UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea Triennale in Matematica

Codici lineari

Relatore:
Prof.ssa Eloisa M. Detomi

Laureanda:
Nadia Giusto

Matricola:
1144847

ANNO ACCADEMICO 2021/2022

23 giugno 2022

Indice

Introduzione **iii**

1 Teoria delle comunicazioni **1**
1.1 Sistemi di comunicazione 1
1.2 Rumore 2
1.3 Canali simmetrici 2
1.4 Ridondanza 3

2 Teoria dei codici **5**
2.1 Alfabeti 5
2.2 Codici a blocchi 7
2.3 Codici rivelatori e correttori 8
2.4 Efficienza di un codice 12
2.5 Codici equivalenti 13

3 Codici lineari **15**
3.1 Motivazioni 15
3.2 Definizione e proprietà 15
3.3 Peso di Hamming 17
3.4 Matrice generatrice 18
3.5 Equivalenza 21

4 Codifica e decodifica di codici lineari **25**
4.1 Codifica 25
4.2 Decodifica tramite tabella standard 27
4.3 Codice duale e matrice di controllo 31
4.4 Decodifica per sindrome 34

5 Esempi di codice lineare **38**
5.1 Codice di ripetizione 38
5.2 Codice di parità 39
5.3 Codice di Hamming 39

Conclusioni **42**

Introduzione

La comunicazione digitale è alla base della nostra società e l'utilizzo sempre più ampio della tecnologia richiede lo sviluppo di varie tecniche affinché essa sia adeguata alle esigenze. Il padre della teoria delle comunicazioni è Claude Shannon che, con l'articolo riportato nel libro [6], pone le basi per implementare sistemi di comunicazione efficaci e in grado di gestire i problemi di trasmissione delle informazioni. Nelle sue parole troviamo lo scopo della teoria dei codici, di cui ci occuperemo in questa tesi:

> *"il problema fondamentale delle comunicazioni è quello di riprodurre esattamente o approssimativamente in un certo punto un messaggio scelto in un altro punto".*

Nelle comunicazioni digitali, infatti, la trasmissione di informazioni avviene attraverso un canale fisico ed è soggetta a errori che avvengono con una probabilità mai nulla: l'informazione ricevuta potrebbe non corrispondere del tutto a quella inviata o potrebbe non avere un significato, rendendo inutile la comunicazione. È dunque necessario sviluppare delle tecniche che permettano di rilevare la presenza di questi errori ed eventualmente correggerli o richiedere una nuova trasmissione dei dati corrotti. Esse si basano sui codici, insiemi di sequenze di simboli che possono avere delle particolari strutture algebriche.

Lo scopo di questa tesi è quello di presentare una descrizione in parte elementare dei codici lineari: partendo dalla definizione generale di codice, si esporranno alcuni dei risultati principali e si mostreranno alcuni algoritmi in cui essi vengono utilizzati. Abbiamo scelto un approccio abbastanza semplice, accessibile ad un pubblico non specialistico, per un eventuale uso didattico.

Nel Capitolo 1 descriveremo brevemente la teoria delle comunicazioni sviluppata da Shannon, in modo da vedere i motivi che hanno portato allo sviluppo della teoria dei codici. In particolare parleremo della struttura dei sistemi di comunicazione soffermandoci sui canali fisici soggetti a rumore.

Nel Capitolo 2 introdurremo la teoria dei codici dando le definizioni e presentando i risultati principali necessari a proseguire nello studio dei codici lineari. In questo capitolo ci soffermeremo sulla rilevazione e correzione di errori dovuti al rumore e su un particolare tipo di codici, i codici a blocchi.

Nel Capitolo 3 definiremo i codici lineari a partire dai codici a blocchi e dimostreremo alcuni teoremi. Presenteremo inoltre le proprietà dei codici lineari che li rendono particolarmente adatti ad essere utilizzati nelle applicazioni pratiche.

Nel Capitolo 4 descriveremo alcuni algoritmi di codifica e decodifica che utilizzano i codici lineari e introducendo allo scopo il codice duale.

Per concludere, nel Capitolo 5 presenteremo brevemente alcuni esempi di codici lineari e le loro applicazioni pratiche.

Capitolo 1

Teoria delle comunicazioni

1.1 Sistemi di comunicazione

Un sistema di comunicazione è un insieme di hardware che permette lo scambio di informazioni a distanza tra due o più persone o dispositivi.
La sua struttura di base è data da:

1. una **sorgente**: genera il messaggio da inviare che può eventualmente essere crittografato;
2. un **codificatore** (o trasmettitore): trasforma il messaggio producendo un segnale adatto a viaggiare lungo il canale;
3. un **canale**: è il mezzo utilizzato per trasferire il messaggio (sotto forma di segnale);
4. un **ricevitore**: riceve il segnale e ricostruisce il messaggio;
5. una **destinazione**: è la persona o il dispositivo a cui è stato inviato il messaggio.

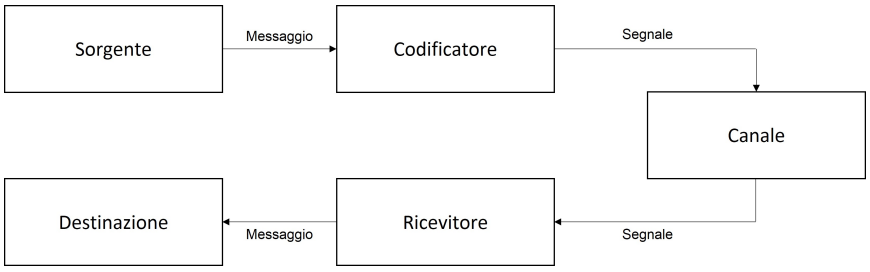

Figura 1.1: Struttura sistema di comunicazione

Un classico esempio di sistema di comunicazione è dato dalla telefonia. In questo caso la sorgente del messaggio è una persona che pronuncia una

sequenza di parole. L'apparecchio utilizzato deve quindi rilevare i suoni prodotti, cioè delle variazioni di pressione dell'aria, e trasformarli in una serie di impulsi elettrici che potranno viaggiare lungo il filo del telefono. Una volta che il segnale elettrico sarà giunto al ricevitore, verrà riconvertito in suono e potrà essere ascoltato dal destinatario del messaggio.

I sistemi di comunicazione possono essere classificati in tre categorie in base al tipo di messaggio e segnale che possono trasmettere:

- **discreti**: messaggio e segnale sono formati da un numero finito di simboli. Ad esempio, nel telegrafo il messaggio è una sequenza di lettere e il segnale una sequenza di punti, linee e spazi.
- **continui**: messaggio e segnale sono rappresentati da funzioni continue. Per esempio, nella radio vengono trasmesse a varie frequenze delle onde continue.
- **misti**: sono combinazioni di entrambi i precedenti. Per esempio, sistemi in cui si passa da un segnale analogico a uno digitale.

Nel seguito considereremo i sistemi discreti.

1.2 Rumore

Durante una comunicazione reale la probabilità che il segnale trasmesso non venga ricevuto del tutto correttamente non è mai nulla. Ciò è dovuto alla presenza di rumore: difetti fisici e deterioramento nei dispositivi coinvolti oppure interferenze di varia natura possono alterare il segnale.

Supponiamo di voler comunicare l'esito positivo o negativo di un esperimento avendo a disposizione solo due simboli, 0 e 1, per comporre il messaggio. Potremmo pensare di inviare il simbolo 1 per l'esito positivo e il simbolo 0 altrimenti. Trasmettendo solamente il simbolo di nostro interesse, a causa del rumore, il destinatario riceve il simbolo sbagliato con una probabilità p e il simbolo corretto con una probabilità $(1 - p)$.
Per rendere la comunicazione efficace, è quindi necessario sviluppare delle tecniche che permettano di individuare la maggior parte degli errori che avvengono durante la trasmissione ed eventualmente di correggerli.

1.3 Canali simmetrici

Il modo in cui vengono riconosciuti e corretti gli errori dipende sostanzialmente dal tipo di canale utilizzato e dai disturbi previsti. Sono particolarmente interessanti i canali discreti privi di memoria, di cui un semplice caso è dato da quelli binari simmetrici.

Definizione 1 (Canale discreto)**.** Dato un insieme finito di simboli $S = \{s_1, \ldots, s_n\}$, si definisce *canale di trasmissione discreto* un dispositivo o un sistema fisico Γ che, per ogni elemento $s_i \in S$ in entrata, restituisce in uscita un altro elemento $s_j \in S$ non necessariamente uguale.

Trasmettendo una sequenza ordinata di simboli $(s_{i_1}, \ldots, s_{i_k})$, detta *parola*, si riceve la sequenza $(s_{j_1}, \ldots, s_{j_k})$. Il numero di simboli corrispondenti diversi tra le due parole è detto *numero di errori* commesso durante la trasmissione della parola di partenza.

Definizione 2 (Canale privo di memoria)**.** Siano s_i e s_j due simboli di S e sia $P(s_i, s_j)$ la probabilità che inviando s_i si riceva s_j. Il canale Γ si dice *privo di memoria* se P dipende solo dalla coppia (s_i, s_j).

Al canale Γ è possibile associare una matrice $P = (p_{ij})$ ponendo

$$p_{ij} = P(s_i, s_j) \qquad \forall s_i, s_j \in S$$

Ogni riga di questa matrice è formata dalle probabilità con cui inviando la lettera corrispondente alla riga si riceve ogni possibile lettera. Quindi la somma delle entrate su ogni riga è 1.

Definizione 3 (Canale simmetrico)**.** Il canale Γ si dice *simmetrico* se la probabilità p_{ij} è la stessa per ogni coppia di simboli distinti, cioè

$$p_{ij} = p \qquad \forall i \neq j$$

Il numero p è detto *probabilità di errore del canale.*

Considerando in particolare il caso di molte applicazioni informatiche in cui si utilizzano solamente i due simboli 0 e 1, abbiamo un *canale simmetrico binario* e la sua matrice è:

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

1.4 Ridondanza

Riprendiamo l'esempio del paragrafo 1.2. Inviando solamente il dato di nostro interesse questo rischia di essere compromesso durante la trasmissione e, una volta giunto a destinazione, non è possibile sapere con certezza se è stato ricevuto correttamente. Occorre quindi inserire una ridondanza, cioè delle informazioni aggiuntive che permettono di controllare la correttezza del messaggio ricevuto.

Nell'esempio precedente al posto di inviare solamente il simbolo 1 o 0 per comunicare l'esito dell'esperimento si può inviare la coppia 11 per l'esito positivo e la coppia 00 per quello negativo.

In questo modo diminuisce la probabilità di fraintendere l'esito. Infatti, se la probabilità di ricevere un simbolo sbagliato, cioè di avere l'esito opposto, era p, ora questa probabilità diventa p^2 perché devono avvenire due errori nella stessa coppia.
Rispetto al caso iniziale in cui non si riconosceva nessun errore, è ora possibile capire che ne è avvenuto uno nella trasmissione di una coppia quando si riceve 10 oppure 01. Non siamo ancora, però, in grado di correggerlo perché non sappiamo se è avvenuto nel primo o nel secondo simbolo, cioè non possiamo sapere se era stato inviato 00 oppure 11.

Aumentando ancora la ridondanza e trasmettendo le terne 111 oppure 000, diminuiamo ulteriormente la probabilità di fraintendimento, riuscendo a individuare e correggere un numero maggiore di errori. Infatti, ricevendo la terna 100 è più probabile che sia avvenuto un solo errore nel primo simbolo e che quindi fosse stata trasmessa la terna 000 invece di 111.

Quindi, maggiore è la ridondanza applicata, più errori si riusciranno a riconoscere e correggere. Questo però appesantisce il messaggio e comporta un costo in termini di tempo e risorse. Bisogna dunque trovare il giusto equilibrio tra l'efficienza e la correttezza della trasmissione, che è l'obiettivo della teoria dei codici.

Capitolo 2

Teoria dei codici

2.1 Alfabeti

L'insieme finito di simboli $S = \{s_1, \ldots, s_k\}$, con $k > 1$, è detto *alfabeto* e ogni n-upla ordinata formata dai suoi elementi è una *parola* di lunghezza n su S.
Una parola $\mathbf{p}$ può essere denotata in vari modi:

$$\mathbf{p} = p_1 \ldots p_n, \quad \mathbf{p} = (p_1, \ldots, p_n), \qquad (p_1, \ldots, p_n).$$

Si indica con $\emptyset$ la parola nulla, cioè di lunghezza 0 su S.

A partire da un alfabeto S, si può definire l'insieme di tutte le sue possibili parole:

$$S^* = \bigcup_{i=0}^{\infty} S^i$$

dove S^i è l'insieme contenente le parole di lunghezza i.
Date due parole $\mathbf{p} = p_1 \ldots p_n$ e $\mathbf{q} = q_1 \ldots q_m$ di S^*, con $n, m > 1$, la loro *somma* o *concatenazione* è la parola di lunghezza $n + m$:

$$\mathbf{p} * \mathbf{q} = p_1 \ldots p_n q_1 \ldots q_m$$

L'alfabeto più piccolo possibile è quello formato solamente da due simboli, generalmente 0 e 1. È detto *binario* ed è molto importante nelle applicazioni informatiche.

Definizione 4 (Codice)**.** Si definisce *codice* qualsiasi insieme finito $\mathcal{C}$ di parole definite sullo stesso alfabeto S.
La sua cardinalità $|\mathcal{C}|$ si dice *grandezza* del codice.

A priori, questa definizione non dà nessuna ipotesi né sulla struttura di $\mathcal{C}$ né sulla lunghezza delle sue parole. Sono però di particolare importanza i codici che possiedono una struttura algebrica come i codici lineari che verranno trattati in seguito.

Se l'alfabeto coincide con il campo finito $\mathbb{F}_q$ di q elementi, il codice $\mathcal{C}$ si dice *q-ario*. In particolare, tutti i codici definiti su $\mathbb{F}_2$ sono codici binari.

Nelle applicazioni può essere necessario passare da un codice a uno diverso, per esempio, passare dal codice utilizzato nella lingua parlata al codice binario dei dispositivi digitali e viceversa. Definiamo quindi le funzioni di codifica e di decodifica.

Definizione 5. Dati due codici $\mathcal{C}$ e $\mathcal{C}'$, si dice *funzione di codifica* da $\mathcal{C}$ in $\mathcal{C}'$ ogni applicazione iniettiva $\varphi : \mathcal{C} \mapsto \mathcal{C}'$.
Si dice *funzione di decodifica* per $\mathcal{C}'$ ogni inversa sinistra $\psi : \mathcal{C}' \mapsto \mathcal{C}$ di φ.

Una comunicazione efficace non dipende solamente dalla scelta del canale e dal modo in cui vengono gestiti gli errori, ma sono molto importanti anche le caratteristiche del codice utilizzato per trasmettere il messaggio. Supponiamo, ad esempio, di essere nel caso ideale senza rumore. Se dopo aver trasmesso una determinata parola quest'ultima può essere decodificata in più modi, c'è il rischio che i due interlocutori si fraintendano rendendo inutile la comunicazione. Bisogna quindi utilizzare dei codici *univocamente decodificabili*.

Definizione 6. Dato un codice $\mathcal{C}$, poniamo $\mathcal{C}^1 = \mathcal{C}$ e definiamo

$$\mathcal{C}^k = \{\mathbf{a} * \mathbf{b} : \mathbf{a} \in \mathcal{C}^{k-1}, \mathbf{b} \in \mathcal{C}\} \qquad \forall k > 1$$

$\mathcal{C}$ è *univocamente decodificabile* se, per $k \geq 1$, ogni elemento di $\mathcal{C}^k$ può essere scritto in modo unico come concatenazione di k parole di $\mathcal{C}$.
In particolare, se

$$\boldsymbol{\gamma_1} * \cdots * \boldsymbol{\gamma_k} = \boldsymbol{\delta_1} * \cdots * \boldsymbol{\delta_k} \in \mathcal{C}^k$$

allora

$$\boldsymbol{\gamma_1} = \boldsymbol{\delta_1}, \ldots, \boldsymbol{\gamma_k} = \boldsymbol{\delta_k}$$

Esempio 1 (Codice Morse)**.** Il **codice Morse** è un codice formato da parole di lunghezza variabile definite su un alfabeto di tre simboli: punto (•), linea (−) e spazio (⋄). Fu ideato nel 1835 dall'inventore americano Samuel Morse per la trasmissione di messaggi con il telegrafo. In questo tipo di canale possono viaggiare solo segnali elettrici. Una volta definita l'unità fondamentale di tempo, il punto corrisponde ad un'unità di tempo di trasmissione del segnale, la linea corrisponde ad un segnale continuo di durata tripla rispetto al punto e lo spazio corrisponde a una assenza di segnale di durata pari a quella della linea.

A •−	B −•••	C −•−•	D −••	E •	F ••−•	G −−•	H ••••	I ••
J •−−−	K −•−	L •−••	M −−	N −•	O −−−	P •−−•	Q −−•−	R •−•
S •••	T −	U ••−	V •••−	W •−−	X −••−	Y −•−−	Z −−••	

Figura 2.1: Codice Morse

Come si vede dalla tabella di Figura 2.1, nella codifica delle lettere dell'alfabeto vengono usati solo i simboli linea e punto ma non lo spazio che viene riservato come carattere separatore tra due lettere consecutive. Se non fosse previsto un carattere separatore, il codice non sarebbe univocamente decodificabile, ad esempio:

$$N = - \bullet = TE$$

Per decodificare in modo univoco il messaggio bisogna prima identificare i separatori e poi tradurre i simboli che formano ogni lettera:

$$N = - \bullet \neq - \diamond \bullet = TE$$

2.2 Codici a blocchi

Nella maggior parte delle applicazioni non si utilizzano codici qualsiasi ma si prediligono quelli di più facile implementazione e che possono dare un vantaggio in termini di risorse impiegate. Sotto questo punto di vista sono interessanti i *codici a blocchi*: le informazioni da trasmettere vengono suddivise in un numero discreto di blocchi (o pacchetti), ciascuno dei quali è formato da un numero n fissato di simboli. In questo modo ogni blocco può essere decodificato indipendentemente dai precedenti o dai successivi e un singolo blocco danneggiato non compromette il resto del messaggio. Inoltre, per descrivere il procedimento di codifica e decodifica del codice, è sufficiente farlo per un solo blocco.

Diamo ora una definizione più rigorosa [2, p.12].

Definizione 7 (Codici a blocchi)**.** Dato un alfabeto A_q di $q > 1$ simboli, si dice *codice a blocchi di lunghezza n ed ordine q* (oppure *q-ario*) un qualunque sottoinsieme non vuoto $\mathcal{C}$ di A_q^n.
Se $|A_q^n| = M$, cioè se A_q^n contiene M parole di lunghezza n, $\mathcal{C}$ si indica con $(n, M) - codice$.

Dimostriamo il seguente teorema [3, Cap. 2, p.26]

Teorema 8. *Ogni codice a blocchi $\mathcal{C}$ è univocamente decodificabile.*

Dimostrazione. Sia $\mathcal{C}$ un $(n, M) - codice$, dunque ogni sua parola ha lunghezza n. In particolare, per ogni k abbiamo che ogni $\mathbf{a} \in \mathcal{C}^k$ si può scrivere come

$$\mathbf{a} = \underbrace{a_1 \dots a_n}_{\sigma_1} * \underbrace{a_{n+1} \dots a_{2n}}_{\sigma_2} * \cdots * \underbrace{a_{(k-1)n+1} \dots a_{kn}}_{\sigma_k}$$

Dalla definizione di $\mathcal{C}^k$ e dal fatto che tutte le parole di $\mathcal{C}$ hanno lunghezza n segue che tutti i $\sigma_i \in \mathcal{C}$ sono univocamente determinati. Quindi $\mathcal{C}$ è univocamente decodificabile.

□

2.3 Codici rivelatori e correttori

Uno tra i primi a studiare le tecniche di correzione degli errori fu il matematico americano Richard Hamming che, nel 1950, pubblicò un articolo ispirato dai computer di "Bell Telephone Laboratory" [4]. Questi calcolatori dovevano svolgere un numero elevato di operazioni per ottenere un risultato e, nel caso fosse stato rilevato un errore, il lavoro veniva interrotto finché questo non veniva individuato e corretto manualmente. Era quindi necessario trovare un metodo che permettesse ai computer di individuare e correggere automaticamente gli errori senza fermare il processo.

Nel suo articolo Hamming utilizza un codice binario a blocchi di lunghezza n in cui m simboli sono associati all'informazione da trasmettere. I restanti $k = n - m$, invece, sono indispensabili per il riconoscimento e la correzione degli errori, anche se riducono la capacità di trasmissione del canale. Egli definisce *ridondanza* il rapporto

$$R = \frac{n}{m}.$$

Esempio 2 (Codice rivelatore)**.** Un esempio di codice rivelatore è il **controllo di parità** che può individuare un singolo errore: rimanendo nelle ipotesi di codice binario a blocchi di lunghezza n, associamo ai primi $n-1$ simboli della parola l'informazione e nell'ultimo inseriamo uno 0 oppure un 1 in modo che, nell'intera parola, il simbolo 1 compaia un numero pari di volte. Se la parola ricevuta presenta un numero dispari di simboli 1, significa che si è presentato un errore (o un numero dispari di errori).
In questo tipo di codice la ridondanza è

$$R = \frac{n}{n-1} = 1 + \frac{1}{n-1}.$$

Per aumentare la capacità di trasmissione del canale bisogna diminuire la ridondanza R, cioè occorre aumentare n. In questo modo, però, aumenta la probabilità che si verifichino più errori nella stessa parola e, se questi avvengono in numero pari, non vengono rivelati.

Esempio 3 (Codice correttore)**.** Un esempio di codice correttore è il *codice* $(7, 4)$ *di Hamming* [2, p.13]. Utilizziamo un codice binario a blocchi di lunghezza 7 per trasmettere delle informazioni composte da 4 cifre, le restanti 3 servono per l'individuazione e la correzione degli errori: se (a_1, a_2, a_3, a_4) è la quaterna contente l'informazione, inviamo invece la 7-upla

$$(a_1, a_2, a_3, a_4, a_1 + a_3 + a_4, a_1 + a_2 + a_4, a_1 + a_2 + a_3).$$

Poiché stiamo lavorando sul campo $\mathbb{F}_2$, ogni parola del codice $(x_1, \ldots, x_7)$ soddisfa il seguente sistema

$$\begin{cases} x_1 + x_3 + x_4 + x_5 = 0 \\ x_1 + x_2 + x_4 + x_6 = 0 \\ x_1 + x_2 + x_3 + x_7 = 0 \end{cases}$$

detto *sistema delle condizioni di parità del codice*. Infatti, per come abbiamo definito le ultime tre entrate della $7-upla$, abbiamo $x_5 = x_1 + x_3 + x_4$. Dunque la prima equazione risulta $2x_1 + 2x_3 + 2x_4 = 0$ ed è verificata. Analogamente sono soddisfatte le altre due equazioni.
Questo sistema ci permette di rilevare la presenza di errori in quanto almeno un'equazione risulterebbe non soddisfatta. Se possiamo inoltre supporre che durante la trasmissione avvenga solo un errore, siamo anche in grado di individuarne la posizione e correggerlo. In caso di un solo errore le possibilità sono le seguenti:

- nessuna equazione verificata: errore al **primo** posto;
- solo la prima equazione verificata: errore al **secondo** posto;
- solo la seconda equazione verificata: errore al **terzo** posto;
- solo la terza equazione verificata: errore al **quarto** posto;
- solo la prima equazione non verificata: errore al **quinto** posto;
- solo la seconda equazione non verificata: errore al **sesto** posto;
- solo la terza equazione non verificata: errore al **settimo** posto.

Una volta individuata la posizione dell'errore è sufficiente sostituirvi 1 a 0 o viceversa.

Siamo interessati a capire come costruire un codice correttore che possa anche gestire la presenza di più errori nella stessa parola. A questo scopo è importante trovare un modo per quantificare il grado di somiglianza tra due parole della stessa lunghezza, in quanto ogni errore modifica un simbolo della sequenza e, più errori avvengono, più risultano diverse tra loro la parola inviata e quella ricevuta.
Hamming, nel suo articolo, utilizza un approccio geometrico identificando le parole del codice (tutte di lunghezza n) con i vertici di un cubo unitario n-dimensionale. In questo spazio di 2^n punti è possibile definire una metrica.

Definizione 9. Date due parole $\mathbf{x}$ e $\mathbf{y}$ definite sullo stesso codice, si dice *distanza di Hamming* il numero di coordinate per le quali $\mathbf{x}$ e $\mathbf{y}$ differiscono:

$$d(\mathbf{x}, \mathbf{y}) = |\{i : x_i \neq y_i\}|$$

La distanza di Hamming è una metrica, infatti:

1. $d(\mathbf{x}, \mathbf{y}) = 0$ se e solo se $\mathbf{x}$ e $\mathbf{y}$ non differiscono per nessuna coordinata, cioè se e solo se $\mathbf{x} = \mathbf{y}$;
2. $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ segue dalla definizione precedente;

3. vale la disuguaglianza triangolare $d(\mathbf{x},\mathbf{z}) \leq d(\mathbf{x},\mathbf{y}) + d(\mathbf{y},\mathbf{z})$ in quanto
   $d(\mathbf{x},\mathbf{z}) = |\{i : x_i \neq z_i\}|$
   $d(\mathbf{x},\mathbf{y}) + d(\mathbf{y},\mathbf{z}) = |\{i : x_i \neq y_i\}| + |\{i : y_i \neq z_i\}|$
   ma se $x_i \neq z_i$, allora $x_i \neq y_i$ oppure $y_i \neq z_i$. Quindi per ogni entrata diversa tra $\mathbf{x}$ e $\mathbf{z}$, ce n'è almeno una diversa tra $\mathbf{x}$ e $\mathbf{y}$ oppure tra $\mathbf{y}$ e $\mathbf{z}$.

Definizione 10. Sia $\mathcal{C}$ un codice a blocchi di lunghezza n sull'alfabeto A_q e sia $\mathbf{x} \in \mathcal{C}$. La *sfera di Hamming* di centro $\mathbf{x}$ e raggio r è

$$B_r(\mathbf{x}) = \{\mathbf{y} \in A_q^n : d(\mathbf{x},\mathbf{y}) \leq r\}.$$

Definizione 11. Si dice *distanza minima* del codice $\mathcal{C}$ il minimo delle distanze tra due qualsiasi parole distinte di $\mathcal{C}$:

$$d(\mathcal{C}) = min\{d(\mathbf{x},\mathbf{y}) : \mathbf{x},\mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}\}.$$

Definizione 12. Il codice $\mathcal{C}$ si dice $(n, M, d)_q$-*codice* se:

- q sono i simboli che compongono l'alfabeto;
- n é la lunghezza dei blocchi;
- $M = |\mathcal{C}|$;
- d é la distanza minima.

I valori q, n, M, d si dicono *parametri del codice.*

Esempio 4. Consideriamo un codice di lunghezza 3, quindi un cubo tridimensionale. Ogni vertice di questo cubo è associato a una parola del codice:

$$(0\,0\,0) \quad (0\,0\,1) \quad (0\,1\,0) \quad (0\,1\,1)$$
$$(1\,0\,0) \quad (1\,0\,1) \quad (1\,1\,0) \quad (1\,1\,1)$$

Se consideriamo la parola $(1\,1\,1)$ come il centro di una sfera di raggio 2, quest'ultima contiene tutte le altre tranne $(0\,0\,0)$, che si trova a distanza 3 dal centro. In particolare, si trovano sulla superficie della sfera le terne $(0\,0\,1)$, $(0\,1\,0)$, $(1\,0\,0)$.
La distanza minima di questo codice è 1.
Se utilizziamo, invece, il codice formato solamente dalle parole $(0\,0\,1)$, $(0\,1\,0)$, $(1\,0\,0)$, $(1\,1\,1)$, la sua distanza minima è 2.

Se tutte le parole di un codice si trovano a distanza maggiore o uguale a 2 l'una dall'altra, significa che, presa una qualsiasi coppia di parole, queste differiscono tra loro per almeno due coordinate. Dunque, quando avviene un singolo errore durante la trasmissione, si parte da una sequenza di simboli appartenente al codice e se ne riceve una senza significato, poiché cambiando una sola coordinata si ottiene una parola che non appartiene al codice.

Se la distanza minima di un codice è almeno 3, allora ogni singolo errore può essere corretto, in quanto partendo da una parola del codice e cambiando una sola coordinata, si ottiene una parola senza significato che, nel cubo n-dimensionale, si trova più vicina alla parola iniziale che non a tutte le altre. Procedendo con il ragionamento si può capire per ogni distanza minima quanti errori un codice è in grado di individuare e correggere.

A questo punto il problema di trovare un codice adatto alle proprie esigenze si traduce nel trovare un sottoinsieme di punti nel giusto spazio che rispettino le condizioni sulla distanza minima riassunte nella Tabella 2.1 [4, p. 155].

Distanza minima	**Errori corretti**	**Errori rilevati**
1	0	0
2	0	1
3	1	2
4	1	3
5	2	4
...	...	...

Tabella 2.1: Significato della distanza minima

Per concludere la sezione sui codici rivelatori e correttori, enunciamo e dimostriamo alcuni teoremi che mettono in relazione la distanza minima del codice e il numero di errori che possono essere gestiti [2, p. 16-17].

Definizione 13. Un codice $\mathcal{C}$ si dice *h-correttore*, risp. *h-rivelatore*, se h è il numero massimo di errori che $\mathcal{C}$ può correggere, risp. rivelare.

Teorema 14. *Il codice $\mathcal{C}$ è h-rivelatore se e solo se la sua distanza minima è $d = h + 1$.*

Dimostrazione. Sia $d = h+1$ la distanza minima del codice e sia $\mathbf{x}$ la parola che viene trasmessa. Se durante la trasmissione avviene un numero $t \leq h$ di errori, si riceve la sequenza di simboli $\mathbf{y}$ che non è una parola del codice, in quanto $d(\mathbf{x}, \mathbf{y}) = t < d$. Quindi possono essere rivelati t errori.
Se invece il numero di errori è maggiore o uguale a $d = h + 1$, la sequenza di simboli ricevuta può essere un'altra parola del codice e gli errori possono passare inosservati.
Quindi possono essere rivelati al più h errori con $h = d - 1$.

Viceversa, sia $\mathcal{C}$ h-rivelatore. Poiché vengono rivelati h errori, due diverse parole di $\mathcal{C}$ devono differire per almeno $h + 1$ coordinate. Quindi $d \geq h + 1$. Se fosse $d > h + 1$, allora $\mathcal{C}$ sarebbe $(h + 1)$-correttore, assurdo. Dunque, per $\mathcal{C}$ h-correttore la distanza minima è $d = h + 1$. □

Teorema 15. *Un codice $\mathcal{C}$ con distanza minima d è $\lfloor \frac{d-1}{2} \rfloor$-correttore, con $\lfloor \frac{d-1}{2} \rfloor$ parte intera di $\frac{d-1}{2}$.*

Dimostrazione. Sia d la distanza minima del codice e sia $\mathbf{x}$ la parola da trasmettere. Supponiamo che durante la trasmissione avvenga un numero di errori $t \leq \lfloor \frac{d-1}{2} \rfloor$, riceviamo dunque la parola $\mathbf{y}$. Abbiamo che $d(\mathbf{x}, \mathbf{y}) \leq \lfloor \frac{d-1}{2} \rfloor$.
Proviamo che per ogni altra parola del codice $\mathbf{z}$ si ha $d(\mathbf{y}, \mathbf{z}) \geq \lfloor \frac{d-1}{2} \rfloor + 1$, cioè $\mathbf{x}$ è la parola più vicina a $\mathbf{y}$ e possiamo correggere l'errore.
Supponiamo per assurdo che esista $\mathbf{z} \in \mathcal{C}$ tale che $\mathbf{z} \neq \mathbf{x}$ e $d(\mathbf{y}, \mathbf{z}) \leq \lfloor \frac{d-1}{2} \rfloor$.
Per la disuguaglianza triangolare $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{y}, \mathbf{x}) + d(\mathbf{y}, \mathbf{z}) \leq 2\lfloor \frac{d-1}{2} \rfloor \leq d-1$.
Quindi $d(\mathbf{x}, \mathbf{z}) \leq d - 1$, assurdo perché $d(\mathcal{C}) = d$. □

2.4 Efficienza di un codice

Non tutti i codici sono utili per le applicazioni: in base alle esigenze del caso si sceglieranno quelli meno costosi a livello computazionale piuttosto che quelli più pesanti ma in grado di correggere un maggior numero di errori. Per capire quanto un codice è "buono" si utilizzano i suoi parametri.

Consideriamo un $(n, M, d)_q$-codice.
La lunghezza n rappresenta il costo del codice: più sono lunghi i blocchi, più impegnativa sarà la gestione della trasmissione a livello di tempo, energia e denaro.
La grandezza M rappresenta la ricchezza del codice: più parole sono permesse dal codice, maggiore è la qualità dell'informazione che può essere trasmessa.
Questi due parametri possono essere combinati nel seguente modo:

$$R = \frac{\log_q M}{n}.$$

La quantità R è detta *efficienza* del codice e rappresenta il suo rapporto "qualità/prezzo". Se si vuole un codice poco costoso è necessario che R sia il più possibile vicino a 1, mentre se si vuole un codice efficiente in grado si scoprire e correggere molti errori, questa quantità deve diventare più grande.
Un altro valore che misura l'adeguatezza del codice è il *rapporto di separazione* δ:

$$\delta = \frac{d}{n}.$$

Maggiore è δ, più errori possono essere rilevati e corretti.

Per costruire il codice più adatto, bisogna dunque modificare i suoi parametri in modo che vengano rispettate le esigenze dell'applicazione di interesse. Tuttavia, i parametri non variano in modo indipendente l'uno dall'altro, ma ci sono delle limitazioni, i cui dettagli di dimostrazione si trovano in [2, p. 22-24]

- **Stima di Singleton**: $M \leq q^{n-d+1}$

- **Stima di Hamming**: sia $\mathcal{C}$ h-correttore, allora $M \leq \frac{q^n}{\sum_{i=0}^{h} \binom{n}{i}(q-1)^i}$

Un codice per cui vale l'uguaglianza nella stima di Hamming si dice *perfetto*.

2.5 Codici equivalenti

Dopo aver visto che la coppia (A_q^n, d) è uno spazio metrico, è interessante capire quali codici definiti sullo stesso insieme A_q^n godono delle stesse proprietà metriche. Diamo dunque le seguenti definizioni [3, par. 3.7].

Definizione 16 (Isometria)**.** Sia A un alfabeto e $n \geq 1$ un intero. L'applicazione biiettiva $\varphi : A^n \to A^n$ è detta *isometria* se

$$d(\varphi(\mathbf{l}), \varphi(\mathbf{m})) = d(\mathbf{l}, \mathbf{m}) \qquad \forall \mathbf{l}, \mathbf{m} \in A^n$$

Definizione 17 (Codici equivalenti)**.** I codici $\mathcal{C}_1$, $\mathcal{C}_2 \subseteq A_q^n$ si dicono *equivalenti* se è possibile ottenere $\mathcal{C}_2$ applicando alle parole di $\mathcal{C}_1$ un numero finito di volte le seguenti operazioni:

1. permutare le posizioni dei simboli di tutte le parole di $\mathcal{C}_1$;
2. permutare i caratteri in ogni singola posizione di $\mathcal{C}_1$.

Il seguente esempio chiarisce come agiscono le operazioni della Definizione 17.

Esempio 5. Consideriamo il codice $\mathcal{C}_1$ definito sull'alfabeto $\{a, b, c\}$ e composto dalle parole

$$(a\,a\,a), \quad (a\,b\,c), \quad (c\,b\,b).$$

Permutando in ogni parola le prime due coordinate (operazione 1) si ottiene il codice equivalente $\mathcal{C}_2$ composto dalle parole

$$(a\,a\,a), \quad (b\,a\,c), \quad (b\,c\,b).$$

Applicando, invece, la permutazione $(a\,b\,c)$ alla prima coordinata di tutte le parole di $\mathcal{C}_1$ (operazione 2) otteniamo il codice equivalente $\mathcal{C}_3$ composto dalle parole

$$(b\,a\,a), \quad (b\,b\,c), \quad (a\,b\,b).$$

Se applichiamo alle parole di $\mathcal{C}_1$ entrambe le operazioni eseguite sopra nello stesso ordine otteniamo il codice equivalente $\mathcal{C}_4$ composto dalle parole

$$(b\,a\,a), \quad (c\,a\,c), \quad (c\,c\,b).$$

Le due operazioni descritte nella Definizione 16 sono isometrie.
Infatti, applicando la stessa permutazione σ a tutte le parole, la distanza tra due parole rimane invariata: se $\mathbf{x}, \mathbf{y} \in \mathcal{C}$, allora $d(\mathbf{x}, \mathbf{y}) = |\{i : x_i \neq y_i\}|$ e $d(\sigma(\mathbf{x}), \sigma(\mathbf{y})) = |\{i : x_{\sigma(i)} \neq y_{\sigma(i)}\}|$. Se $x_i = y_i$, allora anche $x_{\sigma(i)} = y_{\sigma(i)}$ e se $x_i \neq y_i$ anche $x_{\sigma(i)} \neq y_{\sigma(i)}$. Dunque la prima operazione rispetta la distanza in quanto $d(\mathbf{x}, \mathbf{y}) = d(\sigma(\mathbf{x}), \sigma(\mathbf{y}))$.
Invece, applicando la permutazione ω ai simboli dell'i-esima coordinata in tutte le parole, se $x_i = y_i$ allora $x_{\omega(i)} = y_{\omega(i)}$ e se $x_i \neq y_i$ allora $x_{\omega(i)} \neq y_{\omega(i)}$. Quindi anche applicando la seconda operazione, la distanza tra le parole rimane invariata.
In conclusione, entrambe le operazioni definite sopra rispettano la distanza tra le parole del codice e sono dunque isometrie. Componendole un numero finito di volte si ottiene sempre un'isometria perché la distanza rimane comunque invariata. Ne segue che due codici equivalenti sono in corrispondenza tra loro tramite un'isometria e godono delle stesse proprietà metriche come la distanza minima. Inoltre, sono composti dallo stesso numero di parole, dunque sono uguali anche ridondanza ed efficienza.

Capitolo 3

Codici lineari

3.1 Motivazioni

Abbiamo visto nel capitolo precedente che la Definizione 4 non dà nessuna ipotesi sulla struttura algebrica di un codice, il quale può quindi essere un insieme qualsiasi di parole. Non tutti i codici sono equivalenti e per le applicazioni è importante valutarne i parametri per scegliere quello più adatto. Una volta scelto il codice da utilizzare, quest'ultimo deve essere implementato attraverso i seguenti algoritmi:

1. uno per la codifica;
2. uno per la rivelazione e l'eventuale correzione degli errori;
3. uno per la decodifica.

L'efficienza di questi algoritmi dipende sostanzialmente dalle caratteristiche del codice. Se esso non possiede una qualche struttura algebrica, in fase di implementazione sarà necessario descrivere per esteso l'insieme delle parole e la loro corrispondenza con i messaggi da trasmettere: in questo modo gli algoritmi saranno più complessi e costosi, soprattutto per codici di grandi dimensioni. Utilizzando, invece, codici a blocchi dotati anche della struttura di spazio vettoriale, vengono semplificate molte operazioni come codifica, decodifica, trasmissione dei messaggi e calcolo dei parametri del codice. Dunque, gli algoritmi elencati sopra saranno più semplici, meno costosi e più efficienti.

3.2 Definizione e proprietà

D'ora in avanti supporremo che i codici siano definiti su un campo finito di q elementi, cioè l'alfabeto sia $A = \mathbb{F}_q$. Se $\mathcal{C}$ è un codice a blocchi di lunghezza n, possiamo considerare le sue parole (tutte n-uple) come elementi dello spazio vettoriale $V_n(\mathbb{F}_q)$. Il caso più interessante si verifica quando $\mathcal{C}$ è a sua volta uno spazio vettoriale [7, par. 3.2]:

Definizione 18. Il codice $\mathcal{C}$, definito sull'alfabeto $\mathbb{F}_q$, è un codice *lineare* se è un sottospazio vettoriale di $V_n(\mathbb{F}_q)$.
Se la dimensione di $\mathcal{C}$ è k, allora diciamo che $\mathcal{C}$ è un $[n,k]$-codice.

Notazione. Con $[n,k,d]$-codice indichiamo un codice lineare k-dimensionale di lunghezza n e distanza minima d.
Con (n,M,d)-codice indichiamo, invece, un qualsiasi codice di M parole di lunghezza n e distanza minima d.

In particolare, per i codici lineari abbiamo che $|\mathcal{C}| = M = q^k$. Inoltre, $V_n(\mathbb{F}_q)$ ha s_k sottospazi k-dimensionali, dove

$$s_k = \frac{(q^n-1)(q^{n-1}-1)\dots(q^{n-k+1}-1)}{(q^k-1)(q^{k-1}-1)\dots(q-1)}.$$

Dunque è possibile costruire esattamente s_k $[n,k]$-codici lineari diversi definiti sull'alfabeto A.

Come detto sopra, i codici lineari semplificano diverse operazioni, tra cui la codifica. Se anche l'insieme di tutti i possibili messaggi che si possono inviare è uno spazio vettoriale, come funzione di codifica φ si può utilizzare un omomorfismo di spazi vettoriali: in questo modo è sufficiente descriverne l'azione solo sugli elementi di una base dello spazio di partenza.
Vediamo un esempio [3, par. 4.1].

Esempio 6. Vogliamo costruire un codice $\mathcal{C}$ che sia un $[5,2]$-codice, non ci occupiamo ora di scegliere la distanza minima. Dunque $\mathcal{C}$ deve essere un sottospazio 2-dimensionale di $V_5(\mathbb{F}_2)$ e una sua base può essere:

$$\mathfrak{B} = \{(1\,0\,1\,1\,1),\ (1\,1\,1\,1\,0)\}.$$

Supponiamo che $K = \{(0\,0),(0\,1),(1\,0),(1\,1)\}$ sia l'insieme di tutti i possibili messaggi. Abbiamo che $K = V_2(\mathbb{F}_2)$ è uno spazio vettoriale. Possiamo scegliere come base $\{(1\,0),(0\,1)\}$ e definire come funzione di codifica dei messaggi l'applicazione lineare iniettiva

$$\begin{aligned} \varphi\colon V_2(\mathbb{F}_2) &\to V_5(\mathbb{F}_2) \\ (1\,0) &\mapsto (1\,0\,1\,1\,1) \\ (0\,1) &\mapsto (1\,1\,1\,1\,0) \end{aligned}$$

Per gli altri due elementi di K, dalla linearità di φ, abbiamo:

- $\varphi((0\,0)) = \varphi((1\,0)) + \varphi((1\,0)) = (0\,0\,0\,0\,0)$
- $\varphi((1\,1)) = \varphi((1\,0)) + \varphi((0\,1)) = (0\,1\,0\,0\,1)$

Il codice cercato è quindi

$$\mathcal{C} = \{(0\,0\,0\,0\,0),\ (0\,1\,0\,0\,1),\ (1\,1\,1\,1\,0),\ (1\,0\,1\,1\,1)\}.$$

In generale, per i codici lineari la funzione di codifica è un monomorfismo $\varphi\colon V_k(\mathbb{F}_q) \to V_n(\mathbb{F}_q)$, cioè un omomorfismo iniettivo di spazi vettoriali. Ciò significa che possiamo facilmente descrivere la trasformazione attraverso un'opportuna matrice. Nell'esempio precedente, però, abbiamo scelto arbitrariamente le basi degli spazi vettoriali per costruire il codice lineare e la funzione di codifica. Scegliendo basi diverse che non generano lo stesso sottospazio, si ottengono codici diversi che non hanno necessariamente le stesse proprietà. Vediamolo nel seguente esempio [2, p. 34].

Esempio 7. Costruiamo, partendo da basi diverse, due $[3, 2]$-codici. Siano $\mathfrak{B}_1$ e $\mathfrak{B}_2$ le basi, rispettivamente, di $\mathcal{C}_1$ e $\mathcal{C}_2$:

$$\mathfrak{B}_1 = \{(1\,0\,0),\ (0\,1\,0)\}$$
$$\mathfrak{B}_2 = \{(1\,1\,0),\ (1\,0\,1)\}$$

I codici sono dunque:

$$\mathcal{C}_1 = \{(0\,0\,0),\ (1\,0\,0),\ (0\,1\,0),\ (1\,1\,0)\}$$
$$\mathcal{C}_2 = \{(0\,0\,0),\ (1\,1\,0),\ (1\,0\,1),\ (0\,1\,1)\}$$

Questi codici sono entrambi sottospazi vettoriali di $V_3(\mathbb{F}_2)$, ma non sono equivalenti come codici: una verifica diretta mostra che la distanza minima di $\mathcal{C}_1$ è 1, mentre quella di $\mathcal{C}_2$ è 2.

In conclusione, codici lineari ottenuti a partire da basi diverse sono isomorfi come spazi vettoriali poiché hanno la stessa dimensione, ma non possiedono le stesse proprietà metriche e non sono dunque equivalenti.

3.3 Peso di Hamming

Come abbiamo visto nel Capitolo 2, un parametro importante per i codici è la distanza minima che ci permette di capire quanti errori è in grado di rilevare o correggere il codice utilizzato. Dato un codice qualunque, per calcolare la sua distanza minima è necessario confrontare a due a due tutte le parole. Se per alcune applicazioni è utile avere codici in grado di trasmettere un gran numero di messaggi, è evidente che più aumenta la dimensione del codice, più questa operazione diventa lunga e dispendiosa: infatti, devono essere eseguiti $\binom{M}{2}$ confronti, dove $M = |\mathcal{C}|$. Sui codici lineari è possibile usare un metodo alternativo per calcolare questo parametro che ci permette di effettuare un minor numero di confronti.

Definizione 19 (Peso di Hamming)**.** Dato $\mathbf{x} \in F_q^n$, il suo *peso di Hamming* è la quantità

$$w(\mathbf{x}) = |\{i : x_i \neq 0\}|.$$

Cioè il peso di Hamming di una parola $\mathbf{x}$ è il numero di componenti non nulle di $\mathbf{x}$.

Con il seguente teorema mettiamo in relazione la distanza minima di un codice e il peso di Hamming calcolato sulle sue parole [2, p.34].

Teorema 20. *Sia $\mathcal{C}$ un codice lineare con distanza minima d. Allora:*

1. $d(\boldsymbol{x},\boldsymbol{y}) = w(\boldsymbol{x}-\boldsymbol{y}), \quad \forall\, \boldsymbol{x},\boldsymbol{y} \in \mathcal{C}$;
2. *d è uguale al minimo peso delle parole non nulle di $\mathcal{C}$.*

Dimostrazione. (1) Siano $\mathbf{x},\mathbf{y} \in \mathcal{C}$. La distanza $d(\mathbf{x},\mathbf{y})$ corrisponde al numero di coordinate per cui $\mathbf{x}$ e $\mathbf{y}$ differiscono, quindi corrisponde al numero di coordinate non nulle della parola $\mathbf{x}-\mathbf{y} \in \mathcal{C}$. Cioè $d(\mathbf{x},\mathbf{y}) = w(\mathbf{x}-\mathbf{y})$.
(2) Sia d la distanza minima di $\mathcal{C}$. Esistono $\mathbf{x},\mathbf{y} \in \mathcal{C}$ tali che $d(\mathbf{x},\mathbf{y}) = d$. Per il punto precedente $d(\mathbf{x},\mathbf{y}) = w(\mathbf{x}-\mathbf{y})$, quindi il peso minimo è d oppure un intero minore di d.
Se il peso minimo fosse minore di d, esisterebbe $\mathbf{z} \in \mathcal{C}$ tale che $w(\mathbf{z}) < d$. Quindi si avrebbe $d(\mathbf{z},\mathbf{0}) = w(\mathbf{z}) < d$ e d non sarebbe la distanza minima, assurdo.
Dunque, il peso minimo è d. □

Il teorema appena dimostrato ci permette di conoscere la distanza minima di un codice lineare attraverso il calcolo del peso di Hamming delle parole non nulle del codice. Quindi, invece di calcolare e confrontare tra loro $\binom{M}{2}$ distanze, possiamo calcolare e confrontare $M-1$ pesi di Hamming.

3.4 Matrice generatrice

Un modo per descrivere le parole di un codice qualunque è quello di elencarle una ad una. Anche in questo caso, se siamo di fronte a un codice di grandi dimensioni, l'operazione è lunga e dispendiosa. Gli elementi di un codice lineare, invece, formano uno spazio vettoriale e possiamo dunque concentrarci solo sugli elementi di una sua base. In particolare, possiamo associare ad un codice lineare una *matrice generatrice* [7, par. 3.2].

Definizione 21 (Matrice generatrice)**.** Una *matrice generatrice* G per un $[n,k]$-codice $\mathcal{C}$ è una matrice $k \times n$ le cui righe formano una base di $\mathcal{C}$.

Poiché la base di un codice non è unica, non lo è neanche la sua matrice generatrice. Partendo da una matrice generatrice di un codice ed eseguendo le seguenti operazioni sulle sue righe si ottiene una matrice diversa che genera lo stesso codice [3, par. 4.4]:

1. moltiplicazione per uno scalare;
2. somma di una riga con il multiplo scalare di un'altra;
3. permutazione.

Eseguire queste operazioni sulle righe della matrice generatrice equivale a eseguirle sui vettori della base del codice, ciò significa che lo spazio vettoriale generato sarà invariato e dunque anche il codice. In generale, i pesi di Hamming delle righe di queste matrici saranno diversi.
Eseguire le stesse operazioni sulle colonne, invece, può produrre matrici le cui righe non sono più una base dello spazio generato dalle righe della matrice di partenza.

Esempio 8. Consideriamo il codice dell'Esempio 6:

$$\mathcal{C} = \{(0\,0\,0\,0\,0),\ (0\,1\,0\,0\,1),\ (1\,1\,1\,1\,0),\ (1\,0\,1\,1\,1)\}$$

La base $\mathfrak{B}$ utilizzata per costruirlo è formata dai vettori:

$$\mathbf{b}_1 = (1\,0\,1\,1\,1), \quad \mathbf{b}_2 = (1\,1\,1\,1\,0).$$

La sua matrice generatrice è dunque:

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

Sommando la prima riga alla seconda, si ottiene la matrice:

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

le cui righe $\mathbf{c}_1 = (1\,0\,1\,1\,1)$ e $\mathbf{c}_2 = (0\,1\,0\,0\,1)$ sono ancora una base $\mathfrak{B}'$ di $\mathcal{C}$.
In particolare i pesi di Hamming dei vettori di $\mathfrak{B}$ e $\mathfrak{B}'$ sono:

$$\begin{aligned} w(\mathbf{b}_1) &= 4, & w(\mathbf{b}_2) &= 4; \\ w(\mathbf{c}_1) &= 4, & w(\mathbf{c}_2) &= 2. \end{aligned}$$

Una forma particolare che può assumere la matrice generatrice è la *forma standard*.

Definizione 22 (Matrice generatrice standard)**.** Siano $\mathcal{C}$ un codice su $\mathbb{F}_q$ e G una sua matrice generatrice. G è detta *matrice generatrice standard* se è nella forma

$$G = (I_k|A),$$

dove I_k è la matrice identica $k \times k$ ed $A \in M_{k\times(n-k)}(\mathbb{F}_q)$.

Un codice che ammette una matrice generatrice standard si dice *sistematico*.

In una matrice generatrice in forma standard è immediato distinguere quali sono le coordinate delle parole del codice che contengono il messaggio effettivamente inviato da quelle che rappresentano la ridondanza: i primi k simboli sono detti *simboli di informazione* e i rimanenti $n - k$ sono detti *simboli di controllo*.

Definizione 23 (Insieme di informazione)**.** Sia $\mathcal{C}$ un $[n, k]$-codice sull'alfabeto A_q. L'insieme di k posizioni $I = \{i_1, \ldots, i_k\}$ si dice *insieme di informazione* se, comunque data una k-upla di simboli $(v_1, \ldots, v_k)$, $v_i \in A_q$, esiste un'unica parola di codice $\mathbf{c}$ tale che $c_{i_1} = v_1, \ldots, c_{i_k} = v_k$.

In generale, per un (n, k)-codice viene fissato un algoritmo che durante la codifica suddivide il messaggio in k-uple ad ognuna delle quali aggiunge $n - k$ simboli di controllo per formare delle n-uple.

Osservazione. $I = \{i_1, \ldots, i_k\}$ è un insieme di informazione per un codice lineare $\mathcal{C}$ se, e solo se, le colonne della matrice generatrice di $\mathcal{C}$ individuate da I sono linearmente indipendenti.
Consideriamo la matrice G' ottenuta dalla matrice generatrice G prendendo le colonne individuate da I. G' è una matrice $k \times k$ e se le colonne sono linearmente indipendenti, allora lo sono anche le righe, le quali formano una base per un codice lineare $\mathcal{C}'$ di lunghezza k. Quindi ogni parola del codice si scrive in modo unico come combinazione lineare di questi vettori. Tornando alla matrice G, per generare una parola $\mathbf{c} \in \mathcal{C}$ che nelle coordinate individuate da I abbia i simboli $\mathbf{v} = (v_1, \ldots, v_k)$, è necessario utilizzare la stessa combinazione lineare di righe che si utilizza per scrivere $\mathbf{v}$ nel codice $\mathcal{C}'$. Poiché la combinazione lineare è unica, lo è anche la parola $\mathbf{c}$. Dunque I è insieme di informazione per $\mathcal{C}$.
Viceversa, sia I un insieme di informazione per $\mathcal{C}$. Allora, data la k-upla $(v_1, \ldots, v_k)$ esiste un'unica parola del codice $\mathbf{c}$ tale che $c_{i_1} = v_1, \ldots, c_{i_k} = v_k$. Consideriamo G' come sopra e supponiamo per assurdo che le sue colonne non siano linearmente indipendenti. Allora non sono linearmente indipendenti neanche le righe. Dunque esiste una combinazione lineare di righe tale che $\mathbf{0} = \sum_{j=1}^{k} \alpha_j \mathbf{r}_j$, dove $\mathbf{r}_j$ è la j-esima riga e α_j sono scalari non tutti nulli. Sia $\mathbf{v} = (v_1, \ldots, v_k) \in \mathcal{C}'$. Allora anche $\mathbf{w} = \mathbf{v} + \sum_{j=1}^{k} \alpha_j \mathbf{r}_j = (v_1, \ldots, v_k)$. Se applichiamo a G le combinazioni lineari di righe utilizzate in G' per ottenere le parole $\mathbf{v}$ e $\mathbf{w}$, otteniamo in $\mathcal{C}$ due parole che hanno i simboli $(v_1, \ldots, v_k)$ nelle coordinate individuate da I, ma non hanno necessariamente tutte le altre coordinate uguali tra loro. Quindi in generale avremmo due parole diverse con il simbolo v_j in posizione i_j per $j = 1, \ldots, k$. Assurdo perché $\mathbf{c}$ è unica. Dunque le colonne di G individuate da I sono linearmente indipendenti.

In particolare, l'insieme di informazione per una matrice in forma standard è $I = \{1, \ldots, k\}$, poiché le prime K colonne sono linearmente indipendenti. Di conseguenza in un codice sistematico si trova una copia del messaggio originale nelle prime k coordinate di ogni parola codificata.

Osservazione. Data una matrice generatrice per un codice lineare, si possono individuare k colonne linearmente indipendenti in più modi. Ad esempio, per la matrice generatrice di un $[3, 2]$-codice

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

possono essere insiemi di informazione sia $\{1, 2\}$ che $\{2, 3\}$. Dunque, la distinzione tra simboli di informazione e simboli di controllo, cioè quali posizioni contengono il messaggio e quali la ridondanza, è "artificiale".

3.5 Equivalenza

Nel capitolo precedente abbiamo visto che più codici a blocchi possono essere equivalenti tra loro e avere dunque le stesse proprietà metriche. Applicando la definizione di codici equivalenti (Definizione 17) ad un codice lineare, si ottiene un codice equivalente che non è necessariamente ancora lineare, come si vede nel seguente esempio [3, par. 4.5].

Esempio 9. Consideriamo il codice lineare $\mathcal{C} = \{(0, 1), (0, 0)\}$. È un $[2, 1]$-codice su $\mathbb{F}_2$ generato dal vettore $(0, 1)$. Se applichiamo alla prima coordinata di tutte le parole la permutazione $(0, 1)$, otteniamo il codice equivalente $\mathcal{C}' = \{(1, 1), (1, 0)\}$ che non è lineare. Infatti la parola $(0, 0) \notin \mathcal{C}'$, ma tutti gli spazi vettoriali contengono il vettore nullo.

Modifichiamo dunque la definizione di codici equivalenti data precedentemente affinché tra codici equivalenti non solo siano rispettate le proprietà metriche, ma anche l'essere lineari [2, p.38].

Definizione 24 (Codici equivalenti)**.** Due codici $\mathcal{C}$, $\mathcal{C}'$ si dicono *equivalenti* se è possibile ottenere tutte le parole di $\mathcal{C}'$ applicando a quelle di $\mathcal{C}$ un numero finito di volte le seguenti operazioni:

1. permutare le posizioni degli n simboli;
2. moltiplicare i simboli di una data posizione per uno stesso scalare non nullo.

In particolare, abbiamo modificato l'operazione 2 riducendo il numero di permutazioni consentite. Infatti, lavorando in un campo finito con q elementi, moltiplicare i simboli di una fissata posizione per uno stesso scalare non nullo equivale a effettuare una permutazione. In particolare, sono ora possibili solo $q-1$ permutazioni, mentre la Definizione 17 le consentiva tutte, cioè $q!$.
La Definizione 24 rispetta la struttura di spazio vettoriale. Infatti, sia $\mathcal{C}$ un codice lineare, M una sua matrice generatrice e $\mathcal{C}'$ il codice equivalente ottenuto da $\mathcal{C}$ tramite le operazioni consentite dalla definizione. Applicare una stessa permutazione a tutte le parole di $\mathcal{C}$, equivale a permutare le colonne di M. Moltiplicare i simboli di una data posizione per uno stesso scalare non nullo, invece, significa moltiplicare per questo scalare una colonna di M. Le righe della matrice generatrice sono una base di $\mathcal{C}$, quindi sono linearmente indipendenti. Di conseguenza, sono linearmente indipendenti anche le colonne di M e permutarle o moltiplicarle per uno stesso scalare non nullo non

modifica la loro indipendenza. Dopo aver eseguito le operazioni descritte sopra abbiamo una nuova matrice con righe linearmente indipendenti che generano un codice lineare $\mathcal{C}'$ equivalente a $\mathcal{C}$.

L'equivalenza tra codici lineari può essere verificata attraverso le rispettive matrici generatrici, come mostrato dal seguente teorema [3, par. 4.5].

Teorema 25. *Siano $\mathcal{C}, \mathcal{C}'$ due codici lineari e siano G, G' le loro rispettive matrici generatrici. Allora $\mathcal{C}$ e $\mathcal{C}'$ sono equivalenti se, e solo se, è possibile ottenere G' da G ripetendo un numero finito di volte le seguenti operazioni:*

1. *moltiplicazione di una riga per uno scalare non nullo;*
2. *somma di una riga con il multiplo di un'altra;*
3. *permutazione delle righe tra loro;*
4. *permutazione delle colonne tra loro;*
5. *moltiplicazione di una colonna per uno scalare non nullo.*

Dimostrazione. Applichiamo alla matrice G le operazioni descritte sopra e vediamo che si ottiene un codice equivalente a $\mathcal{C}$.
Eseguire un numero finito di volte le prime tre operazioni sulle righe della matrice significa eseguirle sui vettori di base di uno spazio vettoriale, cioè si ottiene una nuova base che genera lo stesso spazio vettoriale. Dunque cambiano le entrate di G ma il codice $\mathcal{C}$ rimane invariato.
Permutare le colonne tra loro e moltiplicare una colonna per uno scalare non nullo, equivalgono rispettivamente ad applicare a tutte le parole di $\mathcal{C}$ l'operazione (1) e l'operazione (2) della Definizione 24: si ottiene un codice equivalente a $\mathcal{C}$.
Quindi se è possibile ottenere la matrice G' da G applicando un numero finito di volte le precedenti operazioni, il codice $\mathcal{C}'$ generato da G' è equivalente a $\mathcal{C}$.

Viceversa, siano $\mathcal{C}$ e $\mathcal{C}'$ codici lineari equivalenti e siano G e G' le rispettive matrici generatrici, le cui righe sono delle basi $\mathfrak{B}$ e $\mathfrak{B}'$ degli spazi vettoriali formati dai due codici.
Essendo i due codici equivalenti, si possono ottenere tutte le parole di $\mathcal{C}'$ applicando un numero finito di volte le operazioni della Definizione 24 alle parole di $\mathcal{C}$. In particolare, le applichiamo alla base $\mathfrak{B}$. Ciò significa permutare le colonne di G oppure moltiplicare una colonna per uno scalare non nullo. Otteniamo una matrice generatrice di $\mathcal{C}'$: le sue righe sono una base di $\mathcal{C}'$ che, però, non è necessariamente la base $\mathfrak{B}'$. Per ottenere esattamente $\mathfrak{B}'$ è ora sufficiente eseguire un cambio di base, cioè basta eseguire le operazioni elementari sulle righe della matrice ottenuta: moltiplicare una riga per uno scalare non nullo, sommare una riga con il multiplo di un'altra oppure

permutare le righe. Dunque è possibile ottenere esattamente la matrice G' eseguendo le cinque operazioni descritte nell'enunciato.

□

Osservazione. Sommare una colonna della matrice generatrice con il multiplo di un'altra non garantisce l'equivalenza tra il codice di partenza e quello ottenuto. Quindi righe e colonne non ricoprono lo stesso ruolo. Infatti, questa operazione non corrisponde a nessuna operazione permessa dalla Definizione 24 e neppure a un cambio di base per il codice, il quale si opera sulle righe.

Come già visto nel paragrafo 3.4, la matrice generatrice di un codice lineare non è unica e, soprattutto per lavorare con codici di grandi dimensioni, è importante che essa sia nella forma più semplice possibile. Non tutti i codici lineari, però, ammettono una matrice generatrice in forma standard. È comunque possibile dimostrare che ogni codice lineare è equivalente ad un codice sistematico [3, par. 4.5].

Teorema 26. *Per ogni $[n,k]$-codice $\mathcal{C}$ definito su $\mathbb{F}_q$ esiste una matrice*

$$G = (I_k|A)$$

le cui righe generano $\mathcal{C}$, oppure generano un $[n,k]$-codice $\mathcal{C}'$ equivalente a $\mathcal{C}$.

Dimostrazione. Sia $G = (g_{ij})$ una matrice generatrice di $\mathcal{C}$. Dimostriamo per induzione su k, la dimensione del codice.
• $k = 1$. G è una matrice $1 \times n$: esiste almeno una colonna che contiene un elemento non nullo. A meno di permutare l'ordine delle colonne, ottenendo un codice equivalente, possiamo supporre che questa colonna sia la prima. Se $g_{11} = 1$, la matrice è in forma standard. Altrimenti, possiamo moltiplicare la riga per g_{11}^{-1} ottenendo un codice sistematico equivalente a $\mathcal{C}$.
• Supponiamo che ogni codice $(k-1)$-dimensionale sia equivalente a un codice sistematico. Sia $\mathcal{C}$ un $[n,k]$-codice e sia $\mathcal{C}'$ il codice ottenuto cancellando l'ultima riga di G. $\mathcal{C}'$ è un codice lineare di dimensione $k-1$. Allora, $\mathcal{C}'$ è equivalente a un codice $\mathcal{C}''$ che ammette una matrice generatrice standard. Questa matrice viene ottenuta dalla matrice generatrice $G'' = (g''_{ij})$ di $\mathcal{C}''$ attraverso le operazioni sulle righe definite dal Teorema 25 ed eventualmente una permutazione σ sulle colonne. Un codice equivalente a $\mathcal{C}$ viene generato dalla seguente matrice:

$$\widetilde{G} = \begin{pmatrix} 1 & 0 & \dots & 0 & g''_{1,k} & \dots & g''_{1,n} \\ 0 & 1 & \dots & 0 & g''_{2,k} & \dots & g''_{2,n} \\ \vdots & & \ddots & \vdots & \vdots & & \vdots \\ \vdots & \dots & & 1 & g''_{k-1,k} & \dots & g''_{k-1,n} \\ g_{k,\sigma(1)} & g_{k,\sigma(2)} & \dots & g_{k,\sigma(k-1)} & g_{k,\sigma(k)} & \dots & g_{k,\sigma(n)} \end{pmatrix}.$$

Sottraiamo all'ultima riga di $\widetilde{G}$ $g_{k,\sigma(j)}$ volte la j-riga, $j = 1, \ldots, k-1$, e moltiplichiamola per $g_{k,\sigma(k)}^{-1}$: otteniamo una matrice in forma standard che genera un codice lineare equivalente a $\mathcal{C}$.

□

Capitolo 4

Codifica e decodifica di codici lineari

4.1 Codifica

Codificare un messaggio composto da k simboli di $\mathbb{F}_q$ attraverso un (n, k)-codice significa associare in modo univoco a questa k-upla una parola del codice. Questa operazione è particolarmente semplice per i codici lineari se si utilizza una loro matrice generatrice.

Sia $\mathcal{C}$ un $[n, k]$-codice, cioè un codice lineare di dimensione k e lunghezza n, su $\mathbb{F}_q$. Fissata una base $\{\mathbf{g}_1, \ldots, \mathbf{g}_k\}$, ogni parola $\mathbf{x}$ di $\mathcal{C}$ si scrive in modo unico come combinazione lineare dei vettori di base:

$$\mathbf{x} = \alpha_1 \mathbf{g}_1 + \cdots + \alpha_k \mathbf{g}_k, \qquad \alpha_1, \ldots, \alpha_k \in \mathbb{F}_q.$$

Se G è la matrice generatrice di $\mathcal{C}$, le cui righe sono i vettori di base $\mathbf{g}_i$, con $i = 1, \ldots, k$, allora dato $\mathbf{u} = (u_1, \ldots, u_k) \in \mathbb{F}_q^k$, abbiamo che $\mathbf{u} \cdot G = u_1 \mathbf{g}_1 + \cdots + u_k \mathbf{g}_k$ è una parola del codice.
Per semplicità possiamo supporre che i posti di informazione siano i primi k; in caso contrario è sufficiente una permutazione delle colonne di G per ottenere un codice equivalente che abbia insieme di informazione $I = \{1, \ldots, k\}$. Sia $\mathbf{m} = (m_1, \ldots, m_k)$ il messaggio da codificare. Se la lunghezza di $\mathbf{m}$ fosse maggiore di k, allora sarebbe necessario dividerlo in k-uple da codificare separatamente. La parola di $\mathcal{C}$ associata al messaggio è $\mathbf{c} = \mathbf{m} \cdot G$:

$$\mathbf{c} = (m_1, \ldots, m_k) \begin{pmatrix} g_{1,1} & \cdots & g_{1,k} & \cdots & g_{1,n} \\ g_{2,1} & \cdots & g_{2,k} & \cdots & g_{2,n} \\ \vdots & & & & \\ g_{k,1} & \cdots & g_{k,k} & \cdots & g_{k,n} \end{pmatrix} = (c_1, \ldots, c_n).$$

Sfruttando le proprietà degli spazi vettoriali possiamo dunque associare ad ogni k-upla di simboli contenenti il messaggio una ed una sola parola del

codice, aggiungendo in automatico la ridondanza. Per un codice sistematico la codifica è ancora più immediata. Infatti, utilizzando una matrice generatrice in forma standard, si ritrova un'esatta copia del messaggio nelle prime k posizioni delle parole codificate ed è dunque necessario calcolare solo gli $n-k$ simboli di controllo. In questo caso $G = (I_k|A)$:

$$\mathbf{c} = (m_1, \dots, m_k) \begin{pmatrix} 1 & 0 & \dots & 0 & g_{1,k+1} & \dots & g_{1,n} \\ 0 & 1 & \dots & 0 & g_{2,k+1} & \dots & g_{2,n} \\ \vdots & & \ddots & \vdots & \vdots & & \vdots \\ 0 & \dots & & 1 & g_{k,k+1} & \dots & g_{k,n} \end{pmatrix}.$$

Dunque $\mathbf{c} = (m_1, \dots, m_k, \sum_{i=1}^{k} m_i g_{i,k+1}, \dots, \sum_{i=1}^{k} m_i g_{i,n})$.

Esempio 10. Consideriamo $\mathcal{C}$ un $[6,3]$-codice su $\mathbb{F}_3$ con G una matrice generatrice in forma standard:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2 \end{pmatrix}.$$

Vogliamo codificare il messaggio $\mathbf{m} = (1\,0\,2\,1\,2\,0\,0\,1\,0\,0\,2\,2)$. Poiché la sua lunghezza supera la dimensione del codice che è 3, è necessario suddividerlo in terne:

$$\mathbf{m}_1 = (1\,0\,2) \quad \mathbf{m}_2 = (1\,2\,0) \quad \mathbf{m}_3 = (0\,1\,0) \quad \mathbf{m}_4 = (0\,2\,2).$$

Codifichiamo ogni sequenza attraverso la matrice generatrice:

$$\begin{aligned} \mathbf{c}_1 &= \mathbf{m}_1 \cdot G = (1\,0\,2\,0\,2\,2) \\ \mathbf{c}_2 &= \mathbf{m}_2 \cdot G = (1\,2\,0\,1\,1\,1) \\ \mathbf{c}_3 &= \mathbf{m}_3 \cdot G = (0\,1\,0\,2\,1\,0) \\ \mathbf{c}_4 &= \mathbf{m}_4 \cdot G = (0\,2\,2\,1\,2\,1) \end{aligned}$$

Notiamo che in tutte le quattro parole codificate sopra le prime tre posizioni sono occupate da una delle terne in cui è stato suddiviso il messaggio iniziale. Questo è dovuto al fatto che la matrice generatrice del codice è in forma standard.
Una volta codificate tutte le parti del messaggio, queste ultime possono essere riunite e inviate come un'unica sequenza:

$$(1\,0\,2\,0\,2\,2\,1\,2\,0\,1\,1\,1\,0\,1\,0\,2\,1\,0\,0\,2\,2\,1\,2\,1).$$

4.2 Decodifica tramite tabella standard

Dopo la codifica, il messaggio viene inviato e durante la trasmissione possono verificarsi degli errori. È dunque in fase di decodifica che devono essere riconosciuti per poter associare alle sequenze ricevute le giuste parole del codice. Un metodo di decodifica è quello di utilizzare una tabella, detta *tabella standard*. L'algoritmo utilizzato per costruire questa tabella è basato sulla definizione di *laterale* di un codice lineare [5, par. 4.8].

Definizione 27. Siano $\mathcal{C}$ un $[n,k]$-codice su $\mathbb{F}_q$ e $\mathbf{u} \in \mathbb{F}_q^n$. Il *laterale* di $\mathcal{C}$ determinato da $\mathbf{u}$ è l'insieme

$$\mathcal{C} + \mathbf{u} = \{\mathbf{v} + \mathbf{u} : \mathbf{v} \in \mathcal{C}\}.$$

Notiamo che, se consideriamo l'operazione di somma vettoriale, $\mathbb{F}_q^n$ è un gruppo abeliano finito. Quindi, un codice lineare $\mathcal{C}$ di lunghezza n su $\mathbb{F}_q$ è un sottogruppo di $\mathbb{F}_q^n$ e la definizione precedente di laterale di un codice coincide con la definizione di classe laterale di un sottogruppo data dalla teoria dei gruppi.

Definizione 28. Dato un laterale di $\mathcal{C}$, si dice *leader* o *direttrice* del laterale una sua parola con il peso di Hamming minimo.

Osserviamo che in un laterale possono esserci più parole con lo stesso peso di Hamming, in particolare con il peso minimo. Pertanto la scelta della parola leader non è univoca.

Per completezza, richiamiamo con il seguente teorema le proprietà delle classi laterali che serviranno per la costruzione degli algoritmi di decodifica.

Teorema 29. *Sia $\mathcal{C}$ un $[n,k,d]$-codice definito su $\mathbb{F}_q$. Allora:*

1. *ogni vettore di $\mathbb{F}_q^n$ appartiene a qualche laterale di $\mathcal{C}$;*
2. *per ogni $\boldsymbol{u} \in \mathbb{F}_q^n$, $|\mathcal{C} + \boldsymbol{u}| = |\mathcal{C}| = q^k$;*
3. *per ogni $\boldsymbol{u}$, $\boldsymbol{v} \in \mathbb{F}_q^n$, $\boldsymbol{u} \in \mathcal{C} + \boldsymbol{v}$ implica che $\mathcal{C} + \boldsymbol{u} = \mathcal{C} + \boldsymbol{v}$;*
4. *due laterali coincidono oppure hanno intersezione vuota;*
5. *ci sono q^{n-k} laterali di $\mathcal{C}$ distinti;*
6. *per ogni $\boldsymbol{u}$, $\boldsymbol{v} \in \mathbb{F}_q^n$, $\boldsymbol{u} - \boldsymbol{v} \in \mathcal{C}$ se, e solo se, $\boldsymbol{u}$ e $\boldsymbol{v}$ appartengono allo stesso laterale.*

Dimostrazione. (1) È immediato vedere che il vettore $\mathbf{v} \in \mathbb{F}_q^n$ appartiene ad almeno un laterale: $\mathbf{v} \in \mathcal{C} + \mathbf{v}$.
(2) Per definizione, ogni laterale $\mathcal{C}+\mathbf{u}$ di $\mathcal{C}$ ha al più $|\mathcal{C}| = q^k$ elementi. Poiché $\mathcal{C}$ è un codice lineare, e dunque uno spazio vettoriale, due elementi $\mathbf{c} + \mathbf{u}$ e

$\mathbf{c}' + \mathbf{u}$ di $\mathcal{C} + \mathbf{u}$ coincidono se, e solo se, $\mathbf{c} = \mathbf{c}'$. Quindi $|\mathcal{C} + \mathbf{u}| = |\mathcal{C}| = q^k$.
(3) Se $\mathbf{u} \in \mathcal{C} + \mathbf{v}$, dalla definizione di laterale segue che $\mathcal{C} + \mathbf{u} \subseteq \mathcal{C} + \mathbf{v}$. Per il punto (2) i due laterali hanno la stessa cardinalità, quindi $\mathcal{C} + \mathbf{u} = \mathcal{C} + \mathbf{v}$.
(4) Consideriamo due laterali $\mathcal{C} + \mathbf{u}$ e $\mathcal{C} + \mathbf{v}$ di $\mathcal{C}$ e sia $\mathbf{x} \in (\mathcal{C} + \mathbf{u}) \cap (\mathcal{C} + \mathbf{v})$. Per il punto (3), poiché $\mathbf{x} \in \mathcal{C} + \mathbf{u}$, $\mathcal{C} + \mathbf{u} = \mathcal{C} + \mathbf{x}$. Allo stesso modo $\mathbf{x} \in \mathcal{C} + \mathbf{v}$ e $\mathcal{C} + \mathbf{v} = \mathcal{C} + \mathbf{x}$. Dunque $\mathcal{C} + \mathbf{u} = \mathcal{C} + \mathbf{v}$.
(5) Per il punto (1) ogni vettore di $\mathbb{F}_q^n$ appartiene a qualche laterale di $\mathcal{C}$, in particolare abbiamo q^n elementi. Per il punto (2) ogni laterale contiene esattamente q^k elementi e per il punto (4) se uno stesso elemento appartiene a due laterali, questi devono coincidere. Quindi i q^n elementi di $\mathbb{F}_q^n$ devono essere suddivisi in gruppi di q^k elementi, cioè ci sono $\frac{q^n}{q^k} = q^{n-k}$ laterali distinti.
(6) Se $\mathbf{u} - \mathbf{v} = \mathbf{c} \in \mathcal{C}$, allora $\mathbf{u} = \mathbf{c} + \mathbf{v} \in \mathcal{C} + \mathbf{v}$. Quindi $\mathbf{u} \in \mathcal{C} + \mathbf{v}$. Poiché anche $\mathbf{v} \in \mathcal{C} + \mathbf{v}$, $\mathbf{u}$ e $\mathbf{v}$ appartengono allo stesso laterale.
Viceversa, supponiamo che $\mathbf{u}$ e $\mathbf{v}$ appartengano allo stesso laterale $\mathcal{C} + \mathbf{x}$. Allora esistono $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$ tali che $\mathbf{u} = \mathbf{c} + \mathbf{x}$ e $\mathbf{v} = \mathbf{c}' + \mathbf{x}$. Quindi $\mathbf{u} - \mathbf{v} = \mathbf{c} - \mathbf{c}' \in \mathcal{C}$.

□

Esempio 11. Consideriamo $\mathcal{C} = \{(0\,0\,0), (1\,0\,1), (0\,1\,0), (1\,1\,1)\}$, un $[3, 2]$-codice definito su $\mathbb{F}_2$. I suoi possibili laterali sono:

$$\begin{aligned}
\mathcal{C} + (0\,0\,0) &= \{(0\,0\,0), (1\,0\,1), (0\,1\,0), (1\,1\,1)\};\\
\mathcal{C} + (0\,0\,1) &= \{(0\,0\,1), (1\,0\,0), (0\,1\,1), (1\,1\,0)\};\\
\mathcal{C} + (0\,1\,0) &= \{(0\,1\,0), (1\,1\,1), (0\,0\,0), (1\,0\,1)\};\\
\mathcal{C} + (0\,1\,1) &= \{(0\,1\,1), (1\,1\,0), (0\,0\,1), (1\,0\,0)\};\\
\mathcal{C} + (1\,0\,0) &= \{(1\,0\,0), (0\,0\,1), (1\,1\,0), (0\,1\,1)\};\\
\mathcal{C} + (1\,0\,1) &= \{(1\,0\,1), (0\,0\,0), (1\,1\,1), (0\,1\,0)\};\\
\mathcal{C} + (1\,1\,0) &= \{(1\,1\,0), (0\,1\,1), (1\,0\,0), (0\,0\,1)\};\\
\mathcal{C} + (1\,1\,1) &= \{(1\,1\,1), (0\,1\,0), (1\,0\,1), (0\,0\,0)\}.
\end{aligned}$$

Osserviamo che alcuni laterali coincidono:

$$\begin{aligned}
&\mathcal{C} + (0\,0\,0) = \mathcal{C} + (0\,1\,0) = \mathcal{C} + (1\,0\,1) = \mathcal{C} + (1\,1\,1) = \mathcal{C};\\
&\mathcal{C} + (0\,0\,1) = \mathcal{C} + (0\,1\,1) = \mathcal{C} + (1\,0\,0) = \mathcal{C} + (1\,1\,0) = \mathbb{F}_2^3 \setminus \mathcal{C}.
\end{aligned}$$

Come affermato dal teorema precedente, ogni laterale contiene esattamente $q^k = 2^2 = 4$ elementi e ci sono $q^{n-k} = 2^{3-2} = 2$ laterali distinti. Osserviamo inoltre che per il laterale coincidente con $\mathcal{C}$ è possibile un'unica scelta del leader, infatti $(0\,0\,0)$ è l'unica parola di peso minimo; mentre per l'altro laterale si può scegliere come leader sia $(0\,0\,1)$ che $(1\,0\,0)$.

Vediamo ora il principio di funzionamento dell'algoritmo di decodifica mediante tabella standard di un codice lineare $\mathcal{C}$ di lunghezza n e dimensione

k [2, p. 43].
Sia $m = q^k$. I laterali distinti di $\mathcal{C}$ sono $\mathcal{C}_i = \mathcal{C} + \mathbf{a}_i$, con $i = 1, \ldots, m$ e $\mathbf{a}_i \in \mathbb{F}_q^n$. Dal Teorema 29 segue che ogni elemento di $\mathbb{F}_q^n$ appartiene ad un solo laterale di $\mathcal{C}$, dunque, ogni sequenza di simboli $\mathbf{y}$ ricevuta appartiene ad un solo $\mathcal{C}_i$. Esiste $\overline{i} \in \{1, \ldots, m\}$ tale che $\mathbf{y} \in \mathcal{C}_{\overline{i}}$ e si può scrivere $\mathbf{y} = \mathbf{c} + \mathbf{a}_{\overline{i}}$, con $\mathbf{c} \in \mathcal{C}$.
Se trasmettendo la parola $\mathbf{x}$ si riceve la parola $\mathbf{y}$, il vettore $\mathbf{e}$ che rappresenta l'errore commesso è così definito: $\mathbf{e} = \mathbf{y} - \mathbf{x} = \mathbf{c} + \mathbf{a}_{\overline{i}} - \mathbf{x} = \mathbf{a}_{\overline{i}} + \overline{\mathbf{c}}$, con $\overline{\mathbf{c}} \in \mathcal{C}$. Quindi $\mathbf{e}$ appartiene allo stesso laterale di $\mathbf{y}$.
L'errore più probabile è quello di peso minimo. Infatti, se p è la probabilità che avvenga un errore, p^r è la probabilità che ne avvengano r all'interno della stessa parola. Di conseguenza $p^r < p^s$ se $r > s$, in quanto $0 \leq p \leq 1$. L'algoritmo di decodifica sceglie dunque come errore il vettore $\mathbf{e}^*$ di peso minimo in $\mathcal{C}_i$. La sequenza $\mathbf{y}$ ricevuta viene decodificata come $\mathbf{y} - \mathbf{e}^*$.

Utilizziamo i laterali di $\mathcal{C}$ per costruire la tabella standard. Questa tabella è rappresentata da una matrice $\Sigma = (\sigma_{ij})$, di q^{n-k} righe e q^k colonne, le cui entrate $\sigma_{ij} \in \mathbb{F}_q^n$ sono tutti e soli gli elementi di $\mathbb{F}_q^n$. Devono essere soddisfatte le seguenti proprietà:

1. sulla prima riga si trovano tutte le parole di $\mathcal{C}$ e $\sigma_{11} = (0, 0, \ldots, 0)$;
2. l'elemento σ_{i1} è la parola $\mathbf{a}_i$ scelta come leader del laterale $\mathcal{C}_i$, cioè la prima colonna contiene tutte le parole leader scelte, una per ogni laterale;
3. la riga i-esima contiene tutti gli elementi del laterale $\mathcal{C}_i$, cioè per ogni coppia di indici i, j si ha $\sigma_{ij} = \mathbf{a}_i + \sigma_{1j} = \sigma_{i1} + \sigma_{1j}$.

La tabella standard può essere costruita tramite un semplice algoritmo [2, p. 44]:

passo 1 : si pone $\sigma_{1,1} = \mathbf{0}$ e si distribuiscono le altre parole di $\mathcal{C}$ nelle restanti entrate della prima riga;

passo 2 : si sceglie una parola $\mathbf{a}_2$ con il peso di Hamming minimo in $\mathbb{F}_q^n \setminus \mathcal{C}$ e si pone $\sigma_{21} = \mathbf{a}_2$;

passo 3 : si distribuiscono le restanti parole di $\mathcal{C} + \mathbf{a}_2$ nella seconda riga di Σ in modo che $\sigma_{2j} = \mathbf{a}_2 + \sigma_{1j}$, per $j = 2, \ldots, q^k$;

$\vdots$

passo h : si sceglie una parola $\mathbf{a}_h$ con il peso di Hamming minimo in $\mathbb{F}_q^n \setminus \{\mathcal{C} \cup (\mathcal{C} + \mathbf{a}_2) \cup \cdots \cup (\mathcal{C} + \mathbf{a}_{h-1})\}$ e si pone $\sigma_{h1} = \mathbf{a}_h$;

passo h + 1 : si distribuiscono le restanti parole di $\mathcal{C} + \mathbf{a}_h$ nella h-esima riga di Σ in modo che $\sigma_{hj} = \mathbf{a}_h + \sigma_{1j}$, per $j = 2, \ldots, q^k$.

L'algoritmo termina quando vengono inserite tutte le parole di $\mathbb{F}_q^n$. Questa procedura ci permette di inserire ogni elemento di $\mathbb{F}_q^n$ una ed una sola volta. Notiamo che l'algoritmo appena descritto non fornisce una tabella standard univocamente determinata ma, se ci sono più parole con il peso di Hamming minimo, la disposizione delle sue entrante dipende dalle scelte effettuate sulle parole leader, oltre che dall'ordine in cui si dispongono gli elementi di $\mathcal{C}$ nella prima riga. Ciò potrebbe variare l'esito della decodifica, come evidenzieremo nell'Esempio 12.
Una volta costruita la tabella standard, il decodificatore può procedere all'effettiva decodifica del messaggio tramite il seguente algoritmo:

passo 1 : viene ricevuta una sequenza di simboli $\mathbf{y} \in \mathcal{C}_i$;

passo 2 : viene individuata la posizione (i, j) di $\mathbf{y}$ nella tabella standard;

passo 3 : $\mathbf{y}$ viene decodificata come la parola del codice che si trova nella stessa colonna, cioè $\mathbf{y} - \mathbf{a}_i$. La parola restituita è dunque σ_{1j}.

Vediamo con un esempio quanto descritto finora [5, par. 4.8].

Esempio 12. Consideriamo $\mathcal{C} = \{(0\,0\,0\,0), (1\,0\,1\,1), (0\,1\,0\,1), (1\,1\,1\,0)\}$, un codice lineare binario, e costruiamo la sua matrice standard.

- La prima riga contiene il laterale $\mathcal{C} + (0\,0\,0\,0)$ e la prima entrata è $\mathbf{0}$. Abbiamo: $0\,0\,0\,0 \quad 1\,0\,1\,1 \quad 0\,1\,0\,1 \quad 1\,1\,1\,0$.
- Scegliamo $(0\,0\,0\,1)$ come parola leader. La seconda riga contiene il laterale $\mathcal{C} + (0\,0\,0\,1)$ e la prima entrata è $0\,0\,0\,1$. Le altre entrate si ottengono sommando la parola leader alle parole della prima riga. Abbiamo: $0\,0\,0\,1 \quad 1\,0\,1\,0 \quad 0\,1\,0\,0 \quad 1\,1\,1\,1$.
- Scegliamo $(0\,0\,1\,0)$ come parola leader. La terza riga contiene il laterale $\mathcal{C} + (0\,0\,1\,0)$ e la prima entrata è $0\,0\,1\,0$. Come sopra, le altre entrate si ottengono sommando la parola leader alle parole della prima riga. Abbiamo: $0\,0\,1\,0 \quad 1\,0\,0\,1 \quad 0\,1\,1\,1 \quad 1\,1\,0\,0$.
- Ora l'unica parola leader possibile è $(1\,0\,0\,0)$. La quarta riga contiene il laterale $\mathcal{C} + (1\,0\,0\,0)$ e la prima entrata è $1\,0\,0\,0$. Allo stesso modo abbiamo: $1\,0\,0\,0 \quad 0\,0\,1\,1 \quad 1\,1\,0\,1 \quad 0\,1\,1\,0$.

La matrice standard che abbiamo costruito è dunque:

$$\begin{array}{cccc} 0\,0\,0\,0 & 1\,0\,1\,1 & 0\,1\,0\,1 & 1\,1\,1\,0 \\ 0\,0\,0\,1 & 1\,0\,1\,0 & 0\,1\,0\,0 & 1\,1\,1\,1 \\ 0\,0\,1\,0 & 1\,0\,0\,1 & 0\,1\,1\,1 & 1\,1\,0\,0 \\ 1\,0\,0\,0 & 0\,0\,1\,1 & 1\,1\,0\,1 & 0\,1\,1\,0 \end{array}$$

Supponiamo ora di ricevere le sequenze $\mathbf{v} = 1\,1\,0\,1$ e $\mathbf{w} = 1\,1\,1\,1$.
Decodifichiamo $\mathbf{v}$. Questa sequenza appartiene al laterale $\mathcal{C} + \mathbf{v}$ che coincide

con la quarta riga della tabella standard. L'unica possibile parola leader di questa riga è 1 0 0 0, quindi **v** viene decodificata con la parola del codice

$$1\,1\,0\,1 - 1\,0\,0\,0 = 1\,1\,0\,1 + 1\,0\,0\,0 = 0\,1\,0\,1.$$

Decodifichiamo **w**. Questa sequenza appartiene al laterale $\mathcal{C}+\mathbf{w}$ che coincide con la seconda riga della tabella. In questo caso, durante la costruzione della tabella, abbiamo effettuato una scelta tra due possibili parole leader dello stesso laterale. Seguendo la tabella precedente, **w** viene decodificata con la parola del codice 1 1 1 0. Se avessimo invece scelto l'altra parola leader, cioè 0 1 0 0, la decodifica di **w** sarebbe stata

$$1\,1\,1\,1 - 0\,1\,0\,0 = 1\,1\,1\,1 + 0\,1\,0\,0 = 1\,0\,1\,1.$$

In conclusione, se in ogni laterale del codice c'è una sola parola con il peso di Hamming minimo, la decodifica è univoca in quanto, dopo aver disposto le parole del codice nella prima riga, le possibili tabelle standard si differenziano tra loro solo per un'eventuale permutazione delle righe che non coinvolge la prima. Quindi la parola del codice codificata si trova in ogni caso in cima alla colonna a cui appartiene la sequenza ricevuta. Invece, se un laterale possiede più parole di peso minimo, l'esito della decodifica potrebbe essere diverso a seconda delle scelte effettuate sulle parole leader. In questo caso, l'implementazione dell'algoritmo dipende dal tipo di decodifica che si vuole effettuare. Se si opta per una decodifica *completa*, la scelta è arbitraria e ad ogni sequenza ricevuta viene associata una parola del codice seguendo l'algoritmo. Se si opta per una decodifica *incompleta*, invece, nel caso in cui la sequenza ricevuta appartenga ad un laterale con più possibili parole leader, viene richiesta la ritrasmissione.

4.3 Codice duale e matrice di controllo

La decodifica tramite matrice standard è utile per codici di lunghezza n piccola. Risulta invece sempre meno efficiente in termini di tempo man mano che la dimensione del codice aumenta. In quest'ultimo caso è più conveniente la *decodifica per sindrome*. Per descrivere l'algoritmo di questo tipo di decodifica dobbiamo introdurre il *codice duale* e la *matrice di controllo* [7, par. 3.2].

Definizione 30 (Codice duale)**.** Sia $\mathcal{C}$ un $[n,k]$-codice. Il codice *duale* di $\mathcal{C}$ è

$$\mathcal{C}^{\perp} \coloneqq \{\mathbf{y} \in \mathbb{F}_q^n : \mathbf{x}\cdot\mathbf{y} = 0 \quad \forall\, \mathbf{x} \in \mathcal{C}\},$$

dove il simbolo $\cdot$ rappresenta l'operazione di prodotto scalare tra vettori.

Un codice $\mathcal{C}$ e il suo codice duale possono avere un'intersezione più grande di $\{\mathbf{0}\}$. I vettori che appartengono a questa intersezione si dicono *isotropi*. Se $\mathcal{C} = \mathcal{C}^{\perp}$, allora $\mathcal{C}$ si dice *codice autoduale*.

Per completezza riportiamo alcuni risultati sui codici duali che derivano dalla teoria degli spazi vettoriali [2, p. 46-48].

Teorema 31. *Sia $\mathcal{C}$ un $[n,k]$-codice su $\mathbb{F}_q$ con matrice generatrice G. Allora un vettore $\boldsymbol{x} \in \mathbb{F}_q^n$ è una parola del codice duale $\mathcal{C}^\perp$ se, e solo se, è ortogonale a tutte le righe di G, cioè:*

$$\boldsymbol{x} \in \mathcal{C}^\perp \iff \boldsymbol{x}G^t = \mathbf{0}.$$

Dimostrazione. Se $\mathbf{x} \in \mathcal{C}^\perp$, allora $\mathbf{x} \cdot \mathbf{c}$ per ogni $\mathbf{c} \in \mathcal{C}$, cioè $\mathbf{x}$ è ortogonale a tutte le parole del codice. In particolare, è ortogonale alle parole che costituiscono la base di $\mathcal{C}$ data dalle righe di G. Quindi $\mathbf{x}G^t = \mathbf{0}$.

Viceversa, sia $\mathbf{x}G^t = \mathbf{0}$. Abbiamo che $\mathbf{x}$ è ortogonale a tutte le righe di G, cioè ai vettori $\mathbf{g}_1, \ldots, \mathbf{g}_k$ che formano una base di $\mathcal{C}$. Poiché ogni parola $\mathbf{c} \in \mathcal{C}$ si scrive come combinazione lineare di questi vettori, $\mathbf{c} = \alpha_1\mathbf{g}_1 + \cdots + \alpha_k\mathbf{g}_k$, abbiamo:

$$\mathbf{x} \cdot \mathbf{c} = \mathbf{x} \cdot (\alpha_1\mathbf{g}_1 + \cdots + \alpha_k\mathbf{g}_k) = \alpha_1(\mathbf{x} \cdot \mathbf{g}_1) + \cdots + \alpha_k(\mathbf{x} \cdot \mathbf{g}_k) = 0.$$

Quindi $\mathbf{x}$ è ortogonale a tutte le parole di $\mathcal{C}$ e di conseguenza $\mathbf{x} \in \mathcal{C}^\perp$.

□

Teorema 32. *Sia $\mathcal{C}$ un $[n,k]$-codice su $\mathbb{F}_q$. Allora $\mathcal{C}^\perp$ è un $[n, n-k]$-codice su $\mathbb{F}_q$.*

Dimostrazione. Per il teorema precedente, i vettori di $\mathbb{F}_q^n$ ortogonali ad una base $\{\mathbf{e}_1, \ldots, \mathbf{e}_k\}$ di $\mathcal{C}$ sono tutti e soli i vettori $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{C}^\perp$. Dunque sono tutte e sole le soluzioni del sistema

$$\begin{cases} e_{11}x_1 + \cdots + e_{1n}x_n = 0 \\ e_{21}x_1 + \cdots + e_{2n}x_n = 0 \\ \quad \vdots \\ e_{k1}x_k + \cdots + e_{kn}x_n = 0 \end{cases}$$

che è un sistema lineare omogeneo composto da k equazioni in n incognite. Il rango della matrice associata è k, di conseguenza ci sono esattamente $n-k$ soluzioni linearmente indipendenti. Queste soluzioni formano una base di $\mathcal{C}^\perp$, in quanto ogni loro combinazione lineare risolve il sistema precedente ed è dunque un vettore del codice duale. Di conseguenza $\mathcal{C}^\perp$ è un codice lineare di dimensione $n-k$.

□

Definizione 33 (Matrice di controllo)**.** Dato $\mathcal{C}$ un codice lineare, una matrice generatrice di $\mathcal{C}^\perp$ si dice *matrice di controllo* o *matrice di controllo di parità* di $\mathcal{C}$. È una matrice $(n-k) \times n$ e si indica con H.

Teorema 34. *Sia $\boldsymbol{x} = (x_1, \ldots, x_n)$ un vettore di $\mathbb{F}_q^n$. Allora:*

$$\boldsymbol{x} \in \mathcal{C} \iff \boldsymbol{x}H^t = \mathbf{0}.$$

Dimostrazione. Poiché $C^\perp$ è un codice lineare, dalla Definizione 30 si deduce che $(\mathcal{C}^\perp)^\perp = \mathcal{C}$. Dunque, $\mathbf{x} \in \mathcal{C}$ se, e solo se, $\mathbf{x} \cdot \mathbf{y} = 0 \quad \forall \mathbf{y} \in \mathcal{C}^\perp$. Ciò accade se, e solo se, $\mathbf{x}$ è ortogonale a tutti i vettori di una base di $\mathcal{C}^\perp$, i quali formano le righe di H. Ne segue che $\mathbf{x} \in \mathcal{C}$ se, e solo se, $\mathbf{x}H^t = 0$.

□

Come visto nei teoremi precedenti, l'importanza della matrice H risiede nel fatto che la si può utilizzare per verificare facilmente se una sequenza di simboli appartiene oppure no al codice. Quindi una matrice di controllo di parità descrive completamente il codice lineare $\mathcal{C}$ a cui è associata.
Poiché in generale la matrice generatrice non è unica, non lo è nemmeno la matrice di controllo di parità. Se il codice è sistematico, però, si può scegliere come matrice generatrice quella in forma standard e costruire H in modo semplice e immediato, come mostra il seguente teorema [2, par. 3.5].

Teorema 35. *Sia $\mathcal{C}$ un $[n, k]$-codice. Se $G = (I_k|A)$ è una matrice generatrice in forma standard di $\mathcal{C}$, allora una matrice di controllo di parità per $\mathcal{C}$ è $H = (-A^t|I_{n-k})$.*

Dimostrazione. Sia $G = (I_k|A)$ una matrice generatrice di $\mathcal{C}$ e consideriamo la matrice $H = (-A^t|I_{n-k})$. H è una matrice di controllo di parità per $\mathcal{C}$ se le sue righe formano una base dello spazio duale $\mathcal{C}^\perp$. Eseguiamo il seguente prodotto di matrici a blocchi:

$$GH^t = (I_k|A)\begin{pmatrix} -A \\ I_{n-k} \end{pmatrix} = -I_k A + AI_{n-k} = -A + A = O.$$

Abbiamo che le righe di H sono ortogonali a quelle di G, cioè sono ortogonali ad una base di $\mathcal{C}$, in particolare sono ortogonali a tutte le parole di $\mathcal{C}$. Poiché H ha esattamente $n - k$ righe linearmente indipendenti, deduciamo che H genera $\mathcal{C}^\perp$ ed è dunque una matrice di controllo di parità per $\mathcal{C}$. □

Un'importante proprietà della matrice di controllo di parità è che, studiando la relazione di linearità tra le sue colonne, possiamo dedurre la distanza minima del codice, come mostrato dal seguente teorema [1, par. 1.2].

Teorema 36. *Sia $\mathcal{C}$ un $[n, k, d]$-codice e sia H una matrice di controllo di parità di $\mathcal{C}$. Allora ogni $(d-1)$-upla di colonne di H è linearmente indipendente e c'è almeno una d-upla di colonne linearmente dipendenti.*

Dimostrazione. Indichiamo con $H_1, \ldots, H_n$ le colonne della matrice H. Sia $\mathbf{c} \in \mathcal{C}$ una parola del codice di peso w. Siano $i_1, \ldots, i_w$ gli indici in $\{1, \ldots, n\}$

tali che $\mathbf{c}_i \neq 0$ se e solo se $i \in \{i_1, \ldots, i_w\}$. Poiché $c \in \mathcal{C}$, per il Teorema 34 abbiamo che

$$\mathbf{c}H^t = 0$$

e questa relazione equivale a

$$c_{i_1}H_{i_1} + \cdots + c_{i_w}H_{i_w} = 0.$$

Quindi le combinazioni lineari nulle di w colonne distinte di H sono in corrispondenza uno a uno con le parole di $\mathcal{C}$ di peso w. Per il Teorema 20, il minimo peso delle parole non nulle di $\mathcal{C}$ è d. Di conseguenza, prese $d-1$ colonne qualunque, queste devono essere linearmente indipendenti, altrimenti ci sarebbe una parola di $\mathcal{C}$ di peso $d-1$. Inoltre, deve esserci almeno una d-upla di colonne linearmente dipendenti, altrimenti nessuna parola di $\mathcal{C}$ avrebbe peso d e la distanza minima sarebbe maggiore di d, assurdo. □

Dal teorema precedente discende direttamente il corollario seguente.

Corollario 37. *Sia $\mathcal{C}$ un codice lineare con matrice di controllo di parità H e distanza minima d. Allora:*

1. *se H non ha colonne nulle, allora $d > 1$;*
2. *se le colonne di H sono a due a due non collineari, allora $d > 2$.*

4.4 Decodifica per sindrome

Concludiamo il capitolo introducendo il concetto di *sindrome* e utilizzandolo per costruire un algoritmo di decodifica che rende più efficiente quello visto in precedenza [5, par. 4.8.3].

Definizione 38 (Sindrome)**.** Sia $\mathcal{C}$ un $[n, k, d]$-codice su $\mathbb{F}_q$ e sia H una matrice di controllo di parità per $\mathcal{C}$. Per ogni vettore $\mathbf{w} \in \mathbb{F}_q^n$, si dice *sindrome* di $\mathbf{w}$ la parola $S(\mathbf{w}) = \mathbf{w}H^t \in \mathbb{F}_q^{n-k}$.

Come si vede dalla definizione, la sindrome di un vettore dipende dalla matrice H. Quindi a seconda della base utilizzata può esserci una sindrome diversa per la stessa parola. Con il seguente teorema vediamo alcune proprietà della sindrome.

Teorema 39. *Sia $\mathcal{C}$ un $[n, k]$-codice e sia H una matrice di controllo di parità per $\mathcal{C}$. Per ogni $\boldsymbol{u}$, $\boldsymbol{v} \in \mathbb{F}_q^n$ abbiamo:*

1. *$S(\boldsymbol{u} + \boldsymbol{v}) = S(\boldsymbol{u}) + S(\boldsymbol{v})$;*
2. *$S(\boldsymbol{u}) = \mathbf{0} \iff \boldsymbol{u} \in \mathcal{C}$;*
3. *$S(\boldsymbol{u}) = S(\boldsymbol{v}) \iff \boldsymbol{u}$, $\boldsymbol{v}$ appartengono allo stesso laterale di $\mathcal{C}$.*

Dimostrazione. (1) Dalla Definizione 38 segue che per ogni $\mathbf{u}$, $\mathbf{v} \in \mathbb{F}_q^n$ si ha:

$$S(\mathbf{u}+\mathbf{v}) = (\mathbf{u}+\mathbf{v})H^t = \mathbf{u}H^t + \mathbf{v}H^t = S(\mathbf{u}) + S(\mathbf{v}).$$

(2) Dalla Definizione 38 segue che $S(\mathbf{u}) = \mathbf{0}$ se e solo se $\mathbf{u}H^t = \mathbf{0}$. Quindi, per il Teorema 34, $\mathbf{u} \in \mathcal{C}$.
(3) Dai punti precedenti segue che

$$\begin{aligned} S(\mathbf{u}) = S(\mathbf{v}) &\iff \\ S(\mathbf{u}) - S(\mathbf{v}) = \mathbf{0} &\iff \\ S(\mathbf{u}-\mathbf{v}) = \mathbf{0} &\iff \mathbf{u}-\mathbf{v} \in \mathcal{C}. \end{aligned}$$

Per il punto (6) del Teorema 29, $\mathbf{u}-\mathbf{v} \in \mathcal{C}$ se e solo se $\mathbf{u}$ e $\mathbf{v}$ appartengono allo stesso laterale di $\mathcal{C}$.

□

Osservazione. Il terzo punto del teorema precedente ci garantisce che possiamo identificare un laterale con la sua sindrome; viceversa, tutte le parole di un dato laterale forniscono la stessa sindrome. Dunque abbiamo una corrispondenza uno a uno tra i laterali e le sindromi.
Dal fatto che H ha rango $n-k$ segue che ogni vettore di $\mathbb{F}_q^{n-k}$ rappresenta una sindrome.

Questo ci permette di semplificare e rendere più efficiente l'algoritmo di decodifica visto precedentemente. Infatti, non è più necessario costruire la tabella standard contenente q^n elementi, ma è sufficiente una tabella di due sole colonne, una per le parole leader e una per le sindromi corrispondenti: abbiamo dunque solo $2q^{n-k}$ elementi e, soprattutto per codici di grandi dimensioni, è più agevole gestire la tabella con meno entrate.
Vediamo dunque l'**algoritmo di decodifica tramite sindrome**:

Passo 1 : si riceve la sequenza $\mathbf{w}$ e si calcola la sua sindrome $S(\mathbf{w})$;

Passo 2 : si individua nella tabella la parola leader $\mathbf{u}$ con la stessa sindrome di $\mathbf{w}$, $S(\mathbf{u}) = S(\mathbf{w})$;

Passo 3 : si decodifica $\mathbf{w}$ con la parola del codice $\mathbf{v} = \mathbf{w} - \mathbf{u}$.

Osservazione. Se durante la trasmissione non avviene nessun errore, la sindrome della parola ricevuta è zero poiché essa appartiene al codice. Non vale, però, il viceversa: se durante la trasmissione avviene un numero di errori tale da trasformare la parola di partenza in un'altra parola del codice, la sindrome calcolata è comunque zero ma la sequenza ricevuta non corrisponde a quella inviata. Aggiungendo delle ipotesi sui parametri del codice, la sindrome ci permette di rilevare ed eventualmente correggere eventuali errori. Ad esempio, se supponiamo che durante la trasmissione può verificarsi al più un

errore in ogni parola, e abbiamo un codice con distanza minima 3, siamo certi che in caso di errore la parola inviata si trasforma in una sequenza di simboli priva di significato. Dunque, se la sindrome della parola ricevuta è il vettore nullo, la parola ricevuta è sicuramente quella inviata; altrimenti, siamo certi che è avvenuto un errore e, se l'applicazione lo prevede, si procede con l'algoritmo per la correzione, oppure si chiede la ritrasmissione.

Vediamo un esempio di decodifica tramite sindrome [2, p. 52].

Esempio 13. Consideriamo il codice binario $\mathcal{C}$, un $[4, 2]$-codice di matrice generatrice

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

Sommando alla prima riga di G la seconda, il codice non cambia ma otteniamo una sua matrice generatrice in forma standard:

$$G' = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix} = (I_2|A).$$

Per il Teorema 35, una matrice di controllo per $\mathcal{C}$ è la seguente:

$$H = (-A^t|I_{4-2}) = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$$

Come fatto per costruire la tabella standard, scegliamo una parola leader per ogni laterale tra quelle di peso minimo e calcoliamone la sindrome. Lo spazio è $\mathbb{F}_2^4$, dunque ci sono $2^{4-2} = 4$ laterali distinti. Scegliamo le seguenti parole leader:

$$\begin{aligned} \mathbf{a}_1 &= 0\,0\,0\,0 & \mathbf{s}_1 &= S(\mathbf{a}_1) = \mathbf{a}_1 H^t = 0\,0 \\ \mathbf{a}_2 &= 1\,0\,0\,0 & \mathbf{s}_2 &= S(\mathbf{a}_2) = \mathbf{a}_2 H^t = 0\,1 \\ \mathbf{a}_3 &= 0\,1\,0\,0 & \mathbf{s}_3 &= S(\mathbf{a}_3) = \mathbf{a}_3 H^t = 1\,1 \\ \mathbf{a}_4 &= 0\,0\,1\,0 & \mathbf{s}_4 &= S(\mathbf{a}_4) = \mathbf{a}_4 H^t = 1\,0 \end{aligned}$$

La tabella corrispondente è la riportata nella Tabella 4.1.

Parola leader	**Sindrome**
0 0 0 0	0 0
1 0 0 0	0 1
0 1 0 0	1 1
0 0 1 0	1 0

Tabella 4.1: Sindrome

Sia $\mathbf{y} = (0\,1\,0\,1)$ la sequenza ricevuta; calcoliamo la sua sindrome:

$$S(\mathbf{y}) = \mathbf{y}H^t = (1\,0).$$

Poiché $S(\mathbf{y}) = \mathbf{s}_4$, sappiamo che $\mathbf{y}$ appartiene al laterale la cui parola leader è $\mathbf{a}_4$ e dunque $\mathbf{y}$ viene decodificata con la parola $\mathbf{y} - \mathbf{a}_4 = 0\,1\,1\,1$.

Notiamo che, anche in questo caso, abbiamo effettuato delle scelte arbitrarie sulle parole di peso minimo e per alcuni laterali sono possibili più parole leader. Allo stesso modo della decodifica tramite tabella standard, a seconda dei fini dell'applicazione si sceglierà una decodifica completa o incompleta, richiedendo un'eventuale ritrasmissione della sequenza. Ad ogni modo, l'algoritmo che utilizza la sindrome è più semplice ed efficiente in quanto deve eseguire meno confronti per individuare il dato necessario nella tabella, oltre al fatto che la costruzione e la memorizzazione di una tabella più piccola richiede meno risorse.

Capitolo 5

Esempi di codice lineare

In questo capitolo vediamo alcuni esempi di codici lineari e alcuni casi in cui possono essere applicati [1, par. 1.3].

5.1 Codice di ripetizione

Per individuare e correggere gli errori di trasmissione è necessario aggiungere la ridondanza ai dati da inviare; il modo più immediato di farlo è ripetere più volte il messaggio.

Possiamo descrivere il *codice di ripetizione* $\mathcal{C}$ di lunghezza n su qualsiasi campo finito $\mathbb{F}_q$ attraverso la sua matrice generatrice $G \in M_{1,n}(\mathbb{F}_q)$, con $G = (1\,1\,\ldots\,1)$. Le parole di $\mathcal{C}$ non nulle sono nella forma $(a\,a\,\ldots\,a) \in \mathbb{F}_q^{\times}$. Questo codice ha dimensione 1 e distanza minima n. La sua efficienza, definita nel paragrafo 2.4, è

$$R_n = \frac{\log_q |\mathcal{C}|}{n} = \frac{\log_q q}{n} = \frac{1}{n}.$$

Per i teoremi sulla distanza minima dimostrati nel Capitolo 2, possiamo affermare che $\mathcal{C}$ è un codice $(n-1)$-rivelatore e $\lfloor(\frac{n-1}{2})\rfloor$-correttore.
Osserviamo che $\lim_{n\to+\infty} R_n = 0$, cioè più ripetizioni del messaggio aggiungiamo, più errori il codice sarà in grado di gestire a discapito dell'efficienza. Infatti, più sono lunghi i blocchi, più risorse dovranno essere impiegate per il trasferimento dei dati rendendo il procedimento sempre più lungo e dispendioso.

In ambito informatico questo tipo di codice è uno di quelli che vengono utilizzati per installare diversi dischi rigidi in modo che formino un unico sistema di archiviazione, migliorando l'affidabilità della memorizzazione e la velocità delle operazioni eseguite sui dati rispetto ad una singola unità di archiviazione. Questo sistema è detto RAID, *Redundant Array of Independent Disks*, e può essere costruito in diversi modi. Il caso di nostro interesse è il RAID 1: vengono installati n dischi ciascuno dei quali contiene una copia dei dati. In questo modo possono essere svolte contemporaneamente più

operazioni e il sistema è affidabile anche in caso di danneggiamento di $n-1$ dischi.

5.2 Codice di parità

Un altro modo di inserire la ridondanza è quello di aggiungere un simbolo alla fine del messaggio da trasmettere. Questo simbolo viene calcolato in base ai precedenti e se una volta ricevuto il messaggio lo stesso calcolo fornisce un simbolo diverso, significa che è avvenuto un errore di trasmissione. Se si utilizza un codice binario questo simbolo fa sì che il numero totale di simboli 1 all'interno della parola sia pari ed è detto *bit di parità*.
Il *codice di parità* è l'immagine della seguente funzione di codifica:

$$\begin{array}{ccc} \mathbb{F}_q^{n-1} & \longrightarrow & \mathbb{F}_q^n \\ (x_1,\dots,x_{n-1}) & \longmapsto & (x_1\dots,x_{n-1},-\sum_{i=1}^{n-1} x_i) \end{array}$$

Una matrice generatrice di questo codice G e una matrice di controllo di parità H sono:

$$G=\begin{pmatrix} 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \dots & 0 \\ & & \ddots & \ddots & \\ 0 & \dots & 0 & 1 & -1 \end{pmatrix} \quad \text{e} \quad H=\begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix}.$$

Questo è un $[n, n-1, 2]$-codice e, poiché la distanza minima è 2, non è in grado di correggere nessun errore, mentre può rilevarne solo uno.

Anche questo tipo di codice viene utilizzato in ambito informatico nel sistema RAID, in particolare nel RAID 5. In questo caso i dati vengono distribuiti in $n-1$ dischi rigidi e l'n-esimo disco è il *disco di parità*: l'i-esimo bit del disco di parità è la somma dei bit che occupano la stessa posizione in tutti gli altri $n-1$ dischi. Come per il RAID 1, il RAID 5 aumenta le prestazioni del computer e la sicurezza dei dati.

Osservazione. Notiamo che la matrice di controllo di parità H del codice di parità coincide con la matrice generatrice del codice di ripetizione. Dunque i due codici sono uno il duale dell'altro.

5.3 Codice di Hamming

Una classe di codici lineari di particolare interesse per la rilevazione e la correzione degli errori di trasmissione è formata dai *codici di Hamming*. Per

ogni intero $l \geq 3$, un codice di Hamming è un codice binario definito da una matrice di controllo di parità H_l di dimensioni $l \times (2^l - 1)$, le cui colonne dispongono nella matrice tutti i vettori non nulli di $\mathbb{F}_2^l$.

Ad esempio, per $l = 3$, un codice di Hamming è descritto dalle seguenti matrici, H_3 matrice di controllo di parità e G_3 matrice generatrice:

$$H_3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \quad \text{e} \quad G_3 = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Vediamo i parametri di un codice di Hamming con la seguente proposizione.

Proposizione 40. *Dato un intero $l \geq 3$, i parametri di un codice di Hamming sono $[2^l - 1, 2^l - 1 - l, 3]$.*

Dimostrazione. Dato $l \geq 3$, la matrice di controllo di parità H_l di un codice di Hamming ha dimensioni $l \times (2^l - 1)$. Di conseguenza le parole del codice hanno lunghezza $2^l - 1$.
Per provare che la dimensione del codice è $2^l - 1 - l$, dobbiamo mostrare che la matrice H_l ha rango pieno. Poiché tutti i vettori non nulli di $\mathbb{F}_2^l$ sono distribuiti nelle colonne di H_l, possiamo estrarre l vettori che formano una base di $\mathbb{F}_2^l$. Quindi H_l ha rango l. Ne segue che il codice duale del codice di Hamming ha dimensione l, quindi il codice di Hamming ha dimensione $2^l - 1 - l$.
Per definizione, le colonne di H_l sono tutte non nulle. In particolare, poiché lavoriamo nel campo $\mathbb{F}_2$, prese due colonne qualunque esse non sono collineari. Quindi per il Corollario 37 la distanza minima è almeno 3. Inoltre, possiamo trovare delle terne di colonne linearmente dipendenti: le colonne di H_l sono tutti i vettori non nulli $\mathbb{F}_2^l$ e prese due colonne distinte H_i, H_j, anche $H_i + H_j$ è una colonna di H_l. Per il Teorema 36 la distanza minima è esattamente 3.

□

Poiché tutti i codici di Hamming hanno distanza minima $d = 3$, essi sono tutti codici 1-correttori e 2-rivelatori.
Una proprietà particolare dei codici di Hamming è che sono *perfetti*. Per dimostrarlo, ricordiamo la definizione di codice perfetto data nel Capitolo 2: un codice $\mathcal{C}$ si dice perfetto se vale l'uguaglianza nella stima di Hamming, cioè se

$$|\mathcal{C}| = \frac{q^n}{\sum_{i=0}^{h} \binom{n}{i} (q-1)^i}.$$

Questa definizione di codice perfetto è equivalente alla seguente, i dettagli si possono trovare in [2, par. 2.4].

Definizione 41 (Codice perfetto)**.** Un codice $\mathcal{C}$ di parametri $[n, k, d]$ su $\mathbb{F}_q$ di dice *perfetto* se

$$\mathbb{F}_q^n = \bigcup_{\mathbf{c}\in\mathcal{C}} B\left(\mathbf{c}, \left\lfloor \frac{d-1}{2} \right\rfloor\right)$$

dove $B\big(\mathbf{c}, \lfloor \frac{d-1}{2} \rfloor\big)$ è la sfera di Hamming di centro $\mathbf{c}$ e raggio $\lfloor \frac{d-1}{2} \rfloor$.

Lemma 42. *I codici di Hamming sono perfetti.*

Dimostrazione. Poiché i codici di Hamming hanno distanza minima $d = 3$, abbiamo $\lfloor \frac{d-1}{2} \rfloor = 1$. Se $n = 2^l - 1$ è la lunghezza del codice, la sfera di Hamming di raggio 1 ha cardinalità $n + 1 = 2^l$: infatti, una parola appartenente alla sfera centrata in $\mathbf{c}$ è proprio $\mathbf{c}$ oppure è una parola ottenuta da $\mathbf{c}$ cambiando una sola delle n entrate. Inoltre, sfere di raggio 1 centrate in parole del codice diverse hanno intersezione vuota. Calcoliamo il volume dell'unione disgiunta di queste sfere

$$\sum_{\mathbf{c}\in\mathcal{C}} \left| B\left(\mathbf{c}, \left\lfloor \frac{d-1}{2} \right\rfloor\right)\right| = \sum_{\mathbf{c}\in\mathcal{C}} 2^l = |\mathcal{C}| \cdot 2^l = 2^{2^l-1-l} \cdot 2^l = 2^{2^l-1},$$

che corrisponde al numero totale di elementi di $\mathbb{F}_q^{2^l-1}$. Dunque i codici di Hamming sono perfetti.

□

L'importanza dei codici perfetti risiede nel fatto che esiste un processo di decodifica che restituisce sempre un'unica parola per ogni vettore ricevuto, non c'è dunque l'ambiguità dovuta a delle scelte effettuate; vediamo l'esempio di un codice di Hamming con $l = 3$.

Esempio 14. Consideriamo $\mathcal{C}$ un codice di Hamming di parametri $[7, 4, 3]$. Siano H una sua matrice di controllo di parità e $\mathbf{c} \in \mathcal{C}$. Inoltre, sia $\mathbf{y} = \mathbf{c} + \mathbf{e}$ una parola contenente un errore con $\mathbf{e}$ un vettore di peso 1.
Poiché $\mathbf{e}$ ha peso 1, esso è un vettore della base canonica di $\mathbb{F}_2^7$, diciamo l'i-esimo vettore per qualche $1 \leq i \leq 7$. Essendo $\mathbf{c}$ una parola del codice, $H\mathbf{c}^t = \mathbf{0}$ e ne segue che

$$H\mathbf{y}^t = H(\mathbf{c} + \mathbf{e})^t = H\mathbf{e}^t.$$

$H\mathbf{e}^t$ è l'i-esima colonna di H.
Utilizzando le considerazioni appena fatte possiamo costruire il seguente algoritmo per la correzione di un errore:

Input : una parola $\mathbf{y} \in \mathbb{F}_2^7$ contenente un errore;

Passo 1 : si calcola $\mathbf{s}$=$H\mathbf{y}^t$;

Passo 2 : si trova $i \in \{1, \ldots, 7\}$ tale che $\mathbf{s}$ corrisponda all'i-esima colonna di H;

Output : la parola $\mathbf{y} + \mathbf{e}_i$, con $\mathbf{e}_i$ l'i-esimo vettore della base canonica.

Conclusioni

Nel primo capitolo abbiamo visto che, durante la trasmissione di informazioni, il canale di comunicazione è soggetto a del rumore, cioè dei disturbi e delle interferenze che causano degli errori nei dati inviati. Per contrastare questo problema e rendere efficace la comunicazione, la teoria dei codici studia delle tecniche per individuare e correggere gli eventuali errori.

La definizione di codice, in generale, non dà nessuna informazione sulla struttura di questo insieme di parole. Per le applicazioni, però, è importante implementare degli algoritmi in grado di rilevare e correggere gli errori che siano efficienti e poco costosi in termini di tempo e risorse impiegate. Un codice qualsiasi può non essere adatto alle esigenze. Abbiamo quindi studiato dei codici con una particolare struttura algebrica e i loro parametri in modo da rendere gli algoritmi più semplici ed efficienti.

In particolare, i codici a blocchi, formati da parole tutte della stessa lunghezza, permettono di semplificare la codifica e la decodifica descrivendo il funzionamento di queste operazioni su un singolo blocco.

Aggiungendo un'ulteriore struttura ai codici a blocchi, otteniamo i codici lineari, le cui parole formano uno spazio vettoriale. In questo caso, gli algoritmi sono ancora più semplici ed efficienti in quanto è possibile lavorare sulle basi dello spazio vettoriale o sulle matrici associate. Utilizzando i risultati visti nei capitoli 2 e 3, è inoltre possibile passare da un codice ad uno equivalente in modo da avere la matrice associata nella forma più semplice possibile.

In conclusione i codici lineari permettono di costruire algoritmi per la codifica e la decodifica dei messaggi in grado di rilevare ed eventualmente correggere gli errori di trasmissione in pochi semplici passaggi, rendendo la comunicazione efficace.

Bibliografia

[1] Couvreur A., *Introducting to Coding Theory*, Institut polytechnique de Paris, 16 novembre 2020.

[2] Fiori C., *Lezioni di Teoria dei Codici ed Elementi di Crittografia*, Università di Modena e Reggio Emilia, Dipartimento di Scienze Fisiche, Informatiche, Matematiche, 2016-17.

[3] Giuzzi L., *Codici Correttori*, Springer-Verlag Italia, Milano, 2006.

[4] Hamming R.W., Error detecting and error correcting codes. *Bell System Technical Journal*, 26:147-160, 1950.

[5] Ling, San., *Coding Theory: a First Course*, Cambridge University Press, 2004.

[6] Shannon C.E., Weaver W., *La teoria matematica delle comunicazioni*, Gruppo editoriale Fabbri-Bompiani, Sonzogno, seconda edizione, 1983.

[7] Van Lint J.H., *Introducting to Coding Theory*, Springer, terza edizione, 1999.