Codifica di Huffman e Lempel-Ziv-Welch

ALBERTO BELUSSI
ANNO ACCADEMICO 2009/2010

Tipi di compressione

- **Senza perdita (lossless)**: permettono di ricostruire perfettamente la rappresentazione del dato originale, conservano tutta l'informazione (esempio, Huffman e Lempel-Ziv-Welch)
- **Con perdita (lossy)**: permettono di ricostruire solo in parte la rappresentazione del dato originale, quindi parte dell'informazione va persa.
  - Le tecniche lossy si dividono in:
    - Tecniche basate sulla predizione
    - Tecniche orientate alla frequenza (Discrete Cosine Transform - DCT; usata ad esempio nella codifica JPEG)
    - Tecniche orientate all'importanza

Codifica di Huffman

Note le caratteristiche della sorgente

- Es.
  - Alfabeto={A,B,C,D}
  - P(A)=0,50 I(A)=-log2(0,50)=1 bit
  - P(B)=0,20 I(B)=-log2(0,20)=2,322 bit
  - P(C)=0,18 I(C)=-log2(0,18)=2,474 bit
  - P(D)=0,12 I(D)=-log2(0,12)=3,059 bit

$$H_{sorgente} = 0{,}5 \times 1 + 0{,}2 \times 2{,}322 + 0{,}18 \times 2{,}474 + 0{,}12 \times 3{,}059 = \mathbf{1{,}777\ bit/simb}$$

Huffman

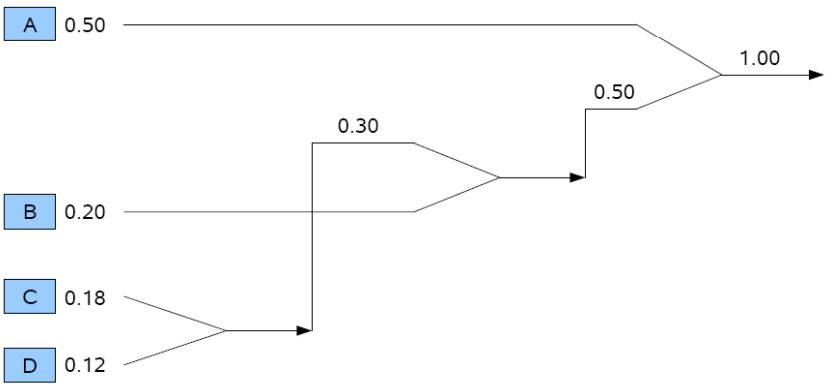
Si applica il seguente algoritmo:

1. Si ordinano i simboli in ordine **decrescente di** probabilità
2. Si unificano i due simboli in fondo alla lista, si sommano le probabilità e si reintroduce tale probabilità nella lista
3. Si ripete il passo precedente fino a quando la lista sarà formata da un solo elemento (di probabilità unitaria)

.... continua →

Huffman

5



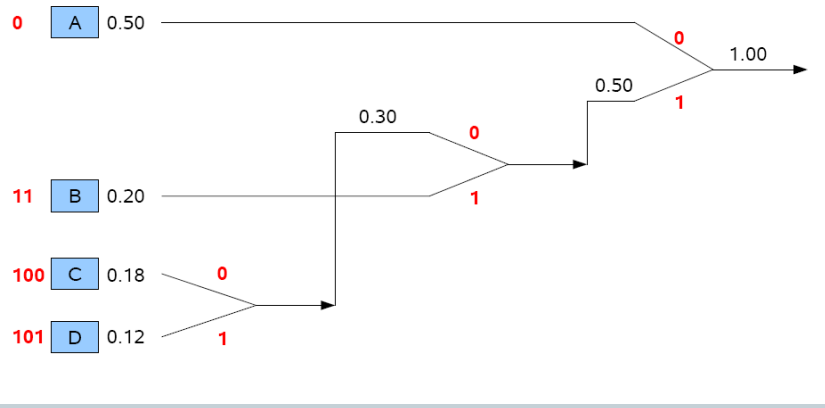
Huffman

6

4. Si etichetta ogni biforcazione con uno 0 e un 1
5. La codifica di ogni simbolo dell'alfabeto è data dal percorso dalla radice dell'albero fino alla foglia contenente il simbolo
6. Per le proprietà degli alberi, ogni foglia ha un unico percorso (codifica) che non sarà mai il prefisso di un altro percorso (la codifica di un simbolo non può essere l'inizio della codifica di un altro simbolo)

Huffman

7



Huffman

8

La codifica è immediata:

- es. AABABCABDACB di 12 simboli, sarà codificato con: 0 0 1 1 0 1 1 1 0 0 0 1 1 1 0 1 0 1 0 0 1 1

il numero totale di bit è 22 (in media 22/12=1.833 bit/simbolo); se fosse codificato senza tener conto delle probabilità (2 bit a simbolo) occuperebbe 24 bit.

Huffman

La **decodifica è univoca e mai ambigua.**

- es. 10010011011000 viene suddivisa senza ambiguità nei simboli
- 100 100 11 0 11 0 0 0
- C C B A B A A A
- Il primo bit di ogni simbolo coincide con la radice dell'albero; ogni bit successivo guida il decodificatore fino ad una foglia (simbolo corrispondente)

Huffman

p_k	$I(s_k)$	$p_k * I(s_k)$	L_k	$L_k * p_k$
0,500	1,000	0,500	1,000	0,500
0,200	2,322	0,464	2,000	0,400
0,180	2,474	0,445	3,000	0,540
0,120	3,059	0,367	3,000	0,360
1,000		1,777		1,800

Entropia: 1,777 bit/simbolo

Lunghezza media del codice (binit/simbolo)

Un messaggio di es. 1000 simboli avrà lunghezza media di 1800 binit (il minimo teorico è 1777)

E' possibile **avvicinarsi quanto si vuole all'entropia**, codificando coppie, terne ecc. di simboli dell'alfabeto (ciascuna con la sua probabilità)

Codifica LZW

11

Confronto con codifica Huffman

- Nella codifica di Huffman ogni simbolo ha una codifica a lunghezza variabile: più il simbolo ha probabilità bassa più la sua codifica è lunga
- Nella codifica LZW si codificano sequenze di simboli di lunghezza variabile con codici di lunghezza fissa

Codifica LZW

12

Algoritmo di codifica

- Si inizializza il dizionario dei simboli con tutti i simboli dell'alfabeto sorgente con la corrispondente codifica
- Si inizializza a "" la variabile PREFISSO
- Si legge il primo carattere dello STREAM di INPUT nella variabile C.

C

prefisso	""

input	ABABCBABABAAA

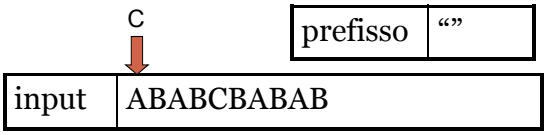
Seq. simboli	code
A	0000
B	0001
C	0010

Codifica LZW

13

Algoritmo di codifica

- Se la sequenza P+C è presente nel dizionario
  - allora P:=P+C;
  - altrimenti
    - output := output+code(P)
    - inserire la sequenza P+C nel dizionario e codificarla
    - P := C
- Se esistono ancora caratteri in INPUT leggere un carattere e tornare al punto precedente

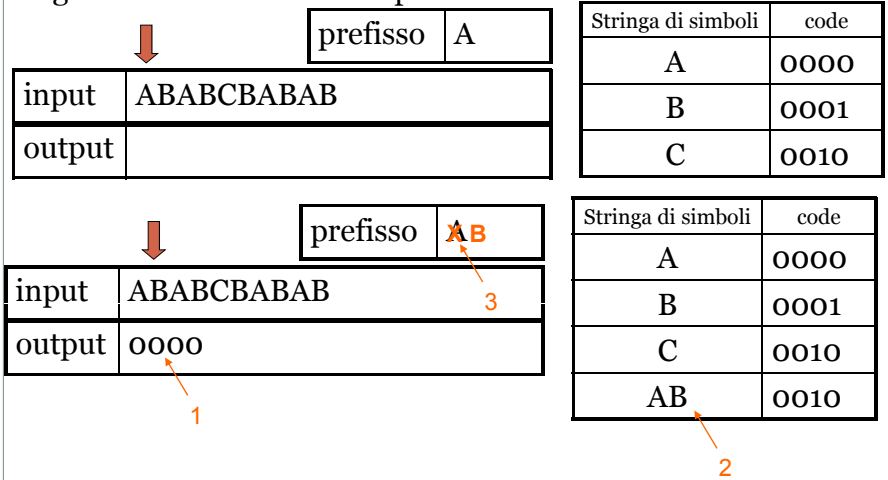


Stringa di simboli	code
A	0000
B	0001
C	0010

Codifica LZW

14

Algoritmo di codifica: esempio



Stringa di simboli	code
A	0000
B	0001
C	0010

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0010

2

Codifica LZW

15

Algoritmo di codifica: esempio

prefisso	B

input	ABABCBABAB
output	0000 0001

prefisso	AB

input	ABABCBABAB
output	0000 0001

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100

Codifica LZW

16

Algoritmo di codifica: esempio

prefisso	C

input	ABABCBABAB
output	0000 0001 0011

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100

A questo punto poiché “ABC” non è presente nel dizionario allora codifico AB, inserisco nel dizionario ABC e metto nella variabile *prefisso* C. Procedendo così ottengo alla fine:

output	0000 0001 0011 0010 0100 0111

Decodifica LZW

17

Algoritmo di decodifica

- Si inizializza il dizionario dei simboli con tutti i simboli dell'alfabeto sorgente con la corrispondente codifica
- Si inizializza a "" la variabile PREFISSO
- Si legge la prima posizione della codifica dello STREAM di INPUT nella variabile CW.
- Si supponga che esista una funzione DECODE(c) che dato un codice c restituisce la corrispondente stringa di simboli del dizionario.

PW CW

prefisso	""

input	0000 0001 0011 0010 0100 0101
output	

Stringa di simboli	code
A	0000
B	0001
C	0010

Decodifica LZW

18

1. Si produce in output la stringa di simboli DECODE(CW)
2. PW := CW; sposto CW sulla posizione successiva
3. Se CW è presente nel dizionario allora
   - Si produce in output la stringa di simboli DECODE(CW)
   - Prefisso := DECODE(PW)
   - Char := DECODE(CW)[1] (primo carattere della stringa decodificata)
   - Si aggiunge la stringa Prefisso+Char nel dizionario
4. Altrimenti
   - Prefisso := DECODE(PW)
   - Char := DECODE(PW)[1] (primo carattere della stringa decodificata al passo precedente)
   - Si produce in output Prefisso+Char e si aggiunge al dizionario
5. Se l'input non è terminato tornare al punto 2.

Decodifica LZW

PW CW

prefisso	“”

input	0000 0001 0011 0010 0100 0111
output	A

Stringa di simboli	code
A	0000
B	0001
C	0010

PW CW

input	0000 0001 0011 0010 0100 0111
output	AB
prefisso	A
char	B

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011

Decodifica LZW

PW CW

input	0000 0001 0011 0010 0100 0111
output	ABAB
prefisso	B
char	A

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100

Decodifica LZW

21

PW CW

input	0000 0001 0011 0010 0100 0111		
output	ABABC		
prefisso	AB	char	C

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100
ABC	0101

Decodifica LZW

22

PW CW

input	0000 0001 0011 0010 0100 0111		
output	ABABCBA		
prefisso	C	char	B

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100
ABC	0101
CB	0110

Decodifica LZW

PW CW

input	0000 0001 0011 0010 0100 0111
output	ABABCBABAB
prefisso	BA
char	B

Stringa di simboli	code
A	0000
B	0001
C	0010
AB	0011
BA	0100
ABC	0101
CB	0110