

---

# On Source and Channel Codes for rate-adaptive asymmetric and symmetric Slepian-Wolf Coding

Christine Guillemot

IRISA/INRIA, Campus universitaire de Beaulieu, 35042 Rennes Cédex, FRANCE  
Christine.Guillemot@irisa.fr

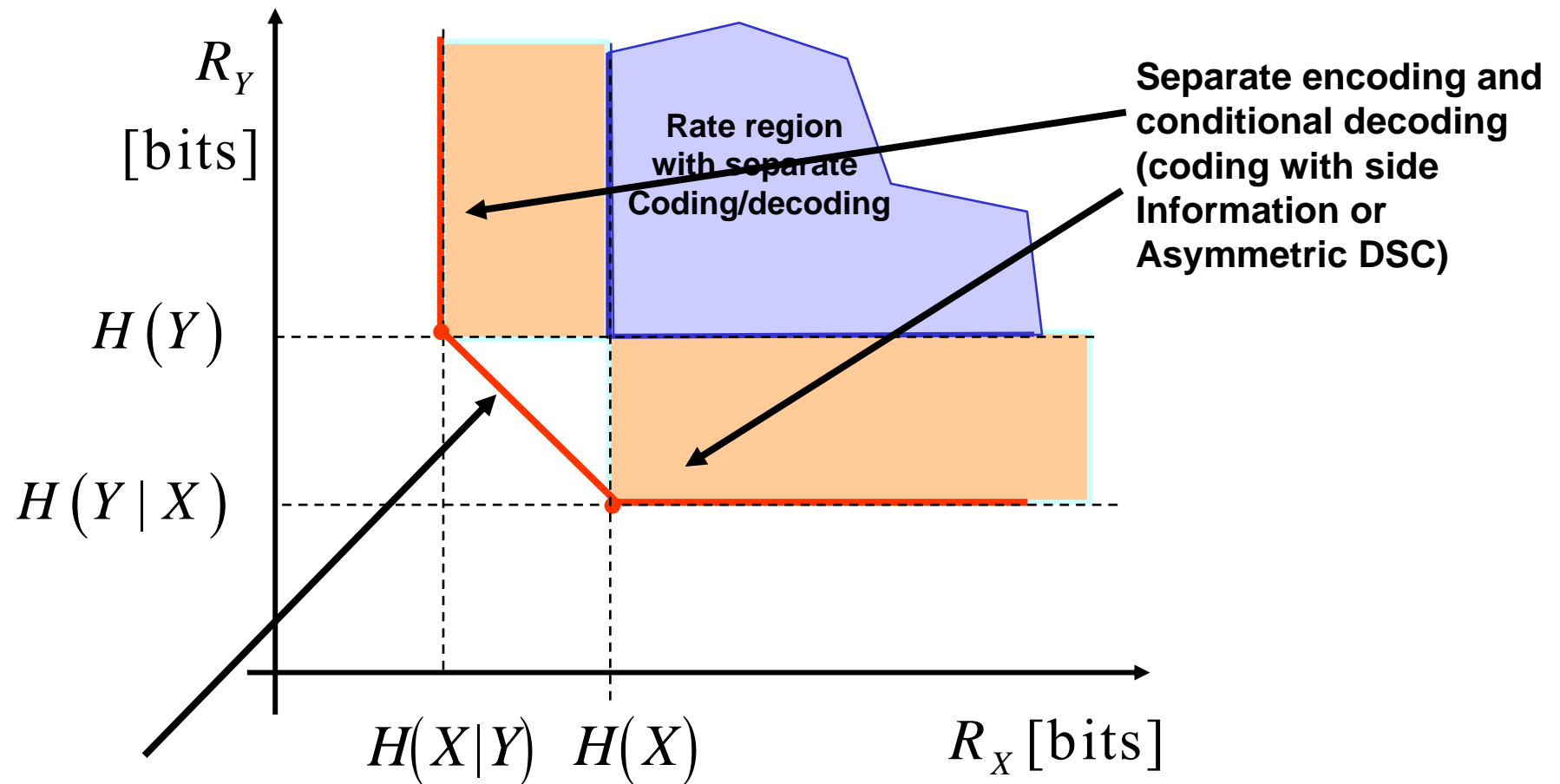
With the collaboration of A. Roumy, K. Lajnef, S. Malinowski, V. Toto-Zarasoia,  
X. Artigas, L. Torres

# Outline

---

- Introduction to the problem of Slepian-Wolf coding
- Solutions based on channel codes (memoryless sources)
  - asymmetric SW coding revisited
  - A Turbo-Syndrome approach
  - SW coding for arbitrarily correlated sources and with flexible rate allocation between the two sources
- Solutions based on source codes (memory sources)
  - Overlapped arithmetic codes
  - Punctured arithmetic codes

# Slepian-Wolf theorem (1973)



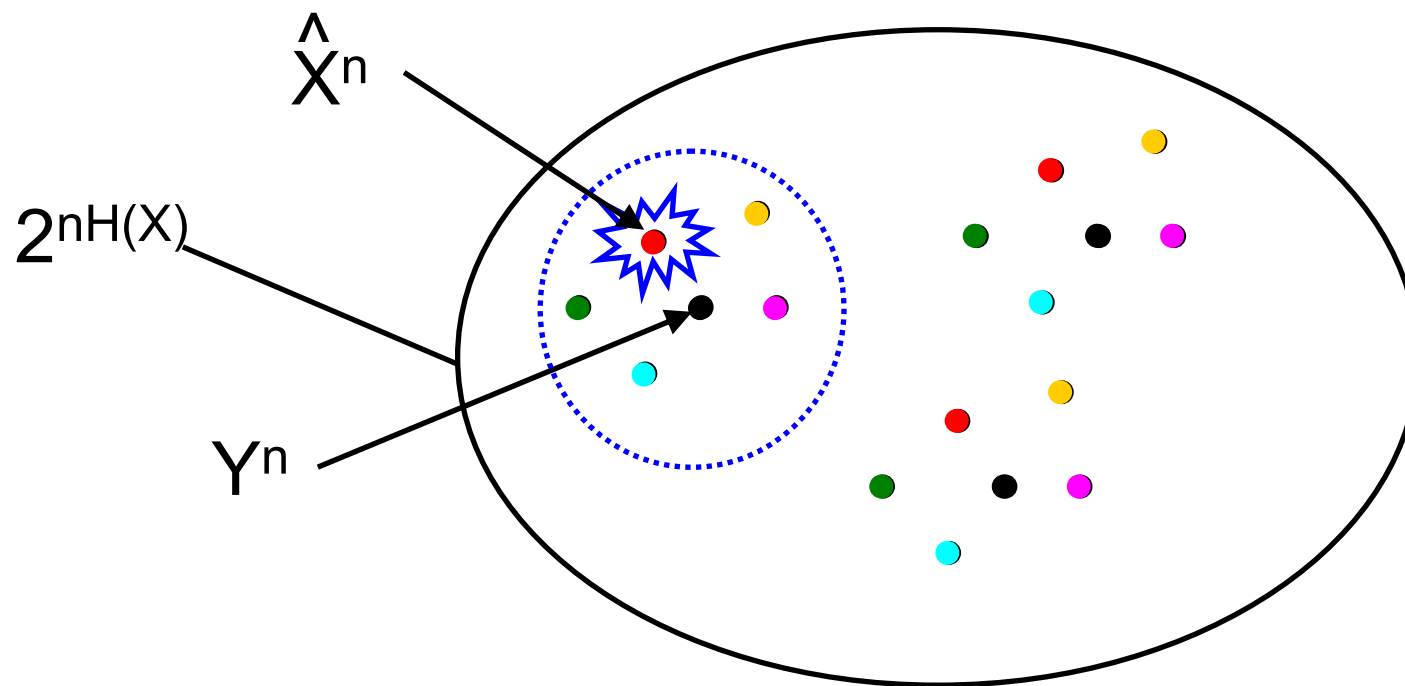
S-W coding with vanishing error probability  
(symmetric DSC)

$$R_X + R_Y = H(X, Y)$$

# Slepian-Wolf theorem (1973)

---

- Decoder finds the sequence in the bin ('red') that is jointly typical with  $Y^n$ .



# How to construct the bins?

---

- Wyner first suggested parity-check codes in 1974 to approach the corner points in the S-W rate region
  - Generate bins as cosets of a “good” parity check code
  - A sequence of  $n$  input bits is mapped into its corresponding  $(n-k)$  syndrome bits, leading to a compression ratio of  $n : (n-k)$
  - The decoder searches for the sequence in the coset that is jointly typical with the side information sequence  $Y^n$

=> This is called the **syndrome approach**

A capacity-achieving code can be turned into an optimal DSC code

# In practice: Binary linear codes (e.g. rate $k/n$ )

---

- The code  $C_0$  contains  $2^k$  codewords in the set of  $2^n$  elements  
 $C_0 = \{ \mathbf{x} : \mathbf{x} = \mathbf{u} \mathbf{G} \}$ ,  $\mathbf{G}$ = generator matrix (size  $k \times n$ )  
 $= \{ \mathbf{x} : \mathbf{H}\mathbf{x}^T = \mathbf{0} \}$ ,  $\mathbf{H}$ = parity check matrix (size  $n-k \times n$ )

- In communication, the decoder searches for:

$$\hat{\mathbf{X}} = \arg \min \{ \mathbf{X} : \mathbf{X} \in C_0 \} \quad d(\mathbf{X}, \mathbf{Y})$$

- In DSC, the encoder computes the syndrome

$$\mathbf{H} \mathbf{X}^T = \mathbf{s}$$

and the decoder has to search for

$$\hat{\mathbf{X}} = \arg \min \{ \mathbf{X} : \mathbf{X} \in C_s \} \quad d(\mathbf{X}, \mathbf{Y}) \quad \dots \text{NOT USUAL}$$

# DISCUS: DIstributed Source Coding Using Syndromes

---

[Ramchandran & Pradhan 1999]

- Goal: find a way to enumerate all possible codewords in the coset  $C_s$  of a convolutional code (or multilevel code)

- Idea:

$$\begin{aligned} \text{Let } C_s &= \{ x : Hx^T = s \} \\ &= C_0 + t, \text{ where } t \in C_s \end{aligned}$$

If the code is systematic, then a candidate is  $t = [0 \ s]$

In practice: add  $[0 \ s]$  to all edges in the trellis (**trellis labeled differently**).

- if  $s=0$ , this is the usual trellis (called *primary trellis*)

- for  $s \neq 0$ , this is the *secondary trellis enumerating the codewords in the coset  $C_s$*

# Syndrome former/Inverse syndrome former

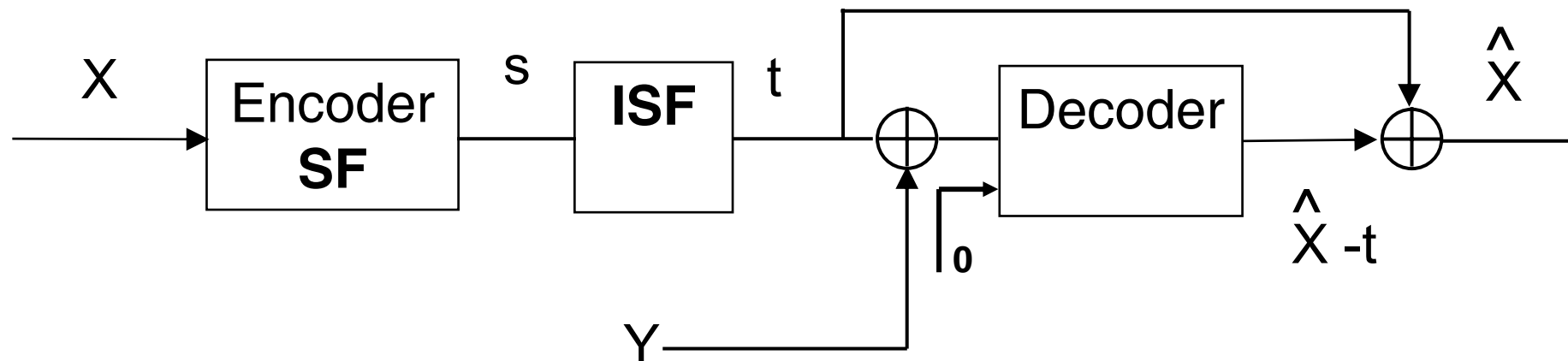
- Decoder:  $\hat{X} = \arg \min \{X: X \in C_s\} d(X, Y)$   
 $= \arg \min \{X: X \in C_s\} d(X-t, Y-t)$

[Tu, Li, Blum 05]

Change of variable  $b=X-t$ , with  $t \in C_s$  ( $t=[0_s]$  for systematic codes)

$$\hat{b} = \arg \min \{b: b \in C_0\} d(b, Y-t) = \hat{X} - t$$

use classical channel decoders



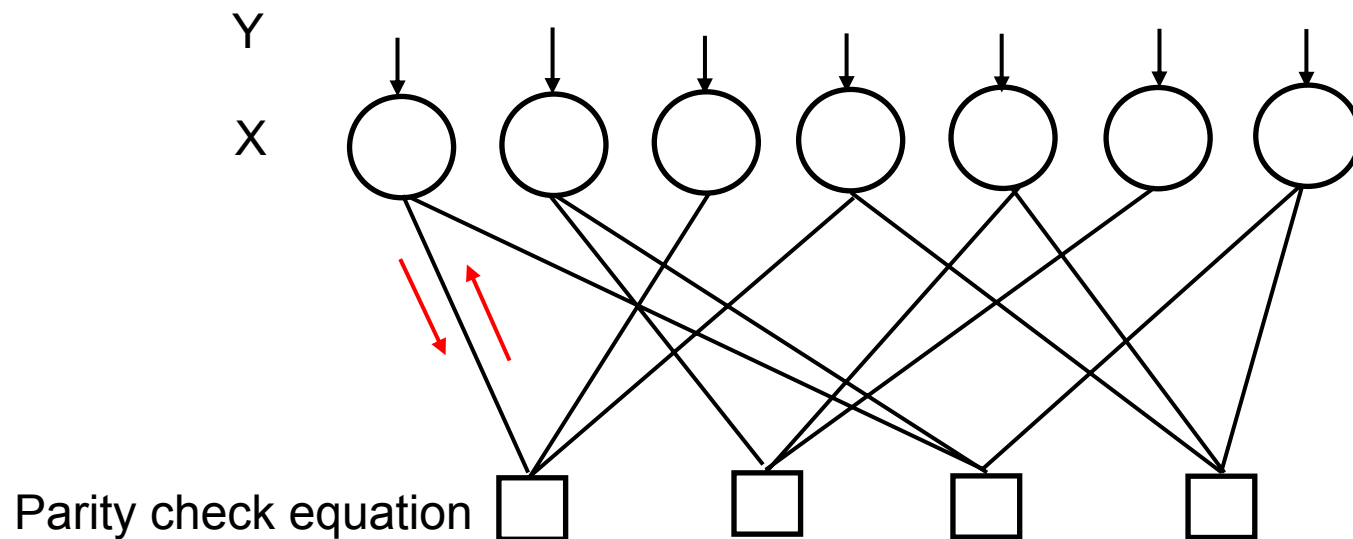
# Syndrome approach with LDPC code

---

[Liveris, Xiong, Georghiades 2002]

Low density Parity Check code:

- defined by their low density matrix  $H$ :  $Hx^T = 0$
- Efficiently decoded by the sum-product algorithm



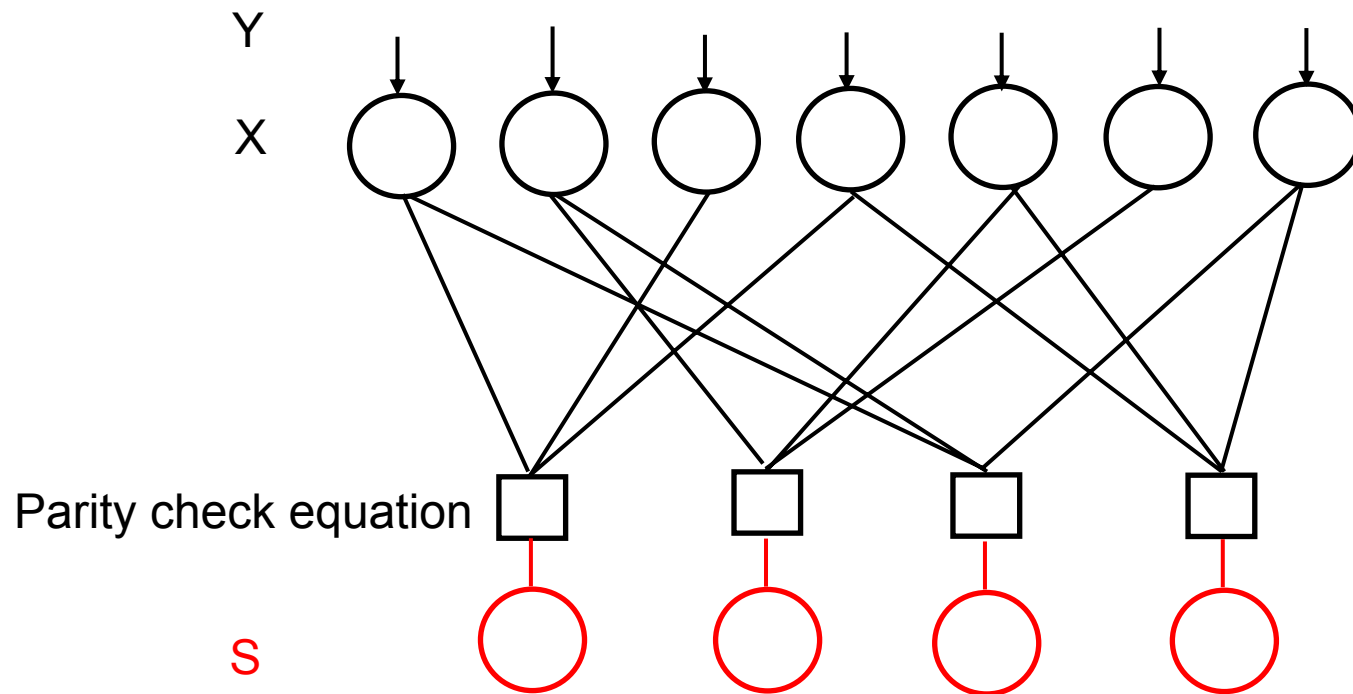
# Syndrome approach with LDPC codes

[Liveris, Xiong, Georghiades 2002]

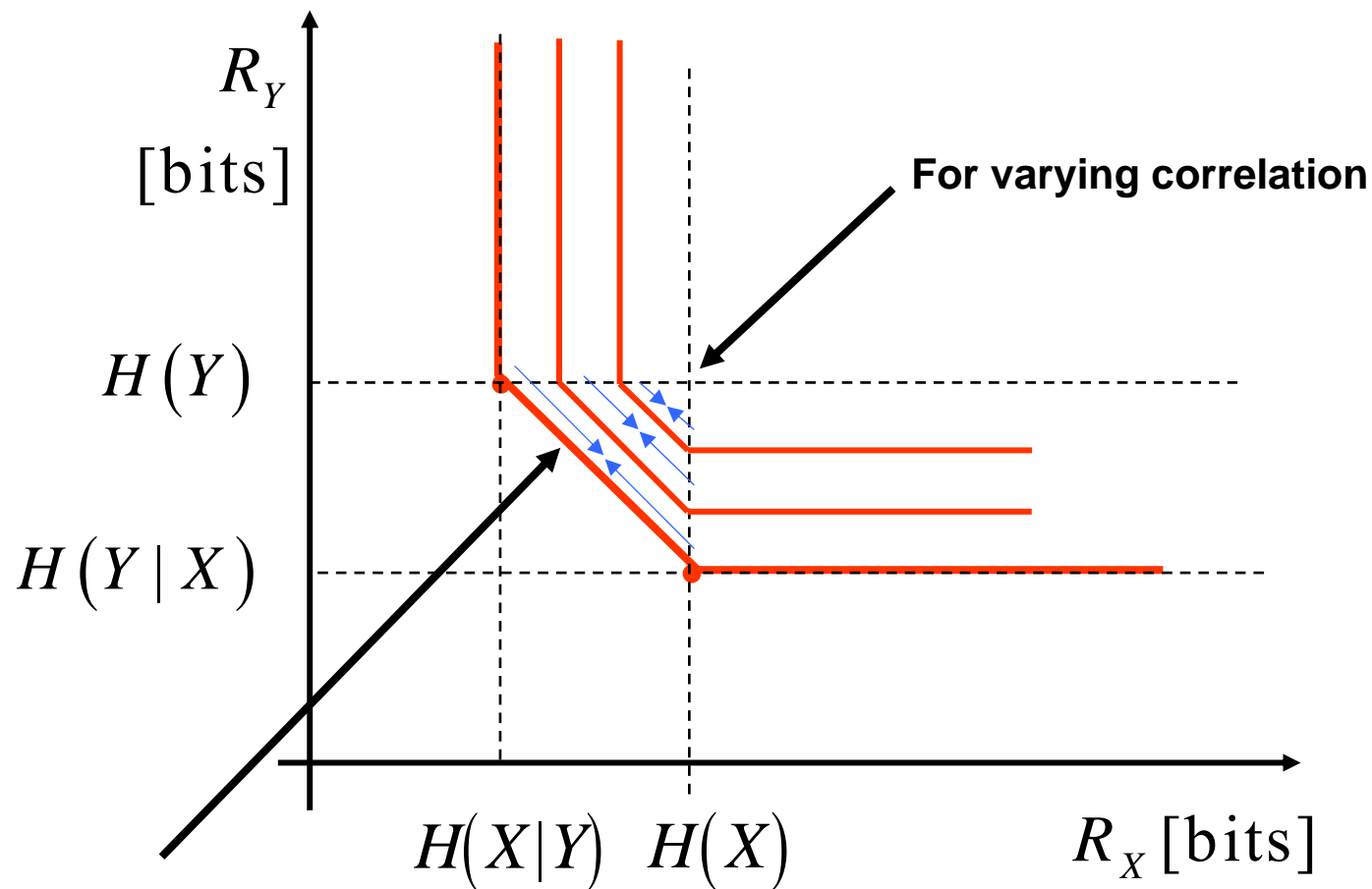
Low density Parity Check code for the S-W problem

- defined by their low density matrix  $H$ :  $Hx^T = s$

Idea: add the syndrome bits known perfectly at the decoder



# Rate-adaptive asymmetric and symmetric coding for arbitrarily correlated sources



Reaching any point of the segment  $R_X + R_Y = H(X, Y)$  for flexible rate allocation between X and Y

# Rate-adaptive asymmetric coding for arbitrarily correlated sources

---

- Parity approach:
    - To adapt the rate: puncture the parity bits
  - Syndrome approach:
    - Puncture the parity bits and compute the new H matrix [Li, Alqamzi 05, Jiang, He, Jagmohan 07...]
      - The resulting code is not *incremental*
    - Puncture the syndromes
- BUT** the syndrome bits are not protected
- Protect them: LDPC with accumulator code [Varodayan & Girod 06]
  - A turbo-syndrome approach [Roumy, Lajnef, Guillemot, Asilomar 07]
    - Decode in the union of all the cosets
    - Single code that is both rate-adaptive and incremental, with the constraint of linear (in the code blocklength) coding and decoding complexity.

# Rate-adaptive **asymmetric** coding for arbitrarily correlated sources

---

- A **Turbo**-Syndrome approach [Roumy, Lajnef, Guillemot 07]
  - A classical trellis is a compact representation of all possible codewords in a convolutional code and is based on the generator polynomials of the code
  - Here, novel construction of a syndrome trellis based on the parity check polynomial matrix without having to expand the matrix into a matrix of equivalent block codes of large dimensions
  - Decode in the union of all the cosets

# Rate-adaptive **asymmetric** coding for arbitrarily correlated sources

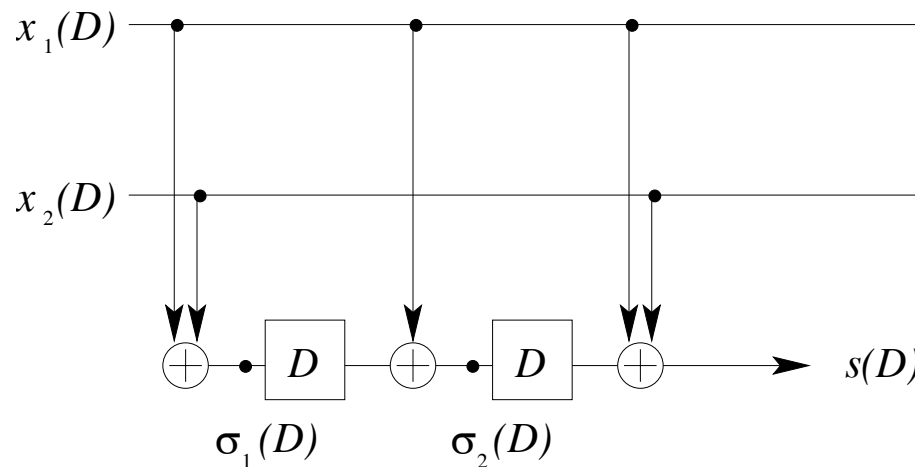
- Example for a rate 1/2 code

$$H = [7,5]; \quad H(D) = [1 + D + D^2, 1 + D^2]$$

$$x_1(D) = \sum_{l \in \mathbb{N}} x_{2l} D^l$$

$$x_2(D) = \sum_{l \in \mathbb{N}} x_{2l+1} D^l$$

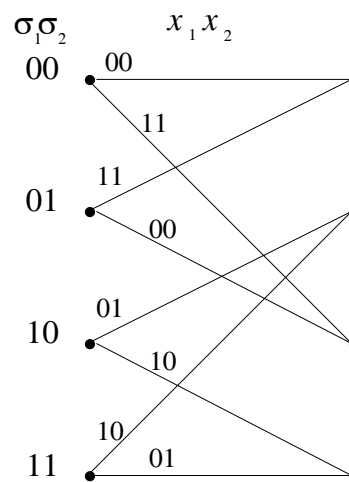
$$s(D) = (1 + D + D^2)x_1(D) + (1 + D^2)x_2(D)$$



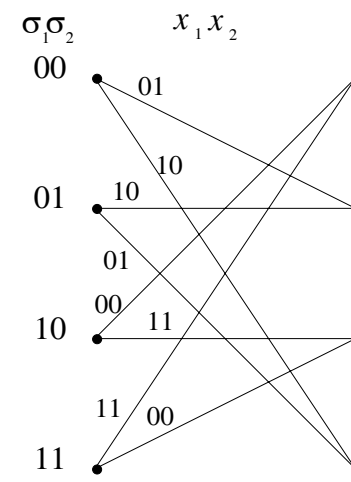
Factorization of the syndrome polynomial into

$$s(D) = x_1(D) + x_2(D) + \overbrace{D(x_1 + D(x_1 + x_2))}^{\sigma_1(D)}$$

$$\underbrace{\hspace{10em}}_{\sigma_2(D)}$$



S=0

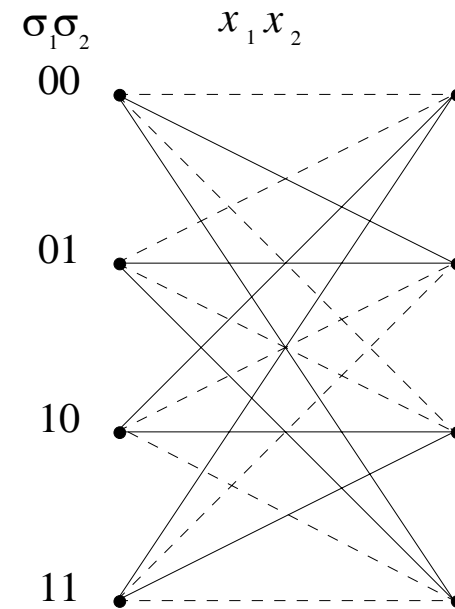


S=1

# Rate-adaptive **asymmetric** coding for arbitrarily correlated sources

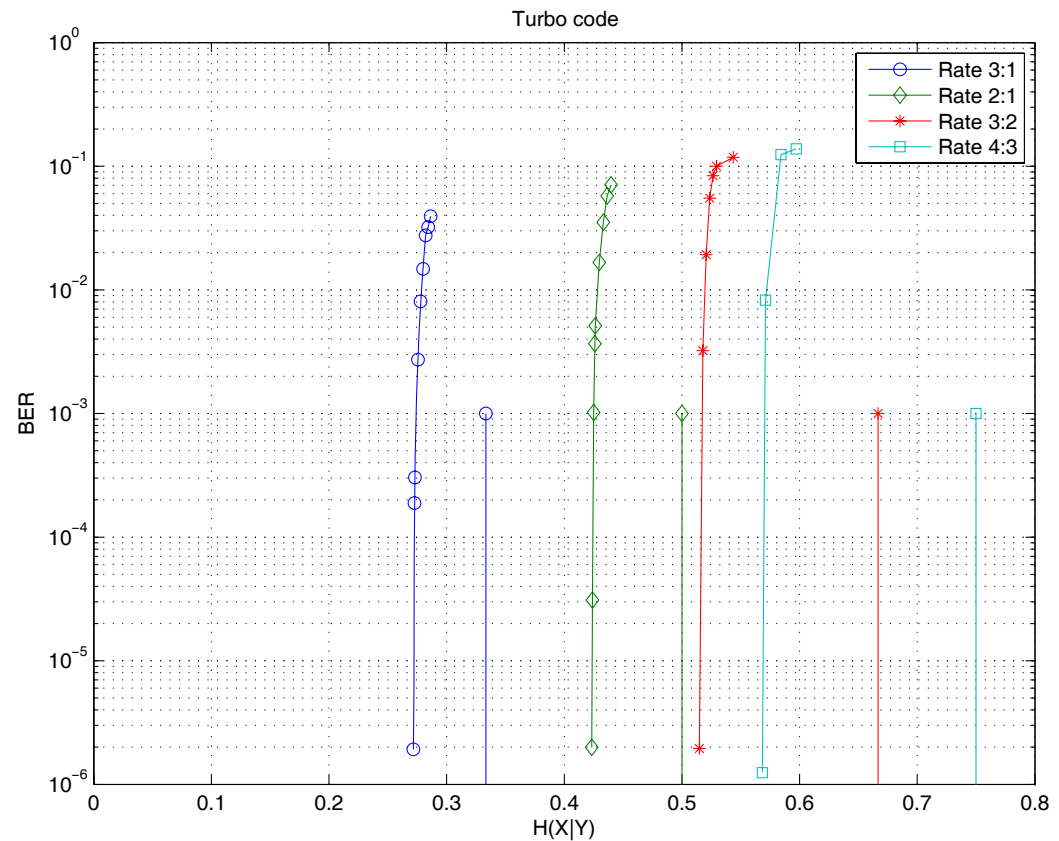
---

- Rate adaptation by puncturing
  - In principle, search for the closest sequence in a coset indexed by the syndrome value
  - BUT the number of cosets (decoding complexity) grows exponentially with the number of punctured syndrome bits
  - Search in a union of cosets
    - Trellis section = union of two trellis sections for punctured positions
    - Complexity grows linearly with the number of punctured positions

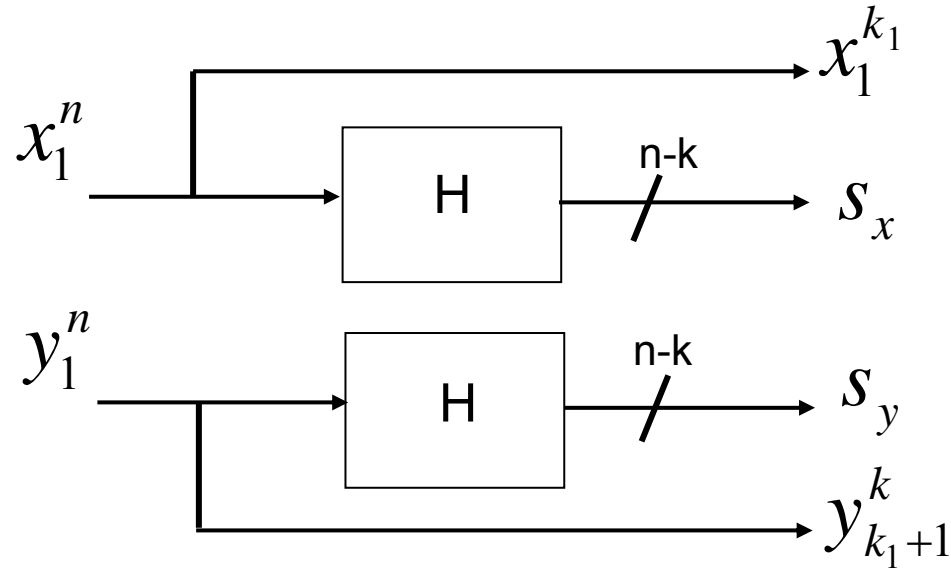


# Rate-adaptive **asymmetric** coding for arbitrarily correlated sources

- Method equivalent to the parity approach if  $G=[I H]$
- With syndrome approach, not necessary to have systematic codes; Any syndrome bit position can be punctured
- Syndrome approach: optimal [Wyner 1974]



# Symmetric Slepian-Wolf Coding

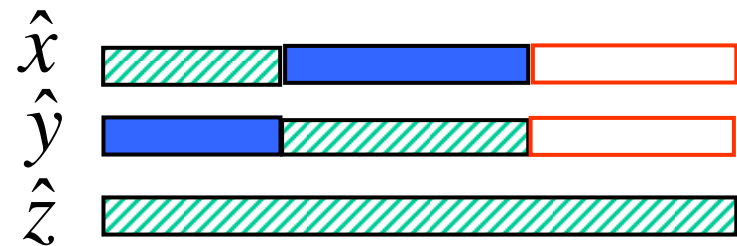
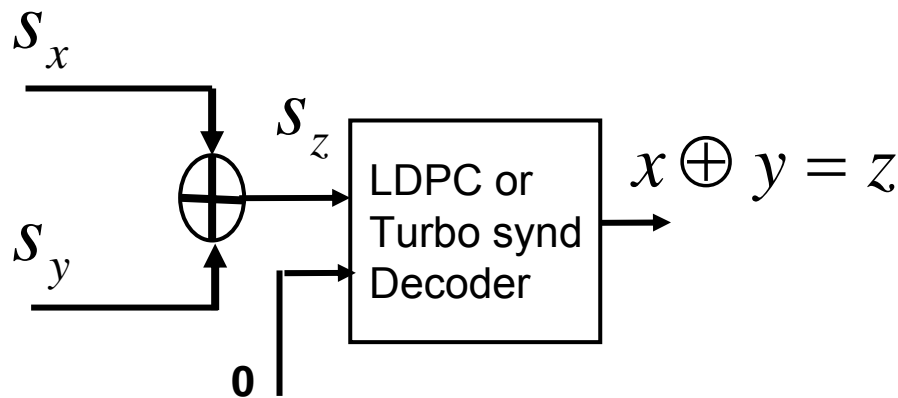


[Gehrig, Dragotti 05]

$$R = \frac{(n-k+k_1)}{n} + \frac{(n-k_1)}{n}$$

$$= 1 + \frac{(n-k)}{n}$$

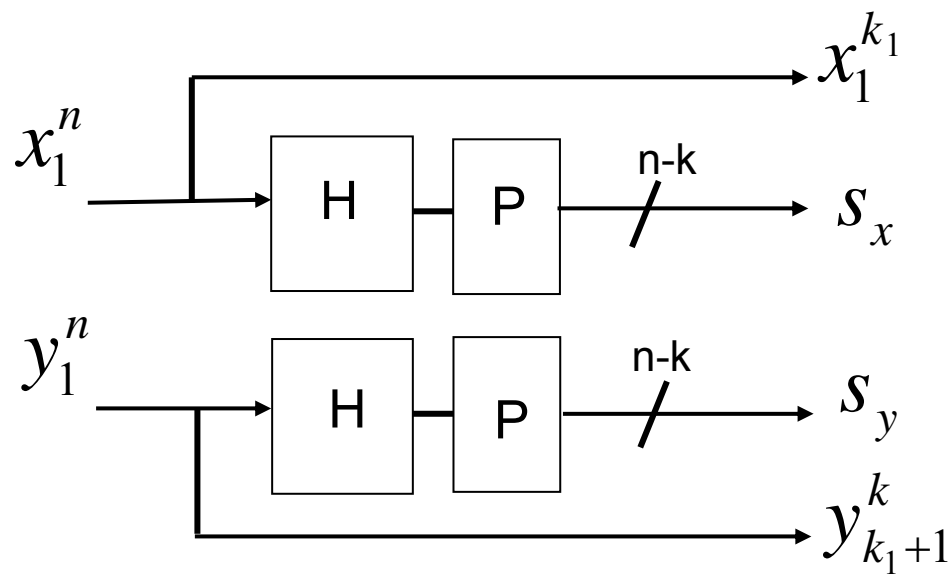
n-k equations  
with n-k unknowns



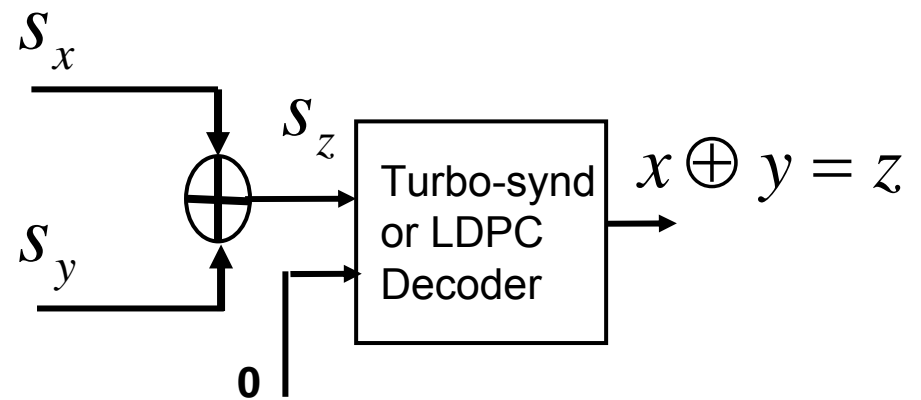
$$H = [AB] \Rightarrow s_x = Ax_1^k \oplus Bx_{k+1}^n$$

$$x_{k+1}^n = B^{-1} (Ax_1^k \oplus s_x)$$

# Rate-adaptive symmetric coding for arbitrarily correlated sources



[Toto-Zaratosoa, Roumy, Guillemot 07]

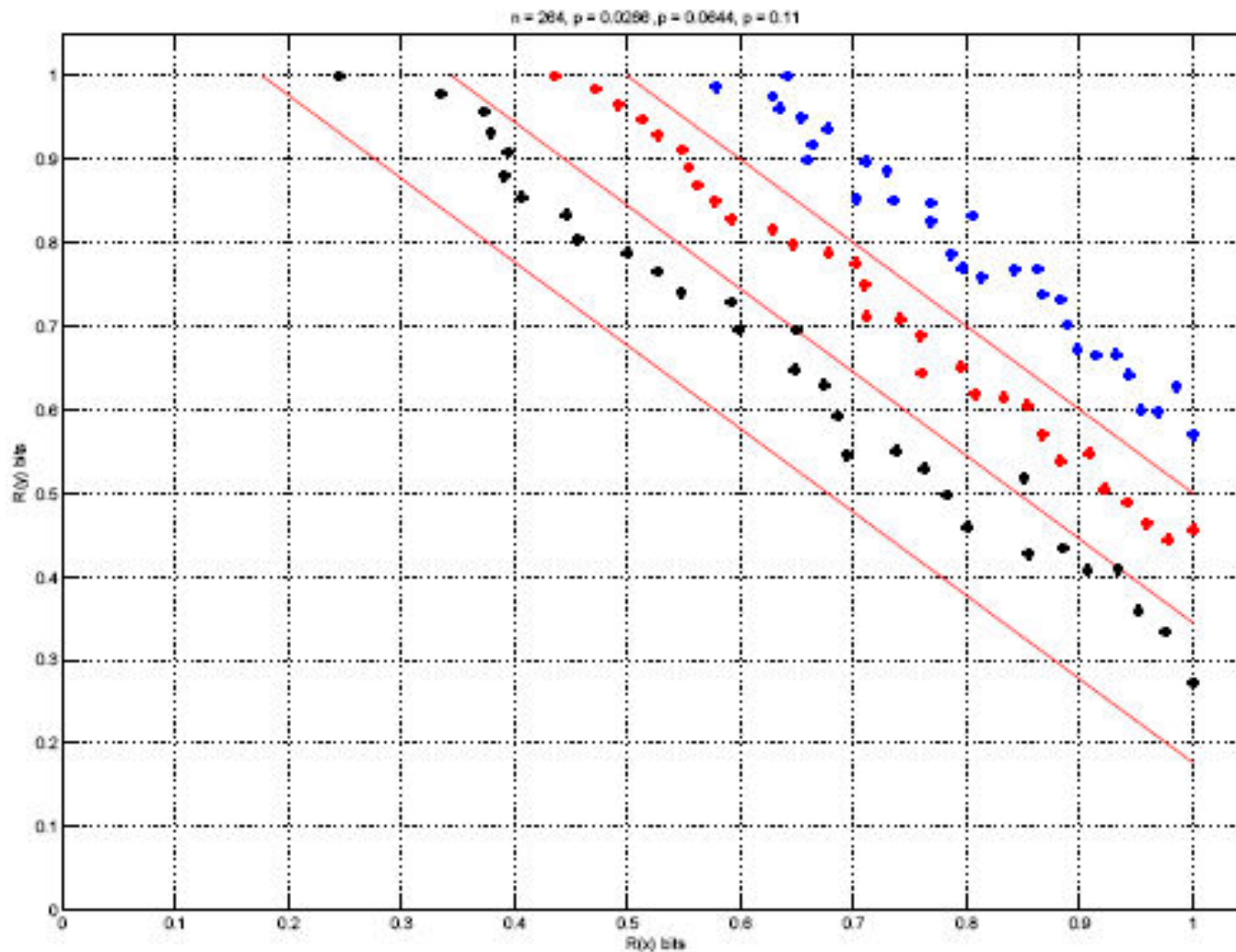


Puncturing some syndrome bits  $\Rightarrow$  Set of matrices  $H_i$

$$x_{k+1}^n = B^{-1}(Ax_1^k \oplus s_x)$$

Choice of the puncturing positions so that the matrix B de dimension  $(n-k) \times (n-k)$  is invertible.

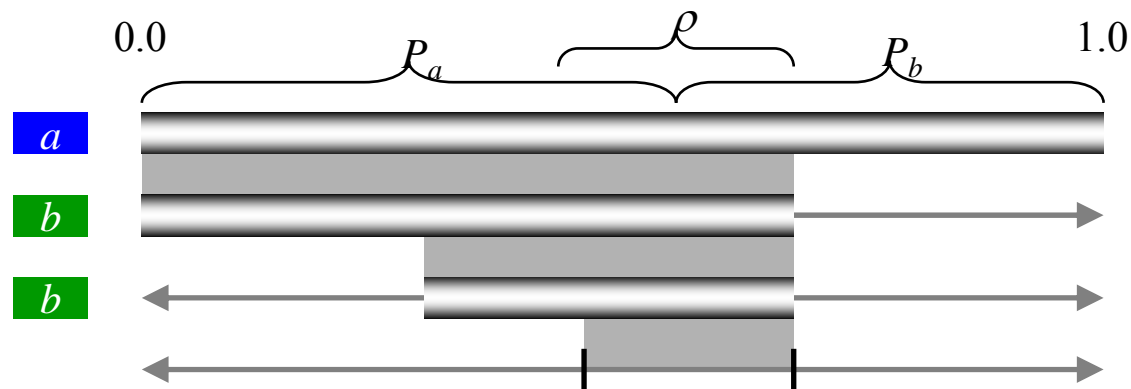
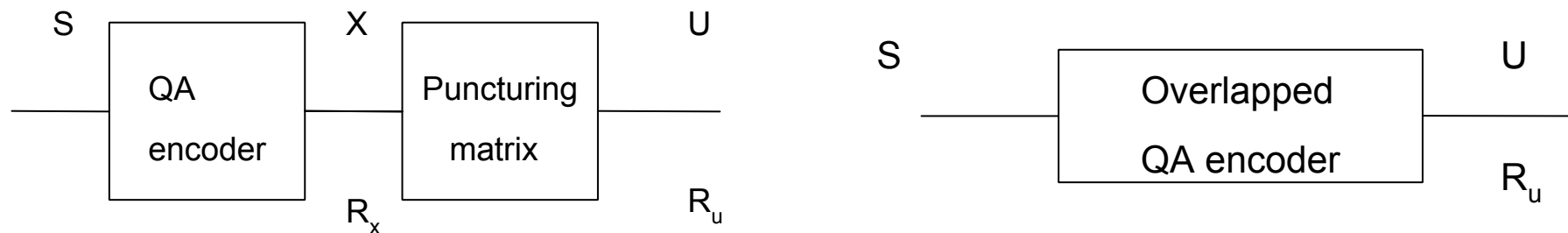
# Rate-adaptive symmetric coding for arbitrarily correlated sources



# SW coding based on source codes

## Arithmetic and quasi-arithmetic codes

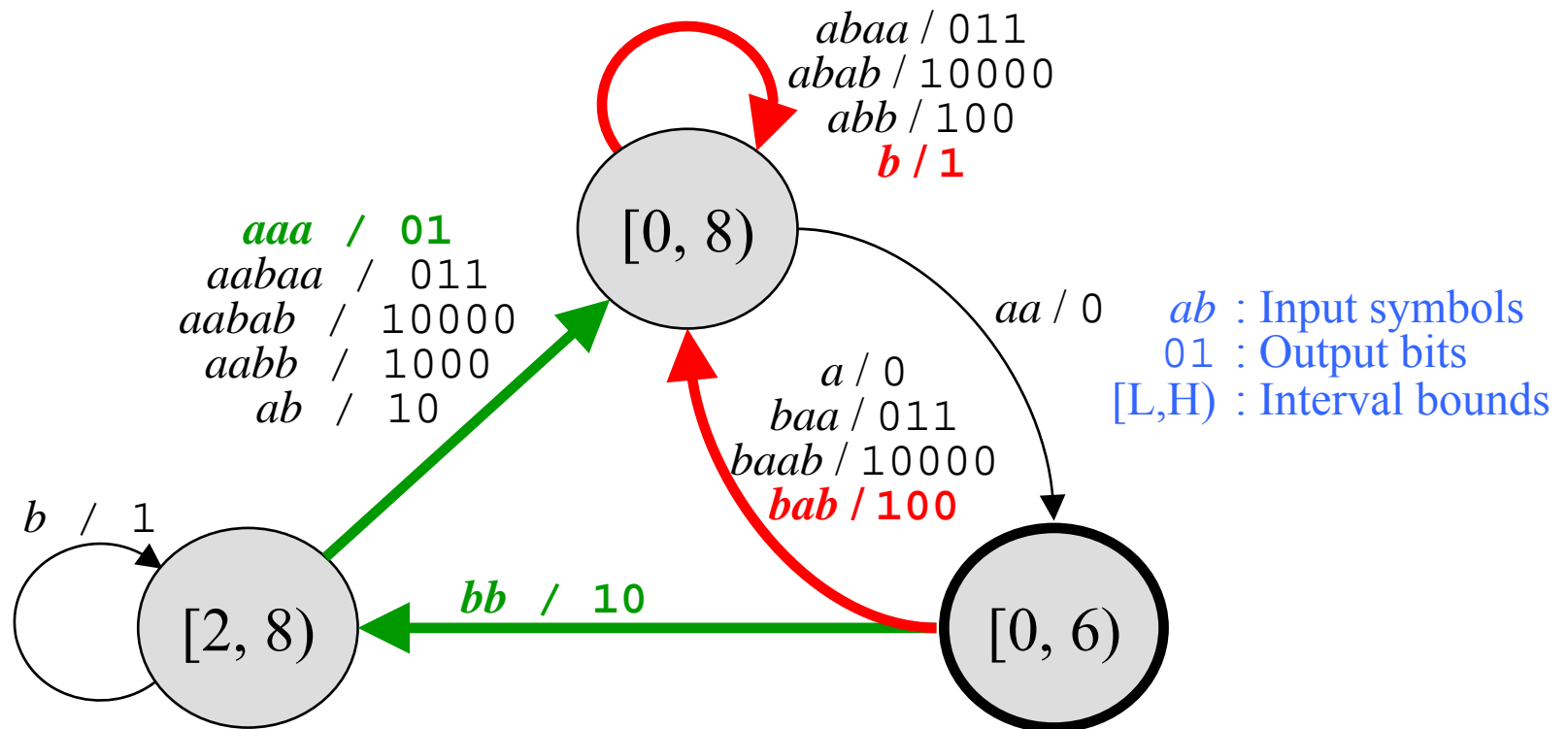
[X. Artigas, S. Malinowski, C. Guillemot, L. Torres, ICIP'07]



$$\begin{cases} \mathcal{I}_{n+1}^a &= [L_n, P_a(H_n - L_n) + \rho N/2) \\ \mathcal{I}_{n+1}^b &= [P_a(H_n - L_n) - \rho N/2, H_n) \end{cases}$$

# SW coding based on source codes

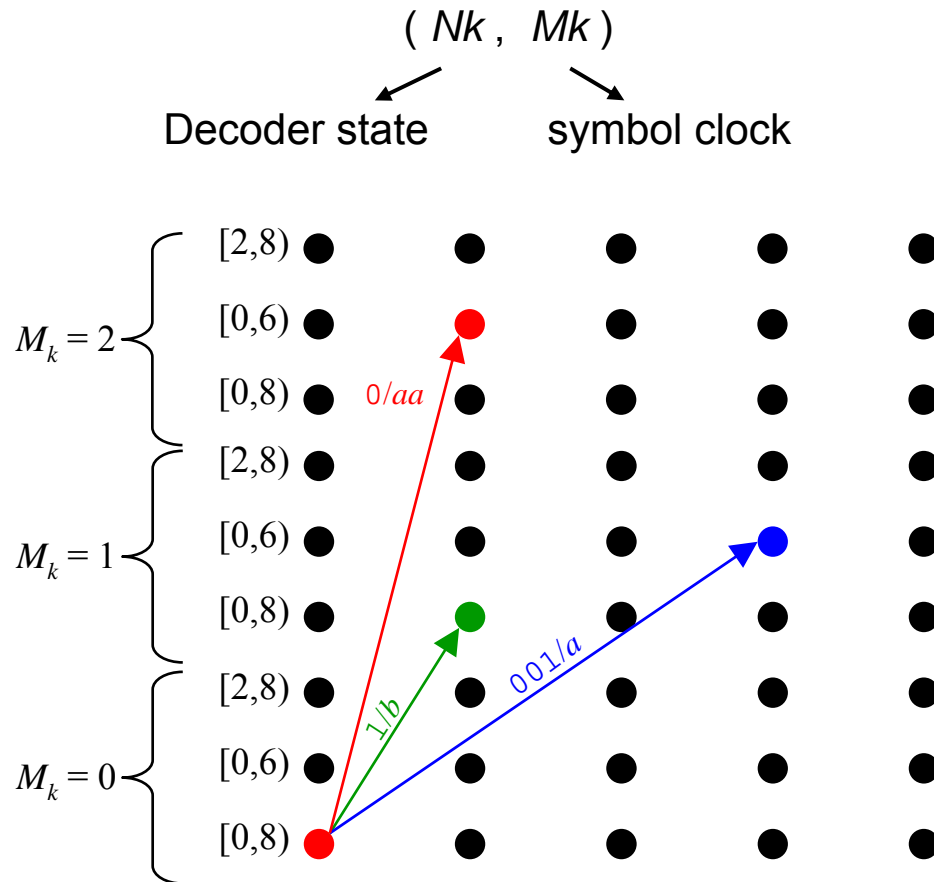
- ❑ Quasi-arithmetic codes (punctured or with overlap) can be represented as FSM
- ❑ Ambiguity on the decoded sequence due to either to overlap or to puncturing



The symbol sequences *babb* and *bbaaa* both produce the bit sequence 1001

# SW coding based on source codes

Similar trellis representations for punctured or overlapped QAC



OQAC: Transition probabilities without SI

$$\gamma = P_a \times P_a ; \gamma = P_b ; \gamma = P_a$$

OQAC: Transition probabilities with SI

$$\gamma = P_a^2 \times P(X_0 = a/Y) \times P(X_1 = a/Y)$$

$$\gamma = P_b \times P(X_0 = b/Y)$$

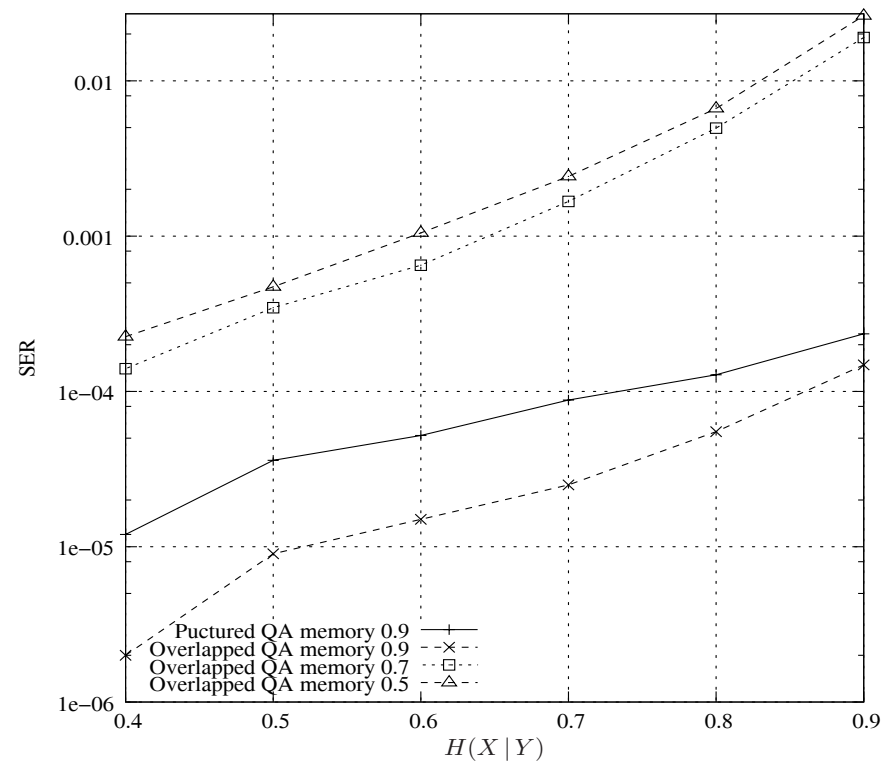
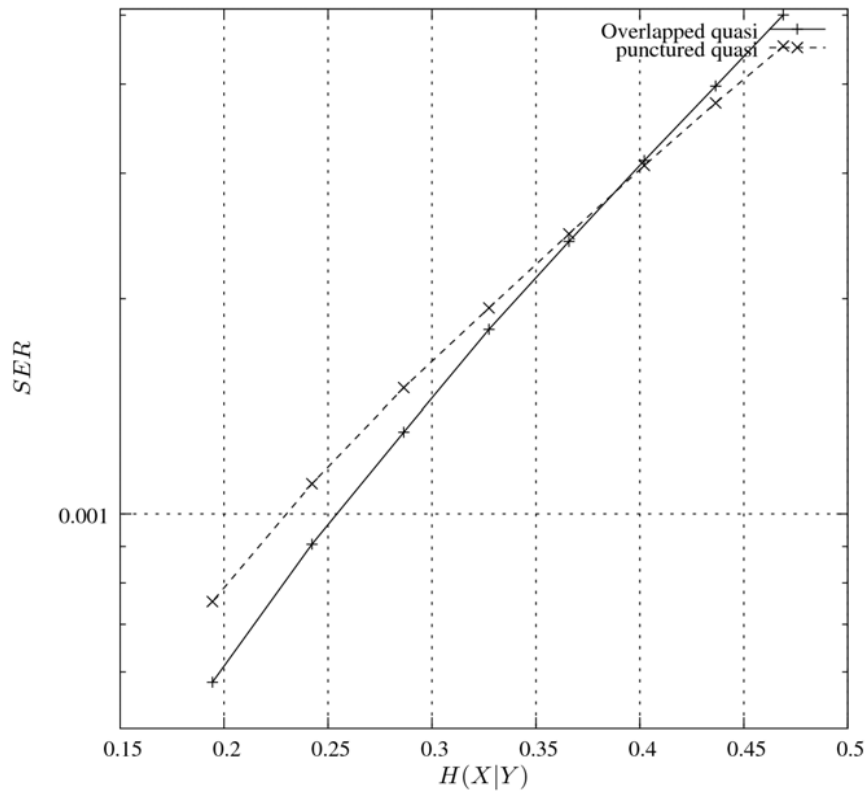
$$\gamma = P_a \times P(X_0 = a/Y)$$

PQAC: Transition probabilities with SI

0.5 probability at position of punctured bits

The transitions have variable-length inputs and outputs

# Decoding performance

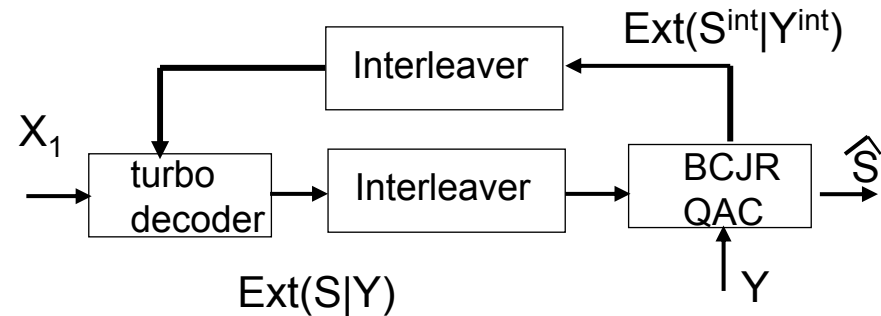
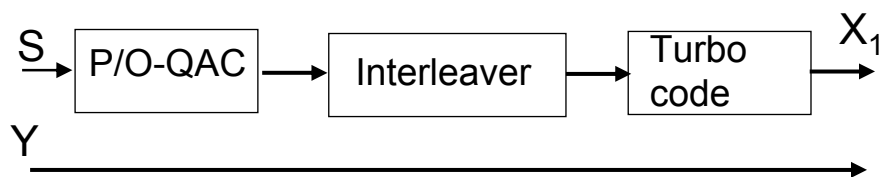
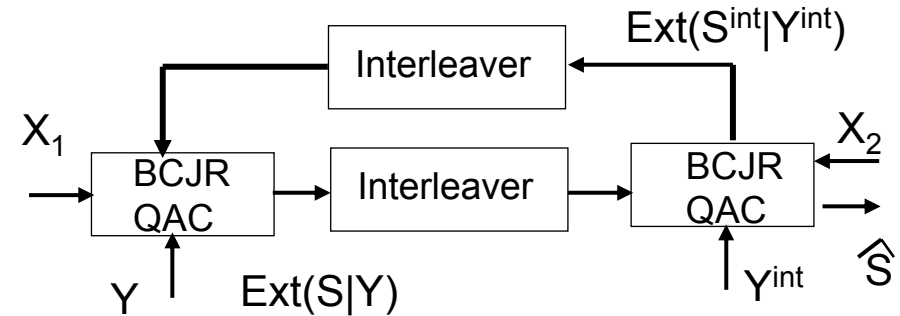
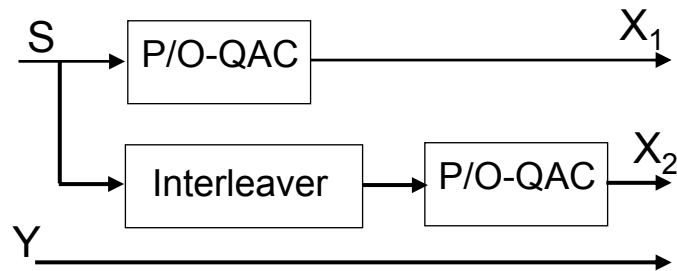


- T=8; memoryless case
- Source entropy = 0.47; Probas: 0.9, 0.1
- Rate of the QA = 0.49, of the OQA = 0.40
- Sequence length = 1000

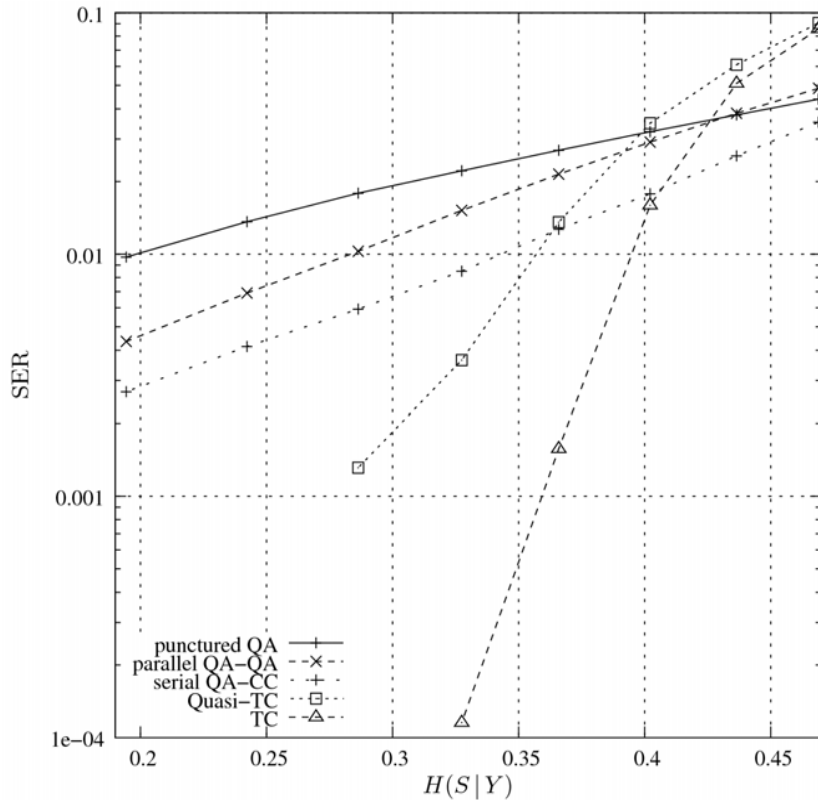
- T=8; memory case
- Probas: 0.9, 0.1, rho=0.9, 0.8, 0.7
- Rate before puncturing: 0.31
- Rate after puncturing: 0.29

# SW coding based on source codes

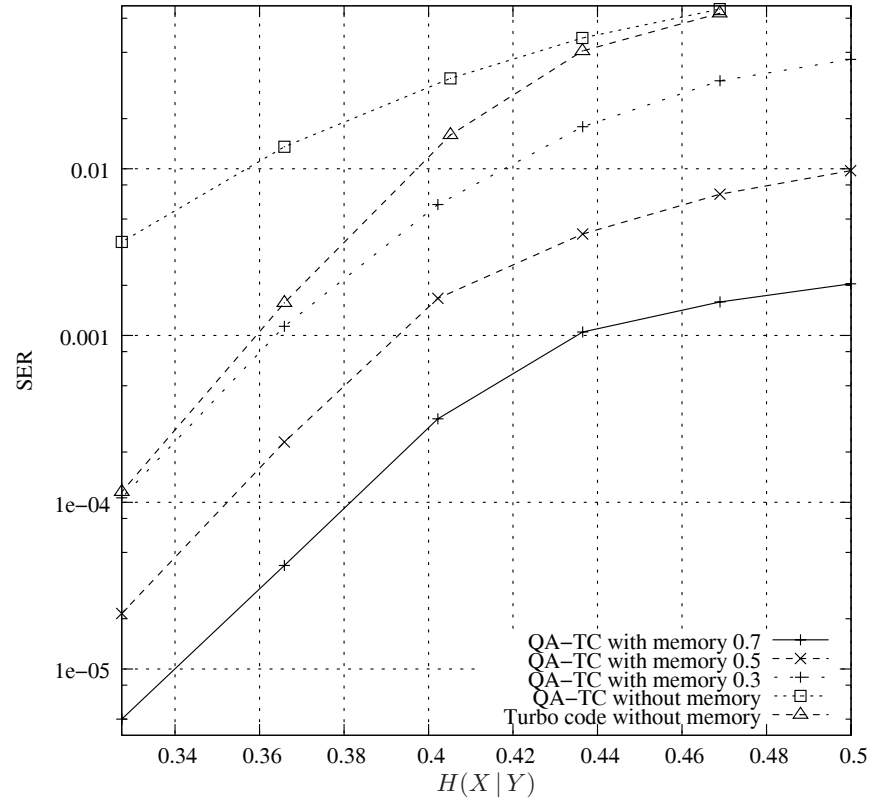
Used in iterative Structures



# Decoding performance



- T=8; memoryless case
- Source entropy = 0.47; Probas: 0.9, 0.1
- Rate of the QA = 0.49, of the OQA = 0.40
- Sequence length = 1000



- T=8; memory case
- Source entropy = 0.47; Probas: 0.7, 0.3
- Rho=0.9
- Rate before puncturing: 0.9
- Rate after puncturing: 0.5

# Conclusion

---

- For memoryless sources
  - Syndrome-based solutions based on channel codes for **arbitrarily correlated sources** and **flexible rate allocation** between the two sources
- For sources with memory
  - Solutions based on source codes
    - Overlapped arithmetic codes
    - Punctured arithmetic codes
  - Iterative source-channel coding for *good* distance properties

# Acknowledgements

---

- *Thanks to my colleague*
  - *A. Roumy*
  
- *And to PhD students*
  - *K. Lajnef*
  - *S. Malinowski*
  - *V. Toto-Zarasoa*
  
- *Thanks to the colleagues of the European IST-DISCOVER project*
  - *with special thanks to X. Artigas and L. Torres for the collaboration on punctured and overlapped QAC*