UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea Triennale in Ingegneria Elettronica

Utilizzo di Algoritmi Information Set Decoding per stima di Distanza minima di codici LDPC

Use of Information Set Decoding Algorithms for Minimum Distance estimation of LDPC codes

Relatore:

Prof. Paolo Santini

Tesi di Laurea di:

Daniele Bartoloni

A.A. 2023 / 2024

UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea Triennale in Ingegneria Elettronica

Utilizzo di Algoritmi Information Set Decoding per stima di Distanza minima di codici LDPC

Use of Information Set Decoding Algorithms for Minimum Distance estimation of LDPC codes

Relatore:

Prof. Paolo Santini

Tesi di Laurea di:

Daniele Bartoloni

A.A. 2023 / 2024

Università Politecnica delle Marche

Facoltà di Ingegneria

Corso di Laurea Triennale in Ingegneria Elettronica

Via Brecce Bianche– 60131 Ancona (AN), Italia

Indice

Introduzione

L'elaborato che segue descrive il lavoro svolto durante il tirocinio presso il Dipartimento di Ingegneria dell'Informazione dell'Università Politecnica delle Marche.

L'obiettivo principale del progetto è stato lo studio, sia teorico che sperimentale, di una famiglia di codici lineari largamente utilizzata nel mondo delle telecomunicazioni: i codici LDPC (Low-Density Parity-Check). Questi codici trovano applicazione in molti campi per via delle loro ottime prestazioni, sia in termini di facilità di decodifica che di capacità correttiva.

In generale, ogni codice è caratterizzato da un parametro fondamentale noto come Distanza Minima. Questo parametro rappresenta il numero minimo di bit che differenziano due parole di codice valide e incide direttamente sulle prestazioni del codice, in particolare sulla sua capacità di rilevare e correggere errori. Pertanto, conoscere la Distanza Minima consente di valutare l'efficacia di un determinato codice.

Per la famiglia dei codici LDPC, tuttavia, non esiste attualmente un metodo che permetta di calcolare la Distanza Minima. L'unico approccio disponibile è di tipo empirico e consiste nell'implementare il codice, testare le sue capacità, e successivamente risalire alla Distanza Minima.

Contributo

Il lavoro si è articolato in una fase iniziale dedicata all'approfondimento teorico, che ha incluso lo studio dei concetti fondamentali della codifica di canale, con un focus specifico sui codici LDPC. La seconda fase invece, ha previsto lo studio sperimentale di codici LDPC, nello specifico, è stata studiata la Distanza Minima.

In modo più dettagliato il presente studio si propone di indagare come la Distanza Minima dei codici LDPC vari in funzione dei parametri utilizzati per generare gli stessi codici, attraverso un'analisi empirica. La sperimentazione è stata effettuata tramite l'esecuzione di un algoritmo ISD che permette di stimare la Distanza

Minima di codici precedentemente definiti. Al fine di effettuare uno studio deterministico (limitando quindi il fattore probabilistico), dato che i risultati ottenuti sono stimati, i test vengono effettuati su un numero di codici elevato in accordo con i limiti fisici.

Va sottolineato che l'obiettivo di questa ricerca non è quello di sviluppare una formalizzazione matematica che determini la distanza minima dei codici LDPC. Piuttosto, l'osservazione dei risultati sperimentali potrebbe rappresentare un punto di partenza per future analisi teoriche nonché uno strumento utile per studiare applicazioni materiali dei codici LDPC.

Anticipando alcuni risultati ottenuti si afferma che aumentando le grandezze definite nella generazione dei codici in modo lineare, sono state ottenute Distanze Minime crescenti seguendo il medesimo legame lineare. Il metodo utilizzato per effettuare lo studio è valido ed attendibile, il suo limite principale è rappresentato dai costi computazionali. Migliorando l'algoritmo di stima della Distanza Minima e utilizzando maggiore potenza di calcolo è possibile pensare di utilizzare il metodo per testare codici LDPC utilizzati in applicazioni concrete, passando quindi dallo studio prettamente teorico ad uno più tangibile.

Organizzazione lavoro

Il Capitolo 1 tratta la codifica di canale e nello specifico i codici LDPC con un approccio teorico, fornendo i concetti essenziali sull'argomento al fine di introdurre e comprendere la sperimentazione descritta nei capitoli successivi. I dettagli dell'analisi sperimentale ed i metodi utilizzati sono presentati nel Capitolo 2, mentre il Capitolo 3 riporta le conclusioni e le possibili implicazioni dei risultati ottenuti.

1 - Codifica di Canale

In questo capitolo vengono descritte le basi della codifica di canale.

1.1 Introduzione

Per comprendere il concetto di codifica di canale, è fondamentale avere prima una chiara comprensione dello schema generale di un sistema di comunicazione.

A tale scopo, in Figura 1 è riportato il modello di riferimento che sintetizza i principali elementi coinvolti nel processo di trasmissione e ricezione dell'informazione e, segnatamente:

- **Sorgente**: La sorgente *(source)* rappresenta l'elemento che genera l'informazione da trasmettere attraverso il canale. Nel presente studio si considera una sorgente binaria, ovvero una sorgente che produce sequenze di bit (simboli che possono assumere i valori 0 e 1).

- **Encoder**: L'*encoder* si occupa del processamento dei dati. Il suo compito è trasformare i dati in ingresso nella forma più adatta alla trasmissione. In questa fase, ai bit in ingresso all'*encoder* (in numero pari a k) vengono aggiunti ulteriori simboli di ridondanza ($n - k$). L'introduzione di questa ridondanza ha lo scopo di rendere la comunicazione più robusta, permettendo al sistema di rilevare e correggere eventuali errori che si verificano durante la trasmissione.

- **Canale di comunicazione**: Il canale *(channel)* è il mezzo fisico attraverso il quale l'informazione viene trasmessa. Durante la fase di trasmissione, il segnale inviato subisce inevitabilmente l'introduzione di disturbi. In alcuni casi si può assumere che il disturbo aggiunto non dipenda dalle caratteristiche dell'informazione trasmessa: in tali situazioni, il rumore viene definito come rumore additivo *(Additive Noise),* rappresentato dall'aggiunta, componente per componente, di un vettore di errore che si somma al vettore informazione (Figura 2). Il modello descritto comporta

che ogni singolo bit sia caratterizzato da una certa probabilità p di essere equivocato. Ipotizzando che la probabilità di errore sul bit sia costante indipendentemente dal valore effettivo del bit, si parla di Canale Binario Simmetrico *(Binary Simmetric Channel)*. In Figura 3 viene riportato il modello grafico per il canale descritto.

- **Decoder**: Il *decoder* ha il compito di interpretare il segnale ricevuto, restituendo l'informazione originaria generata dalla sorgente. In assenza di errori, o, qualora gli errori introdotti dal canale siano corretti, il *decoder* consente la ricostruzione esatta dell'informazione trasmessa.

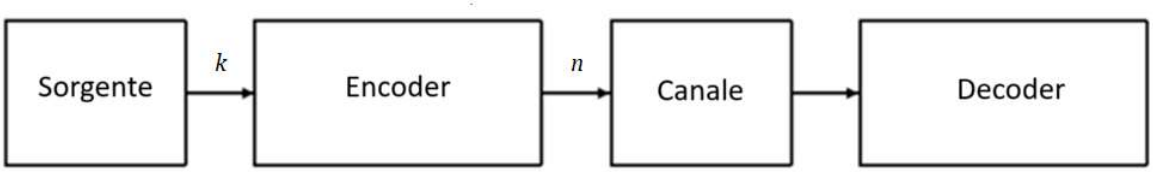

Figura 1: Modello del sistema di comunicazione

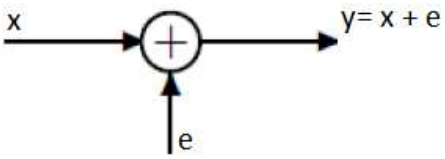

Figura 2: Modello del rumore additivo

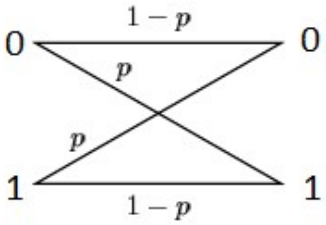

Figura 3: Modello del *Binary Simmetric Channel*

1.2 Codici a Blocco

Viene introdotto un concetto fondamentale partendo dal seguente enunciato.

Definizione: *Un codice a blocco (n, M) definito su un alfabeto finito F, è un sottoinsieme non vuoto C di F^n composto da M elementi.*

Il parametro n è chiamato lunghezza del codice, mentre la dimensione del codice è definita come $k = \log_{|F|} M$. Infine, Il *rate* del codice è pari a $R = k/n$ [1].

Considerando un alfabeto $\{F\} = \{x_1, x_2, x_3, \ldots . x_q\}$, dove $\{x_1, x_2, x_3, \ldots . x_q\}$ indicano le possibili parole di codice, un codice a blocco (n, M) rappresenta quindi una codifica in cui ogni messaggio è costituito da una sequenza di n simboli. L'insieme di tutti i messaggi validi è indicato con C e rappresenta un sottoinsieme dell'insieme di tutte le combinazioni possibili, pari a q^n. Questo concetto è fondamentale per comprendere come sia possibile rilevare ed eventualmente correggere gli errori nelle comunicazioni digitali: il ricevitore, conoscendo l'insieme C rileva un errore se la parola ricevuta non appartiene a tale insieme. La nozione appena descritta verrà approfondita ulteriormente nei prossimi capitoli.

Dal punto di vista pratico, un codice si classifica quale codice a blocco quando l'encoder elabora un insieme di dati (bit nel caso di sorgente binaria) simultaneamente. È importante sottolineare che tale insieme di dati può anche essere costituito da un singolo simbolo, come accade, ad esempio, nel caso dei codici a ripetizione.

Viene utilizzata la lettera k per rappresentare il numero di bit in ingresso all'*encoder* e n per il numero di bit in uscita.

Un valore del *rate* vicino all'unità indica un codice con poca ridondanza, mentre valori più bassi indicano una maggiore ridondanza.

[1] Per una descrizione più dettagliata si rimanda a [2]

1.3 Distanza di Hamming e Distanza Minima

Per analizzare e confrontare le prestazioni delle diverse codifiche è necessario stabilire una metrica di confronto per le parole di codice. Esistono diverse metriche[2]: in questa sede si adotterà la metrica di Hamming.

La distanza di Hamming tra due parole di codice è definita come il numero di posizioni in cui i simboli delle due parole differiscono.

Per chiarire il concetto, si riporta il seguente esempio:

Siano C_1 e C_2 due parole di codice:

$$\mathrm{C}_1 = 0001 \qquad\qquad \mathrm{C}_2 = 0111.$$

La distanza di Hamming $\mathrm{d}_\mathrm{H}(\mathrm{c}_1, \mathrm{c}_2)$ è pari a due in quanto le due parole differiscono in due posizioni (specificamente nella seconda e nella terza cifra binaria). Nel caso di un alfabeto M-ario (non binario), esistono metriche alternative che possono tener conto anche della distanza effettiva tra i simboli nelle stesse posizioni. Tuttavia, come già anticipato, nel contesto di questa trattazione si considera esclusivamente la metrica di Hamming.

Si assuma che date tre parole di codice $x, y, z \in F^n$, valgano:

- $\mathrm{d}_\mathrm{H}(x, y) \geq 0$; $\mathrm{d}_\mathrm{H}(x, y) = 0 \iff x = y$

- $\mathrm{d}_\mathrm{H}(x, y) = \mathrm{d}_\mathrm{H}(y, x)$ (Simmetria)

- $\mathrm{d}_\mathrm{H}(x, y) \leq \mathrm{d}_\mathrm{H}(x, z) + \mathrm{d}_\mathrm{H}(z, y)$ (Disuguaglianza Triangolare)

Il peso di Hamming w di una parola corrisponde al numero di bit diversi da "0" all'interno della stessa. Per esempio, considerando le stesse parole:

$$\mathrm{C}_1 = 0001 \qquad\qquad \mathrm{C}_2 = 0111.$$

[2] È possibile effettuare uno studio accurato della metrica di Lee in [2]

Si ottiene:

$$w(\mathrm{C}_1) = 1, \qquad w(\mathrm{C}_2) = 3$$

Vale inoltre la seguente relazione: $d(x, y) = w(y - x)$ [3]

Definizione: *Si definisce distanza minima (d) di un codice come la distanza di Hamming più piccola tra due parole di codice distinte appartenenti al codice.*

In altre parole, prese le due parole di codice più simili tra loro (cioè quelle che differiscono nel minor numero di posizioni), la distanza di Hamming tra di esse corrisponde alla Distanza Minima del codice. Espresso in termini matematici, il concetto si traduce nella seguente formula:

$$d(C) = \min\left(d_H(C_1, C_2) | C_1, C_2 \in C, C_1 \neq C_2\right)$$

Nella seguente trattazione al fine di snellire la nomenclatura, quando il codice a cui si fa riferimento è inequivocabile, può venir omessa la (C), indicando la distanza minima solo con "d".

1.4 Prestazioni di una Codifica

La Distanza Minima di un codice ha un impatto diretto nelle sue prestazioni. In particolare, essa determina sia il numero massimo di errori che il codice può rilevare, sia il massimo numero di errori che può correggere.

Utilizzando una descrizione discorsiva:

- Un codice è in grado di rilevare un numero di errori tale per cui la parola ricevuta non venga trasformata in una parola di codice valida.
- Un codice è in grado di correggere un numero di errori tale per cui la parola di codice effettivamente inviata rimanga la parola più simile (in termini di distanza di Hamming) a quella ricevuta.

Questo concetto può essere formalizzato matematicamente come segue

[3] Viene omessa la dimostrazione matematica, il lettore può approfondire dalla fonte [2]

Teorema[4]**:**

- *Un codice C può rilevare fino a τ errori se $d(C) \geq \tau + 1$.* ($\tau \leq d(C) - 1$)
- *Un codice C può correggere fino a τ errori se $d(C) \geq 2\tau + 1$.* ($\tau \leq \frac{d(C)-1}{2}$)

Dove $d(C)$ indica la Distanza Minima del codice.

La dimostrazione del primo enunciato risulta piuttosto semplice e si basa su un ragionamento logico, senza ricorrere allo studio matematico.

Dimostrazione

Sia C un codice (M, n, d) definito su F e sia y il messaggio ricevuto da decodificare. Si descrive con una funzione il processo di decodificazione come segue:

$$D(y) = \begin{cases} y & se\ y \in C \\ e & altrimenti \end{cases}$$

La rivelazione dell'errore fallisce solo nel caso in cui y appartiene a C, ma non è la parola di codice effettivamente inviata in trasmissione. Questo è possibile solo se si verificano un numero di errori tali da trasformare una parola di codice in un'altra, ossia se si sono verificati almeno $d(C)$ errori. Conseguenza immediata è che il numero di errori rilevabili è pari a $d(C) - 1$.

La dimostrazione della seconda preposizione viene eseguita ragionando per assurdo.

Dimostrazione

Sia C un codice (M, n, d) definito su F e sia y il messaggio ricevuto da decodificare. Sia $D(y)$ la parola di codice in C più vicina a y. Si assuma c come parola ricevuta dove $d(y, c) \leq \frac{d-1}{2}$. Si ipotizzi per assurdo $c' = D(y) \neq c$.

[4] Enunciato e dimostrazioni consultabili nel testo [2]

Dalla definizione di $D(y)$ risulta allora:

$$d(y,c') \leq d(y,c) \leq \frac{d-1}{2}$$

Utilizzando la diseguaglianza triangolare:

$$d \leq d(c,c') \leq d(y,c) + d(y,c') \leq d-1$$

Che risulta essere una contraddizione.

Si fornisce un'interpretazione geometrica dell'enunciato.

Definizione: *Sia F un alfabeto e t un intero non negativo. Una Sfera di Hamming di raggio t, centrata su una parola x, è l'insieme di tutte le parole di codice che si trovano ad una distanza di Hamming $d_H \leq t$ da x.*

Matematicamente:

$$S(x,t) = \{y \in F^n | \, d_H(x,y) \leq t\}$$

Teorema:

Sia C un codice (n, M, d) su F, allora tutte le sfere (di Hamming) di raggio $\tau = \frac{d(C)-1}{2}$ centrate su ogni parola di codice devono essere disgiunte:

$$\forall \, c, c' \in C \;\; S(c,\tau) \cap S(c',\tau) = \emptyset$$ [5]

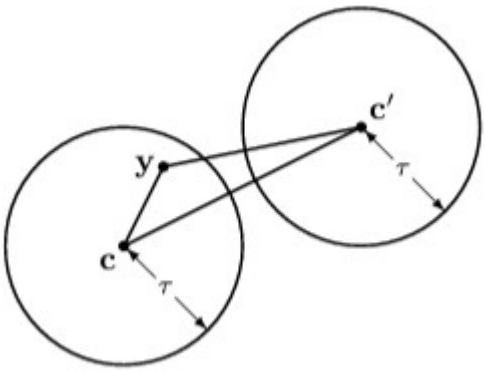

Figura 4: Sfere di Hamming

[5]Riferimento alla fonte [2]

La Figura 4 mostra due sfere di Hamming di raggio τ centrate su due parole di codice c e c'. Correggendo tramite Distanza Minima una parola ricevuta contenuta all'interno di una sfera Ω, il decodificatore (decoder) traduce il messaggio ricevuto nella parola di codice situata al centro di Ω. Ricevendo y il decodificatore corregge il messaggio nella parola di codice c.

Per favorire una migliore comprensione, si riporta un esempio pratico utilizzando un codice a ripetizione $C(3,2,3)$. Questo tipo di codice opera ripetendo n volte il singolo bit in ingresso all'encoder. Nel caso specifico, il valore di k è unitario (poiché l'encoder elabora un bit alla volta), mentre per n si considera un valore pari a tre.

I simboli in ingresso all'encoder vengono quindi trasformati come segue:

$$0 \rightarrow 000 \quad 1 \rightarrow 111$$

Si consideri il caso in cui venga trasmessa la parola di codice 000 e si valutino le possibili combinazioni ricevute:

1) *Ricezione con un solo errore*

   Se la parola ricevuta contiene un singolo errore (001, 010, 100), l'errore viene rilevato poiché nessuna di queste combinazioni corrisponde a una parola di codice valida (000, 111). Inoltre, è facile verificare che tutte e tre le combinazioni sono più simili (in termini di distanza di Hamming) alla parola trasmessa 000. L'errore, quindi, viene corretto selezionando 000 come parola di codice effettiva.

2) *Ricezione con due errori*

   Se la parola ricevuta contiene due errori (011,110,101), l'errore viene comunque rilevato, in quanto nessuna delle combinazioni è una parola di codice valida. Tuttavia, in questo caso, tutte le combinazioni risultano più simili (sempre in termini di distanza di Hamming) alla parola 111. Il processo di correzione attribuisce quindi erroneamente il messaggio

ricevuto alla parola di codice 111, introducendo un ulteriore errore. In questo caso, dunque, l'errore non viene corretto.

3) *Ricezione con tre errori*

Se la parola ricevuta contiene tre errori (111), essa coincide con una parola di codice valida. Tuttavia, questa parola non corrisponde a quella effettivamente trasmessa (000). In questo caso, l'errore non viene né rilevato né corretto poiché la parola ricevuta risulta valida.

Nel codice considerato ($n = 3$), le uniche parole di codice ammesse sono 000 e 111. La distanza minima del codice corrisponde alla distanza di Hamming tra queste due parole, ovvero:

$$d(C) = 3$$

Utilizzando il teorema precedentemente enunciato si ottengono i seguenti risultati:

$3 \geq \tau + 1 \rightarrow \tau \leq 2$ è possibile rilevare al massimo due errori

$3 \geq 2\tau + 1 \rightarrow \ \tau \leq 1$ è possibile correggere al massimo un errore

I risultati sono quindi in linea con quanto previsto.

1.5 Codice Lineare

Viene introdotta un'altra nozione basilare, ossia quella di Codice Lineare.

Definizione: *Un codice Lineare $C(n, M, d)$ definito su un campo F è un sottospazio di F^n e soddisfa le seguenti proprietà:*

- *comprende l'elemento neutro*

- *prese due qualsiasi parole di codice C_1 e C_2, una loro combinazione lineare è ancora una parola di codice: $\forall\, C_1, C_2 \in C$; $\forall\, a_1, a_2 \rightarrow\ a_1C_1 + a_2C_2 \in C$ (dove a_1, a_2 indicano una coppia di scalari)*

Una proprietà importante dei codici lineari, che verrà sfruttata anche in fase di sperimentazione dall'algoritmo utilizzato, è descritta dal seguente:

Teorema

Sia C un codice lineare (n, k, d) su F. Risulta allora:

$$d(C) = \min_{c \epsilon C \setminus \{0\}} w(c)$$

La distanza minima del codice corrisponde alla parola di peso minore, escludendo la parola nulla[6].

Dimostrazione

Per linearità:

$$c_1, c_2 \in C \Longrightarrow c_1 - c_2 \in C$$

Ricordando che:

$$d(c_1, c_2) = w(c_1 - c_2)$$

Dalla definizione di distanza minima:

$$d(C) = \min_{c_1, c_2 \epsilon C:\, c_1 \neq c_2} d(c_1, c_2) = \min_{c_1, c_2 \epsilon C:\, c_1 \neq c_2} w(c_1 - c_2) = \min_{c \epsilon C \setminus \{0\}} w(c)$$

1.5.1 Matrice Generatrice

Assunti i concetti di Codice Lineare e Codice a Blocco, viene introdotta ora una nozione importante per comprendere come questi codici sono definiti.

Definizione: *La matrice generatrice (G) di un codice lineare (n, k, d) è una matrice $k \times n$ le cui righe sono una base per rappresentare ogni parola di codice.*

In altre parole, tramite una combinazione lineare delle righe di (G) è possibile ottenere ogni parola che appartiene al codice.

L'idea dietro i codici lineari è quella di creare un mapping che assegni ad ogni possibile combinazione in ingresso all'*encoder* (quindi prodotta dalla sorgente),

[6] Fonte [2]

una parola di codice in modo biunivoco. Indicando con a il vettore informazione, si deve creare un mapping che assuma la seguente forma:

$$a \rightarrow aG^{7}$$

Tenendo conto che:

- il vettore a ha lunghezza k
- G è una matrice di dimensioni $k \times n$

si deduce che la parola di codice aG ha lunghezza n, in accordo con la Figura 1. Per definizione il rango massimo di una matrice è il minore tra il numero di righe e il numero di colonne della matrice stessa. Assumendo $rank(G) = k$ e ricordando che la condizione $k \leq n$ è sempre verificata, si crea un mapping biunivoco, in altre parole, ogni vettore di informazioni a corrisponde a una parola di codice unica e ogni parola di codice corrisponde a un unico vettore di informazioni.

1.5.2 Matrice di Parità

Affiancata alla matrice generatrice esiste un'altra matrice, detta di Parità, che svolge un ruolo fondamentale nelle codifiche lineari.

Definizione: *Si consideri un codice lineare* $C\ (n, k, d)$ *definito su un campo* F.

La matrice di parità (H)*, di dimensioni* $r \times n$ *, soddisfa la seguente relazione:*

$$\forall\ c \in C, \qquad Hc^{T} = 0 \iff c \in C$$

Per ogni parola di codice il prodotto tra la matrice H e la parola è nullo.

È possibile ricavare le dimensioni della matrice di parità matematicamente. Ricordando che c è un vettore riga composto da n elementi (Interpretato come una matrice $1 \times n$) si ricava che c^{T} abbia dimensioni $n \times 1$. Il prodotto righe per colonne è matematicamente possibile se il numero delle colonne del primo

[7] Per approfondimenti si rimanda a [2]

fattore è uguale al numero di righe del secondo. Quest'ultima osservazione è sufficiente per affermare che il numero delle colonne della matrice H sia pari a n. Per determinare la dimensione r invece occorrono delle osservazioni geometriche.

Il Teorema del Rango afferma che la somma del rango e della dimensione del nucleo di una matrice è pari al numero di colonne della stessa. Applicando l'enunciato alla matrice H si ottiene:

$$n = rank(H) + \dim(\ker H)$$

Il *kernel* della matrice H è l'insieme di tutti i vettori x che soddisfano $Hx^T = 0$, $x \in F^n$.

Per definizione di matrice di parità la relazione è soddisfatta da ogni parola di codice, di conseguenza il codice C è il *kernel* destro di H :

$$C = \ker(H) \rightarrow \dim(kerH) = \dim(C)$$

La dimensione del codice C è nota ed è pari a k. Si ottiene quindi:

$$n = rank(H) + k \rightarrow rank(H) = n - k$$

Considerando questo risultato e considerato che il numero delle colonne di H è pari a n, è banale dedurre che:

$$r = n - k$$ [8]

Esistono diversi metodi per calcolare la matrice di parità in ragione ai vincoli matematici descritti. È possibile estrapolare la matrice H partendo dalla matrice generatrice del codice. (Geometricamente si verifica $HG^T = 0$, si ricava quindi H partendo dallo studio del nucleo di G). In generale esistono diverse matrici generatrici e di conseguenza diverse matrici di parità per lo stesso codice[9].

[8] Per approfondimenti è possibile consultare [1] , [2]

[9] Per approfondire ulteriormente si rimanda alla fonte [2]

1.6 Decodifica di un codice lineare

La decodifica di un codice lineare può avvenire in molteplici schemi, nella seguente trattazione ne verranno introdotti due:

1) Criterio dell'Array-Standard
2) Criterio del Vettore Sindrome

Entrambi i metodi menzionati hanno come obiettivo quello di decodificare il messaggio trovando la parola di codice più vicina a quella ricevuta. Rispettivamente:

- Data una parola ricevuta $y \in F^n$, si trova la parola di codice $c \in C$ che minimizza la distanza $d(y, c)$
- Data una parola ricevuta $y \in F^n$, si trova la parola $e \in F^n$ con il minimo peso di Hamming tale che $y - e \in C$

Anche se il concetto viene esplicato in modo differente, in realtà i due criteri sono equivalenti[10].

1.6.1 Criterio dell'Array Standard

Segue una breve spiegazione dell'approccio utilizzato in fase di decodifica tramite Array Standard.

Definizione: *Sia C un codice lineare (n, k, d) definito su $F = GF(q)$. Un Array Standard per il codice C è una matrice di dimensioni $q^{n-k} \times q^k$ composta da elementi di F^n e definita come segue:*

- *La prima riga della matrice contiene le parole codice di C, partendo dalla parola codice nulla.*
- *Ogni riga successiva inizia con una parola $e \in F^n$ di peso di Hamming minimo che non sia ancora apparsa nelle righe precedenti, seguita dalle*

[10] Si rimanda alla fonte [2]

parole $e + c$, dove c varia su tutte le parole codice non nulle di C nell'ordine in cui appaiono nella prima riga.

Si riporta un esempio:

Sia C un codice lineare $(5,2,3)$ definito su $GF(2)$ con matrice generatrice G:

$$G = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

Un Array standard associato al codice può essere il seguente:

00000	10110	01011	11101
00001	10111	01010	11100
00010	10100	01001	11111
00100	10010	01111	11001
01000	11110	00011	10101
10000	00110	11011	01101
00101	10011	01110	11000
10001	00111	11010	01100

Ogni riga dell'array standard rappresenta un *coset* di C in F^n. Due parole $y_1, y_2 \in F^n$ appartengono alla stessa riga se e solo se $y_1 - y_2 \in C$.

I *coset* di C formano una partizione di F^n in q^{n-k} sottoinsiemi, ciascuno di dimensione q^k.

La prima parola di ogni riga, caratterizzata dal peso minore, prende il nome di Leader del *coset*.

Ricevendo una parola y, l'errore e rilevato dal decodificatore deve essere tale che $y - e \in C$ per definizione. In altre parole, la parola decodificata deve appartenere allo stesso *coset* di y. Inoltre, decodificare cercando la parola più vicina a quella ricevuta implica che e sia la parola con peso minore all'interno del suo *coset*.

La strategia di decodifica è quindi la seguente:

- Data la parola ricevuta $y \in F^n$ viene ricercata la riga dell'Array standard che la contiene e si sceglie il leader (primo elemento) come errore e della parola
- La parola decodificata c è calcolata come $c = y - e$.

In realtà non è necessario effettuare la sottrazione perché si verifica che, per costruzione dell'array standard, c è la prima parola nella colonna che contiene y.

Come esempio si suppone la ricezione di $y = 01111$ corrispondente all'elemento nella quarta riga e terza colonna della matrice riportata. Il leader della quarta riga è 00100, si ha quindi:

$$y = 01111; e = 00100 \rightarrow c = y - e = 01011$$

La combinazione 01011 corrisponde al primo elemento della terza colonna.

Il metodo di decodificazione appena descritto non è molto utilizzato perché poco pratico, resta però un esempio di come la linearità del codice possa venire sfruttata per effettuare la decodifica.

1.6.2 Criterio del Vettore Sindrome

Si introduce il secondo metodo di decodifica trattato nel seguente documento.

Definizione: *Sia C un codice lineare (n, k, d) definito su $F = GF(q)$ e si consideri una sua matrice di parità H. Si definisce vettore sindrome della parola $y \in F^n$ come:*

$$s = Hy^T$$

Si ricorda che per definizione di H vale:

$$Hc^T = 0 \iff c \in C$$

Dati $y_1, y_2 \in F^n$ risulta:

$$y_1 - y_2 \in C \iff Hy_1^T = Hy_2^T$$

Quindi y_1, y_2 appartengono allo stesso *coset* se e solo se le loro sindromi sono uguali. Di conseguenza esiste una corrispondenza biunivoca tra i q^{n-k} coset di C in F^n ed i q^{n-k} valori possibili delle sindromi.

La decodifica assume quindi la seguente forma:

- Data la parola ricevuta $y \in F^n$ si calcola $s = Hy^T$ (Calcolo della Sindrome)
- Si trova il vettore errore $e \in F^n$ col minimo peso che soddisfa: $s = He^T$ (Ricerca del *coset-leader* nel *coset* della parola ricevuta)

Il secondo passaggio corrisponde alla ricerca del minimo sottoinsieme di colonne di H il cui span contiene il vettore s. Questo metodo, nonostante sia largamente utilizzato, risulta essere molto costoso in termini computazionali. Esistono diverse soluzioni che permettono una decodifica (tramite algoritmi) a basso costo.

Un approccio molto utilizzato per rendere ciò possibile si basa sul metodo di costruzione della matrice H, come si vedrà in seguito.

1.7 Codici LDPC

I codici LDPC (Low-Density Parity-Check) sono stati introdotti per la prima volta da Robert G. Gallager nel 1963 durante il suo dottorato al MIT[11]. Tuttavia, nonostante il loro potenziale innovativo, la complessità computazionale degli algoritmi di decodifica disponibili al tempo ne ha limitato l'adozione pratica fino agli anni '90, quando i progressi tecnologici e computazionali hanno permesso di riscoprirli e utilizzarli su larga scala.

I codici LDPC sono codici lineari caratterizzati da una matrice di parità (H) particolarmente "sparsa", ovvero contenente una bassa densità di elementi pari a “1”. Questa proprietà consente di implementare algoritmi di decodifica iterativa ad alta efficienza che li rende estremamente competitivi rispetto ad altre soluzioni.

[11] Fonte [1]

Grazie alle loro eccellenti proprietà di correzione degli errori, i codici LDPC hanno trovato applicazione in numerosi ambiti tecnologici. Tra questi, i più rilevanti includono:

1. Telecomunicazioni

   -5G e Wi-Fi: Nei sistemi di comunicazione wireless di nuova generazione, i codici LDPC sono utilizzati per garantire un'elevata affidabilità dei dati trasmessi nonostante la presenza di interferenze e rumore.

   -DVB-S2 (Digital Video Broadcasting - Satellite Second Generation): Nei sistemi di trasmissione satellitare, i codici LDPC sono stati adottati per la loro capacità di avvicinarsi al limite di Shannon, migliorando la qualità del segnale ricevuto anche in condizioni di trasmissione difficili.

2. Archiviazione dei dati

   -Blu-ray: Nei dischi Blu-ray, i codici LDPC sono utilizzati per la correzione degli errori durante la lettura dei dati, garantendo una riproduzione priva di difetti.

   -Memorie Flash: In molte memorie di tipo NAND Flash, i codici LDPC vengono utilizzati per aumentare l'affidabilità dei dati e prolungare la vita utile dei dispositivi.

3. Applicazioni spaziali

   -Nei sistemi di comunicazione interplanetaria, dove la distanza e il rumore rendono cruciale l'uso di codici a elevata capacità di correzione degli errori.

Nonostante i numerosi vantaggi, i codici LDPC presentano alcune limitazioni:

1. Complessità Computazionale: La decodifica iterativa, sebbene efficiente, può richiedere significative risorse computazionali per lunghezze di codice molto elevate.
2. Prestazioni su Piccoli Blocchi: Per codici con lunghezze ridotte, i codici LDPC non sempre raggiungono le prestazioni di altre famiglie di codici.
3. Ottimizzazione della Matrice di Parità: La costruzione di una matrice H ottimale è ancora oggetto di ricerca, con algoritmi che cercano di bilanciare sparsità ed efficacia.

Questi aspetti rappresentano aree di ricerca attiva, con l'obiettivo di migliorare ulteriormente le loro prestazioni e facilitarne l'adozione in contesti sempre più ampi.

2 – Sperimentazione

Terminata la panoramica sui concetti teorici essenziali, si passa ora alla sperimentazione in oggetto.

2.1 Introduzione

Per un codice LDPC, non esiste attualmente una relazione matematica formalmente definita che lega le grandezze stabilite in fase di generazione (come n, R) e la Distanza Minima ottenuta. Sebbene sia intuitivo supporre che le grandezze citate siano connesse alla Distanza Minima, tale legame non è ancora stato descritto attraverso una formula precisa.

Si presume logicamente che all'aumentare della lunghezza del codice (n), la Distanza Minima tenda ad aumentare. Tuttavia, la domanda che ci si pone è in che modo tale distanza cresce rispetto a n.

L'obiettivo della prima sperimentazione è osservare come varia la Distanza Minima stimata di un codice LDPC al variare della lunghezza del codice.

I dati necessari per questa analisi vengono raccolti sperimentalmente attraverso simulazioni che si basano sull'esecuzione di un algoritmo specifico per la stima della Distanza Minima.

2.2 Algoritmo

L'algoritmo utilizzato nella sperimentazione è composto da due blocchi distinti, ciascuno responsabile di mansioni specifiche.

Primo Blocco:

Il primo blocco dell'algoritmo genera 1.000 codici LDPC ad ogni sua esecuzione. Tutti i codici sono caratterizzati dallo stesso *rate*, dalla stessa lunghezza n, e dallo stesso peso della matrice generatrice G (dove con il termine "peso" si indica il numero di "1" presenti in ogni colonna della matrice G). I codici vengono formati a partire da matrici generatrici che differiscono tra loro solo per la posizione degli elementi pari a "1", ma non per il numero.

Secondo Blocco:

Il secondo blocco dell'algoritmo stima la distanza minima di ciascun codice generato utilizzando un algoritmo chiamato *Information Set Decoding* (ISD), solitamente impiegato nei processi di decodifica.

2.2.1 ISD – Information Set Decoding

L'ISD rappresenta una soluzione di tipo algoritmico alla decodifica di codici lineari. Solitamente viene utilizzato per testare la sicurezza di codici lineari nell'ambito della crittografia. L'idea alla base dell'algoritmo consiste nel selezionare un sottoinsieme di informazioni *(Information Set)* di posizioni di bit supposti non corrotti. Una volta scelto il set, l'algoritmo cerca di decodificarlo tentando correzioni iterative nel caso si verifichino degli errori, modificando se necessario il set scelto.

Generalmente, utilizzando la nomenclatura O-grande (limite asintotico superiore) si attribuisce a questo algoritmo una complessità esponenziale:

$$O(2^k)$$

Indicando con k il numero di informazioni da decodificare.

La complessità è stimata nel caso peggiore. In generale l'efficacia dell'ISD dipende fortemente dalla capacità decisionale sulla scelta del *Set*.

L'efficienza di un algoritmo è strettamente legata al costo in termini di risorse richieste (dove per "risorse" si intendono tempo di esecuzione e memoria necessaria). Sebbene possa sembrare banale affermare che un algoritmo dovrebbe richiedere il minor numero possibile di risorse, è importante sottolineare che in alcune situazioni i costi possono essere talmente elevati da renderne impraticabile l'esecuzione.

La funzione distanza di un codice a controllo di parità $N(w)$, è definita come il numero di parole di codice di peso w. $N(w)$ rappresenta quindi il numero di parole a distanza w da una data parola di codice. Si ricordi che per codici lineari la

distanza minima $d(C)$ è definibile anche come il più piccolo valore di w tale che $N(w) \neq 0$. Pertanto, stimare la distanza minima equivale a individuare tale valore.

Un approccio di *brute force* al problema dovrebbe testare ogni possibile parola di codice c tale che $Hc^T = 0$, individuando quella con il peso minimo. Tuttavia, la ricerca di una parola di peso w in un codice di lunghezza n comporterebbe un numero di operazioni pari a $\binom{n}{w}$, il che rende questa strategia impraticabile per valori di n e w elevati.

L'algoritmo *Information Set Decoding* utilizza un approccio ibrido che combina la brute *force* con una componente di "guessing" (Come per esempio nella scelta degli elementi appartenenti all' Information Set).

L'ISD sfrutta anche osservazioni geometriche al fine di ridurre significativamente il numero di operazioni necessarie:

si scompongono i fattori del prodotto Hc^T come segue:

$$H = (H_1 H_2)\ ;\ c = (c_1 c_2)$$

Dove si assume che H_1 sia quadrata e invertibile.

$$Hc^T = 0 \rightarrow\ H_1 c_1^T + H_2 c_2^T = 0 \rightarrow\ H_1^{-1}(H_1 c_1^T + H_2 c_2^T) = 0 \rightarrow\ c_1^T =\ -H_1^{-1} H_2 c_2^T$$

Poiché si opera in un alfabeto binario, il segno negativo può essere ignorato, e si ottiene:

$$c_1^T =\ H_1^{-1} H_2 c_2^T$$

In altre parole, per ogni c_2, è possibile calcolare c_1 direttamente, senza la necessità di enumerare tutte le possibili combinazioni di c_1

Dato che c_2 è caratterizzato da lunghezza e peso ridotti rispetto a c, enumerando c_2 e calcolando matematicamente c_1, il costo computazionale dell'algoritmo si riduce notevolmente.

La componente di "guessing" viene introdotta anche nella scelta del peso w da ricercare nella parola c_2. Se viene trovata una parola valida, si calcola c_1 e si verifica che il peso totale della parola c corrisponda al valore desiderato.

2.3 Esecuzione

La sperimentazione consiste in due indagini distinte, vengono ora descritte in modo dettagliato nei paragrafi successivi.

2.3.1 Prima Sperimentazione

Come già introdotto, lo scopo della prima sperimentazione è osservare la variazione della Distanza Minima in funzione della lunghezza del codice. Le simulazioni sono state effettuate variando il parametro n ad ogni esecuzione, mantenendo però costanti il *rate* del codice (k) ed il peso della matrice generatrice (v).

Si prevede che, all'aumentare di n, aumenti anche la Distanza Minima. È importante sottolineare che incrementare la lunghezza del codice implica un aumento del numero di simboli trasmessi che non contengono informazione, con conseguente aumento della larghezza di banda, ma al contempo un miglioramento dell'efficienza del codice.

Per ogni simulazione sono state registrate sia la Distanza Minima massima che quella minima rilevate tra i mille codici generati. La Distanza Minima media è riportata come valore decimale, nonostante ciò non abbia un significato fisico nella realtà. Questa opzione è stata prediletta per evidenziare anche le più piccole variazioni della Distanza Minima media al variare dei valori di n.

Analogamente alla Distanza Minima, il costo in risorse dell'algoritmo cresce con l'aumentare della lunghezza del codice. Non è banale identificare valori di k e di v che consentano un'ampia analisi al variare di n, senza incorrere in simulazioni talmente onerose da risultare impraticabili.

La simulazione è stata eseguita con i seguenti parametri: $k = 0.5$; $v = 7$

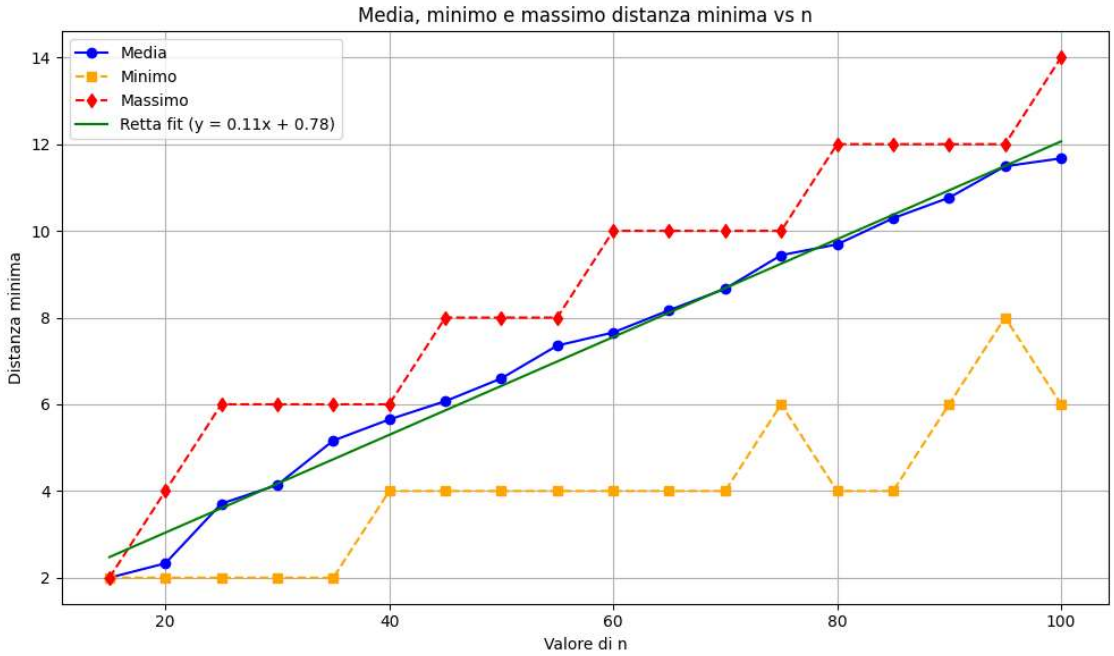

Figura 5: Prima Simulazione

In Figura 5 sono riportati i risultati ottenuti per una lunghezza di codice compresa nel range $15 \div 100$. La spezzata in blu rappresenta la congiunzione dei risultati sperimentali, dove per ogni valore di n è riportata la Distanza Minima media calcolata tra le mille simulazioni effettuate.

In verde è mostrata una retta ottenuta tramite regressione lineare (indicatore del legame lineare) che minimizza la distanza rispetto ai risultati sperimentali.

Per valutare la dispersione dei risultati, le curve in giallo e in rosso rappresentano, rispettivamente, i minimi e i massimi valori della distanza minima rilevati per ciascun valore di n.

L'andamento della curva blu, formata dai risultati sperimentali, risulta pseudo-lineare. Tuttavia, è importante considerare che le Distanze Minime sono stimate e che l'analisi è stata condotta su un range limitato di valori di n, a causa dei vincoli fisici già descritti.

Un'osservazione rilevante riguarda la dispersione dei risultati: a parità di n, non è possibile determinare con precisione quale codice sia caratterizzato dalla Distanza Minima maggiore o minore registrata. Visivamente, si nota che per alti valori di n la curva blu (Distanza Minima media) tende ad avvicinarsi

maggiormente alla curva dei massimi piuttosto che a quella dei minimi. In altre parole, è statisticamente più probabile selezionare un codice LDPC con una Distanza Minima vicina a quella media o addirittura superiore. Tuttavia, esiste la possibilità che venga selezionato il codice che, per quel determinato valore di n, ha registrato il valore minimo, con conseguenti prestazioni peggiori rispetto a quelle attese.

Potrebbe essere interessante confrontare le prestazioni rilevate per i codici LDPC con quelle di altre tipologie di codici, come i codici Random. Quest'ultimi sono caratterizzati da una matrice H i cui elementi binari sono generati casualmente (matrice non sparsa, a differenza degli LDPC). La distanza minima di questi codici può essere stimata ricorrendo alla teoria della probabilità.

Si inizia il ragionamento chiedendosi quale sia la probabilità $\Pr[Hc^T = 0]$ assumendo che:

- H sia una matrice di dimensioni $r \times n$ generata casualmente
- c sia una parola di codice casuale di peso w e lunghezza n (matrice $n \times 1$)

Il prodotto Hc^T genera un vettore (o matrice $r \times 1$):

$$Hc^T = \begin{pmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_r \end{pmatrix}$$

Dove $s_1 \dots \dots \dots s_r$ rappresentano i r bit che compongono il vettore.

Si assume che ogni bit abbia una probabilità del 50% ($1/2$) di essere pari ad "1" in quanto random. La $\Pr[Hc^T = 0]$ può essere calcolata come l'intersezione delle singole probabilità $\Pr[s_1 = 0], \Pr[s_2 = 0], \dots \dots \dots \Pr[s_r = 0]$.

Esplicando matematicamente si ottiene:

$$\Pr[Hc^T = 0] = (\tfrac{1}{2})^r = 2^{-r} = 2^{-n(1-R)}$$

Nell'ultimo passaggio si è tenuto conto del fatto che:

$$k = Rn \rightarrow r = n - k = n\left(1 - {}^{k}/_{n}\right) = n(1 - R)$$

Indicando con A_w il numero medio di parole di peso w, vale:

$$A_w = \binom{n}{w} * 2^{-n(1-R)}$$

Imporre $A_w \geq 1$ significa imporre che esista, in media, almeno una parola di peso w nel codice. In base a questa condizione, è statisticamente probabile trovare almeno due parole di codice a distanza w l'una dall'altra (assumendo una distribuzione uniforme delle parole nei codici Random). Da queste considerazioni, si conclude che la Distanza Minima di un codice Random corrisponde al minimo valore di w per cui la condizione $A_w \geq 1$ è soddisfatta.

$$d(C) = \min\left(w \middle| \binom{n}{w} * 2^{-n(1-R)} \geq 1\right)$$

Tutti i parametri della formula sono noti per ogni simulazione, di conseguenza $d(C)$ viene semplicemente calcolata.

Graficando in nero la Distanza Minima stimata dei codici random e lasciando inalterata la curva blu della Figura 5 si ottiene:

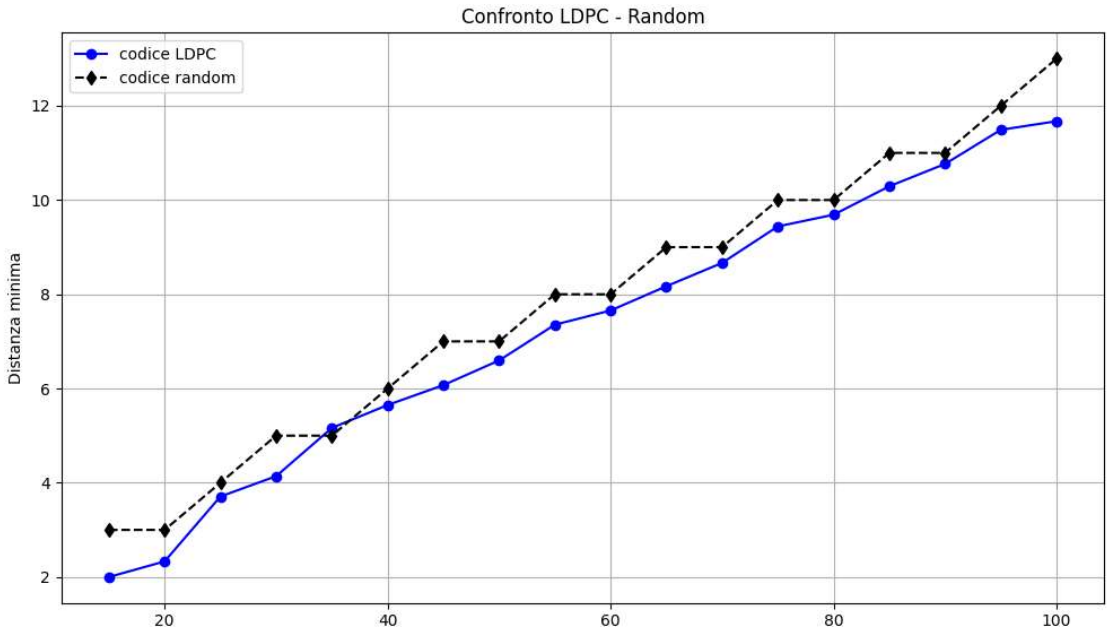

Figura 6: Seconda Simulazione

La Figura 6 evidenzia che la Distanza Minima stimata per i codici Random presenta un comportamento migliore rispetto a quella dei codici LDPC. Tuttavia, questo risultato non invalida le qualità della famiglia dei codici LDPC, il cui principale punto di forza risiede nella facilità di decodifica, pur mantenendo prestazioni complessivamente eccellenti.

2.3.2 Seconda Sperimentazione

Un ulteriore studio da condurre consiste nell'analizzare la variazione simultanea di due grandezze, nello specifico la lunghezza del codice e il peso della matrice generatrice. L'idea è quella di stabilire una relazione matematica tra le due grandezze, in modo che, fissata una di esse, l'altra venga definita univocamente. Le possibilità di definire questo legame sono infinite, ma il principale vincolo rimane il costo in termini di risorse computazionali: se nella prima sperimentazione tale costo rappresentava già un ostacolo significativo, incrementare contemporaneamente entrambe le grandezze comporta un aumento dei costi computazionali molto più repentino.

Per effettuare lo studio è stato utilizzato un legame lineare tra le due grandezze, con i seguenti valori iniziali: $n = 10 \quad v = 3$.

Le sperimentazioni hanno evidenziato che l'algoritmo mostra prestazioni migliori (in termini di tempo di esecuzione) in corrispondenza di rate (k) molto alti o molto bassi. Di conseguenza, per questa analisi è stato selezionato un valore di rate pari a 0.7.

Vengono riportati i risultati ottenuti.

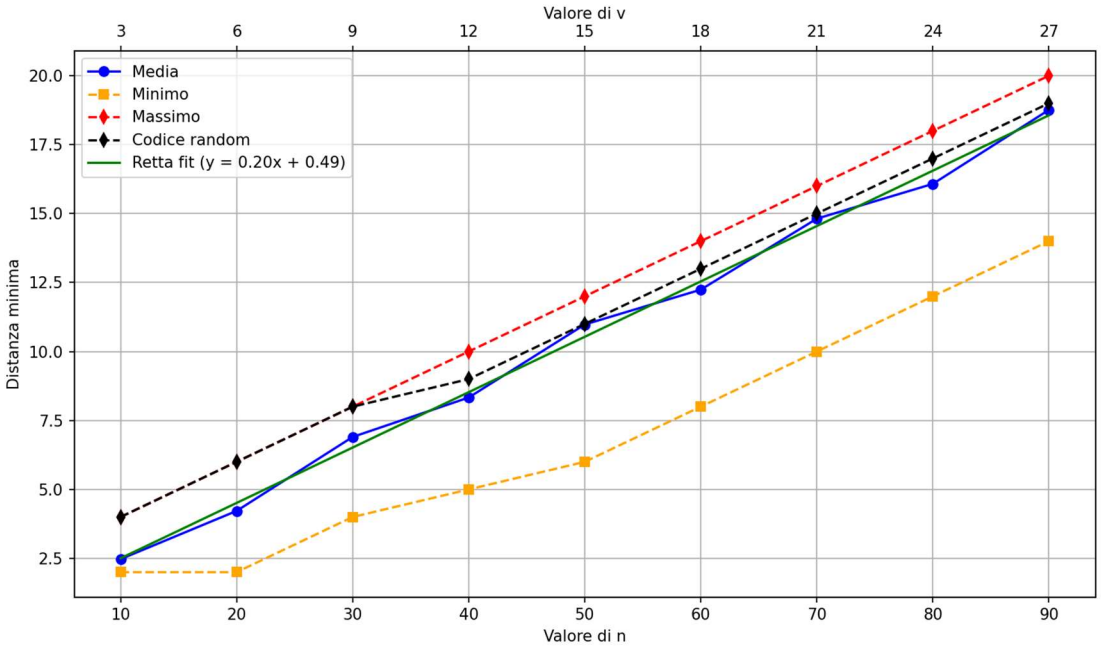

Figura 7: Terza Simulazione

Di seguito vengono riportate alcune osservazioni relative alla Figura 7:

- Per bassi valori di v e n, i codici LDPC presentano una Distanza Minima inferiore rispetto a quella dei codici Random, confermando quanto osservato nella simulazione precedente. Tuttavia, con l'aumentare di entrambe le grandezze, questa differenza tende ad annullarsi, e la distanza minima dei codici LDPC sembra convergere a quella dei codici Random. Questo risultato non afferma la superiorità dei codici Random rispetto ai LDPC: Se per basse lunghezze di codice i Random performano meglio, superato un certo breakpoint le Distanze Minime delle due famiglie sono sostanzialmente le stesse. Il paragone è però effettuato solo in termini di prestazioni correttive, tralasciando i costi di decodifica che, come già detto, sono il punto di forza dei codici LDPC. Nelle applicazioni fisiche, infatti, dovendo scegliere tra due codici con stesse capacità correttive, ma con una notevole differenza di facilità di decodifica, si opta ovviamente per il codice che richiede meno costo computazionale (Algoritmi di decodifica).

- La dispersione dei risultati risulta molto più lineare rispetto a quanto osservato nei risultati precedenti. La curva dei massimi appare come una retta, mentre la curva dei minimi può essere considerata anch'essa lineare a partire da $n = 50$.

- L'andamento della curva che rappresenta la media della Distanza Minima è caratterizzata da un comportamento pseudo-lineare. In particolare, la distanza tra la spezzata blu e la retta ottenuta tramite regressione lineare è molto ridotta, indicando un'elevata coerenza tra i dati sperimentali e il modello lineare.

- Le Distanze Minime rilevate sono superiori a quelle ottenute nella seconda sperimentazione, come d'altronde era prevedibile. Infatti, a partire dai valori $n = 30,\ v = 9$, si osservano pesi maggiori rispetto a quelli utilizzati nella prima sperimentazione, mantenendo la stessa lunghezza di codice.

- La curva blu è costantemente più vicina alla curva dei massimi rispetto a quella dei minimi. Nonostante il fattore probabilistico non può venir eliminato, in questo caso è statisticamente più probabile che, preso uno dei codici testati, esso sia caratterizzato da una Distanza Minima superiore a quella media piuttosto che inferiore.

3 - Conclusioni

In conclusione alla presente trattazione vengono fatte alcune considerazioni.

Al fine di rendere la sperimentazione più esaustiva, sarebbe opportuno ampliare il range delle lunghezze di codice adottate nello studio. In molte applicazioni pratiche, infatti, vengono utilizzati codici LDPC con lunghezze di codice significativamente maggiori rispetto a quelle esplorate in questa trattazione.

Per esempio, nel sistema DVB-S2 (Digital Video Broadcasting - Satellite Second Generation), i codici LDPC raggiungono lunghezze fino a 64.800 bit, mentre nelle applicazioni Wi-Fi e nelle reti mobili 5G, le lunghezze dei codici sono molto inferiori, tipicamente intorno a 2.000 bit. Questi riferimenti ad applicazioni concrete hanno lo scopo di offrire al lettore un "metro di paragone" con scenari applicativi reali.

Le simulazioni condotte in questo studio, tuttavia, hanno incontrato un limite fisico per valori di n appena superiori ai 100 bit. Questo limite è attribuibile principalmente alla potenza computazionale disponibile ed alla natura interpretata del linguaggio Python, utilizzato per implementare il metodo di simulazione. Per superare questa barriera, si potrebbe considerare non solo un incremento delle risorse di calcolo, ma anche l'adozione di strategie di ottimizzazione dell'algoritmo, come la riscrittura in un linguaggio più efficiente dal punto di vista computazionale (ad esempio, C++ o linguaggi orientati alle prestazioni).

Un’osservazione importante riguarda anche l’algoritmo: esso non tiene conto della sparsità della matrice H, in altre parole sarebbe possibile alleggerire le simulazioni modificando il codice per renderlo specifico al caso in esame. Nonostante questi limiti, i risultati sperimentali ottenuti dimostrano che il metodo utilizzato è una via affidabile per ottenere stime accurate della Distanza Minima, dal punto di vista probabilistico. Con le opportune ottimizzazioni e risorse aggiuntive, sarebbe possibile estendere lo studio a lunghezze di codice di maggiore interesse pratico, avvicinandosi così ai requisiti delle applicazioni reali.

4-Bibliografia

[1] - Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.

[2] - Ron M Roth. *Introduction to coding theory*. Cambridge University Press, 2006.