Network-Channel Separation in Adversarial Networks

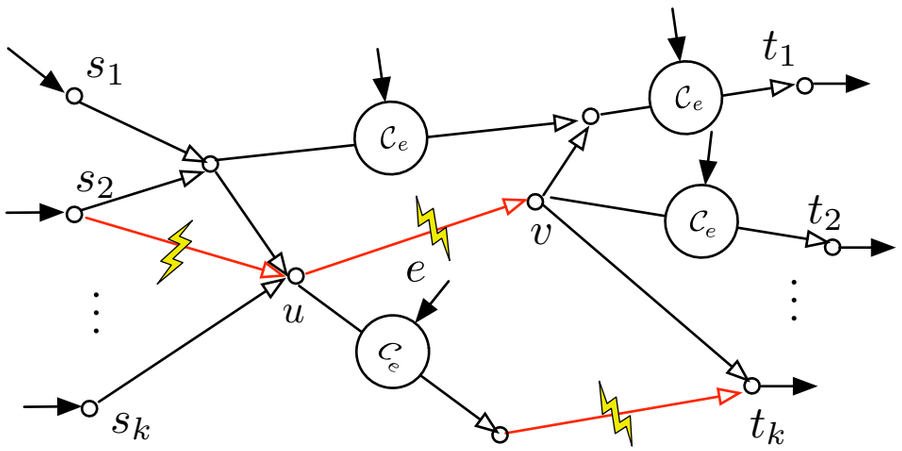
Mayank Bakshi

INC 網絡編碼研究所 Institute of Network Coding

Mar 22, 2012

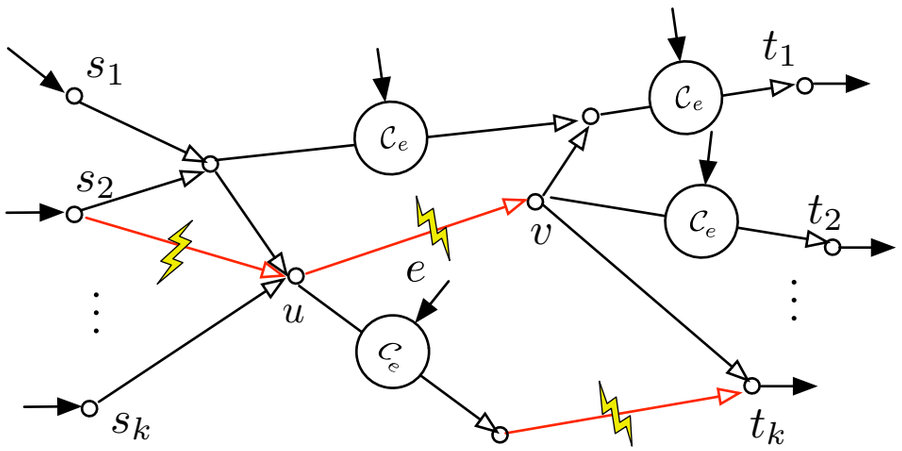
Joint work with

Prof Michelle Effros and Prof Tracey Ho
Department of Electrical Engineering, Caltech



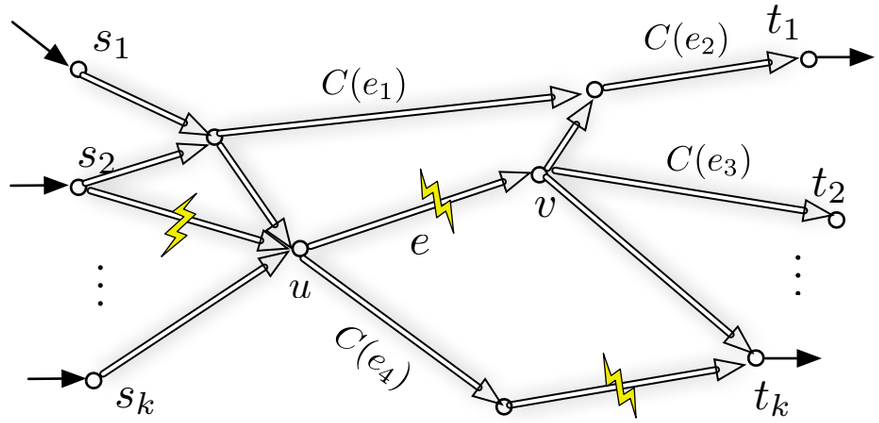
noisy channels + adversary

Q: Can Network Coding and Channel Coding be separated?

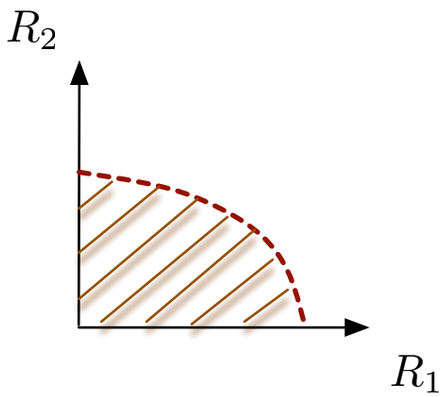


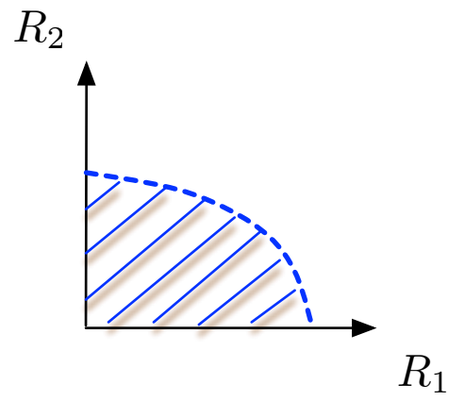
noisy channels + adversary

?



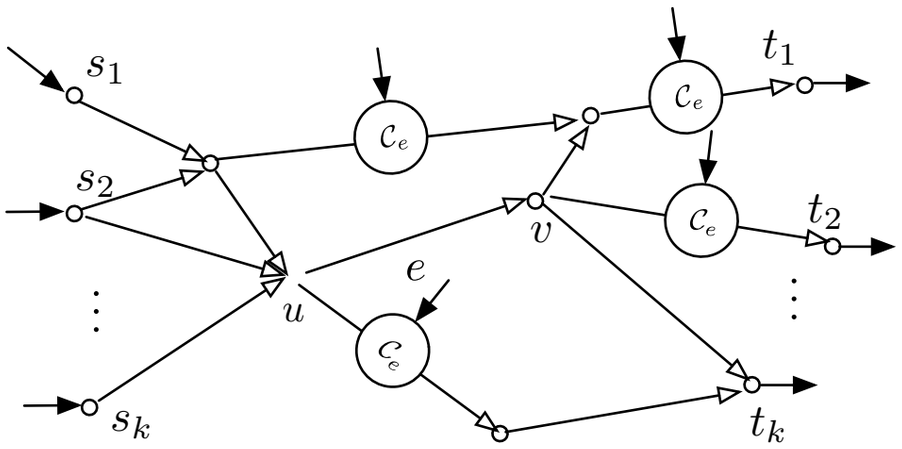
noiseless links + adversary

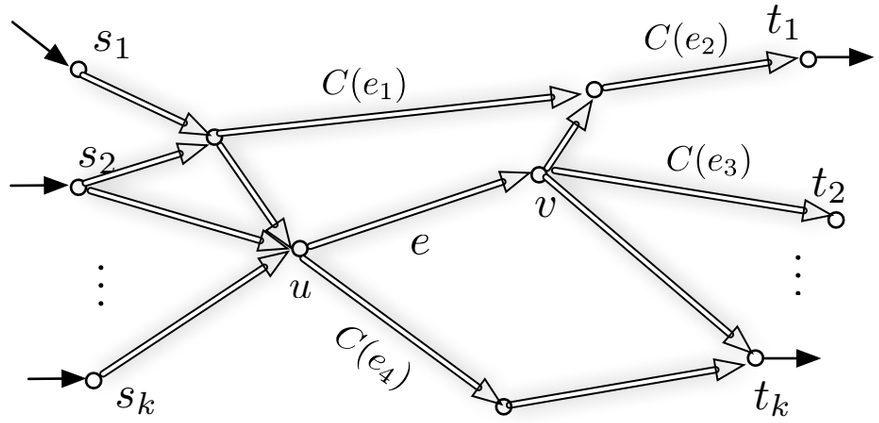




Q: Do the above two networks have same capacity regions?

[Koetter, Effros, Medard ‘2011, Song, Yeung, Cai ‘2006]:

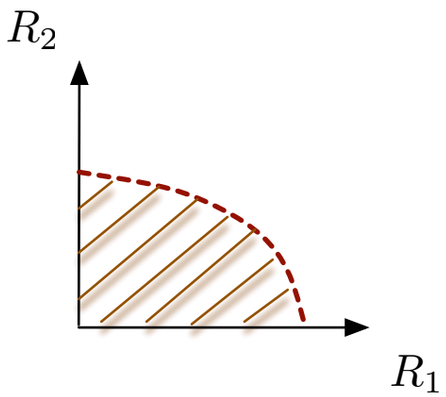


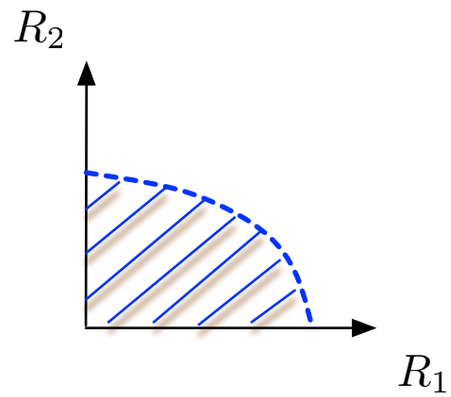


independent noisy channels

$\equiv$

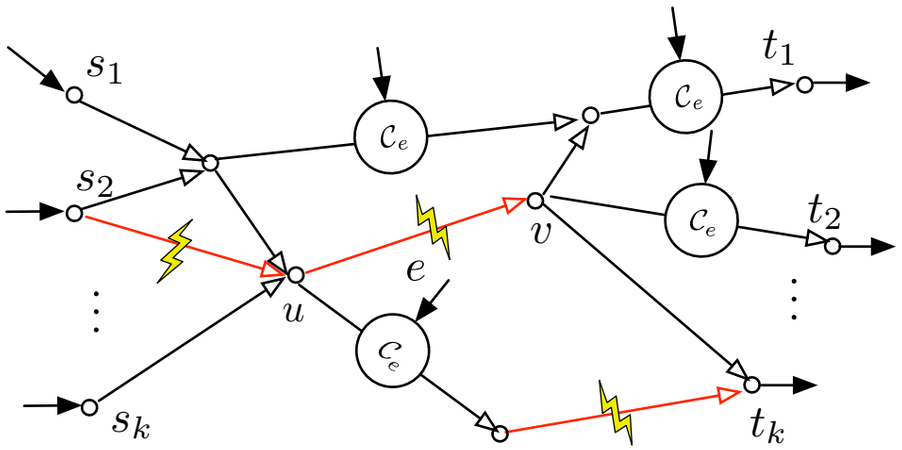
noiseless links





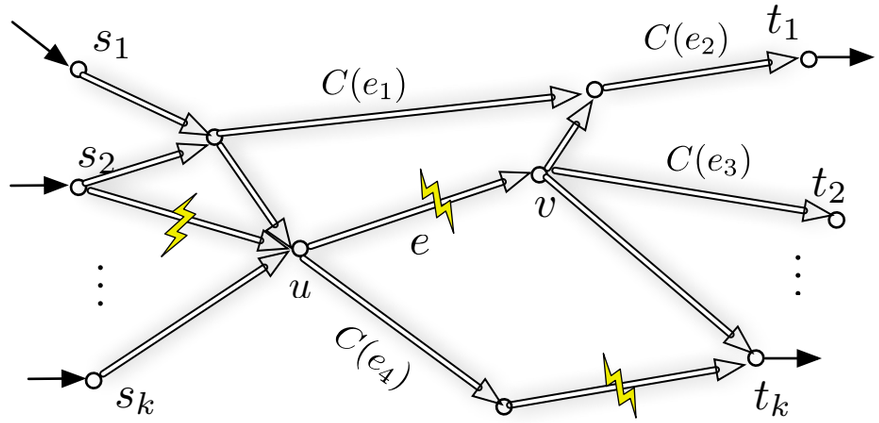
Previous works: Separation holds when there is no adversary

Motivation



noisy channels + adversary

?

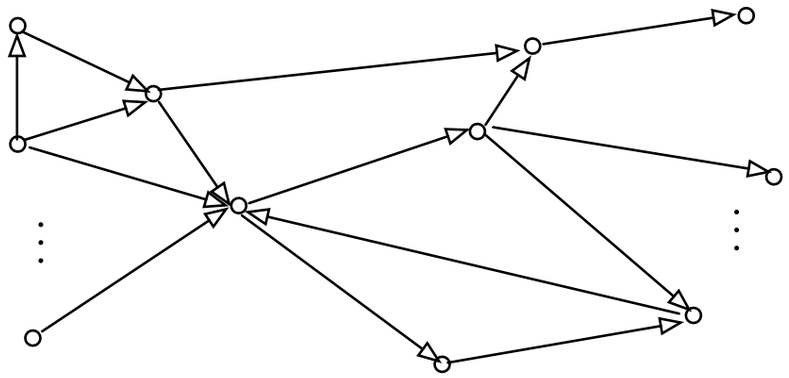


noiseless links + adversary

- Separation based strategies simpler to design and analyze

- Noiseless links + adversary better understood than noisy channels + adversary

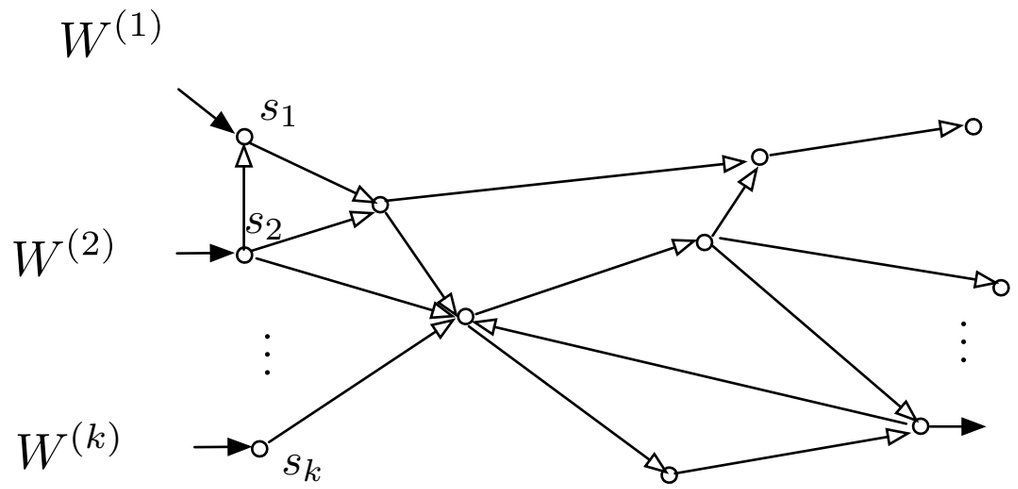
System Model



Network model:

- Directed edges (can be cyclic)

System Model

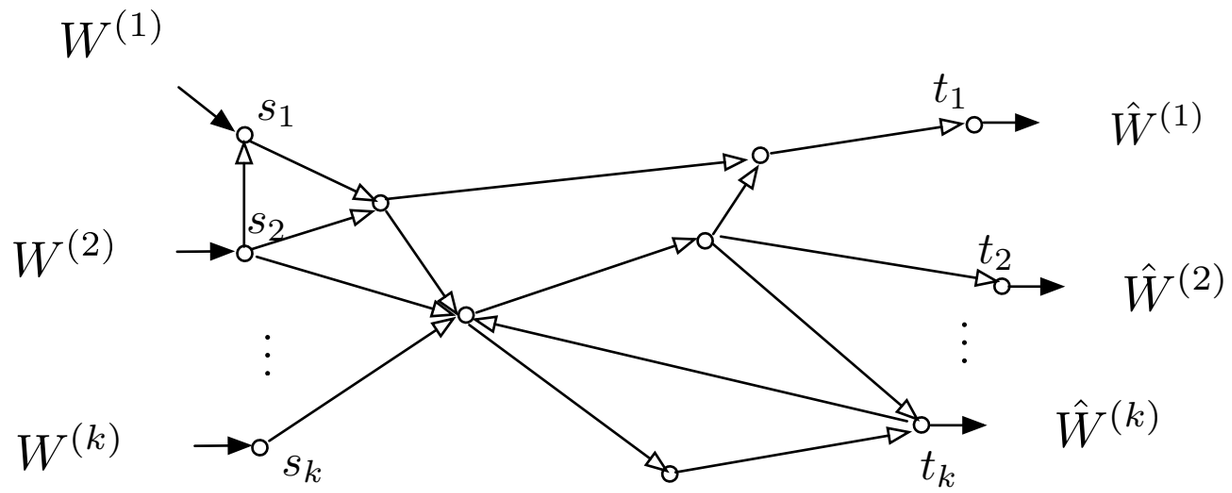


source nodes

Network model:

- Directed edges
- Independent sources

System Model



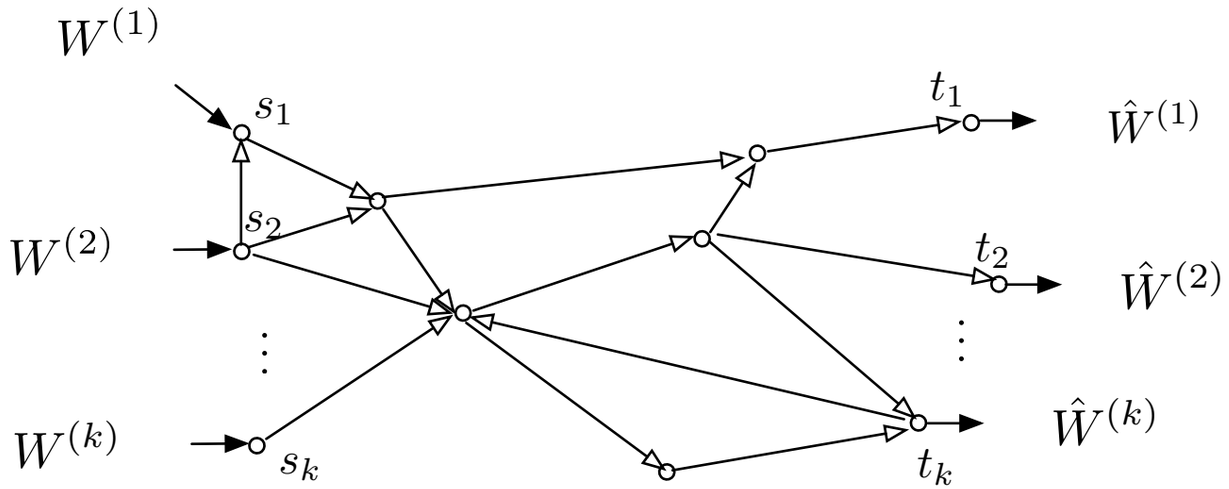
source nodes

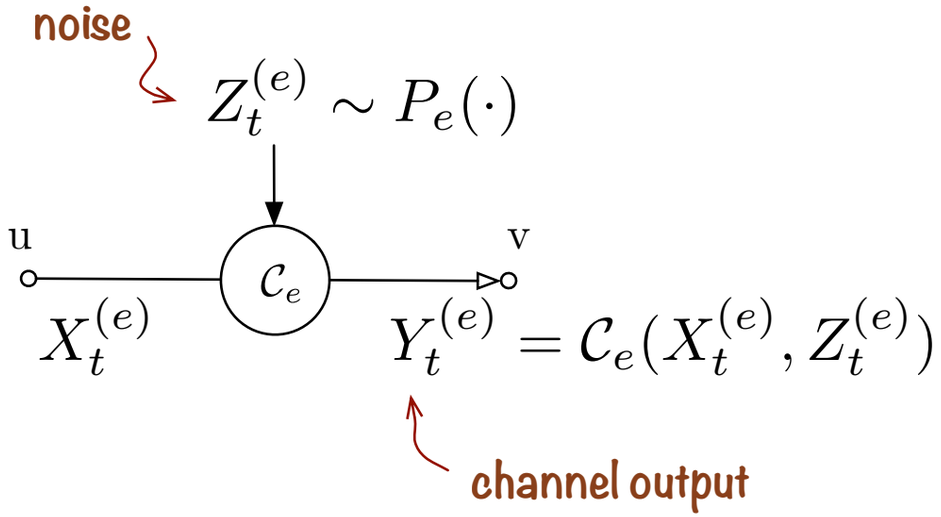
sink nodes

Network model:

- Directed edges
- Independent sources

System Model





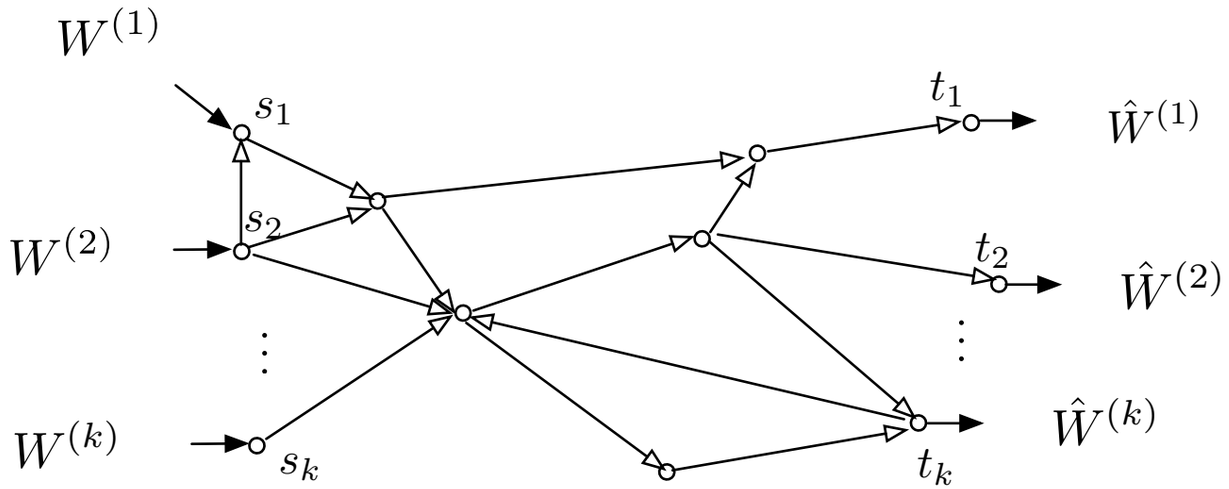
"Noisy Channel"

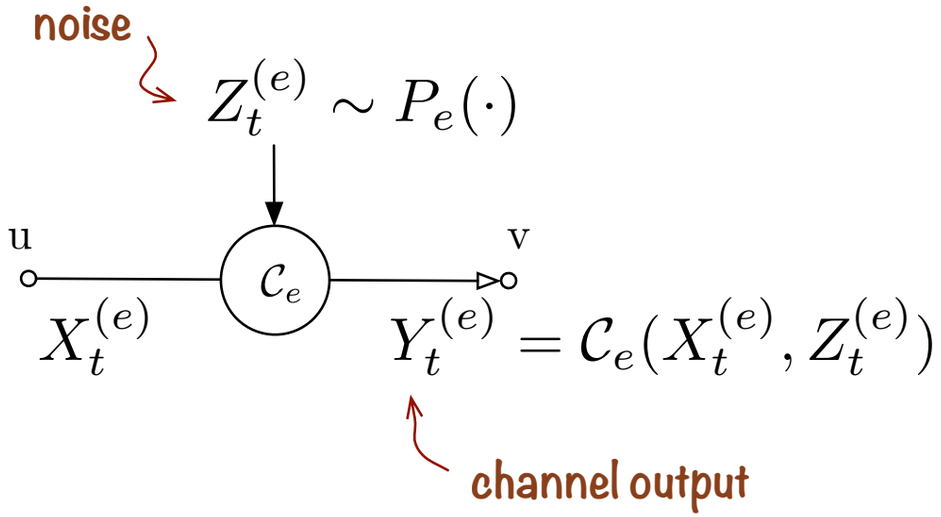
Network model:

- Directed edges
- Independent sources
- Independent, memoryless point-to-point channels

$$P(z_i^{(e)} : e \in E, i = 1, \ldots, n) = \prod_{e \in E} \prod_{i=1}^{n} P_e(z_i^{(e)})$$

System Model





"Noisy Channel"

Network model:

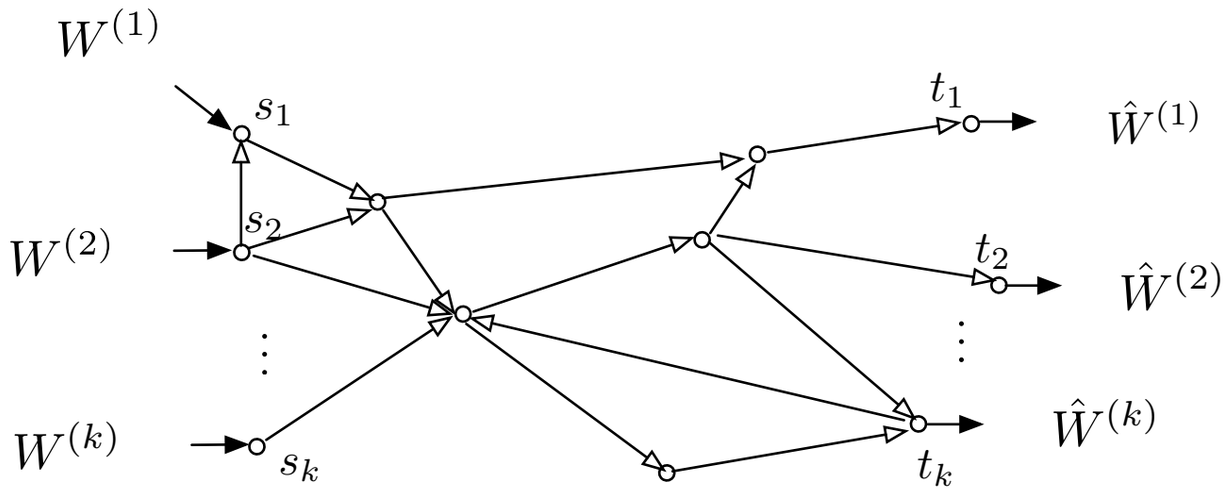
- Directed edges
- Independent sources
- Independent, memoryless point-to-point channels

$$P(z_i^{(e)} : e \in E, i = 1, \ldots, n) = \prod_{e \in E} \prod_{i=1}^{n} P_e(z_i^{(e)})$$

Channel capacity:

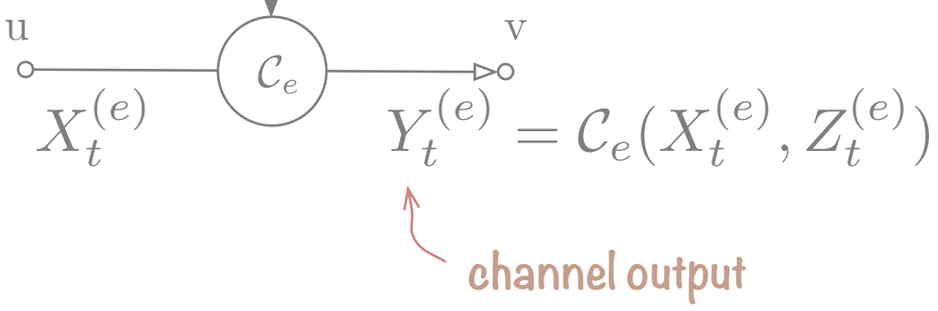
$$C(e) \triangleq \max_{P_{X^{(e)}}(\cdot)} I(X^{(e)}; Y^{(e)})$$

System Model



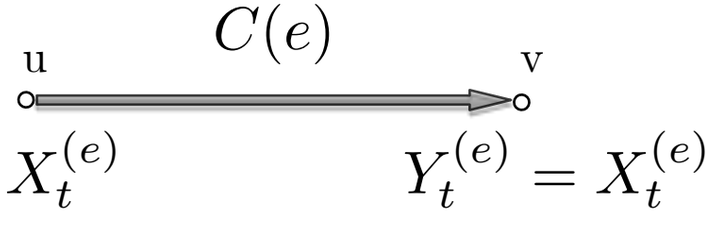
noise

$$Z_t^{(e)} \sim P_e(\cdot)$$



"Noisy Channel"

Special case:



"Noiseless Channel" – $C(e)$ bits/channel use

$$X_t^{(e)}, Y_t^{(e)} \in \{0,1\}^{C(e)} \ \forall\ t$$

Network model:

- Directed edges
- Independent sources
- Independent, memoryless point-to-point channels

$$P(z_i^{(e)} : e \in E, i = 1, \ldots, n) = \prod_{e \in E} \prod_{i=1}^{n} P_e(z_i^{(e)})$$

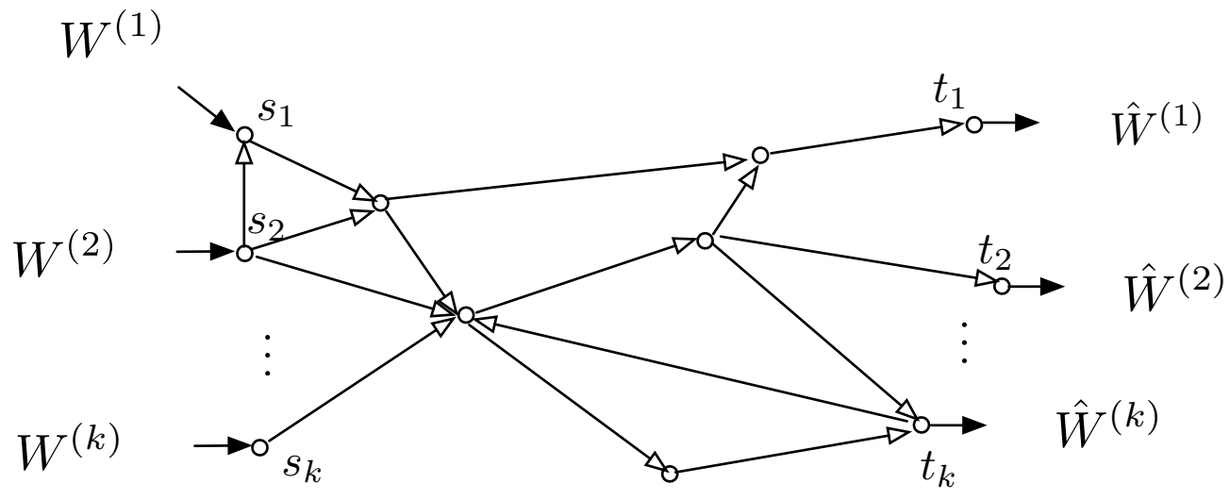
Channel capacity:

$$C(e) \triangleq \max_{P_{X^{(e)}}(\cdot)} I(X^{(e)}; Y^{(e)})$$

Assume (for now):

$$C(e) \in \mathbb{Z}^+ \quad \forall\ e \in E$$

System Model

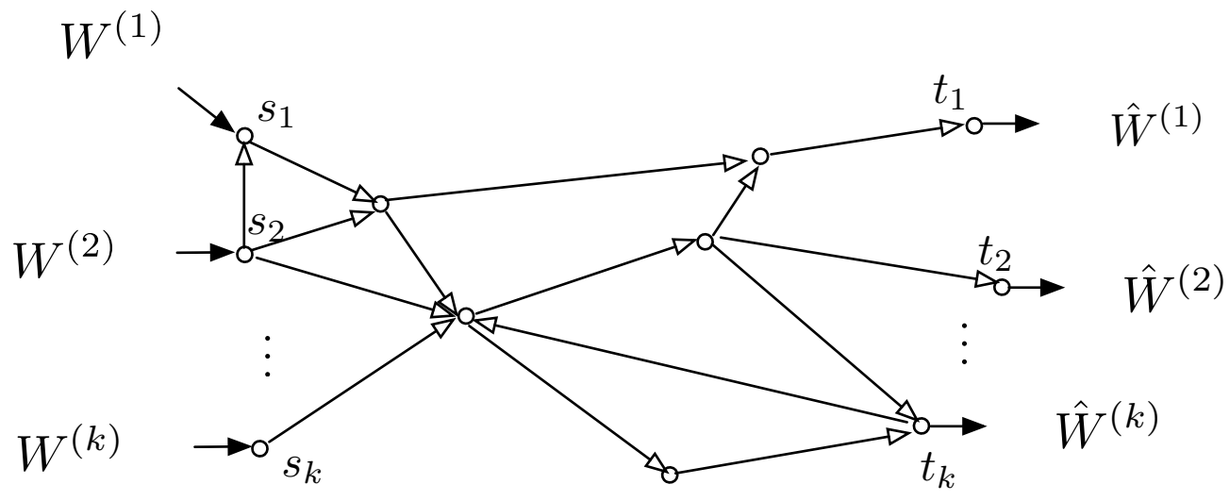


Network Code:

System Model

rate

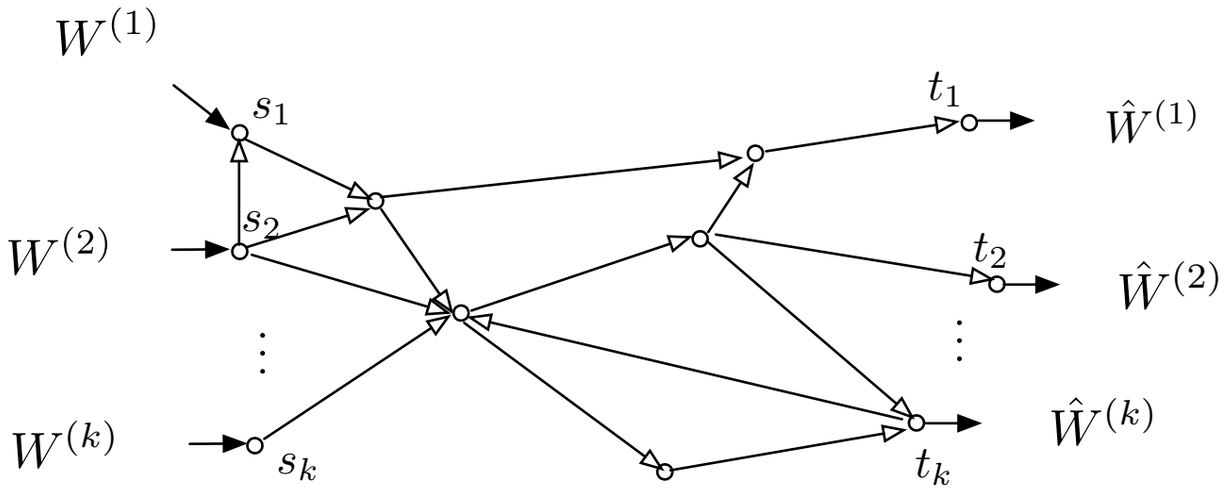
$W^{(i)} \in \{1, 2, \ldots, 2^{nR_i}\}$



Network Code:

- Block network code
  - blocklength $n \in \mathbb{Z}^+$, rate $\mathbf{R} = (R_1, \ldots, R_k)$

System Model



rate

$W^{(i)} \in \{1, 2, \ldots, 2^{nR_i}\}$

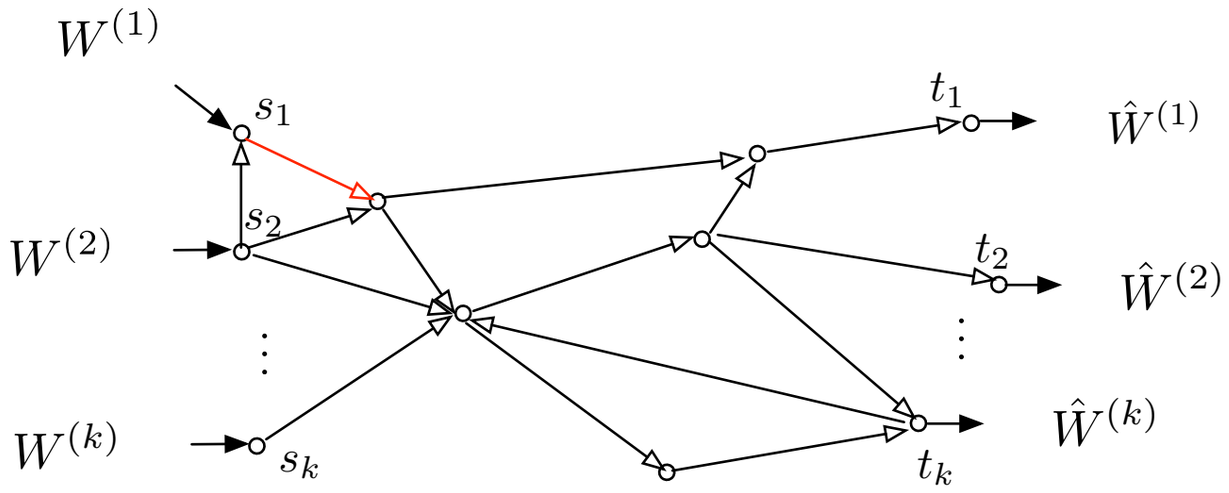
Network Code:

- Block network code
  - blocklength $n \in \mathbb{Z}^+$, rate $\mathbf{R} = (R_1, \ldots, R_k)$
  - mappings $(f_t^{(e)} : e \in E, t = 1, 2, \ldots, n)$
    $(g^{(i)} : i = 1, 2, \ldots, k)$

System Model

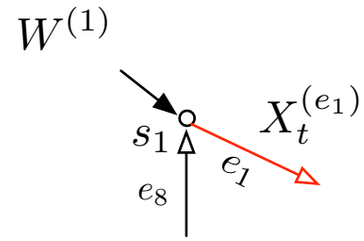
rate

$W^{(i)} \in \{1, 2, \ldots, 2^{nR_i}\}$



Network Code:

- Block network code
  - blocklength $n \in \mathbb{Z}^+$, rate $\mathbf{R} = (R_1, \ldots, R_k)$
  - mappings $(f_t^{(e)} : e \in E, t = 1, 2, \ldots, n)$
    $(g^{(i)} : i = 1, 2, \ldots, k)$

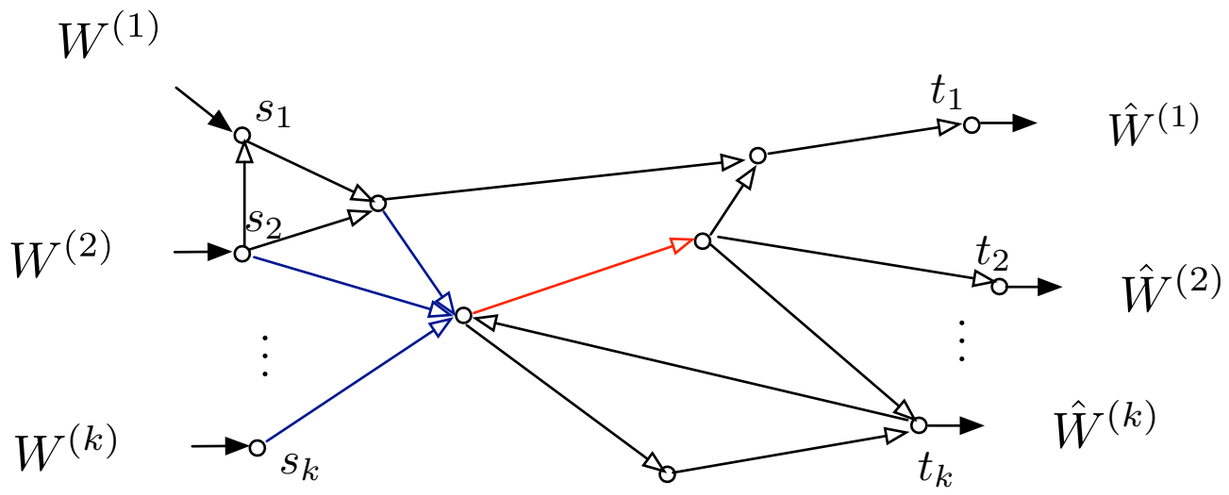


$$X_t^{(e_1)} = f_t^{(e_1)}(W^{(1)}, Y_{1:t-1}^{(e_8)}) \ , \ t = 1, 2, \ldots, n$$

System Model

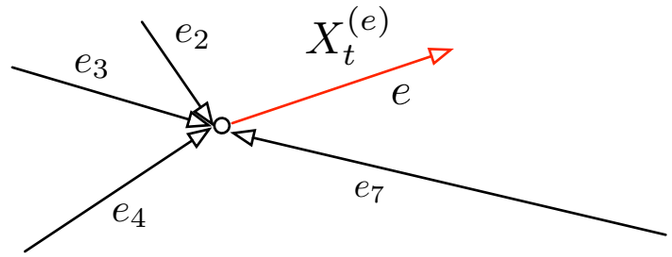
rate

$W^{(i)} \in \{1, 2, \ldots, 2^{nR_i}\}$



Network Code:

- Block network code
  - blocklength $n \in \mathbb{Z}^+$, rate $\mathbf{R} = (R_1, \ldots, R_k)$
  - mappings $(f_t^{(e)} : e \in E, t = 1, 2, \ldots, n)$
    $(g^{(i)} : i = 1, 2, \ldots, k)$

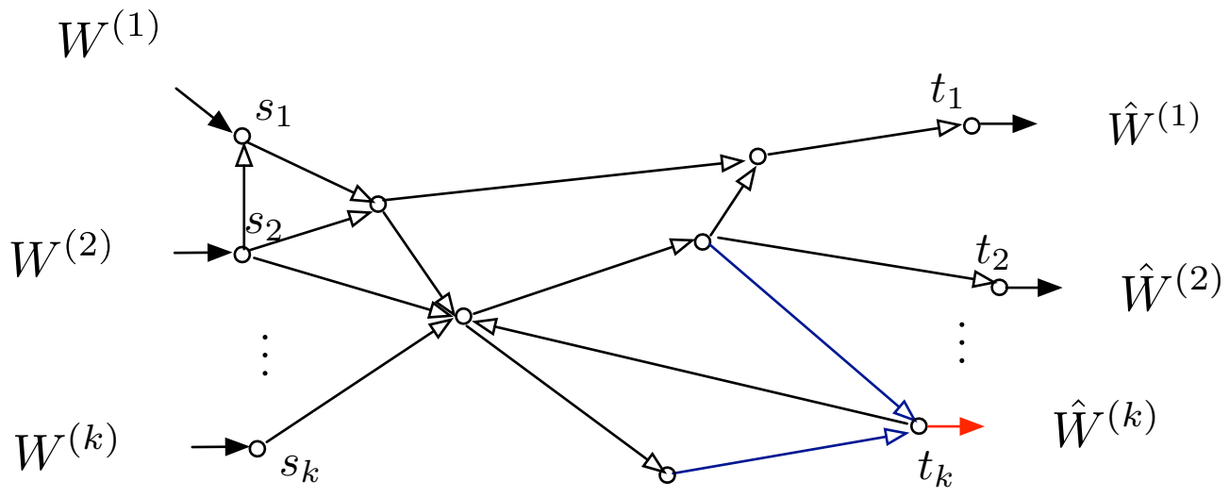


$$X_t^{(e)} = f_t^{(e_1)}(Y_{1:t-1}^{(e_2)}, Y_{1:t-1}^{(e_3)}, Y_{1:t-1}^{(e_4)}, Y_{1:t-1}^{(e_8)})$$

System Model

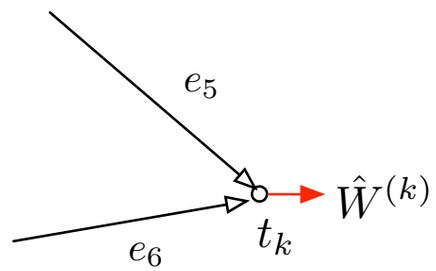
rate

$W^{(i)} \in \{1, 2, \ldots, 2^{nR_i}\}$



Network Code:

- Block network code
  - blocklength $n \in \mathbb{Z}^+$, rate $\mathbf{R} = (R_1, \ldots, R_k)$
  - mappings $(f_t^{(e)} : e \in E, t = 1, 2, \ldots, n)$
    $(g^{(i)} : i = 1, 2, \ldots, k)$

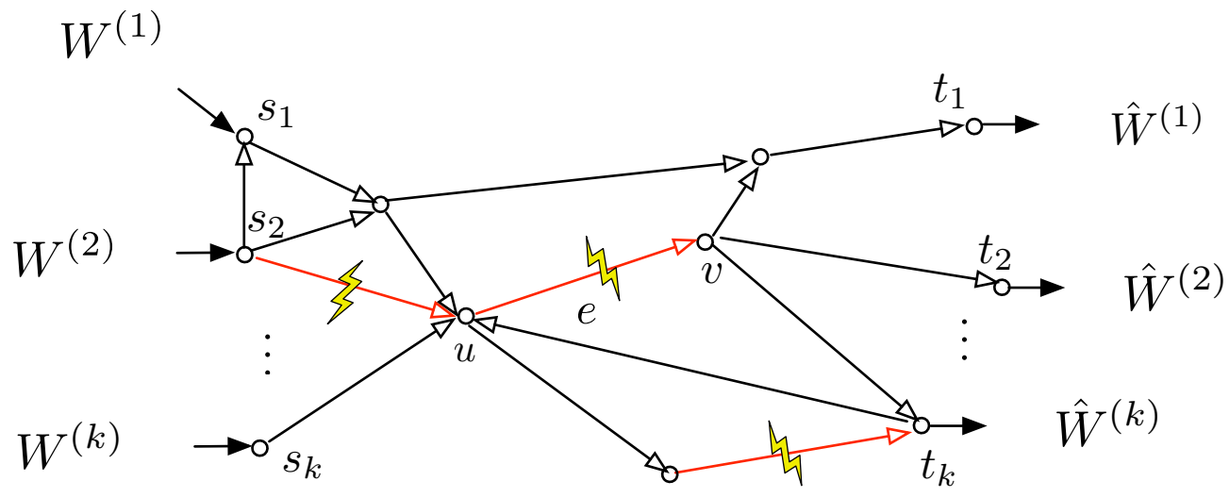


$\hat{W}^{(k)} = g^{(k)}(Y_{1:n}^{(e_5)}, Y_{1:n}^{(e_6)})$

System Model

rate

$W^{(i)} \in \{1, 2, \ldots, 2^{nR_i}\}$

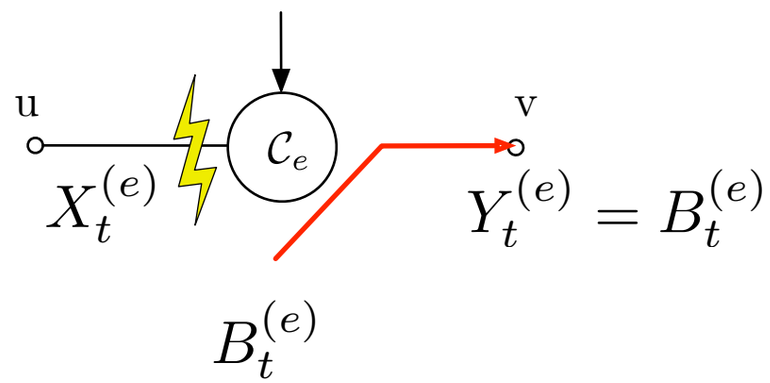


Network Code:

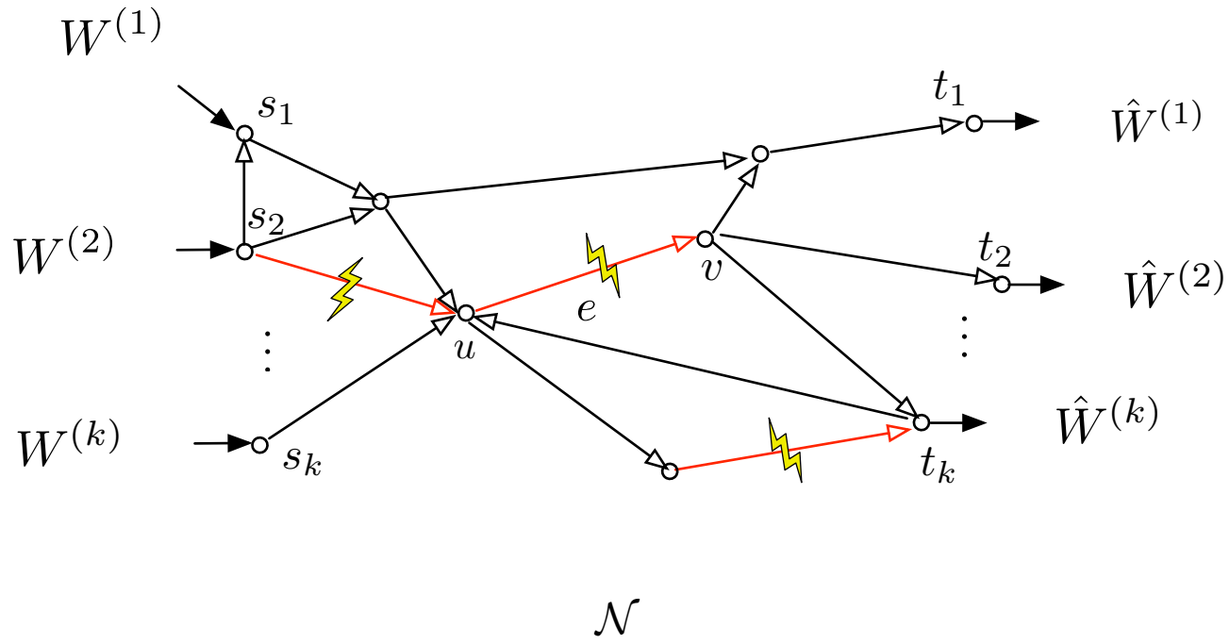
- Block network code
  - blocklength $n \in \mathbb{Z}^+$, rate $\mathbf{R} = (R_1, \ldots, R_k)$
  - mappings $(f_t^{(e)} : e \in E, t = 1, 2, \ldots, n)$
    $(g^{(i)} : i = 1, 2, \ldots, k)$

Byzantine Adversary:

- knows all **messages** and **noise values**
- can replace **received** vectors on any K edges



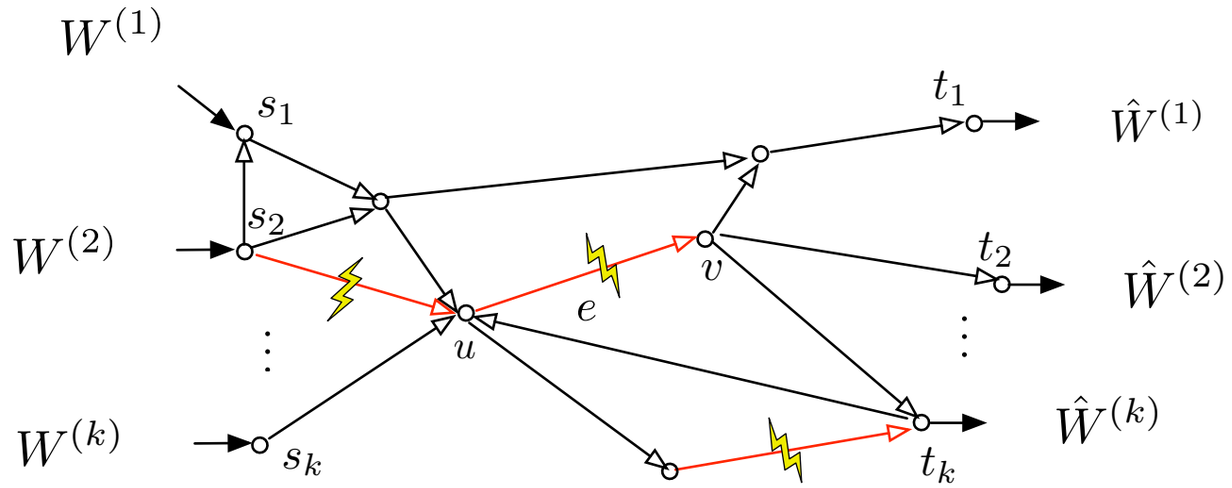
System Model



Communication goal:

"Reliably" communicate $W^{(1)}, W^{(2)}, \ldots, W^{(k)}$ ***irrespective of adversary's attack strategy***

System Model



$\mathcal{N}$

Communication goal:

"Reliably" communicate $W^{(1)}, W^{(2)}, \ldots, W^{(k)}$ ***irrespective of adversary's attack strategy***

Rate vector:

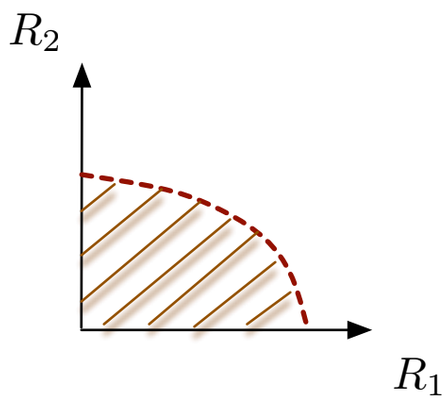
$$\mathbf{R} = (R(1), R(2), \ldots, R(k))$$

Capacity region $\mathcal{R}(\mathcal{N}, K)$

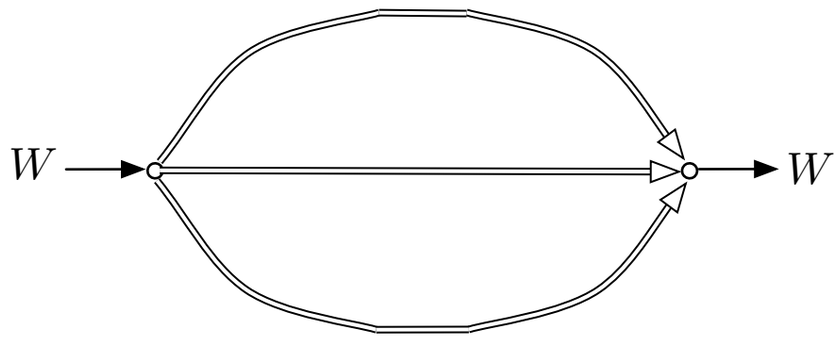
$\mathbf{R} \in \mathcal{R}(\mathcal{N}, K)$ iff there exists a network code for some block-length n such that

$$\Pr(W^{(1)}, W^{(2)}, \ldots \neq \hat{W}^{(1)}, \hat{W}^{(2)}, \ldots) < \lambda$$

for every $\lambda > 0$ ***irrespective of adversary's strategy.***



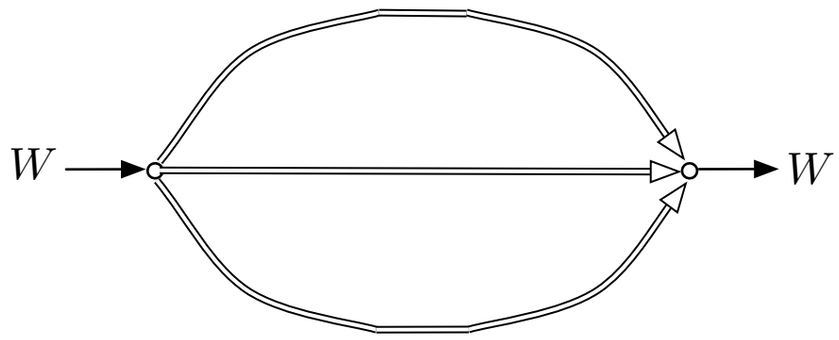
Example



- $C(e) = 1$ for each $e \in E$

- Adversary can attack any one edge

Example



- $C(e) = 1$ for each $e \in E$

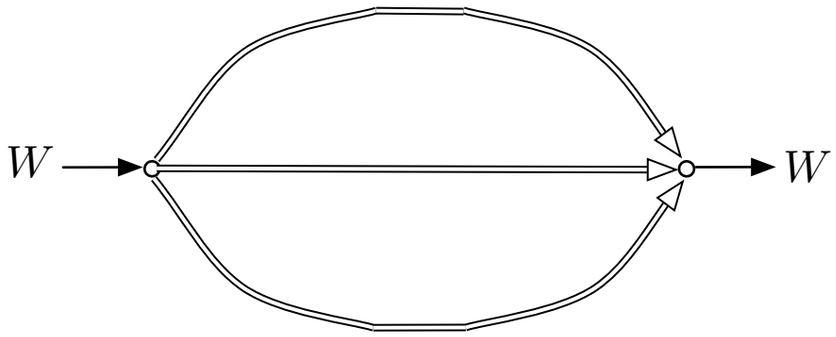
- Adversary can attack any one edge

- Rate 1 is achievable

Example

$W \in \{0, 1\}$

Blocklength =1



- $C(e) = 1$ for each $e \in E$

- Adversary can attack any one edge

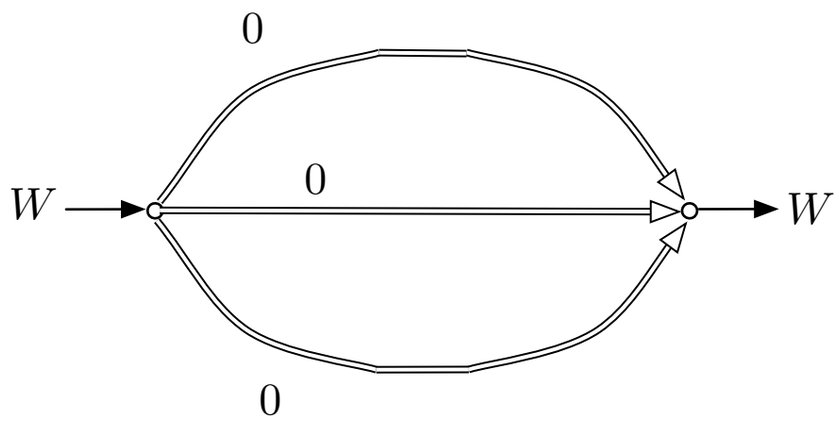
- Rate 1 is achievable

Example

$W = 0$

$W \in \{0, 1\}$

Blocklength =1



- $C(e) = 1$ for each $e \in E$

- Adversary can attack any one edge

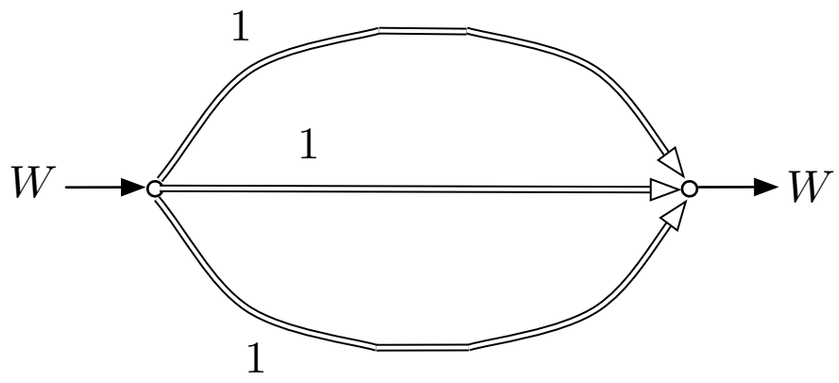
- Rate 1 is achievable

Example

$W = 1$

$W \in \{0, 1\}$

Blocklength =1



- $C(e) = 1$ for each $e \in E$

- Adversary can attack any one edge

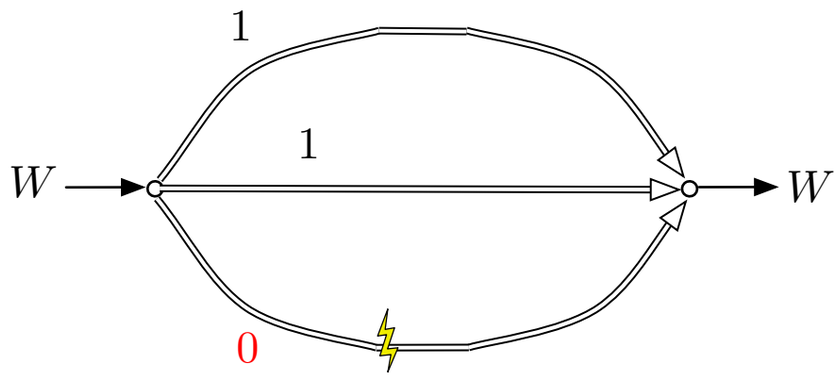
- Rate 1 is achievable

Example

$W = 1$

$W \in \{0, 1\}$

Blocklength =1

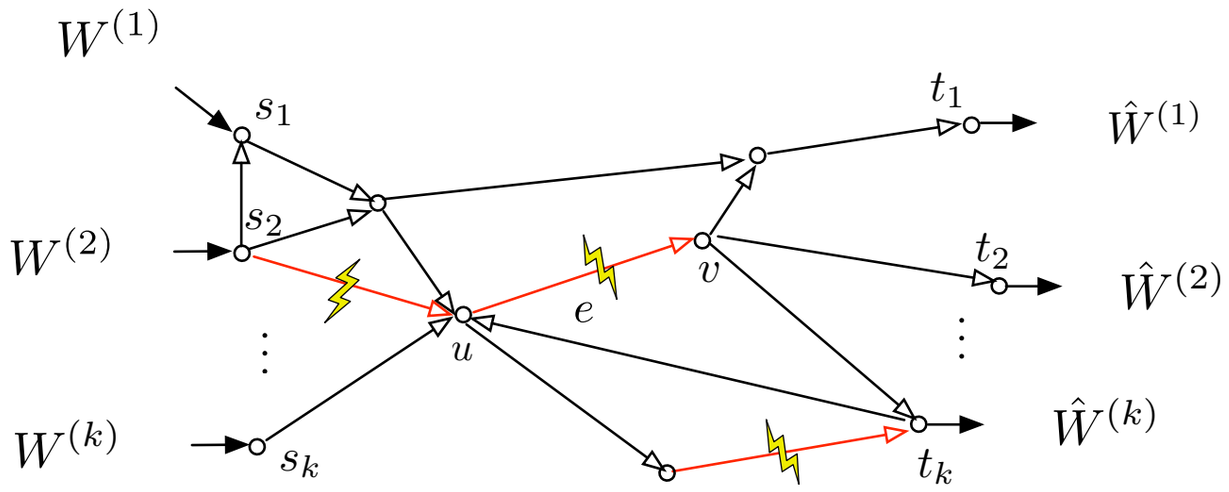


- $C(e) = 1$ for each $e \in E$

- Adversary can attack any one edge

- Rate 1 is achievable

Some known results

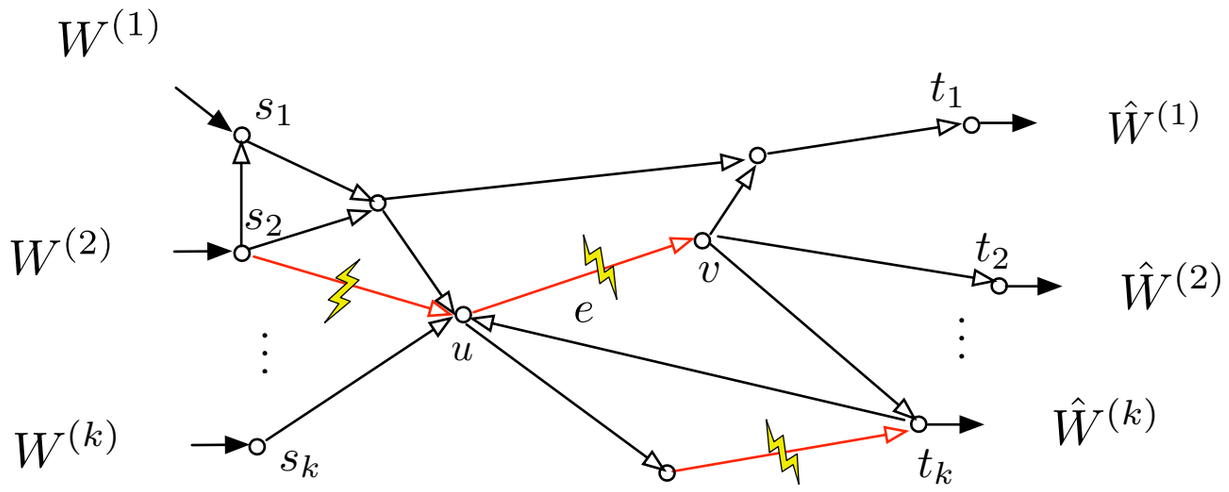


Noiseless networks

- equal link capacities [Yeung et al 2006, Jaggi et al 2007]

  - Capacity = min-cut- $2K$ if each link has capacity 1

Some known results



Noiseless networks

- equal link capacities [Yeung et al 2006, Jaggi et al 2007]

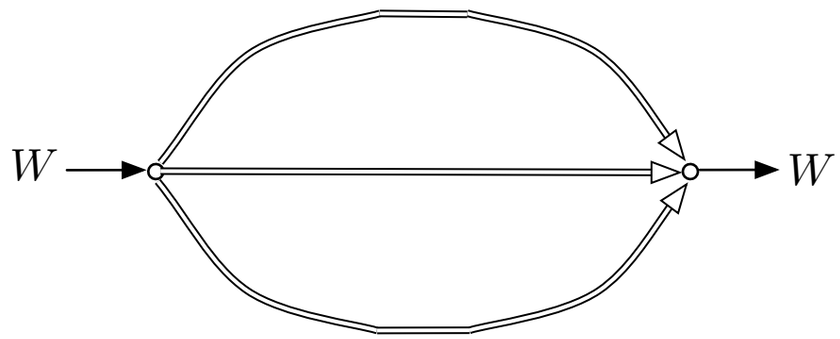
  - Capacity = min-cut- $2K$ if each link has capacity 1

    Previous example:

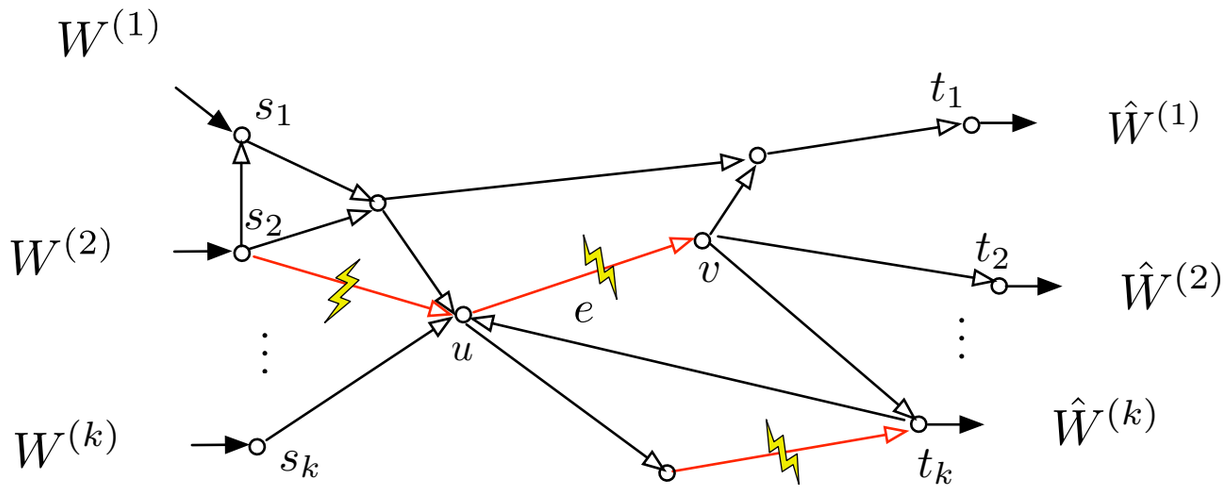
    min-cut = 3

    $K = 1$

    => *Capacity = 1*



Some known results



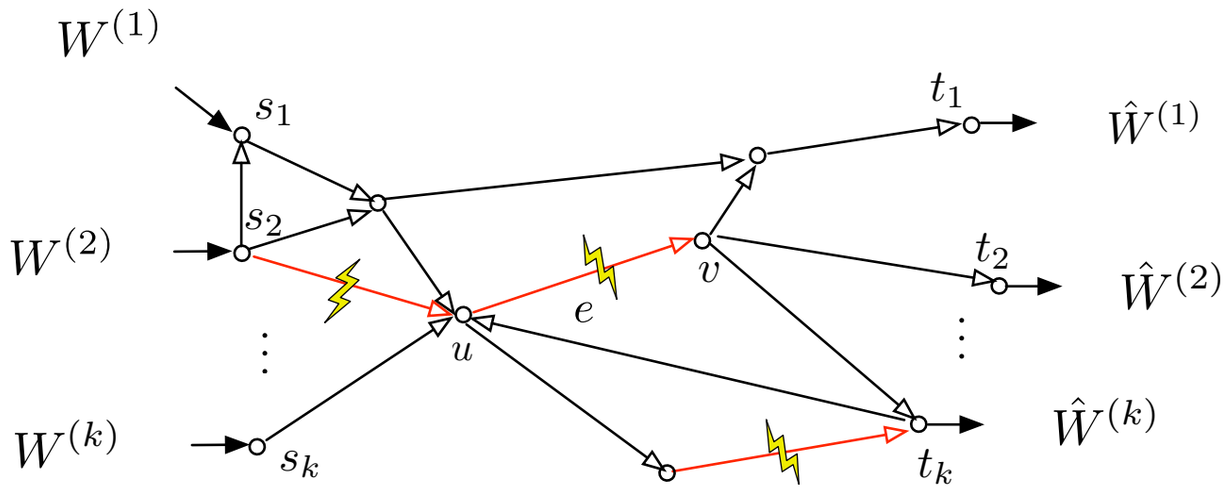
Noiseless networks

- equal link capacities [Yeung et al 2006, Jaggi et al 2007]

  - Capacity = min-cut- $2K$ if each link has capacity 1

- partial results for unequal link capacities [Kim et al 2009]

Some known results



Noiseless networks

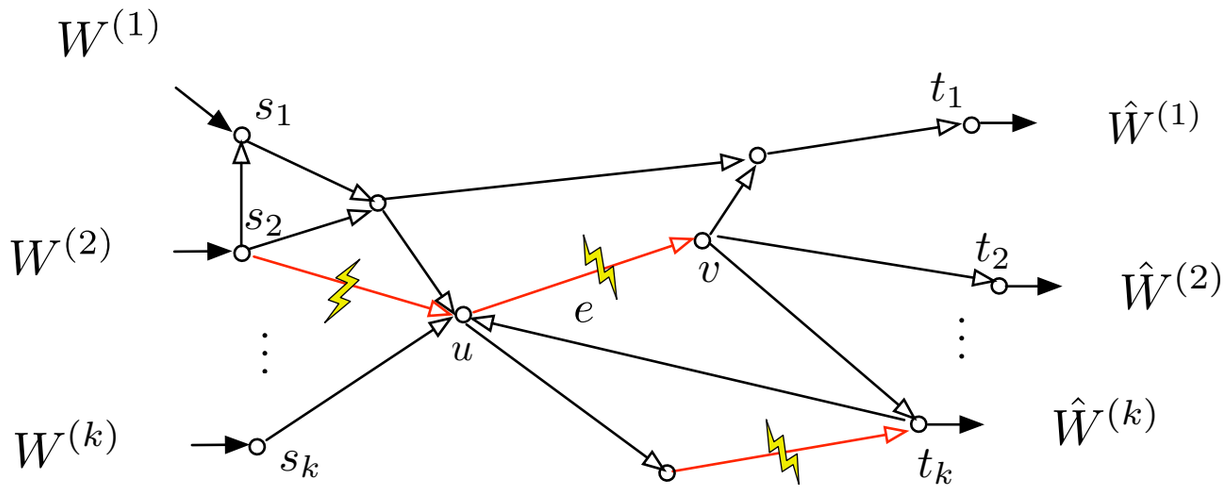
- equal link capacities [Yeung et al 2006, Jaggi et al 2007]

  - Capacity = min-cut- $2K$ if each link has capacity 1

- partial results for unequal link capacities [Kim et al 2009]

- node based adversaries [Kosut et al 2009]

Some known results



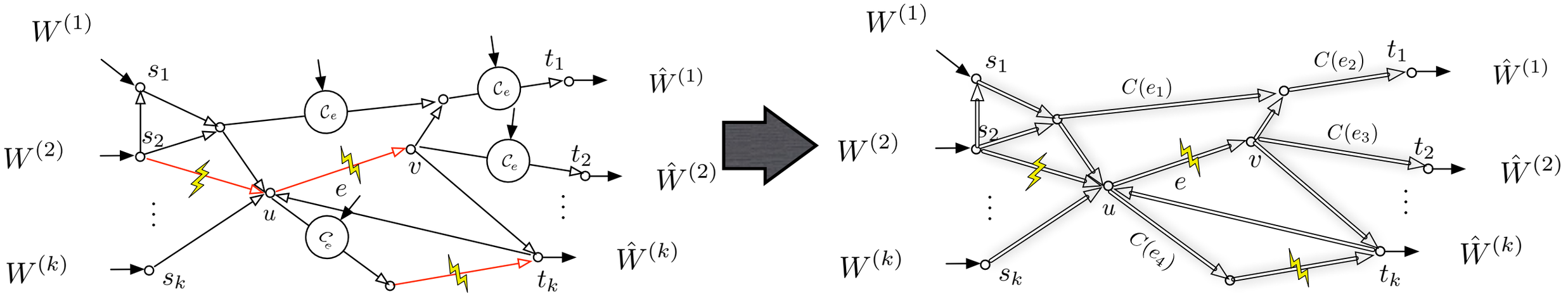
Noisy networks

[Jaggi et al 2007]

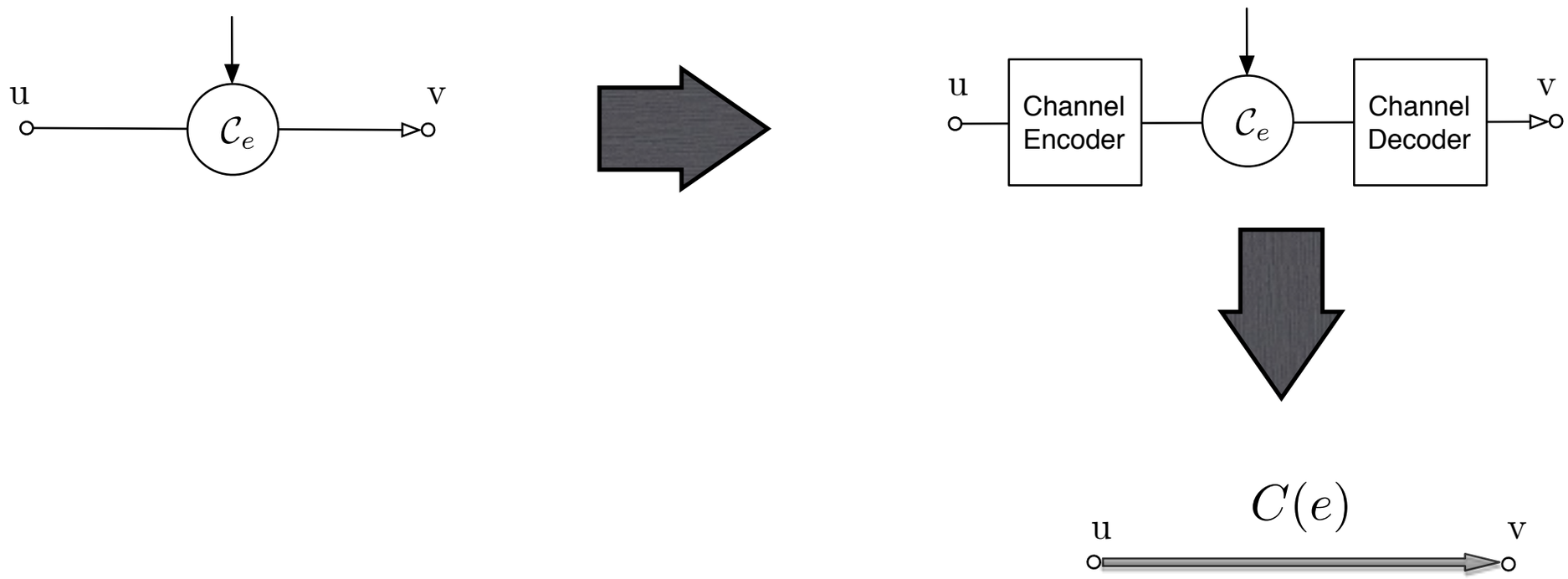
- Noisy problem harder to deal with

- Separating channel coding and network coding gives achievable schemes!

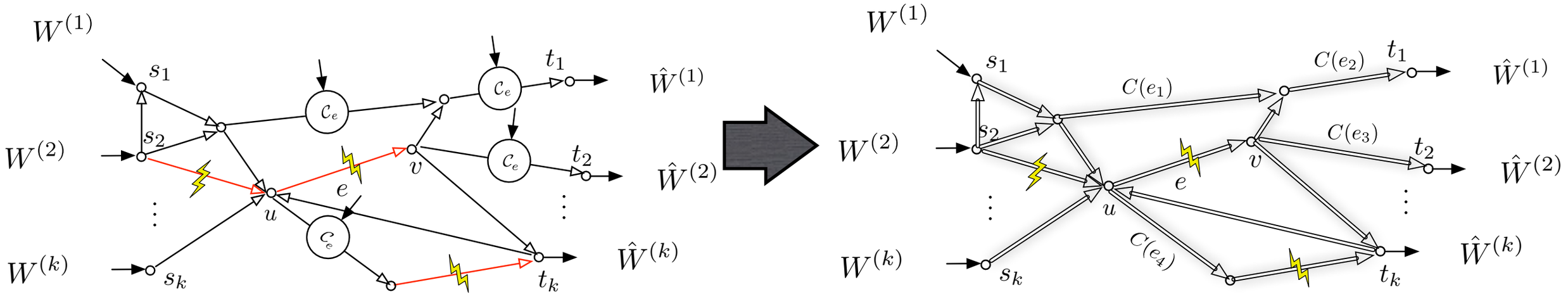
Separating network and channel coding



Step 1: Channel code on each link



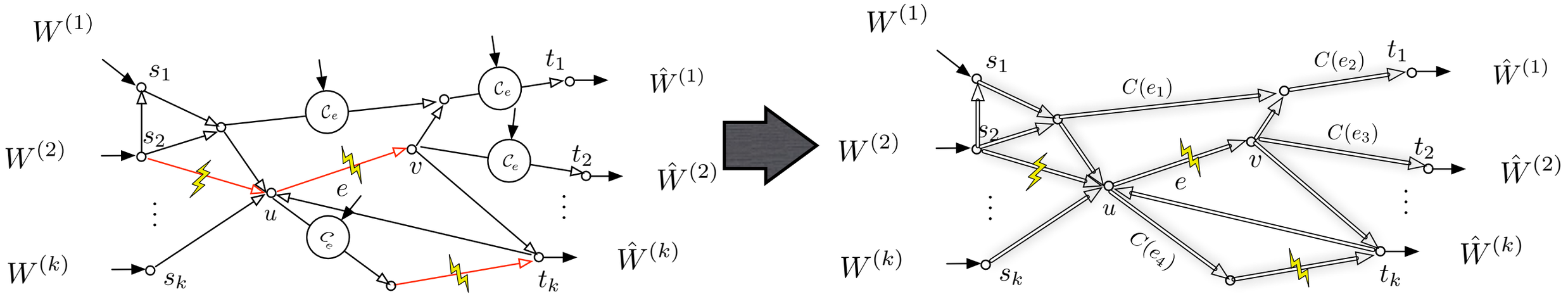
Separating network and channel coding



Step 1: Channel code on each link

Step 2: Network code for noiseless network => network code for noisy network

Separating network and channel coding

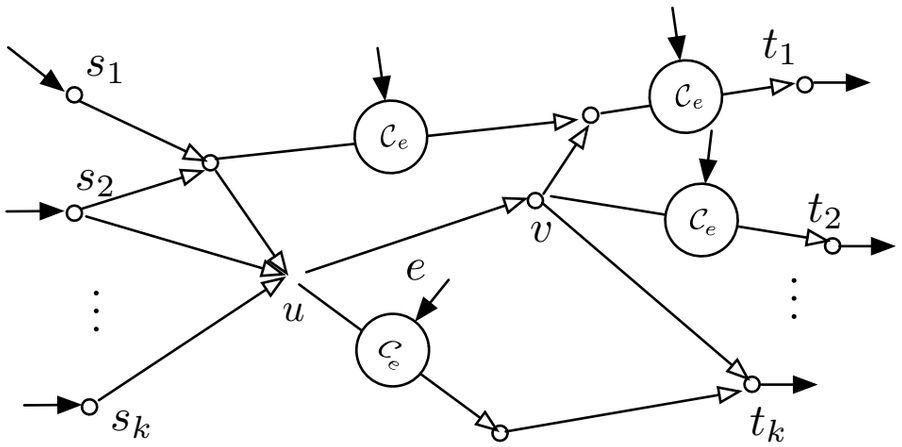


Step 1: Channel code on each link

Step 2: Network code for noiseless network => network code for noisy network

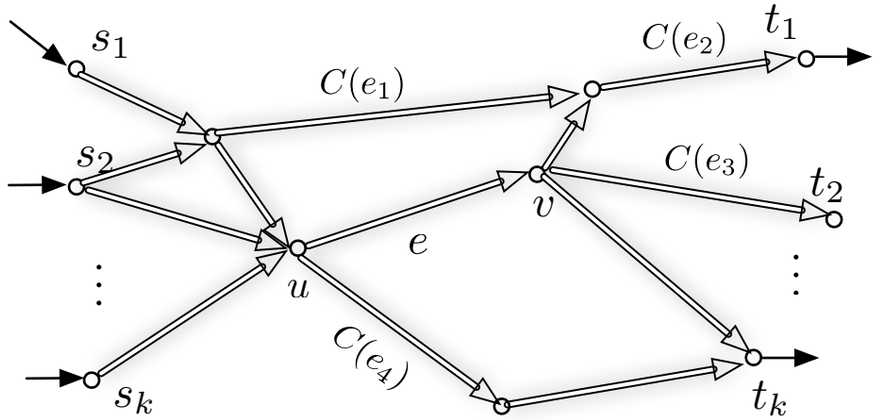
=> Achievable rate vectors for noisy networks

Separating network and channel coding



$\mathcal{N}$

≡



$\hat{\mathcal{N}}$

Noisy network *Noiseless network*

$$\mathcal{R}(\mathcal{N}, 0) = \mathcal{R}(\hat{\mathcal{N}}, 0)$$

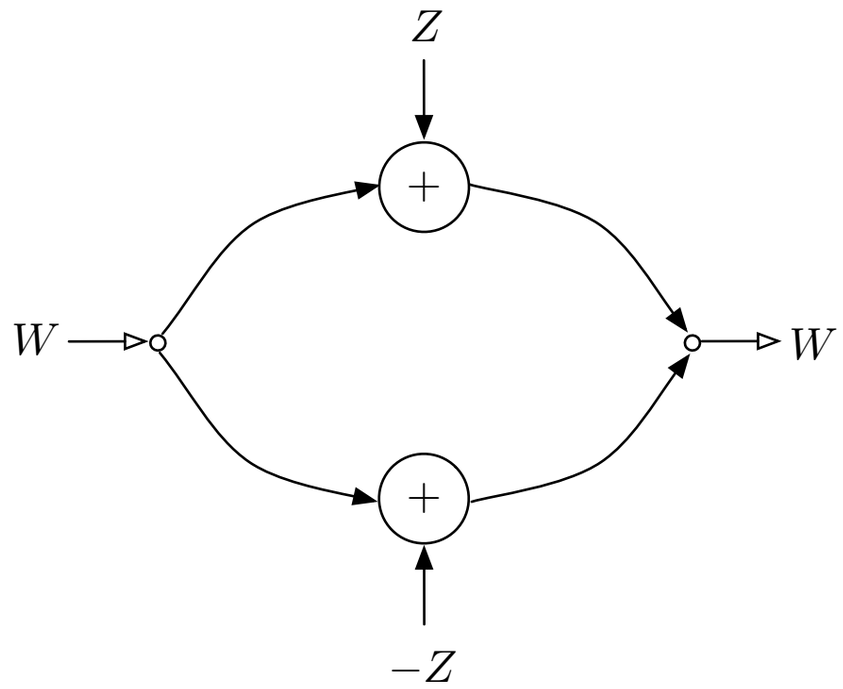
- Separation holds between network coding and channel coding for **independent** channels

- Networks with links of same capacities are "**equivalent**"

[Song, Yeung, Cai ' 2006] for single source multicast
[Koetter, Effros, Medard ' 2011] for general networks

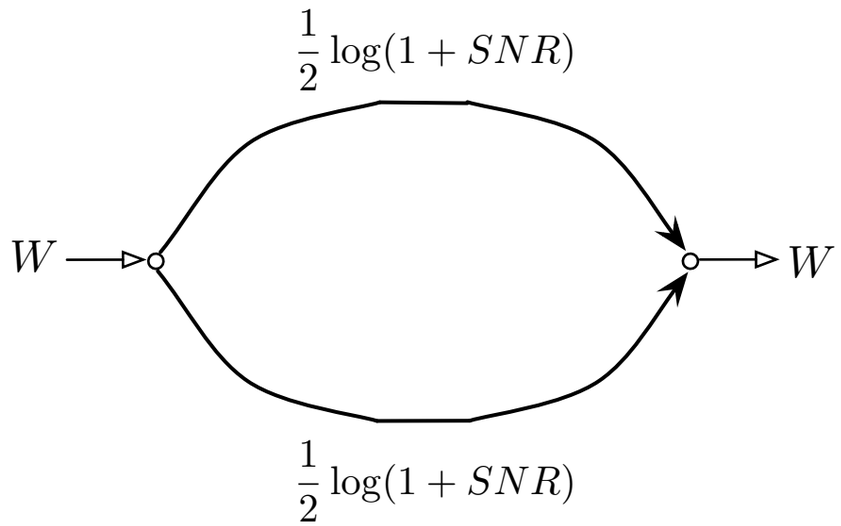
Separation always optimal?

Parallel AWGNs [Koetter et al 2011]



Separation always optimal?

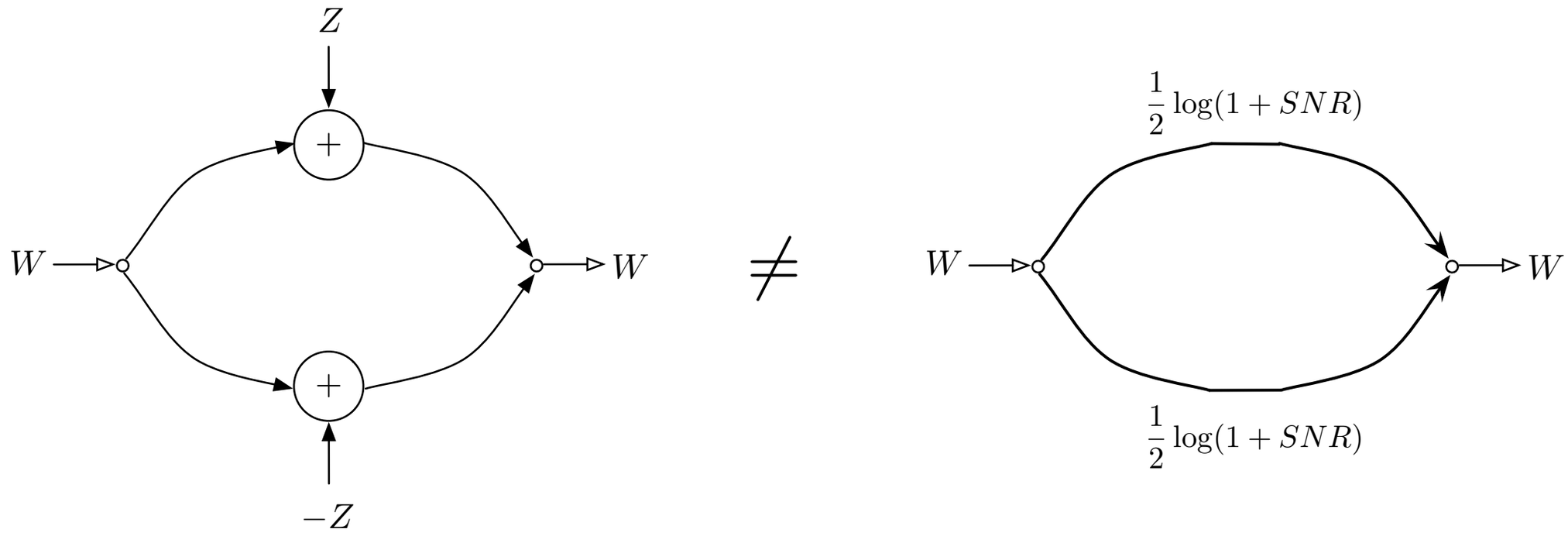
Parallel AWGNs [Koetter et al 2011]



- Separate coding on each link achieves at most $\log(1+SNR)$

Separation always optimal?

Parallel AWGNs [Koetter et al 2011]



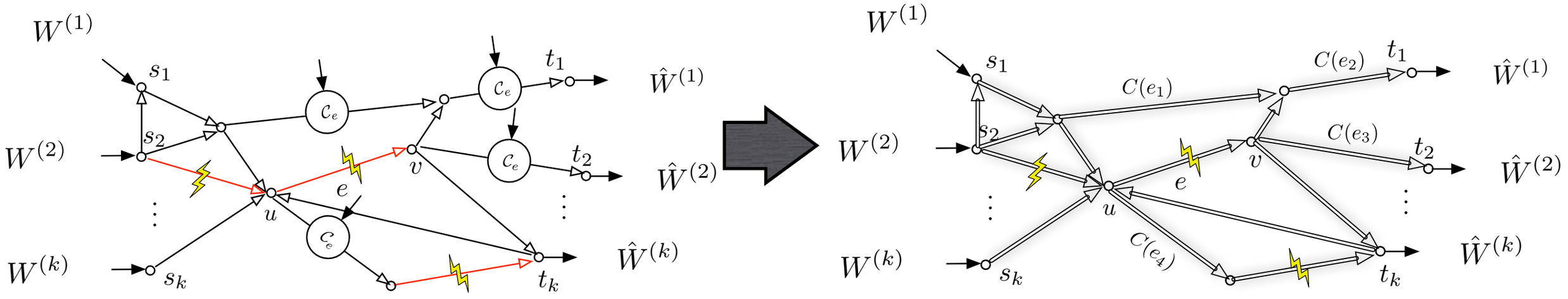
- Separate coding on each link achieves at most $\log(1+SNR)$

- Receiver can “cancel out” noise by adding the two received vectors

- Capacity is infinite

Separation may be **suboptimal** when channels **not independent**!

Separating network and channel coding



Step 1: Channel code on each link

Step 2: Network code for noiseless network => network code for noisy network

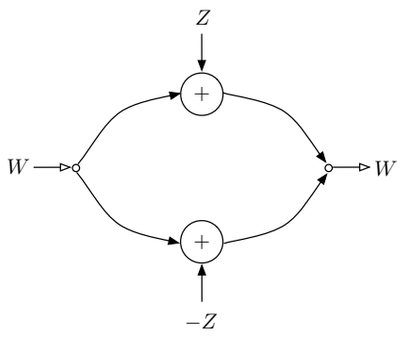
=> Achievable rate vectors for noisy networks

Possible problem: Adversarial noise is not independent on different edges

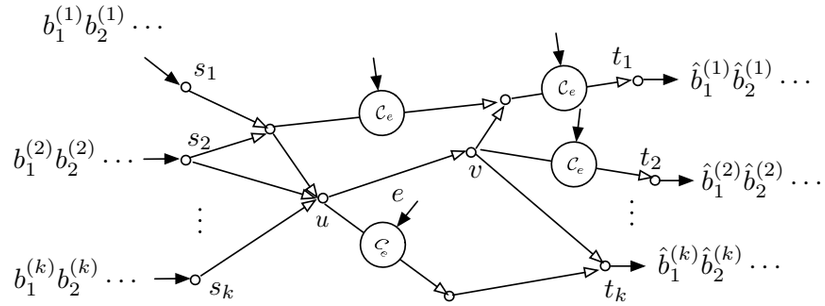
Separation suboptimal

Separation optimal

Networks with jointly distributed noise



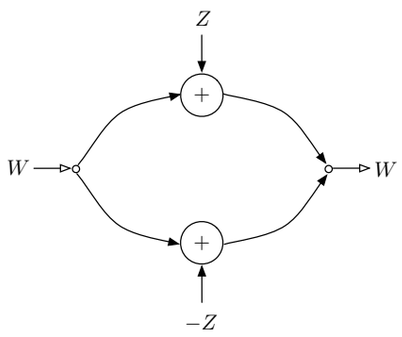
Networks of independent channels



Separation suboptimal

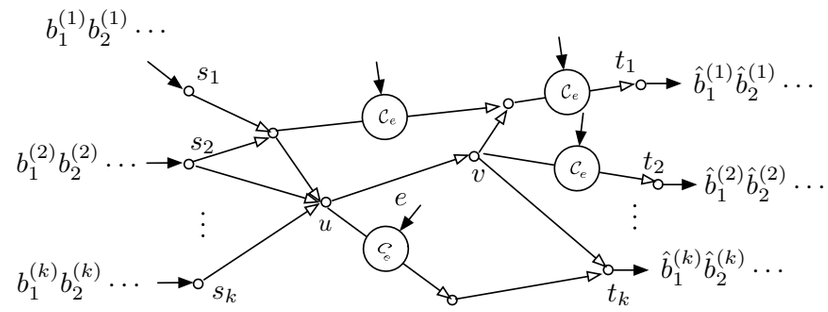
Separation optimal

Networks with jointly distributed noise



?

Networks of independent channels

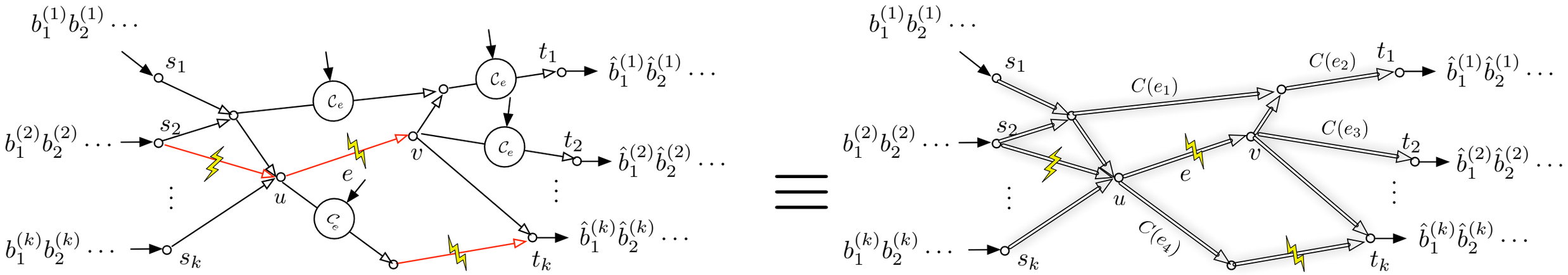


Adversarial Networks?

- independent channels
- adversary may introduce “noise” that is not independent

Our work: Is separation optimal for adversarial networks?

Main Result



$\mathcal{N}$ $\equiv$ $\hat{\mathcal{N}}$

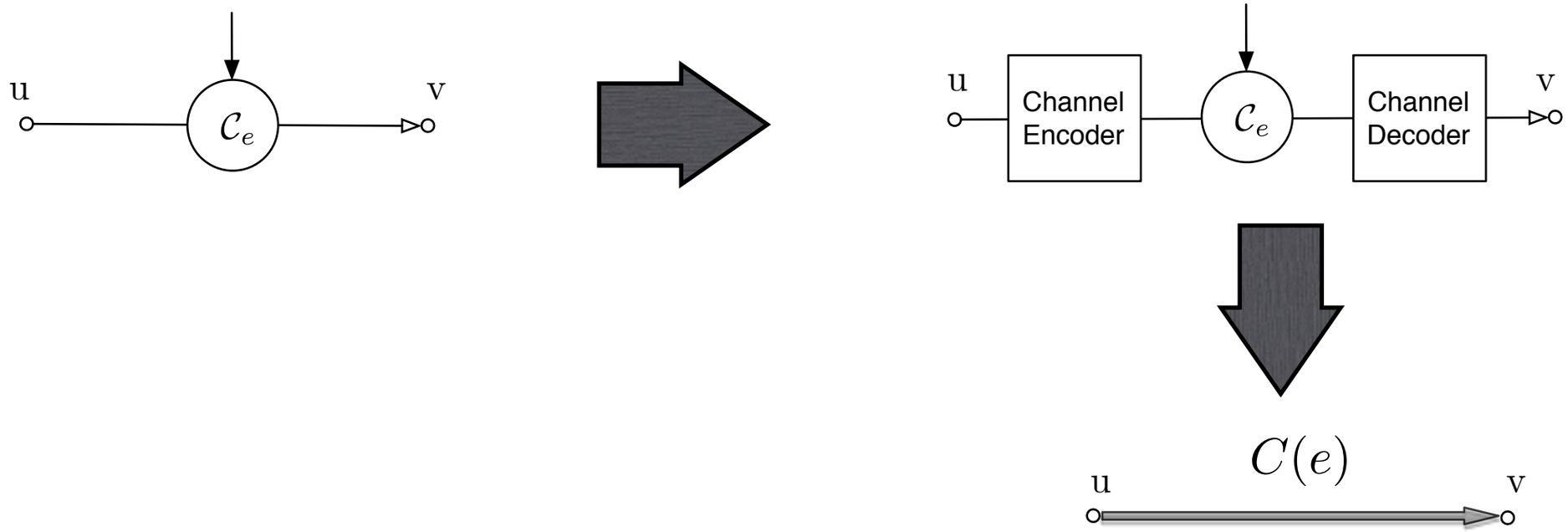
Noisy network *Noiseless network*

Theorem: $\mathcal{R}(\mathcal{N}, K) = \mathcal{R}(\hat{\mathcal{N}}, K)$

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \supseteq \mathcal{R}(\hat{\mathcal{N}}, K)$: separate network coding and channel coding

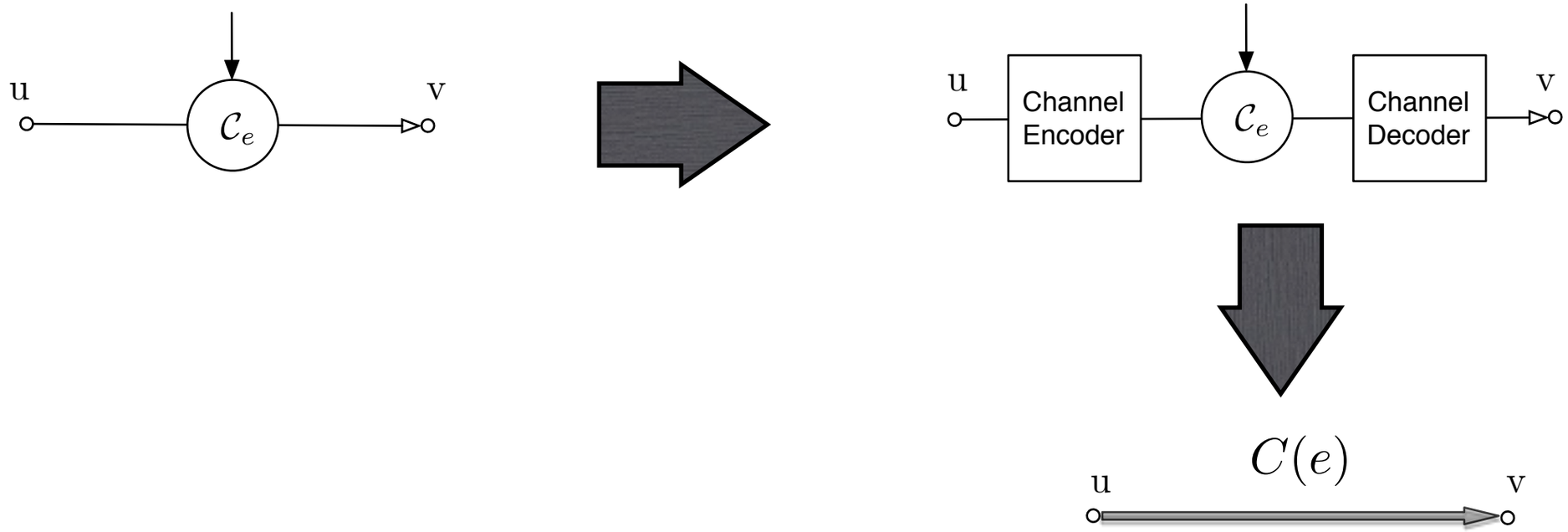
Step 1: Channel code on each link



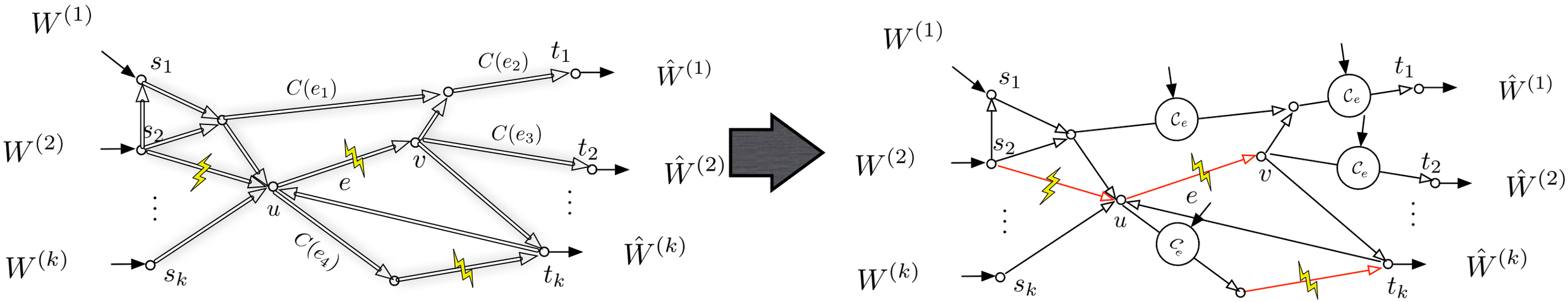
Key Ideas

$\mathcal{R}(\mathcal{N}, K) \supseteq \mathcal{R}(\hat{\mathcal{N}}, K)$: separate network coding and channel coding

Step 1: Channel code on each link



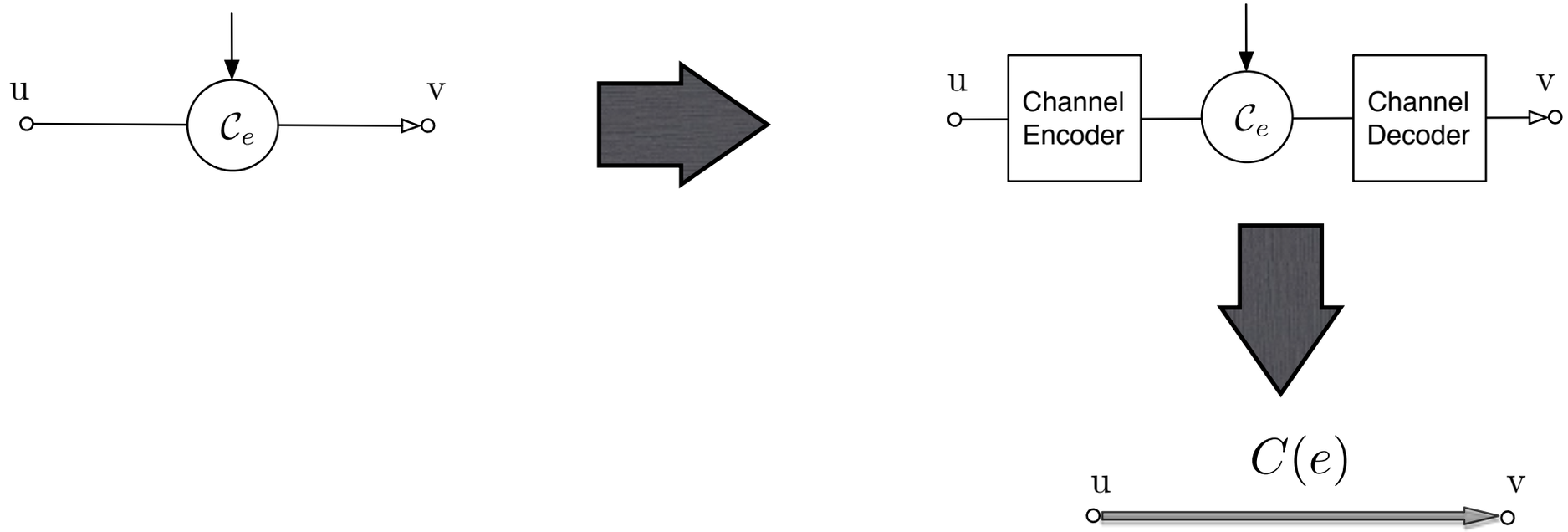
Step 2: Network code for noiseless network => network code for noisy network



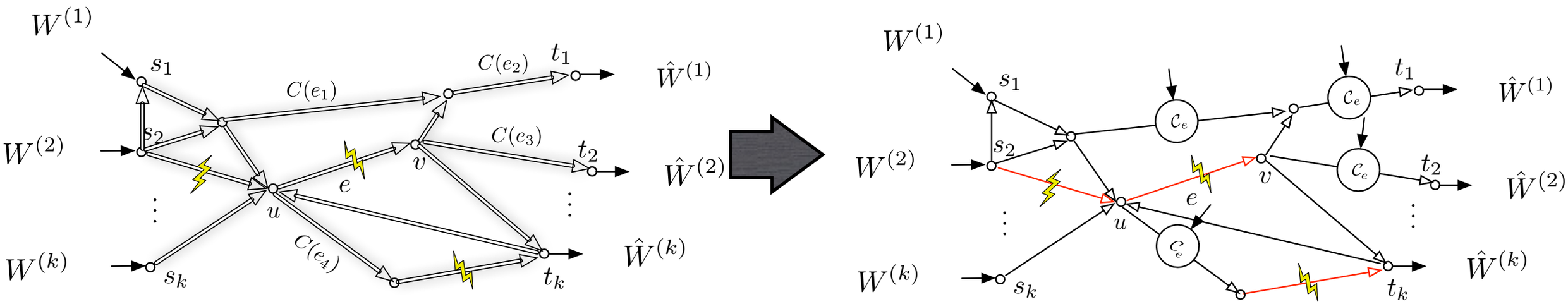
Key Ideas

$\mathcal{R}(\mathcal{N}, K) \supseteq \mathcal{R}(\hat{\mathcal{N}}, K)$: separate network coding and channel coding

Step 1: Channel code on each link



Step 2: Network code for noiseless network => network code for noisy network



=> $\mathbf{R} \in \mathcal{R}(\hat{\mathcal{N}}, K) \implies \mathbf{R} \in \mathcal{R}(\mathcal{N}, K)$

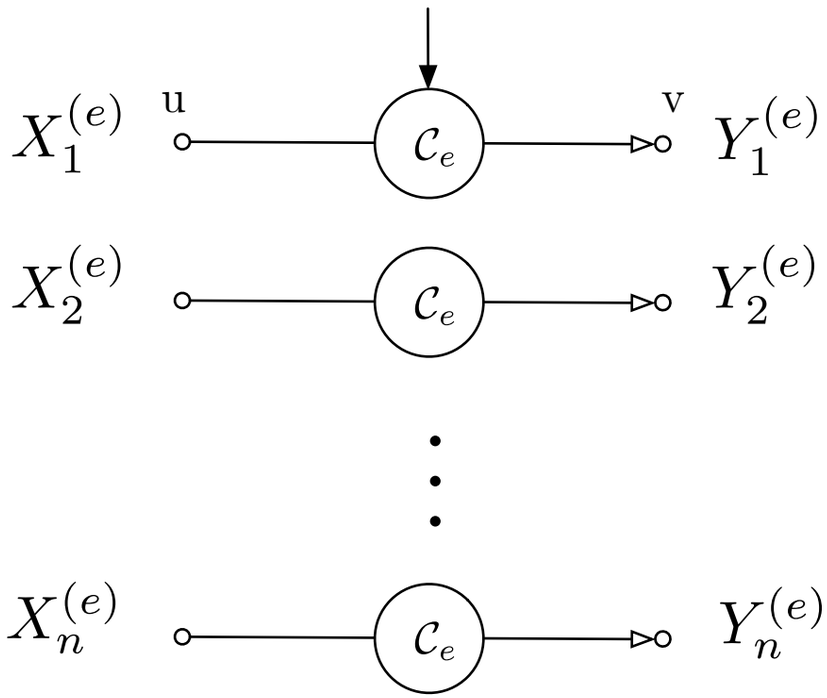
(Same as Koetter et al's proof)

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

Step 1: "Emulate" channel noise on noiseless links

Channel coding:

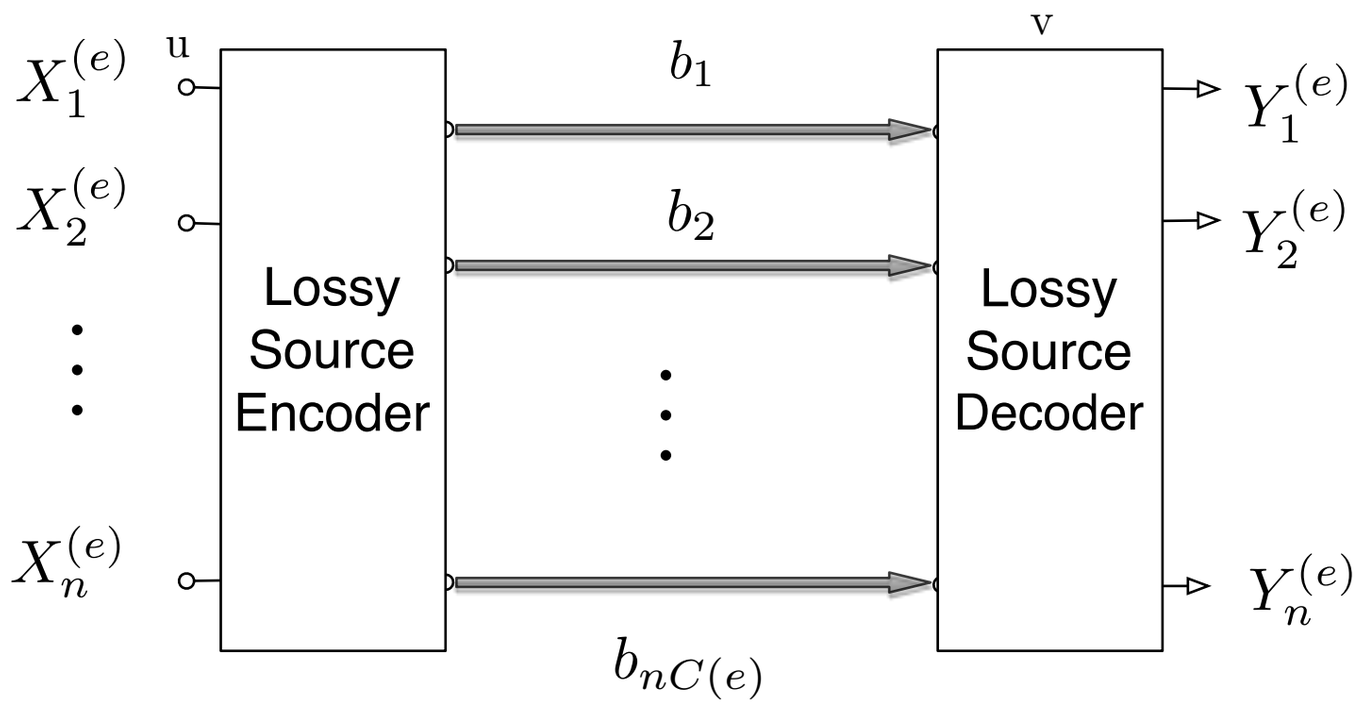


Given: $p(y|x)$

Design: $p(x)$

$C = max_{p(x)} I(X^{(e)}; Y^{(e)})$

Lossy source coding:



Given: $p(x)$

Design: $p(y|x)$

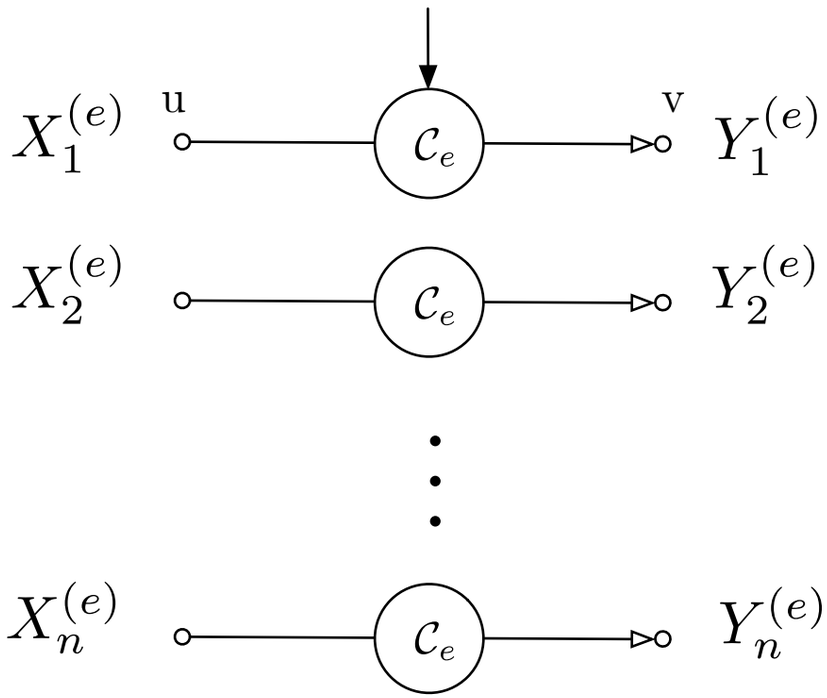
$R = min_{p(y|x)} I(X^{(e)}; Y^{(e)})$

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

Step 1: "Emulate" channel noise on noiseless links

Channel coding:

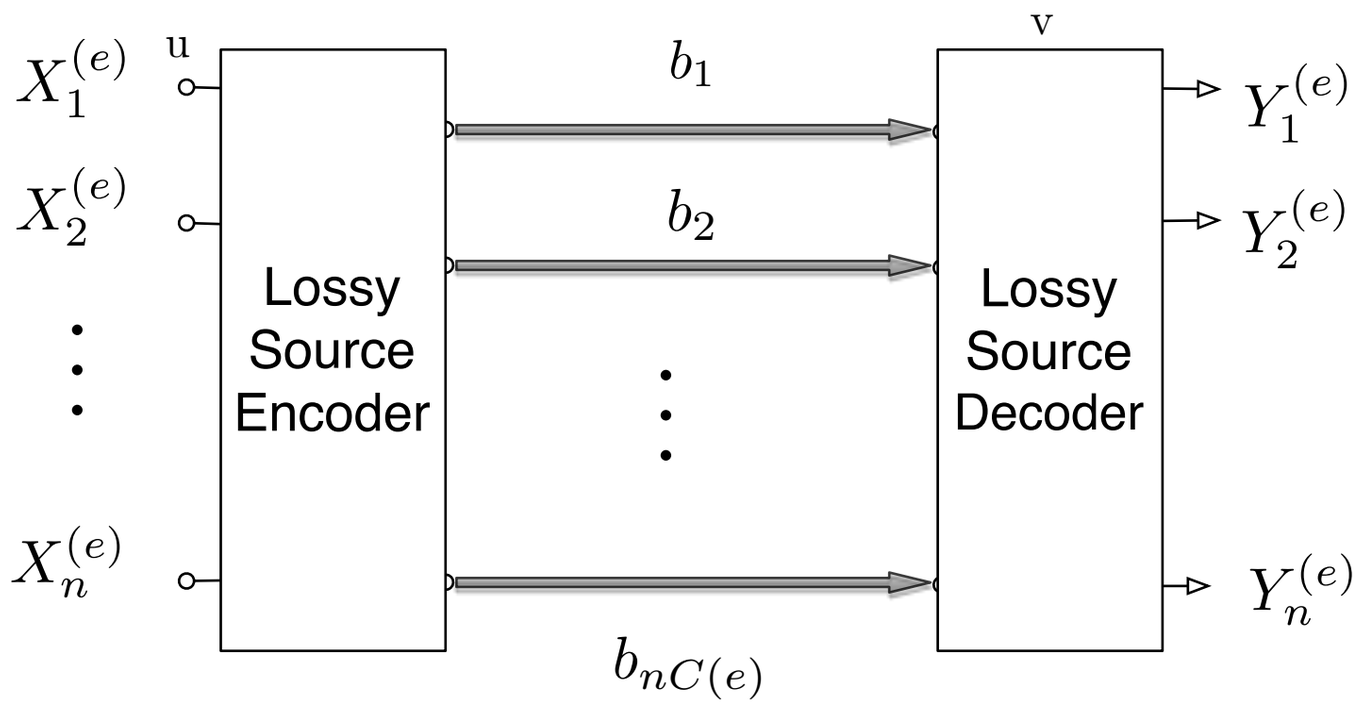


Given: $p(y|x)$

Design: $p(x)$

$C = max_{p(x)} I(X^{(e)}; Y^{(e)})$

Lossy source coding:



Given: $p(x)$

Design: $p(y|x)$

$R = min_{p(y|x)} I(X^{(e)}; Y^{(e)})$

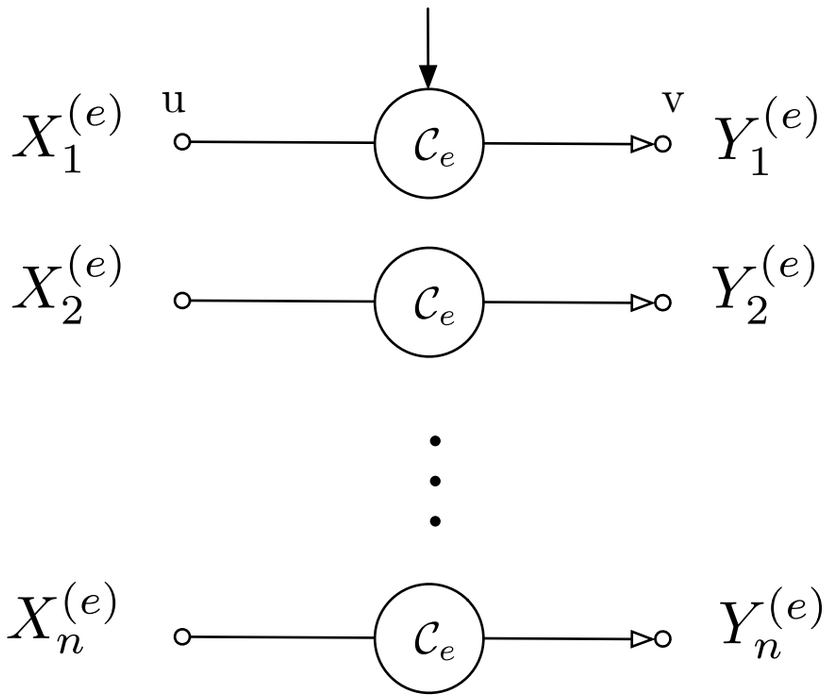
For a fixed $p(y|x)$, $R = I(X^{(e)}; Y^{(e)})$

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

Step 1: "Emulate" channel noise on noiseless links

Channel coding:

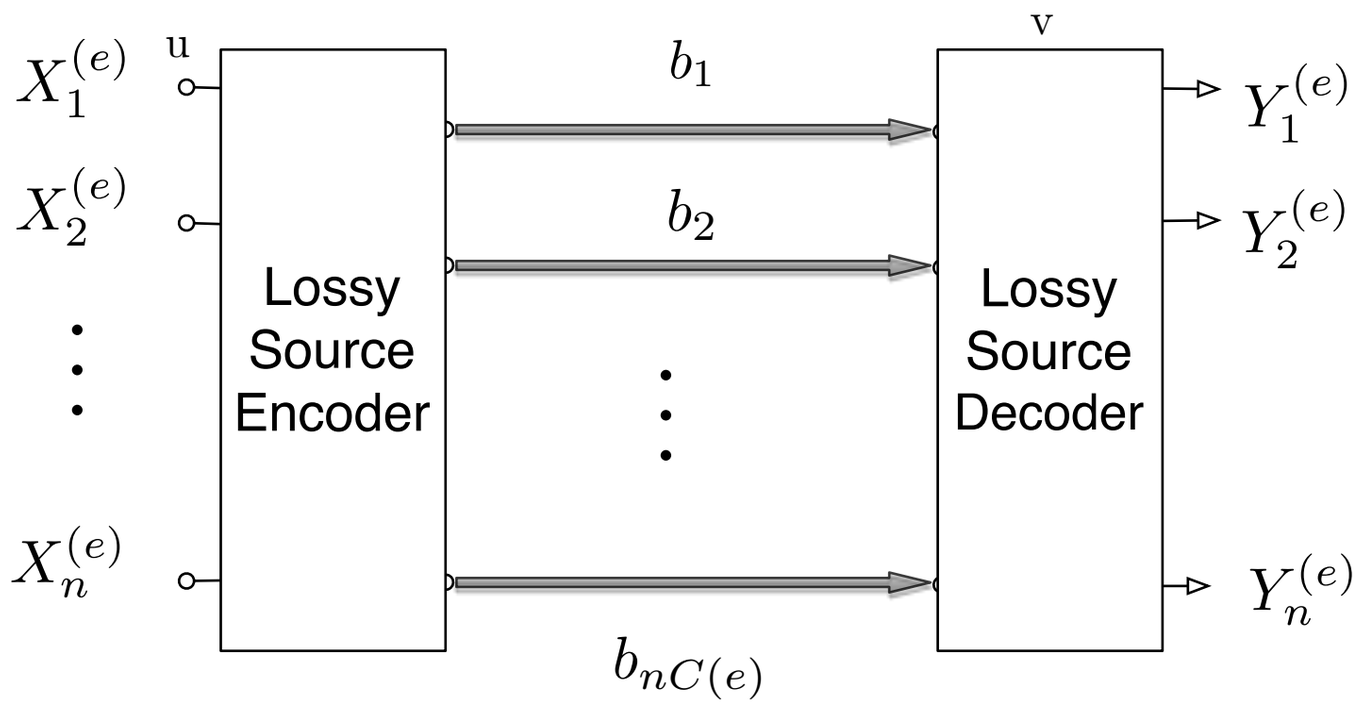


Given: $p(y|x)$

Design: $p(x)$

$C = max_{p(x)} I(X^{(e)}; Y^{(e)})$

Lossy source coding:



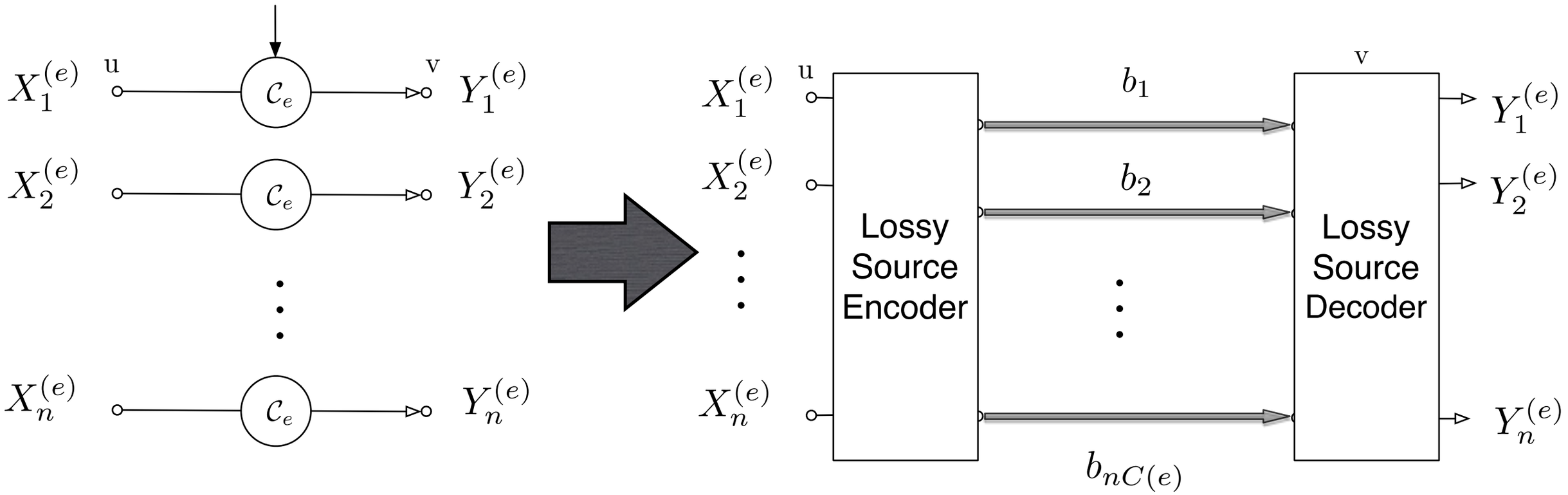
Given: $p(x)$

Design: $p(y|x)$

$R = min_{p(y|x)} I(X^{(e)}; Y^{(e)})$

For a fixed $p(y|x)$, $R = I(X^{(e)}; Y^{(e)}) \leq C$

Emulating channel noise



Noisy channel

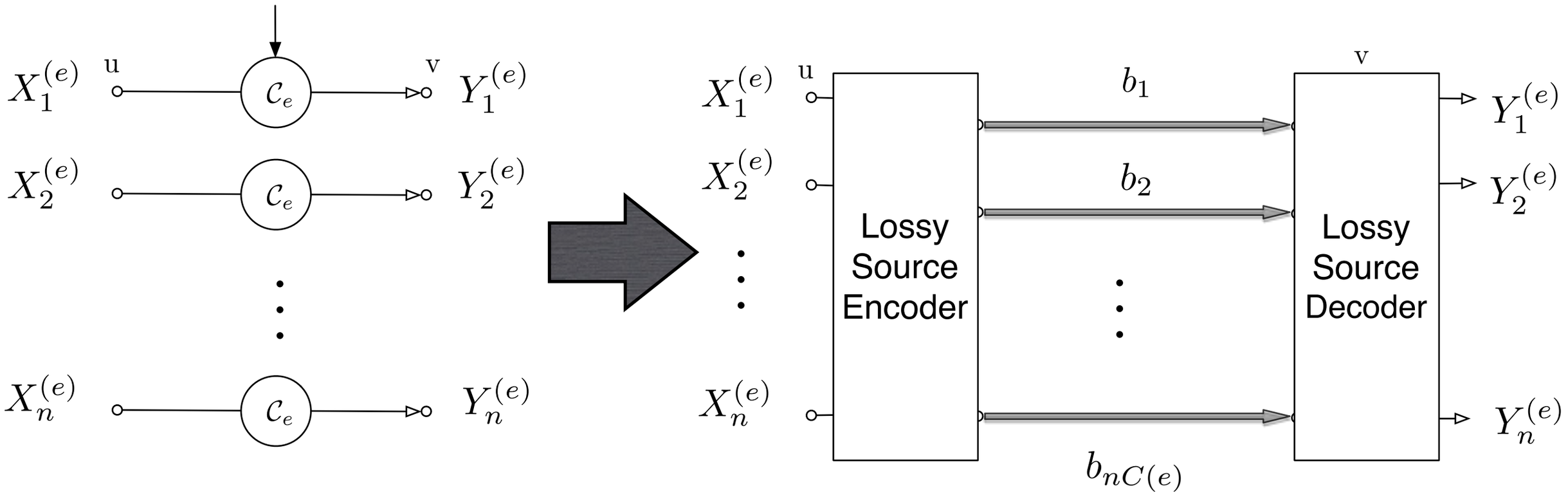
Noiseless channel

- mimic channel transition probability $p_e(y|x)$

- random codebook $p(y) = \sum_x p(x)p(y|x)$

- map input to a typical codeword from the Lossy Source Code according to joint distribution $p(x)p_e(y|x)$

Emulating channel noise



Noisy channel

Noiseless channel

- mimic channel transition probability $p_e(y|x)$

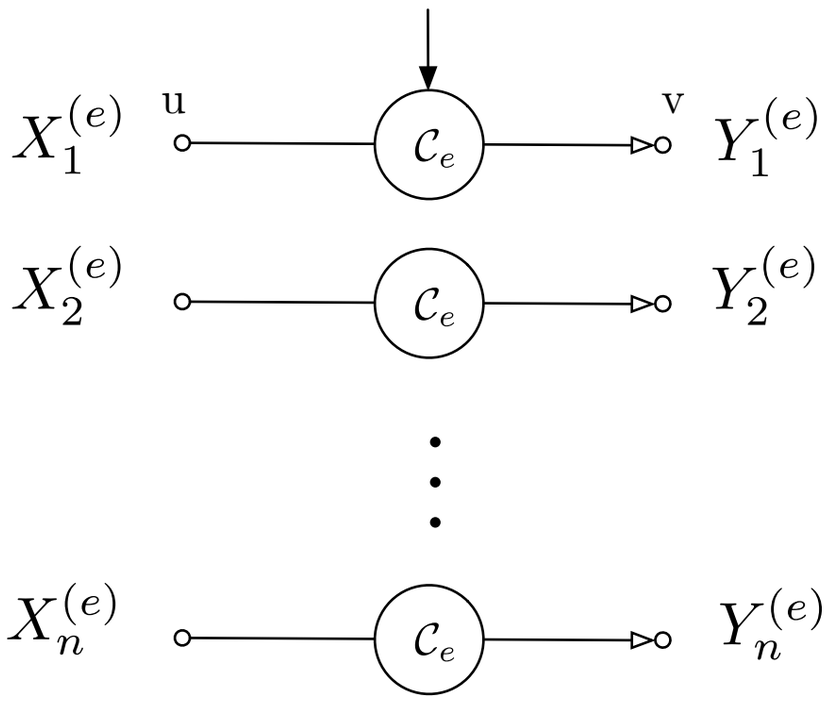
- random codebook $p(y) = \sum_x p(x)p(y|x)$

- map input to a typical codeword from the Lossy Source Code according to joint distribution $p(x)p_e(y|x)$

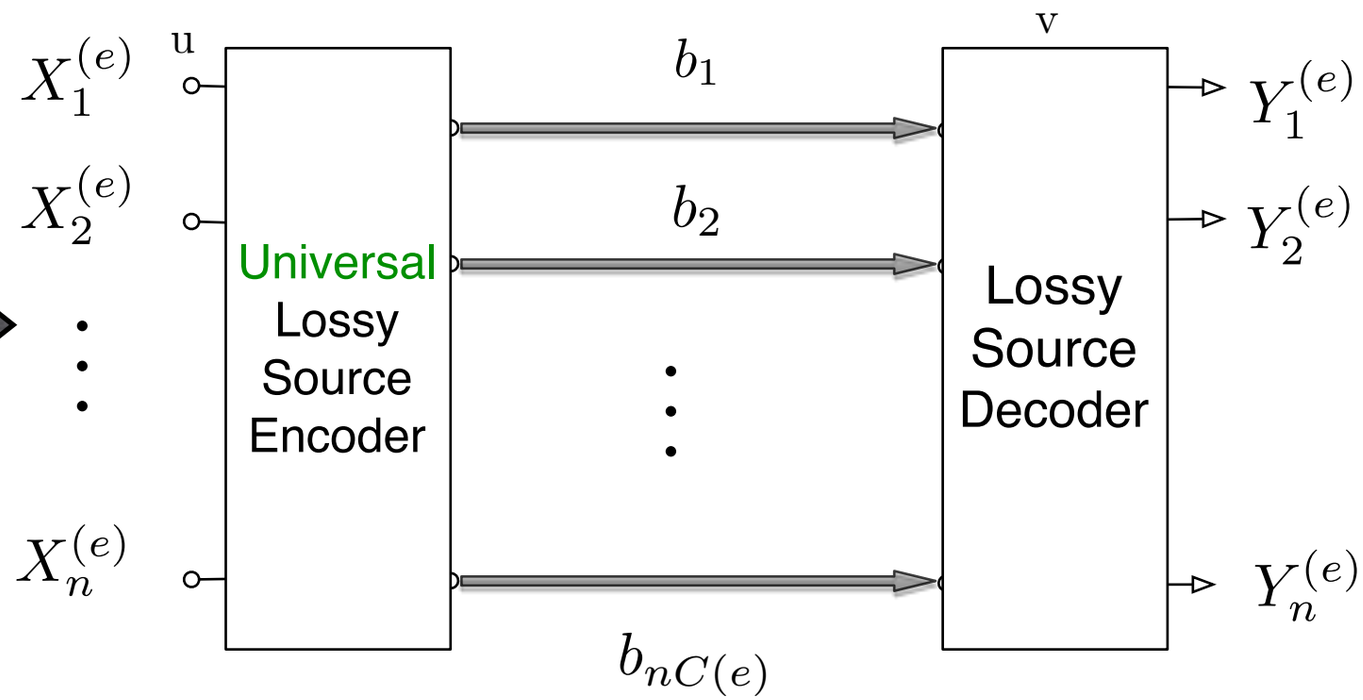
- **Requires input empirical distribution to be close to** $p(x)$

Adversary can control $p(x)$

Emulating channel noise



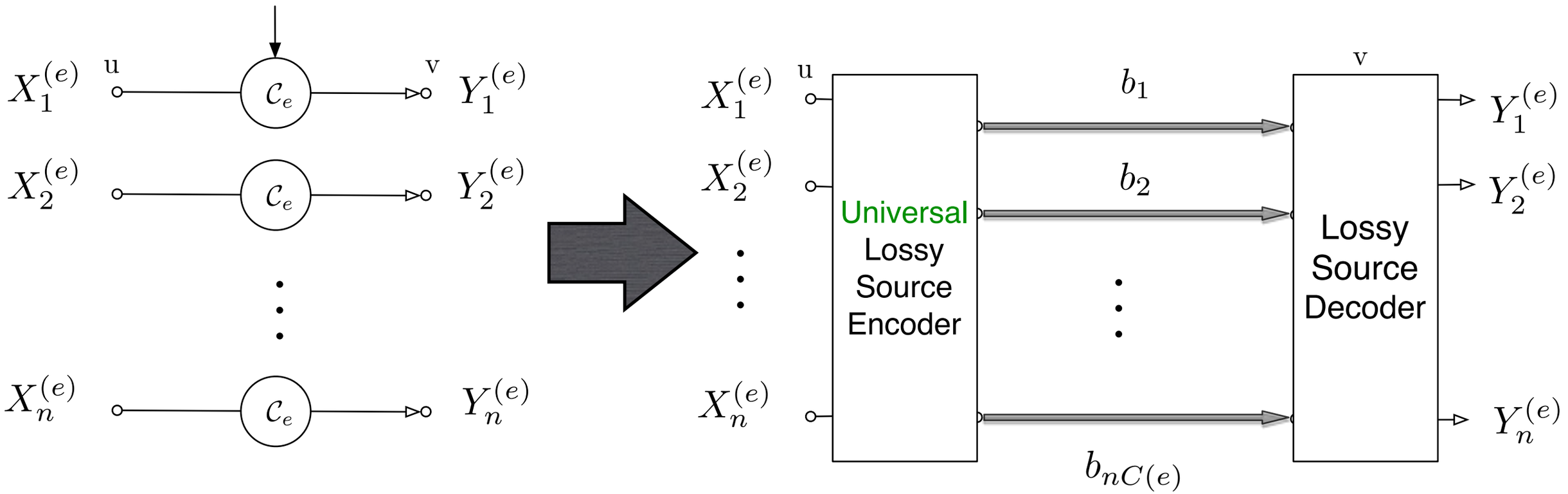
Noisy channel



Noiseless channel

- Design different random codebook for each input Type

Emulating channel noise



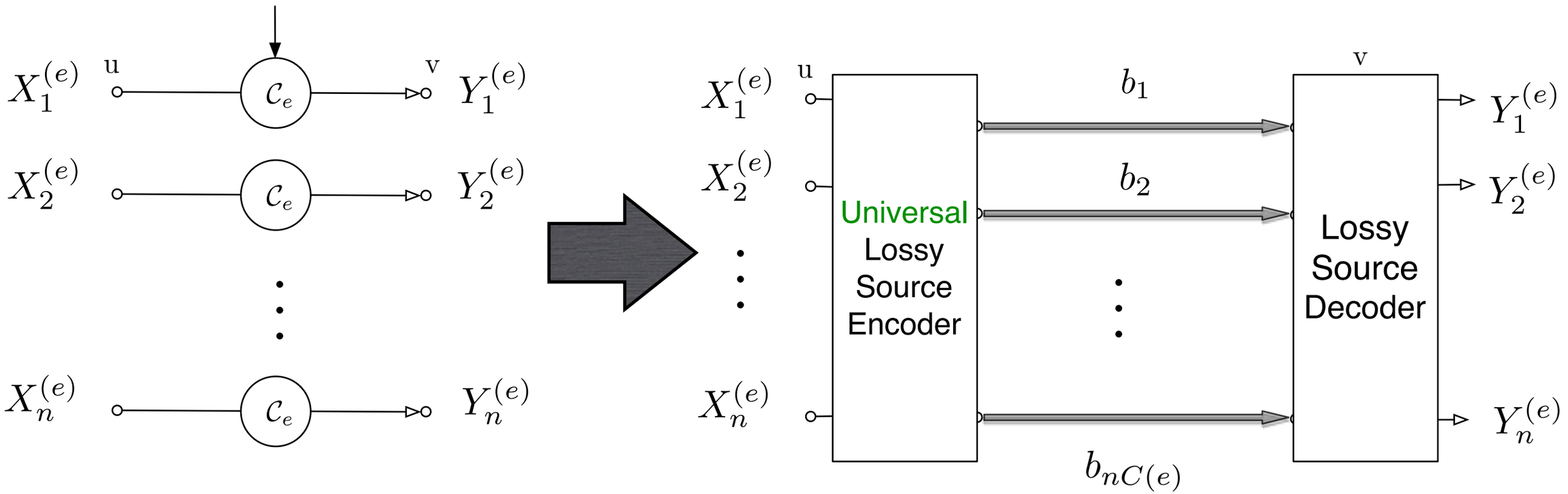
Noisy channel

Noiseless channel

- Design different random codebook for each input Type

Step 1: Describe empirically observed type

Emulating channel noise



Noisy channel

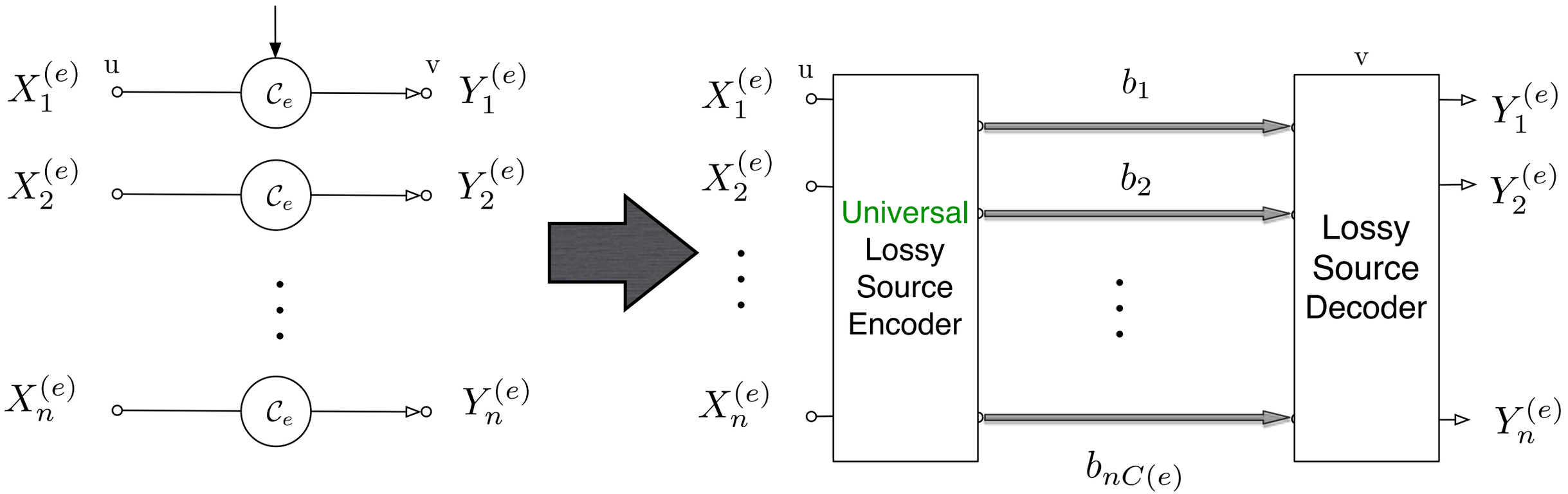
Noiseless channel

- Design different random codebook for each input Type

Step 1: Describe empirically observed type

Step 2: Apply Lossy Source Code designed for the observed type

Emulating channel noise



Noisy channel

Noiseless channel

- Design different random codebook for each input Type

Step 1: Describe empirically observed type

Step 2: Apply Lossy Source Code designed for the observed type

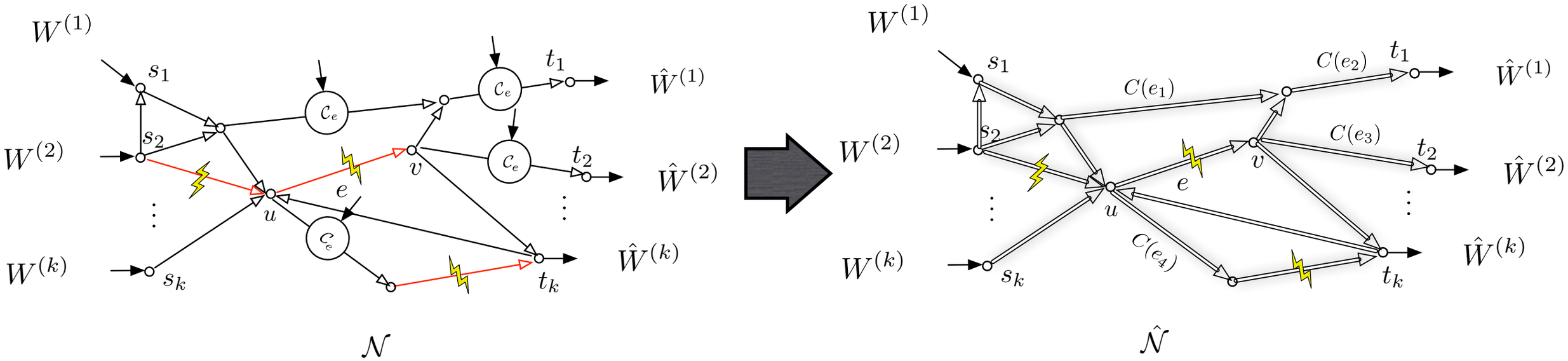
- **Works for all input distributions**

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

Step 1: "Emulate" channel noise on noiseless links

Step 2: **Operate a code for noisy network on the noiseless network by channel emulation**

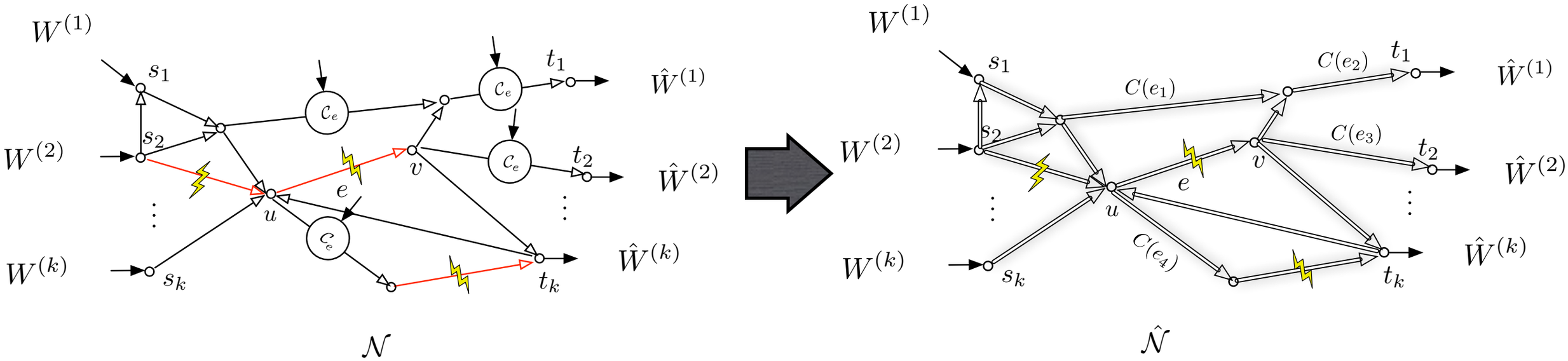


Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

Step 1: "Emulate" channel noise on noiseless links

Step 2: **Operate a code for noisy network on the noiseless network by channel emulation**



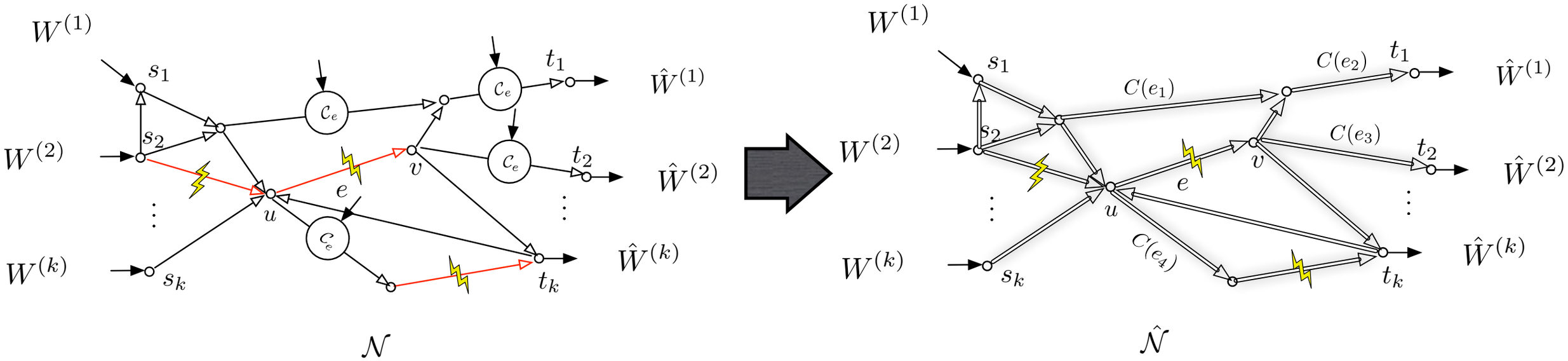
$$\Pr(\text{error}, \hat{\mathcal{N}}) \leq \Pr(\text{error}, \mathcal{N}) \cdot 2^{n\epsilon}$$

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

Step 1: "Emulate" channel noise on noiseless links

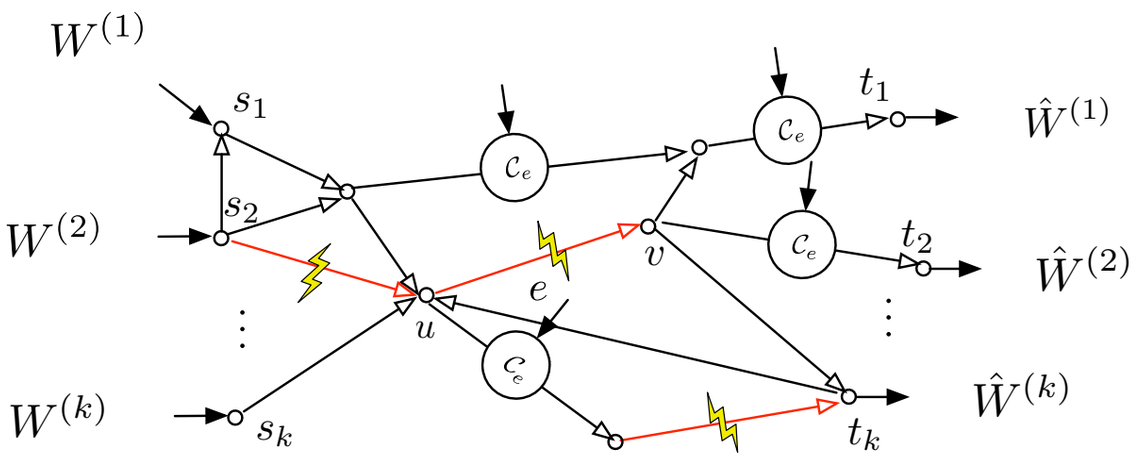
Step 2: **Operate a code for noisy network on the noiseless network by channel emulation**



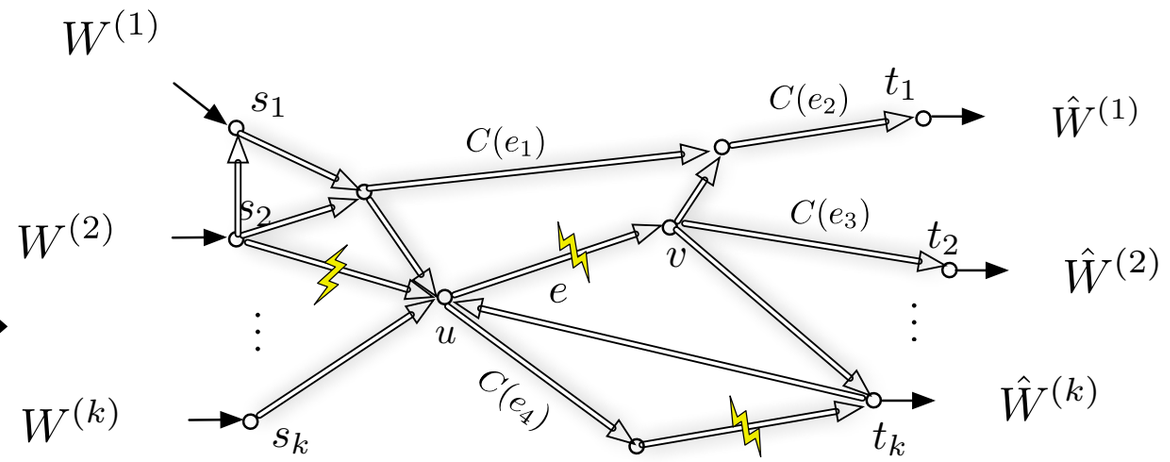
$$\Pr(\text{error}, \hat{\mathcal{N}}) \leq \Pr(\text{error}, \mathcal{N}) \cdot 2^{n\epsilon}$$

- need $\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$ for some $\delta > \epsilon$

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$



$\mathcal{N}$



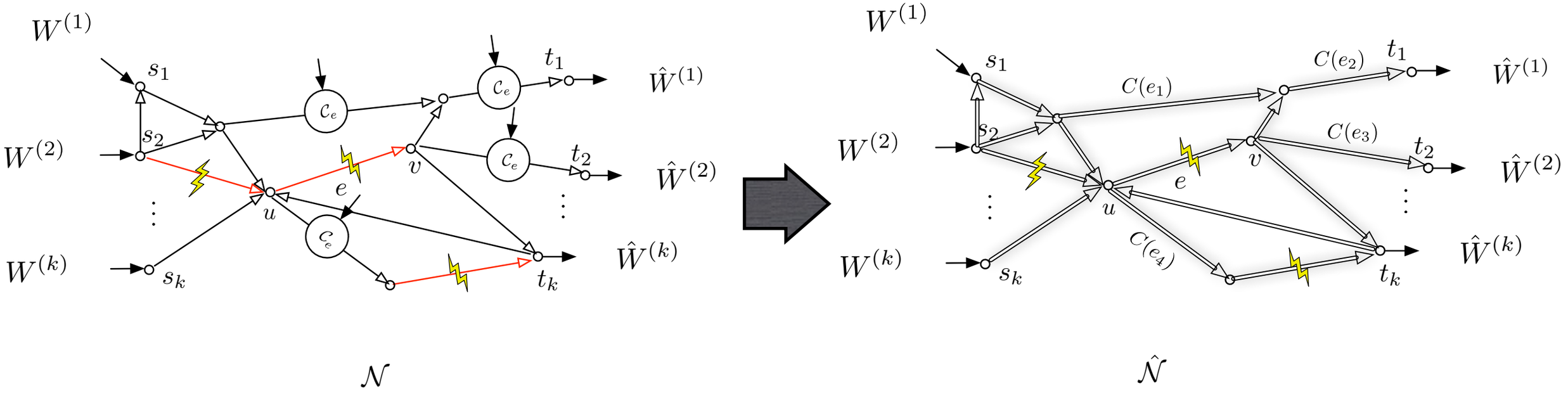
$\hat{\mathcal{N}}$

$$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$$

- Given network code of blocklength m s.t. $W^{(i)} \in \{0,1\}^{mR_i}$

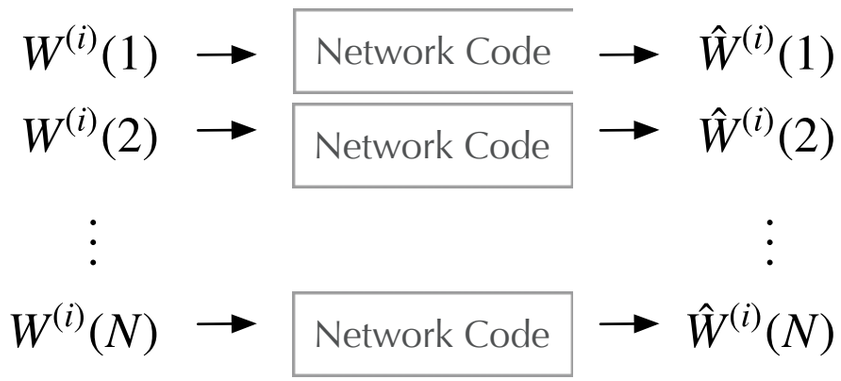
$$W^{(i)} \rightarrow \boxed{\text{Network Code}} \rightarrow \hat{W}^{(i)}$$

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$

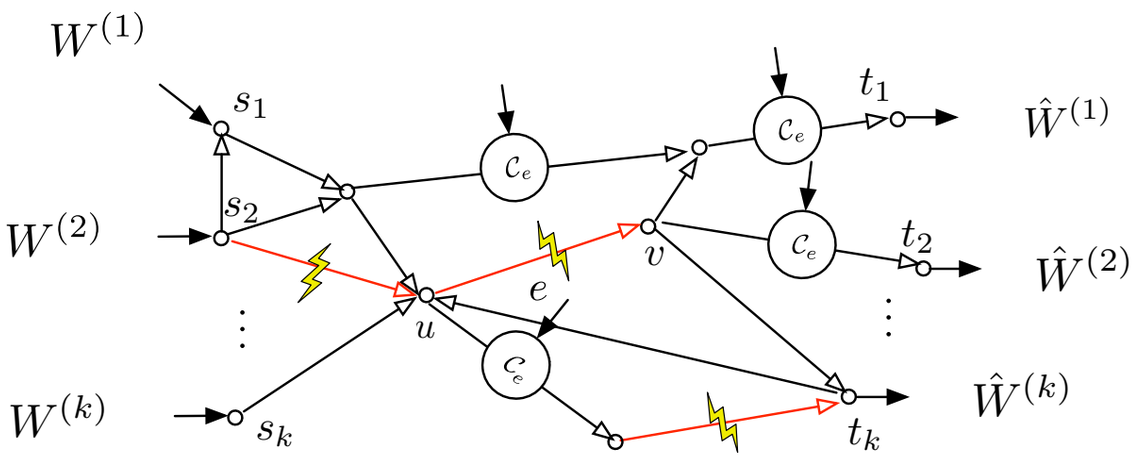


$$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$$

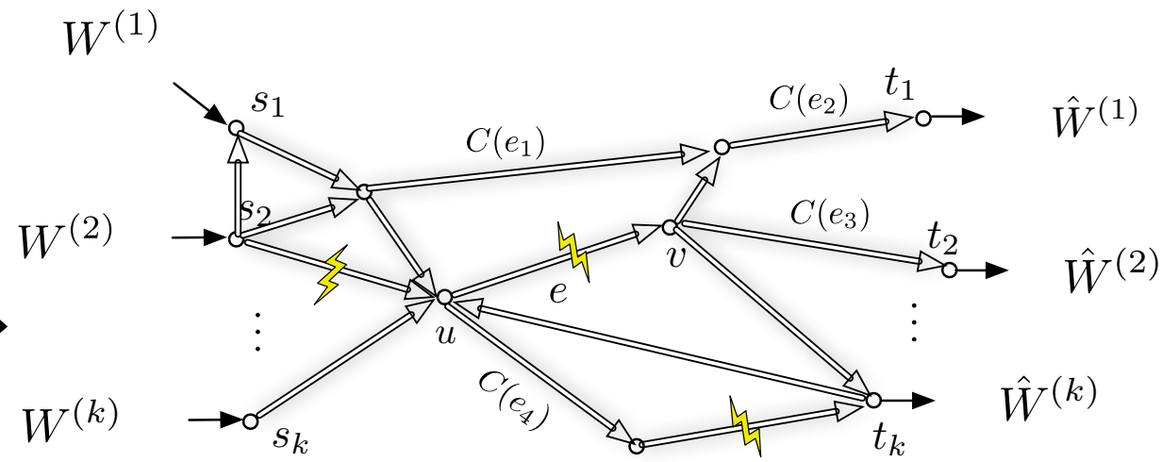
- Given network code of blocklength m s.t. $W^{(i)} \in \{0,1\}^{mR_i}$
- Design code of blocklength mN by concatenation



$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$



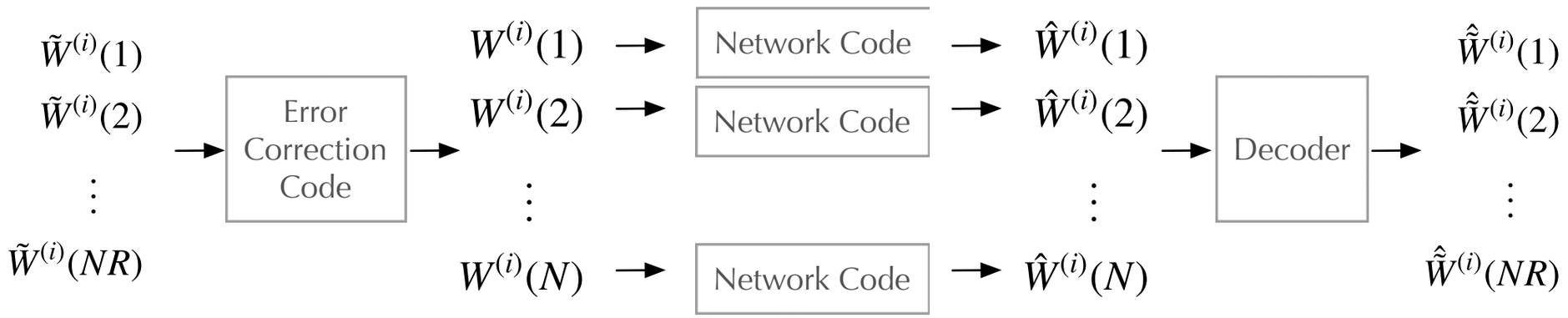
$\mathcal{N}$



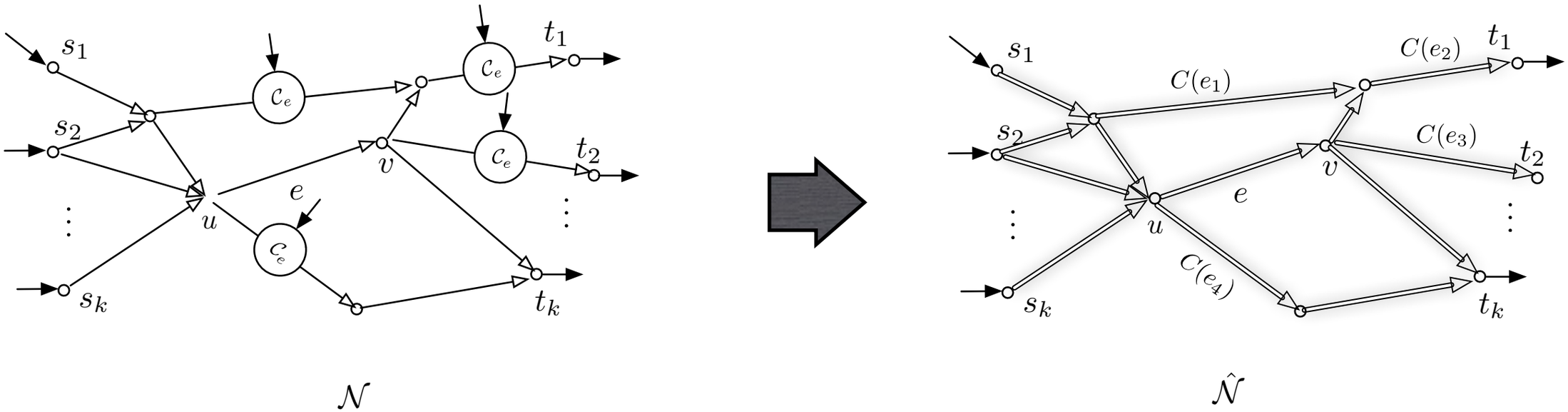
$\hat{\mathcal{N}}$

$$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$$

- Given network code of blocklength m s.t. $W^{(i)} \in \{0,1\}^{mR_i}$
- Design code of blocklength mN by concatenation
- Add outer code, $R = 1 - \alpha$



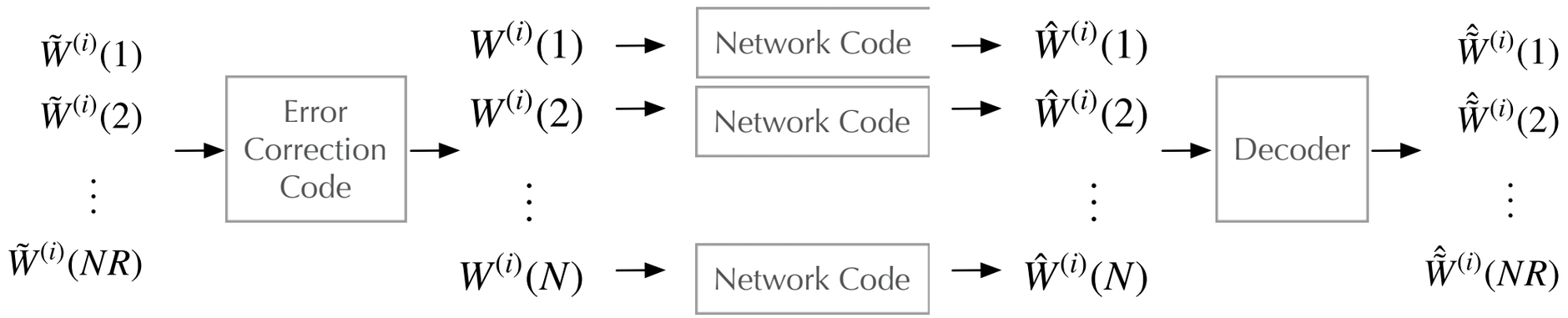
$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$



$\mathcal{N}$ $\qquad\qquad$ $\hat{\mathcal{N}}$

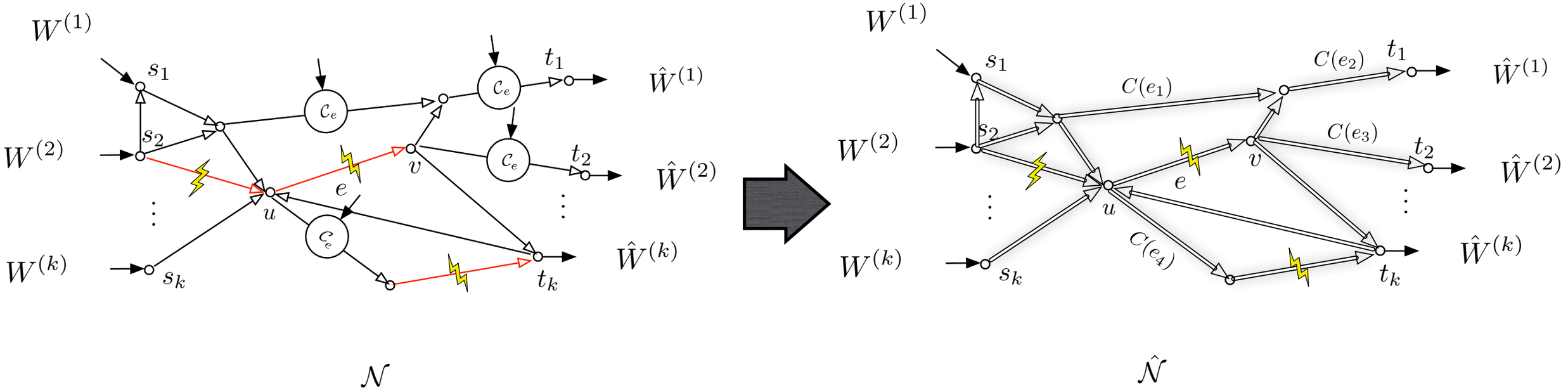
$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$

- Given network code of blocklength m s.t. $W^{(i)} \in \{0,1\}^{mR_i}$
- Design code of blocklength mN by concatenation
- Add outer code, $R = 1 - \alpha$

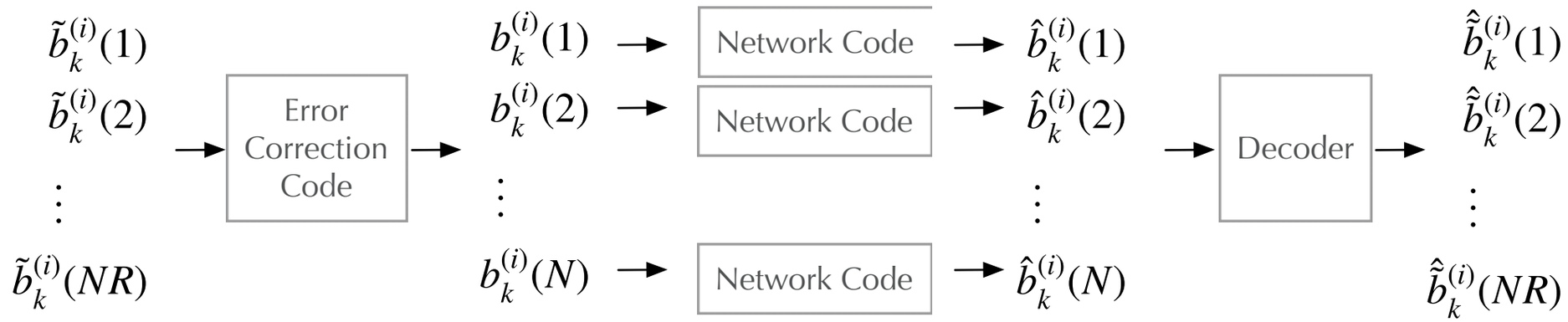


Koetter et al: i.i.d. noise => Gallagher's error exponents/Large Deviations => $\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$



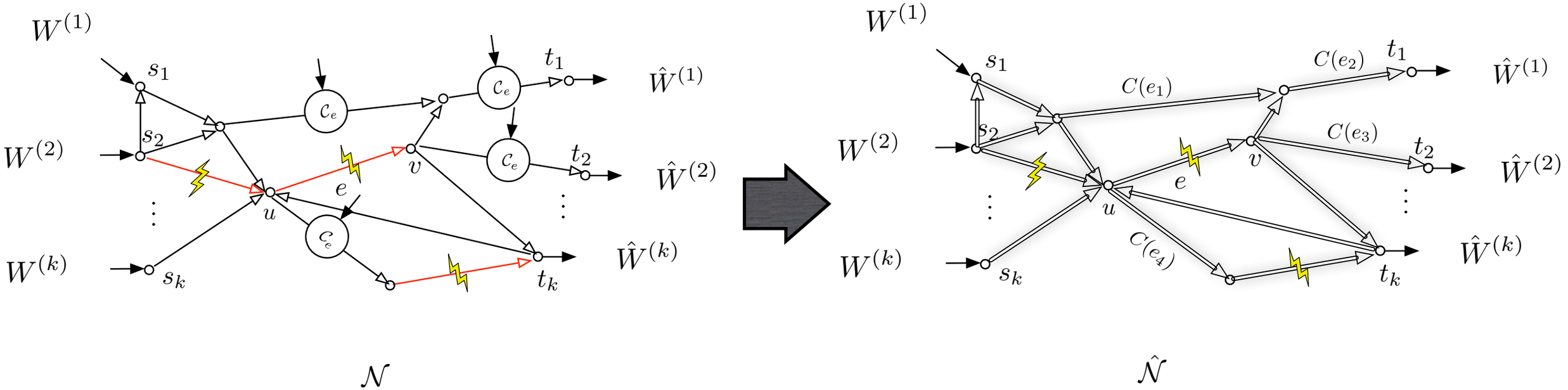
$$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$$



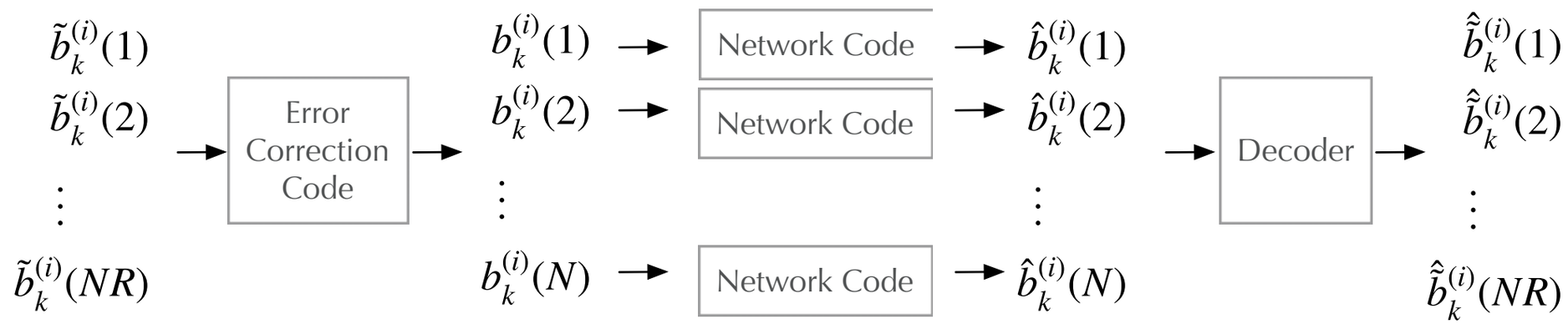
Our proof:

- Each $W^{(i)}(j) = b_1^{(i)}(j), \ldots, b_{mR_i}^{(i)}(j)$

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$



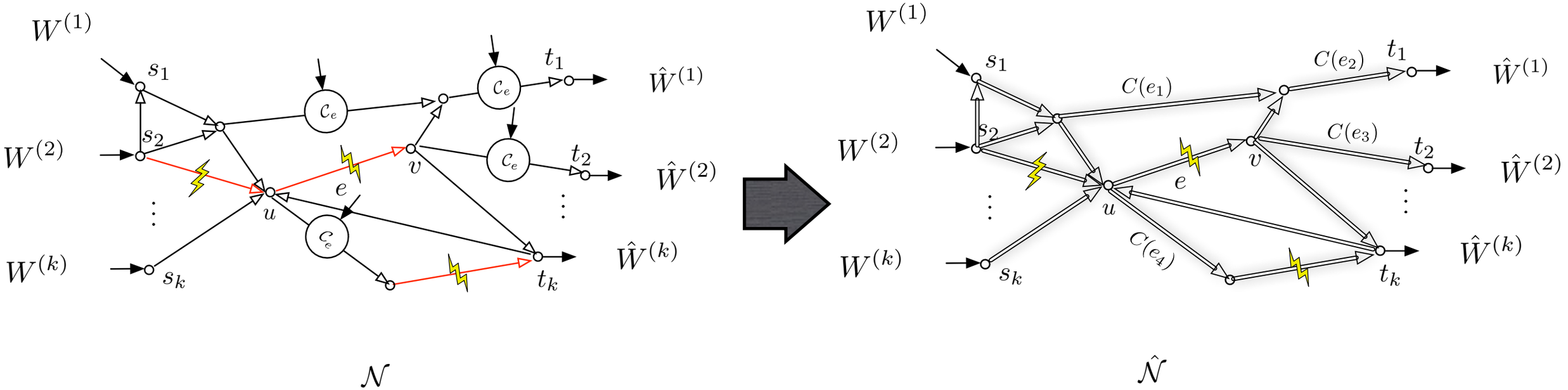
$$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$$



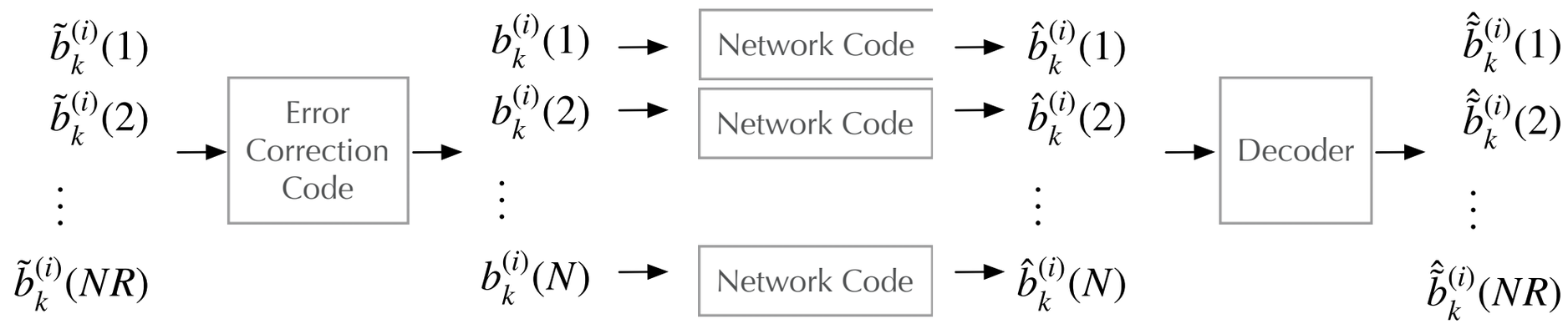
Our proof:

- Each $W^{(i)}(j) = b_1^{(i)}(j), \ldots, b_{mR_i}^{(i)}(j)$
- Minimum Hamming distance decoding
  - error probability is maximized when adversary acts independently at different time steps

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$



$$\Pr(\text{error}, \mathcal{N}) \leq 2^{-n\delta}$$



Our proof:

- Each $W^{(i)}(j) = b_1^{(i)}(j), \ldots, b_{mR_i}^{(i)}(j)$
- Minimum Hamming distance decoding
  - error probability is maximized when adversary acts independently at different time steps
- Upper bound error on probability

in the worst case

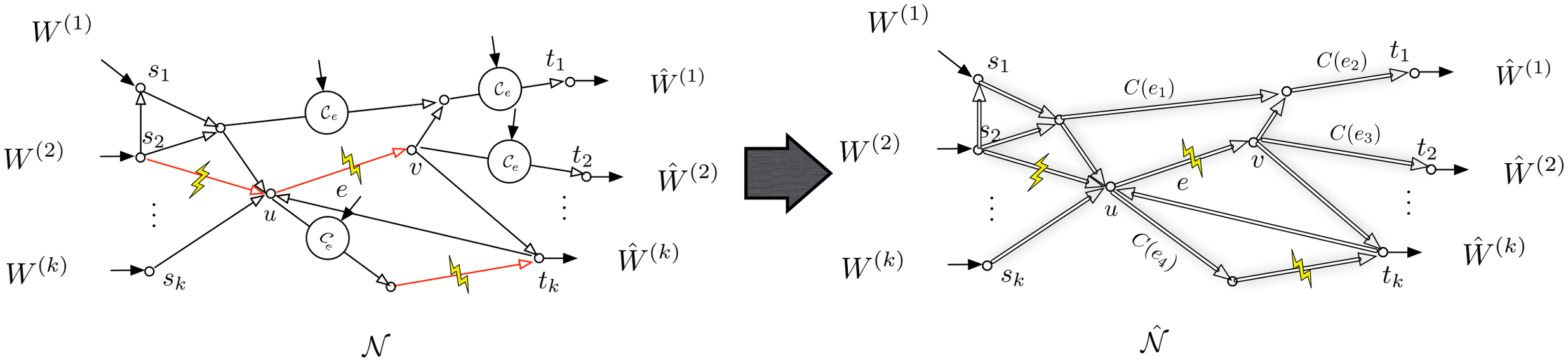
$$P(\underline{W} \neq \underline{\hat{W}}) < 2^{-N\delta}$$

Key Ideas

$\mathcal{R}(\mathcal{N}, K) \subseteq \mathcal{R}(\hat{\mathcal{N}}, K)$:

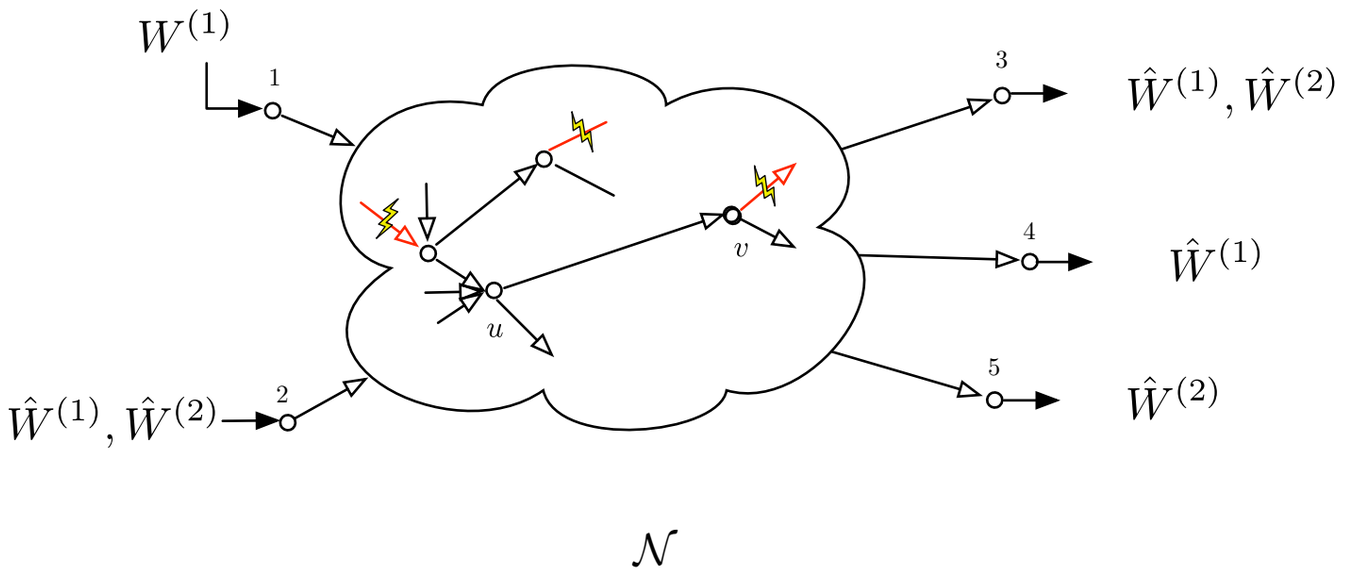
Step 1: "Emulate" channel noise on noiseless links

Step 2: Operate a code for noisy network on the noiseless network by channel emulation



$$\mathbf{R} \in \mathcal{R}(\mathcal{N}, K) \implies \mathbf{R} \in \mathcal{R}(\hat{\mathcal{N}}, K)$$

Generalization



Network:

- Arbitrary demand model (e.g., some unicast, some multicast etc)

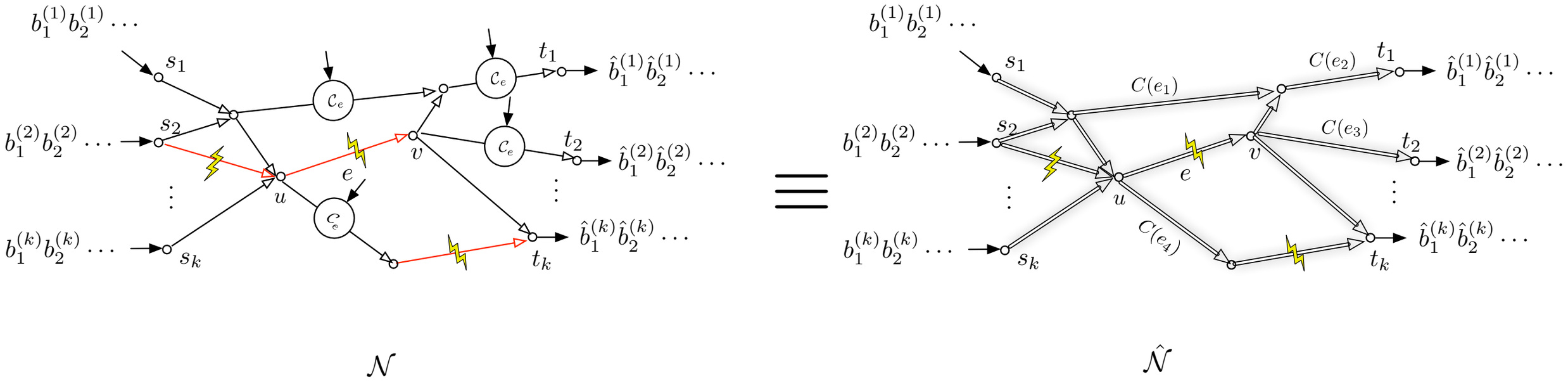
- Channels of arbitrary capacity

Byzantine Adversary:

- can replace received vectors on any edges in any subset σ, where $\sigma \in \mathbf{\Sigma}$, the set of allowed attack sets.

e.g. node-based attacks

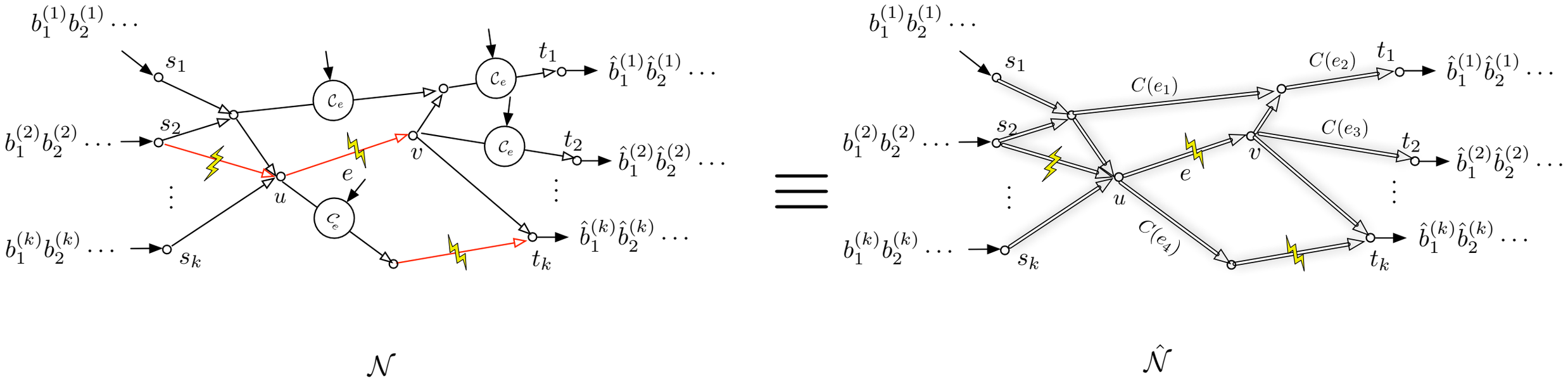
Implications



- Code design: simplifies design of codes

  - channel code for each link
  - network code against adversary

Implications



$\mathcal{N}$ $\hat{\mathcal{N}}$

- Code design: simplifies design of codes

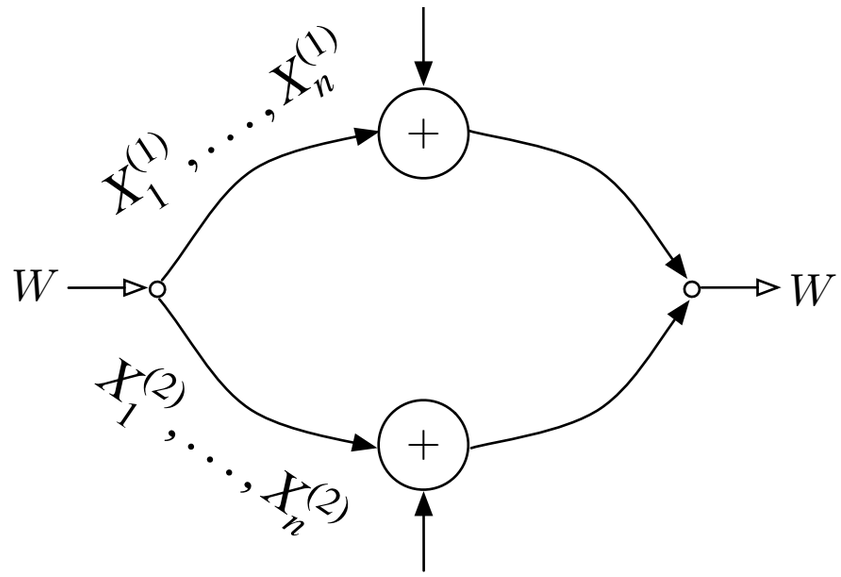
  - channel code for each link
  - network code against adversary

  Note: separation may perform poorly for small blocklength

Small blocklength

Note: separation may perform poorly for small blocklength

$$W \in \{0,1\}^{nR}$$



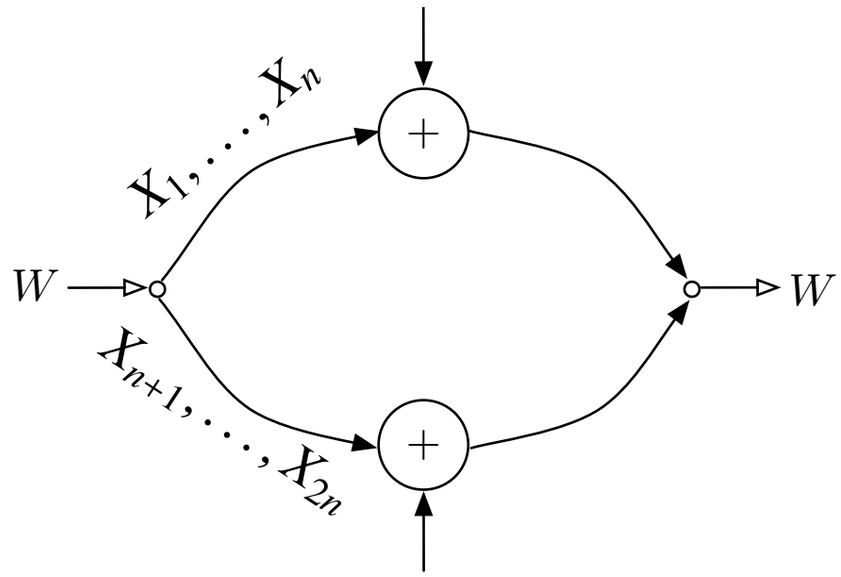
With separation:

$$\Pr(error) \sim 2^{-n\delta}$$

Small blocklength

Note: separation may perform poorly for small blocklength

$W \in \{0,1\}^{nR}$



With separation:

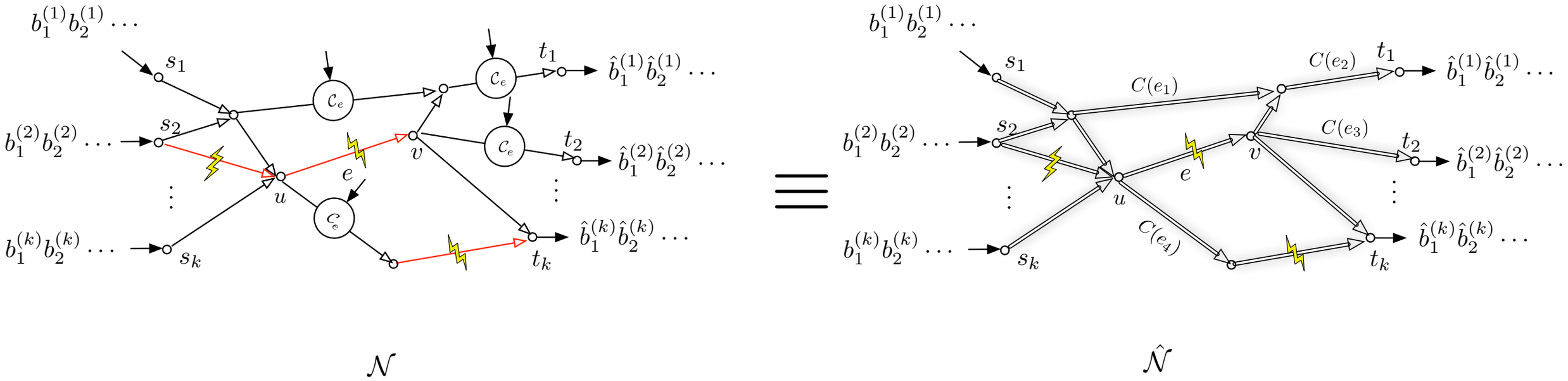
$$\Pr(error) \sim 2^{-n\delta}$$

Without separation:

Channel code of length $2n$ can be used

$$\Pr(error) \sim 2^{-2n\delta}$$

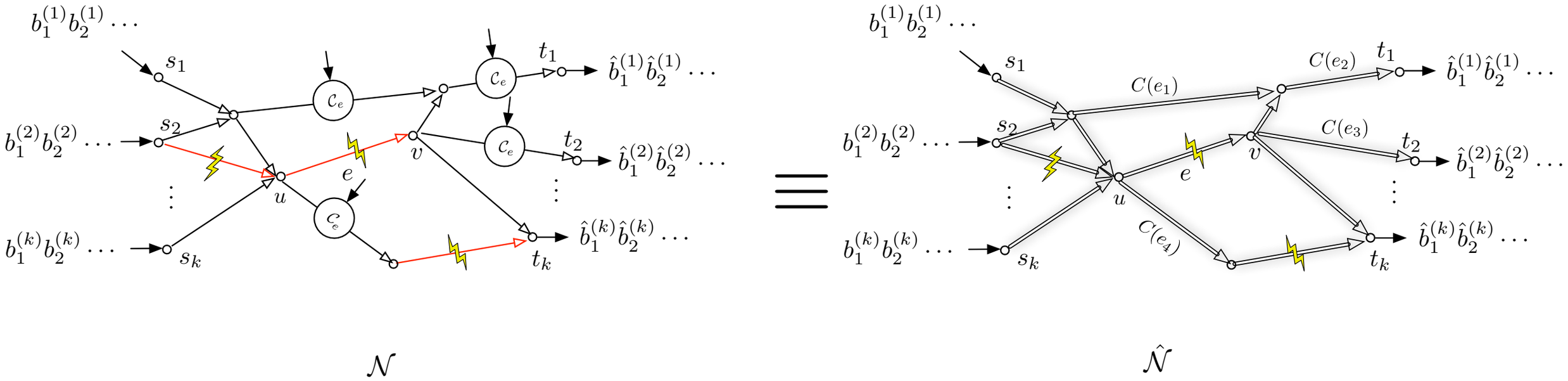
Implications



- Code design: simplifies design of codes if asymptotic rate is the criterion

  - channel code for each link
  - network code against adversary

Implications



$\mathcal{N}$ $\qquad$ $\hat{\mathcal{N}}$

- Code design: simplifies design of codes if asymptotic rate is the criterion

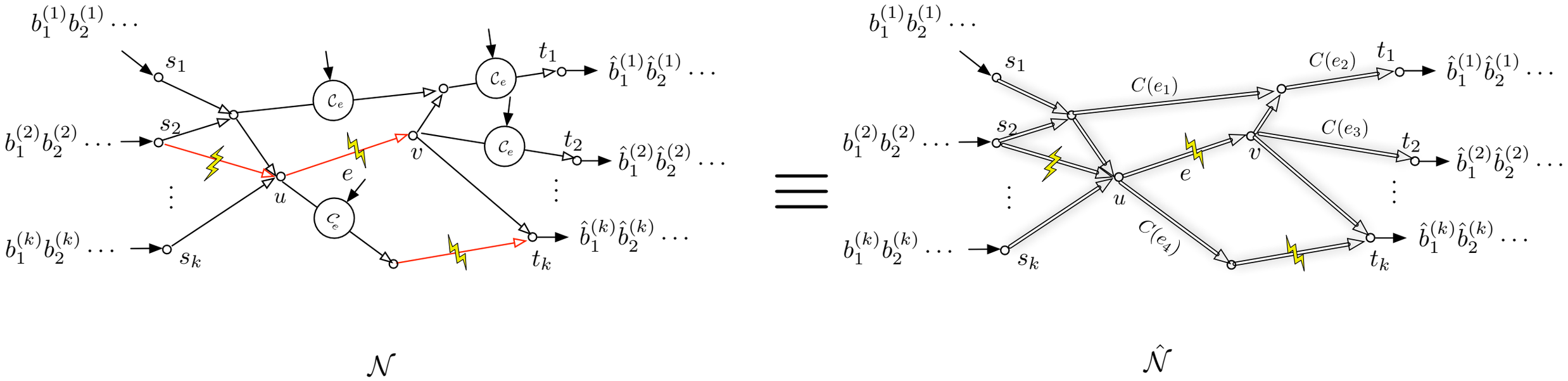
  - channel code for each link
  - network code against adversary

- Capacity calculation: reduces the class of problems

  Noiseless networks <=> Noisy networks

  - equal link capacities [Yeung et al 2006, Jaggi et al 2007]
  - partial results for unequal link capacities [Kim et al 2009]
  - node based adversaries [Kosut et al 2009]

Implications



- Code design: simplifies design of codes <u>if asymptotic rate is the criterion</u>

  - channel code for each link
  - network code against adversary

- Capacity calculation: **<u>reduces the class of problems</u>**

  Noiseless networks <=> Noisy networks

  - equal link capacities [Yeung et al 2006, Jaggi et al 2007]
  - partial results for unequal link capacities [Kim et al 2009]
  - node based adversaries [Kosut et al 2009]

General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

- Apply similar approach to other network problems => reduces the class of problems

  - Network-Channel separation

General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

- Apply similar approach to other network problems => reduces the class of problems

  - Network-Channel separation

  - Network Decomposition

  - Feedback

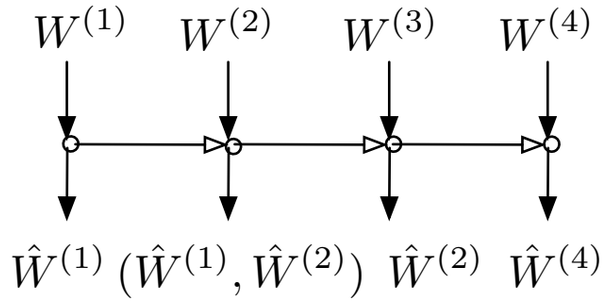
  - Side Information

General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

- Apply similar approach to other network problems

  - Network-Channel separation
  - Network Decomposition
  - Feedback
  - Side Information

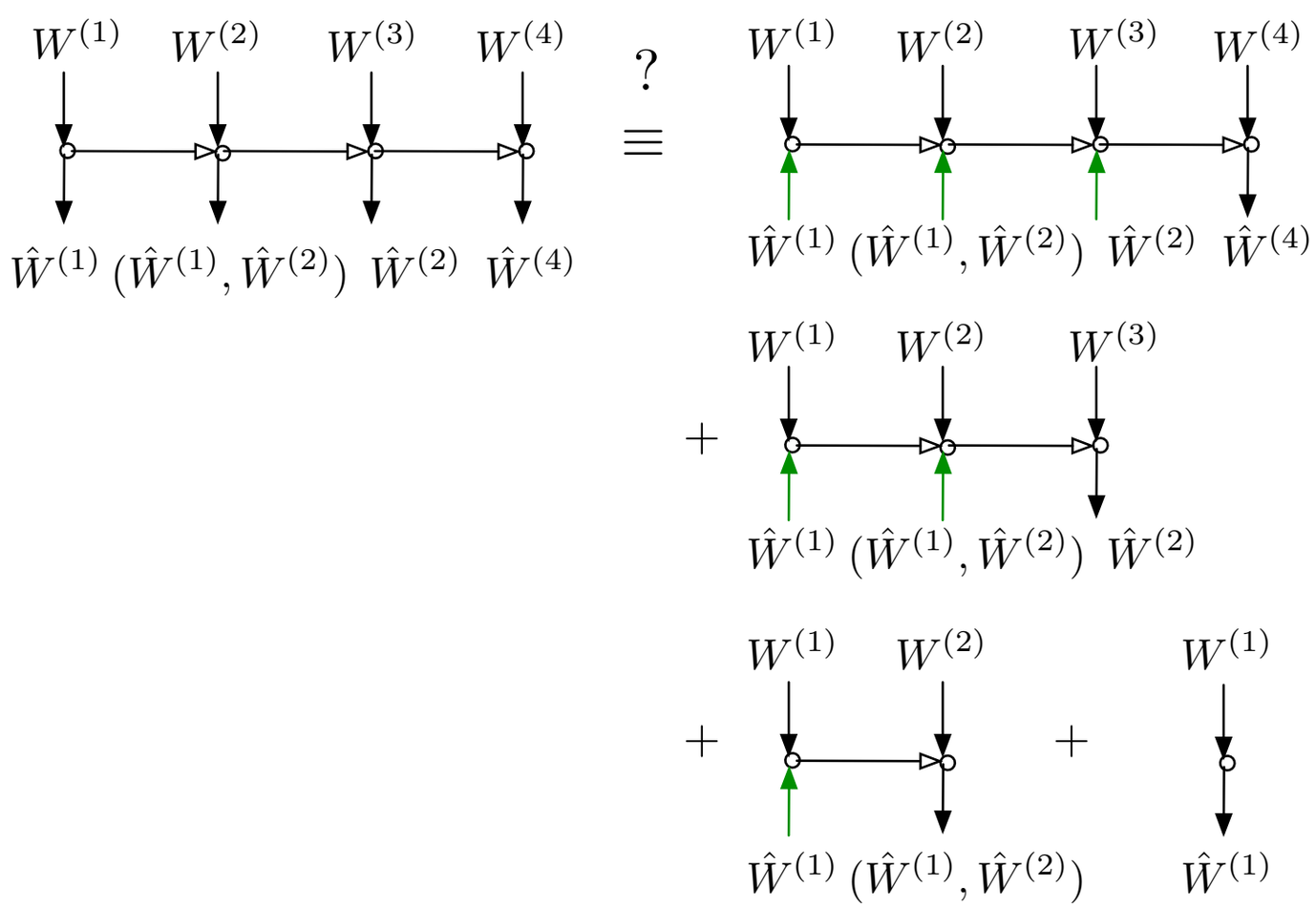


General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

- Apply similar approach to other network problems

  - Network-Channel separation
  - Network Decomposition
  - Feedback
  - Side Information

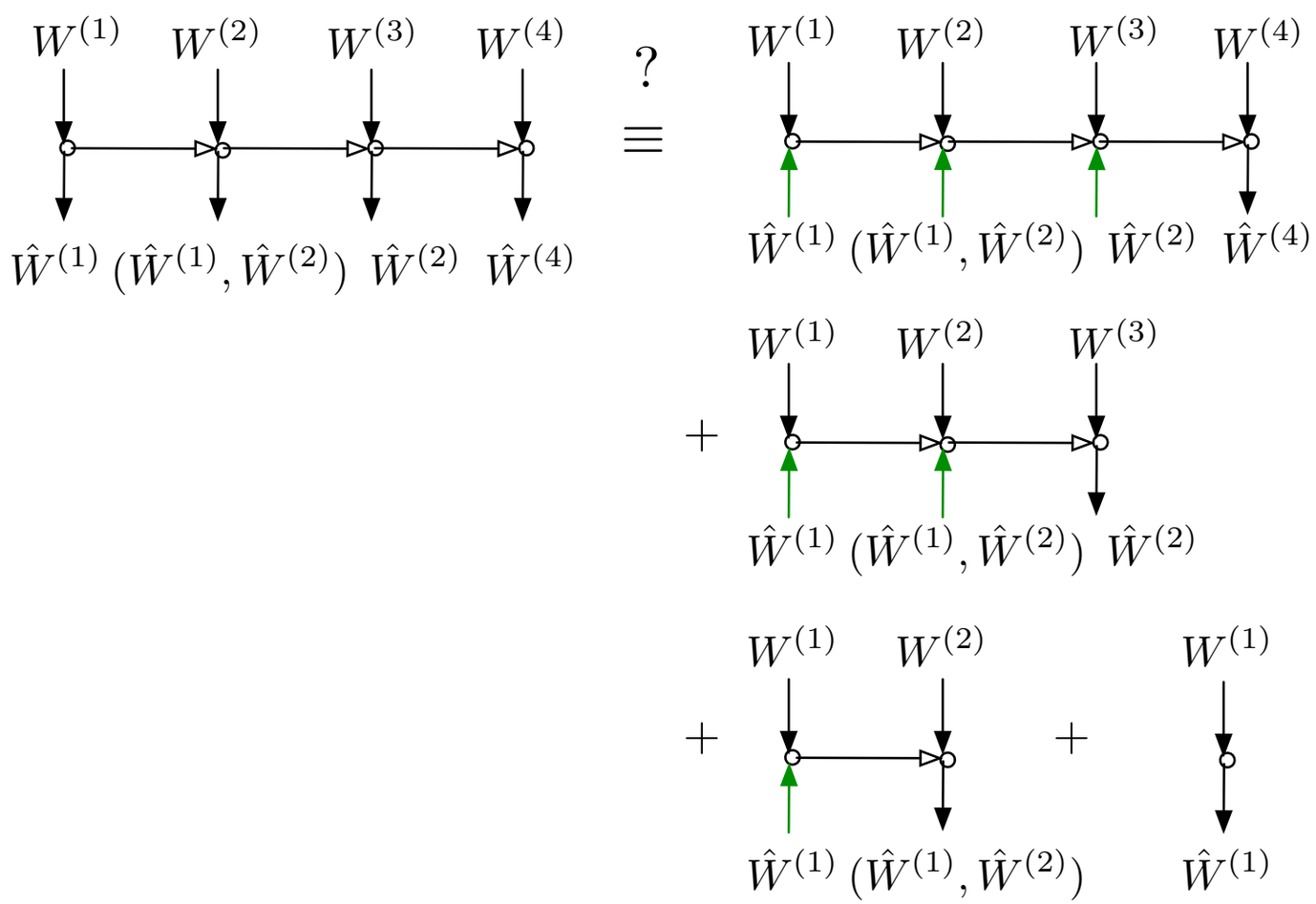


General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

- Apply similar approach to other network problems

  - Network-Channel separation
  - Network Decomposition
  - Feedback
  - Side Information



Our results:

- Holds for several classes of demand [BEGK'07]
- Does not hold in general

General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

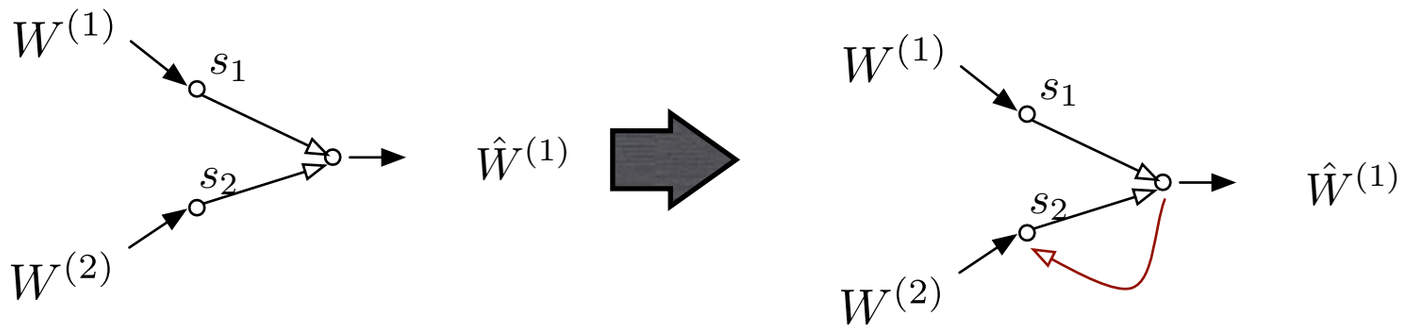
- Apply similar approach to other network problems

  - Network-Channel separation

  - Network Decomposition

  - Feedback

  - Side Information



Observation:

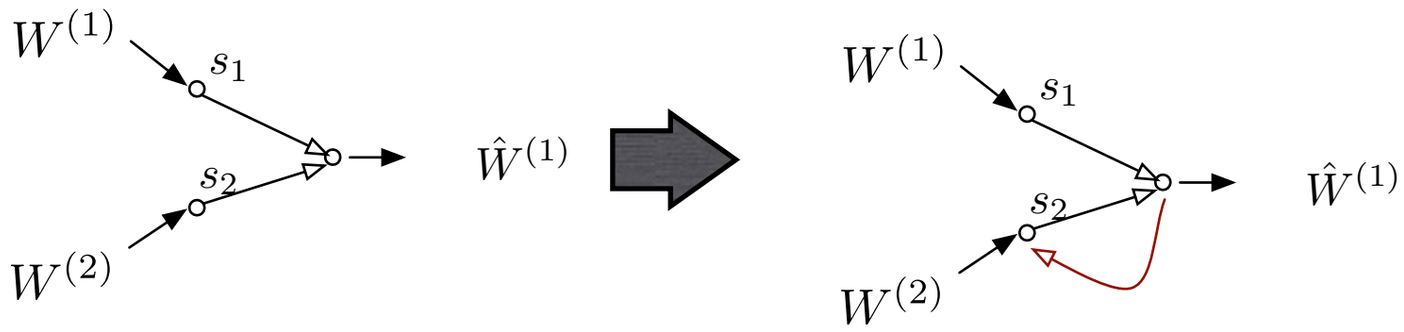
- Feedback increases capacity [BE'09]
- Exact capacity region for the above network

General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

- Apply similar approach to other network problems

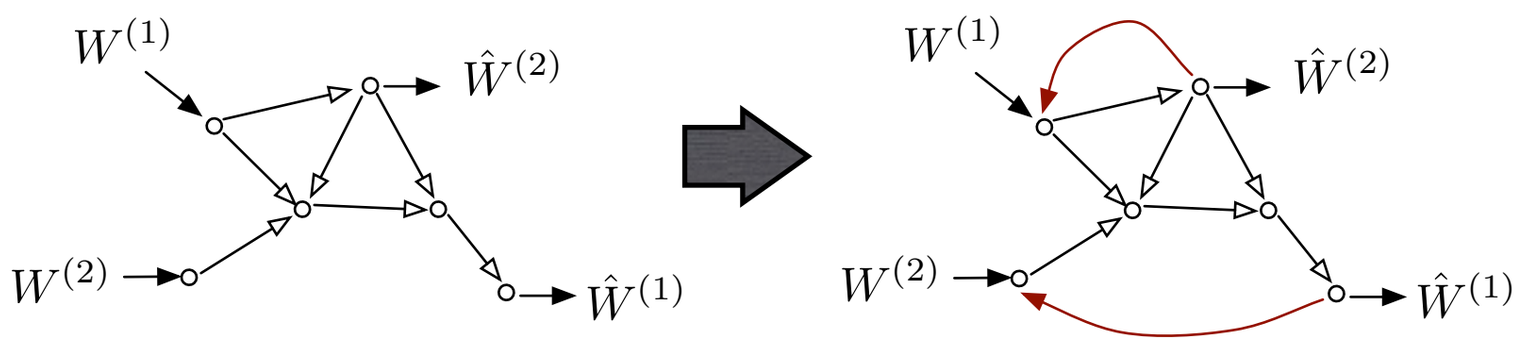
  - Network-Channel separation
  - Network Decomposition
  - Feedback
  - Side Information



Observation:

- Feedback increases capacity [BE'09]
- Exact capacity region for the above network

Bounds on capacities for a class of networks with feedback



General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

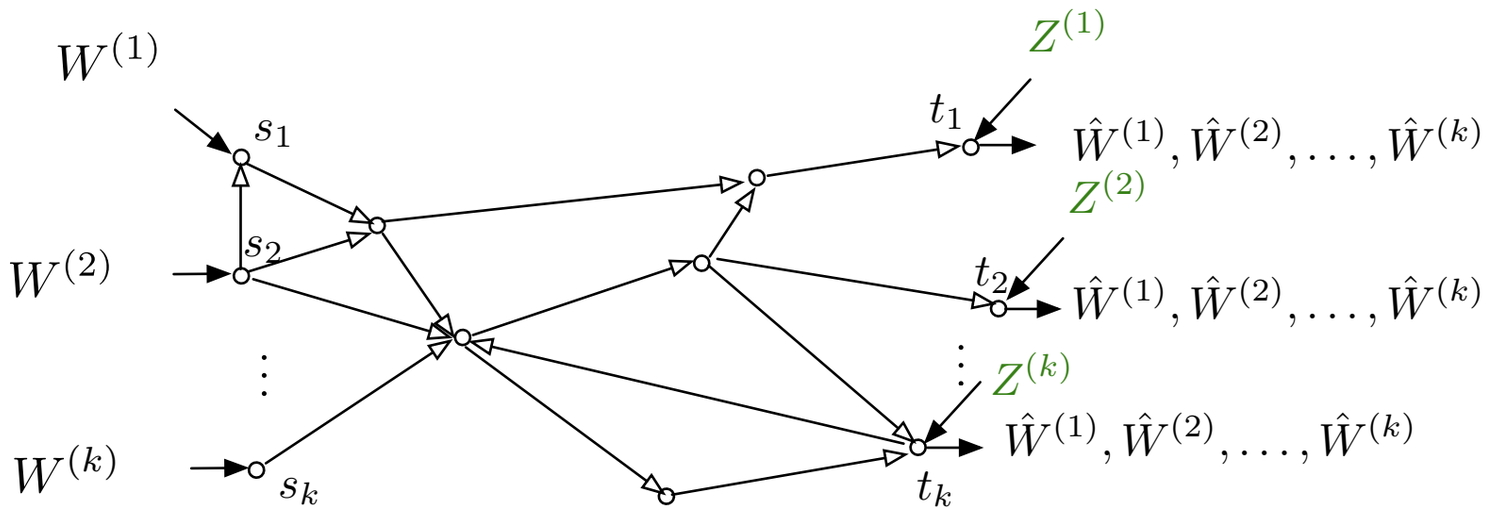
- Apply similar approach to other network problems

  - Network-Channel separation

  - Network Decomposition

  - Feedback

  - Side Information



Multicast with side information only at sinks

- Exact capacity region: cut-set bounds [BE'08]

General philosophy

- Bounds and techniques for one problem => Bounds and techniques for another problem

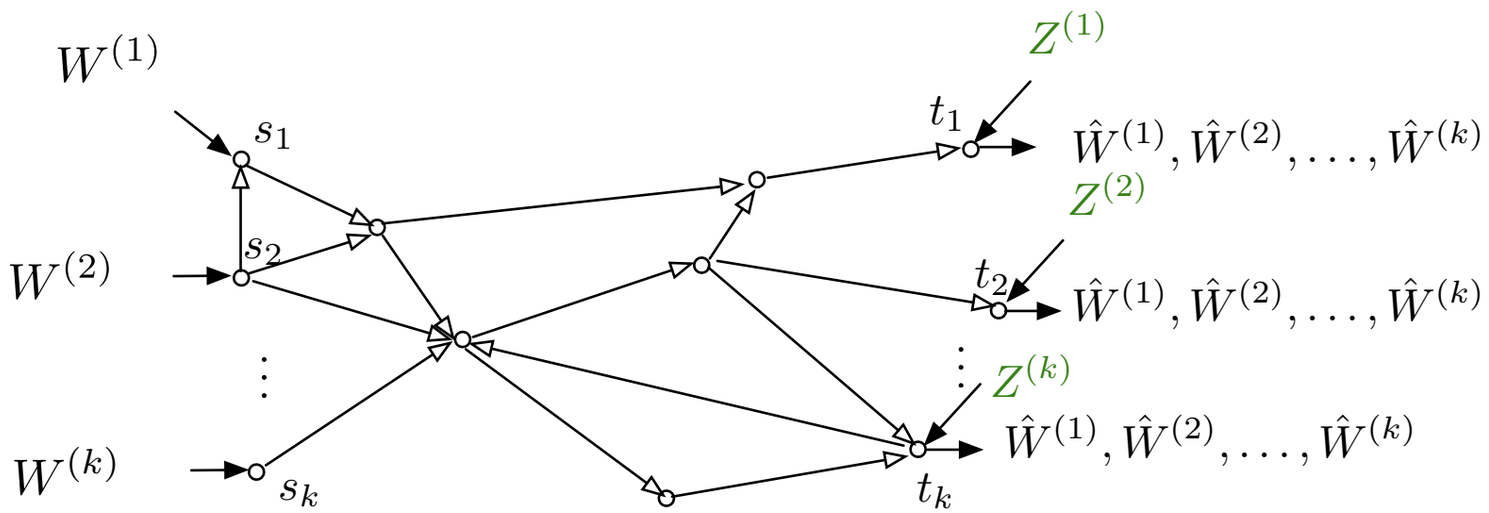
- Apply similar approach to other network problems

  - Network-Channel separation

  - Network Decomposition

  - Feedback

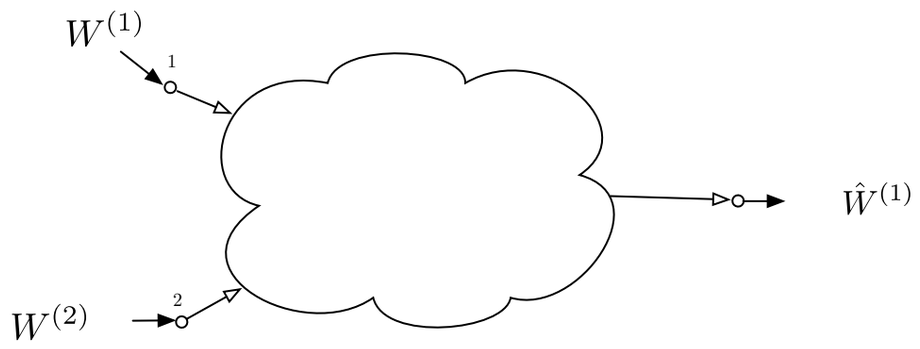
  - Side Information



Multicast with side information only at sinks

- Exact capacity region: cut-set bounds [BE'08]

Bounds when side information not at sink



Thank you!