

Teoria dell'informazione e della trasmissione

Indice

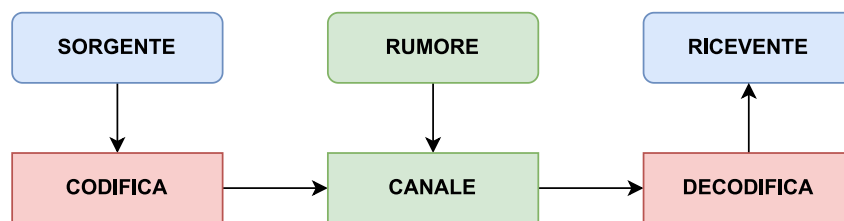
1. Lezione 01 [24/09]	3
2. Lezione 02 [01/10]	4
2.1. Introduzione storica	4
2.2. Cosa faremo	5
2.3. Esempio: informazione di un messaggio	5
2.4. Esempio: codice ZIP	5
2.5. Richiami matematici	6
3. Lezione 03 [04/10]	7
3.1. Codici	7
4. Lezione 04 [08/10]	10
4.1. Disuguaglianza di Kraft	10
5. Lezione 05 [11/10]	15
5.1. Entropia e tante dimostrazioni	15
5.2. Esercizi	19
6. Lezione 06 [15/10]	20
6.1. Molti teoremi	20
6.2. Esercizi	24
7. Lezione 07 [19/10]	26
7.1. Codice di Huffman	26
8. Lezione 08 [22/10]	30
8.1. Disuguaglianza di Kraft-McMillan	30
8.2. Esercizi	33
9. Lezione 09 [25/10]	36
9.1. Parenti dell'entropia	36
10. Lezione 10 [29/10]	40
10.1. Teoria	40
10.2. Esercizi	40
11. Lezione 11 [08/11]	42
11.1. Disuguaglianza di Fano	42
11.2. Canale	43
11.3. Esercizi	44
12. Lezione 12 [15/11]	46
12.1. Esercizi	46
13. Lezione 13 [22/11]	50
14. Lezione 14 [03/12]	55
14.1. Introduzione ai codici	55
14.1.1. Bit di parità	56
14.1.2. Codice ASCII	56
14.1.3. Burst di errori	56
14.1.4. Somma pesata	57
14.1.5. Codice ISBN	58
14.1.6. Codice UPC	58
14.2. Introduzione matematica	59

1. Lezione 01 [24/09]

2. Lezione 02 [01/10]

2.1. Introduzione storica

Lo schema di riferimento che andremo ad usare durante tutto il corso è il seguente:



La prima persona che lavorò alla teoria dell'informazione fu **Claude Shannon**, un impiegato della TNT (Telecom americana) al quale è stato commissionato un lavoro: massimizzare la quantità di dati che potevano essere trasmessi sul canale minimizzando il numero di errori che poteva accadere durante la trasmissione.

Nel 1948 pubblica un lavoro intitolato «A mathematical theory of communication», un risultato molto teorico nel quale modella in maniera astratta il canale e capisce come l'informazione può essere spedita «meglio» se rispetta certe caratteristiche. Ci troviamo quindi di fronte ad un risultato che non ci dice cosa fare nel caso specifico o che codice è meglio, ma è probabilistico, ti dice nel caso medio cosa succede.

Questo approccio è sicuramente ottimale, ma rappresenta un problema: va bene il caso medio, ma a me piacerebbe sapere cosa succede nel caso reale. Questo approccio più reale è quello invece seguito dal russo **Kolmogorov**, un accademico che vuole capire cosa succede nei singoli casi senza usare la probabilità. A metà degli anni '60 propone la sua idea di teoria dell'informazione, focalizzandosi quindi sui casi reali e non sui casi medi.

Questi due mostri sacri della teoria dell'informazione, nel nostro schema di lavoro, si posizionano nei rettangoli di sorgente e codifica, mentre la teoria della trasmissione si concentra sui rettangoli sottostanti.

Altri due personaggi che hanno lavorato allo stesso problema di Kolmogorov sono due russi, che lavoravano uno in America e uno nell'est del mondo, ma non sono ricordati perché Kolmogorov era il più famoso dei tre.

Un altro personaggio importante è **Richard Hamming**, un ricercatore di Bell Lab che doveva risolvere un problema: i job mandati in esecuzione dalle code batch delle macchine aziendali se si piantavano durante il weekend potevano far perdere un sacco di tempo. La domanda che si poneva Hamming era «che maroni, perché se le schede forate hanno errori, e le macchine lo sanno, io non posso essere in grado di correggerli?»

Lui sarà il primo che, per risolvere un problema pratico, costruisce il primo codice di rilevazione e correzione degli errori, il famoso e usatissimo **codice di Hamming**.

Vediamo alcune date che rivelano quanto è stata importante la teoria dell'informazione:

- 1965: prima foto di Marte in bianco e nero grande $\approx 240K$ bit, inviata con velocità 6 bit al secondo, ci metteva ore per arrivare a destinazione;
- 1969: stessa foto ma compressa, inviata con velocità 16K bit al secondo, ci metteva pochi secondi;
- 1976: prima foto di Marte a colori da parte di Viking;
- 1979: prima foto di Giove e delle sue lune a colori da parte di Voyager;

- anni '80: prima foto di Saturno e delle sue lune da parte di Voyager.

2.2. Cosa faremo

Nel corso vedremo due operazioni fondamentali per spedire al meglio un messaggio sul canale:

- **compressione:** dobbiamo ottimizzare l'accesso al canale, ovvero se possiamo mandare meno bit ma questi mi danno le stesse informazioni dei bit totali ben venga;
- **aggiunta di ridondanza:** dobbiamo aggiungere dei bit per permettere il controllo, da parte del ricevente, dell'integrità del messaggio.

La compressione riguarda la **source coding**, e viene rappresentata nel **primo teorema di Shannon**, mentre la ridondanza riguarda la **channel coding**, e viene rappresentata nel **secondo teorema di Shannon**.

Voglio massimizzare l'informazione da spedire sul canale ma alla quale devo aggiungere ridondanza.

Quello che fa Shannon è modellare il canale secondo una matrice stocastica, che quindi permette di sapere in media cosa succede evitando tutti i casi precisi.

IN/OUT	a	b	c	d	e
a	0.7	0.0	0.1	0.1	0.1
b	0.2	0.8	0.0	0.0	0.0
c	0.1	0.0	0.6	0.2	0.1
d	0.0	0.0	0.2	0.5	0.3
e	0.0	0.0	0.0	0.0	1.0

Questa matrice indica, per ogni carattere del nostro alfabeto, quale è la probabilità di spedire tale carattere e ottenere i vari caratteri presenti nell'alfabeto.

2.3. Esempio: informazione di un messaggio

Quando un messaggio contiene più informazioni di un altro?

Lancio n volte due monete, una truccata e una non truccata.

Quale delle due monete mi dà più informazioni? Sicuramente quella non truccata: il lancio della moneta «classica» è un evento randomico, non so mai cosa aspettarmi, mentre il lancio della moneta truccata è prevedibile, so già cosa succederà.

Possiamo quindi dire che un evento prevedibile porta pochissima informazione, mentre un evento imprevedibile porta tantissima informazione.

2.4. Esempio: codice ZIP

Uno dei codici di compressione più famosi è il codice **ZIP**.

Come funziona:

1. creo un dizionario, inizialmente vuoto, che contiene le coppie (stringa-sorgente, codifica), dove «codifica» è un numero;
2. usando un indice che scorre la stringa carattere per carattere, e partendo con numero di codifica uguale a 1, fino all'ultimo carattere eseguo iterativamente:
 1. parti da una stringa vuota che useremo come accumulatore;

2. aggiungi il carattere corrente all'accumulatore;
3. se l'accumulatore non è presente come chiave nel dizionario la aggiungo a quest'ultimo con codifica il numero corrente di codifica; riparto poi dal punto 2.1 con numero di codifica aumentato di 1;
4. se l'accumulatore è presente come chiave nel dizionario riparto dal punto 2.2.

Ad esempio, la stringa *AABACDABDCAABBA* viene codificata con:

- $A \rightarrow 1$;
- $A \rightarrow AB \rightarrow 2$;
- $A \rightarrow AC \rightarrow 3$;
- $D \rightarrow 4$;
- $A \rightarrow AB \rightarrow ABD \rightarrow 5$;
- $C \rightarrow 6$;
- $A \rightarrow AA \rightarrow 7$;
- $B \rightarrow 8$;
- $B \rightarrow BA \rightarrow 9$.

Prima avevo 15 caratteri, ora ne uso 9, letsgosky.

2.5. Richiami matematici

In questa parte abbiamo rivisto la definizione di monoide, gruppo, anello e campo, oltre alla definizione di generatore.

Abbiamo anche visto i **campi di Galois**, utilissimi per rendere il nostro calcolatore un campo algebrico, ovvero il campo $\mathbb{GF}(2^{64})$ dei polinomi di grado massimo 63 con coefficienti sul campo \mathbb{Z}_2 .

3. Lezione 03 [04/10]

Dalla lezione scorsa abbiamo capito che dobbiamo fare due cose:

1. massimizzare l'informazione trasmessa ad ogni utilizzo del canale; il «ogni utilizzo» è importante perché ogni volta che utilizzo il canale devo essere top, adesso lo devo essere. Se devo mandare n informazioni lo posso fare in n accessi oppure in 1 accesso, in entrambi i casi massimizzo l'informazione trasmessa ma non uso sempre tutta la banda possibile;
2. minimizzare il numero di errori di trasmissione.

Shannon userà la tecnica del Divide Et Impera, ovvero risolverà le due task separatamente e poi unirà i due risultati parziali. Per puro culo, questa soluzione sarà ottima (*non sempre succede*).

Il punto 1 riguarda la sorgente, il punto 2 riguarda la codifica e il canale.

3.1. Codici

Prima di vedere un po' di teoria dei codici diamo alcune basi matematiche.

Sia X un insieme di **simboli** finito. Un/a **messaggio/parola** $\bar{x} = x_1 \cdot \dots \cdot x_n \in X^n$ è una concatenazione di n caratteri x_i di X .

Dato $d > 1$, definiamo $D = \{0, \dots, d-1\}$ insieme delle **parole di codice**.

Definiamo infine $c : X \rightarrow D^+$ **funzione di codifica** che associa ad ogni carattere di X una parola di codice. Questa è una codifica in astratto: a prescindere da come è formato l'insieme X io uso le parole di codice che più mi piacciono (???). L'insieme D^+ è tale che

$$D^+ = \bigcup_{n \geq 1} \{0, \dots, d-1\}^n.$$

Voglio massimizzare la compressione: sia $l_c(x)$ la lunghezza della parola $x \in X$ nel codice c .

Quello che ho ora mi basta? NO: mi serve anche la **probabilità** $p(x)$ di estrarre un simbolo. Questo perché alle parole estratte molto spesso andremo a dare una lunghezza breve, mentre alle parole estratte di rado andremo a dare una lunghezza maggiore. Un codice che segue queste convenzioni è il codice morse.

Definiamo quindi il **modello sorgente** la coppia $\langle X, p \rangle$.

Ora va bene? NON ANCORA: noi non vogliamo la probabilità di estrarre un simbolo perché noi lavoriamo con le parole, non con i simboli.

Definiamo quindi $P_n(\bar{x} = x_1 \cdot \dots \cdot x_n) = \prod_{i=1}^n p(x_i)$

Qua vediamo una prima enorme semplificazione di Shannon: stiamo assumendo l'indipendenza tra più estrazioni consecutive, cosa che nelle lingue invece non è vera.

Il codice ZIP invece non assume l'indipendenza e lavora con la lingua che sta cercando di comprimere.

Dati il modello $\langle X, p \rangle$ e la base $d > 1$, dato il codice $c : X \rightarrow D^+$ il modello deve essere tale che

$$\mathbb{E}[l_c] = \sum_{x \in X} l_c(x) p(x)$$

sia minimo.

Il primo problema che incontriamo sta nella fase di decodifica: se codifico due simboli con la stessa parola di codice come faccio in fase di decodifica a capire quale sia il simbolo di partenza?

Imponiamo che il codice sia un **codice non singolare**, ovvero che la funzione c sia iniettiva. Stiamo imponendo quindi che il codominio sia abbastanza capiente da tenere almeno tutti i simboli di X .

Definiamo $C : X^+ \rightarrow D^+$ l'**estensione del codice** c che permette di codificare una parola intera.

Se c è non singolare, come è C ?

Purtroppo, la non singolarità non si trasmette nell'estensione del codice.

Andiamo a restringere l'insieme dei codici «buoni»: chiamiamo codici **univocamente decodificabili** (UD) i codici che hanno C non singolare.

Per dimostrare che un codice è UD esista l'algoritmo di Sardinas-Patterson, che lavora in tempo $O(mL)$, con $m = |X|$ e $L = \sum_{x \in X} l_c(x)$.

Ora però abbiamo un altro problema: gli UD non sono **stream**. In poche parole, un codice UD non ci garantisce di decodificare istantaneamente una parola di codice in un carattere di X quando mi arrivano un po' di bit, ma dovrei prima ricevere tutta la stringa e poi decodificare.

Sono ottimi codici eh, però ogni tanto aspetteremo tutta la codifica prima di poterla decodificare. Eh ma non va bene: in una stream non posso permettermi tutto ciò, e inoltre, se la codifica è veramente grande potrei non riuscire a tenerla tutta in memoria.

Potremmo utilizzare i **codici a virgola**, ovvero codici che hanno un carattere di terminazione per dire quando una parola è finita, e quindi risolvere il problema stream negli UD, ma noi faremo altro.

Restringiamo per l'ultima volta, definendo i **codici istantanei** (CI). Questi codici hanno la proprietà che stiamo cercando, ovvero permettono una decodifica stream, quindi non dobbiamo aspettare tutta la codifica prima di passare alla decodifica, ma possiamo farla appena riconosciamo una parola di codice.

Per capire se un codice è CI devo guardare i prefissi: nessuna parola di codice deve essere prefissa di un'altra. Questo controllo è molto più veloce rispetto al controllo che dovevo fare nei codici UD.

Ho quindi la seguente gerarchia:

$$CI \subset UD \subset NS \subset \text{codici}.$$

Ritorniamo all'obiettivo di prima: minimizzare la quantità

$$\mathbb{E}[l_c] = \sum_{x \in X} l_c(x)p(x).$$

Vediamo come, mettendo ogni volta dei nuovi vincoli sul codice, questo valore atteso aumenta, visto che vogliamo dei codici con buone proprietà ma questo va sprecato in termini di bit utilizzati. Inoltre, il codice che abbiamo sotto mano è il migliore?

A noi gli UD andavano bene (*stream esclusi*), quanto pago per andare nei codici istantanei?

Teorema 3.1.1: Se un codice è istantaneo allora è anche univocamente decodificabile.

Dimostrazione 3.1.1: Dimostro il contrario, ovvero che se un codice non è UD allora non è CI.

Sia $c : X \rightarrow D^+$ e sia C la sua estensione. Assumiamo che c sia non singolare. Se c non è UD allora esistono due messaggi x_1, x_2 diversi che hanno la stessa codifica $C(x_1) = C(x_2)$.

Per mantenere i due messaggi diversi possono succedere due cose:

- un messaggio è prefisso dell'altro: se x_1 è formato da x_2 e altri m caratteri, vuol dire che i restanti m caratteri di x_1 devono essere codificati con la parola vuota, che per definizione di codice non è possibile, e soprattutto la parola vuota sarebbe prefissa di ogni altra parola di codice, quindi il codice c non è istantaneo;
- esiste almeno una posizione in cui i due messaggi differiscono: sia i la prima posizione dove i due messaggi differiscono, ovvero $x_1[i] \neq x_2[i]$ e $x_1[j] = x_2[j]$ per $1 \leq j \leq i - 1$, ma allora $c(x_1[i]) \neq c(x_2[i])$ e $c(x_1[j]) = c(x_2[j])$ perché c è non singolare, quindi per avere la stessa codifica devo tornare al punto 1 e avere x_1 come prefisso di x_2 (o viceversa), ma, quindi otteniamo che c non è istantaneo.

Ma allora il codice non è istantaneo. ■

4. Lezione 04 [08/10]

4.1. Disuguaglianza di Kraft

I codici UD ci vanno bene (stream escluso), mentre i CI sono perfetti ma perché ci danno di più ma ci fanno spendere più bit: quanti?

Noi stavamo minimizzando il valore atteso delle lunghezze delle parole di codice, ovvero

$$\min_{c \in \mathcal{C}} \mathbb{E}[l_c] = \sum_{x \in X} l_c(x) p(x).$$

Cosa possiamo dire sui CI? Sono molto graditi perché, oltre a identificare un CI in maniera molto semplice guardando i prefissi, possiamo capire **se esiste** un codice istantaneo senza vedere il codice, ovvero guardando solo le lunghezze delle parole di codice.

Questa è una differenza abissale: prima guardavamo le parole di codice e controllavo i prefissi, ora non ho le parole, posso guardare solo le lunghezze delle parole di codice e, in base a queste, posso dire se esiste un CI con quelle lunghezze. Il problema principale è che non so che codice è quello istantaneo con quelle lunghezze, sto solo dicendo che sono sicuro della sua esistenza

Date le lunghezze, l'unica cosa che posso dire è se un CI non esiste (quando sono sbagliate le lunghezze date), ma non posso dire che il codice che ho dato con quelle lunghezze è un CI, perché questo ci dice solo della sua esistenza, non se il codice che ho dato con quelle lunghezze è CI, per quello dovrei guardare i prefissi.

Come mai seguire questa via e non quelle dei prefissi? Se il codice è molto grande, guardare questa proprietà è meno oneroso rispetto al guardare i prefissi.

Questa proprietà è detta **disuguaglianza di Kraft**.

Teorema 4.1.1 *Disuguaglianza di Kraft*: Data una sorgente $X = \{x_1, \dots, x_m\}$, data $d > 1$ base del codice e dati m interi $l_1, \dots, l_m > 0$ che mi rappresentano le lunghezze dei simboli del codice, esiste un CI $c : X \rightarrow D^+$ tale che

$$l_c(x_i) = l_i \quad \forall i = 1, \dots, m$$

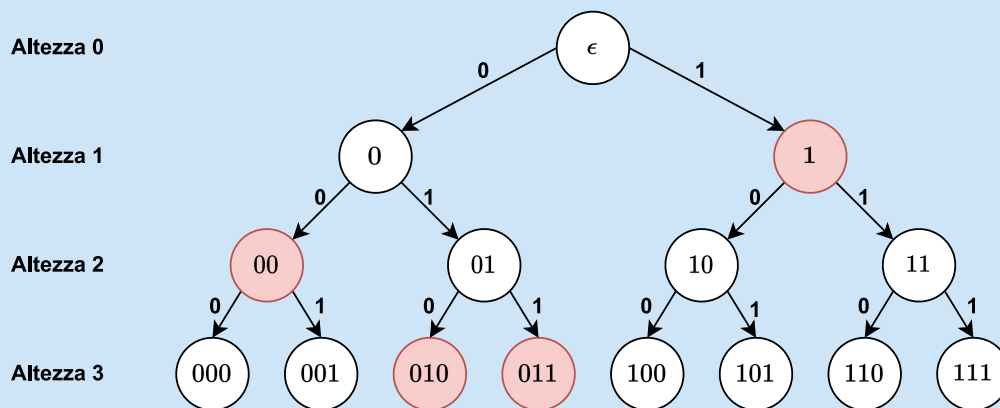
se e solo se

$$\sum_{i=1}^m d^{-l_i} \leq 1.$$

Dimostrazione 4.1.1:

(\Rightarrow) Esiste un CI con quelle proprietà, dimostro che vale la disuguaglianza di Kraft.

Sia c il nostro CI e d la base di c , dobbiamo definire la profondità del nostro codice. Possiamo usare un albero di codifica, ovvero un albero d -ario che rappresenta come sono codificate le varie parole di codice. Vediamo un breve esempio.



Vogliamo sapere

$$l_{\max} = \max_{i=1, \dots, m} l_c(x_i).$$

Costruiamo l'albero di codifica per il nostro CI e mettiamo dentro questo albero le parole del nostro codice. Come lo costruiamo? Creiamo l'albero d -ario alto l_{\max} completo e scegliamo, in questo albero, delle parole ad altezza $l_i \quad \forall i = 1, \dots, m$.

Dividiamo il nostro albero in sotto-alberi grazie alle parole che abbiamo inserito: ogni sotto-albero ha come radice una delle parole che abbiamo scelto. Contiamo quante foglie che ogni nodo copre. Sono tutti alberi disgiunti, visto che non posso avere prefissi essendo c un CI. Il numero massimo di foglie è $d^{l_{\max}}$, noi non le potremmo coprire tutte quindi

$$\sum_{i=1}^m |A_i| \leq d^{l_{\max}},$$

con A_i sotto-albero con radice la parola x_i . Notiamo che

$$\sum_{i=1}^m d^{l_{\max} - l_i} = \sum_{i=1}^m |A_i| \leq d^{l_{\max}}.$$

Ora possiamo dividere tutto per $d^{l_{\max}}$ e ottenere

$$\sum_{i=1}^m \frac{d^{l_{\max} - l_i}}{d^{l_{\max}}} = \sum_{i=1}^m d^{-l_i} \leq 1.$$

(\Leftarrow) Vale la disuguaglianza di Kraft, dimostro che esiste un CI con quelle proprietà.

Abbiamo le lunghezze che rendono vera la disuguaglianza di Kraft, quindi costruiamo l'albero di codifica: scegliamo un nodo ad altezza l_i e poi proibiamo:

- a tutti gli altri nodi di inserirsi nel suo sotto-albero;
- di inserire nodi che conterrebbero dei nodi già inseriti.

Una volta costruito l'albero vedo che rappresenta un CI, perché tutti i nodi non hanno nodi nel loro sotto-albero, quindi non ho prefissi per costruzione, quindi questo è un CI. ■

Il nostro obiettivo rimane sempre e comunque quello di cercare il codice migliore, quello che minimizza il valore atteso delle lunghezze di tale codice, ovvero vogliamo trovare delle lunghezze l_1, \dots, l_m tali che

$$\min_{l_1, \dots, l_m} \sum_{i=1}^m l_i p_i.$$

Non voglio solo minimizzare, voglio che valga anche Kraft, così da avere un codice che sia istantaneo, quindi devono valere contemporaneamente

$$\begin{cases} \min_{l_1, \dots, l_m} \sum_{i=1}^m l_i p_i \\ \sum_{i=1}^m d^{-l_i} \leq 1 \end{cases}$$

con $p_i = p(x_i) \quad \forall i = 1, \dots, m$. Cosa possiamo dire?

Notiamo che

$$\sum_{i=1}^m d^{-l_i} \leq \sum_{i=1}^m p_i = 1.$$

Per comodità, associamo p_i al simbolo x_i con lunghezza l_i . Allora guardiamo tutti i singoli valori della sommatoria, perché «se rispetto i singoli rispetto anche le somme», quindi

$$\begin{aligned} d^{-l_i} &\leq p_i \quad \forall i = 1, \dots, m \\ l_i &\geq \log_d \left(\frac{1}{p_i} \right). \end{aligned}$$

Con questa relazione ho appena detto come sono le lunghezze l_i del mio codice: sono esattamente

$$l_i \geq \left\lceil \log_d \left(\frac{1}{p_i} \right) \right\rceil.$$

Posso quindi costruire un codice mettendo in relazione le lunghezze del codice con la probabilità di estrarle dalla sorgente.

Esempio 4.1.1: Dati

$$X = \{x_1, x_2, x_3, x_4\} \quad | \quad P = \left\{ \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8} \right\} \quad | \quad d = 2$$

costruire un codice con la tecnica appena vista.

- $l_1 \geq \lceil \log_2(2) \rceil = 1 \rightarrow 0;$
- $l_2 \geq \lceil \log_2(4) \rceil = 2 \rightarrow 10;$
- $l_3 \geq \lceil \log_2(8) \rceil = 3 \rightarrow 110;$
- $l_4 \geq \lceil \log_2(8) \rceil = 3 \rightarrow 111.$

Questo codice ha valore atteso delle lunghezze

$$\mathbb{E}[l_c] = \frac{2}{4} + \frac{2}{4} + \frac{3}{4} = \frac{7}{4}.$$

Il valore atteso ora diventa

$$\mathbb{E}[l_c] = \sum_{i=1}^m p_i \log_d \left(\frac{1}{p_i} \right).$$

grazie alla nuova definizione delle lunghezze delle parole di codice.

La tecnica che associa ad ogni lunghezza un valore in base alla probabilità ci consente di generare un codice chiamato **codice di Shannon** (o *Shannon-Fano*. Il bro Fano era il dottorando di Shannon).

Inoltre, notiamo come il valore atteso delle lunghezze dipenda solo dalla distribuzione di probabilità dei simboli sorgente.

La quantità

$$\sum_{i=1}^m p_i \log_d \left(\frac{1}{p_i} \right)$$

viene chiamata **entropia**. Come vedremo, sarà una misura che definisce quanto i nostri codici possono essere compressi, una sorta di misura di compattezza: oltre quella soglia non posso andare sennò perdo informazione.

Questa costruzione dei codici di Shannon-Fano è molto bella:

- se ho una probabilità bassa di estrarre il simbolo x_i allora $\frac{1}{p_i}$ è grande e $\log_d \left(\frac{1}{p_i} \right)$ è grande;
- se ho una probabilità alta di estrarre il simbolo x_i allora $\frac{1}{p_i}$ è piccolo e $\log_d \left(\frac{1}{p_i} \right)$ è piccolo.

Purtroppo, i codici di Shannon-Fano non sono ottimali: una sorgente con due simboli a probabilità $p_1 = 0.1$ e $p_2 = 0.9$ darà un codice che non è ottimale.

Esempio 4.1.2: Siano

$$m = 4 \quad | \quad c : 1, 011, 01, 111.$$

Fai alcune considerazioni su questo codice.

Sicuramente non è CI: abbiamo dei prefissi che non vanno bene. Non è nemmeno UD: abbiamo una codifica ambigua per 111.

Esempio 4.1.3: Siano

$$m = 5 \quad | \quad c : 1, 001, 0000, 01, 0001.$$

Fai alcune considerazioni su questo codice.

Sicuramente è CI: non abbiamo prefissi, quindi è anche UD.

Posso vederlo come codice a virgola con il simbolo 1 che fa da separatore tra tutte le sequenze di 0.

Esempio 4.1.4: Siano

$$m = 5 \quad | \quad c : 000, 001, 01, 111, 110.$$

Fai alcune considerazioni su questo codice.

Sicuramente è CI: non abbiamo prefissi, quindi è anche UD.

Non è ottimale perché due foglie dell'albero non vengono coperte. Si potrebbe ridurre $\mathbb{E}[l_c]$ se scambiassimo 110 con 10 e 111 con 11.

Esempio 4.1.5: Siano

$$X = \{x_1, \dots, x_6\} \quad | \quad d = 2 \quad | \quad P = \left\{ \frac{1}{15}, \frac{1}{3}, \frac{1}{6}, \frac{1}{9}, \frac{1}{5}, \frac{1}{29} \right\}.$$

Costruisci un codice di Shannon per queste lunghezze.

Le probabilità non sommano a 1.

Esempio 4.1.6: Siano

$$X = \{x_1, \dots, x_6\} \quad | \quad d = 2 \quad | \quad P = \left\{ \frac{1}{12}, \frac{1}{3}, \frac{1}{5}, \frac{1}{3}, \frac{1}{72}, x \right\}.$$

Costruisci un codice di Shannon per queste lunghezze.

Le probabilità devono sommare a 1 quindi

$$p_6 = 1 - \left(\frac{1}{12} + \frac{1}{3} + \frac{1}{5} + \frac{1}{3} + \frac{1}{72} \right) = 1 - \frac{347}{360} = \frac{13}{360}.$$

- $l_1 \geq \lceil \log_2(12) \rceil = 4 \rightarrow 1100;$
- $l_2 \geq \lceil \log_2(3) \rceil = 2 \rightarrow 01;$
- $l_3 \geq \lceil \log_2(5) \rceil = 3 \rightarrow 100;$
- $l_4 \geq \lceil \log_2(3) \rceil = 2 \rightarrow 00;$
- $l_5 \geq \lceil \log_2(72) \rceil = 7 \rightarrow 1111111;$
- $l_6 \geq \lceil \log_2(\approx 28) \rceil = 5 \rightarrow 11010.$

5. Lezione 05 [11/10]

5.1. Entropia e tante dimostrazioni

Data la sorgente $\langle X, p \rangle$ associamo ad essa la variabile casuale $\mathbb{X} : X \rightarrow \{a_1, \dots, a_m\}$ tale che $P(\mathbb{X} = a_1) = p_1$. Abbiamo l'entropia d -aria $H_d(\mathbb{X}) = \sum_{i=1}^m p_i \log_d \left(\frac{1}{p_i} \right)$ che dipende solamente dalla distribuzione di probabilità della mia sorgente.

Cosa succede se vogliamo cambiare la base dell'entropia? Ricordiamo che

$$\log_d(p) = \frac{\ln(p)}{\ln(d)} = \frac{\ln(p)}{\ln(d)} \cdot \frac{\ln(a)}{\ln(a)} = \frac{\ln(p)}{\ln(a)} \cdot \frac{\ln(a)}{\ln(d)} = \log_a(p) \log_d(a)$$

quindi

$$H_d(\mathbb{X}) = \log_d(a) H_a(\mathbb{X}),$$

ovvero per cambiare la base dell'entropia pago una costante moltiplicativa $\log_d(a)$.

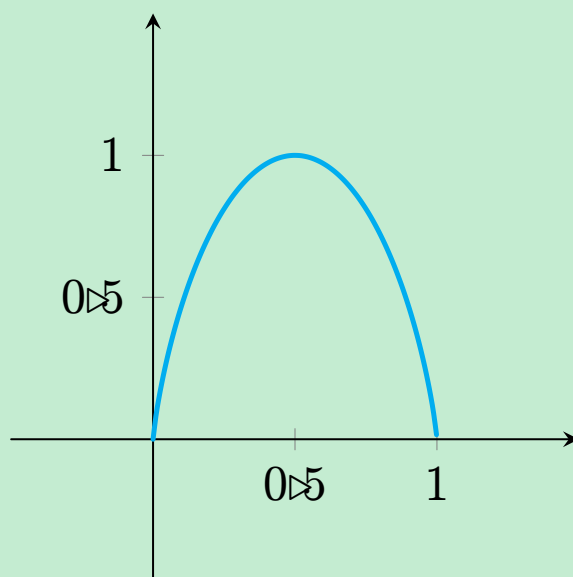
Esempio 5.1.1: Sia $\mathbb{X} : X \rightarrow \{0, 1\}$ variabile aleatoria bernoulliana tale che

$$P(\mathbb{X} = 1) = p \quad | \quad P(\mathbb{X} = 0) = 1 - p.$$

Calcoliamo l'entropia

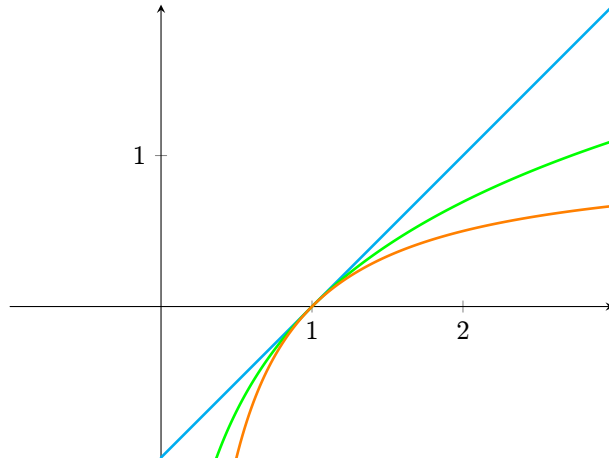
$$H(\mathbb{X}) = p \log_2 \left(\frac{1}{p} \right) + (1 - p) \log_2 \left(\frac{1}{1 - p} \right)$$

e vediamo quanto vale nell'intervallo $[0, 1]$.



Ce lo potevamo aspettare? Sì: nel caso di evento certo e evento impossibile io so esattamente quello che mi aspetta, quindi l'informazione è nulla, mentre in caso di totale incertezza l'informazione che posso aspettarmi è massima.

Vediamo ora due bound del logaritmo che ci permetteranno di dimostrare tanti bei teoremi.



Come vediamo dal grafico (senza label lol non sono capace) abbiamo che

$$1 - \frac{1}{x} \leq \ln(x) \leq x - 1.$$

Teorema 5.1.1: Sia \mathbb{X} una variabile casuale che assume i valori $\{a_1, \dots, a_m\}$, allora

$$H_d(\mathbb{X}) \leq \log_d(m) \quad \forall d > 1.$$

In particolare,

$$H_d(\mathbb{X}) = \log_d(m)$$

se e solo se \mathbb{X} è distribuita secondo un modello uniforme su $\{a_1, \dots, a_m\}$.

Dimostrazione 5.1.1: Andiamo a valutare $H_d(\mathbb{X}) - \log_d(m)$ per vedere che valore assume.

Consideriamo base dell'entropia e d come lo stesso valore: se così non fosse avremmo un fattore moltiplicativo davanti all'entropia che però non cambia la dimostrazione successiva.

Valutiamo quindi

$$\begin{aligned} H_d(\mathbb{X}) - \log_d(m) &= \sum_{i=1}^m p_i \log_d\left(\frac{1}{p_i}\right) - \log_d(m) = \sum_{i=1}^m p_i \log_d\left(\frac{1}{p_i}\right) - \sum_{i=1}^m p_i \log_d(m) = \\ &= \sum_{i=1}^m p_i \left(\log_d\left(\frac{1}{p_i}\right) - \log_d(m) \right) = \sum_{i=1}^m p_i \left(\log_d\left(\frac{1}{p_i m}\right) \right) \\ &\leq \sum_{i=1}^m p_i \left(\frac{1}{p_i m} - 1 \right) = \sum_{i=1}^m \frac{1}{m} - \sum_{i=1}^m p_i = 1 - 1 = 0. \end{aligned}$$

Ma allora

$$H_d(\mathbb{X}) - \log_d(m) \leq 0 \implies H_d(\mathbb{X}) \leq \log_d(m).$$

In particolare, se

$$P(X = a_i) = \frac{1}{m} \quad \forall i = 1, \dots, m$$

allora

$$H_d(\mathbb{X}) = \sum_{i=1}^m \frac{1}{m} \log_d(m) = m \frac{1}{m} \log_d(m) = \log_d(m). \quad \blacksquare$$

Nella scorsa lezione abbiamo detto che l'entropia ci dice quanto possiamo compattare il nostro messaggio prima che iniziamo a perdere informazioni. Per dimostrare questo bound introduciamo prima l'entropia relativa.

Siano \mathbb{X}, \mathbb{W} due variabili aleatorie definite sullo stesso dominio S , e siano $p_{\mathbb{X}}$ e $p_{\mathbb{W}}$ le distribuzioni di probabilit  delle due variabili aleatorie, allora l'**entropia relativa**

$$\mathbb{D}(\mathbb{X} \parallel \mathbb{W}) = \sum_{s \in S} p_{\mathbb{X}}(s) \log_d \left(\frac{p_{\mathbb{X}}(s)}{p_{\mathbb{W}}(s)} \right)$$

  la quantit  che misura la distanza che esiste tra le variabili aleatorie \mathbb{X} e \mathbb{W} . In generale $\mathbb{D}(\mathbb{X} \parallel \mathbb{W}) \neq \mathbb{D}(\mathbb{W} \parallel \mathbb{X})$ perch  essa non   una distanza metrica, ma solo una distanza in termini di diversit .

Teorema 5.1.2 *Information inequality:*

$$\mathbb{D}(\mathbb{X} \parallel \mathbb{W}) \geq 0.$$

Dimostrazione 5.1.2: Come nella dimostrazione precedente, se volessimo cambiare la base del logaritmo avremmo un fattore moltiplicativo davanti alla sommatoria, che per  non cambia la dimostrazione successiva. Noi manteniamo la base d in questa dimostrazione.

Valutiamo

$$\begin{aligned} \sum_{s \in S} p_{\mathbb{X}}(s) \log_d \left(\frac{p_{\mathbb{X}}(s)}{p_{\mathbb{W}}(s)} \right) &\geq \sum_{s \in S} p_{\mathbb{X}}(s) \left(1 - \frac{p_{\mathbb{W}}(s)}{p_{\mathbb{X}}(s)} \right) = \sum_{s \in S} p_{\mathbb{X}}(s) - p_{\mathbb{W}}(s) = \\ &= \sum_{s \in S} p_{\mathbb{X}}(s) - \sum_{s \in S} p_{\mathbb{W}}(s) = 1 - 1 = 0. \end{aligned}$$

Ma allora

$$\mathbb{D}(\mathbb{X} \parallel \mathbb{W}) \geq 0. \quad \blacksquare$$

Se $\mathbb{D}(\mathbb{X} \parallel \mathbb{W}) = 0$ allora ho spakkato, non ho distanza tra le due variabili aleatorie.

Vediamo finalmente il bound che ci d  l'entropia sulla compattezza.

Teorema 5.1.3: Sia $c : X \rightarrow D^+$ un CI d -ario per la sorgente $\langle X, p \rangle$, allora

$$\mathbb{E}[l_c] \geq H_d(\mathbb{X}).$$

Dimostrazione 5.1.3: Chiamo $\mathbb{W} : X \rightarrow \mathbb{R}$ una variabile casuale con una distribuzione di probabilit  $q(x)$ tale che

$$q(x) = \frac{d^{-l_c(x)}}{\sum_{x' \in X} d^{-l_c(x')}}.$$

Valutiamo

$$\begin{aligned} \mathbb{E}[l_c] - H_d(\mathbb{X}) &= \sum_{x \in X} l_c(x)p(x) - \sum_{x \in X} p(x) \log_d \left(\frac{1}{p(x)} \right) = \\ &= \sum_{x \in X} p(x) \left(l_c(x) - \log_d \left(\frac{1}{p(x)} \right) \right) = \\ &= \sum_{x \in X} p(x) \left(\log_d d^{l_c(x)} - \log_d \left(\frac{1}{p(x)} \right) \right) = \\ &= \sum_{x \in X} p(x) \log_d (p(x) d^{l_c(x)}) = \\ &= \text{massaggiamo pesantemente la formula} = \\ &= \sum_{x \in X} p(x) \log_d \left(\frac{p(x)}{d^{-l_c(x)}} \right) = \\ &= \sum_{x \in X} p(x) \log_d \left(\frac{p(x)}{d^{-l_c(x)}} \cdot \frac{\sum_{x' \in X} d^{-l_c(x')}}{\sum_{x' \in X} d^{-l_c(x')}} \right) = \\ &= \sum_{x \in X} p(x) \log_d \left(\frac{p(x)}{q(x)} \right) + \sum_{x \in X} p(x) \log_d \left(\frac{1}{\sum_{x' \in X} d^{-l_c(x')}} \right) = \\ &= \mathbb{D}(\mathbb{X} \parallel \mathbb{W}) + \log_d \left(\frac{1}{\sum_{x' \in X} d^{-l_c(x')}} \right) \sum_{x \in X} p(x) = \\ &= \mathbb{D}(\mathbb{X} \parallel \mathbb{W}) + \log_d \left(\frac{1}{\sum_{x' \in X} d^{-l_c(x')}} \right). \end{aligned}$$

Per l'information inequality sappiamo che

$$\mathbb{D}(\mathbb{X} \parallel \mathbb{W}) \geq 0,$$

mentre per la disuguaglianza di Kraft sappiamo che

$$\sum_{x \in X} d^{-l_c(x)} \leq 1$$

e quindi che

$$\frac{1}{\sum_{x \in X} d^{-l_c(x)}} \geq 1 \implies \log_d(\geq 1) \geq 0.$$

Otteniamo quindi che

$$\mathbb{E}[l_c] - H_d(\mathbb{X}) \geq 0 \implies \mathbb{E}[l_c] \geq H_d(\mathbb{X}).$$

■

Questo bound ci dice che un codice non può comunicare meno di quanto vale l'entropia di quella sorgente, indipendentemente dal codice scelto.

5.2. Esercizi

Vediamo l'algoritmo di Sardinas-Patterson per verificare se un codice dato è UD.

Dato l'insieme S_1 contenente le parole del codice c , l'algoritmo esegue i seguenti passi:

1. si parte da $i = 1$;
2. sia $x \in S_1$, se esiste $xy \in S_i$ allora $y \in S_{i+1}$;
3. sia $z \in S_i$, se esiste $zy \in S_1$ allora $z \in S_{i+1}$;
4. osserviamo S_{i+1} :
 - se $S_{i+1} = \emptyset$ allora c è UD;
 - se $S_{i+1} \cap S_1 \neq \emptyset$ allora c non è UD;
 - se S_{i+1} non rientra nei due casi precedenti ritornare al punto 2 con $i = i + 1$.

Esempio 5.2.1: Sia $S_1 = \{A, E, C, ABB, CED, BBEC\}$, il codice c con le parole di codice contenute in S_1 è UD?

- $S_1 = \{A, E, C, ABB, CED, BBEC\}$;
- $S_2 = \{BB, ED\}$;
- $S_3 = \{D\} \cup \{EC\} = \{D, EC\}$;
- $S_4 = \{C\} \cup \{\emptyset\} = \{C\}$.

Ma $S_1 \cap S_4 \neq \emptyset$ quindi c non è UD.

6. Lezione 06 [15/10]

6.1. Molti teoremi

Teorema 6.1.1: Per ogni sorgente $\langle X, p \rangle$ con:

- $X = \{x_1, \dots, x_m\}$ insieme dei simboli sorgente;
- $P = \{p_1, \dots, p_m\}$ probabilità associate ai simboli di X ;
- $c : X \rightarrow D^+$ codice di Shannon con lunghezze l_1, \dots, l_m tali che $l_i = l_c(x_i) \quad \forall i = 1, \dots, m$ costruite con

$$l_i = \left\lceil \log_d \left(\frac{1}{p_i} \right) \right\rceil \quad \forall i = 1, \dots, m$$

Vale

$$\mathbb{E}[l_c] < H_d(\mathbb{X}) + 1.$$

Dimostrazione 6.1.1: Verifichiamo che

$$\begin{aligned} \mathbb{E}[l_c] &= \sum_{i=1}^m p_i l_i = \sum_{i=1}^m p_i \left\lceil \log_d \left(\frac{1}{p_i} \right) \right\rceil \\ &< \sum_{i=1}^m p_i \left(\log_d \left(\frac{1}{p_i} \right) + 1 \right) = \sum_{i=1}^m p_i \log_d \left(\frac{1}{p_i} \right) + \sum_{i=1}^m p_i = H_d(\mathbb{X}) + 1. \end{aligned}$$

Che fastidio questo quadrato. ■

Ho trovato quindi un bound superiore alle lunghezze delle parole di codice di un codice di Shannon. Se lo uniamo al bound trovato nelle scorse lezioni otteniamo

$$H_d(\mathbb{X}) \leq \mathbb{E}[l_c] \leq H_d(\mathbb{X}) + 1.$$

Iniziamo a vedere però qualche problematica: questa risiede nel +1 dell'upper bound. Ogni volta che facciamo la codifica di un singolo carattere perdiamo poco, e questo «poco» è dato da quel +1. È una perdita locale, ma quando consideriamo un'intera parola o un intero messaggio la perdita si accumula.

Shannon si accorge di questa problematica perché usando l'estensione $C : X^+ \rightarrow D^+$ la lunghezza delle parole di codice è data da

$$l_C(x_1 \dots x_n) = \sum_{i=1}^n \left\lceil \log_d \left(\frac{1}{p_i} \right) \right\rceil.$$

Shannon allora propone una soluzione, che come vedremo non funzionerà nel caso reale.

Cosa fa Shannon? Vediamo un esempio.

Esempio 6.1.1: Siano:

- $l_c(x_1) = \lceil 2.5 \rceil \rightarrow 3$;
- $l_c(x_2) = \lceil 2.1 \rceil \rightarrow 3$;
- $l_c(x_3) = \lceil 2.1 \rceil \rightarrow 3$.

Come lunghezza totale della parola $x = x_1x_2x_3$ abbiamo 9.

Se pago al più un bit su ogni oggetto, il trucco che posso fare è fare il $\lceil \cdot \rceil$ della somma delle lunghezze, non la somma dei $\lceil \cdot \rceil$ delle lunghezze.

In questo caso otteniamo $\lceil 2.5 + 2.1 + 2.1 \rceil = \lceil 6.7 \rceil = 7$, che è minore di 9.

In poche parole ho «*spostato*» la sommatoria dentro $\lceil \cdot \rceil$.

Creiamo un nuovo codice: sia C_n un **codice a blocchi**, ed è un codice tale che

$$C_n : X^n \rightarrow D^+.$$

Sembra una soluzione fantastica, ma cosa ci nasconde Shannon? Vediamolo con un altro esempio.

Esempio 6.1.2: Sia $X = \{0, \dots, 9\}$, ho i codici c e C e mi viene chiesto di scrivere C_n con $d = 2$ e $n = 5$. Dove risiede il problema?

Dovrei codificare un numero enorme di parole di codice, in questo caso abbiamo 10^5 .

Ecco cosa stava nascondendo Shannon: la complessità del codice è enorme.

Vedremo dopo che se la dimensione del blocco cresce all'infinito il codice di Shannon che sto costruendo è ottimale, altrimenti stiamo sprecando delle informazioni.

Prendiamo un messaggio e lo dividiamo in blocchi di dimensione n numerandoli da 1 a m . Questi blocchi sono estratti dalla nostra sorgente $\langle X, p \rangle$. Ogni blocco è nella forma (x_1, \dots, x_n) , e contiene n simboli estratti tramite estrazioni indipendenti e identicamente distribuite, ovvero

$$p(x_1, \dots, x_n) = \prod_{i=1}^n p_i.$$

Questa quantità verrà indicata con

$$P_n(x_1, \dots, x_n).$$

Definisco una nuova sorgente $\langle X^n, P_n \rangle$ sulla quale definisco il mio codice a blocchi C_n . È una sorgente che pesca n oggetti da $\langle X, p \rangle$.

Vediamo cosa succede all'entropia di questa nuova sorgente. Ho n variabili casuali (n *mani*), ognuna che estrae un simbolo da X : devo calcolare quindi

$$\begin{aligned}
H_d(\mathbb{X}_1, \dots, \mathbb{X}_n) &= \sum_{x_1, \dots, x_n} P_n(x_1, \dots, x_n) \log_d \left(\frac{1}{P_n(x_1, \dots, x_n)} \right) = \\
&= \sum_{x_1} \dots \sum_{x_n} \left(\prod_{i=1}^n p(x_i) \right) \left(\log_d \left(\frac{1}{\prod_{i=1}^n p(x_i)} \right) \right) = \\
&= \sum_{x_1} \dots \sum_{x_n} \left(\prod_{i=1}^n p(x_i) \right) \left(\log_d \left(\prod_{i=1}^n p(x_i)^{-1} \right) \right) = \\
&= \sum_{x_1} \dots \sum_{x_n} \left(\prod_{i=1}^n p(x_i) \right) \left(\sum_{i=1}^n \log_d \left(\frac{1}{p(x_i)} \right) \right).
\end{aligned}$$

Come semplificare questo schifo? Vediamo il caso generale con un esempio.

Esempio 6.1.3: Sia $n = 2$, andiamo a calcolare l'entropia come

$$\begin{aligned}
H_d(\mathbb{X}_1, \mathbb{X}_2) &= \sum_{x_1} \sum_{x_2} \left(\prod_{i=1}^2 p(x_i) \right) \left(\sum_{i=1}^2 \log_d \left(\frac{1}{p(x_i)} \right) \right) = \\
&= \sum_{x_1} \sum_{x_2} (p(x_1)p(x_2)) \left(\log_d \left(\frac{1}{p(x_1)} \right) + \log_d \left(\frac{1}{p(x_2)} \right) \right) = \\
&= \sum_{x_1} \sum_{x_2} (p(x_1)p(x_2)) \log_d \left(\frac{1}{p(x_1)} \right) + p(x_1)p(x_2) \log_d \left(\frac{1}{p(x_2)} \right) = \\
&= \sum_{x_1} \sum_{x_2} (p(x_1)p(x_2)) \log_d \left(\frac{1}{p(x_1)} \right) + \sum_{x_1} \sum_{x_2} p(x_1)p(x_2) \log_d \left(\frac{1}{p(x_2)} \right) = \\
&= \left(\sum_{x_1} p(x_1) \log_d \left(\frac{1}{p(x_1)} \right) \right) \left(\sum_{x_2} p(x_2) \right) + \\
&+ \left(\sum_{x_1} p(x_1) \right) \left(\sum_{x_2} p(x_2) \log_d \left(\frac{1}{p(x_2)} \right) \right) = \\
&= H_d(\mathbb{X}_1) \cdot 1 + 1 \cdot H_d(\mathbb{X}_2) = H_d(\mathbb{X}_1) + H_d(\mathbb{X}_2).
\end{aligned}$$

Ma allora

$$\begin{aligned}
H_d(\mathbb{X}_1, \dots, \mathbb{X}_n) &= \sum_{x_1} \dots \sum_{x_n} \left(\prod_{i=1}^n p(x_i) \right) \left(\sum_{i=1}^n \log_d \left(\frac{1}{p(x_i)} \right) \right) = \\
&= H_d(\mathbb{X}_1) + \dots + H_d(\mathbb{X}_n) = nH_d(\mathbb{X}).
\end{aligned}$$

L'ultimo passaggio è vero perché tutte le variabili casuali stanno pescando dalla stessa sorgente con le stesse probabilità.

Teorema 6.1.2 *Primo teorema di Shannon:* Sia $C_n : X^n \rightarrow D^+$ codice a blocchi di Shannon d -ario per la sorgente $\langle X, p \rangle$ con

$$l_{C_n}(x_1, \dots, x_n) = \left\lceil \log_d \frac{1}{P_n(x_1, \dots, x_n)} \right\rceil,$$

allora

$$\lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}[l_{C_n}] = H_d(\mathbb{X}).$$

Dimostrazione 6.1.2: La dimostrazione è banale:

$$H_d(\mathbb{X}_1, \dots, \mathbb{X}_n) \leq \mathbb{E}[l_{C_n}] < H_d(\mathbb{X}_1, \dots, \mathbb{X}_n) + 1$$

$$nH_d(\mathbb{X}) \leq \mathbb{E}[l_{C_n}] \leq nH_d(\mathbb{X}) + 1$$

$$\frac{1}{n}(nH_d(\mathbb{X})) \leq \frac{1}{n}\mathbb{E}[l_{C_n}] \leq \frac{1}{n}(nH_d(\mathbb{X}) + 1)$$

$$H_d(\mathbb{X}) \leq \frac{1}{n}\mathbb{E}[l_{C_n}] \leq H_d(\mathbb{X}) + \frac{1}{n}.$$

Se passo al limite per $n \rightarrow \infty$, per il teorema dei due carabinieri vale

$$\lim_{n \rightarrow \infty} \frac{1}{n} \mathbb{E}[l_{C_n}] = H_d(\mathbb{X}).$$

■

Questo teorema ci dice che se tendiamo a ∞ la grandezza del blocco n allora il codice è ottimale, perché è uguale al lower bound che avevamo definito per $\mathbb{E}[l_{C_n}]$.

Cosa succede se per la sorgente $\langle X, p \rangle$ non ho la distribuzione p ma ho una distribuzione q di una variabile casuale \mathbb{Y} che ha campionato \mathbb{X} ?

Teorema 6.1.3: Data una sorgente $\langle X, p \rangle$, se $c : X \rightarrow D^+$ è un codice di Shannon avente lunghezze

$$l_c(x) = \left\lceil \log_d \frac{1}{q(x)} \right\rceil$$

dove q è una distribuzione di probabilità su X campionata con \mathbb{Y} , allora

$$\mathbb{E}[l_c] < H_d(\mathbb{X}) + 1 + \mathbb{D}(\mathbb{X} \parallel \mathbb{Y}).$$

Dimostrazione 6.1.3: Banale anche questo:

$$\begin{aligned}
\mathbb{E}[l_c] &= \sum_{x \in X} p(x) \left\lceil \log_d \left(\frac{1}{q(x)} \right) \right\rceil \\
&< \sum_{x \in X} p(x) \left(\log_d \left(\frac{1}{q(x)} \right) + 1 \right) = \\
&= \sum_{x \in X} p(x) \log_d \left(\frac{1}{q(x)} \right) + \sum_{x \in X} p(x) = \\
&= \sum_{x \in X} p(x) \log_d \left(\frac{p(x)}{p(x)q(x)} \right) + 1 = \\
&= \sum_{x \in X} p(x) \log_d \left(\frac{p(x)}{q(x)} \right) + \sum_{x \in X} p(x) \log_d \left(\frac{1}{p(x)} \right) + 1 = \\
&= H_d(\mathbb{X}) + 1 + \mathbb{D}(\mathbb{X} \parallel \mathbb{Y}).
\end{aligned}$$

Odio il quadratino. ■

6.2. Esercizi

Esempio 6.2.1: Sia $S_1 = \{A, E, C, ABB, CED, BBEC\}$ l'insieme dell'esercizio della scorsa lezione, eseguire Sardinas-Patterson ma non fermarsi quando si trova un elemento che apparteneva a S_1 .

- $S_1 = \{A, E, C, ABB, CED, BBEC\}$;
- $S_2 = \{BB, ED\}$;
- $S_3 = \{D\} \cup \{EC\} = \{D, EC\}$;
- $S_4 = \{C\} \cup \{\emptyset\} = \{C\}$;
- $S_5 = \{ED\}$;
- $S_6 = \{D\}$;
- $S_7 = \emptyset$.

L'algoritmo accetterebbe c anche se non è UD.

Esempio 6.2.2: Sia $S_1 = \{0, 01, 10\}$, il codice c con le parole di codice contenute in S_1 è UD?

- $S_1 = \{0, 01, 10\}$;
- $S_2 = \{1\}$;
- $S_3 = \{0\}$.

Ma $S_1 \cap S_3 \neq \emptyset$ quindi c non è UD.

Se non ci fermiamo quando troviamo un elemento di S_1 otteniamo:

- $S_4 = \{1\}$.

Ma $S_4 = S_2$, quindi abbiamo trovato un insieme che abbiamo già visitato e quindi il codice c non viene accettato di nuovo.

Va aggiunto infatti un altro caso di terminazione a Sardinas-Patterson, ovvero l'algoritmo termina non accettando il codice c come UD quando trova un insieme che è già stato incontrato precedentemente.

Esempio 6.2.3: Sia

$$P = \left\{ \frac{1}{3}, \frac{1}{4}, \frac{1}{6}, \frac{1}{8}, \frac{1}{9}, x \right\}.$$

Calcolare le lunghezze di codice per un codice di Shannon c che abbia 6 simboli estratti le probabilità di P .

Calcoliamo

$$x = 1 - \left(\frac{1}{3} + \frac{1}{4} + \frac{1}{6} + \frac{1}{8} + \frac{1}{9} \right) = 1 - \frac{24 + 18 + 12 + 9 + 8}{72} = 1 - \frac{71}{72} = \frac{1}{72}.$$

Calcoliamo le lunghezze come:

- $l_1 = \lceil \log_2 3 \rceil \rightarrow 2;$
- $l_2 = \lceil \log_2 4 \rceil \rightarrow 2;$
- $l_3 = \lceil \log_2 6 \rceil \rightarrow 3;$
- $l_4 = \lceil \log_2 8 \rceil \rightarrow 3;$
- $l_5 = \lceil \log_2 9 \rceil \rightarrow 4;$
- $l_6 = \lceil \log_2 72 \rceil \rightarrow 7.$

L'albero di codifica non copre tutte le foglie: infatti, questo codice non è ottimale perché ci sono dei rami dell'albero che possiamo potare.

7. Lezione 07 [19/10]

7.1. Codice di Huffman

Il codice di Shannon era un codice interessante, aveva dei buoni bound ma non era il CI ottimale. Inoltre, era impraticabile se la grandezza del blocco cresceva.

Vediamo il **codice di Huffman**. Data una sorgente $\langle X, p \rangle$ e dato $d > 1$, l'algoritmo per creare il codice di Huffman per la sorgente esegue i seguenti passi:

- ordina le probabilità p_i in maniera decrescente;
- rimuovi i d simboli meno probabili da X e crea una nuova sorgente aggiungendo un nuovo simbolo che abbia come probabilità la somma delle probabilità dei simboli rimossi;
- se la nuova sorgente ha più di d simboli torna al punto 1.

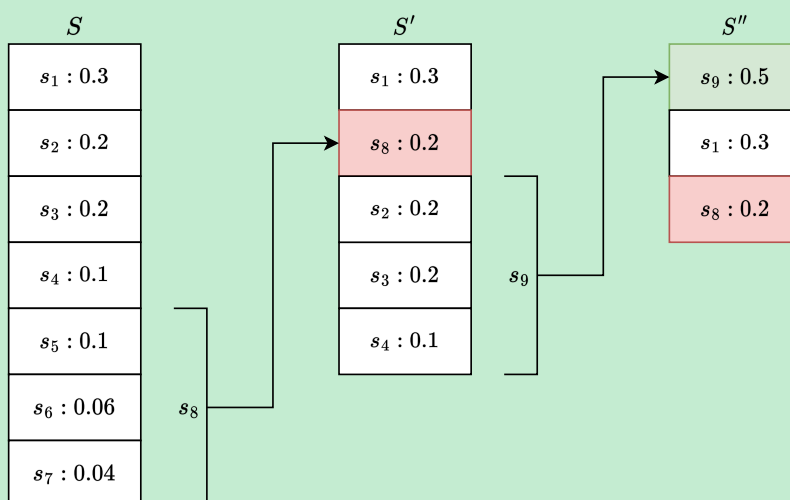
Come funziona nella realtà? Vediamolo con un esempio.

Esempio 7.1.1: Data la sorgente $\langle S, p \rangle$ con:

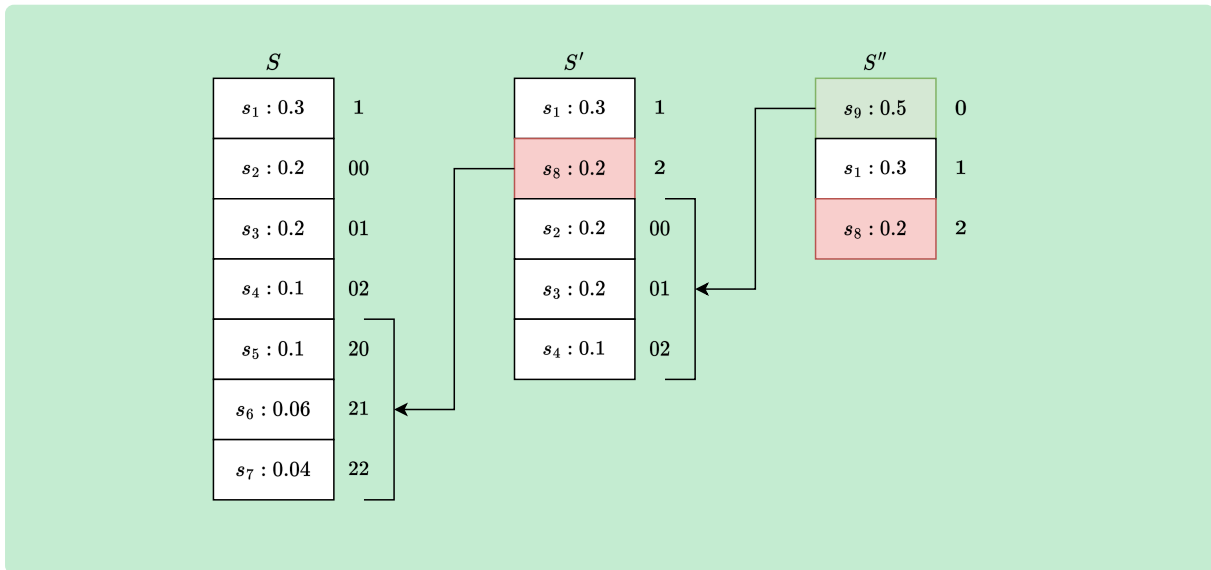
- $S = \{s_1, \dots, s_7\}$ e
- $P = \{0.3, 0.2, 0.2, 0.1, 0.1, 0.06, 0.04\}$,

trovare il codice di Huffman ternario per questa sorgente.

Nella prima fase andiamo a comprimere i simboli meno probabili arrivando a definire la sorgente S'' dopo due iterazioni dell'algoritmo.



Nella seconda fase invece andiamo a fare un rollback: a partire dall'ultima sorgente creata andiamo ad assegnare i simboli di D ai simboli sorgente correnti, propaghiamo i simboli alla sorgente precedente e andiamo ad eseguire lo stesso procedimento per i d simboli che sono stati compattati.



Questo codice è un codice che a noi piace molto perché rispetta due condizioni per noi fondamentali: minimizza il valore atteso delle lunghezze delle parole di codice e rispetta Kraft. Infatti:

$$CH = \left\{ \min_{l_1, \dots, l_m} \sum_{i=1}^m l_i p_i \mid \sum_{i=1}^m d^{-l_i} \leq 1 \right\}.$$

Lemma 7.1.1: Sia c un codice d -ario di Huffman per la sorgente $\langle X', p' \rangle$ con $X' = \{x_1, \dots, x_{m-d+1}\}$ e probabilità $p_1 \geq \dots \geq p_{m-d+1}$. Data ora la sorgente $\langle X, p \rangle$ con $X = \{x_1, \dots, x_m\}$ costruita da X' togliendo il simbolo x_k e aggiungendo d simboli x_{m-d+2}, \dots, x_m con probabilità $p_k \geq p_{m-d+2} \geq \dots \geq p_m$ tali che $\sum_{i=2}^d p_{m-d+i} = p_k$. Allora

$$c(x) = \begin{cases} c'(x) & \text{se } x \neq x_k \\ c'(x_k) \cdot i & \text{se } k \in \{m-d+2, \dots, m\} \wedge \forall i \in D \end{cases}$$

è un codice di Huffman per la sorgente X .

Grazie a questo lemma ora possiamo dimostrare un risultato importante sul codice di Huffman.

Teorema 7.1.1: Data la sorgente $\langle X, p \rangle$ e dato $d > 1$, il codice d -ario di Huffman c minimizza $\mathbb{E}[l_c]$ tra tutti i codici d -ari per la medesima sorgente.

Dimostrazione 7.1.1: Dimostriamo per induzione su m .

Il passo base è $m = 2$, quindi abbiamo due simboli x_1, x_2 che, indipendentemente dalla probabilità assegnatagli, avranno 0 e 1 come codifica, che è minima.

Assumiamo ora che Huffman sia ottimo per sorgenti di grandezza $m - 1$ e dimostriamo che sia ottimo per sorgenti di grandezza m .

Fissata $\langle X, p \rangle$ sorgente di m simboli, siano $u, v \in X$ i due simboli con le probabilità minime. Costruiamo la sorgente $\langle X', p' \rangle$ dove u, v sono sostituiti da z tale che

$$p'(x) = \begin{cases} p(x) & \text{se } x \neq z \\ p(u) + p(v) & \text{se } x = z \end{cases}.$$

Ho $m - 1$ simboli, quindi per ipotesi induttiva c' è un codice di Huffman ottimo. Per il lemma precedente, anche il codice c è di Huffman ed è tale che

$$c(x) = \begin{cases} c'(x) & \text{se } x \neq u \wedge x \neq v \\ c'(x) \cdot 0 & \text{se } x = u \\ c'(x) \cdot 1 & \text{se } x = v \end{cases}.$$

Dimostriamo che il codice c è ottimo.

Calcoliamo il valore atteso delle lunghezze delle parole del codice c come

$$\begin{aligned} \mathbb{E}[l_c] &= \sum_{x \in X} l_c(x)p(x) = \\ &= \sum_{x \in X'} l_{c'}(x)p'(x) - l_{c'}(z)p'(z) + l_c(u)p(u) + l_c(v)p(v) = \\ &= \mathbb{E}[l_{c'}] - l_{c'}(z)p'(z) + (l_{c'}(z) + 1)p(u) + (l_{c'}(z) + 1)p(v) = \\ &= \mathbb{E}[l_{c'}] - l_{c'}(z)p'(z) + (l_{c'}(z) + 1)(p(u) + p(v)) = \\ &= \mathbb{E}[l_{c'}] - l_{c'}(z)p'(z) + (l_{c'}(z) + 1)p'(z) = \\ &= \mathbb{E}[l_{c'}] - l_{c'}(z)p(z) + l_{c'}(z)p'(z) + p'(z) = \\ &= \mathbb{E}[l_{c'}] + p'(z). \end{aligned}$$

Per dimostrare l'ottimalità di c consideriamo un altro codice c_2 per la sorgente $\langle X, p \rangle$ e verifichiamo che $\mathbb{E}[l_c] \leq \mathbb{E}[l_{c_2}]$. Sia c_2 istantaneo per $\langle X, p \rangle$ e siano $r, s \in X$ tali che $l_{c_2}(r)$ e $l_{c_2}(s)$ sono massime. Senza perdita di generalità assumiamo che r, s siano fratelli nell'albero di codifica di c_2 . Infatti:

- se sono fratelli GG, godo;
- se non sono fratelli ma uno tra r e s ha un fratello (sia f fratello di s ad esempio) andiamo a scegliere s e f al posto di s e r ;
- se non sono fratelli perché sono su due livelli diversi (*a distanza uno*) possiamo sostituire la codifica di quello più basso con quella del padre e ritornare in una delle due situazioni precedenti.

Definiamo il codice \bar{c}_2 tale che

$$\bar{c}_2 = \begin{cases} c_2(x) & \text{se } x \notin \{u, v, r, s\} \\ c_2(u) & \text{se } x = r \\ c_2(r) & \text{se } x = u \\ c_2(v) & \text{se } x = s \\ c_2(s) & \text{se } x = v \end{cases}.$$

In poche parole, abbiamo scambiato r con u e s con v .

Analizziamo $\mathbb{E}[l_{\bar{c}_2}] - \mathbb{E}[l_{c_2}]$ per dimostrare che il primo è minore o uguale del secondo. Notiamo prima di tutto che i simboli $x \notin \{u, v, r, s\}$ non compaiono nel conto successivo: questo

perché nei due codici hanno lo stesso contributo in probabilità e lunghezza, quindi consideriamo solo i simboli appena citati visto che vengono scambiati.

$$\begin{aligned}
\mathbb{E}[l_{\bar{c}_2}] - \mathbb{E}[l_{c_2}] &= p(r)l_{c_2}(u) + p(u)l_{c_2}(r) + p(s)l_{c_2}(v) + p(v)l_{c_2}(s) \\
&\quad - p(r)l_{c_2}(r) - p(u)l_{c_2}(u) - p(s)l_{c_2}(s) - p(v)l_{c_2}(v) = \\
&= p(r)(l_{c_2}(u) - l_{c_2}(r)) + p(u)(l_{c_2}(r) - l_{c_2}(u)) + \\
&\quad + p(s)(l_{c_2}(v) - l_{c_2}(s)) + p(v)(l_{c_2}(s) - l_{c_2}(v)) = \\
&= (p(r) - p(u))(l_{c_2}(u) - l_{c_2}(r)) + (p(s) - p(v))(l_{c_2}(v) - l_{c_2}(s)).
\end{aligned}$$

Ma noi sappiamo che:

- $p(r) - p(u) \geq 0$ dato che u è un simbolo a probabilità minima;
- $l_{c_2}(u) - l_{c_2}(r) \leq 0$ dato che r è un simbolo a lunghezza massima;
- $p(s) - p(v) \geq 0$ dato che v è un simbolo a probabilità minima;
- $l_{c_2}(v) - l_{c_2}(s) \leq 0$ dato che s è un simbolo a lunghezza massima.

Stiamo sommando due quantità negative, quindi

$$\mathbb{E}[l_{\bar{c}_2}] - \mathbb{E}[l_{c_2}] \leq 0 \implies \mathbb{E}[l_{\bar{c}_2}] \leq \mathbb{E}[l_{c_2}].$$

Introduciamo ora il codice c'_2 fatto come segue:

$$c'_2 = \begin{cases} \bar{c}_2(x) & \text{se } x \neq z \\ \omega & \text{se } x = z \end{cases}.$$

Questo codice è definito sulla sorgente $\langle X', p' \rangle$. Ma allora

$$\begin{aligned}
\mathbb{E}[l_{\bar{c}_2}] &= \sum_{x \in X' \mid x \neq z} p'(x)l_{\bar{c}_2}(x) + p'(z)(l_{c'_2}(z) + 1) + p'(v)(l_{c'_2}(z) + 1) = \\
&= \sum_{x \in X' \mid x \neq z} p'(x)l_{\bar{c}_2}(x) + p'(z)l_{c'_2}(z) + p'(z) = \\
&= \mathbb{E}[l_{c'_2}] + p'(z) \\
&\geq \mathbb{E}[l_{c'}] + p'(z).
\end{aligned}$$

Mettendo insieme i due risultati otteniamo

$$\mathbb{E}[l_c] = \mathbb{E}[l_{c'}] + p'(z) \leq \mathbb{E}[l_{c'_2}] + p'(z) = \mathbb{E}[l_{\bar{c}_2}] \leq \mathbb{E}[l_{c_2}].$$

Ma allora

$$\mathbb{E}[l_c] \leq \mathbb{E}[l_{c_2}].$$

■

8. Lezione 08 [22/10]

8.1. Disuguaglianza di Kraft-McMillan

Per ora abbiamo sempre considerato dei CI perché ci davano una serie di proprietà belle, ma ce l'abbiamo un codice UD che abbia lunghezze minime e che magari sia meglio di un CI? Se rilasso il vincolo *no-stream* esiste negli UD una cosa buona come Huffman o anche di meglio?

Riprendiamo alcuni concetti utili per il prossimo teorema.

Che codici abbiamo visto fin'ora? Ne abbiamo visti tre:

- $c : X \rightarrow D^+$ codifica iniziale;
- $C : X^+ \rightarrow D^+$ estensione di c ;
- $C_n : X^n \rightarrow D^+$ codice a blocchi.

A noi ora non serve il codice a blocchi ma solo l'estensione, ma modifichiamola leggermente: sia $C_k : X^k \rightarrow D^+$ l'estensione k -esima del codice c , con $k \geq 1$, tale che

$$C_k(x_1, \dots, x_k) = c(x_1) \cdot \dots \cdot c(x_k).$$

Il codice c deve essere sicuramente non singolare, ma anche C_k lo deve essere, quindi il codice è UD.

Le lunghezze di C_k sono

$$l_{C_k}(x_1 \dots x_k) = \sum_{i=1}^k l_c(x_i).$$

Per convenzione sia

$$l_{\max} = \max_{i=1, \dots, k} l_c(x_i).$$

Ma allora

$$l_{C_k}(x_1, \dots, x_k) \leq k l_{\max}.$$

Siamo pronti per dimostrare il seguente teorema.

Teorema 8.1.1 *Disuguaglianza di Kraft-McMillan*: Data una sorgente $X = \{x_1, \dots, x_m\}$, data $d > 1$ base del codice e dati m interi $l_1, \dots, l_m > 0$ che mi rappresentano le lunghezze dei simboli del codice, esiste un codice UD $c : X \rightarrow D^+$ tale che

$$l_c(x_i) = l_i \quad \forall i = 1, \dots, m$$

se e solo se

$$\sum_{i=1}^m d^{-l_i} \leq 1.$$

Uguale alla disuguaglianza di Kraft nei CI, ma qua viene definiti negli UD.

Dimostrazione 8.1.1: Dimostriamo la doppia implicazione.

$\{\Leftarrow\}$

Pezzo gratis: per la disuguaglianza di Kraft sui CI esiste un CI, ma i CI sono anche UD, quindi questa implicazione è vera.

$\{\Rightarrow\}$

Partiamo dal valore

$$\sum_{i=1}^m d^{-l_i} = \sum_{x \in X} d^{-l_c(x)}.$$

e vediamo che valori assume la quantità

$$\left(\sum_{x \in X} d^{-l_c(x)} \right)^k.$$

Come possiamo riscrivere la potenza di una sommatoria? Vediamolo con un esempio.

Esempio 8.1.1:

$$\left(\sum_i a_i \right)^2 = \sum_i a_i \sum_j a_j = \sum_i \sum_j a_i a_j.$$

Visto questo esempio, possiamo dire che

$$\begin{aligned} \left(\sum_{x \in X} d^{-l_c(x)} \right)^k &= \sum_{x_1 \in X} \dots \sum_{x_k \in X} d^{-l_c(x_1)} \cdot \dots \cdot d^{-l_c(x_k)} = \\ &= \sum_{x_1, \dots, x_k \in X^k} d^{-(l_c(x_1) + \dots + l_c(x_k))} = \\ &= \sum_{x_1 \dots x_k \in X^k} d^{-l_{C_k}(x_1, \dots, x_k)}. \end{aligned}$$

Cosa abbiamo dentro l'insieme X^k ? Abbiamo tutte le possibili combinazioni (*in realtà disposizioni*) di k simboli, ma come sono le lunghezze di queste parole?

La minima è sicuramente k , e questo succede quando ad ogni simbolo $x_i \mid i = 1, \dots, k$ assegno la lunghezza 1 (*in questo caso dovrei avere $m \leq d$ senno il codice è singolare e non va bene*), mentre la massima è kl_{\max} , come abbiamo mostrato prima.

Andiamo a partizionare l'insieme X^k in insiemi X_t^k , ognuno dei quali contiene delle parole di k simboli lunghi in totale $t \geq 1$. In poche parole

$$X_t^k = \left\{ (x_1, \dots, x_k) \in X^k \mid l_{C_k}(x_1, \dots, x_k) = t \right\}.$$

Nell'insieme $X_{kl_{\max}}^k$ abbiamo le parole di lunghezza massima.

Riscriviamo la sommatoria come

$$\begin{aligned}
\sum_{x_1 \dots x_k \in X^k} d^{-l_{C_k}(x_1, \dots, x_k)} &= \sum_{n=1}^{kl_{\max}} \sum_{(x_1, \dots, x_k) \in X_n^k} d^{-l_{C_k}(x_1, \dots, x_k)} = \\
&= \sum_{n=1}^{kl_{\max}} \sum_{(x_1, \dots, x_k) \in X_n^k} d^{-n} = \\
&= \sum_{n=1}^{kl_{\max}} |X_n^k| d^{-n}.
\end{aligned}$$

La nostra funzione C_k è iniettiva (*non singolare*) perché il codice è UD, quindi il codominio di questo codice deve essere più grande o al massimo uguale al dominio. Il dominio è X_t^k , il codominio è D^n , quindi $|X_n^k| \leq |D^n| = d^n$, quindi

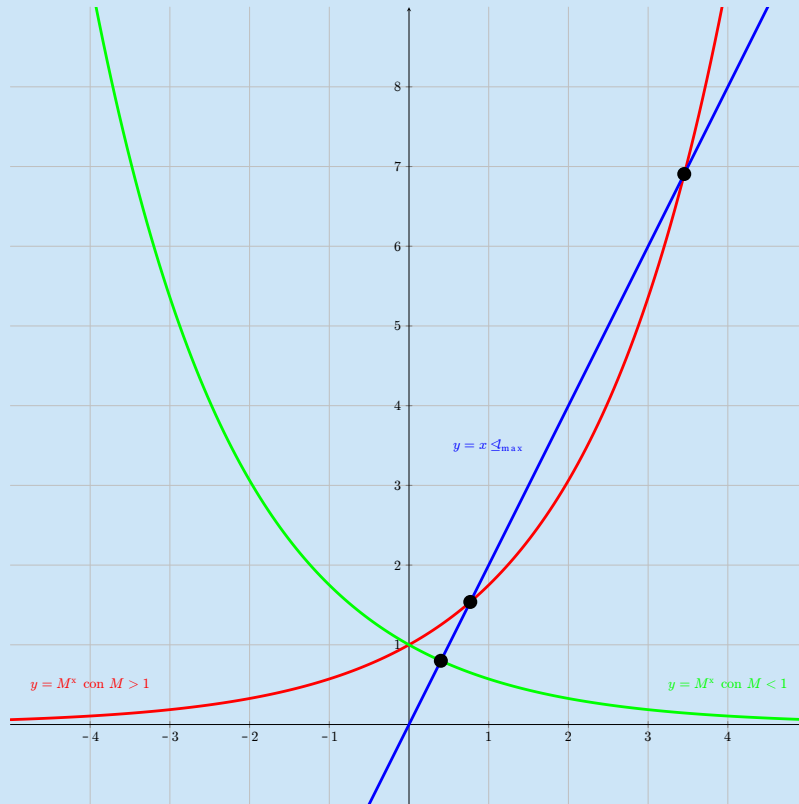
$$\sum_{n=1}^{kl_{\max}} |X_n^k| d^{-n} \leq \sum_{n=1}^{kl_{\max}} d^n d^{-n} = \sum_{n=1}^{kl_{\max}} 1 = kl_{\max}.$$

Siamo arrivati ad avere, dopo tutta sta catena, alla relazione

$$\left(\sum_{x \in X} d^{-l_c(x)} \right)^k \leq kl_{\max}.$$

Chiamo $M = \sum_{x \in X} d^{-l_c(x)}$, quindi devo verifica che

$$M^k \leq kl_{\max}.$$



Se disegno M^k ottengo due grafici diversi:

- se $M > 1$ ho un esponenziale crescente, che da un certo punto k_0 rende falsa la relazione;
- se $0M \leq 1$ ho un esponenziale decrescente, che da un certo punto k_0 rende vera la relazione.

Ma allora

$$M \leq 1 \implies \sum_{i=1}^m d^{-l_c(x_i)} \leq 1. \quad \blacksquare$$

8.2. Esercizi

Esempio 8.2.1: Il signor Giovanni Rossi ha

$$X = \{x_1, x_2, x_3, x_4, x_5\} \quad | \quad D = \{000, 001, 01, 110, 111\}.$$

Giovanni ha costruito il codice migliore usando una certa distribuzione P_i ma il bro non si ricorda quale ha utilizzato tra:

- $P_1 = \{0.2, 0.2, 0.2, 0.2, 0.2\}$;
- $P_2 = \{0.4, 0.2, 0.2, 0.1, 0.1\}$;
- $P_3 = \{0.1, 0.1, 0.2, 0.4, 0.2\}$.

Quale di queste probabilità è stata utilizzata?

Calcoliamo i valori attesi:

$$\mathbb{E}[l_c | P_1] = \frac{3}{5} + \frac{3}{5} + \frac{2}{5} + \frac{3}{5} + \frac{3}{5} = \frac{14}{5}$$

$$\mathbb{E}[l_c | P_2] = \frac{12}{10} + \frac{6}{10} + \frac{4}{10} + \frac{3}{10} + \frac{3}{10} = \frac{14}{5}$$

$$\mathbb{E}[l_c | P_3] = \frac{3}{10} + \frac{3}{10} + \frac{4}{10} + \frac{12}{10} + \frac{6}{10} = \frac{14}{5}$$

Sono tutte uguali. Soluzione alternativa: con Huffman dovevamo vedere se le lunghezze che uscivano erano uguali a quelle date.

Esempio 8.2.2: Date

$$X = \{a_1, \dots, a_8, a_9, \dots, a_{12}\} \quad | \quad P = \left\{ \underbrace{0.1, \dots, 0.1}_{1, \dots, 8}, \underbrace{0.05, \dots, 0.05}_{9, \dots, 12} \right\}.$$

Scrivere un CI con $d = 5$.

Applicando Huffman si ottiene:

- $a_1 = 2$;
- $a_2 = 3$;
- $a_3 = 00$;
- $a_4 = 01$;

- $a_5 = 02$;
- $a_6 = 03$;
- $a_7 = 04$;
- $a_8 = 10$;
- $a_9 = 11$;
- $a_{10} = 12$;
- $a_{11} = 13$;
- $a_{12} = 14$.

Posso fare meglio perché non ho un numero giusto di simboli, 5 foglie non vengono coperte. Soluzione alternativa: creare l'albero di codifica cercando di coprire tutte le foglie.

Riguardo l'ultimo esercizio, come faccio se devo usare per forza Huffman? Il problema dell'ultimo esercizio è nel numero di simboli, che non sono nel numero corretto. Quale è il numero corretto? Mo ci arrivo, calma.

Supponiamo di partire da m simboli, ad ogni iterazione rimuoviamo $d - 1$ simboli:

$$m \rightarrow m - (d - 1) \rightarrow m - 2(d - 1) \rightarrow \dots \rightarrow m - t(d - 1).$$

Chiamiamo $\blacksquare = m - t(d - 1)$. Siamo arrivati ad avere \blacksquare elementi, con $\blacksquare \leq d$.

Noi vorremmo avere esattamente d elementi per avere un albero ben bilanciato e senza perdita di rami, quindi aggiungiamo a \blacksquare un numero k di **simboli dummy** tali per cui $\blacksquare + k = d$. I simboli dummy sono dei simboli particolari che hanno probabilità nulla e che usiamo solo come riempimento. Supponiamo di eseguire ancora un passo dell'algoritmo, quindi da $\blacksquare + k$ andiamo a togliere $d - 1$ elementi, lasciando la sorgente con un solo elemento.

Cosa abbiamo ottenuto? Ricordando che $\blacksquare = m - t(d - 1)$, abbiamo fatto vedere che:

$$\begin{aligned}\blacksquare + k - (d - 1) &= 1 \\ m - t(d - 1) + k - (d - 1) &= 1 \\ m + k - (t + 1)(d - 1) &= 1.\end{aligned}$$

In poche parole, il numero m di simboli sorgente, aggiunto al numero k di simboli «fantoccio», è congruo ad 1 modulo $(d - 1)$, ovvero

$$m + k \equiv 1 \pmod{d - 1}.$$

Esempio 8.2.3: Rifare l'esercizio precedente sapendo che possiamo utilizzare i simboli dummy.

Applicando Huffman si ottiene:

- $a_1 = 2$;
- $a_2 = 3$;
- $a_3 = 4$;
- $a_4 = 00$;
- $a_5 = 01$;
- $a_6 = 02$;
- $a_7 = 03$;

- $a_8 = 04$;
- $a_9 = 10$;
- $a_{10} = 11$;
- $a_{11} = 12$;
- $a_{12} = 13$;
- $a_{13} = 14$ (*questo è dummy*).

Ora l'albero di codifica non ha nessuna foglia scoperta.

9. Lezione 09 [25/10]

9.1. Parenti dell'entropia

L'**entropia congiunta** indica la quantità di informazione media che serve per comunicare due messaggi a partire dai simboli di una stessa sorgente. Data la sorgente $\langle X, p \rangle$ e date le variabili aleatorie \mathbb{X} e \mathbb{Y} , essa è definita come

$$H(\mathbb{X}, \mathbb{Y}) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(x, y)} \right).$$

L'**entropia condizionata** indica invece la quantità di informazione media che serve per comunicare un messaggio estratto da \mathbb{Y} sapendo che ho già mandato un messaggio estratto da \mathbb{X} dalla stessa sorgente. Data la sorgente $\langle X, p \rangle$ e date le variabili aleatorie \mathbb{X} e \mathbb{Y} , essa è definita come

$$\begin{aligned} H(\mathbb{Y} | \mathbb{X}) &= \sum_{x \in X} p(x) H(\mathbb{Y} | \mathbb{X} = x) = \\ &= \sum_{x \in X} p(x) \left(\sum_{y \in Y} p(y | x) \log \left(\frac{1}{p(y | x)} \right) \right) = \\ &= \sum_{x \in X} \sum_{y \in Y} p(x) p(y | x) \log \left(\frac{1}{p(y | x)} \right). \end{aligned}$$

Sapendo che

$$p(y | x) = \frac{p(x, y)}{p(x)}$$

e che la *probabilità marginale* (servirà dopo) è

$$p(x) = \sum_{y \in Y} p(x, y)$$

allora

$$H(\mathbb{Y} | \mathbb{X}) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(y | x)} \right).$$

Se $\mathbb{X} = f(\mathbb{Y})$ allora $H(\mathbb{X} | \mathbb{Y}) = 0$ perché la variabile \mathbb{X} la posso ricavare facilmente dalla \mathbb{Y} che è già stata spedita.

Esempio 9.1.1: Sia \mathbb{X} una variabile e sia $\mathbb{Y} = \mathbb{X} + 2$ un'altra variabile casuale. Quanto vale $H(\mathbb{Y} | \mathbb{X})$?

Data la dipendenza di \mathbb{X} da \mathbb{Y} , allora $H(\mathbb{Y} | \mathbb{X}) = 0$.

Esempio 9.1.2: Sia \mathbb{X} una variabile e sia $\mathbb{Y} = \mathbb{X}^2$ un'altra variabile casuale. Entrambe pescano dall'insieme $\{-1, 0, 1\}$. Quanto vale $H(\mathbb{Y} | \mathbb{X})$?

Data la dipendenza di \mathbb{X} da \mathbb{Y} , allora $H(\mathbb{Y} | \mathbb{X}) = 0$.

Quanto vale invece $H(\mathbb{X} \mid \mathbb{Y})$?

Sicuramente $H(\mathbb{X} \mid \mathbb{Y}) > 0$ perché il valore 1 estratto da \mathbb{Y} può essere ottenuto sia da 1 che da -1 .

Teorema 9.1.1 *Chain rule per l'entropia:*

$$H(\mathbb{X}, \mathbb{Y}) = H(\mathbb{X}) + H(\mathbb{Y} \mid \mathbb{X}) = H(\mathbb{Y}) + H(\mathbb{X} \mid \mathbb{Y}).$$

Dimostrazione 9.1.1: Verifichiamo che

$$\begin{aligned} H(\mathbb{X}, \mathbb{Y}) &= \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(x, y)} \right) = \\ &= - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log(p(x)p(y \mid x)) = \\ &= \underbrace{\sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(x)} \right)}_{\text{congiunta}} + \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(y \mid x)} \right) = \\ &= \sum_{x \in X} p(x) \log \left(\frac{1}{p(x)} \right) + H(\mathbb{Y} \mid \mathbb{X}) = H(\mathbb{X}) + H(\mathbb{Y} \mid \mathbb{X}). \end{aligned}$$

Odio il quadratino. ■

Questa regola vale anche negli spazi condizionati, ovvero

$$H(\mathbb{X}, \mathbb{Y} \mid \mathbb{W}) = H(\mathbb{X} \mid \mathbb{W}) + H(\mathbb{Y} \mid \mathbb{X}, \mathbb{W}).$$

Un altro parente, penso il prozio o il cugino, è l'**informazione mutua**: essa ci indica la quantità di informazione media che rilascia \mathbb{Y} rispetto a \mathbb{X} . Si calcola con

$$I(\mathbb{X}, \mathbb{Y}) = \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{p(x, y)}{p(x)p(y)} \right).$$

È molto simile all'entropia relativa: con questa condivide la proprietà di essere ≥ 0 , infatti non possiamo togliere informazioni da \mathbb{X} avendo anche \mathbb{Y} . Inoltre, a differenza dell'entropia relativa, l'informazione mutua è simmetrica.

Lemma 9.1.1:

$$I(\mathbb{X}, \mathbb{Y}) = H(\mathbb{X}) - H(\mathbb{X} \mid \mathbb{Y}).$$

Dimostrazione 9.1.2: Valutiamo

$$\begin{aligned}
 I(\mathbb{X}, \mathbb{Y}) &= \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{p(x, y)}{p(x)p(y)} \right) = \\
 &= \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{p(y)p(x | y)}{p(x)p(y)} \right) = \\
 &= \underbrace{\sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(x)} \right)}_{\text{marginale}} - \sum_{x \in X} \sum_{y \in Y} p(x, y) \log \left(\frac{1}{p(x | y)} \right) = \\
 &= \sum_{x \in X} p(x) \log \left(\frac{1}{p(x)} \right) - H(\mathbb{X} | \mathbb{Y}) = H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y}).
 \end{aligned}$$

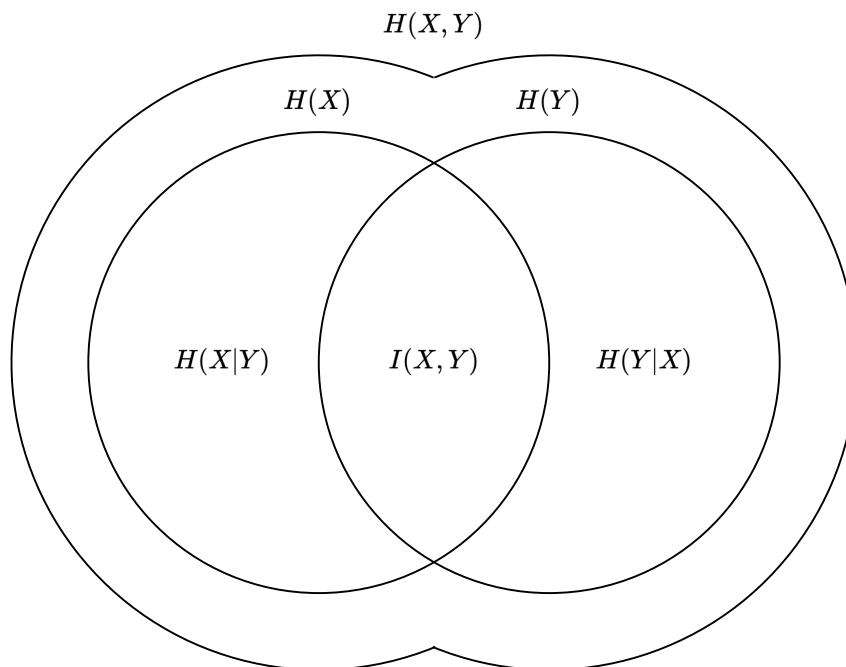
Quadratino di merda. ■

Se \mathbb{X} e \mathbb{Y} sono indipendenti allora $I(\mathbb{X}, \mathbb{Y}) = 0$ perché \mathbb{Y} non rilascia informazioni su \mathbb{X} . Se invece le due variabili sono dipendenti allora $I(\mathbb{X}, \mathbb{Y}) = H(\mathbb{X})$ perché \mathbb{X} è ricavabile totalmente da \mathbb{Y} .

Un modo forse più comodo di scrivere l'informazione mutua è

$$I(\mathbb{X}, \mathbb{Y}) = H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y}) = H(\mathbb{X}) - (H(\mathbb{Y}) + H(\mathbb{X}, \mathbb{Y})) = H(\mathbb{X}) + H(\mathbb{Y}) - H(\mathbb{X}, \mathbb{Y}),$$

che è molto simile alla probabilità dell'unione di due eventi.



Teorema 9.1.2 Data Processing Inequality: Siano $\mathbb{X}, \mathbb{Y}, \mathbb{W}$ variabili aleatorie con codominio finito tali che $p(x, y, w)$ soddisfa $p(x, w | y) = p(x | y)p(w | y) \quad \forall x, y, w$ (ovvero x, w indipendenti dalla y condizionante) allora

$$I(\mathbb{X}, \mathbb{Y}) \geq I(\mathbb{X}, \mathbb{W}).$$

Dimostrazione 9.1.3: Valutiamo

$$\begin{aligned} I(\mathbb{X}, (\mathbb{Y}, \mathbb{W})) &= \sum_{x, y, w \in X} p(x, y, w) \log \left(\frac{p(x, y, w)}{p(x)p(y, z)} \right) = \\ &= \sum_{x, y, w \in X} p(x, y, w) \log \left(\frac{p(y | x, w)p(x, w)}{p(x)p(y | w)p(w)} \right) = \\ &= \sum_{x, w \in X} \underbrace{\sum_{y \in X} p(x, y, w)}_{\text{marginale}} \log \left(\frac{p(x, w)}{p(x)p(w)} \right) + \sum_{x, y, w \in X} p(x, y, w) \log \left(\frac{p(y | x, w)}{p(y | w)} \right) = \\ &= \sum_{x, w \in X} p(x, w) \log \left(\frac{p(x, w)}{p(x)p(w)} \right) + \sum_{x, y, w \in X} p(x, y, w) \log \left(\frac{p(x, y | w)}{p(y | w)p(x | w)} \right) = \\ &= I(\mathbb{X}, \mathbb{W}) + I(\mathbb{X}, \mathbb{Y} | \mathbb{W}). \end{aligned}$$

Questo vale per ogni terna $\mathbb{X}, \mathbb{Y}, \mathbb{W}$ quindi vale anche $I(\mathbb{X}, (\mathbb{Y}, \mathbb{W})) = I(\mathbb{X}, \mathbb{Y}) + I(\mathbb{X}, \mathbb{W} | \mathbb{Y})$ quindi

$$I(\mathbb{X}, \mathbb{Y}) + \underbrace{I(\mathbb{X}, \mathbb{W} | \mathbb{Y}, 0)}_{\geq 0} = I(\mathbb{X}, \mathbb{W}) + \underbrace{I(\mathbb{X}, \mathbb{Y} | \mathbb{W})}_{\geq 0}$$

ma allora

$$I(\mathbb{X}, \mathbb{Y}) \geq I(\mathbb{X}, \mathbb{W}).$$

■

A cosa serve questo teorema? Abbiamo

$$\mathbb{X} \xrightarrow{\text{rumore}} \mathbb{Y} \xrightarrow{\text{algoritmo}} \mathbb{W},$$

ovvero sappiamo che \mathbb{W} è indipendente da \mathbb{X} e quindi il \mathbb{W} processato non può essere più informativo di \mathbb{Y} su \mathbb{X} .

10. Lezione 10 [29/10]

APPELLI INVERNALI: 24-01 / 07-02 / 21-02

10.1. Teoria

Prendiamo il teorema precedente e vediamo cosa succede se \mathbb{Z} non è indipendente da \mathbb{X} dato \mathbb{Y} .

Esempio 10.1.1: Siano \mathbb{X}, \mathbb{Y} variabili casuali indipendenti e sia $\mathbb{Z} = \mathbb{X} + \mathbb{Y}$. Se $\mathbb{X}, \mathbb{Y} = \{0, 1\}$, con $p_{\mathbb{X}} = p_{\mathbb{Y}} = 0.5$, allora $\mathbb{Z} = \{0, 1, 2\}$. Vediamo quanto valgono

$$I(\mathbb{X}, \mathbb{Y}) = H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y}) = H(\mathbb{X}) - H(\mathbb{X}) = 0$$

e

$$\begin{aligned} I(\mathbb{X}, \mathbb{Y} | \mathbb{Z}) &= H(\mathbb{X} | \mathbb{Z}) - H(\mathbb{X} | \mathbb{Y}, \mathbb{Z}) = \\ &= p(\mathbb{Z} = 0)H(\mathbb{X} | \mathbb{Z} = 0) + p(\mathbb{Z} = 1)H(\mathbb{X} | \mathbb{Z} = 1) + \\ &\quad + p(\mathbb{Z} = 2)H(\mathbb{X} | \mathbb{Z} = 2) + 0 = \\ &= 0 + p(\mathbb{Z} = 1)H(\mathbb{X} | \mathbb{Z} = 1) + 0 = \\ &= p(\mathbb{Z} = 1) = \frac{1}{2} = \varepsilon \end{aligned}$$

perché gli eventi sono $0 + 0/0 + 1/1 + 0/1 + 1 + 1$ e a noi ne vanno bene 2 su 4. Rimane quindi che

$$I(\mathbb{X}, \mathbb{Y}) \geq I(\mathbb{X}, \mathbb{Z}) + \varepsilon \implies 0 \geq I(\mathbb{X}, \mathbb{Z}) + \frac{1}{2}.$$

Quindi questa cosa non funziona, visto che ho dipendenza tra \mathbb{X}, \mathbb{Y} e \mathbb{Z} e quindi non va più la disuguaglianza, ha cambiato verso.

10.2. Esercizi

Esempio 10.2.1 TDE 01/04/2003: Abbiamo un sistema sorgente-canale-ricevente tale che

$$\langle S, p \rangle \quad | \quad S = \{s_1, s_2, s_3, s_4\} \quad | \quad P = \{0.2, 0.3, 0.1, 0.4\}.$$

Ci viene data la matrice stocastica del canale, ovvero cosa succede al dato quando viene immesso sul canale. Essa è definita da

$$M = \begin{bmatrix} 0.2 & 0.2 & 0.3 & 0.2 & 0.1 \\ 0.2 & 0.5 & 0.1 & 0.1 & 0.1 \\ 0.6 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.3 & 0.1 & 0.1 & 0.1 & 0.4 \end{bmatrix}.$$

Sulle righe ho indice $i = 1, \dots, 4$ e sulle colonne ho indice $j = 1, \dots, 5$.

DOMANDA 1: calcolare $H(R | S)$.

Usiamo le formule dell'entropia condizionata, ovvero

$$H(R | S) = \sum_{i=1}^4 p(s_i) H(R | S = s_i)$$

e

$$H(R | S = s_i) = \sum_{j=1}^5 p(r_j | s_i) \log \left(\frac{1}{p(r_j | s_i)} \right).$$

Andiamo a calcolare le quantità $H(R | S = s_i)$:

- $s_1 : 0.2 \log(5) + 0.2 \log(5) + 0.3 \log(\frac{10}{3}) + 0.2 \log(5) + 0.1 \log(10) = 2.247$;
- $s_2 : 0.2 \log(5) + 0.5 \log(2) + 0.1 \log(10) + 0.1 \log(10) + 0.1 \log(10) = 1.961$;
- $s_3 : 0.6 \log(\frac{5}{3}) + 0.1 \log(10) + 0.1 \log(10) + 0.1 \log(10) + 0.1 \log(10) = 1.771$;
- $s_4 : 0.3 \log(\frac{10}{3}) + 0.1 \log(10) + 0.1 \log(10) + 0.1 \log(10) + 0.4 \log(\frac{5}{2}) = 2.046$.

Moltiplichiamo questi valori per le probabilità e otteniamo l'entropia richiesta:

$$H(R | S) = 0.2 \cdot 2.247 + 0.3 \cdot 1.961 + 0.1 \cdot 1.771 + 0.4 \cdot 2.046 = 2.033 \text{ bit.}$$

DOMANDA 2: calcolare $H(S | R)$.

Non so come fare.

11. Lezione 11 [08/11]

11.1. Disuguaglianza di Fano

Vediamo la disuguaglianza di Fano e cerchiamo di capire a cosa serve.

La sorgente manda un messaggio \mathbb{X} e il ricevente riceve \mathbb{Y} . Se il canale non ha rumore nessun problema, ma se il canale distorce alcuni bit di informazione potrei ricevere un messaggio che è diverso da quello originale.

Usiamo quindi una funzione $g(\mathbb{Y})$ che cerca di risalire al simbolo \mathbb{X} che è stato spedito. La funzione potrebbe azzeccare il vero simbolo che è stato spedito ma potrebbe anche darmi un'informazione sbagliata, ad esempio quando sono distorti tanti bit.

Chiamiamo p_e la probabilità che la funzione g sbaglia a predire il simbolo \mathbb{X} spedito, ovvero

$$p_e = P(g(\mathbb{Y}) \neq \mathbb{X}).$$

Ovviamente, più è alto il livello di distorsione del canale, più la funzione g sbaglia.

Non è detto che i due insiemi di simboli X, Y abbiano lo stesso numero di elementi. Ad esempio, se $X = \{x_1, \dots, x_n\}$ e $Y = \{y_1, \dots, y_m\}$ basta scegliere $n \neq m$ per avere due insiemi differenti.

Teorema 11.1.1 *Disuguaglianza di Fano*: Siano \mathbb{X}, \mathbb{Y} due variabili casuali con valori in X, Y insiemi finiti. Sia $g : Y \rightarrow X$ una funzione che mappa valori di Y in valori di X . Sia p_e la probabilità di errore quando uso $g(\mathbb{Y})$ per predire \mathbb{X} , ovvero $p_e = P(g(\mathbb{Y}) \neq \mathbb{X})$. Allora

$$p_e \geq \frac{H(\mathbb{X} | \mathbb{Y}) - 1}{\log_2(|X|)}.$$

Dimostrazione 11.1.1: Introduciamo una variabile bernoulliana \mathbb{E} che indica il comportamento di g : quest'ultima o trova la \mathbb{X} giusta, o trova una \mathbb{X} sbagliata. In poche parole:

$$\mathbb{E} = \begin{cases} 1 & \text{se } g(\mathbb{Y}) \neq \mathbb{X} \\ 0 & \text{se } g(\mathbb{Y}) = \mathbb{X} \end{cases}$$

Per la chain rule dell'entropia sappiamo che:

$$\begin{aligned} H(\mathbb{E}, \mathbb{X} | \mathbb{Y}) &\stackrel{\text{CR}}{=} H(\mathbb{E} | \mathbb{Y}) + H(\mathbb{X} | \mathbb{E}, \mathbb{Y}) = \\ &\stackrel{\text{CR}}{=} H(\mathbb{X} | \mathbb{Y}) + H(\mathbb{E} | \mathbb{X}, \mathbb{Y}). \end{aligned}$$

Consideriamo solo i due membri di destra, ovvero:

$$H(\mathbb{E} | \mathbb{Y}) + H(\mathbb{X} | \mathbb{E}, \mathbb{Y}) = H(\mathbb{X} | \mathbb{Y}) + H(\mathbb{E} | \mathbb{X}, \mathbb{Y})$$

Osserviamo che:

- $H(\mathbb{E} | \mathbb{Y}) \leq H(\mathbb{E})$ perché il condizionamento non aumenta l'entropia, lo vediamo dai diagrammi di Venn della lezione09 oppure osservando che il condizionamento può rilasciare o meno delle informazioni, ma sicuramente non ne toglie;
- $H(\mathbb{E} | \mathbb{X}, \mathbb{Y}) = 0$ perché conoscendo \mathbb{X} e \mathbb{Y} posso calcolare $g(\mathbb{Y})$ e vedere se è uguale a \mathbb{X} , e quindi sapere subito il valore di \mathbb{E} .

Siamo quindi nella seguente situazione:

$$H(\mathbb{E}) + H(\mathbb{X} \mid \mathbb{E}, \mathbb{Y}) \underset{\text{OSS1}}{\geq} H(\mathbb{E} \mid \mathbb{Y}) + H(\mathbb{X} \mid \mathbb{E}, \mathbb{Y}) = H(\mathbb{X} \mid \mathbb{Y}) + \underbrace{H(\mathbb{E} \mid \mathbb{X}, \mathbb{Y})}_{=0 \text{ per OSS2}}.$$

Consideriamo solo i due membri esterni, e quindi:

$$H(\mathbb{E}) + H(\mathbb{X} \mid \mathbb{E}, \mathbb{Y}) \geq H(\mathbb{X} \mid \mathbb{Y}).$$

Valutiamo la quantità

$$\begin{aligned} H(\mathbb{X} \mid \mathbb{E}, \mathbb{Y}) &= \sum_{e \in \mathbb{E}} p_e H(\mathbb{X} \mid \mathbb{E} = e, \mathbb{Y}) = \\ &= p(\mathbb{E} = 0) H(\mathbb{X} \mid \mathbb{E} = 0, \mathbb{Y}) + p(\mathbb{E} = 1) H(\mathbb{X} \mid \mathbb{E} = 1, \mathbb{Y}) = \\ &= (1 - p_e) \underbrace{H(\mathbb{X} \mid \mathbb{E} = 0, \mathbb{Y})}_{=0 \text{ perché ricavo da } g} + p_e H(\mathbb{X} \mid \mathbb{E} = 1, \mathbb{Y}) = \\ &= p_e H(\mathbb{X} \mid \mathbb{E} = 1, \mathbb{Y}). \end{aligned}$$

L'entropia è maggiorata dalla quantità $\log_2(|X|)$, ma se $\mathbb{E} = 1$ allora devo pescare da X tutti i valori tranne uno, quello corretto, perché appunto sto sbagliando a predire \mathbb{X} . Ma allora

$$p_e H(\mathbb{X} \mid \mathbb{E} = 0, \mathbb{Y}) \leq \log_2(|X| - 1).$$

Osserviamo infine che $H(\mathbb{E}) \leq 1$ perché \mathbb{E} è una variabile bernoulliana.

Viste queste ultime due valutazioni abbiamo ottenuto:

$$1 + p_e \log_2(|X| - 1) \geq H(\mathbb{X} \mid \mathbb{Y}) \implies p_e \geq \frac{H(\mathbb{X} \mid \mathbb{Y}) - 1}{\log_2(|X|)}.$$

■

Abbiamo mostrato che l'entropia è anche un **lower bound** per la probabilità di errore, oltre che per il valore atteso delle lunghezze del codice. Questo lower bound però è modulato sulla grandezza della sorgente, ma è comunque un lower bound.

11.2. Canale

Per ora ci siamo concentrati sulla prima parte del problema che aveva formulato Shannon, ovvero cercare di comprimere al massimo il messaggio da spedire sul canale. Ora ci concentriamo sul secondo problema, ovvero cercare di codificare il canale stesso per aggiungere ridondanza

Un **canale** è definito dalla tupla

$$\langle X, Y, p(y \mid x) \rangle$$

formata da:

- X insieme dei simboli della sorgente;
- Y insieme dei simboli del ricevente;
- $p(y \mid x)$ probabilità di ottenere $y \in Y$ sapendo che è stato ricevuto $x \in X$. Questa probabilità è la **matrice di canale**, che ci consente di definire il comportamento del canale.

Noi useremo un **canale discreto senza memoria**:

- *discreto*: lavoriamo con i simboli della sorgente e del ricevente che sono un numero finito;
- *senza memoria*: sia $x^n = (x_1 \cdots x_n)$ un messaggio di n simboli di X e sia $y^n = (y_1 \cdots y_n)$ messaggio di n simboli di Y . Osserviamo che

$$p(a, b, c) = p(a | bc)p(b | c)p(c).$$

Ma allora

$$p(y^n | x^n) = p(y_n | y^{n-1}x^n)p(y_{n-1} | y^{n-2}x^n) \cdots p(y_3 | y^2x^n)p(y_2 | y^1x^n)p(y_1 | x^n).$$

Il canale è senza memoria, ovvero il canale si disinteressa di quello che è stato spedito prima del carattere appena ricevuto e di quello che verrà spedito dopo il carattere appena ricevuto. Quindi:

$$p(y^n | x^n) = p(y_n | x_n)p(y_{n-1} | x_{n-1}) \cdots p(y_1 | x_1) = \prod_{i=1}^n p(y_i | x_i).$$

Questo è il prodotto di tutti (*circa*) gli elementi della matrice di canale.

Dobbiamo definire un'ultima cosa: la **capacità del canale**. Essa rappresenta la massima informazione che possiamo trasmettere quando accediamo al canale. Se immaginiamo un fiume, la capacità di quest'ultimo è la sua portata, ovvero quanta acqua passa nell'unità di tempo. In un canale, l'informazione è l'acqua e l'accesso al canale è l'unità di tempo.

La capacità di un canale $\langle X, Y, p(y | x) \rangle$ è definita come

$$C = \max_{p(x)} I(\mathbb{X}, \mathbb{Y}),$$

ovvero su tutte le distribuzioni di probabilità di \mathbb{X} prendiamo l'informazione mutua massima.

Noi sappiamo che $I(\mathbb{X}, \mathbb{Y}) \geq 0$, ma anche che

$$\begin{aligned} I(\mathbb{X}, \mathbb{Y}) &= H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y}) \\ &= H(\mathbb{Y}) - H(\mathbb{Y} | \mathbb{X}). \end{aligned}$$

Nella prima relazione $X(\mathbb{X}) \leq \log_2(|X|)$ quindi $H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y}) \leq \log_2(|X|)$. Possiamo fare un discorso analogo per la seconda relazione, ovvero $H(\mathbb{Y}) - H(\mathbb{Y} | \mathbb{X}) \leq \log_2(|Y|)$. Per far valere entrambe le relazioni prendiamo il minimo tra le due quantità. Ma allora

$$0 \leq C \leq \min(\log_2(|X|), \log_2(|Y|)).$$

11.3. Esercizi

Esempio 11.3.1: Dato un canale binario senza rumore, calcolare la capacità del canale.

Possiamo calcolare la capacità in due modi:

- oldani-style (*non so se corretto*): visto che il canale è binario allora $X = Y = \{0,1\}$, quindi

$$0 \leq C \leq \max(\log_2(2), \log_2(2)) = 1;$$

- visconti-style: siano \mathbb{X}, \mathbb{Y} due variabili casuali che pescano dagli insiemi X e Y . Costruiamo la matrice $p(y | x)$ come

X/Y	0	1
0	1	0
1	0	1

Calcoliamo l'informazione mutua con $I(\mathbb{X}, \mathbb{Y}) = H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y}) = H(\mathbb{X})$. La capacità C è il massimo di tutte le informazioni mutue, quindi il massimo delle entropie di \mathbb{X} , ma \mathbb{X} è una bernoulliana quindi ha valore massimo 1, e quindi anche la capacità è 1.

Esempio 11.3.2: Abbiamo un sistema sorgente-canale-ricevente tale che

$$\langle S, p \rangle \quad | \quad S = \{s_1, s_2, s_3, s_4\} \quad | \quad P = \{0.2, 0.3, 0.1, 0.4\}.$$

Ci viene data la matrice stocastica del canale, ovvero cosa succede al dato quando viene immesso sul canale. Essa è definita da

$$M = \begin{pmatrix} 0.2 & 0.2 & 0.3 & 0.2 & 0.1 \\ 0.2 & 0.5 & 0.1 & 0.1 & 0.1 \\ 0.6 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.3 & 0.1 & 0.1 & 0.1 & 0.4 \end{pmatrix}.$$

Sulle righe ho indice $i = 1, \dots, 4$ e sulle colonne ho indice $j = 1, \dots, 5$.

La prima domanda della scorsa lezione chiedeva di trovare $H(S | R)$, che abbiamo fatto in modo facile con la definizione di entropia condizionata.

Il modo difficile è calcolare la stessa quantità con

$$H(S | R) = H(R) + H(S | R) = H(S) + H(R | S)$$

considerando i due membri di destra. L'unica quantità che conosciamo (*da calcolare in realtà*) è $H(S)$.

Calcoliamo la quantità $H(S | R)$. Noi sappiamo che

$$H(S | R) = \sum_{j=1}^5 p(b_j) H(S | b_j) = \sum_{j=1}^5 p(b_j) \sum_{i=1}^4 p(a_i | b_j) \log \left(\frac{1}{p(a_i | b_j)} \right).$$

Sapendo che $p(a, b) = p(a | b)p(b) = p(b | a)p(a)$, possiamo calcolare

$$p(b_j, a_i) = p(b_j | a_i)p(a_i) = \begin{pmatrix} 0.04 & 0.04 & 0.06 & 0.04 & 0.02 \\ 0.06 & 0.15 & 0.03 & 0.03 & 0.03 \\ 0.06 & 0.01 & 0.01 & 0.01 & 0.01 \\ 0.12 & 0.04 & 0.04 & 0.04 & 0.16 \end{pmatrix}.$$

Calcoliamo la probabilità dei simboli di R , sommando sulle colonne, ovvero

$$p(b \in R) = (0.28 \quad 0.24 \quad 0.14 \quad 0.12 \quad 0.22).$$

Sapendo che $p(a_i | b_j) = \frac{p(a_i, b_j)}{p(b_j)}$, calcoliamo la matrice

$$p(a_i | b_j) = \begin{pmatrix} 0.14 & 0.17 & 0.43 & 0.33 & 0.09 \\ 0.21 & 0.63 & 0.21 & 0.25 & 0.14 \\ 0.21 & 0.04 & 0.07 & 0.08 & 0.05 \\ 0.42 & 0.17 & 0.29 & 0.33 & 0.73 \end{pmatrix}$$

ovvero prendo la matrice fatta prima e divido per le probabilità del ricevente. Facendo tutti i conti (*non ho voglia*) si trova $H(S | R)$.

Ora devo solo trovare $H(R)$ visto che ora abbiamo le probabilità.

12. Lezione 12 [15/11]

Oggi si fanno solo esercizi, vamo caraco.

12.1. Esercizi

Esempio 12.1.1: Dato un canale $\langle X, Y, p(y | x) \rangle$ con:

- X simboli della sorgente con relative probabilità;
- Y simboli del ricevente;
- $p(y | x)$ matrice del canale.

Siano $X = \{0, 1\}$ e $Y = \{0, 1, 2, 3\}$. Definiamo la matrice di canale come

X/Y	0	1	2	3
0	0.5	0.5	0	0
1	0	0	0.5	0.5

Calcolare la capacità del canale.

Calcoliamo $C = \max_{p(x)} H(\mathbb{X}) - H(\mathbb{X} | \mathbb{Y})$, ma la seconda quantità è 0 perché sapendo \mathbb{Y} sappiamo anche la \mathbb{X} . Dobbiamo quindi massimizzare $H(\mathbb{X})$, ma questa è una bernoulliana che ha valore massimo 1, quindi

$$C = 1.$$

Esempio 12.1.2: Dato un canale $\langle X, Y, p(y | x) \rangle$, siano:

- $X = \{0, 1, 2\}$;
- $Y = \{0, 1, 2\}$;

e la matrice di canale

X/Y	0	1	2
0	0.5	0.5	0
1	0	0.5	0.5
2	0.5	0	0.5

Calcolare la capacità del canale.

Calcoliamo $C = \max_{p(x)} H(\mathbb{Y}) - H(\mathbb{Y} | \mathbb{X})$. Valutiamo la seconda quantità come

$$\begin{aligned} H(\mathbb{Y} | \mathbb{X}) &= p(\mathbb{X} = 0)H(\mathbb{Y} | \mathbb{X} = 0) + p(\mathbb{X} = 1)H(\mathbb{Y} | \mathbb{X} = 1) + p(\mathbb{X} = 2)H(\mathbb{Y} | \mathbb{X} = 2) = \\ &= p_0 H(\mathbb{Y} | \mathbb{X} = 0) + p_1 H(\mathbb{Y} | \mathbb{X} = 1) + p_2 H(\mathbb{Y} | \mathbb{X} = 2). \end{aligned}$$

Non conosciamo p_0, p_1, p_2 ma sappiamo che l'entropia viene massimizzata quando le probabilità dei simboli sono distribuite uniformemente, quindi assumiamo $p_0 = p_1 = p_2 = \frac{1}{3}$.

Qui abbiamo due modi di procedere:

- ogni simbolo si comporta come una variabile bernoulliana, visto che si «spezza» in due simboli, e quindi ha entropia 1;
- calcoliamo ogni entropia come

$$H(\mathbb{Y} \mid \mathbb{X} = i) = \sum_{k=0}^2 p(y_k \mid \mathbb{X} = i) \log_2 \left(\frac{1}{p(y_k \mid \mathbb{X} = i)} \right).$$

In entrambi i casi si ottiene il valore 1.

Ma allora

$$H(\mathbb{Y} \mid \mathbb{X}) = p_0 + p_1 + p_2 = 1.$$

Rimane solo $H(\mathbb{Y})$. Scegliamo anche qua qualcosa che massimizzi l'entropia, una distribuzione uniforme, quindi

$$p(\mathbb{Y} = 0) = p(\mathbb{Y} = 1) = p(\mathbb{Y} = 2) = \left(\frac{1}{3} \frac{1}{2}\right) + \left(\frac{1}{3} \frac{1}{2}\right) = \frac{1}{3}.$$

L'entropia diventa quindi

$$H(\mathbb{Y}) = \sum_{i=0}^2 p(y_i) \log_2 \left(\frac{1}{p(y_i)} \right) = 3 \left(\frac{1}{3} \log_2(3) \right) = \log_2(3).$$

La capacità è quindi

$$C = \log_2(3) - 1 \approx 0.585.$$

Esempio 12.1.3: Dato un canale $\langle X, Y, p(y \mid x) \rangle$, siano:

- $X = \{0, 1\}$;
- $Y = \{0, 1\}$;

e la matrice di canale

$$\begin{array}{c|cc} X/Y & 0 & 1 \\ \hline 0 & 1-p & p \\ 1 & p & 1-p \end{array}.$$

Calcolare la capacità del canale.

Calcoliamo $C = \max_{p(x)} H(\mathbb{Y}) - H(\mathbb{Y} \mid \mathbb{X})$. Valutiamo la seconda quantità come

$$H(\mathbb{Y} \mid \mathbb{X}) = \sum_{i=0}^1 p(\mathbb{X} = i) H(\mathbb{Y} \mid \mathbb{X} = i).$$

Calcoliamo $H(\mathbb{Y} \mid \mathbb{X} = i)$ per $i = 0, 1$:

$$\begin{aligned}
H(\mathbb{Y} \mid \mathbb{X} = 0) &= p(\mathbb{Y} = 0 \mid \mathbb{X} = 0) \log_2 \left(\frac{1}{p(\mathbb{Y} = 0 \mid \mathbb{X} = 0)} \right) + \\
&+ p(\mathbb{Y} = 1 \mid \mathbb{X} = 0) \log_2 \left(\frac{1}{p(\mathbb{Y} = 1 \mid \mathbb{X} = 0)} \right) = \\
&= (1-p) \log_2 \left(\frac{1}{1-p} \right) + p \log_2 \left(\frac{1}{p} \right) = H(p) \\
H(\mathbb{Y} \mid \mathbb{X} = 1) &= \text{stessa cosa} = H(p).
\end{aligned}$$

In poche parole, so che $H(\mathbb{Y} \mid \mathbb{X} = i)$ vale come l'entropia di una bernoulliana di parametro p , che però non conosco.

Ma allora

$$H(\mathbb{Y} \mid \mathbb{X}) = p(\mathbb{X} = 0)H(p) + p(\mathbb{X} = 1)H(p) = (p(\mathbb{X} = 0) + p(\mathbb{X} = 1))H(p) = H(p).$$

Il calcolo di $H(\mathbb{Y})$ ricade nuovamente sull'entropia di una variabile distribuita uniformemente. Calcoliamo le $p(\mathbb{Y})$ singolarmente:

$$\begin{aligned}
p(\mathbb{Y} = 0) &= p(\mathbb{X} = 0)p(\mathbb{Y} = 0 \mid \mathbb{X} = 0) + p(\mathbb{X} = 1)p(\mathbb{Y} = 0 \mid \mathbb{X} = 1) = \frac{1}{2}(1-p) + \frac{1}{2}p = \frac{1}{2} \\
p(\mathbb{Y} = 1) &= \text{uguale} = \frac{1}{2}.
\end{aligned}$$

Ma allora

$$H(\mathbb{Y}) = \sum_{i=0}^1 p(y_i) \log_2 \left(\frac{1}{p(y_i)} \right) = \frac{1}{2}1 + \frac{1}{2}1 = 1.$$

La capacità è quindi

$$C = 1 - H(p).$$

Esempio 12.1.4: Dato un canale $\langle X, Y, p(y \mid x) \rangle$, siano:

- $X = \{0, 1\}$;
- $Y = \{0, 1\}$;

e la matrice di canale

$$\begin{array}{c|cc}
X/Y & 0 & 1 \\
\hline
0 & 1-\alpha & \alpha \\
1 & \alpha & 1-\alpha
\end{array}$$

Calcolare la capacità del canale.

Calcoliamo $C = \max_{p(x)} H(\mathbb{Y}) - H(\mathbb{Y} \mid \mathbb{X})$. Valutiamo la seconda quantità, notando che \mathbb{X} è una variabile bernoulliana di parametro α , quindi come l'esercizio precedente $H(\mathbb{Y} \mid \mathbb{X}) = H(\alpha)$.

Introduciamo una variabile \mathbb{W} tale che

$$\mathbb{W} = \begin{cases} 1 & \text{se } \mathbb{Y} = e \\ 0 & \text{altrimenti} \end{cases}.$$

Sappiamo, per la Chain Rule, che $H(\mathbb{Y}, \mathbb{W}) = H(\mathbb{Y}) + H(\mathbb{W} \mid \mathbb{Y}) = H(\mathbb{W}) + H(\mathbb{Y} \mid \mathbb{W})$. Consideriamo l'uguaglianza di destra e isoliamo $H(\mathbb{Y})$ ottenendo

$$H(\mathbb{Y}) = H(\mathbb{W}) + H(\mathbb{Y} \mid \mathbb{W}) - H(\mathbb{W} \mid \mathbb{Y}).$$

Notiamo che $H(\mathbb{W} \mid \mathbb{Y}) = 0$ perché se conosco \mathbb{Y} posso dire con precisione il valore di \mathbb{W} .

Calcoliamo $H(\mathbb{W})$ con la formula classica di entropia, calcolando però prima le singole probabilità $p(\mathbb{W} = i)$ come

$$\begin{aligned} p(\mathbb{W} = 1) &= p(\mathbb{X} = 0)p(\mathbb{W} = 1 \mid \mathbb{X} = 0) + p(\mathbb{X} = 1)p(\mathbb{W} = 1 \mid \mathbb{X} = 1) = \\ &= p(\mathbb{X} = 0)\alpha + p(\mathbb{X} = 1)\alpha = \alpha(p(\mathbb{X} = 0) + p(\mathbb{X} = 1)) = \alpha \\ p(\mathbb{W} = 0) &= 1 - p(\mathbb{W} = 1) = 1 - \alpha. \end{aligned}$$

Allora

$$H(\mathbb{W}) = \sum_{i=0}^1 p(w_i) \log_2 \left(\frac{1}{p(w_i)} \right) = (1 - \alpha) \log_2 \left(\frac{1}{1 - \alpha} \right) + \alpha \log_2 \left(\frac{1}{\alpha} \right) = H(\alpha).$$

Calcoliamo infine $H(\mathbb{Y} \mid \mathbb{W})$ come

$$\begin{aligned} H(\mathbb{Y} \mid \mathbb{W}) &= p(\mathbb{W} = 0)H(\mathbb{Y} \mid \mathbb{W} = 0) + p(\mathbb{W} = 1)H(\mathbb{Y} \mid \mathbb{W} = 1) = \\ &= H(\mathbb{X})(1 - \alpha) + 0 \cdot \alpha = H(\mathbb{X})(1 - \alpha). \end{aligned}$$

Questo vale perché:

- se $\mathbb{W} = 0$ vuol dire che non ho errore, ma se non ho errori quello che mando è quello che ricevo, quindi dipende solo dalla sorgente;
- se $\mathbb{W} = 1$ vuol dire che ho avuto errore e che conosco già il valore di \mathbb{Y} .

Possiamo calcolare, FINALMENTE, il valore di $H(\mathbb{Y})$ come

$$H(\mathbb{Y}) = H(\alpha) + H(\mathbb{X})(1 - \alpha) - 0 = H(\alpha) + H(\mathbb{X})(1 - \alpha).$$

La capacità del canale è quindi

$$C = \max_{p(x)} H(\mathbb{Y}) - H(\mathbb{Y} \mid \mathbb{X}) = H(\alpha) + H(\mathbb{X})(1 - \alpha) - H(\alpha) = H(\mathbb{X})(1 - \alpha) = 1 - \alpha$$

perché $H(\mathbb{X})$ è l'entropia di una bernoulliana, che ha valore massimo 1.

13. Lezione 13 [22/11]

Sappiamo che $C = \max_{p(x)} I(\mathbb{X}, \mathbb{Y})$ ma risolverlo non è banale, questo perché non abbiamo un procedimento algoritmico e siamo obbligati a giocare con l'informazione mutua.

Il canale, così come l'abbiamo definito ora, non ci va bene: come nella parte di codifica, definiremo una sorta di estensione del canale per poter poi «spalmare» l'errore quando scegliamo una estensione molto grande.

Definiamo quindi l'**estensione** n -esima del canale la tupla

$$C_n = \langle X^n, Y^n, p(y^n | x^n) \rangle$$

tale che $p(y^n | x^n) = \prod_{i=1}^n p(y_i | x_i)$ per indipendenza.

Un **codice canale** per il canale C_n è di tipo (M, n) se:

- M è il massimo numero di messaggi che voglio codificare/spedire sul canale, ovvero è l'insieme di messaggi $\{1, \dots, M\}$;
- n è il numero di accessi che facciamo al canale;
- x^n è la funzione di codifica tale che $x^n : \{1, \dots, M\} \rightarrow X^n$;
- g è la funzione di decodifica tale che $y^n : Y^n \rightarrow \{1, \dots, M\}$.

Manca ancora qualcosa, ma per sapere cosa vediamo cosa stiamo facendo ora:

$$M \xrightarrow{x^n} X^n \xrightarrow{\text{send}} (\text{canale con rumore}) \xrightarrow{\text{send}} Y^n.$$

Siamo in Y^n , abbiamo a disposizione g funzione di decodifica, che però potrebbe fare errori: questo è dato dal fatto che siamo passati in un canale con rumore. Dobbiamo quindi applicare g e indovinare il risultato di questa decodifica.

Completiamo il nostro codice canale con:

- λ_i è definita come $P(g(y^n) \neq i | \mathbb{X}^n = x^n(i))$, ovvero la probabilità che la funzione g decodifichi y^n in un valore diverso da i , che è il nostro messaggio di partenza che è stato codificato da x^n ;
- $\lambda^{(n)}$ è definita come $\max_{i=1, \dots, n} \lambda_i$, ovvero la massima probabilità di errore; quantità comoda come upper bound alle probabilità di errore;
- $p_e^{(n)}$ è definita come $\frac{1}{n} \sum_{i=1}^n \lambda_i$; quantità comoda invece per gestire delle probabilità medie.

Vale ovviamente $p_e^{(n)} \leq \lambda^{(n)}$.

Definiamo il **tasso di trasmissione** di un codice di tipo (M, n) come la quantità

$$R = \frac{\log_2(M)}{n}.$$

Nel nostro caso specifico, codificando in binario e usando n accessi al canale, il numero di messaggi disponibili è 2^n , ma allora

$$R = \frac{\log_2(2^n)}{n} = 1.$$

Questo caso è quello *utopico*, ovvero quando il canale è senza rumore. Se quest'ultimo è presente il tasso di trasmissione è ovviamente minore.

Nella realtà viene usato il **tasso di trasmissione raggiungibile**. Un tasso di trasmissione R è **raggiungibile** se esiste una sequenza di codici canale di tipo $(2^{nR}, n)$ per $n = 1, 2, \dots$ tale che

$$\lim_{n \rightarrow \infty} \lambda^{(n)} = 0.$$

Avendo $R < 1$ il valore di M del codice canale viene minore del caso utopico. Possiamo vedere R come la quantità che ci indica quanto sfoltire il 2^n teorico per non avere errori. Ovviamente, prenderemo la parte intera della quantità 2^{nR} perché dobbiamo codificare un numero intero di messaggi.

Vediamo due proprietà, una statistica e uno viscontea, che ci dicono la stessa cosa.

Teorema 13.1 *Legge dei grandi numeri*: Per ogni sequenza $\mathbb{X}_1, \dots, \mathbb{X}_n$ di variabili casuali i.i.d con valore atteso μ finito, allora

$$\forall \varepsilon > 0 \quad \lim_{n \rightarrow \infty} P \left(\left| \frac{1}{n} \sum_{i=1}^n \mathbb{X}_i - \mu \right| > \varepsilon \right) = 0.$$

Questa legge afferma che, usando la media campionaria come stimatore per il valore atteso delle variabili casuali \mathbb{X}_i , la probabilità di commettere un errore maggiore ε è nulla se consideriamo un buon numero di variabili casuali.

Una proprietà analoga è quella di **equipartizione asintotica**.

Teorema 13.2 *AEP*: Per ogni sequenza $\mathbb{X}_1, \dots, \mathbb{X}_n$ di variabili casuali i.i.d con entropia $H(\mathbb{X})$ finita, allora

$$\forall \varepsilon > 0 \quad \lim_{n \rightarrow \infty} P \left(\left| \frac{1}{n} \log_2 \left(\frac{1}{p(x_1) \cdot \dots \cdot p(x_n)} \right) - H(\mathbb{X}) \right| > \varepsilon \right) = 0.$$

La proprietà è analoga alla legge dei grandi numeri, solo che si basa sull'entropia. Inoltre, questa proprietà vale per ogni variabile casuale \mathbb{X} sulla quale calcoliamo l'entropia $H(\mathbb{X})$.

Diamo ora una definizione di **insieme tipico**. Vediamo prima una definizione informale.

Peschiamo n oggetti creando una sequenza (x_1, \dots, x_n) . Visto che le peschate sono i.i.d., la probabilità di pescare tale sequenza è il prodotto delle singole probabilità. Se questo prodotto è compreso tra due quantità che ora vedremo allora tale sequenza appartiene all'insieme tipico.

Vediamo ora la definizione formale. L'insieme tipico è l'insieme

$$A_\varepsilon^{(n)} = \left\{ (x_1, \dots, x_n) \in X^n \mid 2^{-n(H(\mathbb{X})+\varepsilon)} \leq \prod_{i=1}^n p(x_i) \leq 2^{-n(H(\mathbb{X})-\varepsilon)} \right\}.$$

Facciamo vedere che l'insieme tipico è, in realtà, l'insieme formato da tutte le sequenze che rispettano la proprietà di equipartizione asintotica. Infatti:

$$\begin{aligned}
2^{n(H(\mathbb{X})-\varepsilon)} &\leq \frac{1}{\prod_{i=1}^n p(x_i)} \leq 2^{n(H(\mathbb{X})+\varepsilon)} \\
n(H(\mathbb{X})-\varepsilon) &\leq \log_2 \left(\frac{1}{\prod_{i=1}^n p(x_i)} \right) \leq n(H(\mathbb{X})+\varepsilon) \\
H(\mathbb{X})-\varepsilon &\leq \frac{1}{n} \log_2 \left(\frac{1}{\prod_{i=1}^n p(x_i)} \right) \leq H(\mathbb{X})+\varepsilon \\
-\varepsilon &\leq \frac{1}{n} \log_2 \left(\frac{1}{\prod_{i=1}^n p(x_i)} \right) - H(\mathbb{X}) \leq \varepsilon \\
\left| \frac{1}{n} \log_2 \left(\frac{1}{\prod_{i=1}^n p(x_i)} \right) - H(\mathbb{X}) \right| &\leq \varepsilon
\end{aligned}$$

Dentro AEP abbiamo in realtà $< \varepsilon$, ma questo non cambia niente. Infatti, possiamo scrivere l'insieme tipico come

$$A_\varepsilon^{(n)} = \left\{ (x_1, \dots, x_n) \in X^n \mid \left| \frac{1}{n} \log_2 \left(\frac{1}{\prod_{i=1}^n p(x_i)} \right) - H(\mathbb{X}) \right| \leq \varepsilon \right\}.$$

Vista questa definizione, possiamo affermare che

$$\lim_{n \rightarrow \infty} P(A_\varepsilon^{(n)}) = 1$$

o che

$$\lim_{n \rightarrow \infty} P(\overline{A}_\varepsilon^{(n)}) = 0.$$

Il significato operativo di quanto detto fin'ora è il seguente: asintoticamente le sequenze che stanno nell'insieme tipico hanno tutte la stessa probabilità (*visto che le estrazioni sono i.i.d.*) ed è uguale al prodotto delle probabilità singole. Ma queste quantità sono limitate dai due bound a meno di ε , ma allora le probabilità sono tutte uguali a $2^{-nH(\mathbb{X})}$ a meno di ε . In poche parole:

$$P(x_1, \dots, x_n) = \prod_{i=1}^n p(x_i) = \begin{cases} 0 & \text{se } (x_1, \dots, x_n) \notin A_\varepsilon^{(n)} \\ 2^{-nH(\mathbb{X})} & \text{altrimenti} \end{cases}.$$

Teorema 13.3: Siano $\mathbb{X}_1, \dots, \mathbb{X}_n$ delle variabili casuali i.i.d. e sia $A_\varepsilon^{(n)}$ l'insieme tipico ad esse associato. Allora:

1. possiamo dare un upper bound al numero di elementi dell'insieme tipico, ovvero

$$\forall n \quad |A_\varepsilon^{(n)}| \leq 2^{n(H(\mathbb{X})+\varepsilon)};$$

2. possiamo dare un lower bound al numero di elementi dell'insieme tipico, ovvero

$$\exists n_0 \in \mathbb{N} \mid \forall n > n_0 \quad |A_\varepsilon^{(n)}| \geq (1 - \varepsilon) 2^{n(H(\mathbb{X})-\varepsilon)}.$$

Vediamo l'insieme tipico in un'altra versione. Supponiamo di trasmettere un messaggio $x^n \in X^n$ in un canale rumoroso, che quindi distorce l'informazione trasmessa. Questo messaggio verrà mappato nei messaggi che stanno nell'insieme tipico se riceviamo un messaggio che non è distante dalla parola

effettiva. Se invece siamo distanti dalla parola effettiva, non andremo a finire nell'insieme tipico. In poche parole, ogni messaggio $x^n \in X^n$ va a definire un insieme tipico nell'insieme Y^n . Tutti questi insiemi formano delle **bolle** dentro Y^n .

Ogni bolla ha una dipendenza da ciò che abbiamo spedito, ovvero dobbiamo considerare degli insiemi che sono grandi

$$|A_\varepsilon^{(n)}| \approx 2^{nH(\mathbb{Y} | \mathbb{X})}.$$

Vogliamo ovviamente che tutti questi insiemi siano disgiunti, così da poter decodificare senza sovrapposizioni di bolle di messaggi diverse. Prendiamo quindi una parte di questi messaggi, ovvero

$$M = \frac{2^{nH(\mathbb{Y})}}{2^{nH(\mathbb{Y} | \mathbb{X})}} = 2^{n(H(\mathbb{Y}) - H(\mathbb{Y} | \mathbb{X}))} = 2^{nI(\mathbb{X}, \mathbb{Y})}.$$

ASSURDO! Possiamo quindi riscrivere il tasso di trasmissione R come

$$R = \frac{\log_2(M)}{n} = \frac{\log_2(2^{nI(\mathbb{X}, \mathbb{Y})})}{n} = \frac{nI(\mathbb{X}, \mathbb{Y})}{n} = I(\mathbb{X}, \mathbb{Y}).$$

Definiamo quindi la nuova versione come l'**insieme congiuntamente tipico**. Esso è l'insieme

$$B_\varepsilon^{(n)} = \left\{ (x^n, y^n) \in X^n \times Y^n \mid \left| \frac{1}{n} \log_2 \left(\frac{1}{p(x_1, y_1) \cdot \dots \cdot p(x_n, y_n)} \right) - H(\mathbb{X}, \mathbb{Y}) \right| < \varepsilon \right\}.$$

Questo insieme gode di due **proprietà**:

- come per l'insieme tipico classico,

$$\lim_{n \rightarrow \infty} P(B_\varepsilon^{(n)}) = 1;$$

- se:

- \mathbb{X}_n ha distribuzione $p(\mathbb{X}_n = x^n) = \prod_i p(x_i)$ con $p(x_i)$ marginale di X e \mathbb{X}_n sequenze tipiche di X ;
- \mathbb{Y}_n ha distribuzione $p(\mathbb{Y}_n = y^n) = \prod_i p(y_i)$ con $p(y_i)$ marginale di Y e \mathbb{Y}_n sequenze tipiche di Y ;

allora

$$\forall n \geq 1 \quad P((\mathbb{X}_n, \mathbb{Y}_n) \in B_\varepsilon^{(n)}) \leq 2^{-n(I(\mathbb{X}, \mathbb{Y}) - 3\varepsilon)}.$$

In poche parole, l'insieme $B_\varepsilon^{(n)}$ contiene tutte le coppie di sequenze (x^n, y^n) che rispettano tre condizioni:

- le sequenze di \mathbb{X}_n sono tipiche rispetto all'entropia $H(\mathbb{X})$;
- le sequenze di \mathbb{Y}_n sono tipiche rispetto all'entropia $H(\mathbb{Y})$;
- le coppie di sequenze (x^n, y^n) sono tipiche rispetto all'entropia $H(\mathbb{X}, \mathbb{Y})$.

Stiamo rispettando sia le distribuzioni marginali sia quella congiunta.

Le due proprietà ci dicono che per n grande quasi tutte le coppie osservabili di sequenze appartengono all'insieme $B_\varepsilon^{(n)}$. Inoltre, la probabilità di uscire da questo insieme è bassa e decresce esponenzialmente al crescere di n .

Lavorando con l'insieme tipico abbiamo una rappresentazione affidabile e precisa del comportamento delle sequenze originali.

Possiamo vedere infine il secondo teorema di Shannon, madonna.

Teorema 13.4 *Secondo teorema di Shannon:* Se $\langle X, Y, p(y | x) \rangle$ è un canale con capacità C , allora

$$\forall R < C \quad \exists K_1, \dots, K_n \text{ di tipo } (2^{nR_n}, n) \mid \lim_{n \rightarrow \infty} R_n = R \wedge \lim_{n \rightarrow \infty} \lambda^{(n)}(K_n) = 0.$$

In altre parole, è possibile scegliere un codice che si avvicina al massimo della capacità del canale tramite R , ovvero avvicinando il codice al massimo valore che R può assumere. Inoltre, così facendo, almeno teoricamente, si “spalma” l’errore sull’informazione trasmessa (*che è però massima*), perciò all’aumentare di n si riduce al minimo l’errore trasmesso. Ricordiamo anche che, all’aumentare di n , le “sfere” dell’insieme tipico tendono a non toccarsi permettendo di decodificare senza sovrapposizioni.

Da rileggere sicuramente.

14. Lezione 14 [03/12]

14.1. Introduzione ai codici

Il **rumore** è quel merdone che distorce le informazioni che passano sul canale. Noi vogliamo usare dei codici che rilevino gli errori e, non sarebbe male, che li correggano anche.

Noi considereremo il **rumore bianco**. Esso ha le seguenti due proprietà:

- il rumore arriva in maniera indipendente sui bit, cioè la probabilità di avere un bit errato non dipende dalle probabilità di trovare errori nelle altre posizioni. Nella realtà la situazione è praticamente opposta: gli errori in una zona mi indicano grande probabilità di averne altri nelle vicinanze;
- la probabilità di avere un bit errato è fissata a p e questa quantità è uguale per tutte le posizioni che stiamo considerando. Nella realtà, ovviamente, è totalmente diverso: la probabilità di errore potrebbe variare nel tempo.

Esempio 14.1.1: Vogliamo trasmettere un messaggio di n bit, se p è la probabilità di avere un errore, quanto vale la probabilità di spedire n bit e non avere nessun errore?

La probabilità di non avere un bit errato è $(1 - p)$, iterato per n posizioni ci dà

$$P(0 \text{ errori}) = (1 - p)^n.$$

Esempio 14.1.2: Vogliamo trasmettere un messaggio di n bit, se p è la probabilità di avere un errore, quanto vale la probabilità di spedire n bit e avere esattamente un errore?

La probabilità di avere il primo bit errato e i restanti bit corretti è $p(1 - p)^{n-1}$, iterato per $\binom{n}{1} = n$ visto che l'errore può capitare in ogni posizione, ci dà

$$P(1 \text{ errore}) = np(1 - p)^{n-1}.$$

Esempio 14.1.3: Vogliamo trasmettere un messaggio di n bit, se p è la probabilità di avere un errore, quanto vale la probabilità di spedire n bit e avere al massimo l errori?

Dobbiamo sommare le probabilità di avere 0 errori, 1 errore, eccetera, fino a l errori, ovvero

$$\begin{aligned} P(\text{max } l \text{ errori}) &= P(0 \text{ errori}) + P(1 \text{ errori}) + \dots + P(l \text{ errori}) = \\ &= \binom{n}{0} p^0 (1 - p)^n + \binom{n}{1} p^1 (1 - p)^{n-1} + \dots + \binom{n}{l} p^l (1 - p)^{n-l} = \\ &= \sum_{i=0}^l \binom{n}{i} p^i (1 - p)^{n-i}. \end{aligned}$$

Esempio 14.1.4: Vogliamo trasmettere un messaggio di n bit, se p è la probabilità di avere un errore, quanto vale la probabilità di spedire n bit e avere un numero pari di errori?

Dobbiamo sommare le probabilità di avere 0 errori, 2 errori, eccetera, fino al massimo numero pari di errori, ovvero

$$\begin{aligned} P(2k \text{ errori}) &= P(0 \text{ errori}) + P(2 \text{ errori}) + \dots + P(2k \text{ errori}) = \\ &= \binom{n}{0} p^0 (1-p)^n + \binom{n}{2} p^2 (1-p)^{n-2} + \dots + \binom{n}{2k} p^{2k} (1-p)^{n-2k} = \\ &= \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{2i} p^{2i} (1-p)^{n-2i}. \end{aligned}$$

14.1.1. Bit di parità

Vediamo il buon vecchio **bit di parità**. Questo bit si calcola sommando tutti i bit del numero e poi mod 2. In poche parole, date $n - 1$ cifre, il bit di parità lo calcoliamo come

$$x_n = \sum_{i=1}^{n-1} x_i \bmod 2.$$

Come facciamo a capire se abbiamo un problema sul canale e dobbiamo rispedito il messaggio? Prendiamo i dati ricevuti, calcoliamo il bit di parità e vediamo se otteniamo 0.

Questo codice viene usato per rilevare l'errore, ma non per correggerlo. Inoltre, se l'errore si spalma su un numero pari di bit, il bit di parità non riesce nemmeno a rilevare l'errore.

Esempio 14.1.1.1: Vogliamo trasmettere un messaggio di $n - 1$ bit, se p è la probabilità di avere un errore, quanto vale la probabilità di spedire $n - 1$ bit più un bit di parità e non riuscire a riconoscere gli errori?

Dobbiamo sommare le probabilità di avere 2 errori, 4 errori, eccetera, fino al massimo numero pari di errori, ovvero

$$\begin{aligned} P(\text{non riconosce}) &= P(2 \text{ errori}) + \dots + P(2k \text{ errori}) = \\ &= \binom{n}{2} p^2 (1-p)^{n-2} + \dots + \binom{n}{2k} p^{2k} (1-p)^{n-2k} = \\ &= \sum_{i=1}^{\lfloor \frac{n}{2} \rfloor} \binom{n}{2i} p^{2i} (1-p)^{n-2i}. \end{aligned}$$

14.1.2. Codice ASCII

Vediamo il **codice ASCII**. Eh si fratello, il codice ASCII è un codice.

Come funziona questo codice? Esso utilizza 7 bit per codificare un carattere e un bit di parità, posto in testa della codifica,

14.1.3. Burst di errori

Vediamo un **burst** di errori, ovvero una zona di errori definita da un inizio e una fine. Faremo il controllo usando il bit di parità.

Esempio 14.1.3.1: Prendiamo il messaggio «Hello NCTU». Aggiungiamo un **carattere di parità**, non più un bit, che mettiamo alla fine. Questo carattere fa check di correttezza. Divido il messaggio in caratteri, scrivo in binario e poi calcolo i singoli bit di parità facendo il controllo di parità sulla colonna.

Otteniamo

$$\begin{pmatrix} H & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ e & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ l & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ l & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ o & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ N & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ C & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ T & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ U & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \blacksquare & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

La codifica dell'ultima lettera è n .

Se so che l'errore inizia nelle ultime due colonne della prima riga e finisce nella prime colonne della seconda riga, potrei riuscire a sistemare. Il problema arriva se il burst copre più righe, perché più errori potrebbero annullarsi e quindi non venire rilevati.

14.1.4. Somma pesata

Sto spedendo un messaggio di n caratteri. I **codici a somma pesata** richiedono di calcolare la **somma** e la **somma della somma**.

Esempio 14.1.4.1: Vogliamo spedire il messaggio $wxyz$. Calcoliamo somma e somma della somma.

La **somma** si calcola come valore della cifra corrente più somma precedente. La **somma della somma** si calcola come somma corrente più somma della somma precedente.

Otteniamo quindi

$$\begin{pmatrix} C & \text{somma} & \text{somma della somma} \\ w & w & w \\ x & w + x & 2w + x \\ y & w + x + y & 3w + 2x + y \\ z & w + x + y + z & 4w + 3x + 2y + z \end{pmatrix}$$

Il **carattere di controllo** aggiunto alla fine deve far sì che la sua somma di somma sia congrua a 0 modulo numero dei caratteri.

Esempio 14.1.4.2: Supponiamo di avere 37 lettere che codifico con:

- $[0, 9]$ per i numeri da 0 a 9;
- $[10, 35]$ per le lettere maiuscole;
- 36 per lo spazio.

Il messaggio da spedire sul canale è «3b 8». La tabella è:

C	valore	somma	somma della somma
3	3	3	3
B	11	14	17
spazio	36	50	67
8	8	58	125
check	?	$58 + ?$	$183 + ?$

Il numero da aggiungere è 2, codificato con 2, quindi la stringa che spediamo sul canale è «3b 82» con l'ultimo carattere che è di controllo.

Cosa succede se mi arriva la stringa «3b82»? Calcolo:

C	valore	somma	somma della somma
3	3	3	3
B	11	14	17
8	8	22	39
2	2	24	63

Il carattere di controllo dovrebbe essere D perché $63 + 11 \equiv 0 \pmod{37}$, ma noi abbiamo ottenuto 2 quindi richiederemo la stringa.

14.1.5. Codice ISBN

Il **codice ISBN** (*International Standard Book Nigga*) è un codice di controllo usato per identificare in maniera univoca un libro. Di solito ha due formati, a 10 o 13 cifre (*a volte entrambi assieme*).

Consideriamo il codice a 10 cifre, che usa le somme pesate. Esso è definito da:

- prima cifra che identifica il country ID;
- seconda cifra e terza cifra che identificano il publisher ID;
- cifre dalla quarta alla nona che identificano book number;
- decima cifra che è la cifra di controllo, il **check digit**.

L'alfabeto usato contiene 11 caratteri (*perché 11 è primo e va bene con i moduli*): i numeri da 0 a 9 sono codificati con loro stessi, mentre il decimo carattere è la X.

14.1.6. Codice UPC

Il **codice UPC** (*Universal Product Code*) è il comunissimo **codice a barre**. Anche lui è un codice pesato. Ogni codice UPC è diviso in 3 blocchi:

- **manufacturer ID** (*chi ha prodotto l'oggetto*) di 6 cifre;
- **numero dell'oggetto** di 5 cifre;
- **check digit**.

Indicizzando i caratteri da 1, il check digit si calcola come

$$3 \sum_{i \text{ dispari}} x_i + \sum_{i \text{ pari}} x_i \bmod 10.$$

Quando ricevo un codice a barre, controllo se questa somma pesata modulo 10 è uguale a 0.

14.2. Introduzione matematica

Partiamo partiamo dai numeri interi \mathbb{Z} : esso è un **insieme infinito**, ma che non ci piace tanto perché la nostra macchina fa i conti con i numeri finiti.

Trasformiamo \mathbb{Z} nell'insieme \mathbb{Z}_n con la relazione di equivalenza della **congruenza**, che contiene i numeri interi modulo n e quindi è un **insieme finito**.

Lavorando su n non ci va tanto bene perché in generale siamo su un **anello**, nel quale non tutti gli elementi hanno inverso. Questo è un problema, non posso fare le divisioni non avendo l'inverso, che è roba da pazzi se pensiamo di non poter dividere per 2 in un calcolatore. Passiamo quindi a \mathbb{Z}_p , che è definito in modo analogo a \mathbb{Z}_n solo che p è un numero primo e quindi questo insieme è un **campo**, che mi garantisce la presenza di un inverso per ogni elemento dell'insieme.

Un calcolatore lavora con i byte, quindi ho 2^8 possibili valori, ma questo è ancora un problema perché 2^8 non è primo. Risolviamo scrivendo il byte come un **polinomio** a coefficienti in \mathbb{Z}_p , creando l'insieme $\mathbb{Z}_p[x]$.

Siamo nella stessa situazione di prima: sono ancora con un insieme infinito. Infatti, il grado di questi polinomi può anche essere infinito, quindi diamo un tetto al grado facendo il modulo con un polinomio, creando l'insieme finito $\mathbb{Z}_p[x] \bmod m(x)$.

Infine, ultimo problema che risolviamo, è quello dell'anello. Stiamo quozientando su un polinomio random, quindi non tutti gli elementi siamo sicuri abbiano un inverso. Per risolvere quest'ultimo problema cambiamo $m(x)$ in $p(x)$ **polinomio irriducibile** sul campo, e otteniamo l'insieme finale $\mathbb{Z}_p[x] \bmod p(x)$.

Quest'ultimo insieme è esattamente il **campo di Galois**. Un campo di Galois $\text{GF}(p^m)$ è un campo che contiene un numero di elementi uguale a p^m e ognuno di questi è un polinomio che grado massimo $m - 1$ a coefficienti in \mathbb{Z}_p . Il valore m indice anche il grado di $p(x)$.

Esempio 14.2.1: Sia $\text{GF}(4) = \text{GF}(2^2)$ il campo di Galois dei polinomi di grado massimo 1 con coefficienti in \mathbb{Z}_2 . Troviamo un polinomio irriducibile di grado 2 per questo campo.

Abbiamo diverse scelte:

- x^2 : fattorizzabile come $x \cdot x$, non va bene;
- $x^2 + x$: fattorizzabile come $x \cdot (x + 1)$, non va bene;
- $x^2 + 1$: fattorizzabile come $(x + 1) \cdot (x - 1) = (x + 1)^2 = (x - 1)^2$, non va bene;
- $x^2 + x + 1$: irriducibile.

Imponiamo, come dei dittatori, che questo polinomio irriducibile abbia invece una radice α . Con questa operazione troviamo l'**estensione algebrica**. Stiamo dicendo che $\alpha^2 + \alpha + 1 = 0$. Ma sono in \mathbb{Z}_2 quindi

$$\alpha^2 = -\alpha - 1 \longrightarrow \alpha^2 = \alpha + 1.$$

Questa estensione la chiamo $F[\alpha]$.

Con queste informazioni possiamo costruire le tabelline della somma e del prodotto.

Per il prodotto ho

$$\left(\begin{array}{c|cccc} \cdot & 0 & 1 & \alpha & \alpha+1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & \alpha & \alpha+1 \\ \alpha & 0 & \alpha & \alpha+1 & 1 \\ \alpha+1 & 0 & \alpha+1 & 1 & \alpha \end{array} \right)$$

La matrice qua è **simmetrica**.

Per la somma ho

$$\left(\begin{array}{c|cccc} + & 0 & 1 & \alpha & \alpha+1 \\ \hline 0 & 0 & 1 & \alpha & \alpha+1 \\ 1 & 1 & 0 & \alpha+1 & \alpha \\ \alpha & \alpha & \alpha+1 & 0 & 1 \\ \alpha+1 & \alpha+1 & \alpha & 1 & 0 \end{array} \right)$$

Anche in questo caso, la matrice è **simmetrica**.