Fondamenti di Teoria dell'Informazione

di Thomas D. Schneider

Traduzione Italiana *non letterale* interamente svolta da:
Simone Baldi
Laureando in Scienze dell'Informazione presso l'Università degli Studi, Milano (Italy)

version = 2.33 of primer.tex 27 Luglio 1995

Questo documento è stato scritto per quei biologi molecolari che non hanno molta famiglirità con la Teoria dell'Informazione. Il suo scopo, pertanto è presentare queste idee in modo che tutti possano capire come applicarle ai siti di filamenti (di DNA) [1, 2, 3, 4, 5, 6, 7, 8, 9]. La maggior parte di questo materiale si trova facilmente nei testi introduttivi alla Teoria dell'Informazione. Benché il lavoro originale di Shannon sulla Teoria dell'Informazione [10] a volte può risultare di difficile lettura, in altri contesti viene molto decantato. Saltando però le parti più ostiche si può trovarlo interessante e piacevole. Pierce ha già pubblicato un libro a portata di tutti [11] che è una buona introduzione alla Teoria dell'Informazione. Altre introduzioni sono elencate nei riferimenti [1]. Un manuale di grande utilità è in riferimento [12]. Il lavoro completo di Shannon è stato refenziato in [13]. Informazioni su come ordinare questi libri sono fornite nel file di testo:

> http://www.lecb.ncifcrf.gov/∼toms/bionet.info-theory.faq.html
> #REFERENCES-Information_Theory

Reperibile via ftp anonimo.

Altri opuscoli e documentazioni in merito si possono trovare alla pagina del World Wide Web:

> http://www.lecb.ncifcrf.gov/∼toms/

Nota: Se si incontrano problemi nella lettura di uno o più passi di questo documento, per cortesia inviatemi una e-mail indicando esattamente il/i punto/i problematico/i. Se lo riterrò opportuno, modficherò il testo per renderne più agevole la lettura. I miei personali ringraziamenti vanno a tutti coloro che hanno puntualizzato su alcune questioni il documento sino a questa versione.

Informazione e Incertezza

Informazione e Incertezza sono termini tecnici usati per descrivere misurazioni coinvolte in qualunque processo che debba selezionare uno, o più oggetti, in un ben determinato insieme di oggetti. Non vogliamo occuparci del significato o delle implicazioni dell'Informazione dal momento che nessuno sa come fare ciò utilizzando un metodo matematico. Supponiamo di avere un dispositivo che sia in grado di generare (produrre) 3 simboli, A, B, e C. Quando siamo in attesa di ricevere il prossimo simbolo, siamo *incerti* su quale simbolo verrà generato (prodotto). Un simbolo ci giunge, e noi lo osserviamo; la nostra *incertezza* diminuisce, e noi notiamo di aver ricevuto una certa quantità d'*informazione.* Dunque, l'informazione non è altro che una diminuzione di *incertezza.* Come possiamo misurare questa *incertezza*? Il modo più semplice può essere quello di dire che, in questo particolare caso, abbiamo una ≫Incertezza di 3 simboli≪. Questo ragionamento comincia ad esser chiaro dal momento in cui iniziamo ad osservare un secondo dispositivo, il quale, supponiamo, generi (produca) i simboli 1 e 2. Questo secondo dispositivo 'genera in noi' una ≫incertezza≪ di 2 simboli. Se combiniamo i due dispositivi in un unico dispositivo, osserviamo che ci sono 6 possibilità, A1, A2, B1, B2, C1, C2. Quest'ultimo dispositivo ha una ≫incertezza≪ di 6 simboli. Questo non è certo il modo in cui siamo abituati a pensare l'informazione; se riceviamo due libri, preferiamo pensare che otteniamo due volte l'informazione che avremmo ottenuto ricevendo un solo libro. In questa linea di idee ci piacerebbe imporre che la nostra unità di misura abbia una struttura additiva.

... E, infatti, così faremo.

È semplice farlo, basta prendere il logaritmo del numero dei simboli possibili perciò possiamo sommare i logaritmi invece di moltiplicare tra loro il numero dei simboli dei dispositivi.

Nel nostro esempio, il primo dispositivo 'genera in noi' una incertezza di $log(3)$, il secondo di $log(2)$ ed il dispositivo combinato di $log(3)+log(2)=log(6)$. Le basi che prendiamo per calcolare i logaritmi determinano le unità di misura. Quando usiamo la base 2 l'unità è il bit (la base di 10 ci dà i digits e la base naturale dei logaritmi, *e*, ci dà i nats [14] oppure i nits [15]). Quindi se un dispositivo genera un solo simbolo, abbiamo una incertezza pari a $log_2(1) = 0$ bits, cioè noi **non** abbiamo nessuna incertezza su ciò che il dispositivo sta per produrre. Se esso genera due simboli la nostra incertezza sarà pari a $log_2(2) = 1$ bit. (D'ora in poi useremo sempre la base 2.) Leggendo un mRNA, si osserva che se il ribosoma incontra una qualunque delle 4 basi ugualmente probabili, allora l'incertezza è pari a 2 bits. In questo modo, la nostra formula per l'incertezza è $log_2(M)$, dove M è il numero totale di simboli possibili. Il prossimo passo è quello di estendere la formula in modo che si possano trattare i casi in cui i simboli **non** siano equiprobabili. Per esempio, se ci sono 3 simboli possibili ma uno di essi **non** appare **mai**, allora l'incertezza è pari a 1 bit. Se il terzo

simbolo appare raramente rispetto agli altri 2, allora la nostra incertezza potrà essere superiore a 1 bit, ma certamente non raggiungerà mai $log_2(3)$ bits.

Procediamo a rivedere la formula in questo modo:

$$\begin{aligned} \log_2(M) &= -\log_2(M^{-1}) \\ &= -\log_2(\frac{1}{M}) \\ &= -\log_2(P) \end{aligned} \tag{1}$$

dove $P = 1/M$ è la probabilità che uno qualunque dei nostri possibili simboli appaia. (Se non si ricorda questo trucchetto del 'tirare fuori il segno ricordiamo che $logM^b = blogM$ e poniamo $b = -1$.)

Ma ora generalizziamo al caso in cui i simboli abbiano differenti probabilità di apparire e chiamiamo queste probabilità P_i; sappiamo che sommando tutte queste probabilità, estendendo la somma a tutti gli M simboli possibili, si ottiene 1 (ce lo dice il calcolo delle probabilità, e la statistica):

$$\sum_{i=1}^{M} P_i = 1. \tag{2}$$

(Ricordiamo che col simbolo *sum* si intende: 'sommare gli oggetti che stanno entro la parentesi e che sono indicizzati da *i*, e far partire *i* da 1 fermandosi quando *i* arriva a *M*'.)

La *sorpresa* che riceviamo quando vediamo apparire l'i-esimo tipo di simbolo, chiamato anche »surprisal« da Tribus [16], e definita per analogia con $-log_2(P)$ è pari a:

$$u_i = -log_2(P_i). \tag{3}$$

Per esempio, se P_i si avvicina molto a 0, allora saremo molto sorpresi nel vedere apparire l'i-iesimo simbolo (dal momento che quest'ultimo non dovrebbe **mai** apparire), e infatti la formula dice che $u_i = 0$.

L'**incertezza** è dunque la *sorpresa media* per una sequenza infinita di simboli generati dal nostro dispositivo. Per il momento, troviamo la media per una sequenza di soli N simboli. Supponiamo che l'i-esimo simbolo appaia N_i volte così che:

$$N = \sum_{i=1}^{M} N_i. \tag{4}$$

Ci saranno N_i casi in cui avremo sorpresa u_i. La sorpresa media per N simboli sarà:

$$\frac{\sum_{i=1}^{M} N_i u_i}{\sum_{i=1}^{M} N_i}. \tag{5}$$

Inserendo il denominatore della (5) nella somma che sta al numeratore della stessa otteniamo:

$$\sum_{i=1}^{M} \frac{N_i}{N} u_i \tag{6}$$

Se valutiamo questa misura su una sequenza infinita di simboli, allora la frequenza N_i/N tende a P_i, la probabilità dell'i-esimo simbolo. Con questa sostituzione, osserviamo che la nostra *sorpresa media* (che chiameremo H) diventa:

$$H = \sum_{i=1}^{M} P_i u_i. \tag{7}$$

Infine, sostituendo u_i, con la sua espressione esplicita, abbiamo la famosa formula generale di Shannon per l'**incertezza**:

$$\boxed{H = -\sum_{i=1}^{M} P_i \log_2 P_i \quad \text{(Bits pro Symbol).}} \tag{8}$$

Shannon ricavò questa formula attraverso passagi assai più rigorosi rispetto a ciò che abbiamo fatto noi, selezionando moltissime auspicabibli proprietà per l'incertezza e, solo successivamente, derivando la formula. Spero che i passagi che abbiamo finora seguito abbiano dato almeno il senso di come questa formula funziona. Per vedere come si presenta questa funzione possiamo tracciarla nel caso di due simboli. Si presenta come segue: 1:[1]

Tengo a far notare che la curva è simmetrica, raggiunge il suo massimo quando i due simboli sono equiprobabili (probabilità = 0.5).

Decresce bruscamente sino a zero tutte le volte che uno dei simboli diviene dominante a spese degli altri simboli. Infatti se uno dei due simboli ha probabilità molto vicina ad 1 la nostra sorpresa nel vederlo apparire è pressoché nulla!

Come esercizio istruttivo, supponiamo che tutti i simboli siano equiprobabili. A cosa si riduce la formula per H (formula (8))? Prova a pensarci da solo prima di proseguire nella lettura.

****

[1] Il programma usato per creare questo grafico è reperibile via ftp anonimo dal file: http://www.lecb.ncifcrf.gov/~toms/delila/hgraph.html

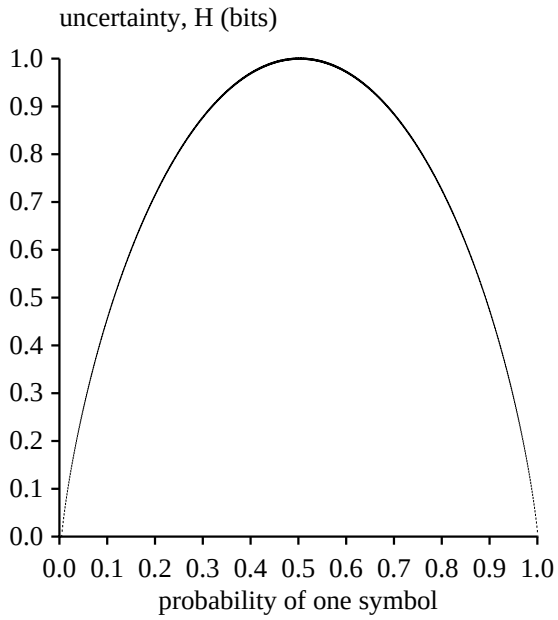

Figura 1: H-Funktion für zwei Symbole.

Equiprobabili significa che $P_i = 1/M$, perciò se sostituiamo nella equazione dell'incertezza otteniamo:

$$H_{equiprobabile} = -\sum_{i=1}^{M} \frac{1}{M} \log_2 \frac{1}{M} \tag{9}$$

Però, ragioniamo, M non è funzione di i, possiamo quindi portarla fuori dalla somma e otteniamo:

$$\begin{aligned} H_{equiprobabile} &= -(\frac{1}{M} \log_2 \frac{1}{M}) \sum_{i=1}^{M} 1 && (10) \\ &= -\left(\frac{1}{M} \log_2 \frac{1}{M}\right) M \\ &= -\log_2 \frac{1}{M} && (11) \\ &= \log_2 M \end{aligned}$$

Che è la semplice equazione con la quale siamo partiti.

Può essere valutata per un dato numero di simboli (ad esempio, con M fissato) allora l'incertezza H ha il suo valore massimo quando i simboli sono equiprobabili. Per esempio una moneta prefettamente bilanciata è molto più difficile da trovare rispetto ad una moneta sbilanciata. Un altro esercizio potrebbe essere: Qual'è l'incertezza se abbiamo 10 simboli e soltanto uno di questi appare? (Suggerimento $\lim_{p \to 0} p \log\ p = 0$ se poniamo $p = 1/M$ e usiamo la regola di de l'Hôpital, allora $0 \log_2 0 = 0$.)

Cosa significa, allora, dire che un segnale ha 1.75 bits per simbolo?

Ciò significa che possiamo convertire il segnale originale in sequenze di zeri e uni (cifre binarie), e, mediamente, dover utilizzare 1.75 cifre binarie per ogni simbolo del segnale originale. Certi simboli (i più rari) richiederanno più cifre binarie, mentre altri (i più comuni) ne richiederanno meno.

Ecco un esempio di quanto abbiamo appena detta: Supponiamo di avere $M = 4$ simboli:

$$A \quad C \quad G \quad T \tag{12}$$

con relative probabilità (P_i):

$$P_A = \frac{1}{2}, \quad P_C = \frac{1}{4}, \quad P_G = \frac{1}{8}, \quad P_T = \frac{1}{8}, \tag{13}$$

che hanno, rispettivamente, sorprese ($-\log_2 P_i$):

$$u_A = 1 \text{ Bit}, \quad u_C = 2 \text{ Bits}, \quad u_G = 3 \text{ Bits}, \quad u_T = 3 \text{ Bits}, \tag{14}$$

con tutto ciò, otteniamo una incertezza pari a:

$$H = \frac{1}{2} \cdot 1 + \frac{1}{4} \cdot 2 + \frac{1}{8} \cdot 3 + \frac{1}{8} \cdot 3 = 1.75 \text{ (bits per simbolo)}. \tag{15}$$

Ricodifichiamo tutto ciò in modo che il numero di cifre binarie eguaglino la sorpresa:

$$\begin{aligned} A &= 1 \\ C &= 01 \\ G &= 000 \\ T &= 001 \end{aligned} \tag{16}$$

Per cui la sequenza di caratteri

$$ACATGAAC \tag{17}$$

che ha una frequenza di apparizione univocamente determinata dalle probabilità precedentemente assegnate ad ogni simbolo, viene codificata come:

$$10110010001101. \tag{18}$$

14 cifre binarie vengono utilizzate per codificare 8 simboli, perciò la media è $14/8 = 1.75$ cifre binarie per simbolo. Questo metodo di codifica viene chiamato codice di Fano.

I codici di Fano hanno la proprietà che si possono decodificare senza bisogno di elementi separatori tra un simbolo e l'altro.

Usualmente occore conoscere la lunghezza del ≫frame di lettura≪, ma in questo esempio, come in tutti i codici di Fano, non è assolutamente necessario conoscerla.

In questo particolare tipo di codice, la prima cifra permette di distinguere l'insieme contenente A, (che abbiamo simboleggiato con A dall'insieme unione di C, G, T questi due insiemi sono equiprobabili, perché $\frac{1}{2} = \frac{1}{4} + \frac{1}{8} + \frac{1}{8}$.

La seconda cifra, che viene usata se la prima è 0, distingue C dall'unione di G e T; anche questi due insiemi sono equiprobabili.

L'ultima cifra distingue G da T.

Essendo ogni scelta equiprobabile (per come abbiamo assegnato le probabilità dei simboli all'inizio dell'esempio), ogni cifra binaria di questo codice porta con se esattamente 1 bit d'informazione.

Attenzione! Questo può non essere sempre vero.

Una cifra binaria può portare con se 1 bit **se e solo se** i due insiemi che essa rappresenta sono equiprobabili (come quelli che sono stati costruiti per questo esempio).

Se questi non sono equiprobabili una cifra binaria può benissimo portare con se meno di un bit d'informazione. (Ricordiamoci che H è massima quando le probabilità sono identiche.)

Quindi se le probabilità fossero:

$$P_A = \frac{1}{2}, \quad P_C = \frac{1}{6}, \quad P_G = \frac{1}{6}, \quad P_T = \frac{1}{6}, \tag{19}$$

non ci sarebbe modo di assegnare un codici finito in modo tale che una cifra binaria abbia il valore di un bit (solo usando grossi blocchi di simboli, si potrebbe approssimare tale comportamento).

Nell'esempio costruito, non c'è modo di utilizzare meno di 1.75 cifre binarie per simbolo, ma potremo rovinarci e utilizzare cifre in più per rappresentare il segnale.

I codici di Fano fanno questo ragionevolmente bene scomponendo l'interno insieme di simboli in sottoinsiemi successivi che siano equiprobabili, come è necessario che sia; si può trovare di più riguardo i codici di Fano sui testi di Teoria dell'Informazione.

L'incertezza misurata ci dice solo ciò che può essere fatto idealmente, a livello teorico; perciò, in ultima analisi, ci dice esattamente che cosa è impossibile fare.

Per esempio, il segnale con 1.75 bits per simbolo non potrà **mai** essere codificato utilizzando soltanto una cifra binaria per simbolo.

Radunando le Idee

All'inizio di questo brevissimo trattato abbiamo preso l'informazione come diminuzione di incertezza. Ora che abbiamo una formula generale per l'incertezza, (8), possiamo esprimere l'informazione usando questa formula. Supponiamo che un computer contenga qualche informazione nella sua memoria. Se andiamo a vedere i singoli flip-flop[2], ci troveremo ad avere una qualche incertezza para a H_{primo} bits per flip-flop. Supponiamo ora di eliminare parte della memoria di questo computer, quindi avremo una nuova incertezza, inferiore alla precedente: H_{dopo}. Allora il computer si ritrova ad aver perso una media di:

$$R = H_{before} - H_{after} \tag{20}$$

bits di informazione per flip-flop. Se invece eliminiamo tutta la memoria, allora $H_{dopo} = 0$ e $R = H_{prima}$.

Ora, consideriamo una telescrivente che riceva caratteri da una linea telefonica. Se non ci fossero disturbi sulla linea telefonica e nessun'altra fonte d'errore, la telescrivente stamperebbe il testo perfettamente. Con i disturbi di linea, si crea una qualche incertezza sul fatto che ciò che è stato stampato sia corretto o meno. Perciò prima che un carattere venga stampato, la telescrivente deve essere *preparata* a ricevere qualunque lettera dell'insieme delle lettere possibili, e questo stato di *pronta* ha una sua incertezza H_{prima}, mentre dopo avere ricevuto ogni lettera resta una incertezza H_{dopo}. Questa incertezza è basata sulla probabilità che il simbolo appena arrivato non sia uguale al simbolo trasmesso, ed essa misura la quantità di rumore (o disturbi di linea che dir si voglia). Shannon ha fatto un esempio di tutto ciò nel capitolo 12 di [10] (o nelle pagine 33 e 34 di [13].

Un sistema avente due simboli equiprobabli che trasmetta con un *clock* di un Hertz è in grado di inviare informazione sul mezzo trasmissivo ad una velocità di un bit al secondo senza errori.

Supponiamo che la probabilità di ricevere uno zero quando uno zero è stato trasmesso sia 0.99 e la probabilità di ricevere un 1, quando un 1 è stato trasmesso, sia 0.01. Questi numeri vengono invertiti se viene ricevuto un 1. Allora l'incertezza dopo aver ricevuto un simbolo è:

$H_{after} = -(0.99 \log_2 0.99 + 0.01 \log_2 0.01) = 0.081$

quindi l'attuale velocità di trasmissione

$R = 1 - 0.081 = 0.919$ bits al secondo[3]

[2] flip-flop = micro circuito elettronico capace di comutare tra due stati, ovvero: [tensione sotto una certa soglia = stato(0), e [tensione maggiore o uguale a quella certa soglia=stato(1)]

[3] Shannon ha usato la notazione $H_y(x)$ intendendola come incertezza condizionale del ricevitore *y* data dal messaggio inviato da *x*, che noi abbiamo chiamato H_{dopo}. Lui ha anche usato il termine *equivocazione.*

La quantità d'informazione che otteniamo è data dalla diminuzione di incertezza, equazione (20). Sfortunatamente molte persone hanno fatto errori soltanto perché non avevano appreso con chiarezza questo punto. Gli errori nascono perché si assume implicitamente che non ci siano disturbi nella comunicazione. Quando non ci sono disturbi, $R = H_{prima}$, così come nella memoria del computer completamente eliminata. Così è, *se non ci sono disturbi, la quantità di informazione comunicata eguaglia l'incertezza che si ha prima della comunicazione.* Quando i disturbi ci sono, ed implicitamente si assume che non ci siano, questo porta a tutto una serie di devianti filosofie. Bisogna **sempre** tener conto che i disturbi, nei mezzi trasmissivi (fisici e quindi reali), **ci sono!**

Una sottigliezza finale. In questo breve trattato si può trovare strano che venga usato il termine *flip-flop.* Tutto ciò perché la parola *bit* viene intenzionalmente evitata. La ragione di questo è che ci sono due significati di questa parola, come abbiamo accennato poco fa trattando i codici di Fano, ed è molto meglio tenerli distinti. I due significati della parola *bit* sono:

1. Una cifra binaria, 0 o 1. Che può solo essere un intero. Questi *bits* sono le unità elementari per la memorizzazione dell'informazione (dati) nei computers.

2. Una misura di incertezza, H, o di Informazione R. Questi possono essere numeri reali in quanto si tratta di una media. È la misura che Shannon ha usato per trattare di sistemi di comunicazione.

Riferimenti bibliografici

[1] T. D. Schneider, G. D. Stormo, L. Gold, and A. Ehrenfeucht. Information content of binding sites on nucleotide sequences. *J. Mol. Biol.*, 188:415–431, 1986. http://www.ccrnp.ncifcrf.gov/˜toms/paper/schneider1986/.

[2] T. D. Schneider. Information and entropy of patterns in genetic switches. In G. J. Erickson and C. R. Smith, editors, *Maximum-Entropy and Bayesian Methods in Science and Engineering*, pages 147–154, Dordrecht, The Netherlands, 1988. Kluwer Academic Publishers.

[3] T. D. Schneider and G. D. Stormo. Excess information at bacteriophage T7 genomic promoters detected by a random cloning technique. *Nucleic Acids Res.*, 17:659–674, 1989.

[4] T. D. Schneider and R. M. Stephens. Sequence logos: A new way to display consensus sequences. *Nucleic Acids Res.*, 18:6097–6100, 1990. http://www.ccrnp.ncifcrf.gov/˜toms/paper/logopaper/.

[5] N. D. Herman and T. D. Schneider. High information conservation implies that at least three proteins bind independently to F plasmid *incD* repeats. *J. Bacteriol.*, 174:3558–3560, 1992.

[6] P. P. Papp, D. K. Chattoraj, and T. D. Schneider. Information analysis of sequences that bind the replication initiator RepA. *J. Mol. Biol.*, 233:219–230, 1993.

[7] R. M. Stephens and T. D. Schneider. Features of spliceosome evolution and function inferred from an analysis of the information at human splice sites. *J. Mol. Biol.*, 228:1124–1136, 1992. http://www.ccrnp.ncifcrf.gov/˜toms/paper/splice/.

[8] T. D. Schneider. Sequence logos, machine/channel capacity, Maxwell's demon, and molecular computers: a review of the theory of molecular machines. *Nanotechnology*, 5:1–18, 1994. http://www.ccrnp.ncifcrf.gov/˜toms/paper/nano2/.

[9] P. K. Rogan and T. D. Schneider. Using information content and base frequencies to distinguish mutations from genetic polymorphisms in splice junction recognition sites. *Human Mutation*, 6:74–76, 1995. http://www.ccrnp.ncifcrf.gov/˜toms/paper/colonsplice/.

[10] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Tech. J.*, 27:379–423, 623–656, 1948. http://cm.bell-labs.com/cm/ms/what/shannonday/paper.html.

[11] J. R. Pierce. *An Introduction to Information Theory: Symbols, Signals and Noise*. Dover Publications, Inc., New York, 1980.

[12] W. Sacco, W. Copes, C. Sloyer, and R. Stark. *Information Theory: Saving Bits*. Janson Publications, Inc., Dedham, MA, 1988.

[13] N. J. A. Sloane and A. D. Wyner. *Claude Elwood Shannon: Collected Papers*. IEEE Press, Piscataway, NJ, 1993.

[14] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., N. Y., 1991.

[15] D. K. C. MacDonald. Information Theory and Its Applications to Taxonomy. *J. Applied Phys.*, 23:529–531, 1952.

[16] M. Tribus. *Thermostatics and Thermodynamics*. D. van Nostrand Company, Inc., Princeton, N. J., 1961.

Appendice: Un Tutorial sui Logaritmi

Capire la Funzione Log(x) (logaritmo di x)

Nell'operazione matematica dell'addizione noi solitamente prendiamo due numeri e li associamo per crearne un terzo:

$$1+1=2. \tag{21}$$

Possiamo ripetere questa operazione:

$$1+1+1=3. \tag{22}$$

La molteplicazione è l'operazione che estende questo concetto:

$$3\times 1=3. \tag{23}$$

Allo stesso modo, possiamo ripetere la moltiplicazione:

$$2\times 2=4. \tag{24}$$

e . . .

$$2\times 2\times 2=8. \tag{25}$$

L'estensione della moltiplicazione è l'elevamento a potenza:

$$2\times 2=2^2=4. \tag{26}$$

e . . .

$$2\times 2\times 2=2^3=8. \tag{27}$$

Questo si legge: ≫due elevato alla terza è otto≪. Essendo che l'elevamento a potenza *conta* semplicemente il numero di moltiplicazioni, gli esponenti si possono sommare:

$$2^2\times 2^3=2^{2+3}=2^5. \tag{28}$$

Il numero 2 si dice base della potenza. Se eleviamo l'esponente a un altro esponente i valori si moltiplicano:

$$\left(2^2\right)^3=2^2\times 2^2\times 2^2=2^{2+2+2}=2^{2\times 3}=2^6. \tag{29}$$

La funzione esponenziale $y = 2^x$ è mostrata in questo grafico [4]:

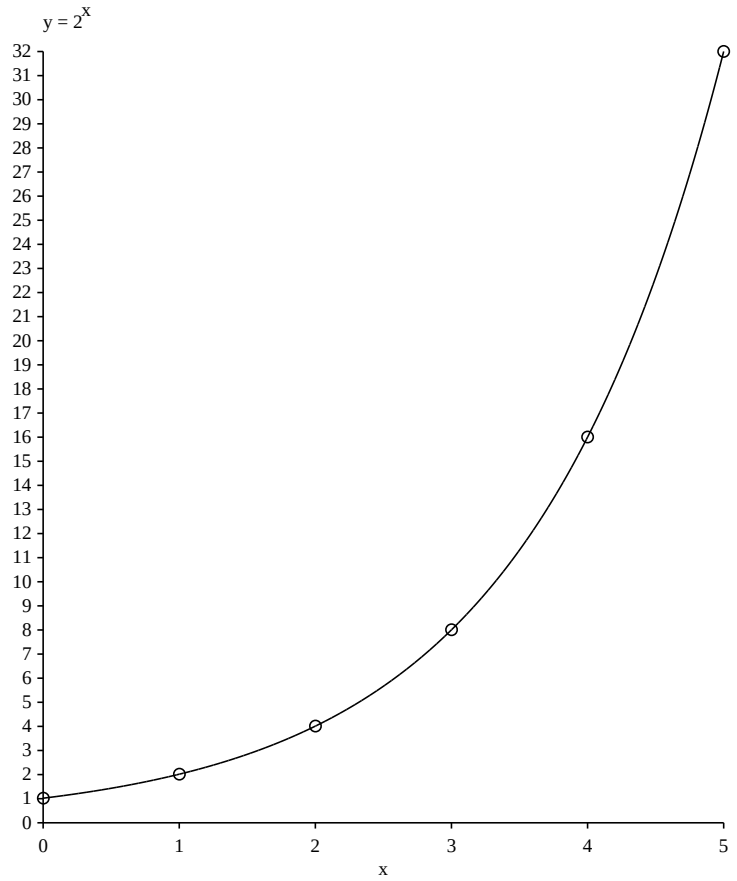

Figura 2: The exponential function.

Ora, pensiamo di avere un numero e di voler sapere quante volte dobbiamo moltiplicare un 2 per ottenere questo numero. Per esempio, poniamo di usare 2 come base, quanti 2 devono essere moltiplicati tra' loro per ottenere 32? Cioè vogliamo risolvere questa equazione:

$$2^B = 32. \tag{30}$$

Ovviamente, $2^5 = 32$, così $B = 5$. Per riuscire a far questo, i matematici hanno costruito una nuova funzione chiamata Logaritmo:

$$\log_2 32 = 5. \tag{31}$$

Che si legge »il logaritmo in base 2 di 32 è 5«. È la funzione inversa dell'elevamento a potenza:

$$2^{\log_2 a} = a \tag{32}$$

[4] Il programma usato per creare questo grafico è reperibile via ftp anonimo dal file: http://www.lecb.ncifcrf.gov/~toms/delila/expgraph.html gefunden werden

e . . .

$$\log_2(2^a) = a. \tag{33}$$

La funzione logaritmica $y = \log_2 x$ è mostrata in questo grafico[5]:

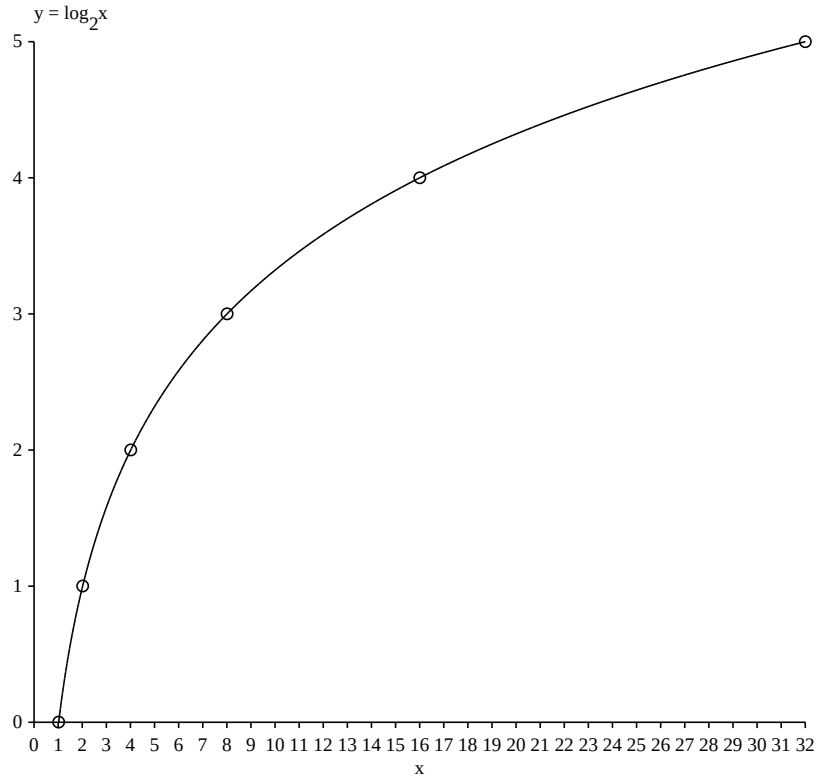

Figura 3: The logarithmic function.

Questo grafico è stato creato semplicemente scambiando la x con la y nel grafico esponenziale, che sarebbe come ruotare quest'ultimo su di un asse post a 45°. Da notare in particolare che $log_2 1 = 0$ und $log_2 0 = -\infty$.

La legge dell'Addizione

Consideriamo questa equazione:

$$\boxed{2^{a+b} = 2^a \times 2^b} \tag{34}$$

che è soltanto una generalizzazione dell'equazione (28). Prendiamo il logaritmo di entrambi i

[5]Il programma usato per creare questo grafico è reperibile via ftp anonimo dal file: http://www.lecb.ncifcrf.gov/∼toms/delila/expgraph.html gefunden werden

membri:

$$\log_2 2^{a+b} = \log_2 \left(2^a \times 2^b\right) \tag{35}$$

Elevamento a potenza e logaritmo sono l'una l'inversa dell'altra, perciò possiamo 'far crollare' la parte sinistra e così ottenere:

$$a+b = \log_2 \left(2^a \times 2^b\right) \tag{36}$$

Ora facciamo furbi, poniamo $\log_2 x = a$ und $\log_2 y = b$:

$$\log_2 x + \log_2 y = \log_2 \left(2^{\log_2 x} \times 2^{\log_2 y}\right) \tag{37}$$

Nuovamente, elevamento a potenza e logaritmo sono l'una l'inversa dell'altra, perciò possiamo 'far crollare' le due potenze nella parte destra:

$$\boxed{\log_2 x + \log_2 y = \log_2 (x \times y)} \tag{38}$$

Questa è la proprietà additiva a cui Shannon era interessato.

La regola del 'tirare in avanti'

Dall'equazione (32):

$$a = 2^{\log_2 a}. \tag{39}$$

Eleviamo entrambi i membri ad u:

$$a^u = \left(2^{\log_2 a}\right)^u. \tag{40}$$

Ora possiamo unire gli esponenti moltiplicandoli come abbiamo fatto in (29):

$$a^u = 2^{u \log_2 a}. \tag{41}$$

Infine, prendiamo il logaritmo in base 2 di entrambi i membri e facciamo 'crollare' la parte destra:

$$\boxed{\log_2 a^u = u \log_2 a} \tag{42}$$

Che può essere ricordate come la regola che permette di 'tirare' l'esponente avanti dall'interno del logaritmo.

Come convertire tra basi diverse Le calcolatrici e i computers solitamente non calcolano il logaritmo in base 2, ma possiamo usare una furbizia per convertire nella base desiderata (nel nostro caso la base 2) il risultato ottenendo in una base qualsiasi. Cominciamo ponendo:

$$x = \log_z a / \log_z b \tag{43}$$

Modifichiamola così:

$$\log_z a = x \log_z b. \tag{44}$$

Ora usiamo una 'tira avanti ribaltato' (!)

$$\log_z a = \log_z b^x \tag{45}$$

e lasciamo 'cadere' i logaritmi:

$$a = b^x. \tag{46}$$

Prendiamo ora la base dei logaritmi b:

$$\log_b a = \log_b b^x. \tag{47}$$

che si semplifica come:

$$\log_b a = x. \tag{48}$$

Ma noi sappiamo, dall'equazione (43) che x è:

$$\log_b a = \log_z a / \log_z b \tag{49}$$

La regola di conversione per ottenere il logaritmo in base 2 partendo da una qualsiasi base x è:

$$\boxed{\log_2(a) = \log_z(a) / \log_z(2)} \tag{50}$$

Notiamo che dal momento che la x non appare nella parte sinistra dell'equazione non importa che tipo di logaritmo abbiamo a disposizione, perché possiamo sempre ottenerlo in un'altra base usando questa equazione! Provate questo esempio sulla vostra calcolatrice:

$$\log_2(32) = \frac{\log_{\text{whatever!}}(32)}{\log_{\text{whatever!}}(2)}. \tag{51}$$

Otterete 5.

Trucchetti con le potenze di 2 Nei calcoli impariamo che la base naturale dei logaritmi è $e = 2.718281828459045\ldots$[6] I calcoli in questa baso possono essere fatti molto facilmente da un computer o calcolatrice, ma per molte persone risultano difficili da fare mentalmente. In contraddizione, le potenze di 2 sono facili da memorizzare e ricordare:

choices	bits
M	B
1	0
2	1
4	2
8	3
16	4
32	5
64	6
128	7
256	8
512	9
1024	10

dove $2^B = M$ e $\log_2 M = B$.

Possiamo utilizzare questa tabella ed un trucchetto per dare una veloce, sia pur approssimata, stima di logaritmi di numeri piuttosto alti. Notiamo che

$$2^{10} = 1024 \approx 1000 = 10^3. \tag{52}$$

Perciò per calcolare il logaritmo in base 2 di 4×10^6, procediamo così:

$$\begin{aligned}
\log_2(4 \times 10^6) &= \log_2(4) + \log_2(10^6) && (53)\\
&= 2 + \log_2(10^3 \times 10^3) && (54)\\
&= 2 + \log_2(10^3) + \log_2(10^3) && (55)\\
&\approx 2 + \log_2(2^{10}) + \log_2(2^{10}) && (56)\\
&\approx 2 + 10 + 10 && (57)\\
&\approx 22 && (58)
\end{aligned}$$

Il valore vero è 21.93!!

Simone Baldi, Martedì 12 Marzo 1996.

[6] Che impressione vi fa memorizzare questo numero? Notat che dopo il 2.7 abbiamo due 1828 seguiti da un triangolo 45°-90°-45°.