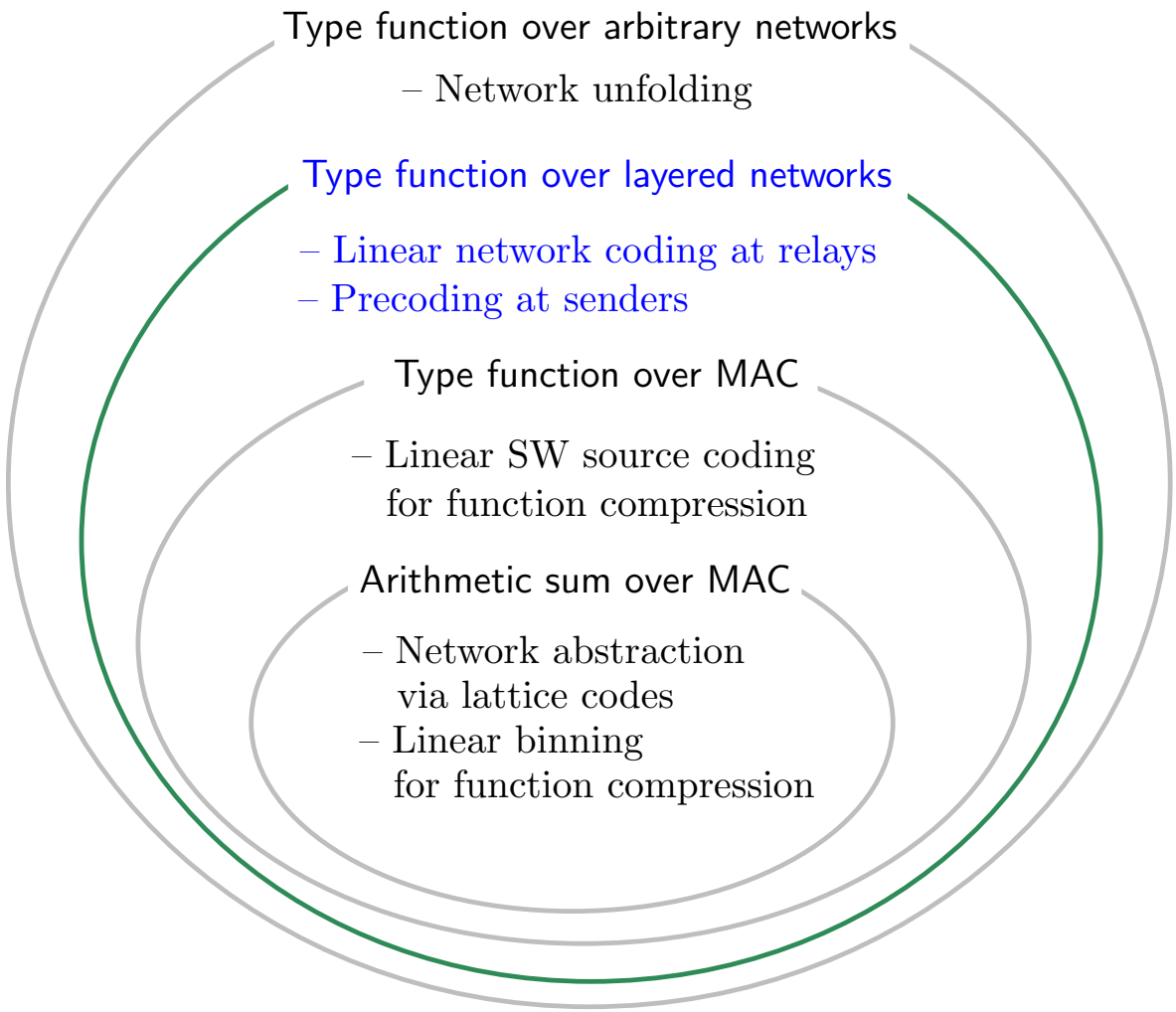


- Type function can be represented as multiple arithmetic sums

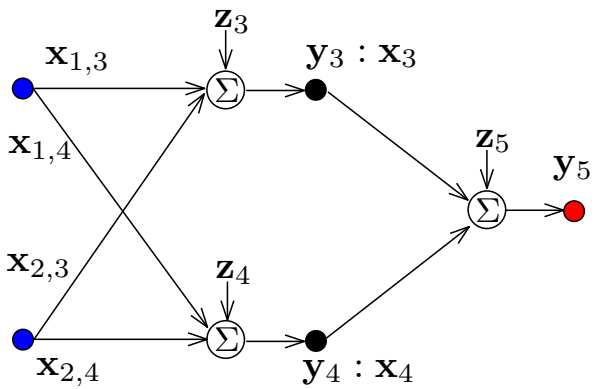
$$f(\mathbf{S}) = \left(\sum_{i=1}^{K} \mathbf{1}_{S_i=0}, \sum_{i=1}^{K} \mathbf{1}_{S_i=1}, \cdots, \sum_{i=1}^{K} \mathbf{1}_{S_i=p-1} \right)$$

- Linear Slepian–Wolf source coding
  - Compression of $\left(\sum_{i=1}^{K} \mathbf{1}_{S_i=0}, \sum_{i=1}^{K} \mathbf{1}_{S_i=1}, \cdots, \sum_{i=1}^{K} \mathbf{1}_{S_i=p-1} \right)$
- Optimal computation over the transformed channel !

Key Ingredients

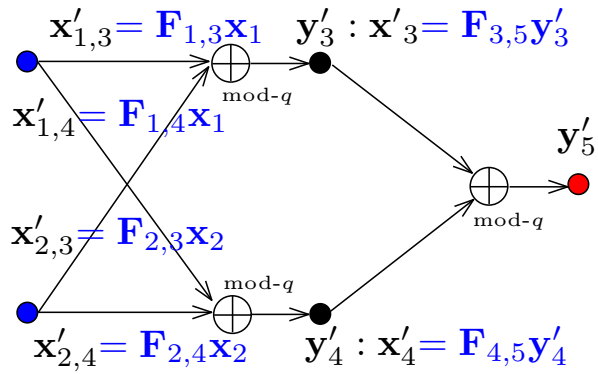


Network Transformation



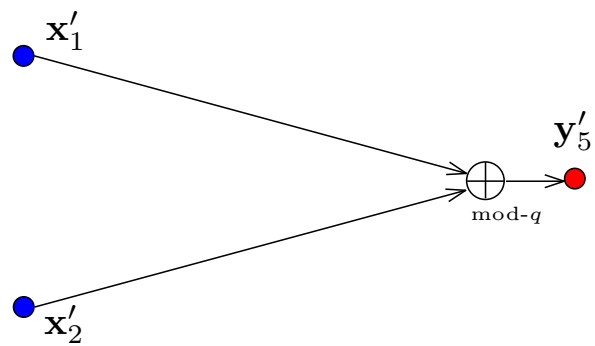
Length-n Gaussian network

Lattice codes



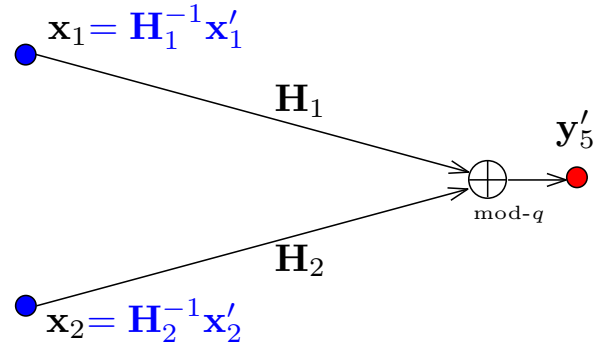
Length-m modulo-q sum network

Linear network coding



Length-m modulo-q sum channel

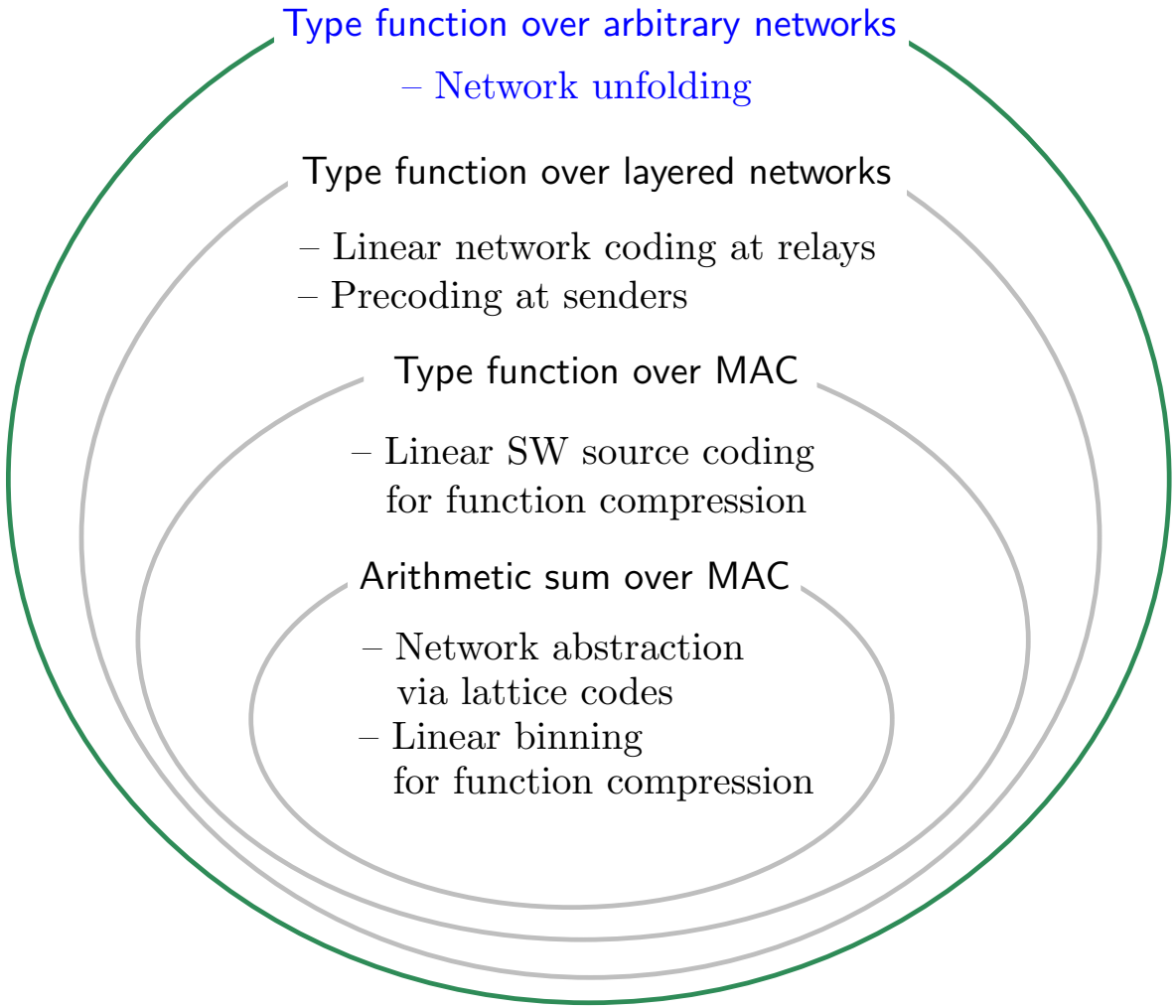
Precoding



Length-m modulo-q sum channel

- Computation over the transformed modulo-q sum channel

Key Ingredients



- Full paper on ArXiv.

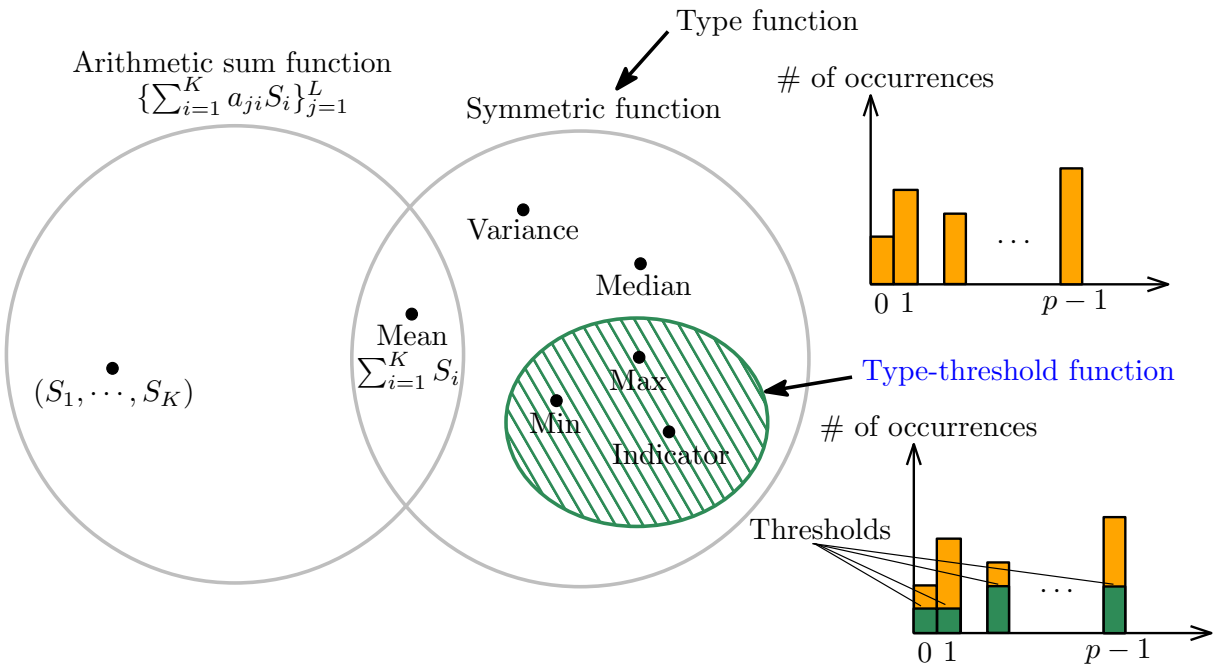
An Afterthought regarding Optimality

- The strategies just discussed give us intuitively pleasing general achievability statements of the form

$$R = \frac{C_{lower}}{H(f(S_1, S_2, \ldots, S_K))} \frac{\text{computed symbols}}{\text{channel use}}$$

  for large classes of functions $f(\cdots)$.
- The resulting performance is sometimes optimal or close to optimal.
- But generally, are we missing fundamental ingredients?
- Yes — here is one: *More clever scheduling.*

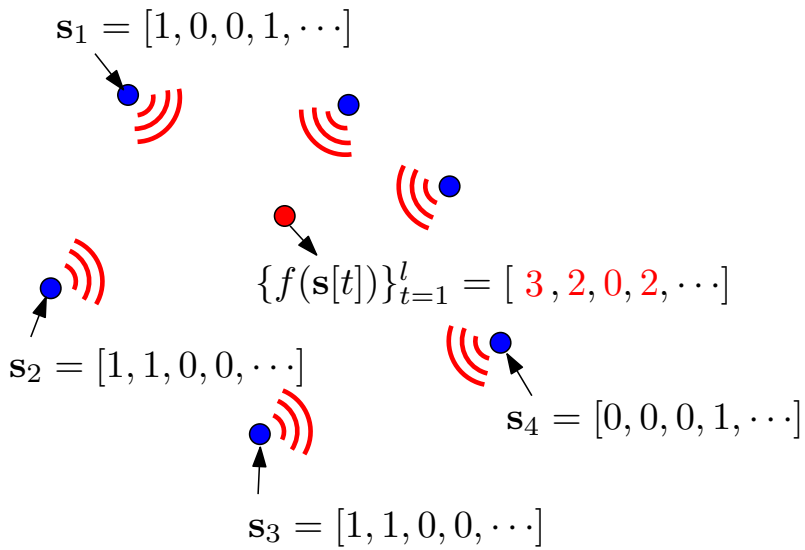
An Afterthought regarding Optimality



- Symmetric function computation from the type function
  - Type function: $H(f(\mathbf{S})) = \Theta(\log K)$
  - Arithmetic sum function: $H(f(\mathbf{S})) = \Theta(\log K)$
  - Maximum function: $H(f(\mathbf{S})) = \Theta(1)$
- How to compute the type-threshold function over Gaussian Networks?

An Afterthought regarding Optimality

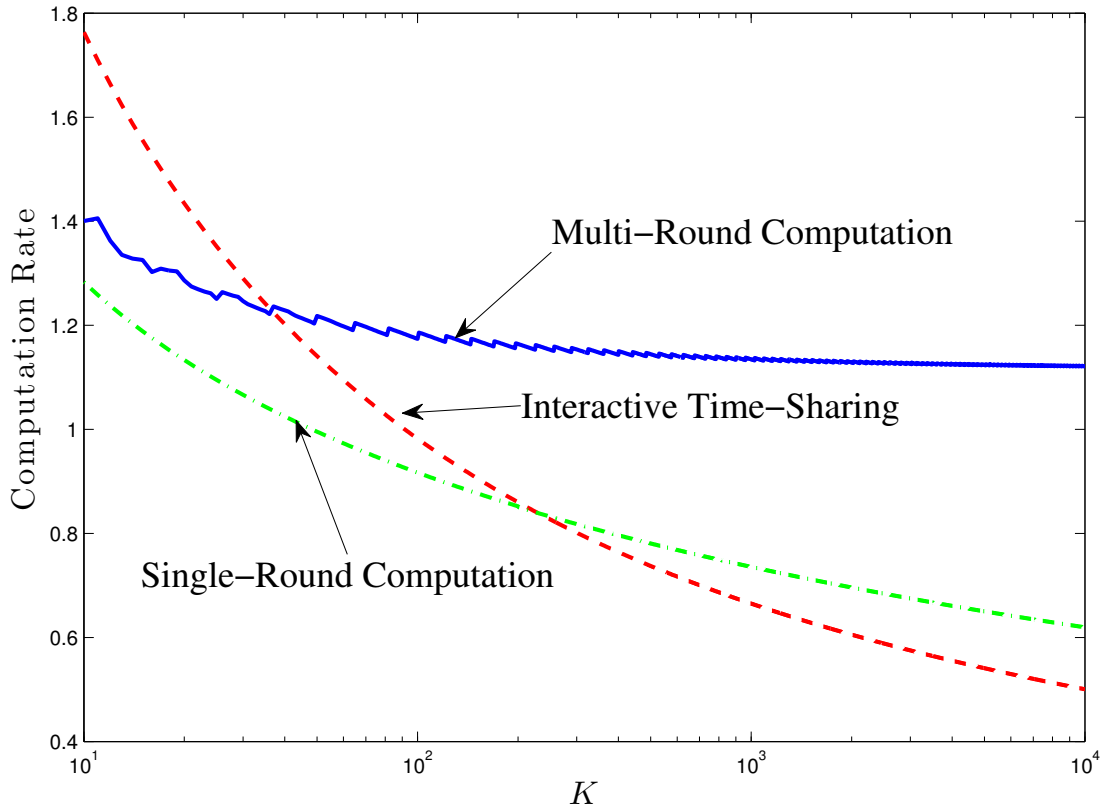
- $\mathbf{S} \in \{0,1\}^K$ are i.i.d. drawn from $\mathrm{Bern}(\alpha)$



- Time-sharing (Multi-round)
  - Interactive time-sharing (Ma–Ishwar–Gupta IT '12)
  - $R = \Theta(1)$ is achievable when $\alpha = 1/2$
  - $R = \Theta(\frac{1}{\log K})$ is achievable when $\alpha = 1/K$
- Using the strategy that we just discussed
  - $R = \Theta(\frac{1}{\log K})$ is achievable when $\alpha = 1/2$
  - $R = \Theta(1)$ is achievable when $\alpha = 1/K$

Binary Maximum Computation

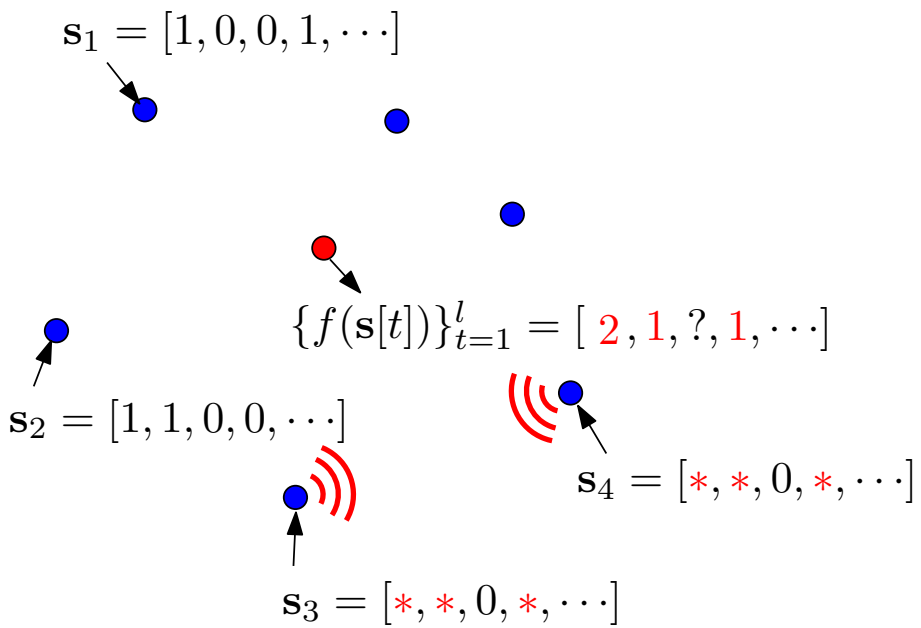
- $\alpha = 1/\sqrt{K}$ and $P = 20$ dB



  - Both interactive time-sharing and single-round computation fail to achieve $R = \Theta(1)$

- Multi-round computation achieves $R = \Theta(1)$

Multi-Round Computation



- Multi-round computation achieves $R = \Theta(1)$
  - For any i.i.d. sources
  - For any type-threshold functions satisfying $H(f(\mathbf{S})) = \Theta(1)$
  - Wang–Jeon–Gastpar ISIT '13

An Afterthought regarding Optimality

- Hence, in general, the fundamental strategy introduced earlier needs to be combined with additional elements.
- For the binary maximum computation in a colocated network, appropriate scheduling is crucial.

Outline

I. A Basic Building Block

II. Type Functions and Arithmetic Sums

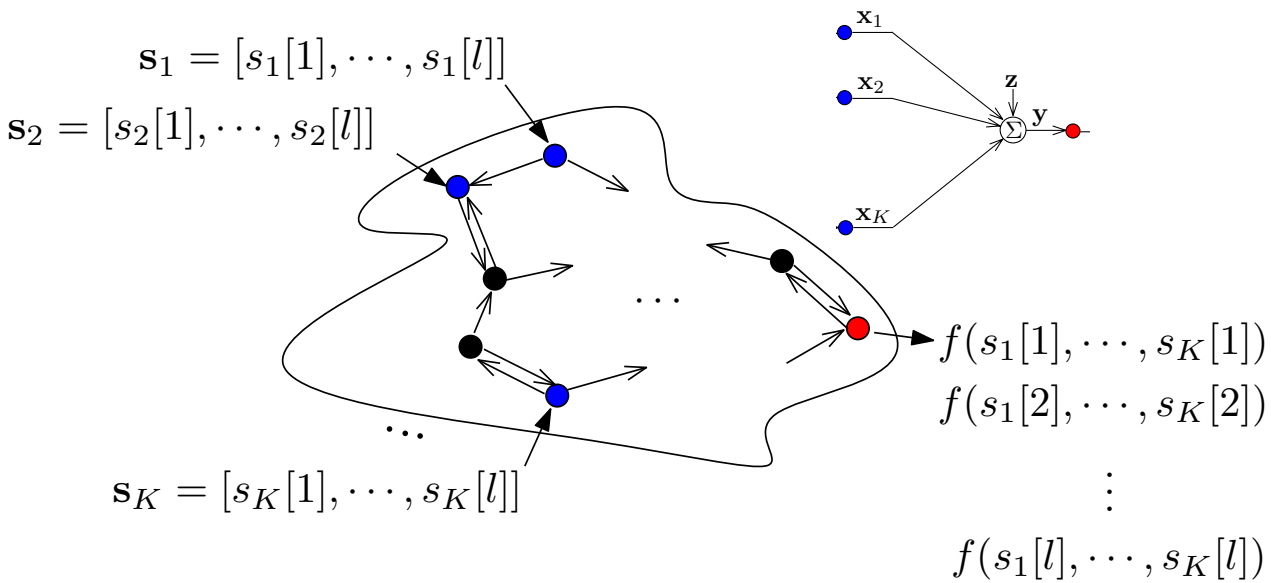
III. Lossy Computation

IV. Multiple Receivers

Lossy Computation

- So far, we have considered the requirement for the receiver to recover the function value without error.
- But we could allow recovery to within a fidelity criterion.
- The general problem is difficult.

Let us consider an example:



- Gaussian sources of unit variance.
- Destination wants $f(s_1, s_2, \ldots, s_K) = s_1 + s_2 + \ldots + s_K$ to within smallest MSE.

Lossy Computation

- Define the *degree* d of the network to be the *maximum number of inputs of any of the multiple-access components that constitute our network.*
- Under some assumptions on the channel coefficients in the MACs, one can prove that the following distortion is achievable:

$$D = K^2 2^{2\alpha} \max_{i \in \mathcal{S}} 2^{-2q\bar{C}(\{i\})},$$

where

$$\alpha = |V|((d+1)\log(d+2) + 2\log d + 1).$$

- One can also prove that any achievable distortion must satisfy

$$D \geq \max_{i \in \mathcal{S}} 2^{-2q\bar{C}(\{i\})}.$$

Here, as before,

$$\bar{C}(\Sigma) = \min_{\Omega \subseteq V \,:\, \Sigma \subseteq \Omega} \sum_{v \in \Omega^c} \frac{1}{2} \log \left(1 + \left(\sum_{u \in \Omega \cap \Gamma_{in}(v)} |h_{u,v}| \right)^2 P \right) \text{ for } \Sigma \subseteq [1:K]$$

Lossy Computation

- The proof involves a channel coding argument much like the one we discussed earlier.
- Plus, it involves a lattice quantization argument for the Gaussian sources.
- The two lattices are coupled in the natural way so as to preserve orderings.
- See Jiening Zhan, Se Yong Park, M. G., and Anant Sahai “Linear Function Computation in Networks: Duality and Constant Gap Results.” *IEEE Journal on Selected Areas in Communications,* 31(4):620-638, April 2013.

Outline

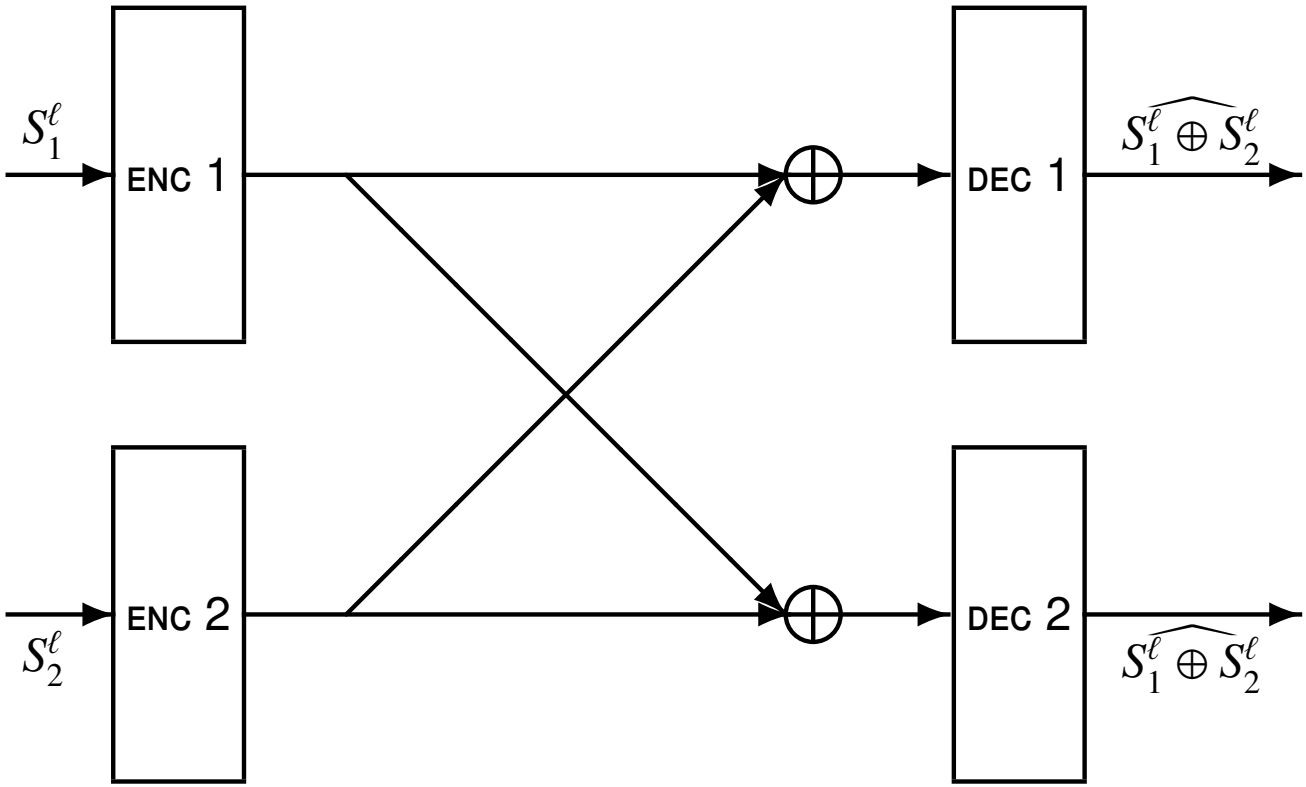
I. A Basic Building Block

II. Type Functions and Arithmetic Sums

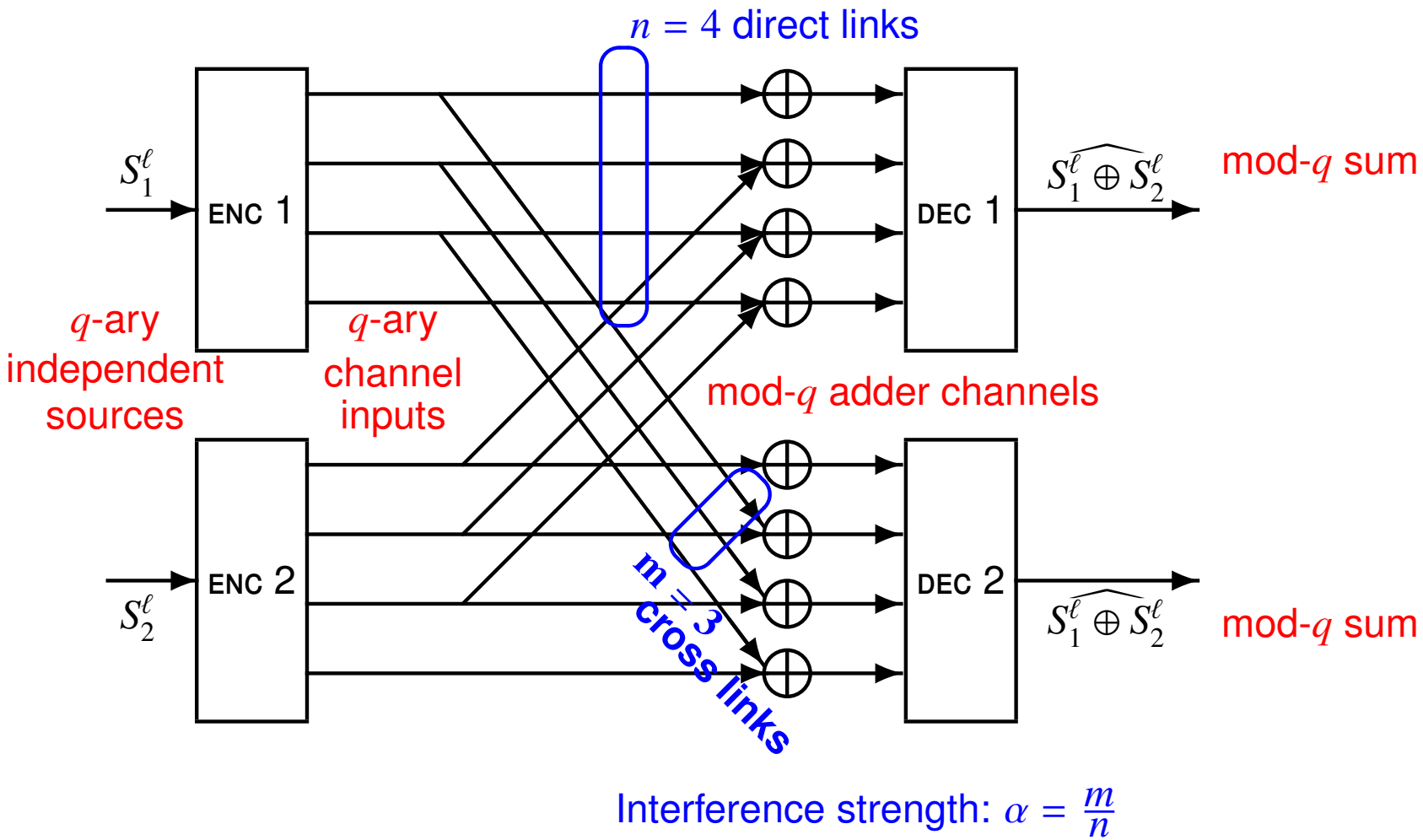
III. Lossy Computation

IV. Multiple Receivers

Multiple Receivers: An Example



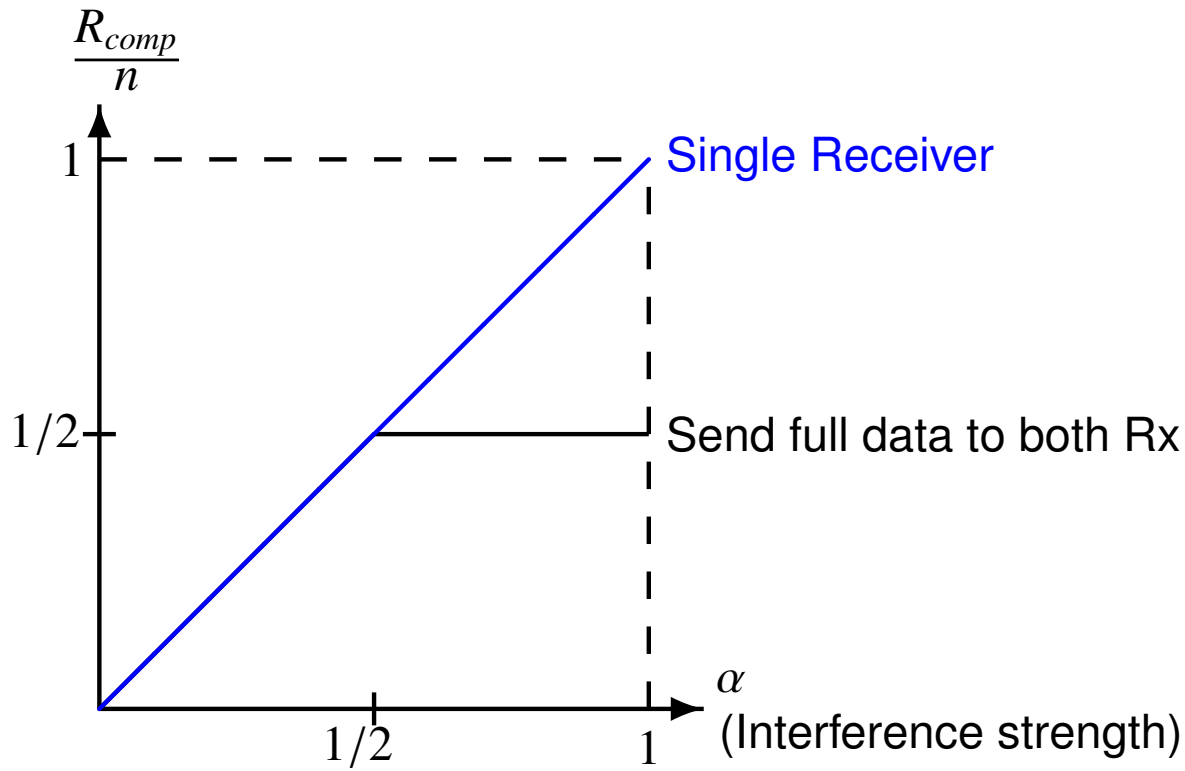
Multiple Receivers: An Example



Interference strength: $\alpha = \frac{m}{n}$

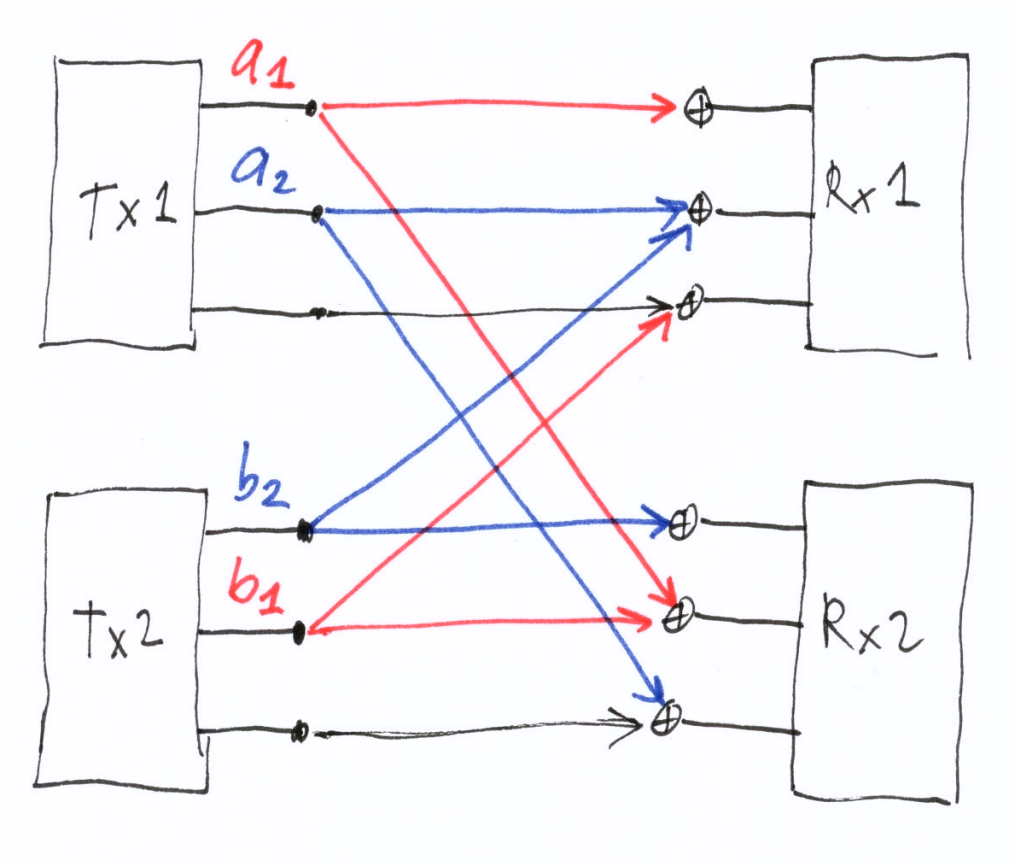
Multiple Receivers

- If we **ignore** Receiver 2, what is the computation rate for Receiver 1?
- Clearly, we can communicate m q-ary sums per channel use.
- Hence, $R_{comp} = m$.
- This is optimal from a simple cut-set bound.
- By contrast, $R_{data} = \min\{m, n/2\}$.



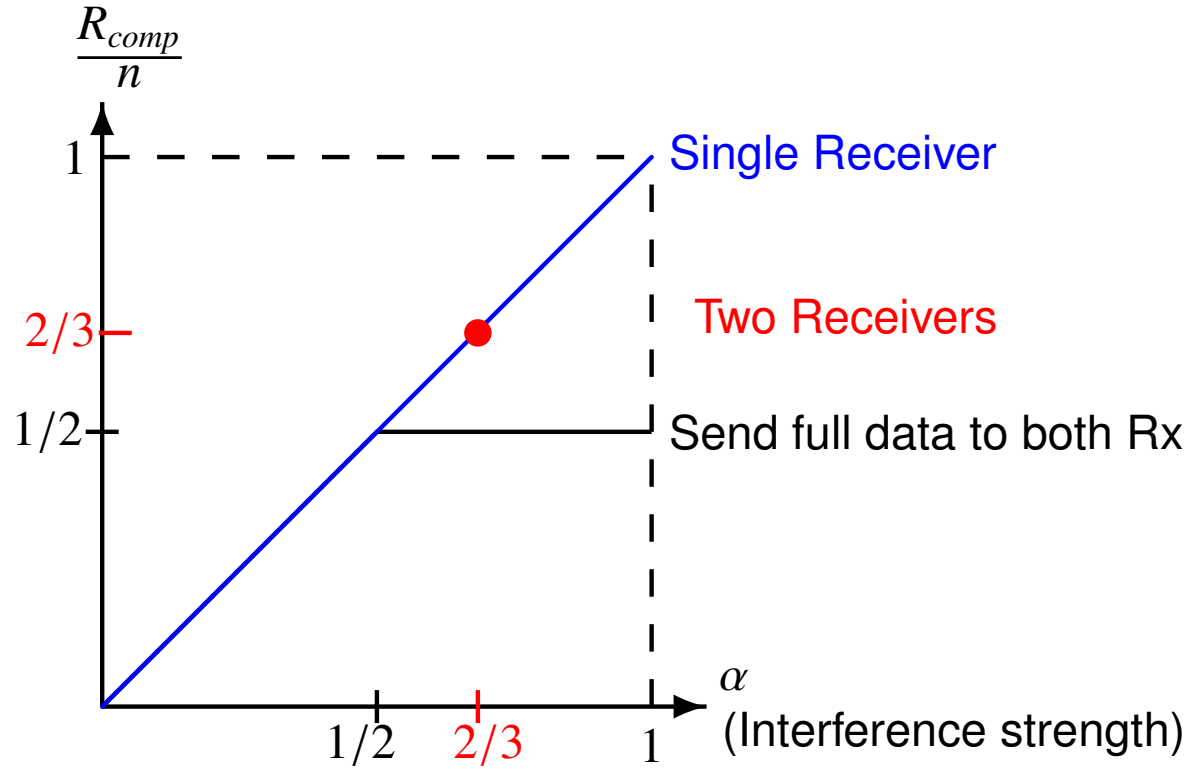
Multiple Receivers

- Now, let's bring Receiver 2 back in.
- If we let both receiver first obtain the full data, we have two multiple-access channels. Due to symmetry, we find again $R_{data} = \min\{m, n/2\}$.
- What can we do for computation?
- Let us consider the special case $n = 3, m = 2$.

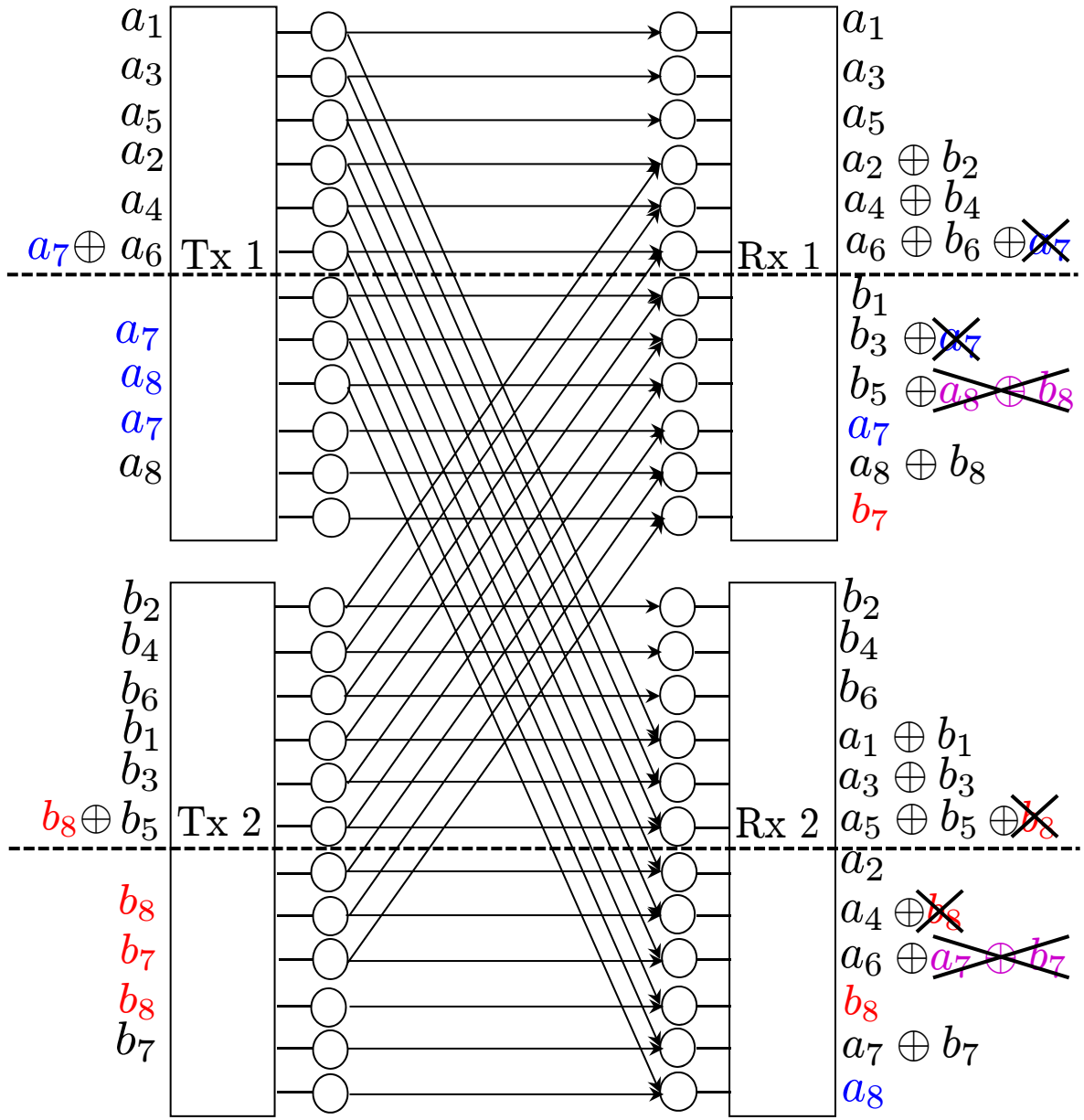


Multiple Receivers

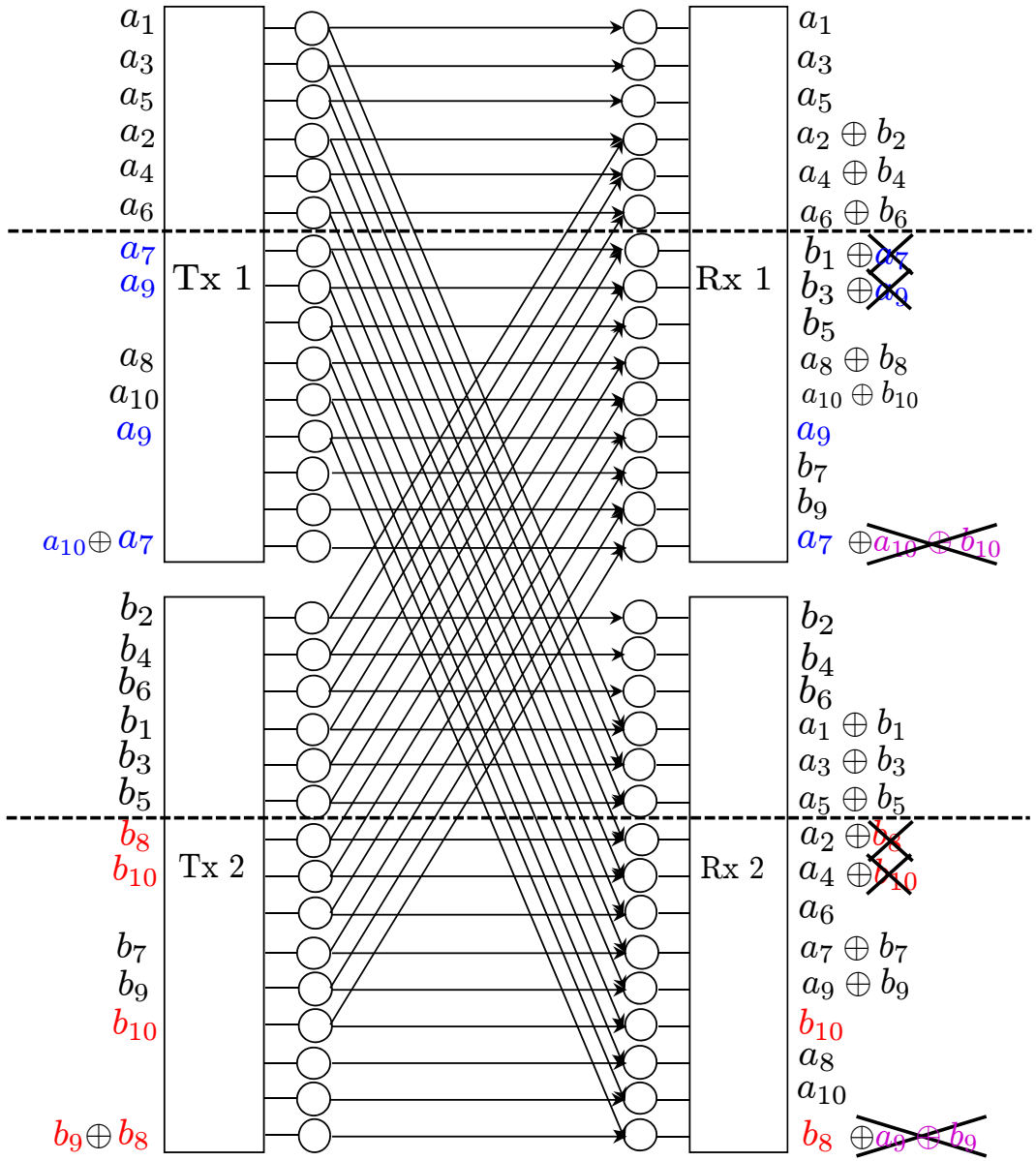
- By a graph coloring argument, we can show that any model $n = 3k, m = 2k$, for all integers k, can be decomposed into **independent** models with $n = 3, m = 2$.
- Hence, we have:



Multiple Receivers: $\alpha = 3/4$



Multiple Receivers: $\alpha = 4/5$



Multiple Receivers

- Can we do better than $2/3$?

$$
\begin{aligned}
& n(3R_{\text{comp}} - \epsilon_n) \\
\leq & I(S_1^\ell \oplus S_2^\ell; Y_1^n) + I(S_1^\ell \oplus S_2^\ell; Y_2^n) + I(S_1^\ell \oplus S_2^\ell; Y_2^n) \\
\leq & [H(Y_1^n) - H(Y_1^n | S_1^\ell \oplus S_2^\ell)] \\
& + [H(Y_2^n) - H(Y_2^n | S_1^\ell \oplus S_2^\ell, Y_1^n)] + I(S_1^\ell \oplus S_2^\ell; Y_2^n) \\
\leq & H(Y_1^n) + H(Y_2^n) \\
& - H(Y_1^n, Y_2^n | S_1^\ell \oplus S_2^\ell) + I(S_1^\ell \oplus S_2^\ell; Y_2^n, S_2^\ell)
\end{aligned}
$$

Now, since S_2^ℓ is independent of $S_1^\ell \oplus S_2^\ell$, we have

$$
\begin{aligned}
= & H(Y_1^n) + H(Y_2^n) \\
& - H(Y_1^n, Y_2^n | S_1^\ell \oplus S_2^\ell) + I(S_1^\ell \oplus S_2^\ell; Y_2^n | S_2^\ell)
\end{aligned}
$$

Next, since X_2^n is a function of S_2^ℓ,

$$
\begin{aligned}
= & H(Y_1^n) + H(Y_2^n) \\
& - H(Y_1^n, Y_2^n | S_1^\ell \oplus S_2^\ell) + H\left(\mathbf{G}X_1^n | S_2^\ell\right)
\end{aligned}
$$

Multiple Receivers

We had:

$$
\begin{aligned}
& n(3R_{\text{comp}} - \epsilon_n) \\
& \leq H(Y_1^n) + H(Y_2^n) \\
& \quad - H(Y_1^n, Y_2^n | S_1^\ell \oplus S_2^\ell) + H\left(\mathbf{G}X_1^n | S_2^\ell\right)
\end{aligned}
$$

Now, if $\mathbf{G}X_1^n$ is a function of (Y_1^n, Y_2^n),

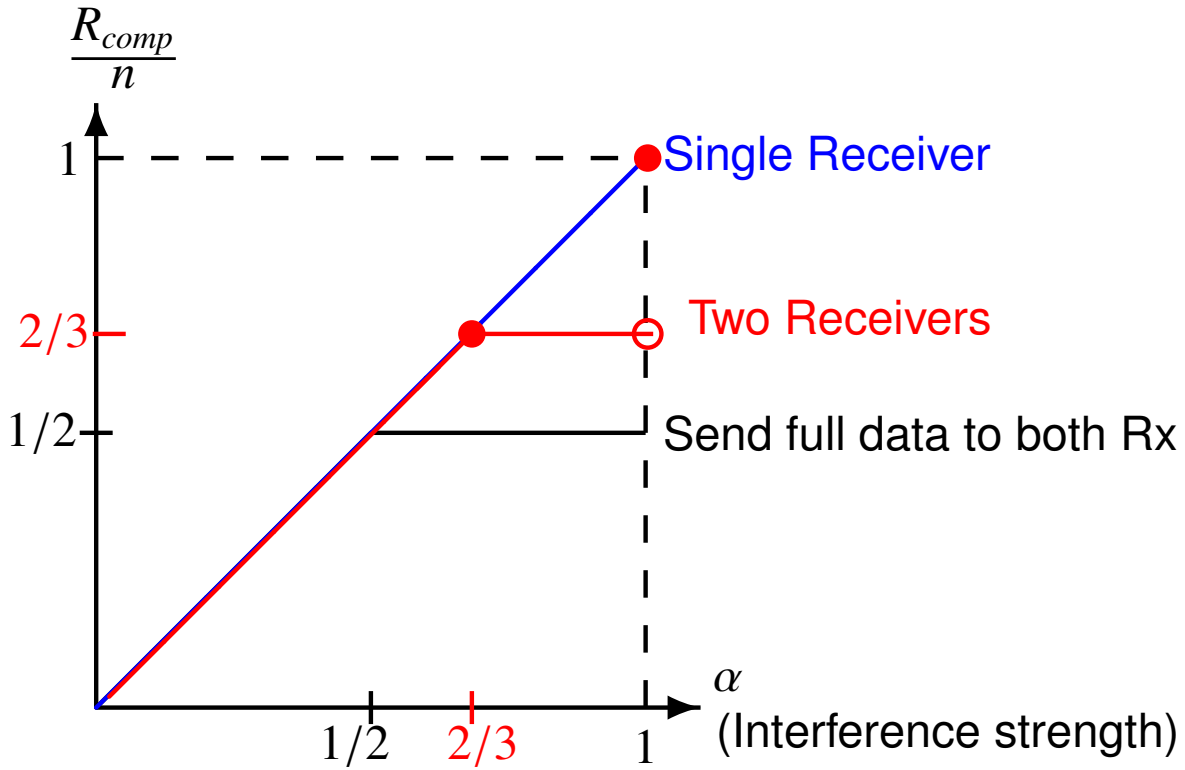
$$
\begin{aligned}
& = H(Y_1^n) + H(Y_2^n) \\
& \quad - H\left(Y_1^n, Y_2^n, \mathbf{G}X_1^n | S_1^\ell \oplus S_2^\ell\right) + H\left(\mathbf{G}X_1^n | S_2^\ell\right) \\
& \leq H(Y_1^n) + H(Y_2^n) \\
& \quad - H\left(\mathbf{G}X_1^n | S_1^\ell \oplus S_2^\ell\right) + H\left(\mathbf{G}X_1^n | S_2^\ell\right)
\end{aligned}
$$

Next, since X_1^n is a function of S_1^ℓ that is independent of $S_1^\ell \oplus S_2^\ell$

$$
\begin{aligned}
& = H(Y_1^n) + H(Y_2^n) - H\left(\mathbf{G}X_1^n\right) + H\left(\mathbf{G}X_1^n | S_2^\ell\right) \\
& \leq \sum_{i=1}^{n} [H(Y_{1i}) + H(Y_{2i})]
\end{aligned}
$$

Multiple Receivers

- Hence, we have:



Concluding Remarks

- Function computation in networks with many sources and a single receiver:
  - For all "Type Functions", the Körner-Marton approach, combined with the **Compute-and-Forward** approach, provides achievable computation rates.
  - Sometimes near-optimal, sometimes optimal.

  Sang-Woon Jeon, Chien-Yi Wang, M. G. "Computation Over Gaussian Networks With Orthogonal Components." arXiv:1310.7112 [cs.IT], October 2013.

  Jiening Zhan, Se Yong Park, M. G., and Anant Sahai "Linear Function Computation in Networks: Duality and Constant Gap Results." *IEEE Journal on Selected Areas in Communications,* April 2013.

- Function computation in networks with many sources and receivers:
  - There is generally a tension between the demands of the receivers.
  - "Computation alignment" is required for optimal performance (at least, in some examples).

  Changho Suh, Naveen Goela, M. G. "Computation in Multicast Networks: Function Alignment and Converse Theorems." arXiv:1209.3358 [cs.IT], October 2012.