Lecture 14

Note Title 2/27/2012

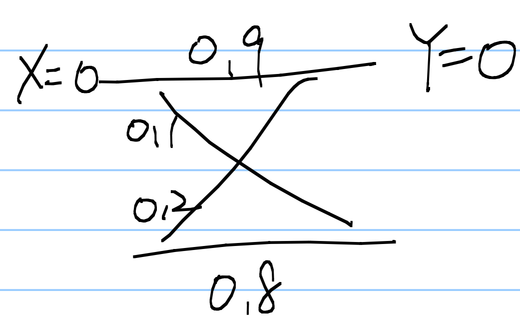
Let us look at two more examples of the FG.

Example 5: linear code: $H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$

$$\vec{y}_{obs} = (1, 1, 1, 0)$$

Each codeword is equally likely
(Or when considering the ML instead of the MAP decoder)

Channel



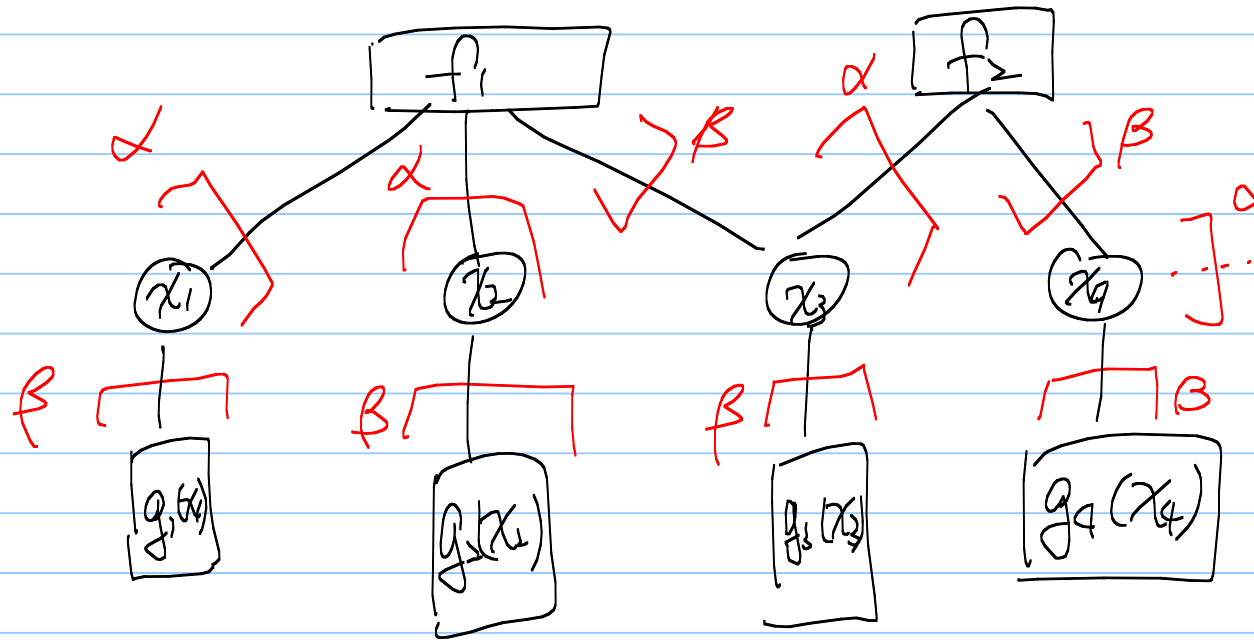
Q: Use the FG to find the bit-based ML decoder of x_3

A: $$\underset{x \in \{0,1\}}{\operatorname{argmax}} \sum_{\substack{\vec{x}|_i = x \\ H\vec{x} = 0}} \prod_{i=1}^{4} P_{Y_i|X_i}(y_i|x_i)$$

$$= \underset{x \in \{0,1\}}{\operatorname{argmax}} \sum_{\vec{x}|_i = x} f_1(x_1, x_2, x_3)\, f_2(x_2, x_3) \prod_{i=1}^{4} P_{Y_i|X_i}(y_i|x_i)$$

where $f_1(x_1, x_2, x_3) = \begin{cases} 1 & \text{if } x_1 + x_2 + x_3 = 0 \\ 0 & \text{otherwise} \end{cases}$

$$f_2(x_3, x_4) = \begin{cases} 1 & \text{if } x_2 + x_4 = 0 \\ 0 & \text{otherwise} \end{cases}$$



where $g_i(\cdot) = P_{Y_i|X_i}(y_i|\cdot)$

Example

$$\alpha_{x_3, f_2}(x_3) = \beta_{f_1, x_3}(x_3) \cdot \beta_{g_3, x_3}(x_3)$$

$$\beta_{f_1, x_3}(x_3) = \sum_{x_1, x_2} \alpha_{x_1, f_1}(x_1) \cdot f(x_1, x_2, x_3) \cdot \alpha_{x_2, f_1}(x_2)$$

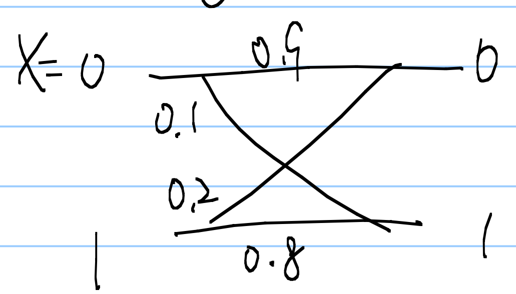
The factor-graph approach is general for different problems.

Example: Continue from the previous example

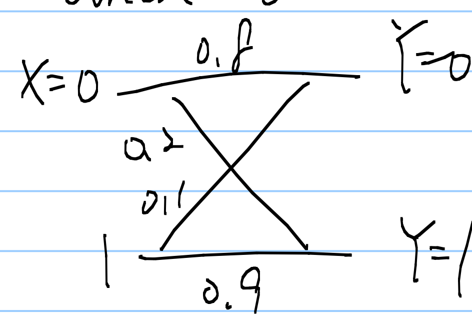
with $H = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}$

but the channel has different states θ (timing-recovery, multipath-fading, change of the ambient light

When $\theta = 0$



When $\theta = 1$



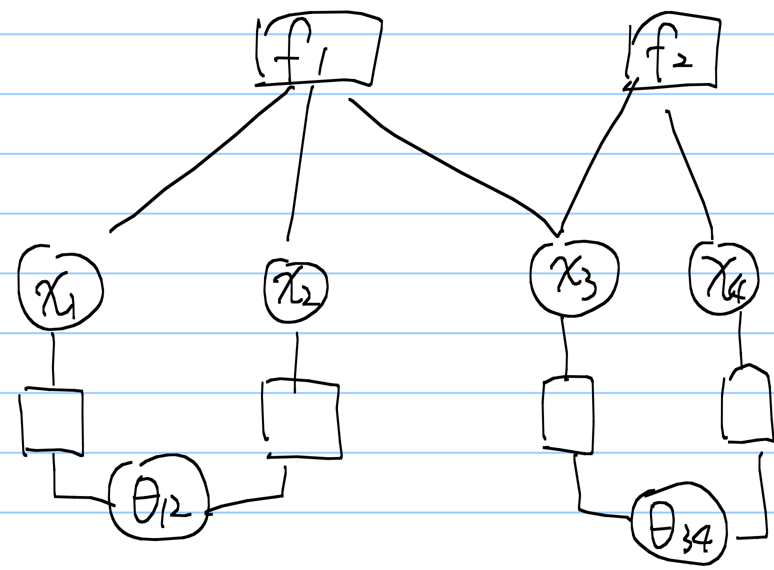
Θ is unknown but changes independently every 2 time slots. We have $\Theta_{1,2}$ governs $t=1,2$

$\Theta_{3,4}$ .... $t=3,4$

Q: Suppose the observed $\vec{y} = 1110$. Find the ML joint channel estimation & ECC decoder $(\Theta_{12}, \Theta_{3,4}, x_1, x_2, x_3, x_4)^*$

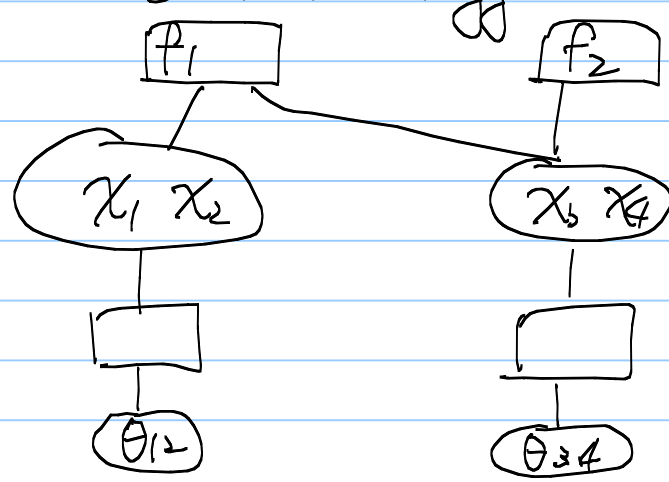
Ans: The objective function is

$$f_1(x_1,x_2,x_3)\, f_2(x_3,x_4) \times P_{Y_1|X_1,\Theta}(y_1|x_1,\theta_{12})$$
$$\times P_{Y_2|X_2,\Theta}(y_2|x_2,\theta_{1,2})$$
$$\times P_{Y_3|X_3,\Theta}(y_3|x_3,\theta_{34})$$
$$\times P_{Y_4|X_4,\Theta}(y_4|x_4,\theta_{3,4})$$



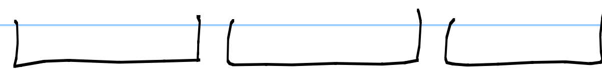
Not cycle-free

We can consider bigger states, which lead to an acyclic FG.



We can then proceed with a VA or a BCJR algorithm

Example: A CDMA detection

Three slots, 

shared by three users.

User 1: Send X_1, X_2, X_3 from $\{-1,1\}^3$

Instead on sending 8 different combinations he/she sends X_1, X_2, X_3 satisfying

$$X_1 \cdot X_2 \cdot X_3 = 1.$$

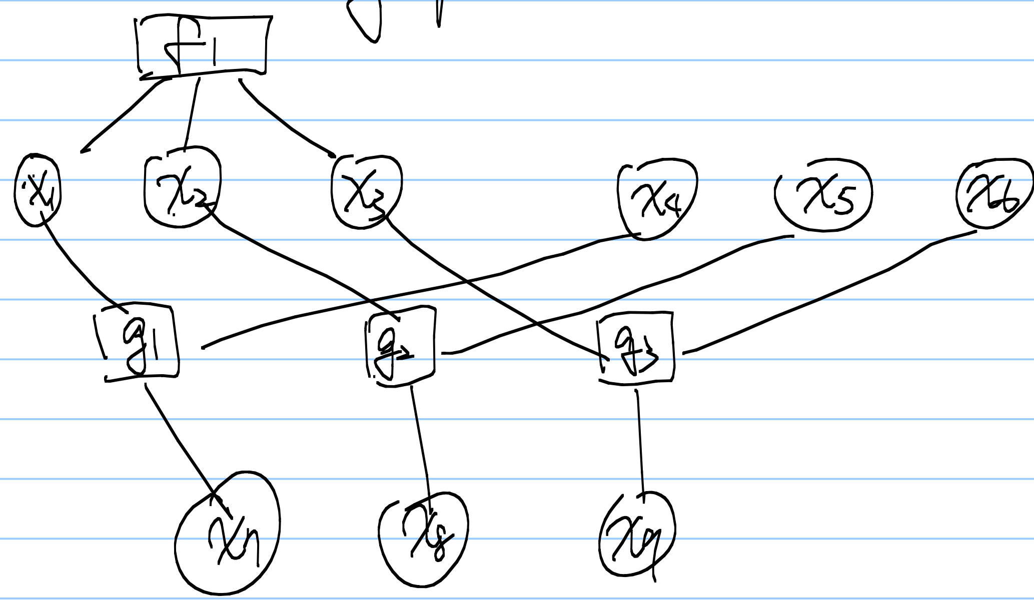
User 2: Send X_4, X_5, X_6 freely from $\{-1,1\}^3$

User 3: Send $X_7\ X_8\ X_9$ freely from $\{-1,1\}^3$

$$Y_1 = X_1 + X_4 + X_7 + N_1$$
$$Y_2 = X_2 + X_5 + X_8 + N_2$$
$$Y_3 = X_3 + X_6 + X_9 + N_3$$

Q: decide the ML bit value of X_1

A: The factor graph is

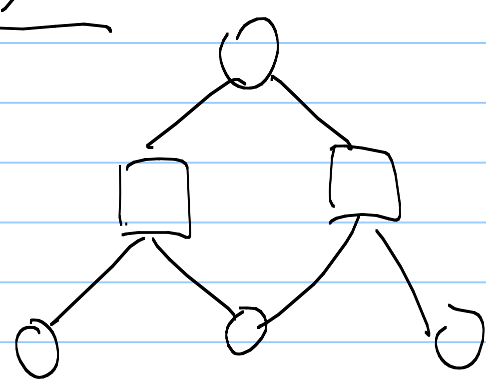


g_1, g_2, g_3 are decided by the observation y_1, y_2, y_3 & the noise model n_1, n_2, n_3

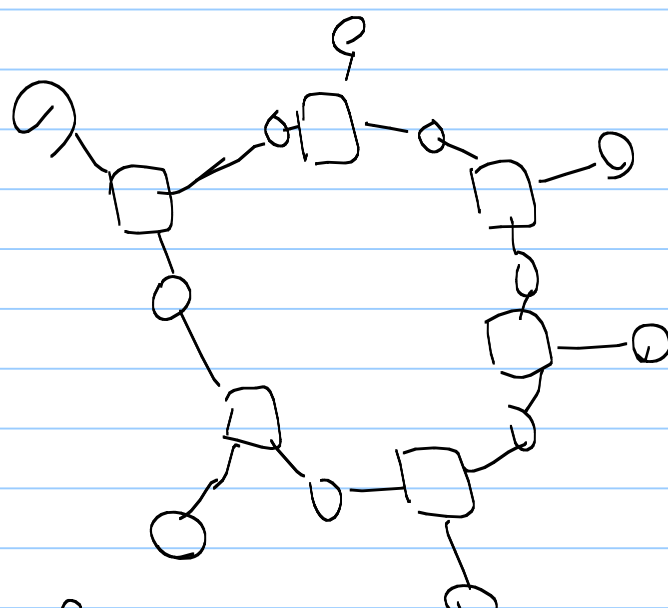
$\Rightarrow$ X_1^* can be obtained by a BCJR algorithm.

$\boxed{\text{Acyclicness}}$ is critical to the optimality of the min-sum (VA-like) & the sum-product (BCJR-like) decoders. However, empirically the min-sum & sum-product algorithms are good suboptimal decoders when the underlying factor graph is free of <u>short cycles</u>

Ex:



min-sum is bad.

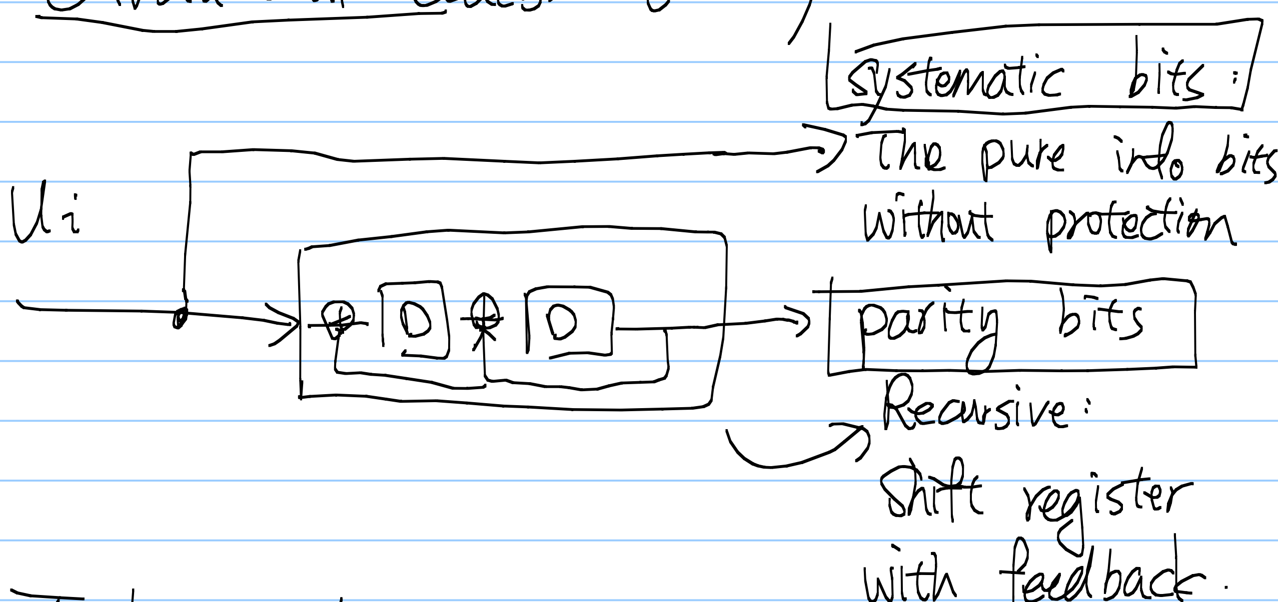


min-sum is empirically good.

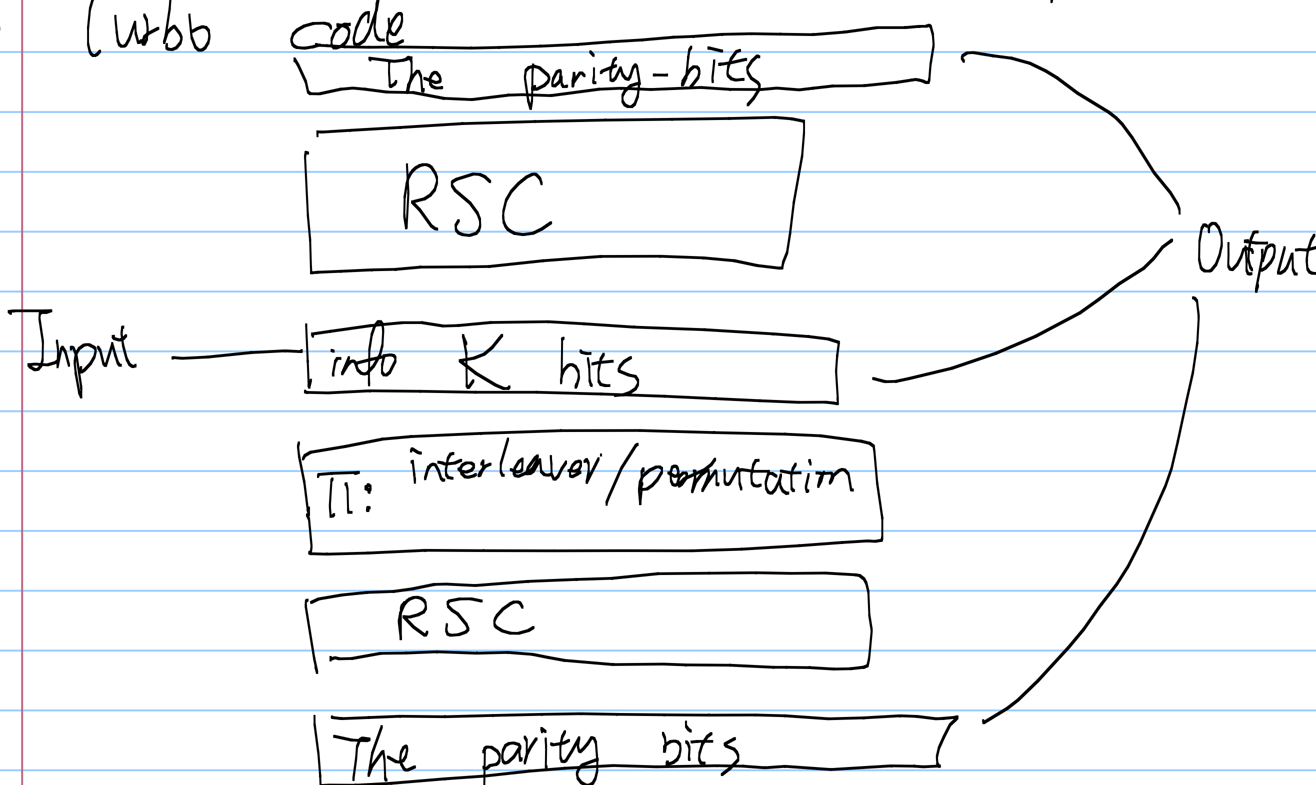
For cyclic factor graphs, the iterative formulas are decoupled from its physical meaning as summaries

* Turbo codes (Berrou-Glavieux 96)
  the first code that has performance close to the Shannon capacity

* We first define Recursive Systematic Convolutional codes. (RSC)



* Turbo code



If the two RSC codes have rate R_1 & R_2

then K info bits $\nearrow K \times \frac{1-R_1}{R_1}$ parity bits

$\searrow K \times \frac{1-R_2}{R_2}$ parity bits

the total rate is

$$\frac{K}{K\left(\left(\frac{1-R_1}{R_1}\right)+\left(\frac{1-R_2}{R_2}\right)+1\right)}$$

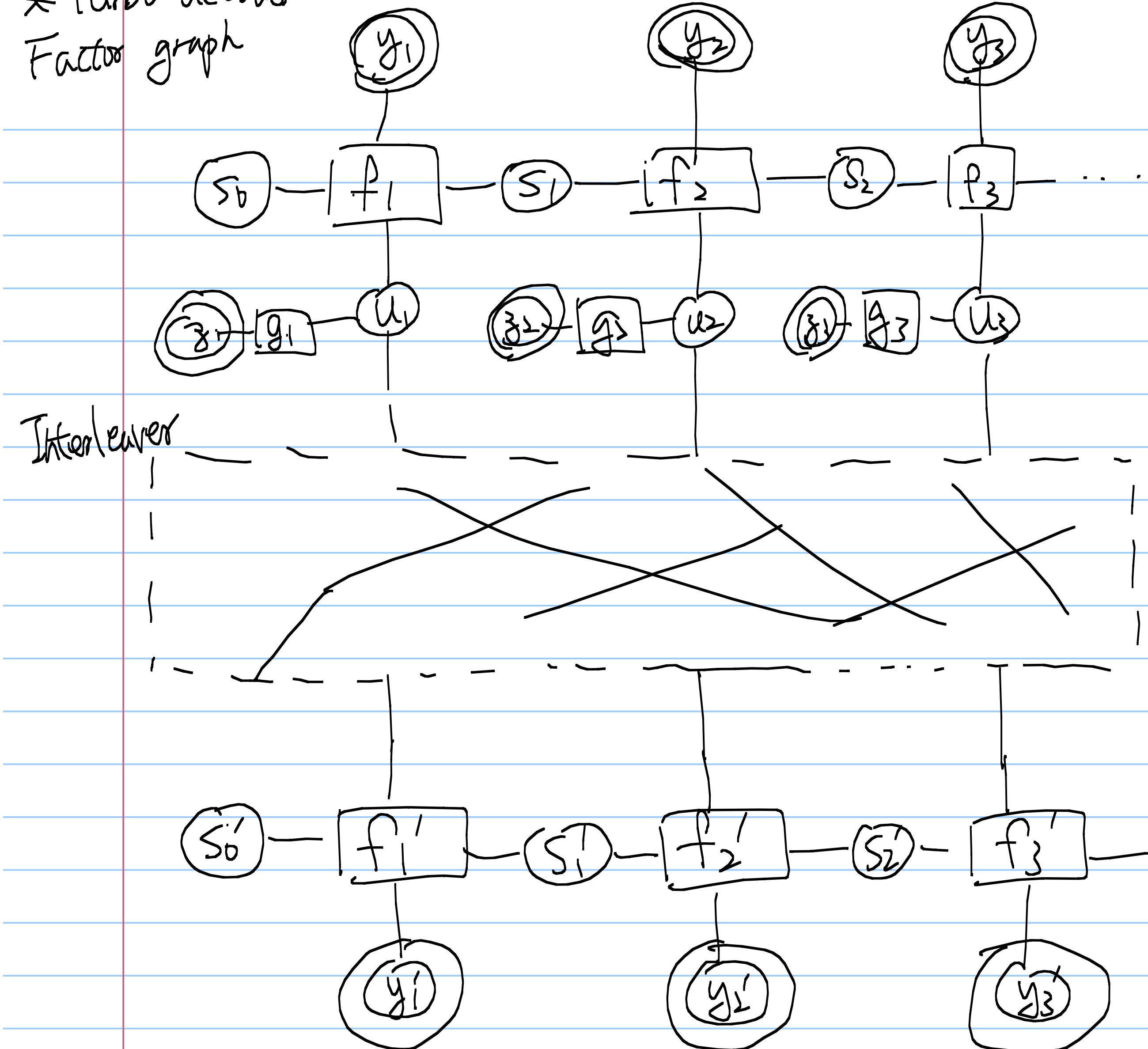
$$= \frac{R_1 R_2}{R_1+R_2-R_1R_2}$$

---

* How to decode a turbo code?

* The factor graph representation

* Turbo decoder:
Factor graph

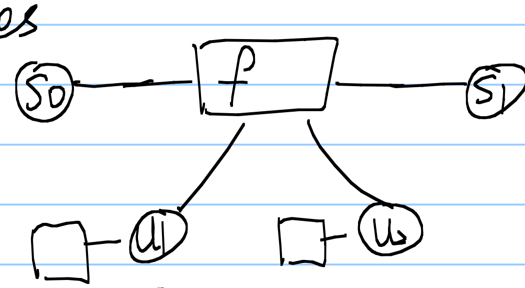


① The chain-based factor graph for RSC 1.

② Single out the impact of u.

(If each trellis is decided by 2 bits.

then it becomes



③ The factor functions $f_1, f_2, f_1, g_1, g_2, g_3$

are initialized by the y bits: the noisy observation of the parity bits

& by the z bits: the noisy observation of the systematic bits

4. The u bits are passed through an interleaver & then encoded by RSC_2

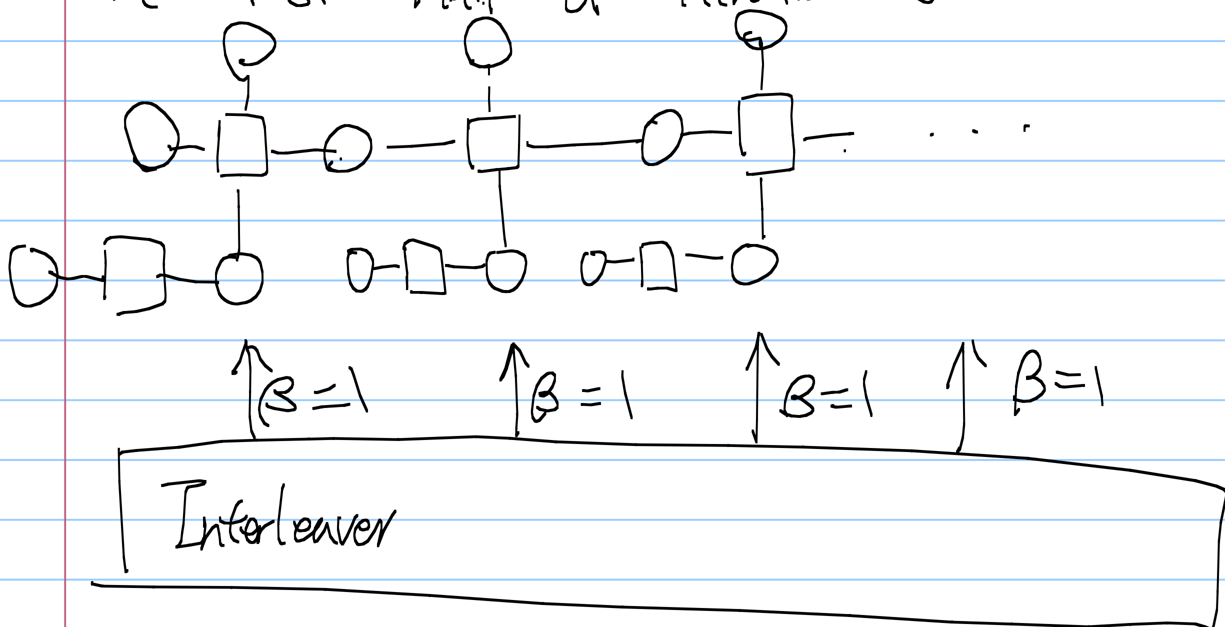
5. With a good interleaver, there will be no short cycles

We can run the BCJR assuming it is cycle free.

Turbo decoding:

① Each iteration contains two parts

② The first half of iteration 1.



Since no information comes from the interleaver all the $\beta_{j,i}(\cdot)$ are initialized to one.

③ We can run the forward & backward iteration of the BCJR decoder.

④ The question is then how to combine the results of the first half (RSC_1) with that of RSC_2