Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE

Corso di Laurea triennale in Matematica

IL CODICE DI GOLAY $\mathscr{G}_{24}$ E I t-DESIGNS

Tesi di Laurea in Algebra

Relatore:
Chiar.ma Prof.ssa
MORIGI MARTA

Presentata da:
RAGGI ALESSIA

Anno Accademico 2016/2017

"The fundamental problem of communication
is that of reproducing at one point,
either exactly or approximately,
a message selected at another point"
(C.E. Shannon)

Introduzione

È solo in tempi recenti che ci si è posti il problema della comunicazione sicura, vale a dire della trasmissione di dati o messaggi privati tramite canali protetti, in cui si fosse certi che nessuna interferenza ne modificasse il contenuto. Tali interferenze presenti nel canale di passaggio vengono solitamente chiamate *fonti rumorose*, le quali spesso provocano la perdita di dati, ma più frequentemente modificano la natura del messaggio senza che il ricevente apparentemente ne percepisca il difetto.
La Teoria dei Codici, nata solamente negli anni '40 del Novecento circa, si preoccupa proprio di ovviare a questo problema.
Il processo che sta alla base di questa teoria è molto semplice. L'idea è quella di *codificare* il messaggio in questione, costituito da blocchi di cifre o parole, attraverso una stringa di numeri *ridondante*, cioè composta da più cifre di quelle strettamente necessarie, in modo tale che il ricevitore possa accorgersi di eventuali cambiamenti del formato iniziale e correggere gli errori presenti.
Dal punto di vista matematico, i codici più interessanti sono i *codici lineari*, per i quali esistono specifiche tecniche di codifica/decodifica, nonché di correzione di errori, attuabili semplicemente utilizzando strumenti di algebra lineare.

Lo scopo principale del Primo Capitolo di questa tesi è quello di enunciare e dimostrare le proprietà e i concetti basilari della teoria dei codici lineari. Come prima cosa, verrà schematizzato e tradotto in termini matematici un generico processo di comunicazione tra utenti, per arrivare a dare la definizione di *codice lineare* come spazio vettoriale su un campo finito $\mathscr{F}$. Tratteremo poi nello specifico i parametri che caratterizzano un codice e le più utilizzate tecniche di decodifica e correzione di errori per codici lineari.

Nel Secondo Capitolo, passeremo, quindi, allo studio più approfondito dei cosiddetti *codici perfetti*, e in particolare di due classi di essi: i *Codici di Hamming* e i *Codici di Golay*.
I primi devono il loro nome al matematico americano *Richard Hamming*, fondatore nel 1948 della Teoria della correzione degli errori e primo in assoluto ad occuparsi della costruzione di codici ottimali. Questi si rivelarono fin da subito estremamente adatti nel caso di trasmissioni tramite onde elettromagnetiche o impulsi elettrici.
I secondi, invece, possono essere considerati storicamente tra i codici più rivoluzionari. Rivolgeremo particolare attenzione al Codice di Golay esteso $\mathscr{G}_{24}$, il quale fu ampiamente utilizzato dall'ente spaziale NASA soprattutto negli anni '80 nelle trasmissioni delle sonde Voyager.
Infine, introdurremo semplici ed utili strutture di carattere geometrico-combinatorico dalle applicazioni più disparate, chiamate t-designs. Il risultato conclusivo mette in relazione queste ultime, in particolare i cosiddetti sistemi di Steiner, con i Codici di Golay e di Hamming.

Indice

Capitolo 1

Codici lineari

1.1 Concetti di base

In questa prima sezione ci occupiamo di dare alcune importanti definizioni, basilari per poter procedere allo studio della Teoria dei Codici lineari nel suo complesso.

Il primo obiettivo che ci poniamo è quello di tradurre in termini matematici le varie fasi di un processo generico di comunicazione. Fissiamo quindi le seguenti notazioni:

- vettore del messaggio inviato: $\boldsymbol{u} = u_1 u_2 \cdots u_k$ dove $u_i \in \{0, 1\}$, $i \in \{1, \ldots, k\}$.
- messaggio codificato o parola: $\boldsymbol{x} = x_1 x_2 \cdots x_n$ dove $x_i \in \{0, 1\}$, $i \in \{1, \ldots, n\}$.
- vettore di errore: $\boldsymbol{e} = e_1 e_2 \cdots e_n$ dove $e_i \in \{0, 1\}$, $i \in \{1, \ldots, n\}$.
- vettore del messaggio ricevuto: $\boldsymbol{y} = x + e \pmod 2$.
- vettore del messaggio finale stimato: $\hat{\boldsymbol{u}}$.

Come ben esplicitato visivamente nella figura 1.1, il meccanismo funziona in questo modo: il messaggio $\boldsymbol{u}$ viene codificato tramite una funzione lineare "Encoder" e trasformato in una parola $\boldsymbol{x}$, la quale viene successivamente trasmessa tramite il canale. Qui, se il canale è appunto *rumoroso*, può accadere che un vettore di errore $\boldsymbol{e}$ ne modifichi la natura, alterando la ricezione del messaggio di partenza in $\boldsymbol{y}$. La funzione lineare

"Decoder" ha infine lo scopo di tradurre ciò che ha ricevuto in una stima $\hat{\boldsymbol{u}}$ quanto più vicina all'originale (*MLD, maximum likelihood decoding*), minimizzando la probabilità di incorrere in errori.

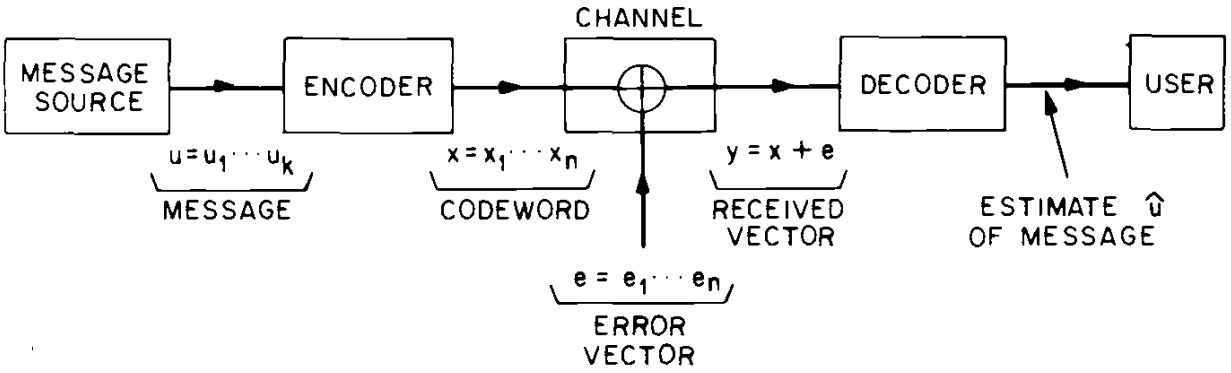

Figura 1.1: Sistema di comunicazione.

Nel linguaggio della teoria dei codici, un insieme finito $\mathscr{F}$ con q elementi è detto *alfabeto finito con q lettere*, dove le lettere sono appunto gli elementi di $\mathscr{F}$.
Una *parola* o *codeword* su $\mathscr{F}$ di lunghezza n è una successione finita $x_1 \dots x_n$ di lettere di $\mathscr{F}$. Tale successione può, più comodamente, essere identificata con una n-pla $\boldsymbol{x} = (x_1, \dots, x_n)$. In questo modo ogni parola di lunghezza n potrà essere vista come un elemento di $\mathscr{F}^n$.
Un'osservazione importante da fare riguarda la natura della generica parola $\boldsymbol{x}$. Essa infatti consiste di due parti ben distinte: i primi k simboli compongono ordinatamente il messaggio $\boldsymbol{u}$ stesso

$$x_1 = u_1,\ x_2 = u_2,\ \dots,\ x_k = u_k,$$

mentre i restanti $n - k$ sono i cosiddetti *simboli di controllo*

$$x_{k+1},\ \dots\ , x_n.$$

Questi ultimi non sono casuali, ma sono scelti in modo tale da soddisfare la relazione

$$H \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = H\boldsymbol{x}^T = \mathbf{0}, \tag{1.1}$$

dove H è una matrice di ordine $(n-k) \times n$ così composta:

$$H = [\ A \mid I_{n-k}\ ] \tag{1.2}$$

dove A è a sua volta una matrice fissata di ordine $(n-k) \times k$, e I_{n-k} è la matrice identità di ordine $(n-k) \times (n-k)$.

Definizione 1.1. La matrice H dell'equazione 1.2 è detta *matrice di controllo di parità*. Di conseguenza, chiamiamo le equazioni che compongono il sistema lineare 1.1 le *equazioni di controllo di parità*.

Definizione 1.2. Un *codice* $\mathscr{C}$ su un alfabeto $\mathscr{F}$ è un qualsiasi sottoinsieme finito e non vuoto di parole su $\mathscr{F}$.
Esso è chiamato *codice a blocchi* se le sue parole hanno tutte stessa lunghezza (in tal caso si parlerà di lunghezza del codice); in caso contrario è detto *codice a lunghezza variabile*.

Definizione 1.3. Sia q una potenza di un primo. Consideriamo come alfabeto $\mathscr{F}$ il campo finito avente q elementi $\mathscr{F}_q$. Un codice $\mathscr{C}$ su $\mathscr{F}$ è detto *lineare* se costituisce un sottospazio vettoriale di $\mathscr{F}^n$.

In questo elaborato ci occuperemo soltanto di codici lineari a blocchi binari, formati cioè da parole della stessa lunghezza ad elementi su un campo finito avente due soli elementi (che saranno denotati con 0 e 1).

A questo punto, combinando le relazioni appena viste, otteniamo

$$\begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix} = I_k \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix} \tag{1.3}$$

poiché i primi k simboli della parola $\boldsymbol{x}$ rappresentano il messaggio $\boldsymbol{u}$, e inoltre

$$\begin{gathered} [\ A \mid I_{n-k}\ ] \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = 0, \\ \begin{pmatrix} x_{k+1} \\ \vdots \\ x_n \end{pmatrix} = -A \begin{pmatrix} x_1 \\ \vdots \\ x_k \end{pmatrix}, \end{gathered} \tag{1.4}$$

ossia,

$$\begin{pmatrix} x_{k+1} \\ \vdots \\ x_n \end{pmatrix} = -A \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix}. \tag{1.5}$$

Notiamo che nel nostro caso, quello cioè di un campo finito di caratteristica 2, vale $-A = A$, perciò accostando una sopra l'altra le equazioni 1.3 e 1.5 otteniamo

$$\begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{bmatrix} I_k \\ -A \end{bmatrix} \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix}. \tag{1.6}$$

Ora, trasponendo, si ha

$$\boldsymbol{x} = \boldsymbol{u}G, \tag{1.7}$$

dove

$$G = [\ I_k \mid -A^T\ ]. \tag{1.8}$$

Definizione 1.4. La matrice G dell'equazione 1.8 è detta *matrice generatice* del codice.

Osservazione 1. L'equazione 1.7 assicura che le possibili parole $\boldsymbol{x}$ relative ad un dato messaggio $\boldsymbol{u}$ sono tutte e sole le combinazioni lineari delle righe di G.

Osservazione 2. Le equazioni 1.1 e 1.7 mostrano che le matrici H e G sono legate dalla relazione

$$GH^T = 0 \quad \text{o} \quad HG^T = 0. \tag{1.9}$$

Definizione 1.5. La *distanza di Hamming* tra due vettori $\boldsymbol{x} = x_1x_2\cdots x_n$ e $\boldsymbol{y} = y_1y_2\cdots y_n$ conta il numero di posti per cui i due differiscono ed è denotata da $\text{dist}(\boldsymbol{x}, \boldsymbol{y})$.

Definizione 1.6. Il *peso di Hamming* $w(\boldsymbol{x})$ di un vettore $\boldsymbol{x} = x_1x_2\cdots x_n$ è il numero delle componenti non nulle del vettore stesso.

Osservazione 3. Segue banalmente dalle definizioni 1.5 e 1.6 che

$$\text{dist}(\boldsymbol{x}, \boldsymbol{y}) = w(\boldsymbol{x} - \boldsymbol{y})$$

poiché entrambi i membri dell'equazione contano il numero di posti per cui $\boldsymbol{x}$ e $\boldsymbol{y}$ differiscono.

Definizione 1.7. Si definisce *minima distanza di Hamming* tra le parole di un codice $\mathscr{C}$ la quantità $d = \min\{\text{dist}(\boldsymbol{x}, \boldsymbol{y}) |\ \boldsymbol{x} \neq \boldsymbol{y}\}$.

Per trovare la minima distanza di un codice lineare non occorre necessariamente comparare ogni paio di parole che compongono il codice stesso, infatti:

Teorema 1.1.1. *La minima distanza di Hamming per un codice lineare $\mathscr{C}$ equivale al minimo peso di una qualsiasi parola non nulla:*

$$d = \min_{\boldsymbol{z} \in \mathscr{C}, \boldsymbol{z} \neq 0} w(\boldsymbol{z}).$$

Dimostrazione. La dimostrazione deriva direttamente dall'osservazione 3. Infatti,

$$d = \min_{\boldsymbol{x}, \boldsymbol{y} \in \mathscr{C}, \boldsymbol{x} \neq \boldsymbol{y}} \text{dist}(\boldsymbol{x}, \boldsymbol{y}) = \min_{\boldsymbol{x}, \boldsymbol{y} \in \mathscr{C}, \boldsymbol{x} \neq \boldsymbol{y}} w(\boldsymbol{x} - \boldsymbol{y})$$

ma per linearità del codice $\boldsymbol{x} - \boldsymbol{y} = \boldsymbol{z} \in \mathscr{C}$ perciò:

$$d = \min_{\boldsymbol{z} \in \mathscr{C}, \boldsymbol{z} \neq 0} w(\boldsymbol{z}).$$

□

Osservazione 4. Siano $\boldsymbol{x}, \boldsymbol{y} \in \mathscr{C}$. Allora, per banali considerazioni riguardo la somma in un campo avente due elementi, vale la relazione

$$w(\boldsymbol{x} + \boldsymbol{y}) = w(\boldsymbol{x}) + w(\boldsymbol{y}) - 2\#\{i \mid x_i = y_i = 1\}.$$

Lemma 1.1.2. *In un codice lineare binario $\mathscr{C}$, o tutte le parole hanno peso pari, oppure esattamente metà hanno peso pari e metà hanno peso dispari.*

Dimostrazione. Le parole di un codice sono combinazioni lineari delle righe della matrice generatrice G. Se queste ultime hanno tutte peso pari, qualsiasi altra parola del codice avrà peso pari, come assicura la relazione dell'osservazione 4.
Supponiamo ora che esista almeno una parola $\boldsymbol{y}$ appartenente al codice di peso dispari. Sia $\boldsymbol{x}$ un'altra parola del codice. Se $\boldsymbol{x}$ ha peso pari, allora $w(\boldsymbol{x}+\boldsymbol{y})$ è dispari, mentre se $\boldsymbol{x}$ ha peso dispari, allora $w(\boldsymbol{x}+\boldsymbol{y})$ è pari, sempre sostituendo nella formula dell'osservazione 4.
Se consideriamo quindi la funzione $f(\boldsymbol{x}) = \boldsymbol{x}+\boldsymbol{y}$, essa definisce una biiezione tra l'insieme delle parole di peso pari e quelle di peso dispari del codice. Quindi i due sottoinsiemi devono avere stessa cardinalità. □

1.2 Proprietà dei codici lineari

In questa sezione vediamo alcune delle più importanti proprietà dei codici lineari.

1. *Parametri di un codice lineare.* Un qualsiasi codice lineare $\mathscr{C}$ è caratterizzato da:

   - *lunghezza n*: numero di simboli (=elementi) di cui è composta ogni parola del codice.
   - *dimensione k*: numero di simboli che fanno effettivamente parte del messaggio (notiamo che necessariamente $n \geq k$).
   - *indice di informazione*: indica il rapporto $R = k/n$.

   Parleremo in questo caso di $[n, k]$-codice.

2. *Matrici generatrici di un codice lineare.* Un codice lineare $\mathscr{C}$ può avere numerose matrici generatrici diverse tra loro: infatti ogni famiglia massimale di parole del codice linearmente indipendenti può essere usata per formare le righe di tale matrice. Nel nostro caso, quello di un campo binario, tali matrici sono interamente composte da soli 0 e 1.

3. *Codici su campi finiti.* Un codice generico può assumere valori su un qualsiasi campo finito, perciò oltre ad utilizzare soltanto 0 e 1, sono ammessi anche altri simboli. Un esempio semplice ma comune è quello dei *codici ternari* composti dai simboli 0, 1 e 2.

4. *Linearità.* Siano $\boldsymbol{x}$ e $\boldsymbol{y}$ parole appartenenti al codice $\mathscr{C}$, e sia c elemento del campo $\mathscr{F}$, allora vale:

$$\boldsymbol{x} + \boldsymbol{y} \text{ è ancora una parola di } \mathscr{C}$$

$$c \cdot \boldsymbol{x} \text{ è ancora una parola di } \mathscr{C}$$

   Questo è il motivo principale per cui tali codici sono detti lineari.

Esempio 1.1. Consideriamo la matrice di parità

$$H = \left[\begin{array}{ccc|ccc} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{array}\right].$$

Essa definisce un codice con parametri $k = 3$ e $n = 6$. In questo caso

$$A = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$

Il generico messaggio $\boldsymbol{u} = u_1u_2u_3$ viene codificato nella parola $\boldsymbol{x} = x_1x_2x_3x_4x_5x_6$ così composta:

$$x_1 = u_1, \qquad x_2 = u_2, \qquad x_3 = u_3,$$

mentre i simboli x_4, x_5, x_6 sono determinati dalla relazione $H\boldsymbol{x}^T = \mathbf{0}$ (1.1) e cioè si ricavano dal sistema lineare:

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

Quindi se consideriamo, ad esempio, il messaggio $\boldsymbol{u} = 011$, si ha $x_1 = 0, x_2 = 1, x_3 = 1$, mentre i simboli di controllo di parità risultano essere

$$\begin{cases} x_4 = -1 - 1 = 1 + 1 = 0 \\ x_5 = -1 = 1 \\ x_6 = -1 = 1 \end{cases}.$$

Cioè la parola diventa $\boldsymbol{x} = 011011$.
Poiché gli unici simboli liberi nel nostro caso sono quelli che costituiscono il messaggio vero e proprio x_1, x_2, x_3, mentre x_4, x_5, x_6 sono vincolati dalle equazioni di controllo parità, esistono $2^3 = 8$ possibili parole appartenenti a questo codice, che sono:

$$0000000,\ 011011,\ 110110,\ 001110,\ 100011,\ 111000,\ 010101,\ 101101.$$

Una matrice generatrice G per questo codice può essere

$$G = \left[I_3 \quad \middle| - A^T \right] = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{array}\right].$$

Possiamo ancora notare che la parola corrispondente al messaggio $\boldsymbol{u} = 011$ considerato in precedenza, ma tramite l'equazione 1.7, è proprio

$$\boldsymbol{x} = \boldsymbol{u}G = 010101 + 001110 = 011011.$$

1.3 Codici duali

Definizione 1.8. Siano $\boldsymbol{u} = (u_1, \ldots, u_n)$ e $\boldsymbol{v} = (v_1, \ldots, v_n)$ vettori di lunghezza n a elementi in uno stesso campo $\mathscr{F}$. Il *prodotto scalare* tra i due è dato da:

$$\boldsymbol{u} \cdot \boldsymbol{v} = u_1 v_1 + \ldots + u_n v_n.$$

Nel caso che ci interessa, cioè quello di vettori binari con componenti pari a 0 o 1, il risultato appartiene al campo con due elementi (0 o 1).

Definizione 1.9. Se $\boldsymbol{u} \cdot \boldsymbol{v} = 0$ allora $\boldsymbol{u}$ e $\boldsymbol{v}$ sono detti *ortogonali* tra loro.

Definizione 1.10. Dato un $[n, k]$-codice lineare $\mathscr{C}$ su un campo $\mathscr{F}$, il suo *duale* o *ortogonale* $\mathscr{C}^\perp$ è l'insieme dei vettori ortogonali (nel senso del prodotto scalare visto in precedenza) a tutte le possibili parole di $\mathscr{C}$:

$$\mathscr{C}^\perp = \{\boldsymbol{u} \mid \boldsymbol{u} \cdot \boldsymbol{v} = 0 \text{ per ogni } \boldsymbol{v} \in \mathscr{C}\}.$$

Di conseguenza, $\mathscr{C}^\perp$ è esattamente l'insieme di tutte le equazioni di controllo di parità di $\mathscr{C}$.

Osservazione 5. Se $\mathscr{C}$ ha come matrice generatrice G e matrice di parità H, allora $\mathscr{C}^\perp$ ha come matrice generatrice H, e matrice di parità G. In sostanza, $\mathscr{C}$ e $\mathscr{C}^\perp$ si scambiano di ruolo le matrici G e H.

Definizione 1.11. Un codice $\mathscr{C}$ si dice *debolmente autoduale* se $\mathscr{C} \subseteq \mathscr{C}^\perp$.

Fissiamo di seguito le seguenti notazioni: siano $\boldsymbol{x} = (x_1, \cdots, x_n)$ e $\boldsymbol{y} = (y_1, \cdots, y_n)$. Indichiamo con $\cdot$ il prodotto scalare nel campo con due elementi, cioè $\boldsymbol{x} \cdot \boldsymbol{y} = x_1 y_1 + \cdots + x_n y_n \in \{0, 1\}$; denotiamo invece con $\star$ il prodotto scalare reale usuale, mentre usiamo $*$ per il prodotto componente per componente, cioè $\boldsymbol{x} * \boldsymbol{y} = (x_1 y_1, \cdots, x_n y_n)$.

Lemma 1.3.1. *Siano $\boldsymbol{x}$ e $\boldsymbol{y}$ due vettori binari. Allora $\boldsymbol{x} \cdot \boldsymbol{y} = 0$ se e solo se $w(\boldsymbol{x} * \boldsymbol{y})$ è pari. Analogamente, $\boldsymbol{x} \cdot \boldsymbol{y} = 1$ se e solo se $w(\boldsymbol{x} * \boldsymbol{y})$ è dispari.*

Dimostrazione. La dimostrazione è immediata poiché:

$$w(\boldsymbol{x} * \boldsymbol{y}) = w(x_1 y_1, \cdots, x_n y_n) = x_1 y_1 + \cdots + x_n y_n = \boldsymbol{x} \star \boldsymbol{y}.$$

Spostandosi ora in un campo di caratteristica 2, il prodotto scalare reale $\star$ diventa il prodotto scalare $\cdot$, e il valore del peso $w(\boldsymbol{x} * \boldsymbol{y})$ diventa 0 se pari, 1 se dispari. □

Lemma 1.3.2. *Se $\mathscr{C}$ è un codice binario debolmente autoduale, ogni parola del codice ha peso pari.*

Dimostrazione. Essendo il codice debolmente autoduale, per ogni $\boldsymbol{x}$ e $\boldsymbol{y}$ parola di $\mathscr{C}$ vale la relazione $\boldsymbol{x} \cdot \boldsymbol{y} = 0$.
Per il lemma 1.3.1 vale che $\boldsymbol{x} \cdot \boldsymbol{y} = 0$ se e solo se $w(\boldsymbol{x} * \boldsymbol{y})$ è pari.
In particolare, $\boldsymbol{x} \cdot \boldsymbol{x} = 0$ implica che $w(\boldsymbol{x} * \boldsymbol{x})$ è pari. Ma $\boldsymbol{x} * \boldsymbol{x} = \boldsymbol{x}$ per ovvie ragioni e così otteniamo la tesi. □

Lemma 1.3.3. *Sia $\mathscr{C}$ un codice binario debolmente autoduale. Se ogni riga della sua matrice generatrice ha peso divisibile per 4, allora ogni parola del codice ha peso divisibile per 4.*

Dimostrazione. Sia G la matrice generatrice del codice $\mathscr{C}$. Per ipotesi sappiamo che le sue righe, che chiameremo $v_i, i \in \{1, \cdots, n\}$, hanno peso divisibile per 4.
Secondo la relazione 1.7, una qualsiasi parola $\boldsymbol{x}$ del codice si può scrivere come combinazione lineare delle righe di G:

$$\boldsymbol{x} = \sum_i \lambda_i \boldsymbol{v_i} \qquad \lambda_i \in \{0, 1\}.$$

Denotando con $\star$ il prodotto scalare reale e con $*$ il prodotto componente per componente, si ha:

$$w(\boldsymbol{x}) = \boldsymbol{x} \star \boldsymbol{x} = \left(\sum_i \lambda_i \boldsymbol{v_i} \right) \star \left(\sum_i \lambda_i \boldsymbol{v_i} \right) = \sum_{i,j} \lambda_i \lambda_j \boldsymbol{v_i} \star \boldsymbol{v_j} = \sum_{i<j} 2\lambda_i \lambda_j \boldsymbol{v_i} \star \boldsymbol{v_j} +$$

$$+ \sum_i \lambda_i \lambda_j \boldsymbol{v_i} \star \boldsymbol{v_i} = \sum_{i<j} 2\lambda_i \lambda_j \underbrace{w(\boldsymbol{v_i} * \boldsymbol{v_j})}_{\text{divisibile per 2 per 1.3.2}} + \sum_i \lambda_i \lambda_i \underbrace{w(\boldsymbol{v_i})}_{\text{divisibile per 4 per ipotesi}} .$$

Concludiamo che il peso di ogni parola di $\mathscr{C}$ è divisibile per 4. □

1.4 Metodi di decodifica

A questo punto è arrivato il momento di analizzare nel concreto alcuni dei più utilizzati metodi di decodifica e/o correzione di errori per codici lineari. Ovviamente nel caso remoto che non vi siano interferenze di alcun tipo nel canale di trasmissione, queste tecniche sono sostanzialmente inutili e non necessarie poiché il vettore ricevuto $\boldsymbol{y}$ coincide esattamente con quello inviato $\boldsymbol{x}$.
Vediamo dal seguente esempio come la riuscita dell'individuazione dell'errore e della correzione di quest'ultimo siano caratteristiche intrinseche al codice, in quanto dipendono soltanto dai parametri dello stesso.

Esempio 1.2. Supponiamo, per semplicità, di disporre di un ipotetico canale di trasmissione con la proprietà di modificare al più un simbolo di ogni vettore binario, di lunghezza non superiore a 5. Supponiamo di dover trasmettere un messaggio tra i seguenti quattro:

$$\text{NORD} = 00,\ \text{SUD} = 01,\ \text{EST} = 10,\ \text{OVEST} = 11.$$

Proviamo ad utilizzare il codice $\mathscr{C}_2$ di lunghezza 3:

$$\mathscr{C}_2 \coloneqq \{000, 011, 101, 110\}$$

ponendo

$$\text{NORD} = 000,\ \text{SUD} = 011,\ \text{EST} = 101,\ \text{OVEST} = 110.$$

In questo caso, se su una parola di $\mathscr{C}_2$ si commette un errore, questo viene rilevato dal codice, ma non corretto. Infatti, se ad esempio 000 si trasforma in 100, il destinatario si accorge della presenza dell'errore in quanto la nuova parola non appartiene al codice, ma non riesce a risalire a quella corretta, in quanto $\mathscr{C}_2$ contiene più di una parola che con il cambio di un'unica lettera può diventare 100.
Utilizziamo ora il codice $\mathscr{C}_3$ di lunghezza 5:

$$\mathscr{C}_3 \coloneqq \{00000 = \text{NORD},\ 01101 = \text{SUD},\ 10110 = \text{EST},\ 11011 = \text{OVEST}\}.$$

Ora, si verifica facilmente che se una parola $\boldsymbol{x} \in \mathscr{C}_3$ si trasforma in $\boldsymbol{y}$ mediante la modifica di un simbolo soltanto, non solo l'errore viene rilevato dal codice, ma è possibile anche correggerlo in quanto esiste solo un modo coerente per farlo.

Decodifica secondo l'intorno più vicino

Questa tecnica consiste nel tentativo di correggere gli errori riconducendosi alla parola del codice più "vicina", o più simile. Per la riuscita del risultato, si richiede però come ipotesi che il vettore pervenuto dopo la trasmissione (perturbato di errori) non sia diventato un'altra parola del codice di partenza, altrimenti risulterebbe impossibile l'individuazione. Vediamo nello specifico come funziona questo metodo direttamente dalla dimostrazione del seguente risultato.

Teorema 1.4.1. *Un codice con distanza minima di Hamming pari a d riesce a correggere* $\lfloor \frac{1}{2}(d-1) \rfloor$ *errori.*
Inoltre, se d è pari, allora il codice riesce a rilevare $\frac{d}{2}$ *errori ma a correggerne soltanto* $\frac{1}{2}(d-2)$.

Dimostrazione. Iniziamo supponendo $d = 3$ per esporre l'idea generale (figura 1.2).
La palla $B_r(\boldsymbol{u})$ di raggio r e centro $\boldsymbol{u}$ contiene tutti i vettori $\boldsymbol{w}$ tali che $d(\boldsymbol{u}, \boldsymbol{w}) \leq r$. Se disegniamo una palla di raggio 1 intorno ad ogni parola del codice, notiamo che queste sfere non si possono intersecare. Ciò significa che se nella parola trasmessa $\boldsymbol{u}$ si presenta un solo errore, allora il vettore ricevuto $\boldsymbol{a}$ è dentro la palla centrata in $\boldsymbol{u}$ e raggio 1, cioè la sua parola più vicina è ancora $\boldsymbol{u}$. In questo caso, l'errore viene corretto.

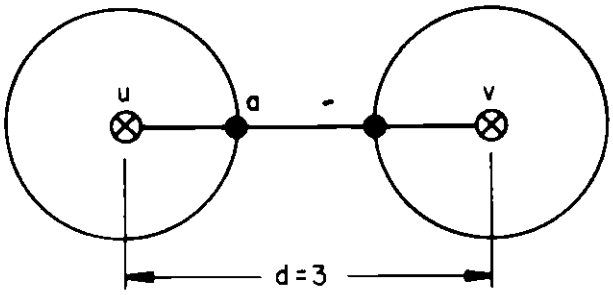

Figura 1.2: Codice con distanza minima $d = 3$.

Più in generale, nel caso $d = 2t + 1$, le palle di raggio t disegnate attorno ad ogni parola del codice non si intersecano, e questo implica che al massimo il codice può individuare e correggere $t = \lfloor \frac{d-1}{2} \rfloor$ errori.

Per quanto riguarda la seconda parte del teorema, supponiamo d pari (figura 1.3). Il fatto che il codice riesca a correggere esattamente $\frac{d-2}{2}$ errori è banale conseguenza di quanto visto precedentemente, con d della forma $d = 2t+2$. Ma se si presentano proprio $d/2$ errori, il vettore ricevuto $\boldsymbol{a}$ potrebbe trovarsi esattamente a metà distanza tra due parole $\boldsymbol{u}$ e $\boldsymbol{v}$ del codice, risultando impossibile capire quale delle due è effettivamente quella esatta. In questo caso, il codice è solo in grado di rilevare la presenza dell'errore senza però sapere come correggerlo. □

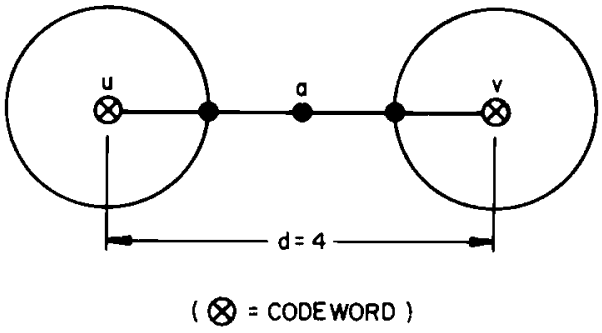

Figura 1.3: Codice con distanza minima $d = 4$.

Il teorema appena dimostrato legittima dunque la seguente definizione:

Definizione 1.12. Si dice che un codice $\mathscr{C}$ *rileva* t errori, dove t è un intero positivo, se per ogni $\boldsymbol{x} \in \mathscr{C}$, la palla centrata in $\boldsymbol{x}$ e di raggio t ha in comune con $\mathscr{C}$ la sola parola $\boldsymbol{x}$. Diciamo invece che $\mathscr{C}$ *corregge* t errori se rileva t errori e due qualsiasi sfere di raggio t centrate in parole distinte di $\mathscr{C}$ hanno intersezione vuota. In tal caso $\mathscr{C}$ sarà detto codice *t-correttore*.

Decodifica secondo la tabella standard

Definizione 1.13. Sia $\mathscr{C}$ un $[n, k]$-codice lineare. Per ogni vettore $\boldsymbol{a}$, chiamiamo *classe* la collezione

$$\boldsymbol{a} + \mathscr{C} = \{\boldsymbol{a} + \boldsymbol{x} : \boldsymbol{x} \in \mathscr{C}\}.$$

Ogni vettore $\boldsymbol{b}$ appartiene ad una determinata classe, e diciamo che $\boldsymbol{a}$ e $\boldsymbol{b}$ sono nella stessa classe se e solo se $(\boldsymbol{a} - \boldsymbol{b}) \in \mathscr{C}$.

Proposizione 1.4.2. *Due classi relative ad un codice $\mathscr{C}$ o sono disgiunte o coincidono, non possono intersecarsi solo parzialmente.*

Dimostrazione. La dimostrazione è immediata, poiché basta notare che la definizione 1.13 è quella di classe laterale usuale, interpretando il codice $\mathscr{C}$ come un sottogruppo del gruppo additivo di uno spazio vettoriale sul campo con due elementi. □

Proposizione 1.4.3. *Sia $\boldsymbol{y}$ il vettore perturbato, ricevuto dopo la trasmissione della parola $\boldsymbol{x}$. I vettori di errore $\boldsymbol{e}$ che possono presentarsi sono tutti e soli quelli contenuti nella classe di $\boldsymbol{y}$.*

Dimostrazione. Supponiamo che $\boldsymbol{y}$ appartenga ad una certa classe $\boldsymbol{a} + \mathscr{C}$. Allora esiste $\boldsymbol{z} \in \mathscr{C}$ tale che $\boldsymbol{y} = \boldsymbol{a} + \boldsymbol{z}$. Sia $\boldsymbol{x}$ la codeword trasmessa in partenza, allora:

$$\boldsymbol{e} = \boldsymbol{y} - \boldsymbol{x} = \boldsymbol{a} + \boldsymbol{z} - \boldsymbol{x} = \boldsymbol{z} + \boldsymbol{x'} \in \boldsymbol{a} + \mathscr{C}.$$

□

Definizione 1.14. Il vettore di minimo peso di una determinata classe è chiamato *leader* della classe, e nel seguito denoteremo quello relativo all'i-esima classe con $\boldsymbol{a_i}$. Se ce ne sono numerosi, tutti con lo stesso peso minimo, si sceglie casualmente uno di essi come leader.

Alla luce di quanto abbiamo visto, possiamo affermare che lo scopo principale di questa strategia consiste nello scegliere, all'interno della stessa *classe* di appartenenza del vettore ricevuto $\boldsymbol{y}$, il vettore di errore $\hat{\boldsymbol{e}}$ di minimo peso possibile (vettore leader della classe) e di conseguenza decifrare $\boldsymbol{y}$ come $\hat{\boldsymbol{x}} = \boldsymbol{y} - \hat{\boldsymbol{e}}$. Come riuscirci?

Una maniera utile per descrivere questo processo è tramite una tabella, detta appunto **matrice standard**, avente sulla prima riga come primo elemento il vettore nullo, seguito dalle parole del codice $\mathscr{C}$, mentre sulle successive i leader $\boldsymbol{a_i}$ di tutte le possibili classi come primo elemento di riga, seguite dalle somme $\boldsymbol{a_i} + \mathscr{C}$. Un algoritmo efficiente nella scelta dei leader di classe consiste nello scegliere dapprima a_1 di peso minimo in $\mathscr{F}_2^n \setminus \mathscr{C}$, poi scegliere a_2 di peso minimo in $\mathscr{F}_2^n \setminus \{\mathscr{C} \cup (a_2 + \mathscr{C})\}$, e così via fino all'esaurimento delle parole in $\mathscr{F}_2^n$.
Si ottiene, infine, una matrice così composta:

1° riga: $\quad \boldsymbol{x}^{(1)} = \mathbf{0}, \quad \boldsymbol{x}^{(2)}, \quad \ldots \quad , \quad \boldsymbol{x}^{(s)}$

i-esima riga: $\quad \boldsymbol{a_i} + \boldsymbol{x}^{(1)}, \quad \boldsymbol{a_i} + \boldsymbol{x}^{(2)}, \quad \ldots \quad , \quad \boldsymbol{a_i} + \boldsymbol{x}^{(s)}$

Vediamo nello specifico il funzionamento del metodo su un esempio.

Esempio 1.3. Sia $\mathscr{C} = \{0000, 1011, 0101, 1110\}$ un [4,2]-codice binario lineare, con matrice generatrice

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

La tabella standard di $\mathscr{C}$ è

codice:	0000	1011	0101	1110
classe:	1000	0011	1101	0110
classe:	0100	1111	0001	1010
classe:	0010	1001	0111	1100

Tabella 1.1: Esempio di tabella standard.

Ecco come viene utilizzata: supponiamo di ricevere $\boldsymbol{y} = 1111$. Una volta individuata la sua posizione nella tabella, si risale al vettore leader $\hat{\boldsymbol{e}}$ situato all'estrema sinistra di $\boldsymbol{y}$, nel nostro caso 0100. Infine $\boldsymbol{y}$ viene decodificato come $\hat{\boldsymbol{x}} = \boldsymbol{y} - \hat{\boldsymbol{e}} = 1011$ che corrisponde proprio alla parola del codice situata in cima alla colonna di $\boldsymbol{y}$.
Risaliamo poi al messaggio corrispondente tramite l'equazione 1.7, cioè 10.

Decodifica tramite sindrome

Un'altra strategia utilizzabile, al posto della matrice standard, per individuare il vettore leader della classe di $\boldsymbol{y}$ è quella che utilizza la cosiddetta **sindrome**.

Definizione 1.15. Definiamo *sindrome* di $\boldsymbol{y} \in \mathscr{F}^n$, il vettore

$$\boldsymbol{S} = H\boldsymbol{y}^T.$$

Proprietà della sindrome

1. $\boldsymbol{S}$ si presenta come un vettore colonna di lunghezza $n-k$ (cioè $\boldsymbol{S} \in \mathscr{F}^{n-k}$).

2. $\boldsymbol{S}$ è nulla se e solo se $\boldsymbol{y}$ è effettivamente una parola del codice. Quindi se non si presentano errori, la sindrome risulta pari a zero. Infatti:

   $$\boldsymbol{S} = H\boldsymbol{y}^T = H\boldsymbol{x}^T + H\boldsymbol{e}^T = H\boldsymbol{e}^T$$

   dove $\boldsymbol{e}$ è il vettore di errore, $\boldsymbol{x} \in \mathscr{C}$ tale che $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{e}$.

3. $\boldsymbol{S}$ coincide con la somma delle colonne della matrice H in cui è presente almeno un errore. (D'altra parte questo è il motivo per cui è chiamata proprio sindrome, in quanto fornisce i "sintomi" degli errori presenti). In simboli:

   $$\boldsymbol{S} = \sum_i e_i \boldsymbol{H_i}$$

   dove $\boldsymbol{H_i}$ è la i-esima colonna di H, mentre e_i è l'elemento di posto i nel vettore di errore $\boldsymbol{e}$.
   Di conseguenza, sono significativi solo gli elementi non nulli di $\boldsymbol{e}$, mentre gli zeri annullano qualsiasi contributo di H. Quindi se ad esempio:

   $$\boldsymbol{e} = 0010011$$

   allora

   $$\boldsymbol{S} = \boldsymbol{H_3} + \boldsymbol{H_6} + \boldsymbol{H_7}.$$

4. Esiste una corrispondenza biunivoca tra sindromi e classi, cioè due vettori appartengono alla stessa classe se e solo se possiedono la stessa sindrome.
   Infatti $\boldsymbol{u}$ e $\boldsymbol{v}$ stanno nella stessa classe di $\mathscr{C}$ se e solo se $(\boldsymbol{u}-\boldsymbol{v}) \in \mathscr{C}$, se e solo se $H(\boldsymbol{u}-\boldsymbol{v})^T = 0$, se e solo se $H\boldsymbol{u}^T = H\boldsymbol{v}^T$.

La proprietà (4) ci fornisce lo strumento principale per risolvere il nostro problema: l'idea è quella di costruire una tabella in cui la prima colonna coincide con quella della matrice standard, mentre la seconda (e ultima) colonna è composta dalle sindromi che caratterizzano univocamente le varie classi. Infatti, sappiamo che necessariamente $\boldsymbol{y}$ e il rappresentante leader della medesima classe di $\boldsymbol{y}$, devono avere stessa sindrome $\boldsymbol{S}$. Di conseguenza, una volta calcolata $\boldsymbol{S}$ per $\boldsymbol{y}$, ci si riconduce al leader $\boldsymbol{a_i}$ della classe avente stessa sindrome e si decodifica la parola come $\boldsymbol{y}-\boldsymbol{a_i}$.
Riprendendo l'esempio 1.3. La tabella diventa:

	leader	sindrome $\boldsymbol{S}$
riga nulla:	0000	$(0,0)$
classe:	1000	$(1,1)$
classe:	0100	$(0,1)$
classe:	0010	$(1,0)$

Tabella 1.2: Esempio di tabella con sindromi.

Risulta chiaro che questo metodo è molto meno dispendioso (in termini di memoria) della ricerca tramite la tabella standard soprattutto se il codice $\mathscr{C}$ ha cardinalità molto grande, poiché richiede di memorizzare una matrice di dimensioni molto ridotte.

Occorre precisare però che non sempre è conveniente per colui che deve decifrare il messaggio cercare di approssimare quanto ricevuto alla parola del codice più vicina possibile. Infatti questo schema, chiamato *decodifica completa*, è utile per messaggi che non possono essere ritrasmessi, come ad esempio fotografie ricevute da Marte o vecchi nastri magnetici, dai quali si cerca di estrarre quante più informazioni possibili.
Al contrario, molto spesso, occorre essere più precisi nella traduzione o semplicemente

non ci si possono permettere tecniche di decodifica dispendiose (in termini di tempo) come quelle esposte precedentemente. Di conseguenza si può optare per la cosiddetta *decodifica incompleta*: se, attraverso l'*error detection*, ci si accorge che si presentano più di l errori si rigetta il messaggio e si chiede una ritrasmissione.
Il vantaggio di quest'ultima strategia è che non solo gli algoritmi di ricerca di errori sono molto più semplici, ma anche che la probabilità di trovare questi ultimi, rispetto alla pretesa di correggerli, aumenta notevolmente.

1.5 Costruzione di nuovi codici

Esistono diverse tecniche che permettono con facilità di creare nuovi codici, manipolando quelli che si conoscono già. In questa sezione vediamo le più utilizzate.

1. *Aggiunta di un bit di parità.* Sia $\mathscr{C}$ un $[n, k, d]$-codice binario. E' possibile formare un nuovo codice binario $\hat{\mathscr{C}}$ aggiungendo uno 0 alla fine di ogni parola di $\mathscr{C}$ con peso pari, e un 1 alla fine di ogni parola di $\mathscr{C}$ con peso dispari. In questo modo, $\hat{\mathscr{C}}$ ha la proprietà di avere tutte le parole di peso pari, che soddisfano l'equazione di controllo di parità:

$$w(\hat{\boldsymbol{x}}) = x_1 + x_2 + \cdots + x_{n+1} = 0, \qquad \text{dove } \hat{\boldsymbol{x}} \in \hat{\mathscr{C}}, \boldsymbol{x} = (x_1, \ldots, x_n) \in \mathscr{C}.$$

   Inoltre, la distanza tra due parole di $\hat{\mathscr{C}}$ è sempre pari, in quanto tutte le parole possibili ora hanno peso pari.
   Attraverso questo metodo, se la minima distanza di $\mathscr{C}$ è dispari, allora quella di $\hat{\mathscr{C}}$ è $d + 1$, così che $\hat{\mathscr{C}}$ diventa un $[n + 1, k, d + 1]$-codice.
   Infine, se $\mathscr{C}$ ha H come matrice di controllo di parità, allora $\hat{\mathscr{C}}$ ha matrice di controllo di parità data da

$$\hat{H} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ & & & 0 \\ & H & & \vdots \\ & & & 0 \end{pmatrix}.$$

Infatti, l'equazione $\hat{H}\hat{\boldsymbol{x}}^T = \mathbf{0}$ (1.1) fornisce proprio le relazioni:

$$\begin{cases} x_1 + x_2 + \cdots + x_{n+1} = 0 \\ H\boldsymbol{x}^T = \mathbf{0} \end{cases} .$$

2. *Estrazione di bit.* Il processo inverso al precedente, consiste nell'eliminazione di una o più coordinate da ogni parola del codice di partenza $\mathscr{C}$, il che equivale allo sfilare una colonna dalla matrice generatrice G del codice. Il codice così ottenuto si indica di solito con $\mathscr{C}^*$.
   In generale, ogni volta che una coordinata viene eliminata, la lunghezza n diminuisce di 1, così come la distanza minima d (tranne che in casi occasionali).

3. *Taglio di parole.* Dato un $[n, k, d]$-codice binario $\mathscr{C}$ che contiene parole di peso pari e dispari, l'unica possibilità è che esso contenga metà parole di peso pari e metà di peso dispari, come visto nel lemma 1.1.2. La seguente tecnica consiste nell'esclusione di tutte le parole del codice di peso dispari, ottenendo in questo modo un $[n, k-1, d']$-codice, con solitamente $d' > d$.
   Il parametro $k-1$ deriva dall'aver escluso dalla matrice G, di ordine $k \times n$, una riga, l'unica dispari presente: a tal proposito, notiamo infatti che se la matrice presenta più righe dispari è sempre possibile fissare una di tali righe al primo posto, e sostituire le restanti i-esime righe dispari, con la somma tra la i-esima e la prima riga, seguendo la regola dei pesi dell'osservazione 4: $w(\boldsymbol{x} + \boldsymbol{y}) = w(\boldsymbol{x}) + w(\boldsymbol{y}) - 2\#\{i \mid x_i = y_i = 1\}$.
   Otteniamo così una matrice formata da tutte righe pari, apparte la prima.

4. *Arricchimento di parole.* Il meccanismo opposto a quello esposto precedentemente consiste nell'aggiungere parole al codice di partenza $\mathscr{C}$. La via più comune è quella di inserire un vettore di tutti 1, che equivale ad inserire una riga di 1 nella matrice generatrice di ordine $k \times n$, ammesso che non sia già presente. In questo modo otteniamo il nuovo codice:
   $$\mathscr{C}^{(a)} = \mathscr{C} \cup \{\mathbf{1} + \mathscr{C}\}$$
   di parametri $[n, k+1, d^{(a)}]$, dove
   $$d^{(a)} = \min\{d, n - d'\}$$

dove d' è il peso più grande di una qualsiasi parola di $\mathscr{C}$.

5. *Allungamento di codici.* Il metodo usuale per allungare un codice è quello di aggiungere la parola composta da tutti **1**, come nel caso 4, e infine estendere il codice effettuando un generico controllo di parità, come nel caso 1. Questo ha l'effetto di aggiungere un nuovo simbolo al messaggio.

Capitolo 2

Codici perfetti

2.1 Probabilità di errore

Torniamo ora, per un attimo, a parlare di decodifica di codici lineari.
Indicheremo di seguito con p la probabilità che un elemento della parola ricevuta $\boldsymbol{y}$ sia errato (supponiamo per semplicità che ogni elemento abbia la stessa probabilità di essere sbagliato). Usando strumenti elementari di calcolo delle probabilità, si ottiene facilmente che, ad esempio, la probabilità di non avere alcun errore su una parola di lunghezza 5 è pari a

$$\text{Prob}\{\boldsymbol{e} = 00000\} = (1-p)^5,$$

o che la probabilità di avere esattamente un errore su una parola di lunghezza 5 è

$$\text{Prob}\{\boldsymbol{e} = 01000\} = p(1-p)^4,$$

e così via. In generale, se $\boldsymbol{v}$ è un vettore di peso a, si ha

$$\text{Prob}\{\boldsymbol{e} = \boldsymbol{v}\} = p^a(1-p)^{n-a}. \tag{2.1}$$

In parole povere, p rappresenta la probabilità che un certo simbolo del vettore di errore $\boldsymbol{e}$ sia 1, mentre $1-p$ è la probabilità che lo stesso sia 0.

Definizione 2.1. La *probabilità di errore* P_{err} di uno schema di decifratura è la probabilità che la funzione "Decoder" fornisca in output una parola scorretta.

Vediamo ora quanto incide questa probabilità sulla decodifica.

Proposizione 2.1.1. *Se la distribuzione della probabilità di uscita delle parole di un codice $\mathscr{C}$ è uniforme e la decifratura è effettuata tramite la matrice standard, allora*

$$P_{err} = 1 - \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i} \tag{2.2}$$

dove α_i rappresenta il numero di leader di classe di peso i, mentre p è la probabilità che un qualsiasi elemento del vettore ricevuto $\boldsymbol{y}$ sia errato.

Dimostrazione. Supponiamo di avere M possibili parole $\boldsymbol{x^{(1)}}, \cdots, \boldsymbol{x^{(M)}}$ di lunghezza n per il codice $\mathscr{C}$, aventi ciascuna uguale probabilità di uscita p. Per ipotesi, la probabilità di errore segue una distribuzione uniforme e si può calcolare come:

$$P_{err} = \frac{1}{M} \sum_{i=1}^{M} \underbrace{\text{Prob}\{\text{codeword decodificata} \neq \boldsymbol{x^{(i)}} \mid \boldsymbol{x^{(i)}} \text{ è la codeword spedita}\}}_{= \text{probabilità che la decodifica sia errata}}.$$

Nel nostro caso, poiché la decodifica è fatta utilizzando la matrice standard, si decodifica in modo sbagliato se e solo se il vettore di errore $\boldsymbol{e}$ non è un leader di classe, perciò

$$P_{err} = \text{Prob}\{\boldsymbol{e} \text{ non è un leader di classe}\} = 1 - \text{Prob}\{\boldsymbol{e} \text{ è un leader di classe}\}.$$

Supponendo che α_i sia il numero di leader di classe di peso i, usando la relazione 2.1 otteniamo

$$P_{err} = 1 - \sum_{i=0}^{n} \alpha_i \text{Prob}\{\boldsymbol{e} \text{ è un leader di classe di peso } i\} = 1 - \sum_{i=0}^{n} \alpha_i p^i (1-p)^{n-i}.$$

□

Lemma 2.1.2. *Se il codice ha minima distanza di Hamming pari a $d = 2t+1$ o $d = 2t+2$, allora*

$$\alpha_i = \binom{n}{i} \qquad 0 \leq i \leq t.$$

Dimostrazione. Sappiamo per il teorema 1.4.1 che il codice riesce a correggere t errori. Quindi ogni vettore di errore di peso $\leq t$ costituisce un leader di classe. Perciò per contare il numero di leader di classe con peso pari a i $(= \alpha_i)$ utilizziamo il calcolo combinatorio e otteniamo la tesi. □

Osservazione 6. Se supponiamo che il canale sia poco rumoroso, e perciò che la probabilità p sia piccola, allora $(1-p) \simeq 1$, quindi $p^i(1-p)^{n-i} \gg p^{i+1}(1-p)^{n-i-1}$ per ogni i. Questo significa che nella formula 2.2 è possibile trascurare i termini con $i > t$, ottenendo:

$$P_{err} = 1 - \sum_{i=0}^{t} \alpha_i p^i (1-p)^{n-i},$$

e nel caso in cui il codice abbia distanza $d = 2t+1$ o $d = 2t+2$ si ottiene la formula semplificata:

$$P_{err} = 1 - \sum_{i=0}^{t} \binom{n}{i} p^i (1-p)^{n-i}.$$

Definizione 2.2. Un codice $\mathscr{C}$ per cui $\alpha_i = 0$ per $i > t = \lfloor (d-1)/2 \rfloor$ si dice *perfetto*.

Da questa definizione abbiamo che un codice perfetto $\mathscr{C}$ di lunghezza n e distanza $d = 2t+1$ corregge tutti gli errori di peso minore o uguale a t e nessuno di peso maggiore a t.

Una formulazione equivalente è che le palle di raggio t centrate nelle parole di $\mathscr{C}$ sono disgiunte e insieme contengono tutti i vettori di lunghezza n.

Teorema 2.1.3 (Limite di Hamming)**.** *Sia $\mathscr{C}$ un codice di lunghezza n e distanza $d = 2t+1$. Allora*

$$|\mathscr{C}| \cdot \left[\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t} \right] \leq 2^n,$$

ovvero

$$|\mathscr{C}| \leq \frac{2^n}{\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}},$$

dove con $|\mathscr{C}|$ indichiamo la cardinalità del codice, ossia il numero di parole che esso contiene.

Dimostrazione. Consideriamo una qualsiasi parola $\boldsymbol{x}$ del codice $\mathscr{C}$ e la relativa palla di centro $\boldsymbol{x}$ e raggio t. Il numero di vettori che distano al più t da $\boldsymbol{x}$ sono pari a:

$$N = \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}.$$

Poiché le palle sono tutte disgiunte, il numero totale di parole distinte del codice che hanno distanza minore o uguale a t da ogni singola parola di $\mathscr{C}$ è:

$$|\mathscr{C}| \cdot N = |\mathscr{C}| \cdot \left[\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t} \right].$$

Ma il numero totale di parole distinte di lunghezza n in $\mathscr{F}_2^n$ è 2^n, perciò:

$$|\mathscr{C}| \leq \frac{2^n}{\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}}.$$

□

Una formulazione equivalente alla definizione data in 2.2 è la seguente:

Definizione 2.3. Un codice $\mathscr{C}$ di lunghezza n e distanza $d = 2t + 1$ si dice *perfetto* se vale:

$$|\mathscr{C}| = \frac{2^n}{\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}}. \tag{2.3}$$

Osservazione 7. Le definizioni di codice perfetto date in 2.2 e 2.3 sono equivalenti. Infatti, l'uguaglianza nella formula 2.3 garantisce che tutte le parole distinte di $\mathscr{C}$ che hanno distanza minore o uguale a t dalle altre coincidono con tutte le parole distinte di lunghezza n possibili. Ciò significa che le palle di raggio t centrate nelle parole del codice sono disgiunte e insieme contengono tutti i possibili vettori di lunghezza n, così come affermato nella definizione 2.2.

Teorema 2.1.4. *Condizione necessaria e sufficiente per l'esistenza di un codice perfetto è che la quantità*

$$\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{t}$$

sia potenza di 2.

Dimostrazione. La tesi è una conseguenza immediata della definizione 2.3 di codice perfetto e del fatto che la cardinalità di un codice binario deve essere una potenza di 2. □

Esempio 2.1. Un codice $\mathscr{C}$ di lunghezza $n = 7$ e distanza $d = 2t + 1 = 3$ è perfetto. Infatti, $t = 1$ e quindi:

$$|\mathscr{C}| = \frac{2^7}{\binom{7}{0} + \binom{7}{1}} = \frac{2^7}{2^3} = 2^4 = 16.$$

Esempio 2.2. Un codice $\mathscr{C}$ di lunghezza $n = 23$ e distanza $d = 2t + 1 = 7$ è perfetto. Infatti, $t = 3$ e quindi:

$$|\mathscr{C}| = \frac{2^{23}}{\binom{23}{0} + \binom{23}{1} + \binom{23}{2} + \binom{23}{3}} = \frac{2^{23}}{2^{11}} = 2^{12} = 4096.$$

2.2 Codici di Hamming

I **codici di Hamming** costituiscono un'importante famiglia di codici perfetti 1-correttori molto semplici da codificare e decodificare. In questo elaborato tratteremo nello specifico solo i codici di Hamming binari.

Definizione 2.4. Un *codice binario di Hamming* $\mathscr{H}_r$ è un codice lineare di lunghezza $n = 2^r - 1$ con $r \geq 2$, avente matrice di parità H, tale che le colonne constino di tutti e soli i vettori binari non nulli di lunghezza r, ognuno dei quali può essere ripetuto soltanto una volta. Di conseguenza la matrice H ha ordine $r \times n$.
Dunque $\mathscr{H}_r$ è un codice di:

lunghezza: $n = 2^r - 1$ con $r \geq 2$

dimensione: $k = 2^r - 1 - r$.

Osservazione 8. Secondo la definizione 2.4, i codici di Hamming sono sicuramente 1-correttori, in quanto, per la proprietà 3 della sindrome, quest'ultima coincide con la somma delle colonne della matrice H in cui è presente almeno un errore. Di conseguenza, per costruire un codice 1-correttore è sufficiente che le colonne di H siano:

1. non nulle, altrimenti un errore in tale posizione non influenzerebbe la sindrome e non sarebbe localizzabile.

2. distinte, poiché se ci fossero anche solo due colonne uguali, gli errori in queste due posizioni sarebbero indistinguibili.

Ma queste sono proprio le condizioni imposte nella definizione del codice $\mathscr{H}_r$.

Osservazione 9. La minima distanza di Hamming di $\mathscr{H}_r$ è 3. Infatti, in accordo con il teorema 1.4.1, poiché il codice per definizione è 1-correttore, vale l'equazione:

$$1 = \frac{1}{2}(d-1)$$

perciò d=3.

Definizione 2.5. Due codici su uno stesso campo $\mathscr{F}$ sono detti *equivalenti* se le loro matrici di parità possono ottenersi l'una dall'altra mediante una successione finita di operazioni dei seguenti due tipi:

- scambio di due colonne (il che equivale a scambiare tra loro due simboli in ogni parola del codice).
- moltiplicazione degli elementi di una colonna per uno scalare non nullo (nel caso di un campo con due elementi, questa operazione è inutile).

Proposizione 2.2.1. *I codici di Hamming $\mathscr{H}_r$, a r fissato, sono tutti equivalenti tra loro.*

Dimostrazione. Una volta fissato il parametro r, l'insieme delle colonne che compone la matrice H è lo stesso, mentre rimane arbitrario l'ordine in cui disporle. □

Esempio 2.3. Il codice di Hamming $\mathscr{H}_3$ ha matrice di parità:

$$H = \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}. \tag{2.4}$$

Scrivendo la matrice in forma standard come in 1.2 otteniamo, ad esempio:

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

Osserviamo che H e H' forniscono codici equivalenti.
In generale,

$$H' = [\ A \mid I_r\ ],$$

dove A contiene colonne con almeno due 1.

Osservazione 10. Una delle caratteristiche più importanti dei codici di Hamming è l'estrema facilità di decodifica. Supponiamo di avere la matrice di controllo di parità H nella forma 2.4, in cui la i-esima colonna $\boldsymbol{H_i}$ contiene la rappresentazione binaria di i. Se ci fosse un errore nell'l-esimo elemento di $\boldsymbol{y}$, allora per la proprietà 2 della sindrome si avrebbe:

$$\boldsymbol{S} = H\boldsymbol{y}^T = H\boldsymbol{e}^T = \boldsymbol{H_l} = \text{rappresentazione binaria di } l.$$

Teorema 2.2.2. *I codici di Hamming sono codici perfetti 1-correttori.*

Dimostrazione. Poiché il codice $\mathscr{H}_r$ può correggere un solo errore, nella rappresentazione tramite sferette della decodifica secondo l'intorno più vicino, le palle centrate nelle parole del codice di raggio 1 sono disgiunte.
Per dimostrare che il codice $\mathscr{H}_r$ è perfetto, occorre contare il totale dei vettori di lunghezza n presenti in ogni sfera di raggio 1: ci sono $2^k = 2^{2^r-1-r}$ palle, ciascuna delle quali ospita $n+1 = 2^r$ vettori, poiché contiamo tutti i possibili vettori che differiscono al massimo per un posto dalla parola centrale. Abbiamo perciò un totale di $2^{2^r-1-r} \cdot 2^r = 2^{2^r-1} = 2^n$. Quindi tutti i vettori possibili di lunghezza n (che sono appunto 2^n) sono contenuti nell'unione di tali palle, quindi per la definizione 2.2 concludiamo che il codice è perfetto. □

Riassunto delle proprietà del Codice di Hamming $\mathscr{H}_r$:

lunghezza:	$n = 2^r - 1$ con $r \geq 2$,
dimensione:	$k = 2^r - 1 - r$,
numero di equazioni di parità:	r,
minima distanza di Hamming:	$d = 3$.

$\mathscr{H}_r$ è un codice perfetto 1-correttore, unico a meno di equivalenze.
La matrice di parità H composta da vettori r-dimensionali non nulli.

Tabella 2.1: Proprietà del Codice di Hamming $\mathscr{H}_r$.

2.3 t-designs

Definizione 2.6. Sia X un insieme di v elementi chiamati, per convenzione, *punti*. Un *t-design* è una collezione di sottoinsiemi distinti di X, chiamati *blocchi*, ciascuno avente k elementi con la proprietà che ogni sottocollezione di t elementi di X è contenuta in esattamente λ blocchi.
Intuitivamente, usando un linguaggio più pittoresco, un *t-design* è una collezione di comitati formati a partire da un gruppo di v persone, ciascuno contenente k individui, e tale che in ciascuno di essi, esattamente t persone si occupano anche di altri λ comitati. Più precisamente, chiamiamo questo tipo di struttura un t-(v, k, λ) *design*.

Esempio 2.4. Consideriamo la figura 2.1. Se interpretiamo le linee come dei blocchi, otteniamo un 2-(7, 3, 1) *design*, dove appunto 7 è il numero di punti totali, 3 è il numero di punti in ciascun blocco (=punti contenuti in ciascuna linea) e 1 è il numero di blocchi che 2 qualsiasi punti condividono (=numero di righe passanti per 2 punti qualsiasi).
I sette blocchi che si vengono a formare sono:

$$013, \quad 124, \quad 235, \quad 346, \quad 450, \quad 561, \quad 602.$$

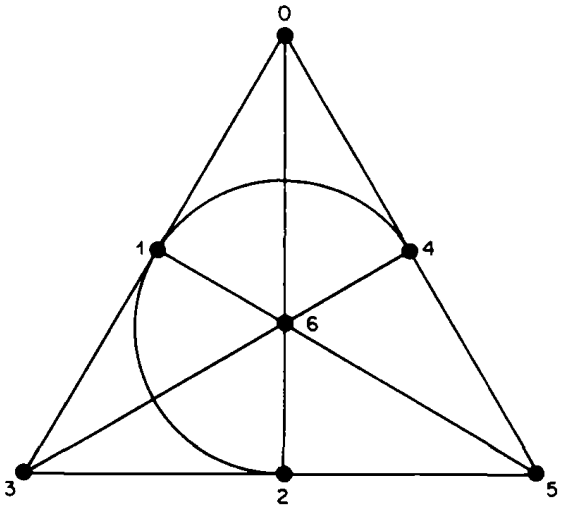

Figura 2.1: Esempio di 2-(7, 3, 1) *design*.

Definizione 2.7. Un *sistema di Steiner* è un t-design dove $\lambda = 1$.
Solitamente viene indicato con $S(t, k, v)$. Quello rappresentato in figura 2.1 è un sistema di Steiner $S(2, 3, 7)$.

Definizione 2.8. Dato un t-(v, k, λ) design, supponiamo di ordinare gli insiemi dei punti in $P_1, \ldots, P_v$ e quello dei blocchi in $B_1, \ldots, B_b$. Allora chiameremo *matrice d'incidenza* $A = (a_{ij})$ la matrice di dimensioni $b \times v$ definita da:

$$a_{ij} = \begin{cases} 1, & \text{se } p_j \in B_i \\ 0, & \text{altrimenti} \end{cases} .$$

Ovviamente tale matrice A dipende dagli ordinamenti scelti sui punti e i blocchi, ed individua completamente un disegno. Ad esempio, una matrice di incidenza per il disegno in figura 2.1 (con opportuna ordinazione dei blocchi) è:

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

Teorema 2.3.1. *Consideriamo un t-(v, k, λ) design. Siano $P_1, \ldots, P_t$ punti distinti. Chiamiamo λ_i il numero di blocchi contenenti $P_1, \ldots, P_i$ per $1 \leq i \leq t$. Allora λ_i è indipendente dalla scelta di $P_1, \ldots, P_i$, e vale:*

$$\lambda_i = \frac{\lambda \binom{v-i}{t-i}}{\binom{k-i}{t-i}} = \lambda \frac{(v-i)(v-i-1)\cdots(v-t+1)}{(k-i)(k-i-1)\cdots(k-t+1)} \qquad 0 \leq i \leq t.$$

Questo implica che un t-(v, k, λ) design è anche un i-(v, k, λ_i) design per $1 \leq i \leq t$.

Dimostrazione. Il risultato è vero per $i = t$ poiché per definizione di t-design si ha che t punti qualsiasi sono contenuti in esattamente λ blocchi e l'enunciato afferma:

$$\lambda_t = \frac{\lambda \binom{v-t}{0}}{\binom{k-t}{0}} = \lambda.$$

Procediamo ora per induzione su i. Supponiamo quindi per ipotesi di induzione che valga il teorema per λ_{i+1} e dimostriamolo per λ_i.
Per ogni blocco B contenente $P_1, \ldots, P_i$ e per ogni punto Q distinto dai precedenti, definiamo la funzione caratteristica

$$\chi(Q, B) = \begin{cases} 1, & \text{se } Q \in B \\ 0, & \text{altrimenti} \end{cases}.$$

Allora per ipotesi di induzione vale:

$$\sum_B \sum_Q \chi(Q, B) = \lambda_i(k - i)$$

in quanto contiamo dapprima il numero di punti in un blocco fissato, eccetto i $P_1, \ldots, P_i$ (che sono in tutto $k - i$), e poi tutti i blocchi B contenenti i punti $P_1, \ldots, P_i$ (che sono λ_i).
Inoltre, per lo stesso motivo vale:

$$\sum_Q \sum_B \chi(Q, B) = \lambda_{i+1}(v - i)$$

poiché questa volta contiamo il numero di blocchi contenenti $P_1, \ldots, P_i, Q$ (che sono λ_{i+1}) e poi sommiamo tutti i punti dello spazio eccetto i $P_1, \ldots, P_i$ (e sono proprio $v - i$).
A questo punto, poiché ogni relazione blocco-punto compare solo una volta, è possibile commutare le due sommatorie ottenendo:

$$\lambda_i(k - i) = \sum_B \sum_Q \chi(Q, B) = \sum_Q \sum_B \chi(Q, B) = \lambda_{i+1}(v - i),$$

che prova l'indipendenza di λ_i dalla scelta di $P_1, \ldots, P_i$ e fornisce la tesi.
L'ultima riga dell'asserzione segue in maniera ovvia. □

Consideriamo, ora, un t-(v, k, λ) design. Chiamiamo $P_1, \ldots, P_k$ i punti appartenenti ad uno dei blocchi. Consideriamo i blocchi che contengono i punti $P_1, \ldots, P_j$ ma non $P_{j+1}, \ldots, P_i$ per $0 \leq j \leq i$. (Per $j = 0$ consideriamo i blocchi che non contengono $P_1, \ldots, P_i$ mentre per $j = i$ consideriamo quei blocchi che li contengono tutti). Se il numero di questi blocchi è costante, indipendentemente dalla scelta dei punti, lo denoteremo con λ_{ij}.

Teorema 2.3.2. *Le quantità λ_{ij} sono ben definite per $i \leq t$. In particolare, $\lambda_{ii} = \lambda_i$ per $i \leq t$ e si calcolano seguendo la formula del teorema 2.3.1. Inoltre i λ_{ij} soddisfano la proprietà di Pascal, ossia*

$$\lambda_{ij} = \lambda_{i+1,j} + \lambda_{i+1,j+1}.$$

Di conseguenza se il design è un sistema di Steiner, si ha $\lambda = 1$, quindi $\lambda_{tt} = \lambda_{t+1,t+1} = \cdots = \lambda_{kk} = 1$ e questo vale per ogni $0 \leq j \leq i \leq k$.

Dimostrazione. Per prima cosa, notiamo che $\lambda_{ii} = \lambda_i$ per $i \leq t$ poiché entrambe le quantità contano il numero di blocchi contenente i punti $P_1, \ldots, P_i$.
Dimostriamo ora che la proprietà di Pascal sussiste: consideriamo $\lambda_{i+1,j}$. Esso conta il numero di blocchi contenenti $P_1, \ldots, P_j$ e non contenenti $P_{j+1}, \ldots, P_{i+1}$. Un modo per contarli è quello di isolare il punto P_{i+1} e contare il numero di blocchi contenenti $P_1, \ldots, P_j$ ma non contenenti $P_{j+1}, \ldots, P_i$ (che sono in tutto λ_{ij}), ma notiamo che questi potrebbero contenere anche P_{i+1}. Quindi dobbiamo togliere il numero di blocchi che contengono $P_1, \ldots, P_j$ e anche P_{i+1}. Per l'indipendenza dei λ_{ij} dalla scelta dei punti, questi ultimi blocchi sono proprio $\lambda_{i+1,j+1}$. Otteniamo perciò:

$$\lambda_{i+1,j} = \lambda_{ij} - \lambda_{i+1,j+1}.$$

□

Definizione 2.9. Le quantità λ_{ij}, definite per $i \leq t$, sono chiamate *numeri di intersezione dei blocchi.*

Per ogni t-(v, k, λ) design è possibile costruire il cosiddetto "Triangolo di Pascal", formato dai suoi numeri di intersezione dei blocchi, in questo modo:

$$\begin{array}{ccccc} \lambda_{00} = \lambda_0 & & & & \\ \lambda_{10} & \lambda_{11} = \lambda_1 & & & \\ \lambda_{20} & \lambda_{21} & \lambda_{22} = \lambda_2 & & \\ \multicolumn{5}{c}{\cdots} \end{array}$$

Esempio 2.5. Per il sistema di Steiner $S(2,3,7)$ che fa riferimento alla figura 2.1, il triangolo di Pascal assume la seguente forma:

$$\begin{array}{ccccccc} & & & 7 & & & \\ & & 4 & & 3 & & \\ & 2 & & 2 & & 1 & \\ 0 & & 2 & & 0 & & 1 \end{array}$$

Infatti, 7 è il numero totale di blocchi (nel nostro disegno le linee), ottenuto dalla somma di 4 e 3, i quali rappresentano rispettivamente il conteggio dei blocchi che non contengono il punto 0 e di quelli che lo contengono; similmente, la riga successiva è formata dal numero di blocchi che non contengono né 0 né 1 (che sono 2), il numero di blocchi che contengono 0 ma non 1 (che sono ancora 2) e il numero di blocchi che contengono entrambi (e ce n'è solo 1). E così via.

A partire da un dato t-(v,k,λ) design con numeri di intersezione dei blocchi λ_{ij}, possiamo ottenere diversi altri disegni. Denotiamo con $\mathscr{B}$ il disegno originale avente b blocchi e i punti $P_1,\dots,P_v$. Supponiamo di eliminare il punto P_1 dal disegno e consideriamo le due nuove possibili strutture che ci restano: la prima, $\mathscr{B}_1$, consiste di tutti i blocchi di k punti che non contengono P_1 e sono proprio λ_{10}, la seconda, $\mathscr{B}_2$, consiste invece dei blocchi rimanenti, che ora saranno composti da $k-1$ punti e sono in totale λ_{11} .

Teorema 2.3.3. *I blocchi di $\mathscr{B}_1$ formano un $(t-1)-(v-1,k,\lambda_{t,t-1})$ design, mentre i blocchi di $\mathscr{B}_2$ formano un $(t-1)-(v-1,k-1,\lambda)$ design.*
Questi nuovi disegni $\mathscr{B}_1$ e $\mathscr{B}_2$ sono chiamati designs derivati.

Dimostrazione. Consideriamo il blocco $\mathscr{B}_1$: avendo scartato il punto P_1, ora i punti in totale saranno $v-1$; il numero di punti facente parte uno stesso blocco rimane invariato, cioè k. Se in partenza avevamo $P_1,\dots,P_t$ punti che facevano parte contemporaneamente

di λ blocchi, ora, di questi, esattamente $t-1$ punti (escludiamo il punto P_1) fa parte di altri $\lambda_{t,t-1}$ blocchi. Infatti $\lambda_{t,t-1}$ conta, per definizione, il numero di blocchi che non contengono P_1.
Consideriamo ora $\mathscr{B}_2$: anche qui i punti totali diventano $v-1$, e per ogni blocco avremo $k-1$ punti proprio perché, per costruzione, stiamo considerando solo quei blocchi che contenevano P_1, che ora è stato escluso. Analogamente a sopra, se in partenza avevamo $P_1, \ldots, P_t$ punti che facevano parte contemporaneamente di λ blocchi, ora ne abbiamo $t-1$, mentre il numero di blocchi resta invariato. □

Corollario 2.3.4. *L'esistenza del sistema di Steiner $S(t,k,v)$ implica l'esistenza del sistema di Steiner $S(t-1,k-1,v-1)$.*

Dimostrazione. La tesi segue da teorema 2.3.3. □

Applicazioni pratiche dei 2-designs

Un 2-design è solitamente chiamato *balanced incomplete block design* grazie proprio all'applicazione originaria di questo tipo di strutture geometriche in agricoltura e biologia.
Un esempio pratico può essere quello di voler testare il funzionamento di v varietà di fertilizzante su b colture differenti. La cosa migliore, idealmente, sarebbe quella di suddividere il terreno in b blocchi, ognuno di larghezza v, così da poter testare l'effetto di ciascun fertilizzante su ogni coltura.
Molto spesso però, per ragioni di dispendiosità economica, questo metodo risulta impraticabile. Per questo motivo entrano in gioco i t-designs: solitamente si utilizza un design che permetta di testare ogni coltura con k differenti varietà di fertilizzante (in questo caso ci sono b blocchi, ciascuno dei quali ha larghezza $k < v$) e in cui due fertilizzanti qualsiasi sono usati contemporaneamente sulla stessa coltura un numero costante λ di volte. Così facendo otteniamo un 2-(v,k,λ) design con b blocchi.
La terminologia deriva dal fatto che la struttura è "incompleta" in quanto $k < v$ ed è "bilanciata" per quanto riguarda le comparazione di coppie diverse di fertilizzanti.

2.4 Codici e designs

E' possibile costruire dei codici a partire da un t-design. Consideriamo un sistema di Steiner $S(t, k, v)$ con relativa matrice di incidenza A di dimensioni $b \times v$. Per costruzione della matrice, ad ogni blocco del disegno corrisponde una riga di A. Se pensiamo alle righe di A come a delle parole, otteniamo un codice non lineare di parametri:

- lunghezza: $n = v$.
- cardinalità: b.
- peso: k.

Riguardo a codici lineari generati da t-designs poco si conosce tuttora, se non alcuni risultati riguardo il piano proiettivo e affine nel caso in cui prendessimo $t = 2$, argomento che non tratteremo nello specifico.
Viceversa, da un codice di cui si conosce lunghezza n, cardinalità M e distanza d, a volte è possibile ottenere un disegno, a partire da parole del codice di lunghezza fissata. Vediamo come.

Definizione 2.10. Diciamo che un vettore $\boldsymbol{v}$ *copre* un vettore $\boldsymbol{u}$ se gli elementi pari a 1 in $\boldsymbol{u}$ sono un sottoinsieme di quelli in $\boldsymbol{v}$. Ad esempio, 1001 copre 1001,1000,0001 e 0000. Formulazioni equivalenti sono:

$$\boldsymbol{u} * \boldsymbol{v} = \boldsymbol{u}$$

$$w(\boldsymbol{u} + \boldsymbol{v}) = w(\boldsymbol{u}) + w(\boldsymbol{v}).$$

Teorema 2.4.1. *Sia $\mathscr{H}_r$ un codice di Hamming di parametri $[n = 2^r - 1,\ k = 2^r - 1 - r,\ d = 3]$. Sia inoltre $\hat{\mathscr{H}_r}$ il codice ottenuto da $\mathscr{H}_r$ aggiungendo ad ogni parola un simbolo di parità. Allora le parole di peso 3 in $\mathscr{H}_r$ formano un sistema di Steiner $S(2, 3, 2^r - 1)$ e le parole di peso 4 in $\hat{\mathscr{H}_r}$ formano un sistema di Steiner $S(3, 4, 2^r)$.*

Dimostrazione. La prima asserzione del teorema segue dalla seconda applicando il corollario 2.3.4. Per dimostrare la seconda affermazione, invece, chiamiamo $P_0, P_1, \ldots, P_n$ le coordinate di una qualsiasi parola di $\hat{\mathscr{H}_r}$, dove P_n è per costruzione un simbolo di parità. Sia $\boldsymbol{u}$ una parola arbitraria del codice $\mathscr{H}_r$ di peso 3: presenterà perciò l'elemento 1 in tre

posizioni, che chiameremo, in ordine, P_h, P_i e P_j. Dobbiamo fare vedere che esiste una sola parola in $\hat{\mathscr{H}_r}$ di peso 4 che copre $\boldsymbol{u}$.
Vediamo innazitutto che se tale parola in $\hat{\mathscr{H}_r}$ esiste, allora è sicuramente unica: infatti se per assurdo ne esistessero 2 o più, la loro distanza sarebbe 2, il che è assurdo. Ora, per dimostrarne l'esistenza, distinguiamo due casi:

$j < n$: Poiché $\mathscr{H}_r$ è un codice perfetto 1-correttore, contiene una parola $\boldsymbol{c}$ a distanza al massimo 1 da $\boldsymbol{u}$. Quindi o $\boldsymbol{c} = \boldsymbol{u}$, e quindi la parola estesa $\hat{\boldsymbol{c}} = \mid \boldsymbol{c} \mid 1 \mid$ ha peso 4, copre $\boldsymbol{u}$ e appartiene a $\hat{\mathscr{H}_r}$; oppure $\boldsymbol{c}$ ha peso 4 e la parola estesa $\hat{\boldsymbol{c}} = \mid \boldsymbol{c} \mid 0 \mid$ ha peso 4, copre $\boldsymbol{u}$ e appartiene a $\hat{\mathscr{H}_r}$.

$j = n$: In questo caso il vettore $\boldsymbol{c'}$ avente 1 nelle coordinate P_h e P_i è coperto da un'unica codeword $\boldsymbol{c} \in \mathscr{H}_r$ di peso 3, e la codeword estesa $\hat{\boldsymbol{c}} = \mid \boldsymbol{c} \mid 1 \mid$ copre $\boldsymbol{u}$, ha peso 4 e appartiene a $\hat{\mathscr{H}_r}$.

□

Teorema 2.4.2. *Sia $\mathscr{C}$ un codice perfetto e-correttore di lunghezza n con e dispari, e sia $\hat{\mathscr{C}}$ il codice ottenuto da $\mathscr{C}$ aggiungendo un simbolo di parità ad sua ogni parola. Allora le parole di peso $2e+1$ in $\mathscr{C}$ formano un sistema di Steiner $S(e+1,\ 2e+1,\ n)$, e le parole di peso $2e+2$ in $\hat{\mathscr{C}}$ formano un sistema di Steiner $S(e+2,\ 2e+2,\ n+1)$.*

Dimostrazione. La dimostrazione è analoga a quella del teorema 2.4.1. In particolare occorre fare vedere che, fissata $\boldsymbol{u} \in \mathscr{C}$ di peso $e+2$, esiste ed è unica la parola in $\hat{\mathscr{C}}$ di peso $2e+2$ che copre $\boldsymbol{u}$. L'unicità deriva dal fatto che $\hat{\mathscr{C}}$ è un codice e-correttore, quindi la distanza minima tra due parole del codice è $d = 2e+1$.
Per l'esistenza, invece, si procede aggiungendo l'elemento 0 o 1 in fondo alla parola $\boldsymbol{c} \in \mathscr{C}$ che dista al massimo e dalla parola $\boldsymbol{u}$ fissata, in modo da ottenere una nuova parola $\hat{\boldsymbol{c}}$ appartenente a $\hat{\mathscr{C}}$, di peso $2e+2$ e che copre $\boldsymbol{u}$. □

2.5 Codice di Golay

Il **Codice di Golay** è probabilmente uno dei più importanti codici della storia, non solo dal punto di visto pratico/scientifico ma anche da quello teorico/speculativo.
Viene presentato nel 1949 dal suo inventore, dal quale appunto prende il nome, *Marcel J. E. Golay*, ingegnere svizzero, il cui articolo originale introduce tale codice e la sua matrice generatrice, strumenti sufficienti per dimostrarne l'esistenza, nonostante non venga citato nella carta il processo mediante il quale l'autore sia giunto a tale formulazione finale. La sua importanza emerse qualche anno dopo, in particolare negli anni '80 per le comunicazioni con la sonda Voyager e in generale per la trasmissione di dati digitali.
Esistono due cosiddetti codici binari di Golay, strettamente collegati uno all'altro: il **codice di Golay esteso** $\mathscr{G}_{24}$, di cui ci occuperemo principalmente, di parametri $[n = 24, k = 12, d = 8]$, generato dal **codice di Golay** $\mathscr{G}_{23}$ di parametri $[n = 23, k = 11, d = 7]$. $\mathscr{G}_{24}$ si ottiene da $\mathscr{G}_{23}$ semplicemente aggiungendo a quest'ultimo un bit di parità.

Definizione 2.11. Il *codice esteso di Golay* $\mathscr{G}_{24}$ ha come matrice generatrice la matrice mostrata in figura 2.2,

$G =$

	← l →												← r →											
row	∞	0	1	2	3	4	5	6	7	8	9	10	∞	0	1	2	3	4	5	6	7	8	9	10
0	1	1												1	1		1	1	1				1	
1	1		1												1	1		1	1	1				1
2	1			1										1		1	1		1	1	1			
3	1				1										1		1	1		1	1	1		
4	1					1										1		1	1		1	1	1	
5	1						1										1		1	1		1	1	1
6	1							1						1				1		1	1		1	1
7	1								1					1	1				1		1	1		1
8	1									1				1	1	1				1		1	1	
9	1										1				1	1	1				1		1	1
10	1											1		1		1	1	1				1		1
11													1	1	1	1	1	1	1	1	1	1	1	1

Figura 2.2: Matrice generatrice del codice $\mathscr{G}_{24}$.

nella quale le colonne sono chiamate $l_\infty, l_0, l_1, \cdots .l_{10}, r_\infty, r_0, r_1, \cdots, r_{10}$, mentre la matrice 11×11 sulla destra è denominata A_{11}.

Osservazione 11. Notiamo che la somma di qualsiasi coppia di righe di A_{11} ha peso pari a 6. Di conseguenza la somma di qualsiasi coppia di righe di G ha peso pari a 8.

Proprietà del codice di Golay esteso

1. $\mathscr{G}_{24}$ ha lunghezza $n = 24$, dimensione $k = 12$ e cardinalità $|\mathscr{G}_{24}| = 2^{12} = 4096$.

2. A_{11} è una matrice simmetrica, cioè $A_{11} = A_{11}^T$.

3. $\mathscr{G}_{24}$ è autoduale, cioè $\mathscr{G}_{24} = \mathscr{G}_{24}^\perp$.

   Infatti, prese $\boldsymbol{u}$ e $\boldsymbol{v}$ righe non necessariamente distinte di G, vale $w(\boldsymbol{u} * \boldsymbol{v}) = 0$, dove $*$ rappresenta il prodotto componente per componente. Ciò significa che ogni riga di G è ortogonale alle altre, e quindi $\mathscr{G}_{24} \subseteq \mathscr{G}_{24}^\perp$.
   Viceversa, G ha rango 12, perciò $\mathscr{G}_{24}$ ha dimensione 12, così come il suo duale, quindi necessariamente $\mathscr{G}_{24} = \mathscr{G}_{24}^\perp$.

4. Una matrice di parità di $\mathscr{G}_{24}$ è $H = \begin{pmatrix} A_{11} \\ I \end{pmatrix}$.

   Infatti, vale:
   $$G \cdot H = (I \quad A_{11}) \begin{pmatrix} A_{11} \\ I \end{pmatrix} = A_{11} + A_{11} = 0.$$

5. Un'altra matrice di parità di $\mathscr{G}_{24}$ è $H' = \begin{pmatrix} I \\ A_{11} \end{pmatrix}$.

   Infatti, vale:
   $$G \cdot H = (I \quad A_{11}) \begin{pmatrix} I \\ A_{11} \end{pmatrix} = I + A_{11}A_{11}^T = I + I = 0.$$

6. $\mathscr{G}_{24}$ contiene la codeword $\mathbf{1}$.

   Infatti, sommando tutte le righe di G troviamo proprio il vettore $\mathbf{1}$.

7. Ogni parola di $\mathscr{G}_{24}$ ha peso divisibile per 4.

   Infatti, basta applicare il lemma 1.3.3 in quanto ogni riga della matrice generatrice G ha peso divisibile per 4.

Lemma 2.5.1. *$\mathscr{G}_{24}$ è invariante rispetto alla permutazione che scambia tra loro le due metà di una parola del codice:*

$$T = (l_\infty r_\infty)(l_0 r_0)(l_1 r_{10})(l_2 r_9)\cdots(l_{10} r_1),$$

cioè, per essere più espliciti, se $\mathscr{G}_{24}$ contiene la parola $\mid L \mid R \mid$ con $L = a_\infty a_0 a_1 \dots a_{10}$ e $R = b_\infty b_0 b_1 \dots b_{10}$, allora contiene anche la parola $\mid L' \mid R' \mid$ dove

$$L' = b_\infty b_0 b_{10} \dots b_1, \quad R' = a_\infty a_0 a_{10} \dots a_1.$$

Dimostrazione. La permutazione T manda la riga 0 di G in

$$0,\ 1,\ 0\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 1,\ 1,\ 1,\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0$$

che si verifica facilmente essere la somma delle righe 0, 2, 6, 7, 8, 10 e 11, e perciò appartiene al codice. Analogamente per le righe da 1 a 10. Per quanto riguarda l'ultima riga, essa viene mandata in

$$1^{12}0^{12}$$

che è il complementare dell'ultima riga e perciò appartiene al codice. □

Osservazione 12. Il lemma 2.5.1 garantisce che ogni volta che $\mathscr{G}_{24}$ contiene la parola $\mid L \mid R \mid$ con $w(L) = i$ e $w(R) = j$, allora contiene anche la parola $\mid L' \mid R' \mid$ con $w(L') = j$ e $w(R') = i$.

Lemma 2.5.2. *$\mathscr{G}_{24}$ non contiene parole di peso pari a 4.*

Dimostrazione. Sappiamo che per ogni parola $\mid L \mid R \mid$ di $\mathscr{G}_{24}$ si ha che $w(L)$ e $w(R)$ sono pari, per le proprietà di costruzione della matrice G.

Per il lemma 2.5.1 possiamo supporre che se, per assurdo, esistesse nel codice una parola di peso pari a 4, sarebbe di uno dei seguenti due tipi:

$$(1) \quad w(L) = 0, \quad w(R) = 4$$

oppure

$$(2) \quad w(L) = 2, \quad w(R) = 2.$$

Ma (1) risulta impossibile in quanto se $w(L) = 0$, allora $w(R) = 0$ oppure 12, poiché deriva da una combinazione lineare dell'ultima riga di G.
Infine, anche (2) risulta impossibile poiché se $w(L) = 2$, allora deriva dalla somma di una o due righe di G, con l'eventualità dell'ultima riga; in ogni caso $w(R) = 6$ per l'osservazione 11 e quindi arriviamo ad una contraddizione. □

Osservazione 13. Dalla proprietà 7 dei codici di Golay sappiamo che i possibili pesi per le parole di $\mathscr{G}_{24}$ sono:

$$0, 4, 8, 12, 16, 20, 24.$$

Per il lemma 2.5.2, i possibili pesi diventano:

$$0, 8, 12, 16, 24.$$

(notiamo che scompare anche la possibilità di peso pari a 20 poiché se $w(\boldsymbol{u}) = 20$ per una certa parola $\boldsymbol{u}$ del codice, allora $w(\boldsymbol{u} + \mathbf{1}) = 4$, per l'osservazione 4, e questo è impossibile).

Lemma 2.5.3. *Ad ogni sottomatrice sinistra L di G corrispondono due possibili sottomatrici destre, che chiameremo R e $\bar{R}$.*

Dimostrazione. Interpretiamo il codice di Golay come uno spazio vettoriale. Consideriamo quindi l'omomorfismo di spazi vettoriali

$$\pi: \qquad \mathscr{F}_2^{24} \qquad \longrightarrow \qquad \mathscr{F}_2^{12}$$

$$(x_1, \ldots, x_{24}) \mapsto (x_1, \ldots, x_{12})$$

Chiamiamo $K \coloneqq ker(\pi_{|_{\mathscr{G}_{24}}}) = \mathscr{G}_{24} \cap ker(\pi)$, che contiene tutti gli elementi di $\mathscr{G}_{24}$ che hanno come immagine tramite π lo zero, ossia:

$$K = \{\boldsymbol{x} \in \mathscr{G}_{24} \mid (x_1, \ldots, x_{12}) = \mathbf{0}\}$$

ma, poiché

$$\boldsymbol{x} = \sum_{i=0}^{11} \lambda_i \boldsymbol{v_i}$$

dove $\lambda_i \in \{0, 1\}$ e $\boldsymbol{v_i}$ rappresenta la i-esima riga di G, si ha che le prime 12 coordinate di $\boldsymbol{x}$ risultano nulle se e solo se

$$(\lambda_0, \ldots, \lambda_{10}) = (0, \ldots, 0).$$

Quindi l'unico parametro libero rimane $\lambda_{11} \in \{0, 1\}$. Di conseguenza $|ker(\pi_{|_{\mathscr{G}_{24}}})| = 2$.
Ora consideriamo $\boldsymbol{x}, \boldsymbol{y} \in \mathscr{G}_{24}$. Poiché $\pi(\boldsymbol{x}) = \pi(\boldsymbol{y})$ se e solo se $\pi(\boldsymbol{x} - \boldsymbol{y}) = \boldsymbol{0}$, se e solo se $\boldsymbol{x} - \boldsymbol{y} \in ker(\pi_{|_{\mathscr{G}_{24}}})$, abbiamo che $\boldsymbol{y}$ ha la stessa parte sinistra di $\boldsymbol{x}$ se e solo se $\boldsymbol{y} = \boldsymbol{x}$ oppure $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{z}$, dove $\boldsymbol{z}$ è l'ultima riga della matrice generatrice G. Ci sono, quindi, due possibilità per $\boldsymbol{y}$. □

Analizziamo ora tutte le possibili parole in $\mathscr{G}_{24}$.
Chiamiamo A_i il numero di parole di peso i. Allora $A_0 = A_{24} = 1$ (poiché abbiamo soltanto i vettori $\mathbf{1}$ e $\mathbf{0}$) e $A_8 = A_{16}$ (poiché è possibile costruire una biiezione tra A_8 e A_{16} tramite la funzione $f(\boldsymbol{x}) = \boldsymbol{x} + \mathbf{1}$).
Ricordiamo ora il lemma 2.5.3: esso afferma che per ogni sottomatrice sinistra L di G abbiamo due possibili sottomatrici destre: R e $\bar{R}$, ottenute considerando o meno nella combinazione lineare l'ultima riga della matrice G. Dunque notiamo che se $w(L) = 0$ allora $w(R) \neq 4$ (per il lemma 2.5.2) e $w(R) \neq 8$ (altrimenti $w(\bar{R}) = 4$, e questo viola nuovamente il lemma 2.5.2), perciò necessariamente $w(R) = 0$ oppure $w(R) = 12$. Similmente se $w(L) = 2$ allora $w(R) = 6$, e così via.
In questo modo si costruisce la seguente tabella:

Cardinalità	$w(L)$	$w(R)$	peso totale	$w(\bar{R})$	peso totale
1	0	0	0	12	12
$11+\binom{11}{2}$	2	6	8	6	8
$\binom{11}{3}+\binom{11}{2}$	4	4	8	8	12
$2(11+\binom{11}{2})$	6	2	8	10	16
792	6	6	12	6	12
$\binom{11}{7}+\binom{11}{8}$	8	4	12	8	16
$\binom{11}{9}+\binom{11}{10}$	10	6	16	6	16
1	12	0	12	12	24

Tabella 2.2: Possibili parole in $\mathscr{G}_{24}$.

Per completare la tabella abbiamo ragionato in questo modo: ad esempio, A_8 conta il numero di parole $\mid L \mid R \mid$ di $\mathscr{G}_{24}$ di peso pari a 8, le quali possono essere suddivise in tre tipologie:

$$w(L)=2, \quad w(R)=6$$

$$w(L)=4, \quad w(R)=4$$

$$w(L)=6 \quad w(R)=2.$$

Per contare quelle del primo tipo, possiamo considerare inizialmente solo la prima metà di parole del codice e suddividere il calcolo in due casi. Contiamo dapprima le parole che sono somma di due righe qualsiasi della matrice L (escludendo l'ultima che non ci dà alcun contributo): possiamo farlo in $\binom{11}{2}$ modi diversi. Successivamente contiamo quelle che sono ottenute prendendo una sola riga di L: abbiamo quindi 11 possibilità. A questo punto abbiamo due possibilità per la scelta di R: moltiplichiamo quindi per 2 il risultato precedente, ottenendo in totale $2(11+\binom{11}{2})$.
Analogamente per il terzo tipo (grazie all'osservazione 12).
Infine per quanto riguarda il secondo tipo, è possibile sommare tre oppure quattro righe dalla matrice L (escludendo l'ultima), cosa che è possibile fare in $\binom{11}{4}+\binom{11}{3}$ modi. In totale avremo perciò:

$$A_8 = 4\left(11+\binom{11}{2}\right)+\binom{11}{3}+\binom{11}{4}=759.$$

Con ragionamenti analoghi si ottiene:

Pesi i:	0	8	12	16	24
A_i:	1	759	2576	756	1

Tabella 2.3: Distribuzione di pesi in $\mathscr{G}_{24}$.

Definizione 2.12. Il *codice di Golay non esteso* $\mathscr{G}_{23}$ si ottiene eliminando l'ultima coordinata di ogni parola di $\mathscr{G}_{24}$. $\mathscr{G}_{23}$ è, quindi, per costruzione, un $[23, 11, 7]$-codice.

Teorema 2.5.4. *Il codice di Golay $\mathscr{G}_{23}$ è un codice perfetto 3-correttore.*

Dimostrazione. Per la dimostrazione si veda l'esempio 2.2. □

Avendo ora a disposizione questi risultati, vediamo che relazione intercorre tra i codici di Golay $\mathscr{G}_{23}$ e $\mathscr{G}_{24}$, e i relativi t-designs.

Corollario 2.5.5. *Le parole di peso 7 nel codice di Golay $\mathscr{G}_{23}$ formano un sistema di Steiner $S(4, 7, 23)$. Quindi per il corollario 2.3.4, esistono anche i sistemi di Steiner $S(3, 6, 22)$ e $S(2, 5, 21)$.*

Dimostrazione. Il risultato deriva da quanto già dimostrato nel teorema 2.4.2 e nel teorema 2.5.4. □

Lemma 2.5.6. *Un qualsiasi vettore binario di peso 5 e lunghezza 24 è coperto da esattamente una parola di $\mathscr{G}_{24}$ di peso 8.*

Dimostrazione. Ogni vettore di peso pari a 5 non può essere coperto da più di un vettore di peso 8 di $\mathscr{G}_{24}$, poiché se per assurdo esistessero due parole del codice $\boldsymbol{u}$ e $\boldsymbol{v}$ che svolgessero questo servizio, la loro distanza sarebbe minore o uguale a 6, una contraddizione poiché per costruzione la minima distanza di Hamming di $\mathscr{G}_{24}$ è $d = 8$.
Inoltre ogni parola di peso 8 copre $\binom{8}{5}$ vettori di peso 5 tutti distinti, e si ha:

$$\underbrace{759\binom{8}{5}}_{\text{numero di vettori di peso 5 coperti da quelli di peso 8}} = \underbrace{\binom{24}{5}}_{\text{numero di vettori di peso 5 in } \mathscr{G}_{24}}$$

e questo basta per concludere la dimostrazione del teorema. □

Corollario 2.5.7. *Le parole di peso 8 nel codice di Golay esteso $\mathscr{G}_{24}$ formano un sistema di Steiner $S(5, 8, 24)$.*

Dimostrazione. Combinando i risultati analizzati nel teorema 2.4.2 e quanto dimostrato nel lemma 2.5.6, la dimostrazione risulta ovvia. □

Teorema 2.5.8. *Il "triangolo di Pascal" relativo al sistema di Steiner formato da parole di $\mathscr{G}_{24}$ di peso 8 (le quali d'ora in poi saranno dette ottadi) è il seguente:*

$i\downarrow$													
1						759							
2					506		253						
3				330		176		77					
4			210		120		56		21				
5		130		80		40		16		5			
6	78		52		28		12		4		1		
7		46		32		20		8		4		0	1
8	30		16		16		4		4		0	0	1
9	30	0	16	0	4	0	0	0	1				

Dimostrazione. Per costruire tale "triangolo di Pascal" occorre ricordare che per definizione $\lambda_{ii} = \lambda_i$ per ogni $i \in \{0, \ldots, 8\}$ e per trovarli basta applicare il risultato trovato nel teorema 2.3.1. Infine si completa il resto della struttura utilizzando la proprietà di Pascal, dimostrata nel teorema 2.3.2. □

Lemma 2.5.9. *Un qualsiasi vettore binario di peso 5 e lunghezza 24 è coperto da esattamente 78 parole di $\mathscr{G}_{24}$ di peso 16.*

Dimostrazione. Sappiamo che in $\mathscr{G}_{24}$ esistono 759 parole di peso 16. Fissiamo una parola di peso 5. Chiamiamo a il numero di vettori di peso 16 che coprono tale parola, allora vale:

$$\frac{759}{a}\binom{16}{5} = \binom{24}{5}$$

perciò

$$a = 78.$$

□

Teorema 2.5.10. *Le parole di peso 16 in $\mathscr{G}_{24}$ formano un* 5-$(24, 16, 78)$ *design, e il corrispondente triangolo di Pascal si ottiene da quello rappresentato nella figura del teorema 2.5.8, omettendo le ultime tre righe e riflettendo la figura dalla colonna centrale, facendo in modo che diventi speculare.*

Dimostrazione. La prima asserzione deriva direttamente dal lemma 2.5.9, mentre la figura si costruisce con il solito metodo. □

Teorema 2.5.11. *Le parole di peso 12 in $\mathscr{G}_{24}$ (le quali d'ora in poi saranno dette dodecadi) formano un* 5-$(24, 12, 48)$ *design. Il "triangolo di Pascal" relativo al sistema di Steiner da esse formato (nel quale i numeri di intersezione dei blocchi λ_{ij} rappresentano il numero delle dodecadi contenenti $P_1, \ldots, P_j$ e non contenenti $P_{j+1}, \ldots, P_i$ con $0 \leq j \leq i \leq 8$) è il seguente:*

$i\downarrow$																	
1								2576									
2							1288		1288								
3						616		672		616							
4					280		336		336		280						
5				120		160		176		160		120					
6			48		72		88		88		72		48				
7		16		32		40		48		40		32		16			
8	0		16		16		24		24		16		16		0		
9	0		0		16		0		24		0		16		0		0

Dimostrazione. Per prima cosa mostriamo che $\lambda_{66} = \lambda_{76} = \lambda_{86} = 16$, per un motivo che sarà più chiaro in seguito.
Fissiamo un'ottade $\boldsymbol{x}$ e chiamiamo $P_1, \ldots, P_8$ i punti che essa contiene.
Esiste una corrispondenza biunivoca tra le dodecadi contenenti i punti $P_1, \ldots, P_6$ (che

non possono contenere P_7 o P_8), e le ottadi contenenti proprio P_7 e P_8 e non contenenti $P_1, \ldots, P_6$. Tale biiezione è la seguente: $f(\boldsymbol{a}) = \boldsymbol{a} - \boldsymbol{x}$ per ogni $\boldsymbol{a}$ dodecade del dominio.
Consideriamo per prima cosa una dodecade contenente i punti $P_1, \ldots, P_6$, facciamo vedere che non può contenere P_7 e P_8. Supponiamo per assurdo che contenga $P_1, \ldots, P_6, P_7, P_8$ ed altri quattro punti, ad esempio $P_9, \ldots, P_{12}$. Allora andando ad applicare la funzione f a tale dodecade, otteniamo una nuova ottade che però ha peso 4, non ammesso.
Se per assurdo, la dodecade contenesse i punti $P_1, \ldots, P_7$ e non P_8 e altri 5 punti arbitrari, allora applicando f otterremmo una nuova parola di peso 6, non ammesso. Analogo ragionamento se contenesse P_8 ma non P_7.
Facciamo ora vedere che la funzione f è una biiezione tra i due insiemi. Infatti, aggiungendo ad una dodecade del dominio di partenza, l'ottade $\boldsymbol{x}$, per la regola dei pesi dell'osservazione 4

$$w(\boldsymbol{x} + \boldsymbol{y}) = w(\boldsymbol{x}) + w(\boldsymbol{y}) - 2\#\{i \mid x_i = y_i = 1\}$$

otteniamo proprio una ottade del codominio. Il viceversa è analogo.
Di conseguenza, le quantità $\lambda_{66}, \lambda_{76}$ e λ_{86} relative alle dodecadi sono uguali tra loro, e per la biiezione corrispondono rispettivamente ai numeri $\lambda_{82}, \lambda_{71}$ e λ_{72} del triangolo di Pascal raffigurato nel teorema 2.5.8, e cioè valgono proprio 16.
A questo punto, ci interessa, per dimostrare il teorema, contare le dodecadi contenenti i punti $P_1, \ldots, P_5$. Per lo stesso ragionamento di prima, tali dodecadi devono contenere anche P_6, P_7 o P_8, altrimenti applicando f si otterrebbero parole di pesi non ammessi. Ma le dodecadi contenenti $P_1, \ldots, P_6$ sono 16, come anche quelle contenenti $P_1, \ldots, P_5, P_7$ e $P_1, \ldots, P_5, P_8$. Quindi $\lambda_{55} = 16 \times 3 = 48$. Abbiamo così dimostrato che le dodecadi formano un 5-$(24, 12, 48)$ design.
Il resto della figura può, infine, essere completato utilizzando i risultati già visti nel teorema 2.3.1 e attraverso la proprietà di Pascal. □

I triangoli di Pascal appena mostrati permettono di calcolare la distribuzione dei pesi delle parole di $\mathscr{G}_{23}$.

Teorema 2.5.12. *$\mathscr{G}_{23}$ è un $[23, 12, 7]$-codice, con distribuzione dei pesi data da:*

Pesi i:	0	7	8	11	12	15	16	23
A_i:	1	253	506	1288	1288	506	253	1

Tabella 2.4: Distribuzione di pesi in $\mathscr{G}_{23}$.

Dimostrazione. La costruzione della tabella segue immediatamente dalla rappresentazione del triangolo di Pascal per gli ottadi e per le parole di peso 12 e 16, in quanto basta, ad esempio, considerare le parole di peso 7 in $\mathscr{G}_{23}$ come ottadi in $\mathscr{G}_{24}$ con l'ultima coordinata pari a 1: il numero di queste è proprio $\lambda_{11} = 253$. Analogo ragionamento con le parole di peso 11 e 15 in $\mathscr{G}_{23}$. Per quelle di peso 8, basta pensare alle ottadi di $\mathscr{G}_{24}$ con l'ultima coordinata pari a 0, e sono proprio $\lambda_{10} = 506$. Si riempie infine la tabella, costruendo la biiezione $f(\boldsymbol{x}) = \boldsymbol{x} + \mathbf{1}$ per ogni $\boldsymbol{x} \in \mathscr{G}_{23}$. □

Applicazioni pratiche del Codice di Golay

Come già preannunciato, uno dei più grandi impieghi del codice di Golay esteso $\mathscr{G}_{24}$ riguardò, negli anni 1979, 1980 e 1981, la trasmissione di centinaia di immagini a colori di Giove e Saturno, immortalate dalle navicelle spaziali Voyager 1 e 2. Le immagini a colori solitamente richiedono il triplo di dati trasmessi rispetto a quelle in bianco e nero, per questo motivo il vecchio codice di Hadamard, utilizzato precedentemente durante le missioni della sonda Mariner, venne sostituito da uno più potente.

Figura 2.3: Giove e Saturno immortalati da Voyager 1, *1979/80*.

Inoltre, il codice di Golay esteso $\mathscr{G}_{24}$ viene tuttora utilizzato nelle comunicazioni radio. Infatti, i nuovi standard americani riguardo al protocollo per sondare la qualità di un collegamento radio articolato su diverse frequenze (*Automatic Link Establishment*, ALE) specificano l'utilizzo di blocchi di Golay per il controllo e la correzione di dati trasmessi tramite canali rumorosi (*Forward Error Correction*, FCE).

Bibliografia

[1] F.J. MacWilliams, N.J,A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland Mathematical Library, Volume 16, 1981.

[2] L. Cauchi, V. La Terra, R. Grasso, F. Gullo, *Appunti di Teoria dei Codici*, Facoltà di Ingegneria, Università di Catania.

[3] *Binary Golay Code*, Wikipedia, the free encyclopedia.

[4] F. Mazzocca, *Codici e Geometrie*, Testo della relazione tenuta a Caserta in occasione del Congresso Nazionale della Mathesis, 27 Ottobre 2011.

[5] Sito web della Nasa: https://www.nasa.gov/