Codifica aritmetica

Non si basa sulla sostituzione di ciascun simbolo con una parola di codice, ma sulla conversione del flusso di simboli in un singolo valore (frazione a precisione arbitraria) in $[0, 1)$.

Codifica

Partendo da $I = [0, 1)$, per ogni simbolo in input si partiziona I in m sottointervalli di ampiezza proporzionale alla probabilità, e si assegna a I l'intervallo associato al simbolo letto. Terminato l'input, si restituisce il limite inferiore di I.

Per consentire la decodifica è necessario inserire un simbolo di EOF (a cui si assegna una probabilità bassa), oppure trasmettere anche la lunghezza del messaggio.

Decodifica

Dato il messaggio codificato $c \in [0, 1)$, si divide $I = [0, 1)$ in come nella codifica, e si restituisce il simbolo corrispondente al sottointervallo che contiene c, poi si ripete aggiornando I al sottointervallo trovato.

Ottimalità

- rappresentare un intervallo di dimensione s costa $O(-\log s)$ bit, quindi intervalli grandi (simboli probabili) richiedono pochi bit.
- la dimensione dell'intervallo finale è il prodotto delle probabilità dei simboli in input, quindi $-\log s = I(a_1) + \cdots + I(a_n)$
- la codifica è in generale subottima, ma dà risultati migliori di Huffman per distribuzioni asimmetriche (e.g. se $p_1 = 0.9$, $I(\sigma_1) = 0.152$ ma con Huffman richiede sempre un bit).

TODO probabilità sbilanciate?