LDPC CODES [Nell'ambito del corso "Comunicazioni Digitali" tenuto da prof. Colavolpe]

ANDREA MODENINI, UNIVERSITÀ DI PARMA, DICEMBRE 2011

STORIA: 1963 → PhD tesi di Robert Gallager. Tuttavia le idee illustrate da Gallager richiedevano elevata potenza di calcolo, rispetto a quella disponibile allora. I codici realizzati erano soltanto alcune decine di bit.

1993 → Nascono i "turbo codes". Hackag, ed in contemporanea Richardson & Urbanke riscoprono i LDPC codes, dimostrando che possono ottenere le stesse prestazioni del TURBO, se non migliori.

Oggi → Usati nelle applicazioni che richiedono elevata efficienza spettrale (es. DVB-S2). Preferiti soprattutto per l'assenza di PATENTS.

RIPASSO: codici a blocco

Supponiamo di voler trasmettere $\underline{m} = [1\,0\,1\,1\,0]^T$. Posso aggiungere il bit di parità $p = 1$, e trasmettere $\underline{c} = [\underline{m}^T, p]^T$.

In generale posso avere q bit di parità, che coinvolgono diversi bit del messaggio:

$$p_i = \sum_{j \in K_i} m_j$$

$K_i \triangleq$ Insieme degli indici dei m_j coinvolti nell'i-esimo check di parità.

È possibile usare forma matriciale

$$\underline{c} = \underline{\underline{G}}\,\underline{m} \qquad \text{con} \qquad \underline{\underline{G}} = \begin{pmatrix} I_{k,k} \\ \hline \underline{\underline{P}} \end{pmatrix} \begin{matrix} \}k \\ \}q \end{matrix} \Big\} n$$

se k è il n° di bit di messaggio, e q di parità, avremo parole di codice lunghe $n \triangleq k+q$, ovvero un codice $\mathcal{C}$ con rate $r \triangleq \frac{k}{n}$

Def Una matrice $\underline{\underline{H}}$ $q \times n$ è detta di parità per il codice $\mathcal{C}$ se

$$\underline{\underline{H}}\,\underline{c} = \underline{0} \iff \underline{c} \text{ parola di codice}$$

Fatto La matrice $\underline{\underline{H}} = (\underline{\underline{P}} \mid I_{q,q})$ è una matrice di parità.

Proof

$$\underline{\underline{H}}\,\underline{c} = \underline{\underline{H}}\,\underline{\underline{G}}\,\underline{m} = \left(\underline{\underline{P}}\, I_{k,k} + I_{q,q}\,\underline{\underline{P}}\right)\underline{m} = \underline{\underline{0}}\,\underline{m} = \underline{0}$$

L'elemento nella riga i, e colonna j di $\underline{\underline{H}}$ lo indichiamo h_{ij}.

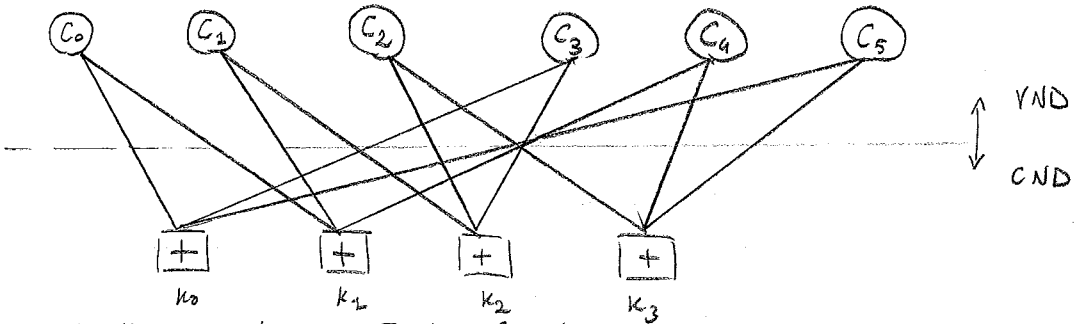
Low Density Parity Check

Sono codici con $\underline{\underline{H}}$ sparsa : il numero di 1 per riga e colonna è $\ll$ rispetto alla block length n

Def: Un codice LDPC si dice **REGOLARE** se per ogni riga ha esattamente d_c "1", e d_v "1" per colonna. In tal caso indicheremo il codice con (d_v, d_c).

Grafo di Tanner Grafo bipartito composto da CHECK NODE (CND) e VARIABLE NODE (VND). Si costruisce dalla $\underline{\underline{H}}$ col seguente metodo:

1. Ogni colonna è un VND. (c_i)
2. Ogni riga è un CND. (k_i)
3. Un arco (non direzionale) collega k_i e $c_j \iff h_{ij} = 1$.



Fatto il grafo di Tanner è un Factor Graph. (FG)

CANALE AWGN $y_i = c_i + w_i$

$$p(\underline{c} \mid \underline{y}) \propto p(\underline{y} \mid \underline{c})\, \chi_{\mathcal{C}}(\underline{c}) = \left[\prod_i p(y_i \mid c_i)\right] \chi_{\mathcal{C}}(\underline{c}) \quad (A)$$

$$\chi_{\mathcal{C}}(\underline{c}) = \begin{cases} 1 & \text{se } \underline{c} \text{ è parola di codice} \\ 0 & \text{else} \end{cases}$$

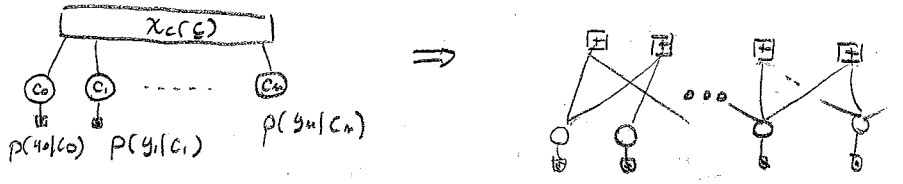
La $\chi_{\mathcal{C}}(\underline{c})$ può essere fattorizzata come segue.

Definiamo

- K - insieme degli indici dei CND $= \{0, 1, \dots, q-1\}$
- V - insieme degli indici dei VND $= \{0, 1, \dots, n-1\}$
- K_i - insieme degli indici dei VND coinvolti in k_i
- V_i - insieme degli indici dei CND coinvolti in c_i

$$\chi_{\mathcal{C}}(\underline{c}) = \prod_{i \in K} \left(\left(\sum_{j \in K_i} c_j \right) + 1 \right)$$ con somme e prodotti in $GF(2)$

Da (A) il FG è

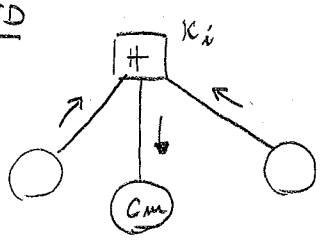


DECODING

La decodifica può essere fatta usando il SUM PRODUCT sul grafo di Tanner per il codice LDPC. Dato che il grafo ha cicli non c'è ottimo, ma solo diversi algoritmi di scheduling.

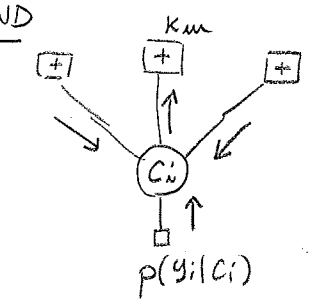
- REGOLE AGGIORNAMENTO (UR)

CND



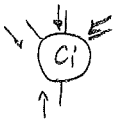
$$\Delta_{i\to m}(c_m) = \sum_{\sim\{c_m\}} \left[\prod_{\substack{j\in K_i \\ j\neq m}} \Lambda_{j\to i}(c_j) \right] \left[\sum_{j\in K_i} c_j + 1 \right]$$

VND



$$\Lambda_{i\to m}(c_i) = \prod_{\substack{j\in V_i \\ j\neq m}} \left[\Delta_{j\to i}(c_i) \right] p(y_i|c_i)$$

- DECISIONE



$$p(c_i|y_i) \approx \prod_{j\in V_i} \left[\Delta_{j\to i}(c_i) \right] p(y_i|c_i)$$

- SCHEDULING: Esistono diversi tipi di scheduling. I più usati sono FLOODING e WAVE.

DOMINIO LOGARITMICO Sebbene a livello teorico le info sul grafo sono funzioni di c_i in scala lineare, nella pratica si usano le scale logaritmiche con info scalari

LLR: $h_v\left(\frac{p_0}{p_1}\right)$ (LLR - log Likelihood Ratio)

Oss La UR ai VND con LLR diventa:

$$\| LLR_{i\to m} = \sum_{\substack{j\in V_i \\ j\neq m}} LLR_{j\to i} + LLR_{ch\,i}$$

dove

$$LLR_{i\to m} = \log\left(\frac{\Lambda_{i\to m}(0)}{\Lambda_{i\to m}(0)}\right), \quad LLR_{j\to i} = \log\left(\frac{\Delta_{j\to i}(0)}{\Delta_{j\to i}(1)}\right)$$

e $LLR_{ch_i} = \log\left(\frac{p(y_i|c_i=0)}{p(y_i|c_i=1)}\right)$

Invece ai CND

(B) $$\| LLR_{i\to m} = 2\tanh^{-1}\left(\prod_{\substack{j\in C_i \\ j\neq m}} \tanh\left(\frac{LLR_{j\to i}}{2}\right) \right)$$

Dimostrazioni: in classe.

Un' alternativa è il dominio SLLD definito come

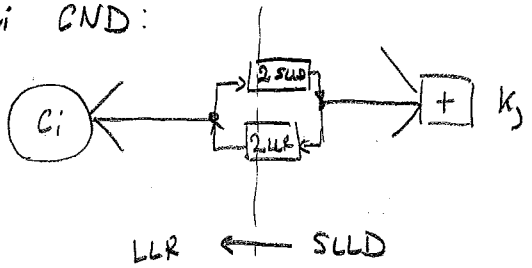
$$sgn(p_1 - p_0)\, ln|p_1 - p_0|$$

(SLLD - signed Log Likelihood Difference)

La UR ai CND diventa

$$SLLD_{i\to m} = \left[\prod_{\substack{j\in K_i \\ j\neq m}} sgn(SLLD_{j\to i})\right]\left[\sum_{\substack{j\in ch_i \\ j\neq m}} |SLLD_{j\to i}|\right]$$

Un' idea molto usata è di passare al dominio LLR ai VND e al SLLD ai CND:



La funzione di conversione si può dimostrare che è la stessa.

$$f_{LLR\to SLLD}(x) = f_{SLLD\to LLR}(x) = sgn\left(\frac{1-e^x}{1+e^x}\right) ln\left|\frac{1-e^x}{1+e^x}\right|$$

$$= -sgn(x)\, ln\, tanh\left|\frac{x}{2}\right|$$

ALGORITMO DECODING (FLOODING)

1. Calcola LLR_{ch_i} e poni $LLR_{i\to m} = LLR_{ch_i}$ $\forall m$, e $IT = 0$
2. Trasforma $LLR_{i\to m}$ in $SLLD_{i\to m}$ $\forall m \in V$, $\forall m \in V_i$
3. Usa UR dei CND
4. trasforma $SLLD_{m\to i}$ in $LLR_{m\to i}$ $\forall m \in C$, $\forall i \in C_m$
5. Fai Decisione. Se soddisfa sindrome $\Rightarrow$ STOP (DECODIFICA RIUSCITA)
   Altrimenti $IT = IT + 1$. Se $IT == IT_{MAX} \Rightarrow$ STOP (FALLITA)
6. Usa UR LLR

NOTA Sebbene abbiamo fatto per AWGN, si può generalizzare a altri casi. ES| ISI CHANNEL BCJR + LDPC

Def Un codice LDPC si dice irregolare se non è regolare.

I codici LDPC irregolari sono definiti da una degree distribution ovvero la distribuzione per ogni d_v e d_c

ES| $d_v = \begin{cases} 2 & \text{per } 30\% \text{ VND} \\ 3 & \text{per } 65\% \\ 7 & \text{per } 5\% \end{cases}$ $\quad d_c = \{ 7 \text{ per } 100\%$ $\quad r \simeq 0.586$

OSS Non tutte le d_v, d_c, r sono possibili. Deve valere

$$\bar{d_v} = (1-r)\bar{d_c}$$

$\bar{d_v}, \bar{d_c}$ - valore medi dei gradi

Formalmente, dette a_i^v la frazione di VND con d_{v_i} rami e a_i^c la frazione dei CND con d_{c_i} rami

$$\bar{d}_v = \sum_i a_i^v d_{v_i} \qquad \bar{d}_c = \sum_i a_i^c d_{c_i}$$

con $\sum_i a_i^v = \sum_i a_i^c = 1$.

Def: la frazione di rami b_i^v, b_i^c con grado d_{v_i} e d_{c_i} è ottenuta come

$$b_i^v = \frac{n a_i^v d_{v_i}}{n \bar{d}_v} \qquad b_i^c = \frac{n a_i^c d_{c_i}}{n \bar{d}_c}$$

Un codice irregolare viene spesso definito da due polinomi

$$\tilde{d}_v(x) = \sum_i b_i^v x^{d_{v_i}-1} \qquad \tilde{d}_c(x) = \sum_i b_i^c x^{d_{c_i}-1}$$

ES Provare a dimostrare che

$$r = 1 - \frac{\int_0^1 \tilde{d}_c(x)\,dx}{\int_0^1 \tilde{d}_v(x)\,dx}$$

PROGETTAZIONE LDPC → EXIT CHART
→ DENSITY EVOLUTION

Dato un certo sistema, come si può decidere la degree distribution?

EXIT CHART (EXTRINSIC INFORMATION TRANSFER CHART)

OSS su canale BAWGN (B- Binary) per bit 0 trasmetto 1, e per 1 trasmetto -1.

$$\begin{cases} c_i = 0 \Rightarrow s_i = 1 \\ c_i = 1 \Rightarrow s_i = -1 \end{cases} \Rightarrow s_i = 1 - 2c_i$$

e ricevo $\boxed{y_i = s_i + w_i}$ $\qquad w_i \sim N(0, N_0)$

la LLR_i è

$$LLR_i = \log\left(\frac{p(y_i|c_i=0)}{p(y_i|c_i=1)}\right) = \log\left(\frac{e^{-\frac{(y_i-1)^2}{2N_0}}}{e^{-\frac{(y_i+1)^2}{2N_0}}}\right) = \ln\left(e^{+\frac{2y_i}{N_0}}\right)$$

$$= \left(\frac{2}{N_0}\right) y_i = \left(\frac{2}{N_0}\right)(s_i + w_i)$$

$\frac{2}{N_0} w_i = \hat{n}_i$ è gaussiano con $\mathrm{Var}(\hat{n}_i) = \frac{4}{N_0^2} N_0 = \frac{4}{N_0}$

ma $\left(\frac{E_b}{N_0}\right)\frac{k}{n} = \frac{E_s}{N_0} = \frac{1/2}{N_0}$ per cui $\boxed{\sigma_n^2 \triangleq \mathrm{Var}(\hat{n}_i) = 8\left(\frac{E_b}{N_0}\right) r}$

e otteniamo

$$\boxed{LLR_i = \frac{\sigma_n^2}{2} s_i + \hat{n}_i} \quad \text{con} \quad \hat{n}_i \sim N\left(0, \sigma_n^2\right)$$

LE LLR SONO GAUSSIANE CONDIZIONATAMENTE A s_i!

più σ_n^2 è alto, più LLR è affidabile

L'informazione mutua tra c_i e LLR_i

$$I(c_i; LLR_i) \underset{\circledast}{=} E\left\{ \log_2 \left(\frac{p(LLR_i|s_i)}{\sum_{s_i} p(LLR_i|s_i)\frac{1}{2}} \right) \right\} =$$

$$= E\left\{ \log_2 \left(\frac{2\, p(LLR_i|s_i)}{p(LLR_i|s_i=-1) + p(LLR_i|s_i=1)} \right) \right\}$$

NOTA! $\circledast$
$I(c_i; LLR_i) = I(s_i; LLR_i)$
perché da c_i a s_i è senza loss

ma $p(LLR_i|s_i) \propto e^{\frac{-\left(LLR_i - \frac{\sigma_n^2}{2} s_i\right)^2}{2\sigma_n^2}}$

$$e \quad \frac{p(LLR_i|s_i)}{p(LLR_i|-s_i)} = e^{-2\frac{\sigma_n^2 LLR_i s_i}{2\sigma_n^2}} = e^{-LLR_i s_i}$$

per cui

$$I(c_i; LLR_i) = 1 + E\left\{ \log_2 \left(\frac{1}{1 + e^{-LLR_i s_i}} \right) \right\} =$$

$$= 1 - E\left\{ \log_2 \left(1 + e^{-LLR_i s_i} \right) \right\}$$

(VERIFICARE X CASA) ↘ X SIMMETRIA

$$= 1 - E\left\{ \log_2 \left(1 + e^{-LLR_i} \right) \,\middle|\, s_i = 1 \right\}$$

$$= 1 - \int_{-\infty}^{+\infty} \log_2 \left(1 + e^{-LLR_i} \right) \frac{e^{-\frac{\left(LLR_i - \frac{\sigma_n^2}{2}\right)^2}{2\sigma_n^2}}}{\sqrt{2\pi\sigma_n^2}} \, d LLR_i$$

$$= J(\sigma_n)$$

La $I(c_i; LLR_i)$ è funzione della $\sigma_n^2 = 8\left(\frac{E_b}{N_0}\right) R$ ed è invertibile come $J^{-1}\left(I(c_i; LLR_i)\right) = \sigma_n$.

Proviamo a ipotizzare che tutte le LLR durante il decoding siano indipendenti e condizionatamente gaussiane a s_i.

Ai VND (PER CASO REGOLARE)

$$LLR_{i \to m} = \sum_{\substack{j \in V_i \\ j \neq m}} LLR_{j \to i} + LLR_{chi}$$

(σ_A^2 sopra la somma; σ_n^2 sopra LLR_{chi})

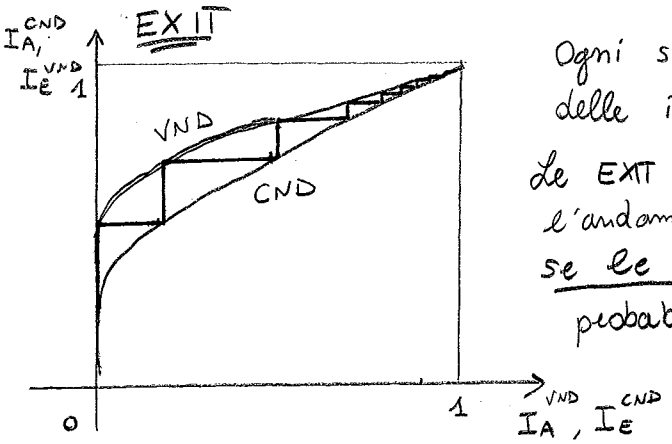
$$I_E^{VND} = I(LLR_{i \to m}, c_i) = J\left(\sqrt{(d_v - 1)\sigma_A^2 + \sigma_n^2} \right), \text{ dove } \sigma_A = J^{-1}(I_A)$$

e $I_A = I(LLR_{j \to i}, c_i)$. Allora si ottiene:

$$\boxed{I_E^{VND}\left(I_A^{VND}, \frac{E_b}{N_0}, R\right) = J\left(\sqrt{(d_v - 1)\left(J^{-1}\left(I_A^{VND}\right)\right)^2 + \left(8 \frac{E_b}{N_0} R\right)} \right)}$$

Per i CND sotto certe ipotesi e semplificazioni si può dimostrare che

$$\boxed{I_E^{CND}\left(I_A^{CND}\right) \approx 1 - J\left(\sqrt{d_c - 1}\; J^{-1}\left(1 - I_A^{CND}\right) \right)}$$



Ogni scalino rappresenta l'andamento teorico delle iterazioni di decoding.

Le EXIT CHART sono quindi grafici che mostrano l'andamento teorico delle info estrinseche.

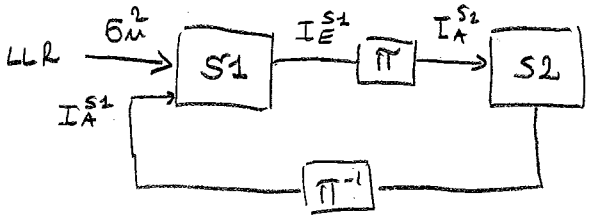
Se le CURVE si intrecciano (TUNNEL CHIUSO) molto probabilmente la decodifica fallirà

Per codici irregolari, si calcola l'informazione mutua mediata sui rami.

$$I_E^{VND} = \sum_i b_i^v \; J\left(\sqrt{(d_{v,i}-1)\left(J^{-1}\left(I_A^{VND}\right)\right)^2 + 8R\frac{E_b}{N_0}}\right)$$

$$I_{E,\dots}^{CND} = 1 - \sum_i b_i^k \; J\left(\sqrt{(d_{c,i}-1)}\; J^{-1}\left(1 - I_A^{CND}\right)\right)$$

Sebbene le abbiamo studiate per LDPC le EXIT possono essere generalizzate a molti sistemi iterativi nella forma:

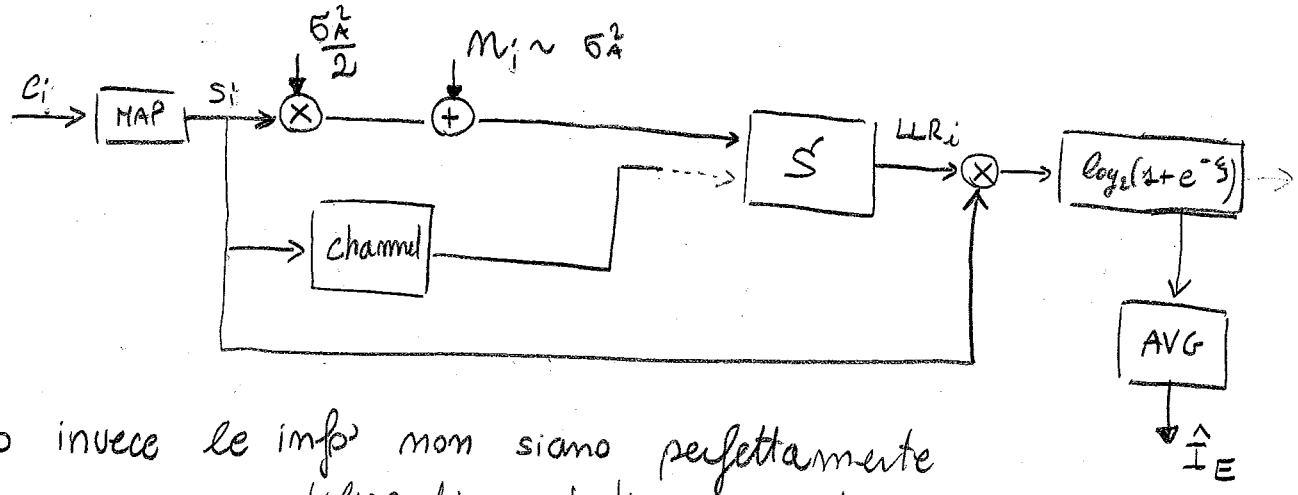


Dove gli interleave Π, Π^{-1} servono per far valere ipotesi indipendenza!

Nel caso LDPC $S1 \equiv VND$ e $S2 \equiv CND$.

Un metodo per misurare la EXIT è il montecarlo.

$$\hat{I}_E) = 1 - \frac{1}{N}\sum_{i=1}^{N} \log_2\left(1 + e^{-LLR_i^{(j)} s_i^{(j)}}\right)$$



Nel caso invece le info non siano perfettamente gaussiane vengono utilizzati metodi numerici per calcolare l'integrale con $\hat{p}(LLR_i|c_i)$ stimata con montecarlo

$$I_E = E\left\{\log_2\left(\frac{\hat{p}(LLR_i|c_i)}{\hat{p}(LLR_i|c_i=0)\frac{1}{2} + \hat{p}(LLR_i|c_i=1)\frac{1}{2}}\right)\right\} =$$

$$= \sum_{c_i}\int_{-\infty}^{+\infty} \hat{p}(LLR_i|c_i)\,p(c_i)\,\log_2\left(\frac{2\hat{p}(LLR_i|c_i)}{\hat{p}(LLR_i|c_i=0) + \hat{p}(LLR_i|c_i=1)}\right) dLLR_i$$

DENSITY EVOLUTION (DE)

Supponiamo di avere un canale con ERASURE (BEC) ed utilizzare codice LDPC. Alcuni bit saranno perfettamente noti, altri invece saranno cancellati.
Usando SP un CND dichiara il bit noto se su tutti gli altri rami ha bit noti, altrimenti dà erasure.
Un VND dichiara il bit noto nel momento in cui riceve un bit noto su un qualsiasi ramo.
Supponiamo che su n bit ricevuti dal canale, una frazione p_0 sono erasure (p_0 - prob. erasure del canale)
Sotto ipotesi di indipenza la probabilità che un CND emetta su un ramo un erasure all'iterazione ℓ

$$q_e^{(i)} = 1 - Pr\{\text{bit noto}\} = 1 - (1-p_{e-1})^{d_{c_i}-1}$$

dove p_{e-1} è la frazione di erasure delle info che arrivano dai VND.
Mediamolo

$$q_e = \sum_i b_i^{\nu} q_e^{(i)} = 1 - \tilde{d}_c(1-p_{e-1}).$$

Nel caso dei VND, la prob. che si ha ERASURE è

$$p_e^{(i)} = p_0 q_e^{d_{v_i}-1} \Rightarrow p_e = p_0 \sum_i b_i^{v} q_e^{d_{v_i}-1} =$$

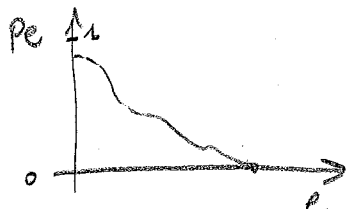
$$= p_0 \tilde{d}_v\left(1 - \tilde{d}_c(1-p_{e-1})\right)$$

ovvero

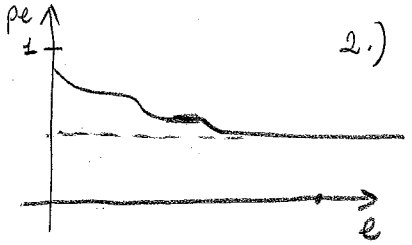
$$\boxed{p_e = p_0 \tilde{d}_v\left(1 - \tilde{d}_c(1-p_{e-1})\right)}$$

abbiamo una funzione di evoluzione iterativa. A partire da p_0 sappiamo $p_1, p_2, \dots, p_n$ ovvero l'andamento delle iterazioni.
Possono succedere due casi:

1.) DECODIFICA RIUSCITA



2.) DECODIFICA FALLITA



Inoltre tramite la DE si può ricavare la massima ERASURE

Def: $p_0^* = \sup\{0 \le p_0 \le 1 : p_e \to 0 \text{ per } \ell \to 0\}$

TEOREMA Sia $p_0 \in [0,1]$ e $f(x,y) = y\,\tilde{d}_v(1 - \tilde{d}_c(1-x))$.
se $x \neq f(x,p_0) \quad \forall x \in (0,p_0) \Rightarrow p_e \to 0$
se $\exists x$ t.c. $x = f(x,p_0) \Rightarrow p_e \ge x \ \forall \ell$

DIM $p_e = p_0 \sum_i b_i^{v}\left(1 - \sum_i b_i^{\nu}(1-p_{e-1})^{d_{c_i}-1}\right)^{d_{v_i}-1} \le p_{e-1}$

Infatti, se $x_0 < x_1 \Rightarrow f(x_0, y) < f(x_1, y)$ con $x_0, x_1 \in [0,1]$

Ma $p_1 = p_0 \underbrace{q e^{d_v - 1}}_{\leq 1} \leq p_0 \Rightarrow p_2 = f(p_1, p_0) \leq f(p_0, p_0) = p_1$

col iterativamente si ricava $p_e \leq p_{e-1}$.

Di conseguenza abbiamo delle p_e non crescenti!

Inoltre in quanto probabilità sono $p_e \geq 0$.

Esiste quindi un limite x.

Il limite deve essere tale $x = f(x, p_0)$. Di conseguenza

se $\nexists\, x \in (0, p_0)$ $p_e \to 0$ ☒

Al variare di p_0 posso trovare il massimo p_0 per cui

$x \neq f(x, p_0) \quad \forall\, x \in (0, p_0)$ ovvero p_0^*.

ALTRI CANALI La density evolution può essere fatta più in generale per altri canali lavorando sulle PDF dei LLR.

Si ottiene (con conti più complicati) che

$$p_e(x) = p_0(x) \otimes \tilde{d}_v\left(\Gamma^{-1}\left(\tilde{d}_c\left(\Gamma\left(p_{e-1}(x)\right)\right)\right)\right)$$

dove $p_0(x)$ - PDF iniziale ai VND

$p_e(x)$ - " l-iterazione "

$\tilde{d}_v(p(x)) = \sum_i b_i^v \left(p(x)\right)^{\otimes(d_{v_i}-1)}$

Γ, Γ^{-1} - funzione che calcola la PDF di una variabile SLLD aleatoria nata dalla trasformazione di una variabile aleatoria LLR (TEOREMA FONDAMENTALE DELLA PROBABILITÀ) vedi TDSA

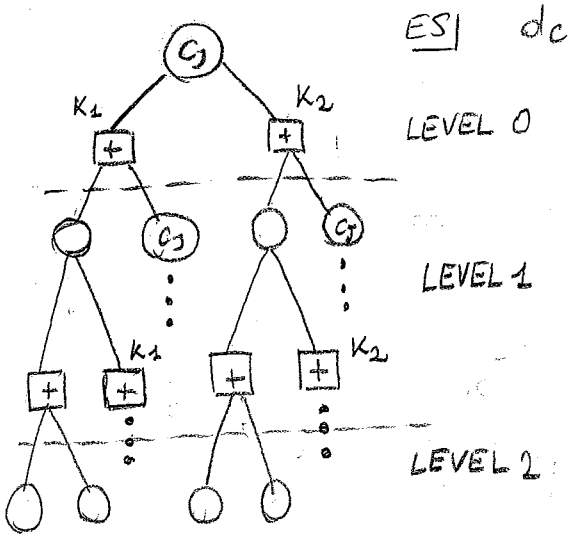
COSTRUZIONE CODICE - PROGRESSIVE EDGE GROWTH (PEG) (XIAO)

Data una degree distribution vogliamo costruire il codice LDPC realmente.

Il PEG è un algoritmo subottimo che tenta di costruire il grafo massimizzando il ciclo minimo.

E_{c_j} - archi collegati a c_j

Def L'insieme dei vicini $N_{c_j}^{\ell}$, sono tutti i CND raggiungibili da c_j a profondità ℓ



ES| $dc=2$ $dv=2$. Nota Alcuni rami sono ripetizioni di altri, e sono indicati con $\left(\vdots\right)$.

PASSO BASE

se devo piazzare un arco su c_j espando finchè $\bar{N}_{c_j}^{e} \neq \emptyset$ e $\bar{N}_{c_j}^{e+1} = \emptyset$ oppure $|N_{c_j}^{e}|$ non cresce più. Scelgo a questo punto un CND e $\bar{N}_{c_j}^{e}$

In questo modo se un CND è a profondità e, il ciclo è almeno $2(e+2)$

XIAO'S TRICK: se $|\bar{N}_{c_j}^{e}| > 1$ scelgo quello con n° minore di edges. In tal modo è più facile avere check con distribuzione "quasi" uniforme.

ALGORITMO PEG

```
for ( j=0; j<n; j++ ) {
    for ( k=0; k<dv,cj; k++ ) {
        if (k==0) {
            Ecj ← (cj, ki) dove ki ha il grado
                  più basso
        }
        else {
            espandi finchè |Ncj^e| ≠ ∅ e |Ncj^(e+1)| = ∅
            oppure |Nc_e^e| non cresce. Seleziona ki e N̄cj^e
            con più basso grado e Ecj ← (cj, ki)
        }
    }
}
```

CODIFICA

Nei codici LDPC è spesso impensabile costruirsi la matrice $\underline{\underline{G}}$.

es. DVB-S2 $r = 1/2$ $n = 64{,}800$ bit.

La $\underline{\underline{G}}$ ha densa la $\underline{\underline{P}}$ di $q \times k$ elementi avrei $\approx (32400)^2$ byte.

Per non parlare dei tempi di calcolo per ottenere $\underline{\underline{P}}$ da $\underline{\underline{H}}$!

Una soluzione è di imporre $\underline{\underline{H}}$ triangolare superiore nelle ultime q colonne (come usa DVB-S2).

In tal modo è possibile codificare con sostituzione all'indietro

$$\underline{\underline{H}}_c = \left(\text{SPARSA} \;\middle|\; \begin{array}{|c} \\ \\ \underline{\underline{0}} \end{array} \right) \begin{pmatrix} c_0 \\ \vdots \\ c_{k-1} \\ \hline c_k \\ \vdots \\ c_{n-1} \end{pmatrix} \begin{matrix} \} \text{ noti} \\ \\ \} \text{ incognite} \end{matrix}$$

ricavo prima c_{n-1} a partire da $c_0, \dots, c_{k-1}$. Poi $c_{n-2}, c_{n-3}, \dots$

Tuttavia questo metodo impone un vincolo sulla costruzione della $\underline{\underline{H}}$.

ALGORITMO DI RICHARDSON & URBANKE (RNU)

Algoritmo in due step. Uno di semitriangolarizzazione (OFF-LINE) ed uno di codifica (ON LINE). Complessità quasi proporzionale a n.

Lo step di codifica suppone di avere una matrice $\underline{\underline{H}}$ come

$$\underline{\underline{H}} = \left(\begin{array}{c|c|c} A & B & T \\ \hline C & D & E \end{array} \right) \begin{matrix} \} n-g \\ \} g \end{matrix} \qquad \boxed{\text{con } g \ll n}$$

(A: n colonne; B: g colonne; T, E: $n-g$ colonne)

Poniamo $\underline{c}^T = [\underline{m}^T, \underline{p}_1^T, \underline{p}_2^T]$ dove $\underline{m}$ il messaggio, $\underline{p}_1$ la prima parte di bit di parità lunga g, e $\underline{p}_2$ lunga $(n-g)$.

se $\underline{\underline{H}}\,\underline{c} = \underline{0} \implies \begin{pmatrix} I & 0 \\ \hline -ET^{-1} & I \end{pmatrix} \underline{\underline{H}}\,\underline{c} = \underline{0}$

$$\implies \left(\begin{array}{c|c|c} A & B & T \\ \hline -ET^{-1}A + C & \phi & 0 \end{array} \right) \begin{pmatrix} \underline{m} \\ \hline \underline{p}_1 \\ \hline \underline{p}_2 \end{pmatrix} = \underline{0} \qquad \text{con } \phi = -ET^{-1}B + C$$

$$\implies \begin{cases} \underline{p}_1 = -\phi^{-1}\left(-ET^{-1}A + C\right)\underline{m} \\ \underline{p}_2 = -T^{-1}\left(A\,\underline{m} + B\,\underline{p}_1\right) \end{cases}$$

Complessità:
i prodotti con A, B, C, E, T ha complessità $\mathcal{O}(n)$ dato che sono sparse.
Per ϕ^{-1} invece $\mathcal{O}(g^2)$.
$\implies \mathcal{O}(n + g^2)$

Ovviamente per fare lo step di codifica occorre prima avere $\underline{\underline{H}}$ nella forma indicata.
Per farlo si fa (off-line) lo step di semitriangolarizzazione con algoritmo AH.

ALG. AH sia α un parametro $\alpha \in (0,1)$ di progetto.

PASSO 0: Data $\underline{\underline{H}}$ dichiara con prob. $1-\alpha$ ciascuna colonna NOTA.
Poni le colonne note in testa e definisci A la sottomatrice di colonne non note.

$$\left(\begin{array}{c|c} (1-\alpha)n & \alpha n \\ & \underline{\underline{A}} \end{array} \right)$$

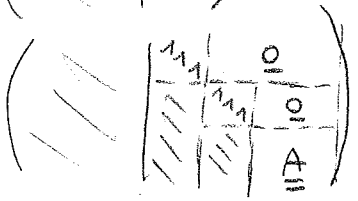
NOTA: nel disegno sono indicati valori medi.

PASSO 1: per ogni riga r_i in $\underline{\underline{A}}$ con peso 1 dichiara la colonna nota. Se non ci sono colonne note $\Rightarrow$ STOP.

PASSO 2: Porta colonne e righe note in testa. Cancella da A tutte le colonne e righe note e torna a PASSO 1.

Effetto ⓪ 

① - ② 

①b - ②b 

e così via...

Quando termina 2 casi

1)  è arrivato in fondo!

2)  bloccato a metà

In tal caso riordino colonne e gap sarà più grosso.

Come salgo α? Con la DE per BEC

Versione AHT: uguale AH ma lavora su trasposta.
In generale funziona meglio perché è più facile (di solito) trovare colonne unitarie che righe.

Esistono anche BH, BHT, CHT. L'unica cosa che cambia è il passo 0. Usano metodi più furbi per dichiarare note le colonne riducendo g ulteriormente con elevata probabilità.