Niccolò Boanini
Matricola: 7197315
Fine Dicembre 2025

Data Security And Privacy
Set 3 di Esercizi

Indice

1 Esercizio 2.2: Strategie e codici

Scenario Ci sono n monete numerate da 1 a n. Il giocatore A effettua su una della monete

- **Una sostituzione *leggera***: cioè sostituisce una moneta i con una più leggera, $(i, -)$;
- **Una sostituzione *pesante***: cioè sostituisce una moneta i con una più pesante, $(i, +)$;
- ***Nessuna* sostituzione**: cioè tutto rimane intatto $\forall$ moneta i, nil.

Indichiamo con $x \in \mathcal{X}$ lo *stato* in cui giunge A dopo la mossa effettuata sulla moneta i, dove $\mathcal{X}$ è lo spazio di tutti gli stati possibili. Con n monete quest'ultimo equivale a:

$$\mathcal{X} = \{(1,+),(1,-),(2,+),(2,-),...,(n,+),(n,-),nil\} \tag{1}$$

con cardinalità $|\mathcal{X}| = 2n+1$: 2 stati possibili per ogni moneta (più leggera o più pesante) + 1 stato per il caso nil.

Il giocatore B deve indovinare lo stato usando una bilancia a due piatti e mettendo sul piatto sinistro (S) e sul piatto destro (T) due sottoinsiemi disgiunti di monete; quindi, effettua una *pesata* che ha come esito:

- **Sinistra (L)**: il piatto S (sinistro) è più pesante;
- **Destra (R)**: il piatto T (destro) è più pesante;
- **Centro (=)**: i piatti S e T hanno lo stesso peso.

Ogni pesata riduce l'incertezza dividendo lo spazio delle possibilità in tre gruppi (in stile albero *ternario*). Vogliamo trovare una sequenza di pesate (una *strategia*) che ci permetta di isolare esattamente uno dei $2n+1$ stati possibili (*esito*). ***Fonti di alcune osservazioni e ragionamenti: [1, 2].***

1.1 Punto (a): Strategia con 5 monete

Si costruisce un albero *ternario* in cui ogni nodo è una pesata e le foglie sono i possibili esiti. Con $n = 5$, abbiamo 11 stati possibili ($2n+1$):

$$x = \{(1,+),(1,-),(2,+),(2,-),(3,+),(3,-),(4,+),(4,-),(5,+),(5,-),nil\} \tag{2}$$

All'inizio (Pesata 1, sulla **radice**), si confrontano due monete con altre due monete, lasciandone una esclusa. Ad esempio $S = \{1,2\}$ vs $T = \{3,4\}$, con 5 esclusa. Da qui l'albero si dirama in tre parti in base all'esito: **L, =, R**.

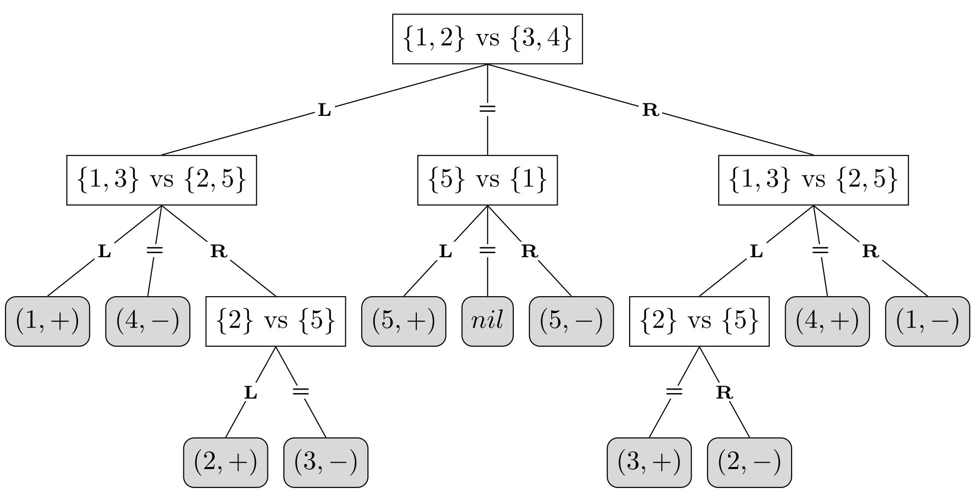

Figura 1: Albero ternario delle strategie con $n = 5$ monete. I nodi grigio scuri sono le *foglie* (esiti).

Analizziamo gli esiti della Pesata 1, percorrendo i rami dell'albero che si creano via via.

Primo ramo (esito L): Il piatto sinistro $S = \{1, 2\}$ è più pesante del destro $T = \{3, 4\}$ Quindi, una moneta tra $\{1, 2\}$ è più pesante, oppure una tra $\{3, 4\}$ è più leggera. La moneta 5 è sicuramente autentica. Lo spazio degli eventi si è ridotto a $\{(1, +), (2, +), (3, -), (4, -)\}$.

- Pesata 2: $\{1, 3\}$ vs $\{2, 5\}$. Confrontiamo la moneta 1 (*forse* pesante) e la 3 (*forse* leggera) contro la 2 (*forse* pesante) e la 5 (autentica).
  - Esito L (piatto sinistro più pesante): Il piatto sinistro è più pesante o il destro è più leggero. Dallo spazio degli eventi rimanente scritto sopra, l'unica possibilità è che la moneta 1 sia pesante[1]. $\rightarrow$ Esito: $(1, +)$
  - Esito = (equilibrio): Le monete pesate $\{1, 2, 3\}$ sono tutte autentiche. La falsificazione è nell'unica moneta possibile che non è stata inclusa in questa pesata, cioè la 4. Sappiamo però dal primo passo che il piatto della 4 era leggero. $\rightarrow$ Esito: $(4, -)$.
  - Esito R (piatto destro più pesante): Il piatto sinistro è leggero oppure il destro è pesante. Questo succede se la moneta 3 è leggera oppure se la moneta 2 è pesante. Spazio degli eventi rimanente $(3, -)$ e $(2, +)$.
    - Pesata 3: $\{2\}$ vs $\{5\}$. Confrontiamo la moneta 2 con una autentica.
      - Esito L: La moneta 2 è pesante. $\rightarrow$ Esito: $(2, +)$
      - Esito =: La moneta 2 è autentica. $\rightarrow$ Esito: $(3, -)$

Secondo ramo (esito =): I piatti sono in equilibrio. Quindi le monete 1, 2, 3, 4 sono tutte autentiche. La moneta falsa, se ce n'è una, è la 5. Usiamo una moneta sicuramente autentica (es. la 1) come riferimento.

- Pesata 2: $\{5\}$ vs $\{1\}$. Ancora tre esiti:
  - Esito L (piatto sinistro più pesante): La moneta 5 è più pesante. $\rightarrow$ Esito: $(5, +)$
  - Esito = (equilibrio): Nessuna moneta è falsa. $\rightarrow$ Esito: *nil*
  - Esito R (piatto destro più pesante): La moneta 5 è più leggera. $\rightarrow$ Esito: $(5, -)$

Terzo ramo (esito R): Il piatto destro $T = \{3, 4\}$ è più pesante del sinistro $S = \{1, 2\}$. Quindi, una moneta tra $\{3, 4\}$ è più pesante, oppure una tra $\{1, 2\}$ è più leggera. La moneta 5 è sicuramente autentica. Lo spazio degli eventi si è ridotto a $\{(1, -), (2, -), (3, +), (4, +)\}$.

- Pesata 2: $\{1, 3\}$ vs $\{2, 5\}$. Confrontiamo la moneta 1 (*forse* leggera) e la 3 (*forse* pesante) contro la 2 (*forse* leggera) e la 5 (autentica).
  - Esito R (piatto destro più pesante): Significa che il piatto sinistro è più leggero o il destro è più pesante. Vedendo lo spazio degli eventi rimanente scritto sopra, l'unica possibilità è che la moneta 1 sia leggera[2]. $\rightarrow$ Esito: $(1, -)$
  - Esito = (equilibrio): Le monete pesate $\{1, 2, 3\}$ sono tutte autentiche. La falsificazione è nell'unica moneta possibile che non è stata inclusa in questa pesata, cioè la 4. Sappiamo però dal primo passo che il piatto della 4 era pesante. $\rightarrow$ Esito: $(4, +)$.
  - Esito L (piatto sinistro più pesante): Il piatto sinistro è pesante oppure il destro è leggero. Questo succede se la moneta 3 è pesante oppure se la moneta 2 è leggera. Spazio degli eventi rimanente $(3, +)$ e $(2, -)$.
    - Pesata 3: $\{2\}$ vs $\{5\}$. Confrontiamo la moneta 2 con una autentica.
      - Esito R: La moneta 2 è leggera (il piatto con la 5 scende). $\rightarrow$ Esito: $(2, -)$
      - Esito =: La moneta 2 è autentica. $\rightarrow$ Esito: $(3, +)$

Osservazione. Si osserva la simmetria risolutiva del Primo e del Terzo ramo e in generale delle foglie ottenute sull'albero in Figura 1.

[1] se la 3 fosse leggera il piatto sinistro salirebbe; se la 2 fosse pesante il piatto destro scenderebbe)

[2] se la 3 fosse pesante il piatto sinistro scenderebbe; se la 2 fosse leggera il piatto destro salirebbe

1.2 Punto (b): Analisi del numero di pesate necessarie

Per risolvere questo punto, analizziamo quanto detto nel punto (a) e osserviamo[3] Figura 1.

Altezza dell'albero. Equivale al numero massimo di pesate necessarie per individuare lo stato (cioè il numero massimo di *nodi* per arrivare alla *foglia*). Osservando i rami più lunghi (ad esempio, quello che porta a $(1,+)$ o $(2,-)$), si vede che la profondità massima è:

$$L_{\max} = \max_{x \in \mathcal{X}} \{\ell(x)\} = 3 \text{ pesate} \tag{3}$$

con $\ell(x)$ il numero di pesate necessarie per identificare lo stato x (ℓunghezza).

Lunghezza media. Equivale al valore atteso della lunghezza rispetto alla distribuzione di probabilità $p(x)$. Assumendo che il giocatore A sceglie la moneta falsa (oppure il caso *nil*) in modo equiprobabile, si ha a che fare con una distribuzione uniforme, cioè:

$$p(x) = \frac{1}{|\mathcal{X}|} = \frac{1}{11} \quad \forall\, x \in \mathcal{X} \tag{4}$$

e quindi la lunghezza media è data dalla formula

$$L_{\text{avg}} = \mathbb{E}[\ell(x)] = \sum_{x \in \mathcal{X}} \ell(x) \cdot p(x) \tag{5}$$

poiché la probabilità è uniforme e può essere raccolta a fattor comune, la formula si semplifica in:

$$L_{\text{avg}} = \frac{1}{11} \sum_{x \in \mathcal{X}} \ell(x) \tag{6}$$

Ora, per calcolare la sommatoria, si contano le foglie che si trovano per ciascuna profondità nell'albero:

- **Foglie a profondità 2:** Sono 5. (Gli stati: $(4,-)$, $(5,+)$, *nil*, $(5,-)$, $(4,+)$).
- **Foglie a profondità 3:** Sono 6. (Gli stati: $(1,+)$, $(2,+)$, $(3,-)$, $(3,+)$, $(2,-)$, $(1,-)$).

Sostituendo questi valori otteniamo:

$$L_{\text{avg}} = \frac{1}{11} \cdot (5 \times 2 + 6 \times 3) = \frac{1}{11} \cdot (10 + 18) = \frac{28}{11} \approx 2,54 \text{ pesate} \tag{7}$$

1.3 Punto (c): Lower-bound numero *massimo* di pesate

Si cerca ora un limite inferiore per il numero massimo di pesate necessarie per trovare la mossa del giocatore A (con in generale n monete).

Relazione tra altezza e foglie. Poiché la bilancia a due piatti può restituire solo tre esiti distinti (Sinistra, Destra, Equilibrio), l'albero di decisione è un albero 3-ario (ternario). È una proprietà nota degli alberi m-ari che un albero di altezza k può avere al massimo m^k foglie [1]. Nel nostro caso ($m = 3$) e quindi:

$$\#\text{MaxFoglie}(k) = 3^k \tag{8}$$

Una strategia è valida se riesce a distinguere ogni possibile esito in modo univoco. In altre parole ogni elemento x dello spazio degli stati $\mathcal{X}$ deve essere associato ad *almeno* una foglia distinta dell'albero.

Pertanto, il numero massimo di foglie disponibili deve essere maggiore o uguale al numero degli stati da rappresentare:

$$\#\text{MaxFoglie}(k) = 3^k \geq |\mathcal{X}| = 2n + 1 \tag{9}$$

[3] L'albero è stato realizzato con il package *forest* di LaTeX.

Derivazione del limite inferiore. Per trovare il valore minimo di k, risolviamo la disuguaglianza rispetto a k. Prendendo quindi il logaritmo in base 3 da entrambi i membri:

$$\log_3(3^k) \geq \log_3(2n+1) \Longrightarrow k \geq \log_3(2n+1) \tag{10}$$

infine, dato che il numero di pesate k è un numero intero, prendiamo la parte intera superiore:

$$k \geq \lceil \log_3(2n+1) \rceil \tag{11}$$

1.4 Punto (d): Lower-bound numero *medio* di pesate

Si cerca infine un limite inferiore per il numero *medio* di pesate necessarie. Sfruttiamo la traccia dell'esercizio proposta.

Equivalenza tra Strategie e Codici. Ogni strategia di pesata valida può essere vista come un **codice istantaneo ternario** che indichiamo con C.

- L'**alfabeto di codifica** è l'insieme degli *esiti* della bilancia: $\mathcal{A} = \{L, R, =\}$, con $|\mathcal{A}| = 3$.
- I **messaggi da codificare** sono gli stati possibili di $\mathcal{X}$ (le *mosse del giocatore* A).
- La sequenza di pesate che porta al raggiungimento di uno stato x sarà la **parola di codice** per x.

  Di conseguenza, la **lunghezza della parola di codice** è esattamente il *numero di pesate* necessarie per quello stato.

Quanto detto ci porta a dire che la lunghezza media della strategia coincide con la lunghezza media del codice, che indichiamo con $L(C)$.

Sul Teorema di Shannon (Codifica di Sorgente). Sappiamo dalla teoria [6] che la lunghezza media di un codice istantaneo su un alfabeto di dimensione $|\mathcal{A}|$ è limitata inferiormente dall'entropia in base $|\mathcal{A}|$ della variabile aleatoria in questione, X. Nel nostro caso (per $|\mathcal{A}| = 3$):

$$L(C) \geq H_3(X) \tag{12}$$

Calcolo dell'Entropia. Assumiamo, come scritto nella traccia, che la mossa del giocatore A sia scelta secondo una distribuzione uniforme tra i $2n+1$ stati possibili. La funzione di massa di probabilità è:

$$p(x) = \frac{1}{2n+1}, \quad \forall x \in X \tag{13}$$

ricordiamo l'entropia **in base 3**:

$$H_3(X) = -\sum_{x \in X} p(x) \log_3 p(x) \tag{14}$$

E sostituiamo a questo punto in $p(x)$:

$$H_3(X) = -\sum_{i=1}^{2n+1} \frac{1}{2n+1} \log_3 \left(\frac{1}{2n+1} \right) \tag{15}$$

poiché la sommatoria contiene $2n+1$ termini costanti:

$$H_3(X) = -(2n+1) \cdot \left[\frac{1}{2n+1} \log_3 \left((2n+1)^{-1} \right) \right] \tag{16}$$

$$H_3(X) = -\log_3 \left((2n+1)^{-1} \right) = \log_3(2n+1) \tag{17}$$

Conclusione. Il limite inferiore per il numero medio di pesate per una strategia risolutiva è dato da:

$$L(C) \equiv L_{\text{medio}} \geq \log_3(2n+1) \tag{18}$$

2 Esercizio 2.5: Sicurezza incondizionata e unicity distance

2.1 Primo punto: Incertezza residua sulla chiave

Claim 1 *Si vuole dimostrare che* $H(K|C^n) = H(M^n) + H(K) - H(C^n)$*. Vogliamo cioè valutare l'incertezza residua sulla chiave dato solamente il chiphertext (cioè mettendoci nei panni dell'attaccante).*

Dimostrazione. Supponiamo l'invio di un messaggio tipico Alice $\rightarrow$ Bob, con attaccante Eve.

a) Il punto di vista di Alice (soggetto attaccato) Alice genera il plaintext M^n e usando la chiave K si calcola il ciphertext C^n.

- Poiché la scelta della chiave K utilizzata è **indipendente** dal messaggio generato si ha

$$H(M^n, K) = H(M^n) + H(K) \tag{19}$$

 e per la stessa ragione

$$H(K|M^n) = H(K) \tag{20}$$

 cioè essere a conoscenza del plaintext non varia l'incertezza sulla chiave.

- La conoscenza del plaintext e della chiave, rende chiaramente l'incertezza sul chiphertext nulla. Ovvero esiste una funzione f tale che $C^n = f(M^n, K)$. Si dice che il processo di cifratura è **deterministico** e vale:

$$H(C^n|M^n, K) = 0 \tag{21}$$

Utilizziamo ora come suggerito la *chain rule*[4] e sfruttiamo quanto appena detto. Tra le $3! = 6$ permutazioni possibili per scrivere la regola della catena con le variabili M^n, K e C^n scegliamo quella che ci sarà poi più conveniente:

$$H(M^n, K, C^n) = \underbrace{H(M^n) + \underbrace{H(K|M^n)}_{H(K)}}_{H(M^n,K)} + \underbrace{H(C^n|M^n, K)}_{0} \tag{22}$$

quindi $H(M^n, K, C^n) = H(M^n) + H(K)$.

b) Il punto di vista di Eve (soggetto attaccante) Eve vede solo il chiphertext C^n e mira a indovinare la chiave K per risalire poi al plaintext M^n. Usiamo ancora la *chain rule*, stavolta usando una permutazione che ci permetta di esplicitare in un addendo l'incertezza residua sulla chiave dato il chiphertext, cioè $H(K|C^n)$:

$$H(C^n, K, M^n) = H(C^n) + H(K|C^n) + \underbrace{H(M^n|C^n, K)}_{0} \tag{23}$$

quindi $H(C^n, K, M^n) = H(M^n, K, C^n) = H(C^n) + H(K|C^n)$.

c) Mettiamo insieme Dato che l'entropia totale del sistema è la stessa a prescindere da come la calcoliamo, possiamo dire che punto di vista di Alice = punto di vista di Eve.

$$\overbrace{H(M^n) + H(K)}^{\substack{\text{Alice} \\ H(M^n,K,C^n)}} = \overbrace{H(C^n) + H(K|C^n)}^{\substack{\text{Eve} \\ H(C^n,K,M^n)}} \tag{24}$$

Se ora isoliamo il termine $H(K|C^n)$ otteniamo quanto richiesto:

$$H(K|C^n) = H(M^n) + H(K) - H(C^n) \tag{25}$$

■

[4] $H(X_1, X_2, \dots, X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, \dots, X_1)$

2.2 Secondo punto: Unicity distance per rimuovere incertezza sulla chiave

Teorema 2 (Formula della unicity distance, [5]) *Vogliamo dare un valore a n_0, cioè quel valore minimo di n per cui l'incertezza sulla chiave dato il ciphertext diventa nulla ($H(K|C^{n_0}) = 0$). Ipotesi:*

- *Vale Equazione 25: $H(K|C^n) = H(M^n) + H(K) - H(C^n)$.*
- *L'entropia del plaintext è approssimabile come $H(M^n) \approx n \cdot h$, dove h è la entropy per letter del linguaggio.*
- *L'entropia del ciphertext è approssimabile come $H(C^n) \approx n \cdot \log|\mathcal{X}|$, con $\mathcal{X}$ l'alfabeto del plaintext.*

Claim 3 *Si dimostra che sotto queste ipotesi vale la formula*

$$n_0 = \frac{H(K)}{\log|\mathcal{X}| - h} \tag{26}$$

dove $|\mathcal{X}|$ è la cardinalità del linguaggio in questione.

Dimostrazione. Ripartiamo riscrivendo Equazione 25 con $n = n_0$:

$$H(K|C^{n_0}) = H(M^{n_0}) + H(K) - H(C^{n_0}) \tag{27}$$

Per definizione di n_0 impongo $H(K|C^{n_0} = 0)$, e sfruttando poi le ipotesi suddette sul plaintext e sul chipertext otteniamo:

$$0 \approx (n_0 \cdot h) + H(K) - (n_0 \cdot \log|\mathcal{X}|) \tag{28}$$

adesso è sufficiente manipolare l'espressione per poi isolare n_0:

$$n_0 \cdot \log|\mathcal{X}| - n_0 \cdot h = H(K) \quad \Longrightarrow \quad n_0 \cdot (\log|\mathcal{X}| - h) = H(K) \quad \Longrightarrow \quad n_0 = \frac{H(K)}{\log|\mathcal{X}| - h} \tag{29}$$

■

2.3 Terzo punto: Unicity distance in funzione della ridondanza del linguaggio

Teorema 4 (Formula per n_0 in funzione di R, [5]) *Vogliamo esprimere la formula trovata sulla unicity distance in funzione della ridondanza del linguaggio, definita come*

$$R = 1 - \frac{h}{\log|\mathcal{X}|} \tag{30}$$

Claim 5 *Si dimostra che la unicity distance in funzione di R vale*

$$n_0 = \frac{H(K)}{R \cdot \log|\mathcal{X}|} \tag{31}$$

dove $|\mathcal{X}|$ è la cardinalità del linguaggio in questione.

Dimostrazione. Ripartiamo manipolando Equazione 30. Mettiamo prima a fattor comune, e poi riscriviamo per far comparire il denominatore presente in Equazione 26:

$$R = \frac{\log|\mathcal{X}| - h}{\log|\mathcal{X}|} \quad \Longrightarrow \quad \underbrace{\log|\mathcal{X}| - h}_{\text{denom. Eq.26}} = \underbrace{R \cdot \log|\mathcal{X}|}_{\text{metto in Eq.26}} \tag{32}$$

quindi $n_0 = \frac{H(K)}{R \cdot \log|\mathcal{X}|}$. ■

2.4 Quarto punto: Applicazione della teoria con dati reali

Vogliamo stimare la unicity distance (n_0) per tre cifrari: quello a sostituzione monoalfabetico, quello di Vigenère e quello di Hill usando i parametri della lingua Inglese:

- Alfabeto ($|\mathcal{X}|$): 26 lettere.
- Entropia del linguaggio (h): 1.5 bit/lettera.
- Entropia massima ($\log_2 |\mathcal{X}|$): $\log_2(26) \approx 4.7$ bit/lettera.
- Denominatore della formula (Equazione 26): $\log_2(26) - h = 4.7 - 1.5 = 3.2$ bit/lettera.

Per i calcoli useremo quindi la formula

$$n_0 = \frac{H(K)}{3.2} \tag{33}$$

per ogni cifrario sarà quindi sufficiente calcolare l'entropia della chiave $H(K)$, esplicitando prima il formato della chiave e poi calcolando lo spazio delle chiavi.

I **Cifrario a Sostituzione Monoalfabetica, [7]**. Qui, per lo spazio delle chiavi si considera il numero di possibili permutazioni di 26 lettere, che è 26!. Calcolo ora l'Entropia della Chiave $H(K)$ e sostituisco il risultato in Equazione 33:

$$H(K) = \log_2(26!) \approx \log_2(4.03 \cdot 10^{26}) \approx 88.4 \text{ bit} \tag{34}$$

da cui l'espressione per n_0:

$$n_0 = \frac{88.4}{3.2} \approx 27.6 \tag{35}$$

La unicity distance è circa 28. Quindi sono sufficienti altrettanti caratteri di chipertext per *rompere* teoricamente una sostituzione monoalfabetica.

II **Cifrario di Vigenère**. Qui, per lo spazio delle chiavi: si considerano le 26^L possibili chiavi di lunghezza L. L'entropia della Chiave è

$$H(K) = \log_2(26^L) = L \cdot \log_2(26) \approx 4.7 \cdot L \text{ bit} \tag{36}$$

da cui:

$$n_0 = \frac{4.7 \cdot L}{3.2} \approx 1.47 \cdot L \tag{37}$$

La unicity distance è circa 1.5 volte la lunghezza della chiave.

III **Cifrario di Hill**. La chiave è considerabile come una matrice quadrata $m \times m$ di numeri interi (modulo 26). Per lo spazio delle chiavi: la matrice ha m^2 elementi totali, ogni elemento può assumere 26 valori. Quindi ci sono 26^{m^2} matrici possibili.

L'entropia della chiave è

$$H(K) = \log_2(26^{m^2}) = m^2 \cdot \log_2(26) \approx 4.7 \cdot m^2 \text{ bit} \tag{38}$$

da cui

$$n_0 = \frac{4.7 \cdot m^2}{3.2} \approx 1.47 \cdot m^2 \tag{39}$$

La unicity distance cresce col quadrato della dimensione del blocco (m).

In Figura 2 sono mostrati i risultati del confronto della Unicity Distance (n_0) tra i cifrari appena descritti.

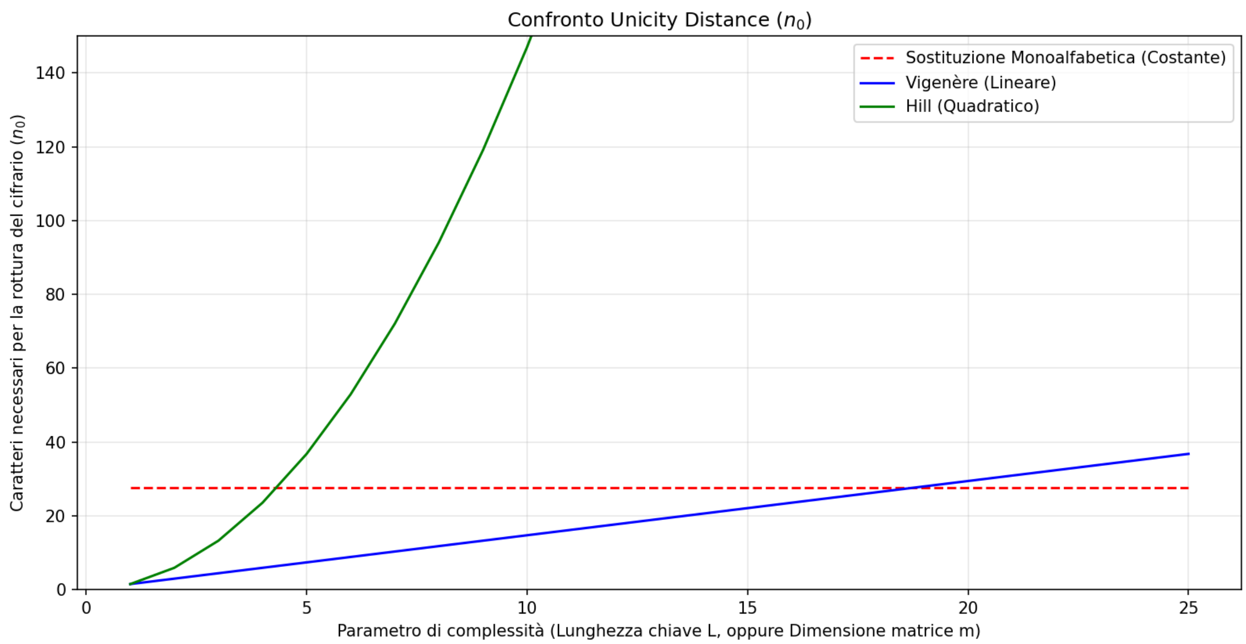

Figura 2: Confronto Unicity Distance (n_0). Dal grafico si vede che per chiavi molto corte ($L < 5$), il cifrario a Sostituzione è addirittura più robusto di Vigenère o Hill. Tuttavia, appena il parametro cresce, Hill in particolare diventa superiore molto rapidamente grazie alla crescita quadratica dello spazio delle chiavi (m^2).

3 Esercizio 3.1: Test di Ipotesi

3.1 Scenario e preliminari di teoria

Vogliamo distinguere se un testo $x = x_1, x_2, \ldots, x_n$ è un testo random oppure è un testo (eventualmente *shiftato*) in lingua inglese. Tale distinzione viene fatta mediante un **Test di Ipotesi** in cui la H_0 (ipotesi nulla) corrisponde a testo random, la H_1 (ipotesi alternativa) corrisponde a testo in lingua inglese. Nel primo caso i caratteri del testo sono estratti uniformemente e in modo indipendente l'uno con l'altro dall'alfabeto, nel secondo il testo viene generato seguendo la distribuzione di frequenza inglese (*letter frequency table*), consultabile in [3].

Dalla traccia dell'esercizio, si definisce la variabile $Y \sim \text{Bernoulli}(p)$ che confronta due caratteri X_i, X_j:

$$Y = \begin{cases} 1 & \text{se } X_i = X_j \quad \text{(coincidenza)} \\ 0 & \text{se } X_i \neq X_j \end{cases} \tag{40}$$

Osserviamo ora come $P(Y = 1)$ varia molto a seconda della ipotesi che si considera. Infatti, secondo l'ipotesi H_0 ogni lettera ha probabilità $1/26$ di essere scelta e la probabilità che due lettere siano uguali è

$$p_0 = P(Y = 1|H_0) = \sum_{i=1}^{26} \left(\frac{1}{26}\right)^2 = \frac{1}{26} \approx 0,038 \tag{41}$$

secondo invece l'ipotesi H_1, e quindi sfruttando le frequenze relative delle lettere in inglese (*letter frequency table*):

$$p_1 = P(Y = 1|H_1) = \sum_{i=1}^{26} p_i^2 \approx 0,065 \tag{42}$$

Quanto detto ci aiuta per questo motivo: dato che lo *shift* su di un testo effettua una permutazione delle lettere ma non fa variare le loro frequenze relative, p_1 rimane invariato anche per testi cifrati con uno *shift*. Questo rende l'Indice di Coincidenza (IC) una misura adatta per questo tipo di test. Infatti IC estende questo confronto a tutte le possibili coppie di lettere nel testo. In un testo di n caratteri, si hanno $\binom{n}{2} = \frac{n(n-1)}{2}$ coppie di lettere. IC fa la media di tutte le possibili $Y_{i,j}$. Si usa la formula tipica[5]:

$$IC = \frac{\sum_{k=1}^{26} \frac{f_k(f_k-1)}{2}}{\frac{n(n-1)}{2}} = \frac{\sum_{k=1}^{26} f_k(f_k-1)}{n(n-1)} \tag{43}$$

```python
def ic(text):
    """Calcola l'Indice di Coincidenza del testo."""
    n = len(text)
    counts = Counter(text) # conta le occorrenze di ogni carattere
    num = 0
    for f in counts.values():
        num += f * (f - 1)
    den = n * (n - 1)
    return num / den
```

Listing 1: Funzione per calcolare l'indice di coincidenza di un dato testo

Riassumendo, la formula per Y risulta essere:

$$\boxed{Y \sim \text{Bernoulli}(p) \quad \text{con} \quad p = \begin{cases} 1/26 \approx 0,038 & \text{sotto } H_0 & \to p_0 \\ \sum p_i^2 \approx 0,065 & \text{sotto } H_1 & \to p_1 \end{cases}} \tag{44}$$

3.2 Descrizione del test, errori e criteri

Criterio di Decisione (soglia/*threshold*) Per discriminare tra le due ipotesi dobbiamo fissare una soglia decisionale T (*threshold*). In prima analisi[6], un buon valore per minimizzare l'errore complessivo (assumendo probabilità a priori uguali per H_0 e H_1) è quello a metà tra p_0 e p_1:

$$T = \frac{p_0 + p_1}{2} = \frac{0,038 + 0,065}{2} \approx 0.05173 \tag{45}$$

La **regola di decisione** utilizzata nel test è quindi:

- Se $IC(x) \leq T \Rightarrow$ Accetto H_0 (Testo Random);
- Se $IC(x) > T \Rightarrow$ Accetto H_1 (Testo Inglese *shiftato*).

```python
def hp_test(text):
    """
    Test di Ipotesi: Ritorna 1 se Inglese (H1), 0 se Random (H0).
    La soglia 0.05173, il punto medio tra p0 (0.038) e p1 (0.065).
    """
    ic_value = ic(text)
    threshold = 0.05173

    if ic_value > threshold:
        return 1 # H1: IC alto, probabilmente Inglese
    else:
        return 0 # H0: IC basso, probabilmente Random
```

Listing 2: Funzione di decisione del Test di Ipotesi basata sulla soglia $T = 0.05173$.

[5] Raggruppa per lettera dell'alfabeto: se lettera A compare f_A volte nel testo, il numero di coppie A-A che si può formare è $\frac{f_A(f_A-1)}{2}$. Se sommiamo questo per tutte le 26 lettere dell'alfabeto si ottiene la formula di Equazione 43.

[6] Ho impostato questo valore perché mi sembrava ragionevole sulla base delle mie conoscenze statistiche, specie quando si ha a che fare con gaussiane (modellazione di IC per H_0 e H_1, vedi Figura 3) in cui ipotizziamo che abbiano la stessa σ.

Stima *teorica* degli Errori (di primo e secondo tipo) Per confrontare i risultati ottenuti con la teoria, calcoliamo le probabilità di errore attese. Per n sufficientemente grande (per il TLC), la distribuzione dell'Indice di Coincidenza può essere approssimata da una Gaussiana $\mathcal{N}(\mu, \sigma^2)$.

Le probabilità di errore si definiscono come:

- **Errore di I tipo (α_n):** Probabilità di classificare come Inglese un testo Random (Falso Positivo).
- **Errore di II tipo (β_n):** Probabilità di classificare come Random un testo Inglese (Falso Negativo).

Utilizzando la funzione di ripartizione della normale standard, $\Phi(z)$, le formule implementate sono:

$$\alpha_n = \mathbf{P}(IC > T|H_0) \approx 1 - \Phi\left(\frac{T - \mu_0}{\sigma_0}\right) \tag{46}$$

$$\beta_n = \mathbf{P}(IC \leq T|H_1) \approx \Phi\left(\frac{T - \mu_1}{\sigma_1}\right) \tag{47}$$

dove si pone $\begin{cases} \mu_0 = p_0 \\ \mu_1 = p_1 \end{cases}$ (valori attesi); $\sigma_0 = \sigma_1 = \frac{\sqrt{2(k-1)}}{k \cdot \sqrt{n(n-1)}}$ (deviazioni standard teoriche dell'IC per un testo di lunghezza n, e con $k = 26$).

```
def phi(z):
    return norm.cdf(z) # CDF normale standard.
    # Alternativa: return 0.5*(1+math.erf(z/math.sqrt(2)))

def normal_cdf(x, mu, sigma):
    """
    Calcola P(X <= x) standardizzando prima la variabile: Phi((x - mu) / sigma).
    """
    z = (x - mu) / sigma # Standardizzazione
    return phi(z) # Applicazione della funzione Phi
```

Listing 3: Implementazione della CDF Normale con standardizzazione $z = (x - \mu)/\sigma$

```
def get_theorical_errors(n, threshold):
    """Calcola alpha e beta (teorici, 'precisi')."""
    # H0: Random
    mu0 = 03846 # approssima 1.0 / 26.0
    sigma0 = math.sqrt(2 * (26 - 1)) / (26 * math.sqrt(n * (n - 1)))
    alpha = 1 - normal_cdf(threshold, mu0, sigma0)

    # H1: Inglese (IID)
    mu1 = 0.065
    sigma1 = sigma0  # ipotesi: gaussiana con stessa varianza
    beta = normal_cdf(threshold, mu1, sigma1)

    print_overlapped_gaussians(mu0, sigma0, mu1, sigma1, threshold, n) # vedi Figura 3

    return alpha, beta
```

Listing 4: Funzione Python per il calcolo degli errori **teorici**

3.3 Setup della Simulazione

La procedura Python esegue il test su un campione di 100 blocchi di testo di lunghezza n variabile. Sono stati generati **tre** tipi di testi distinti:

1. **Generatore H_0 (Random):** Estrae caratteri in modo indipendente ed equiprobabile dall'alfabeto. Ad esempio ($n = 30$): `yubvnoyurkfbcgdnetadbbmnmezxep`.

2. **Generatore H_1 Sintetico (I.I.D.):** Estrae caratteri in modo indipendente rispettando le frequenze della lingua inglese. Vedi Listing 5. Un esempio di testo iid generato ($n = 30$) è `girtlcotaenottndpnowfeossssvbe`.

3. **Generatore H_1 Reale (Testo Reale):** Estrae sottostringhe dal primo capitolo *Moby Dick*. Qui si preservano le regole linguistiche naturali (es digrammi, grammatica... cfr. Osservazione sotto). Un esempio di sottostringa generata ($n = 30$) è `hforthingsremoteilovetosailfor`.

```
def english_text_iid(n):
    text = ""
    # random.choices estrae con rimpiazzo basandosi sui pesi (weights)
    chars = random.choices(
        population=ALPHABET, # alfabeto inglese a-z
        weights=list(LETTER_FREQUENCY_TABLE.values()),
        k=n
    )
    for char in chars:
        text = text + char

    return text
```

Listing 5: Generatore di testo Inglese Sintetico (Modello I.I.D.): rispetta la tabella delle letter frequency.

Osservazione: perché (anche) un testo reale? Il terzo testo (primo capitolo di Moby Dick, [4]), sebbene non esplicitamente richiesto dall'esercizio, è stato introdotto per valutare l'efficacia del test in un contesto *reale*. Se infatti da una parte il generatore I.I.D. rispetta fedelmente la tabella delle frequenze inglesi, dall'altra non tiene conto della struttura linguistica e grammaticale: le estrazioni dei caratteri avvengono infatti in modo completamente indipendente l'una dall'altra. Il testo reale, al contrario, preserva le dipendenze naturali tra le lettere[7], estraendo sottostringhe (caratteri contigui) da un testo vero in lingua inglese.

3.4 Avvio del test, risultati e conclusioni

```
for i in range(hblocks): # 1. TEST H0 (Random Text) -> Calcola Alpha

    text = random_text(n) # genera testo random
    if hp_test(text, threshold) == 1: # se viene classificato come inglese
        fp += 1 # incremento il contatore dei falsi positivi

for j in range(hblocks): # 2. TEST H1 I.I.D. (Sintetico) -> Calcola Beta (I.I.D.)

    plain_text = english_text_iid(n)
    cipher_text = random_shift_cipher(plain_text)
    if hp_test(cipher_text, threshold) == 0: # se viene classificato come random
        fn_iid += 1 # incremento il contatore dei falsi negativi

for k in range(hblocks): # 3. TEST H1 REALE (Moby Dick) -> Calcola Beta (Reale)

    plain_text = english_text_real(n) # genera testo reale (es. Moby Dick)
    cipher_text = random_shift_cipher(plain_text) # cifratura shift
    if hp_test(cipher_text, threshold) == 0: # se viene classificato come random
        fn_real += 1 # incremento il contatore dei falsi negativi
```

Listing 6: Codice di avvio del Test

Osservando le gaussiane in Figura 3 e i valori di output riportati in Tabella 1, si nota chiaramente come la lunghezza del testo (n) sia il fattore più importante per l'affidabilità del test.

Si veda Figura 3 per i risultati grafici teorici (formule), che ora descriviamo confrontandoli anche con i risultati sperimentali i riportati in Tabella 1:

[7](come ad esempio la sequenza 'TH' che è molto frequente in inglese)

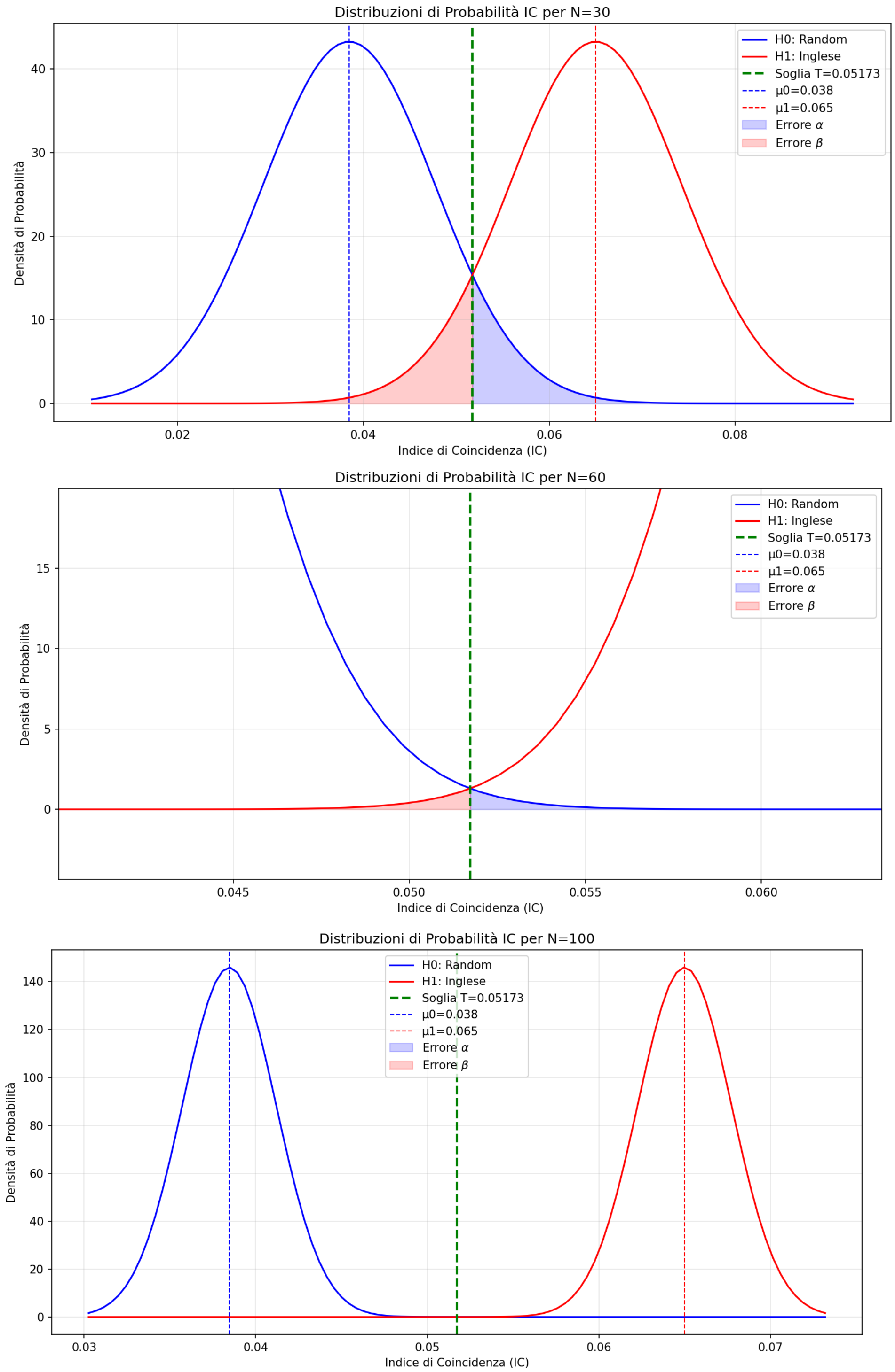

Figura 3: Rappresentazione del test di ipotesi per l'Indice di Coincidenza (IC). Le distribuzioni H_0 (testo casuale, centrata in 0.038) e H_1 (lingua inglese, centrata in 0.065) sono modellate come normali con deviazione standard σ_0. L'area blu α rappresenta la probabilità di errore di primo tipo (falso positivo), mentre l'area rossa β indica l'errore di secondo tipo (falso negativo) rispetto alla threshold fissata a 0.05173. Si nota come, aumentando la lunghezza del testo n, l'errore teorico tenda a svanire. Ciò accade perché la deviazione standard diminuisce ($\sigma \propto 1/\sqrt{n}$) e quindi le curve si restringono attorno alle rispettive medie, riducendo l'area di sovrapposizione (si separano meglio).

Lunghezza n	Errore α (Falsi Positivi)		Errore β (Falsi negativi)		
	Teorico	**Empirico (su H_0)**	**Teorico**	**Sintetico (I.I.D)**	**Reale (Moby Dick)**
$n = 30$	7.50%	8.00%	7.50%	14.00%	26.00%
$n = 60$	0.18%	0.00%	0.18%	5.00%	10.00%
$n = 100$	0.00%	0.00%	0.00%	0.00%	4.00%

Tabella 1: Confronto degli errori teorici (formule) ed empirici (sperimentali, sui blocchi di testo) con al variare della lunghezza del testo n (Soglia $T \approx 0.05173$). Si nota come l'errore sul testo *reale* rimane superiore a quello teorico anche per n elevati. La procedura è testata su 100(+50) blocchi di testo: 50 verificano H_0, 50 verificano H_1 (+ altre 50 sottostringhe estratte dal primo capitolo di Moby Dick per verificare H_1).

1. $n = 30$**: Testo breve, tanti errori**. Con testi cosi' brevi, il grafico teorico mostra una sovrapposizione importante tra le due campane, che si traduce nei dati numerici sperimentali: l'errore teorico previsto è alto (7.5%), ma quello reale lo è ancora di più e, soprattutto, molto instabile. Infatti come notato se eseguiamo il codice con diverse esecuzioni, se viene estratto un blocco di testo *sfortunato* (magari con poche lettere ripetute o parole rare) l'errore β (falsi negativi) può arrivare anche al 26%. L'Indice di Coincidenza con n cosi' piccolo non è uno strumento robusto (la varianza è ancora troppo alta).

2. $n = 60$**: Intanto scompaiono i Falsi Positivi**: Raddoppiando la lunghezza del testo, la situazione cambia in modo importante. La teoria (Figura 3) prevede che le code delle campane siano molto meno sovrapposte, e anche i dati sperimentali lo confermano: l'errore α (scambiare random per inglese) va a zero. Dall'altro lato, per quanto riguarda l'erroe β, continuano a persistere piccoli errori sul testo I.I.D., che sono ancora più alti sul testo reale, seppur notevolmente diminuiti rispetto al caso precedente.

3. $n = 100$**: testo finalmente sufficientemente lungo**: Con 100 caratteri, le due gaussiane di Figura 3 sono ormai visivamente separate. Teoricamente quindi l'errore dovrebbe essere nullo. E infatti, sui dati sintetici (I.I.D.) e sul random, il test è commette lo 0% di errori. Sul testo reale, anche con $n = 100$, rimane un piccolo errore residuo (circa del 4% che rieseguendo il codice varia leggermente o talvolta svanisce).

I test eseguiti confermano che l'Indice di Coincidenza è uno strumento potente in questo contesto, ma la sua predizione teorica (Figura 3) può essere un po' *ottimistica* in termini di errori α e β rispetto a quanto accade nella realtà, specie se utilizziamo blocchi di testo non sufficientemente lunghi.

Riferimenti bibliografici

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press and McGraw-Hill, fourth edition, 2009.

[2] Greg Egan. Find the fake coin.

[3] Tom Linton. Relative frequencies of letters in general english plain text, citato anche dalla pagina wikipedia delle letter frequency.

[4] Herman Melville. *Moby-Dick; or, The Whale*. Project Gutenberg, 1851. Versione digitale: eBook no. 2701, Project Gutenberg. Ultima modifica: 11 settembre 2025.

[5] C.E. Shannon. Communication theory of secrecy systems.

[6] Wikipedia. Shannon: primo teorema. `https://en.wikipedia.org/wiki/Shannon%27s_source_coding_theorem`. Accesso il: 2025-12-20.

[7] Wikipedia. Unicity distance. `https://en.wikipedia.org/wiki/Unicity_distance`.