TECNICHE DI COMPRESSIONE DATI

COMPRESSIONE DATI

- La compressione produce una rappresentazione più compatta delle informazioni – è come se si usassero meno parole per dire la stessa cosa in modo diverso.

- Esistono diversi meccanismi di compressione:

Simmetrici — Senza perdite di info (*lossless*)

Asimmetrici — Con perdita di info (*lossy*)

RAPPORTO DI COMPRESSIONE

- Si definisce **rapporto di compressione** :

  file non compresso / file compresso

- La compressione riduce la dimensione di un blocco di informazioni e serve per memorizzare più informazioni nello stesso spazio.

- Un *compressore* è un programma che comprime i dati originali, mentre un *decompressore* li ricostruisce.

CONTENUTO INFORMATIVO

- Una misura del contenuto informativo di un messaggio o di un'immagine e' fornita dalla teoria dell'informazione
- Shannon defini il concetto di *self-information* con cui si puo' definire l'informazione associata all'accadere di un evento

CONTENUTO INFORMATIVO

• Supponiamo che un evento A abbia una probabilita' $P(A)$ di accadere, allora l'auto-informazione di A, $i(A)$ e' data da

$$i(A) = \log \frac{1}{p(A)} = -\log[p(A)]$$

• Scegliendo 2 come base del log la misura dell'informazione e' in bit

CONTENUTO INFORMATIVO

- ESEMPIO: Si consideri come evento il lancio di una moneta e l'uscita del simbolo testa o croce. Tali eventi hanno uguale probabilita' pari a ½.

- Il contenuto informativo dell'evento testa e' pari a

$$i(testa) = \log \frac{1}{p(testa)} = -\log[p(testa)] = -\log\left[\frac{1}{2}\right] = 1$$

CONTENUTO INFORMATIVO

- ESEMPIO: Si consideri un evento A con probabilita' molto piccola come 0.00005
- Il contenuto informativo dell'evento A sara' pari a

$$i(A) = \log \frac{1}{p(A)} = -\log[p(A)] = -\log[0.00005] = 19.931$$

CONTENUTO INFORMATIVO

- La definizione di auto-informazione puo' essere estesa per misurare il contenuto informativo di un messaggio formato da un insieme di N simboli
- Si definisce Entropia la seguente quantita' *H*

$$H = \sum_i p(s_i)\, i(s_i) = \sum_{i=1}^{N} p(s_i) \log \frac{1}{p(s_i)} = -\sum_{i=1}^{N} p(s_i) \log[p(s_i)]$$

CONTENUTO INFORMATIVO

$$H = \sum_i p(s_i)\, i(s_i) = \sum_{i=1}^{N} p(s_i) \log \frac{1}{p(s_i)} = -\sum_{i=1}^{N} p(s_i) \log[p(s_i)]$$

- Il messaggio e' considerato come una serie di eventi generati da una sorgente (che trasmette dei simboli).

- L'entropia e' definita come la somma della misura dell'informazione associata a ciascun simbolo moltiplicata per la rispettiva probabilita'.

CONTENUTO INFORMATIVO

$$H = \sum_i p(s_i)\, i(s_i) = \sum_{i=1}^{N} p(s_i) \log \frac{1}{p(s_i)} = -\sum_{i=1}^{N} p(s_i) \log[p(s_i)]$$

- Per massimizzare l'informazione di un messaggio in codice binario e' necessario trovare un metodo di codifica che descriva il messaggio con un numero medio di bit pari all'entropia della sorgente.

CONTENUTO INFORMATIVO

- I metodi di compressione basati su una stima accurata della probabilita' dei simboli (metodi non lossy che tendono a minimizzare l'entropia), non sfruttano in modo adeguato la ridondanza dell'informazione.

- I metodi di compressione di tipo lossy rinunciano a codificare in modo esatto il segnale eliminando l'informazione ridondante

METODI DI COMPRESSIONE NON LOSSY (SENZA PERDITA DI INFORMAZIONE)

METODI NON LOSSY

• Tali metodi sono adatti a qui casi in cui si desidera conservare l'informazione originale (file di testo, programmi applicativi, etc.):

Metodo di HUFFMANN

Codifica ARITMETICA

LZW (*Lempel-Ziv-Welch*)

RLC (*Run Lenght Coding*)

METODO DI HUFFMANN

- Il metodo di Huffmann crea un codice di lunghezza variabile in cui ai simboli di maggior frequenza relativa vengono assegnate configurazioni di bit piu' corte.

- Ai due simboli di minima frequenza si assegnano parole di codice di uguale lunghezza (0 e 1)

- In questo metodo si utilizzano delle informazioni di natura statistica (*frequenza* con cui si presentano determinate sottostringhe).

METODO DI HUFFMANN

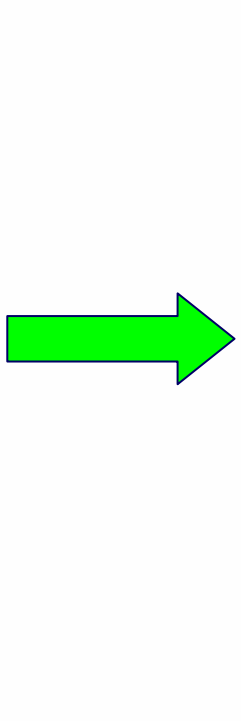
- La stima della probabilita' di occorrenza dei simboli e' il punto critico di questo metodo.

- Vediamo ora un esempio:

Supponiamo di avere un alfabeto di sorgente composto da 5 simboli che si vogliano codificare in formato binario (0 e 1)

METODO DI HUFFMANN

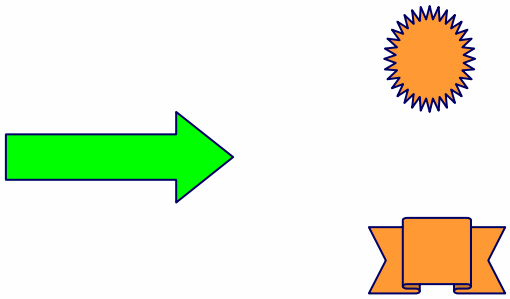
• I 5 simboli hanno le seguenti probabilita' note a priori



0.4	*a*
0.2	*b*
0.2	*c*
0.1	*d*
0.1	*e*

METODO DI HUFFMANN

• I simboli *d* ed *e* con minor probabilita' (pari in questo caso a 0.1) sono codificati la parola di codice cod



d *0*

e *1*

METODO DI HUFFMANN

- Dopo questa operazione l'alfabeto si e' ridotto a 4 elementi con le nuove probabilita'

0.4 *a*

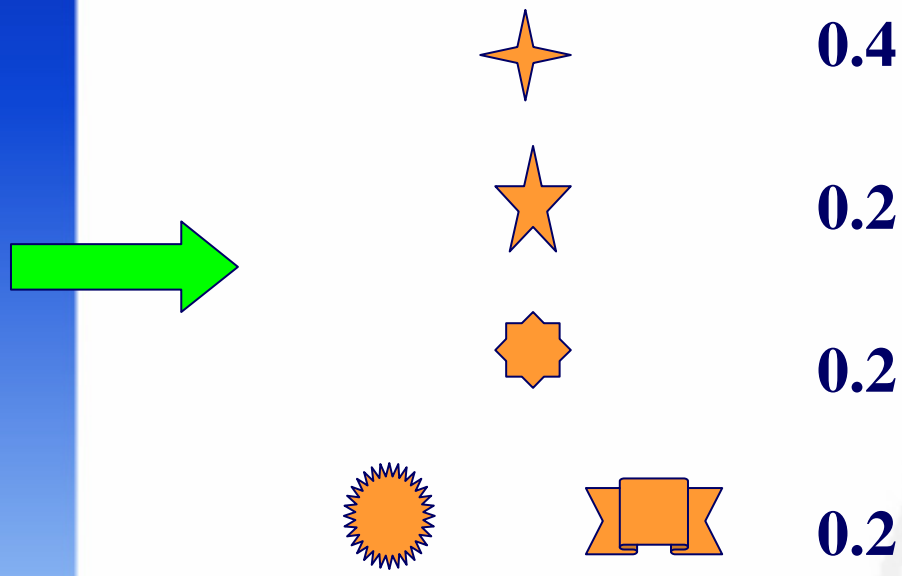
0.2 *b*

0.2 *c*

0.2 0 , 1 *d'*

METODO DI HUFFMANN

• Si ripete l'operazione precedente assegnando a *c* e *d'* il codice *0* e *1*. Per *d'* si ottiene *10 e 11*



METODO DI HUFFMANN

• Dopo questa operazione l'alfabeto si e' ridotto a 3 elementi con le nuove probabilita'

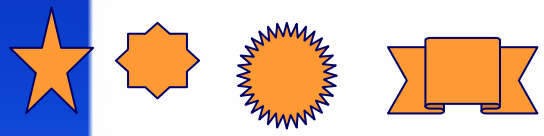
0.4 *a*

0.4 0 ,10, 11 *c'*

0.2 *b*

METODO DI HUFFMANN

- Si ripete l'operazione precedente assegnando a *c'* 0 e a b 1, ottenendo 2 simboli

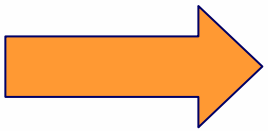


0.6 1 ,0 , 010, 011 *b'*

0.4 *a*

METODO DI HUFFMANN

• L'ultimo step consiste nell'assegnare il codice ai simboli *b'* e *a*, che essendo quelli a frequenza massima saranno composti da un solo simbolo, rispettivamente 0 che assegnamo a *b'* e 1 che assegnamo ad *a*.



METODO DI HUFFMANN

0.4

1

0.2

01

0.2

000

0.1

0010

0.1

0011

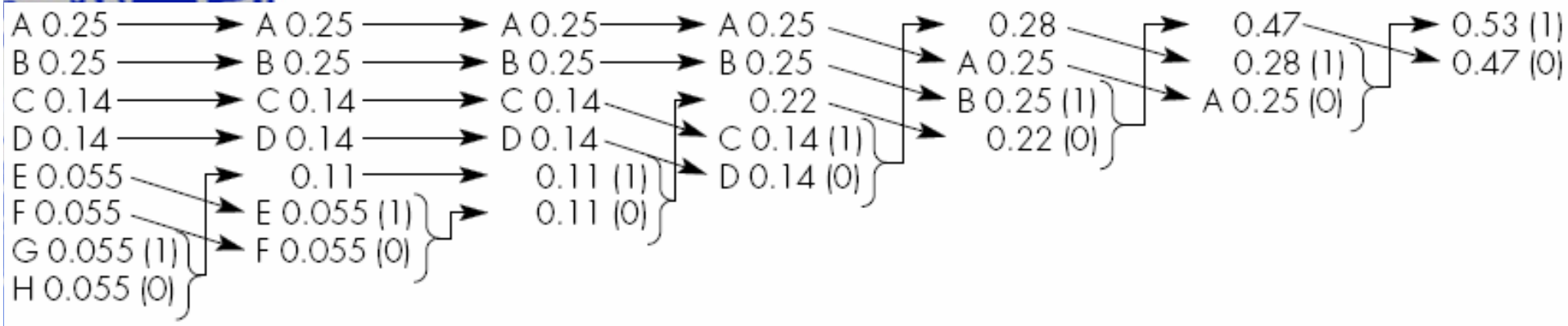
METODO DI HUFFMANN

- *ESERCIZIO* - Input A B C D E F G H

Sottostringa	Frequenza	Codifica
A	25 %	??
B	25 %	??
C	14 %	??
D	14 %	??
E	5,5 %	??
F	5.5 %	??
G	5.5 %	??
H	5.5 %	??

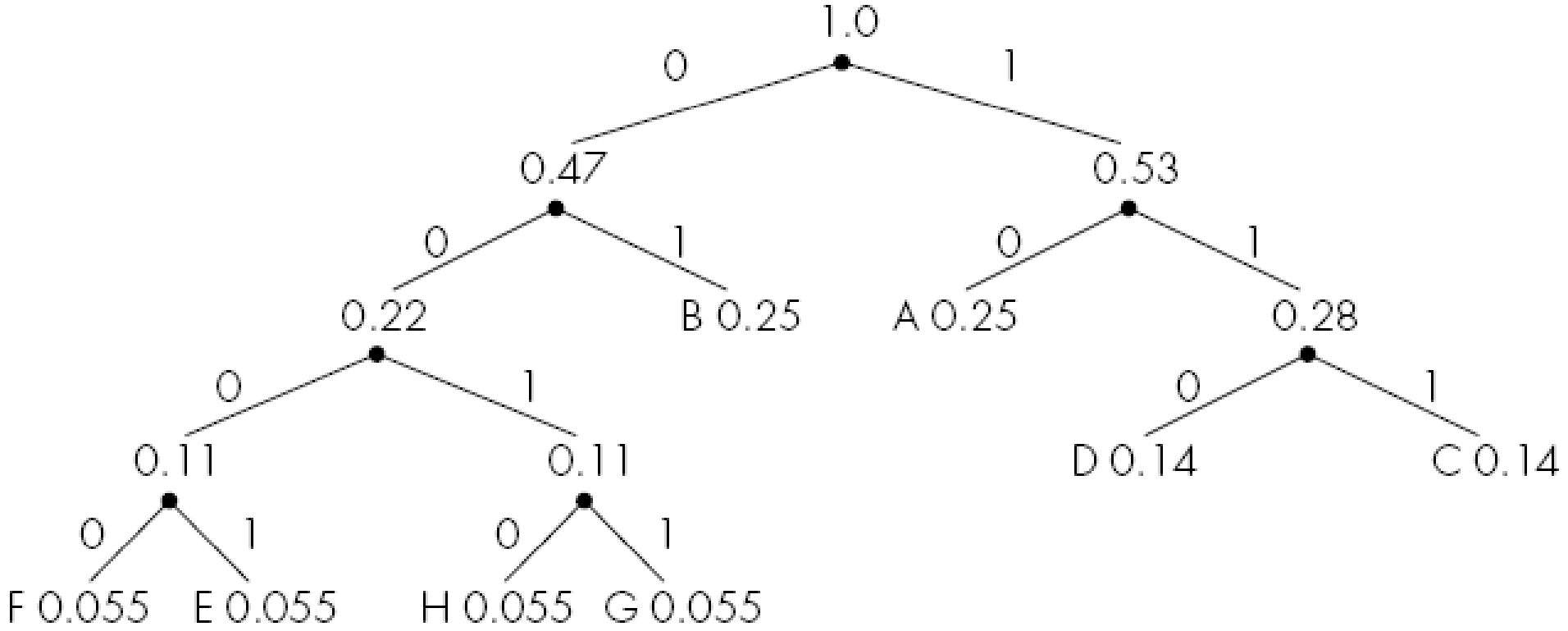
Output compresso: ?????

METODO DI HUFFMANN



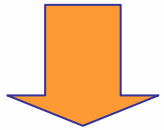
A = (0) (1) → 10
B = (1) (0) → 01
C = (1) (1) (1) → 111
D = (0) (1) (1) → 110
E = (1) (0) (0) (0) → 0001
F = (0) (0) (0) (0) → 0000
G = (1) (1) (0) (0) → 0011
H = (0) (1) (0) (0) → 0010

METODO DI HUFFMANN



Codifica Dinamica di Huffman

Metodo di Huffman

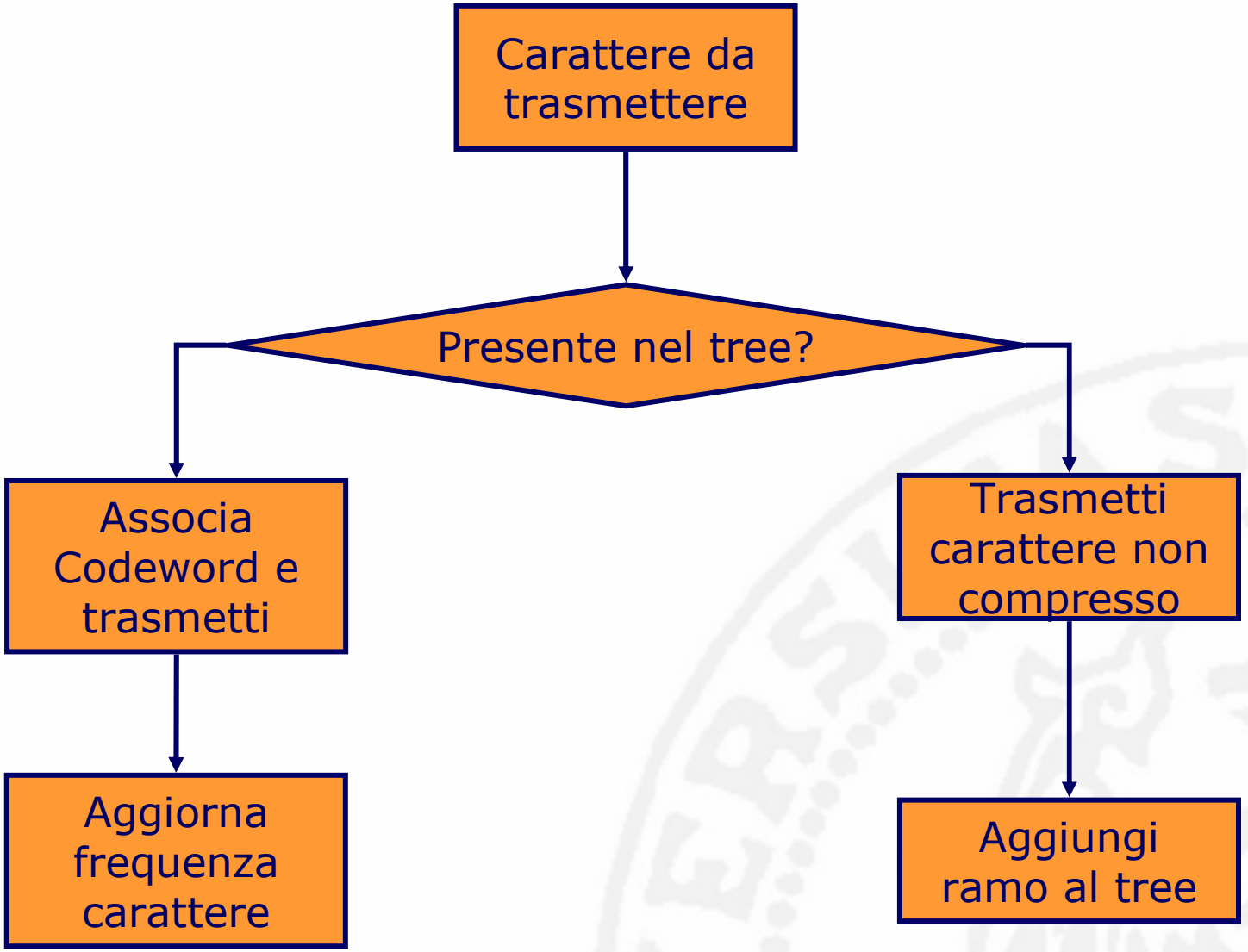


Codificatore/Decodificatore Tabella Codewords

Codifica Dinamica di Huffman

L’albero viene costruito on the fly

Codifica Dinamica di Huffman



CODIFICA ARITMETICA

• Il limite della codifica di Huffmann e' che approssima i valori di probabilita' (sono numeri reali) con numeri interi.

La codifica aritmetica usa numeri reali che rappresentano in modo esatto la probabilita' di occorrenza dei simboli

CODIFICA ARITMETICA

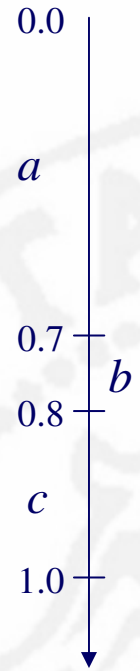
• La codifica aritmetica consiste nel generare un numero reale compreso tra 0 e 1 che rappresenta la funzione di probabilita' cumulativa della sequenza di simboli le cui probabilita' sono note a priori

Per calcolare tale numero il metodo divide l'intervallo [0,1] ricorsivamente individuando gli intervalli successivi a probabilita' crescente

CODIFICA ARITMETICA

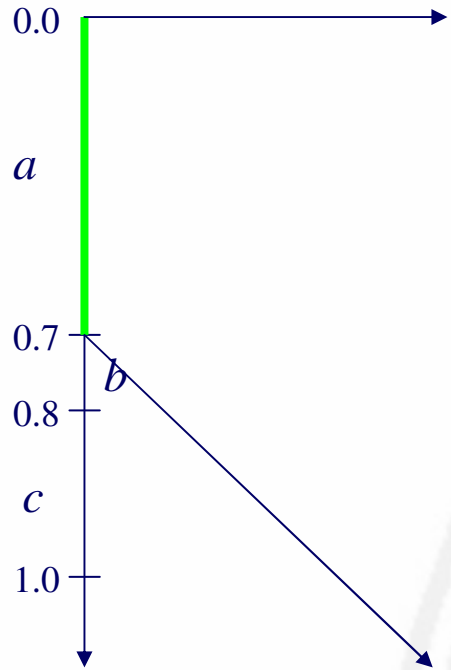
• Sia assegnata una sequenza di simboli *a, b, c* con probabilita' 0.7, 0.1, e 0.2

L'intervallo [0,1] viene suddiviso in tanti intervalli quanti sono i simboli e di ampiezza pari alla probabilità a priori dei simboli



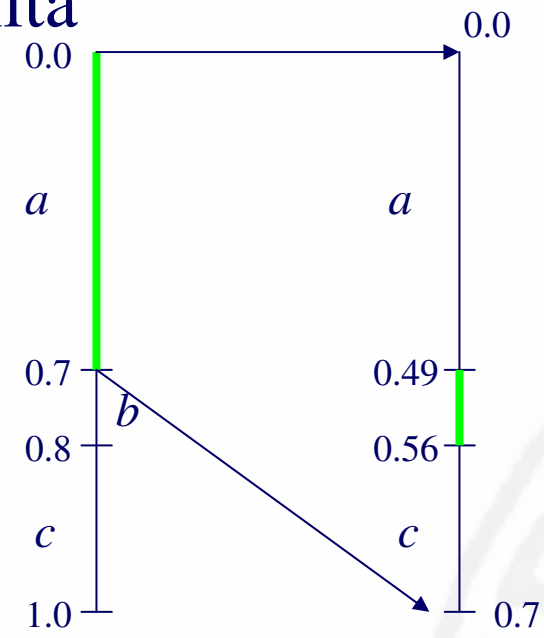
CODIFICA ARITMETICA

• Il primo simbolo determina il sottointervallo da considerare



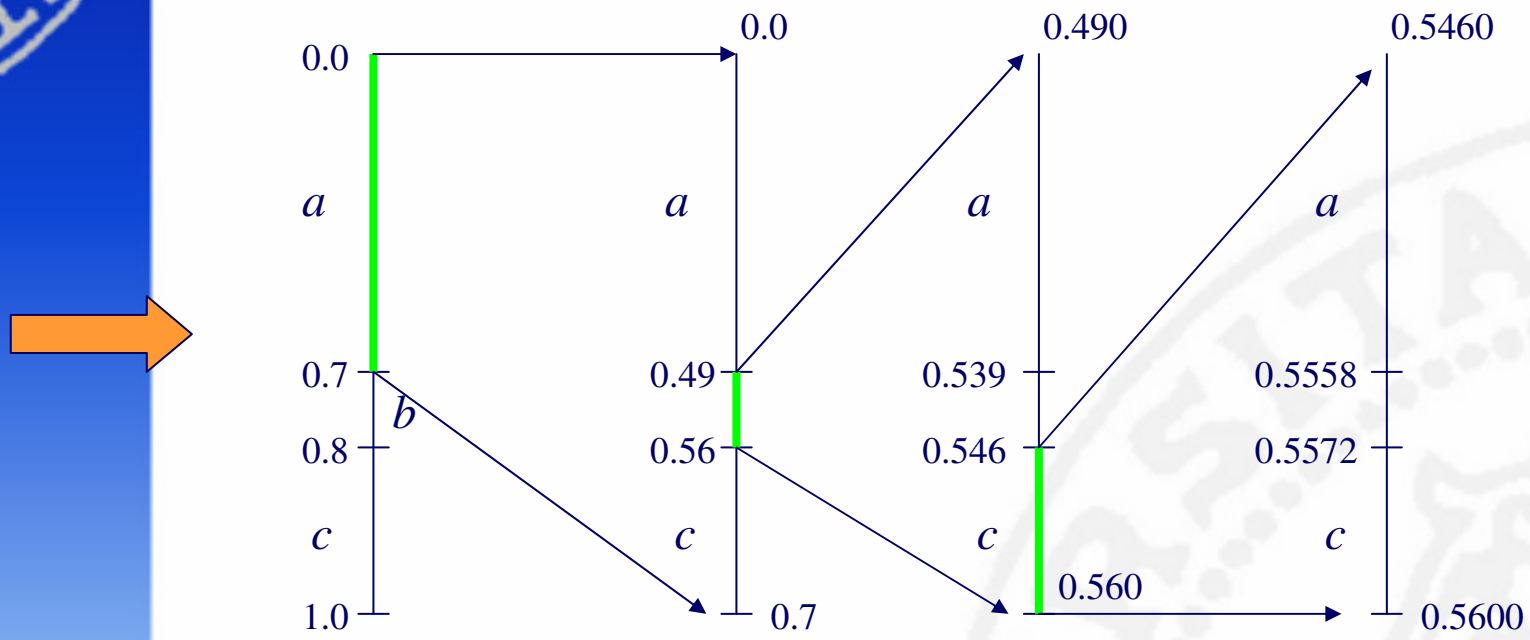
CODIFICA ARITMETICA

• Il sottointervallo selezionato diventa l'intervallo da suddividere in proporzione alla distribuzione di probabilita'



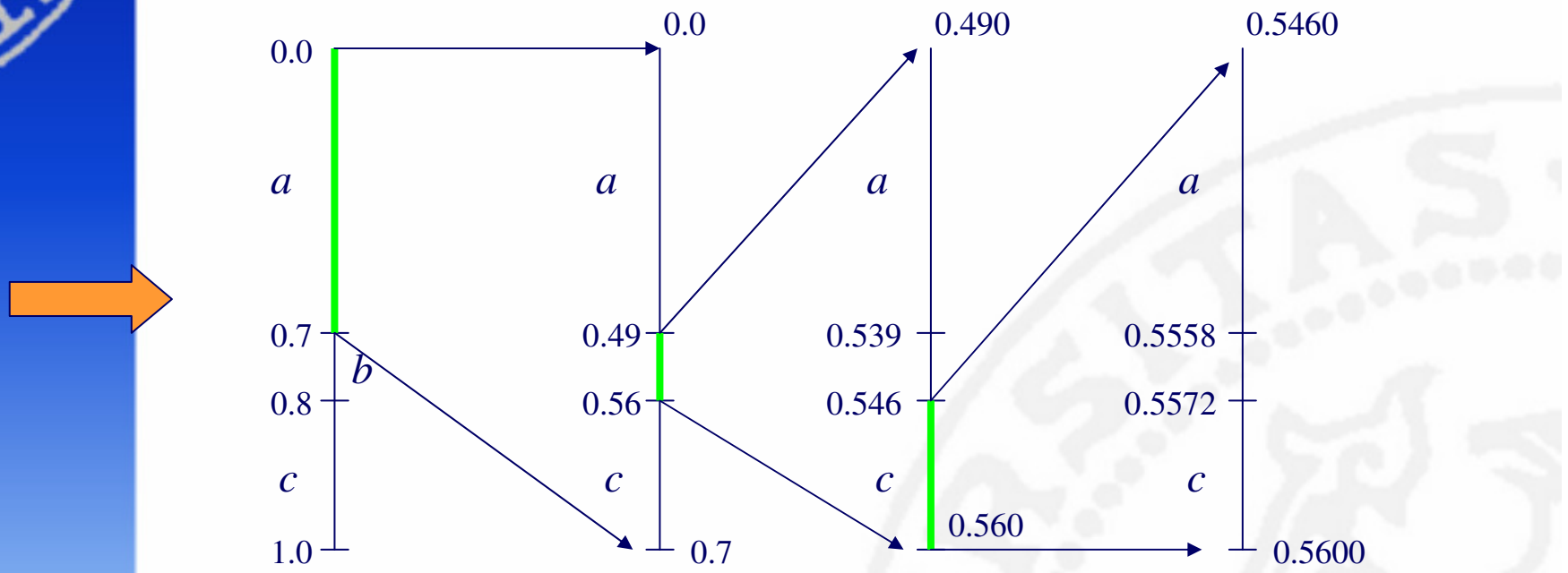
CODIFICA ARITMETICA (6)

- Si procede in questo modo fino al completamento della sequnza di simboli

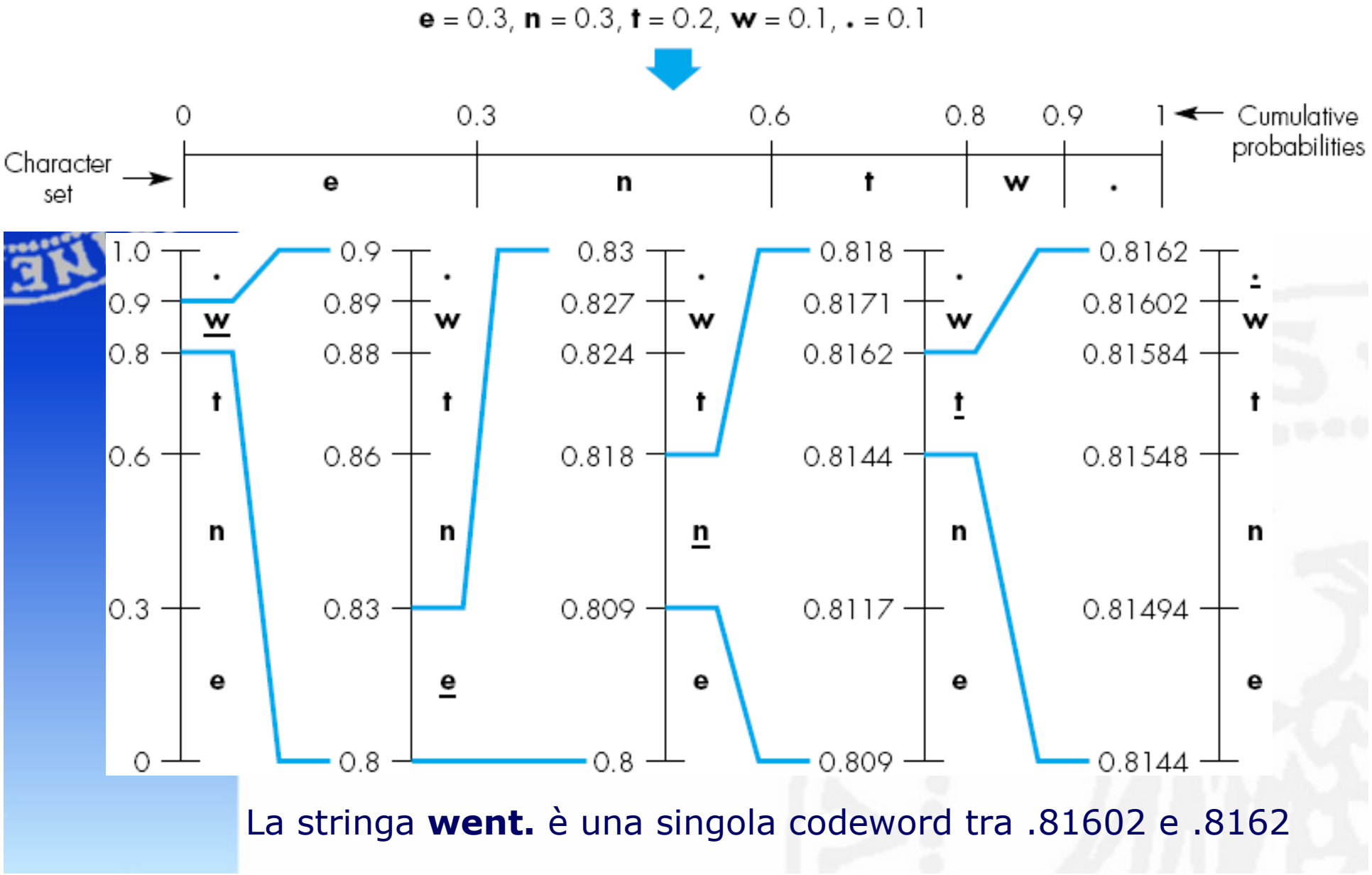


CODIFICA ARITMETICA

• Si sceglie come numero di codice l'estremo inferiore o il valor medio di ciascun intervallo



CODIFICA ARITMETICA



La stringa **went.** è una singola codeword tra .81602 e .8162

CODIFICA ARITMETICA

- La codifica aritmetica approssima con maggior accuratezza l'entropia della sorgente, anche se nella pratica il fattore di compressione che si puo' ottenere non e' molto migliore di quello della tecnica di Huffmann

TECNICHE A DIZIONARIO

- Con tali tecniche si comprimono file in cui determinate configurazioni di valori si ripetono frequentemente

E' piu' vantaggioso registrare l'indirizzo della posizione nel dizionario di una configurazione invece della configurazione stessa

TECNICHE A DIZIONARIO

- Esistono due tecniche principali :

SOSTITUZIONE STATICA

SOSTITUZIONE DINAMICA

SOSTITUZIONE STATICA

- Si basa su un dizionario indipendente dal file che deve essere compresso

- ESEMPIO – Un'analisi della frequenza relativa dei gruppi di lettere dell'italiano puo' far apparire che alcuni gruppi come qu, chi, che, …. appaiono con una data frequenza

SOSTITUZIONE STATICA (2)

- Le coppie o terne di lettere occupano da 16 o 24 bit

Possono essere sostituite con un puntatore ad una tabella che contiene I diversi gruppi di caratteri (file indice)

SOSTITUZIONE DINAMICA

- I metodi dinamici costruiscono il vocabolario durante l'esame del file che deve essere compresso o codificato.

→ Uno degli algoritmi piu' efficienti di questo tipo e' dovuto a Lempel, Ziv e Walsh

→ METODO LZW

METODO LZW

- È un metodo molto diffuso (è quello utilizzato dai programmi di compressione più comuni: pkzip, gzip, etc.).

- LZW lavora su tutti i tipi di dati, è veloce sia nella compressione che nella decompressione.

- LZW è considerato un metodo a sostituzione a dizionario, perchè costruisce un dizionario dei dati.

METODO LZW

• L'immagine può essere vista come una stringa di dati (*sequenza di dati*), all'interno di questa stringa vengono identificati dei pattern (*sottosequenze*) e ricercati nel dizionario.

• Se non li si trova, viene costruito un codice per quel pattern e viene aggiunto al dizionario.

• Se un pattern è già presente nel dizionario il suo codice viene scritto nell'output del file compresso.

METODO LZW

ESEMPIO - La stringa di input è

0101210101000000000

Prima fase: Identificazione dei pattern (si cerca il pattern più corto non ancora presente nel dizionario)

0 1 01 2 10 101 00 000 000

Seconda fase: Codifica della stringa

a b c d e f g h h

0	a
1	b
01	c
2	d
10	e
101	f
00	g
000	h

METODO LZW - Immagini

- LZW funziona su molte profondità di pixel, da 1 a 24.

- Le immagini molto ricche (di colori) o anche molto disturbate" (presenza di rumore) possono compromettere la sua efficienza di compressione.

- In questi casi il dizionario tende a crescere e quindi il rapporto di compressione che si ottiene è scadente.

METODI A PREDIZIONE

- Nei metodi fin qui considerati la conoscenza della distribuzione di probabilita' dei simboli gioca un ruolo determinante.

- Un modo per tener conto del contesto si basa sulla storia della sequenza di simboli

→ METODO RUN LENGTH CODING (RLC)

METODO RLC

- È un metodo che si adatta a qualsiasi tipo di dato ed è veloce da eseguire (anche se non raggiunge rapporti di compressione elevati)
- Una sequenza di caratteri (o codifiche di pixel) formata da più caratteri uguali viene detta run .
- Un run viene "codificato" con due byte

primo byte = numero di caratteri

secondo byte = carattere

METODO RLC

- La sequenza AAAAAAAAAAAAAAA (15 caratteri di tipo A) viene codificata come 15A.

- La sequenza **AAAAAAbbbXXXXXt** viene codificata come **6A3b5X1t**

METODO RLC

- I file di testo (ASCII) raramente contengono run lunghi (spesso contengono run di uno o due caratteri)

In questo caso la codifica RLC è inutile (anzi dannosa perchè produce un file più lungo).

- Un'immagine che contiene molti dati contigui dello stesso colore (nero) può essere compressa bene.

METODI DI COMPRESSIONE LOSSY (CON PERDITA DI INFORMAZIONE)

METODI DI CODIFICA LOSSY

• L'immagine digitale dopo il processo di campionamento e quantizzazione subisce numerose modifiche e l'introduzione di artefatti.

L'approccio lossy cerca di eliminare in fase di codifica l'informazione ridondante.

METODI DI CODIFICA LOSSY (2)

• I principlai metodi di codifica di tipo lossy si possono identificare in :

INTERPOLAZIONE

QUANTIZZAZIONE

COMPRESSIONE FRATTALE

TRASFORMATA DEL COSENO DISCRETA (DCT)

INTERPOLAZIONE

• La piu' semplice tecnica di compressione di tipo lossy si basa sulle caratteristiche del sistema visivo umano

La limitata sensibilità a brusche differenze di colore e la maggiore sensibilita' alle variazioni di luminosita'

INTERPOLAZIONE

- L'immagine a colori viene convertita dallo spazio RGB a quello YUV (Y componente di luminanza e UV componenti di crominanza)

$$Y = 0{,}3R + 0{,}59G + 0{,}11B$$

$$U = 0.493\,(B-Y)$$

$$V = 0.877\,(R-Y)$$

INTERPOLAZIONE

• Nello spazio YUV l'immagine può essere campionata con una risoluzione spaziale piu' ridotta per le componenti di cromaticita'.

ESEMPI

INTERPOLAZIONE

Originale

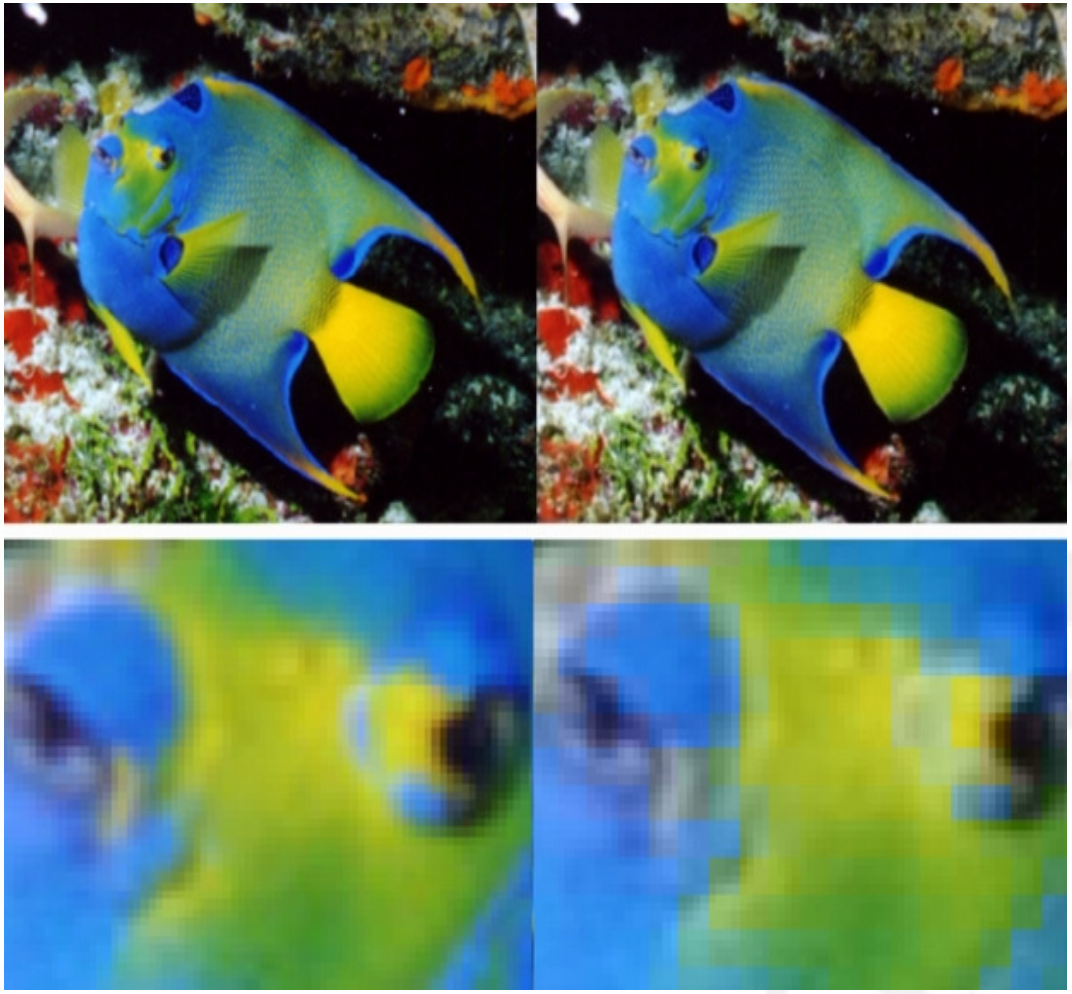


Immagine con componenti

U e V sottocampionate di ¼

INTERPOLAZIONE

Originale

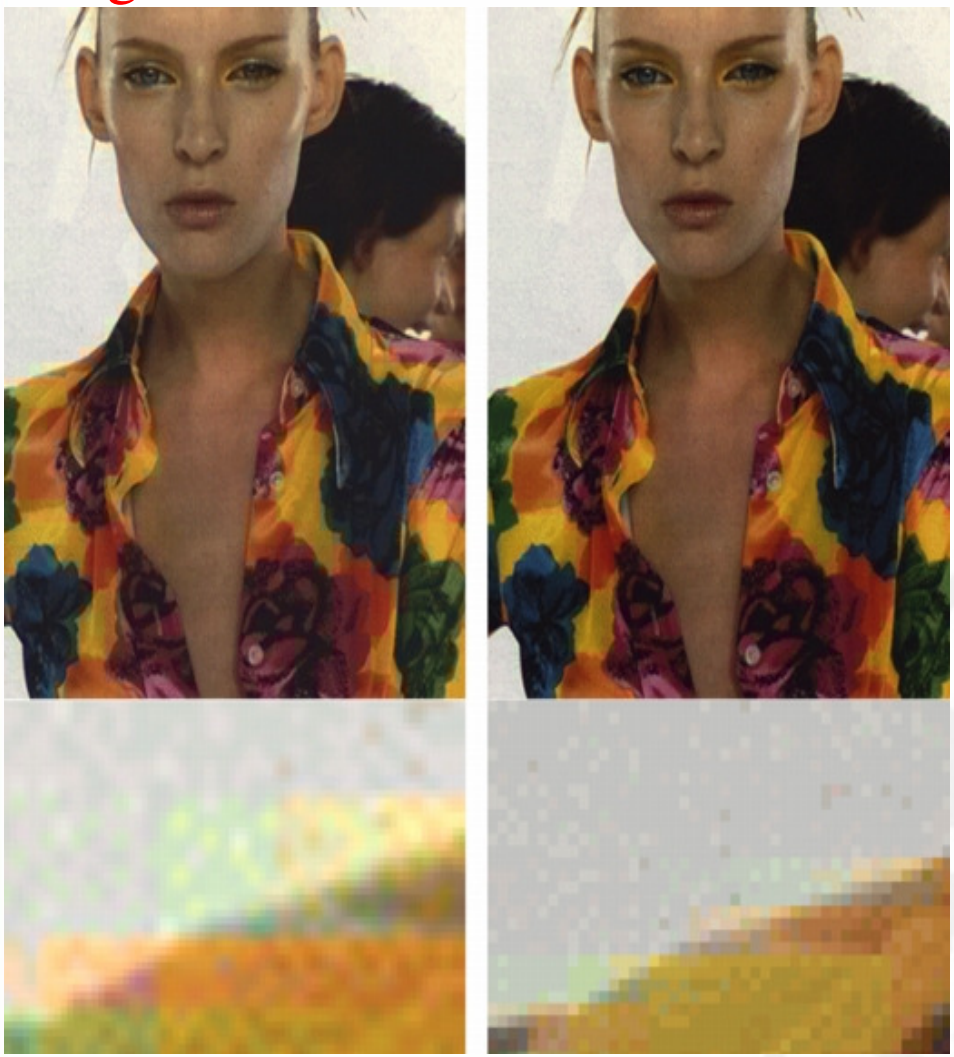


Immagine con componenti U e V sottocampionate di ¼

QUANTIZZAZIONE

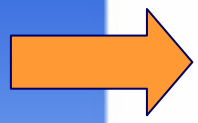
- Esistono due tipo di quantizzazione

QUANTIZZAZIONE SCALARE

QUANTIZZAZIONE VETTORIALE

QUANTIZZAZIONE SCALARE

• La quantizzazione scalare consiste nel determinare la conversione tra un intervallo di valori (continui e rappresentati da interi) molto elevato ad un intervallo di valori piu' limitato composto da numeri reali e non continui.



Il processo di digitalizzazione di un'immagine effettua una operazione di quantizzazione scalare

QUANTIZZAZIONE SCALARE

- La quantizzazione scalare di un'immagine opera una compressione in quanto riducendo l'intervallo dei valori possibili permette di rappresentarli con parole di codice di lunghezza piu' breve.

256 livelli rappresentabili con parole di 8 bit

16 livelli di un'immagine piu' quantizzata rappresentabili con parole di 4 bit

QUANTIZZAZIONE VETTORIALE

- La quantizzazione scalare considera i valori singolarmente

Un metodo di quantizzazione che consideri blocchi di valori potrebbe dare risultati migliori

Quantizzazione vettoriale considera gruppi di singoli in blocchi (*vettori*).

QUANTIZZAZIONE VETTORIALE

Un' immagine viene suddivisa in gruppi di *L* pixels

Ciascun gruppo puo' essere visto come un vettore di *L* dimensioni, le cui componenti sono i pixel di ogni blocco

QUANTIZZAZIONE VETTORIALE

SUDDIVISIONE IN VETTORI

RICERCA VETTORE SIMILE

CODEWORD

INDICE

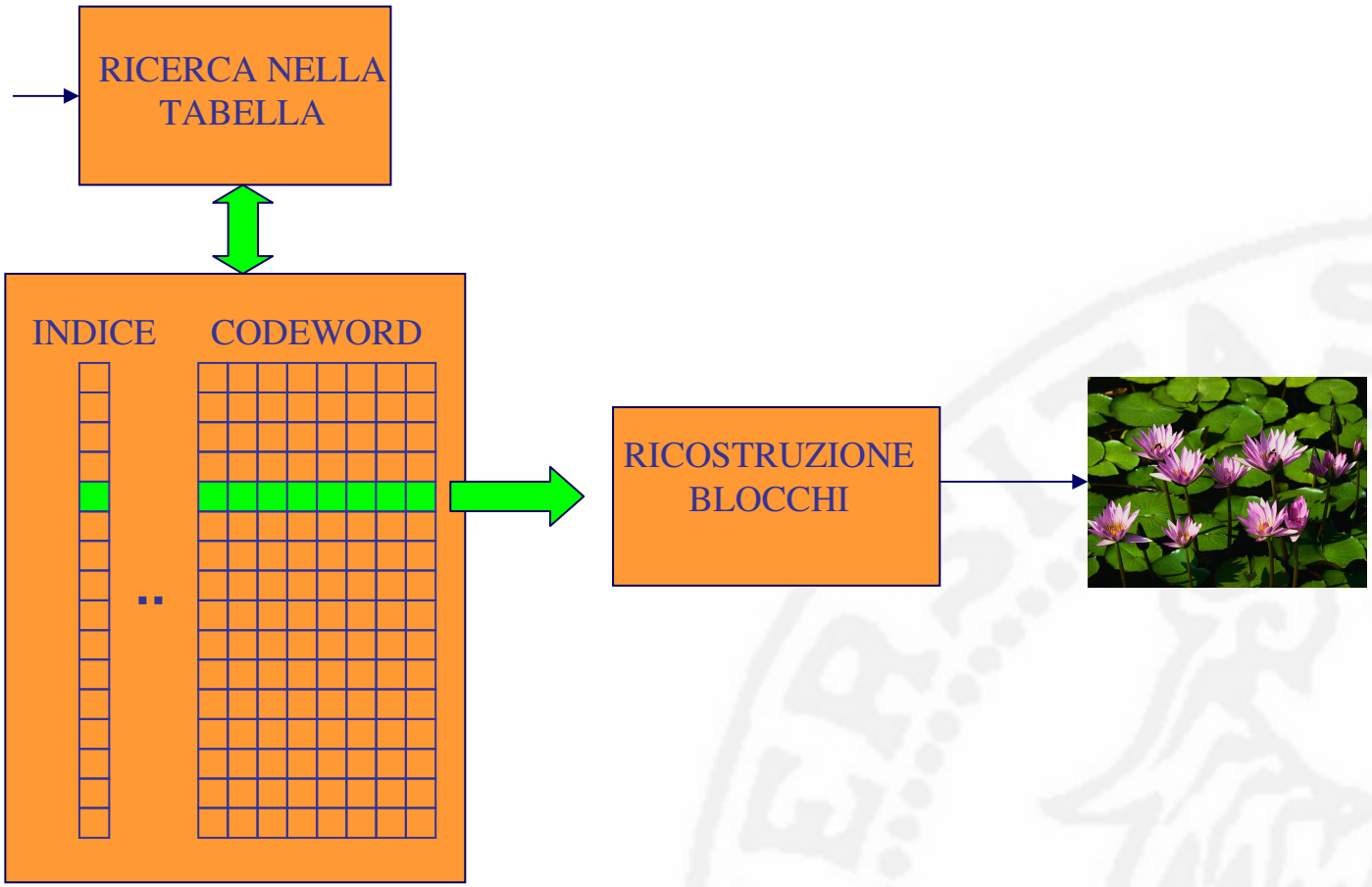
Vettori di L dimensioni

Trasmissione indice

QUANTIZZAZIONE VETTORIALE (4)

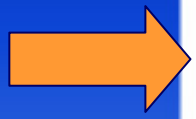
Ricezione

indice



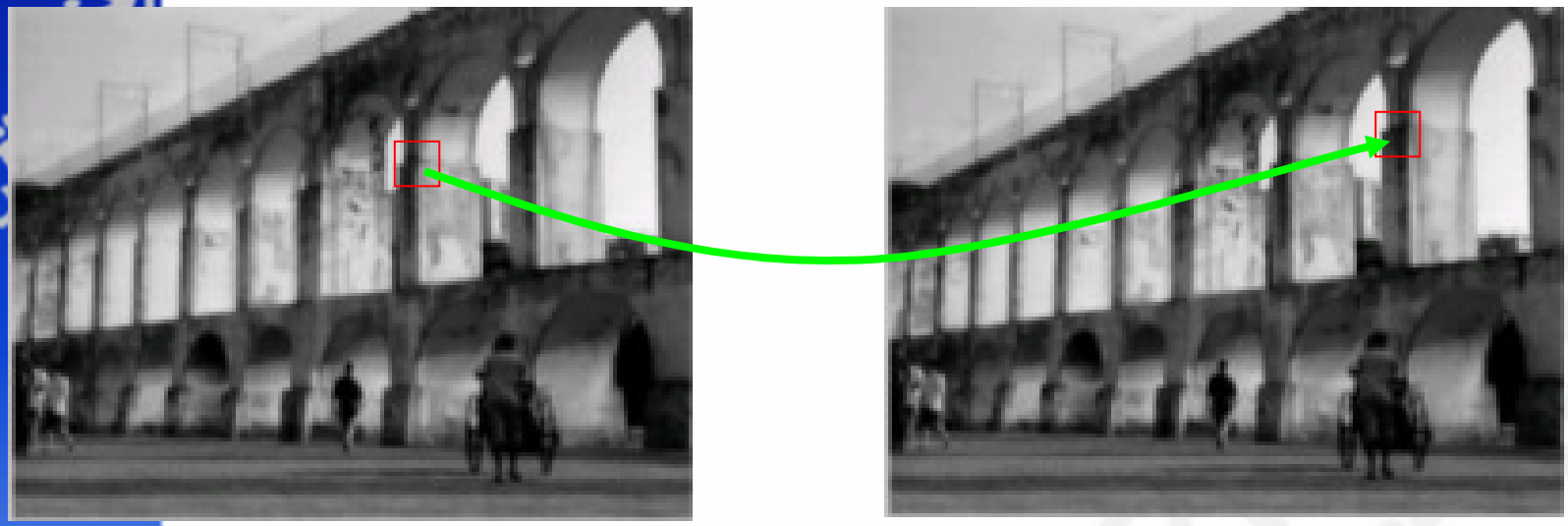
COMPRESSIONE FRATTALE

- La compressione frattale e' un metodo simile alla quantizzazione vettoriale, ma usa principi matematici differenti



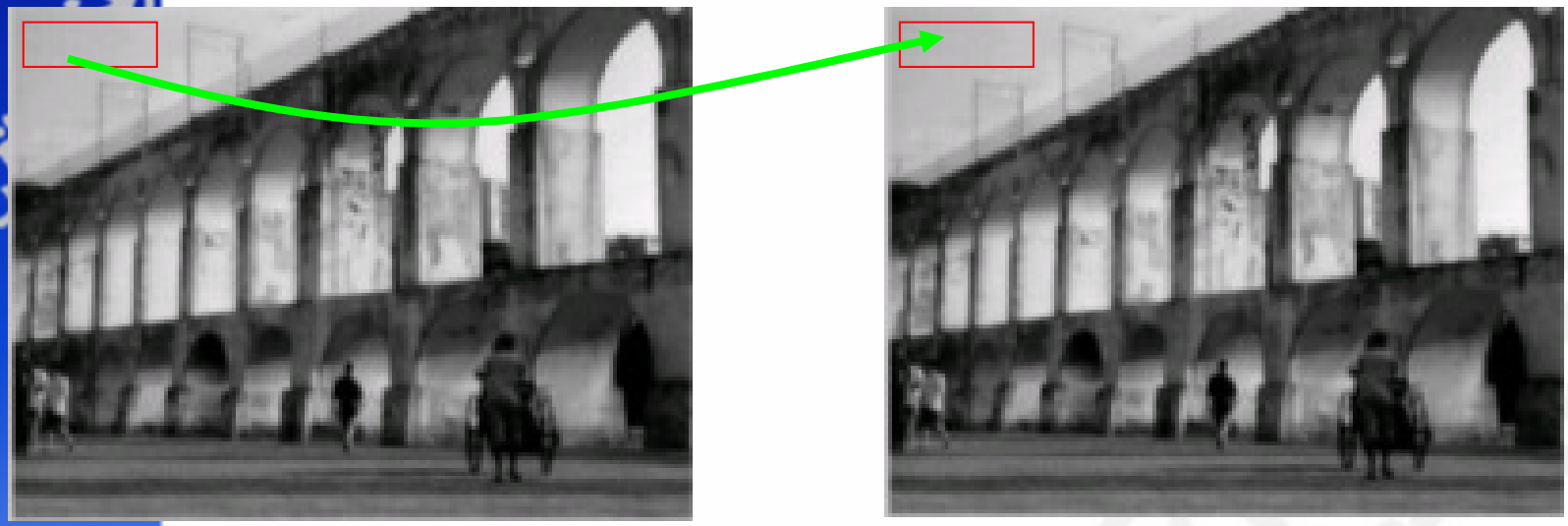
L'idea alla base del metodo e' che l'immagine contiene regioni simili tra loro a meno di trasformazioni geometriche o di colore

COMPRESSIONE FRATTALE



Somiglianza a meno di una trasformazione geometrica di traslazione

COMPRESSIONE FRATTALE



Somiglianza a meno di una
trasformazione di livello di grigio

COMPRESSIONE FRATTALE

• La compressione frattale consiste nel cercare queste regioni simili e nel codificare gli indici che le individuano ed i coefficienti delle trasformazioni che permettono di passare da una all'altra.

Sia le trasformazioni geometriche sia i criteri di somiglianza tra regioni introducono delle approssimazioni

METODI BASATI SU TRASFORMATE

- I metodi fin qui analizzati operano su immagini rappresentate nel dominio spaziale (valori di pixel distribuiti sul piano immagine)

Esistono dei metodi che applicano all'immagine una trasformazione nello spazio delle frequenze

TRASFORMATA DI FOURIER

METODI BASATI SU TRASFORMATE (2)

- Nel dominio delle frequenze spaziali, l'informazione ridondante in un'immagine (non rilevabile dal sistema visivo umano) e' legata alle alte frequenze

ALTE FREQUENZE

Brusche variazioni di luminosita' e contrasto

TRASFORMATA DEL COSENO DISCRETO (DCT)

• La trasformata del coseno discreta (Discrete Cosine Transform o DCT) rappresenta le componenti frequenziali di un'immagine come combinazione della funzione coseno

Studi statistici hanno dimostrato che la probabilita' che i pixel di una regione di 8x8 elementi siano simili tra loro e' massima

TRASFORMATA DCT

- Tale proprieta' ha indotto a proporre di applicare la trasformata a blocchi di 8x8 pixel dell'immagine originale

Rendere MINIMA la probabilita' di eliminare informazione rilevante

TRASFORMATA DCT

• La trasformata 2D del coseno di un'immagine di 8x8 elementi e' data da:

$$F(i,j) = \alpha_i \alpha_j \sum_{i=0}^{7} \sum_{j=0}^{7} P(x,y) \cos \frac{\pi(2x+1)i}{16} \cos \frac{\pi(2y+1)j}{16}$$

$$\alpha_i = \begin{cases} 1/\sqrt{32} & u = 0 \\ 1/4 & 1 \le u \le 7 \end{cases}$$

$$\alpha_j = \begin{cases} 1/\sqrt{32} & v = 0 \\ 1/4 & 1 \le v \le 7 \end{cases}$$

TRASFORMATA DCT

$$F(i,j) = \alpha_i \alpha_j \sum_{i=0}^{7} \sum_{j=0}^{7} P(x,y) \cos \frac{\pi(2x+1)i}{16} \cos \frac{\pi(2y+1)j}{16}$$

I valori *F(i,j)* sono i 64 coefficienti della DCT applicata all'immagine *P(x,y)*

TRASFORMATA DCT

• Questa trasformazione puo' essere invertita per ricostruire l'immagine a partire dai coefficienti della trasformata

$$P(x,y)=\sum_{i=0}^{7}\sum_{j=0}^{7}\alpha_i\alpha_j F(i,j)\cos\frac{\pi(2x+1)i}{16}\cos\frac{\pi(2y+1)j}{16}$$

$$\alpha_i=\begin{cases}1/\sqrt{32} & u=0\\ 1/4 & 1\le u\le 7\end{cases}$$

$$\alpha_j=\begin{cases}1/\sqrt{32} & v=0\\ 1/4 & 1\le v\le 7\end{cases}$$

TRASFORMATA DCT

$$P(x,y) = \sum_{i=0}^{7}\sum_{j=0}^{7} \alpha_i \alpha_j F(i,j) \cos\frac{\pi(2x+1)i}{16} \cos\frac{\pi(2y+1)j}{16}$$

- Poiche' i termini del coseno non dipendono dall'immagine (dai valori dei pixel), possono essere calcolati off-line creando una matrice di 8x8 elementi

TRASFORMATA DCT

Un'immagine si puo' ottenere come somma pesata di 64 funzioni

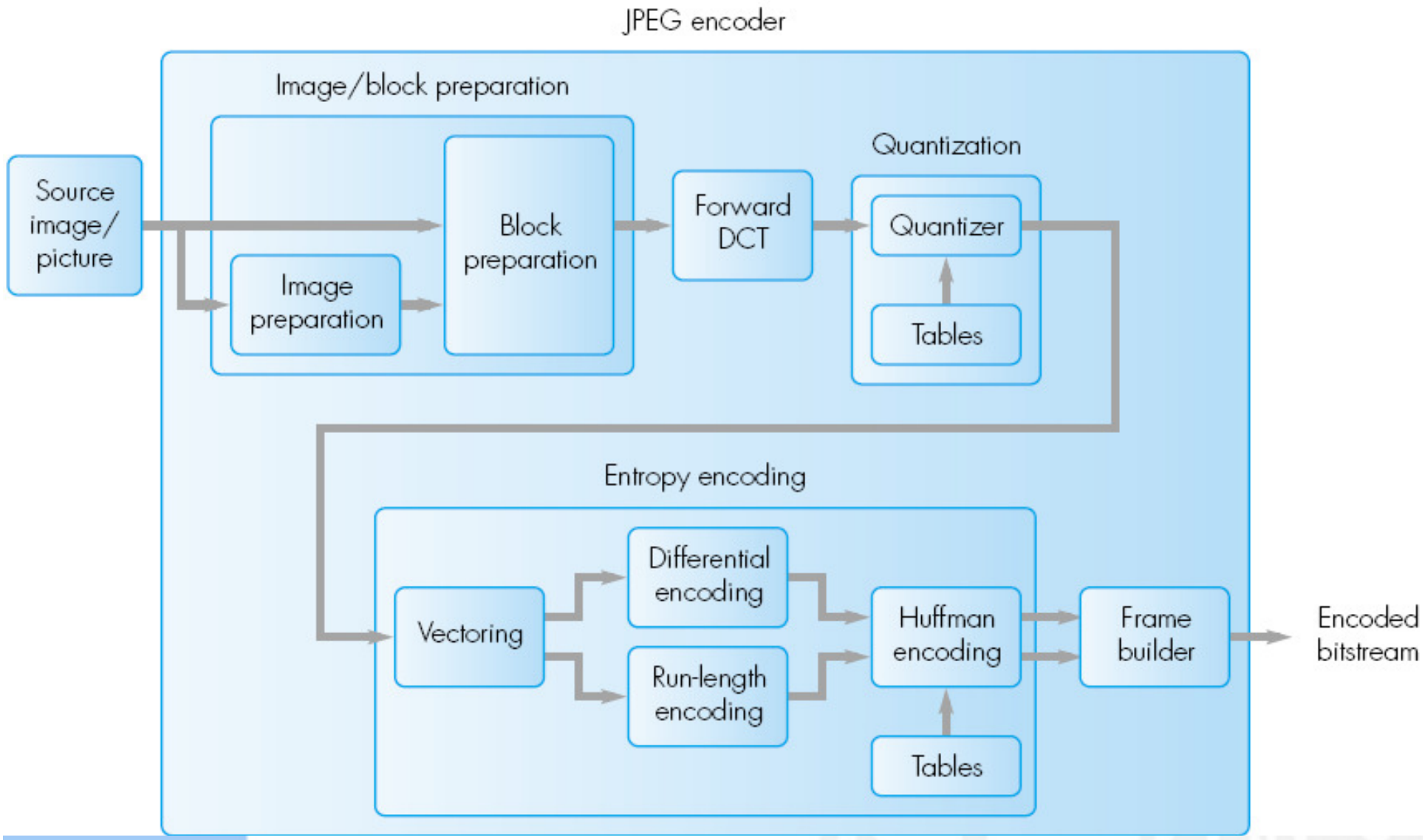
$$\alpha_i \alpha_j \cos\frac{\pi(2x+1)i}{16} \cos\frac{\pi(2y+1)j}{16}$$

ed i pesi sono dati dai coefficienti *F(i,j)* della trasformata dell'immagine

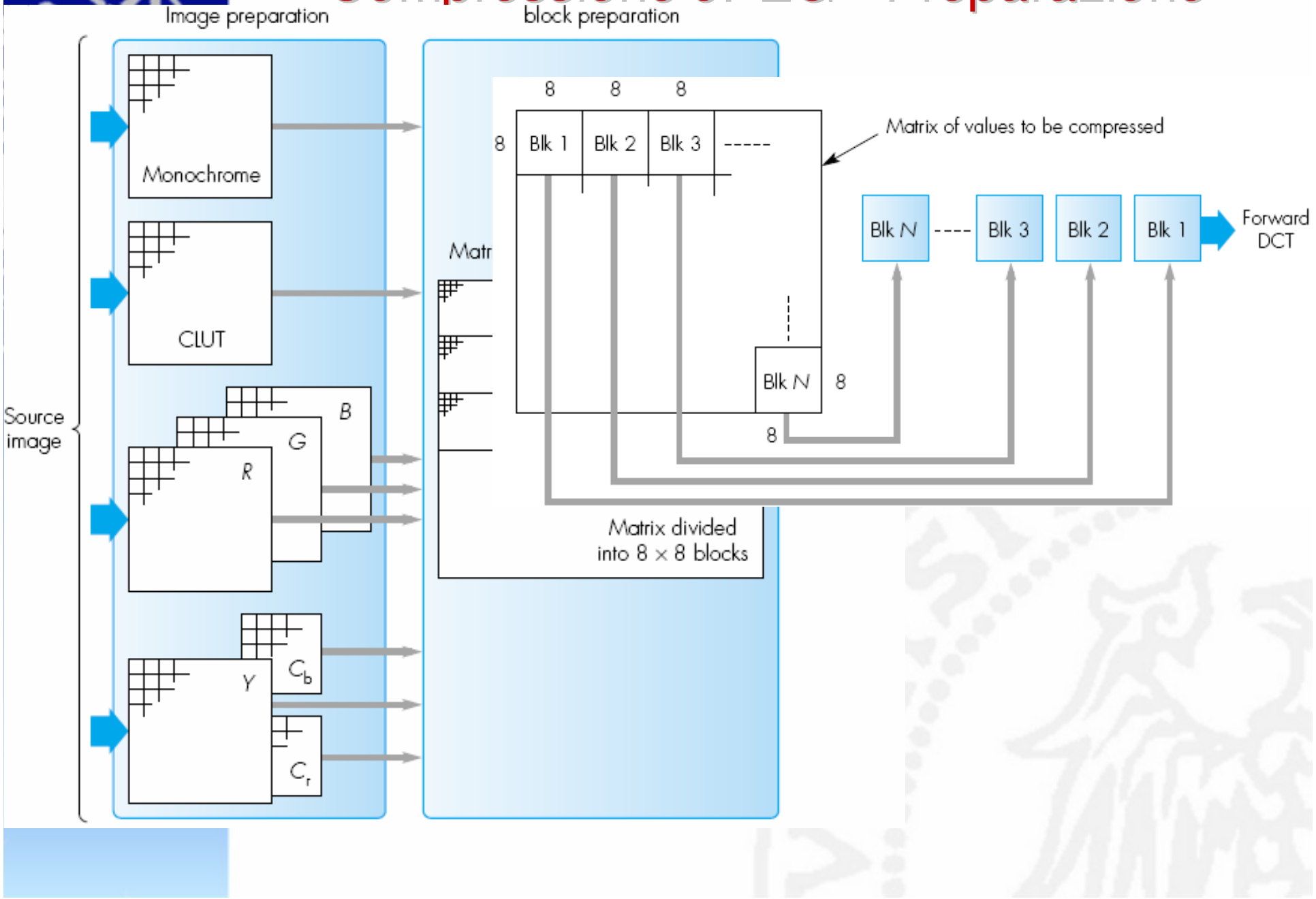
TRASFORMATA DCT

Il metodo di compressione JPEG si basa sulla DCT

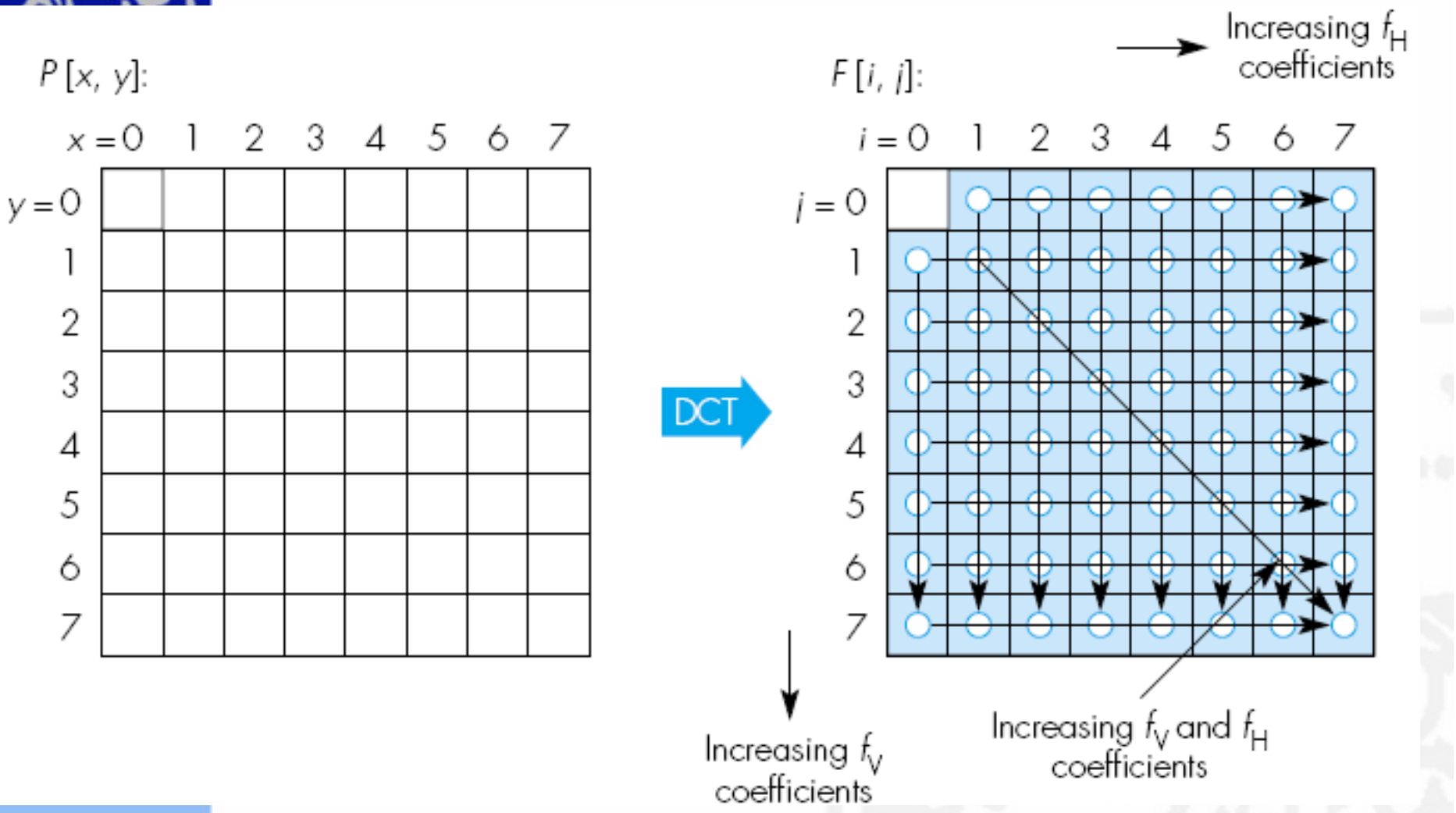
Compressione JPEG - Schema



Compressione JPEG - Preparazione



Effetto della DCT



Quantizzazione

DCT coefficients

120	60	40	30	4	3	0	0
70	48	32	3	4	1	0	0
50	36	4	4	2	0	0	0
40	4	5	1	1	0	0	0
5	4	0	0	0	0	0	0
3	2	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantizer

Quantized coefficients

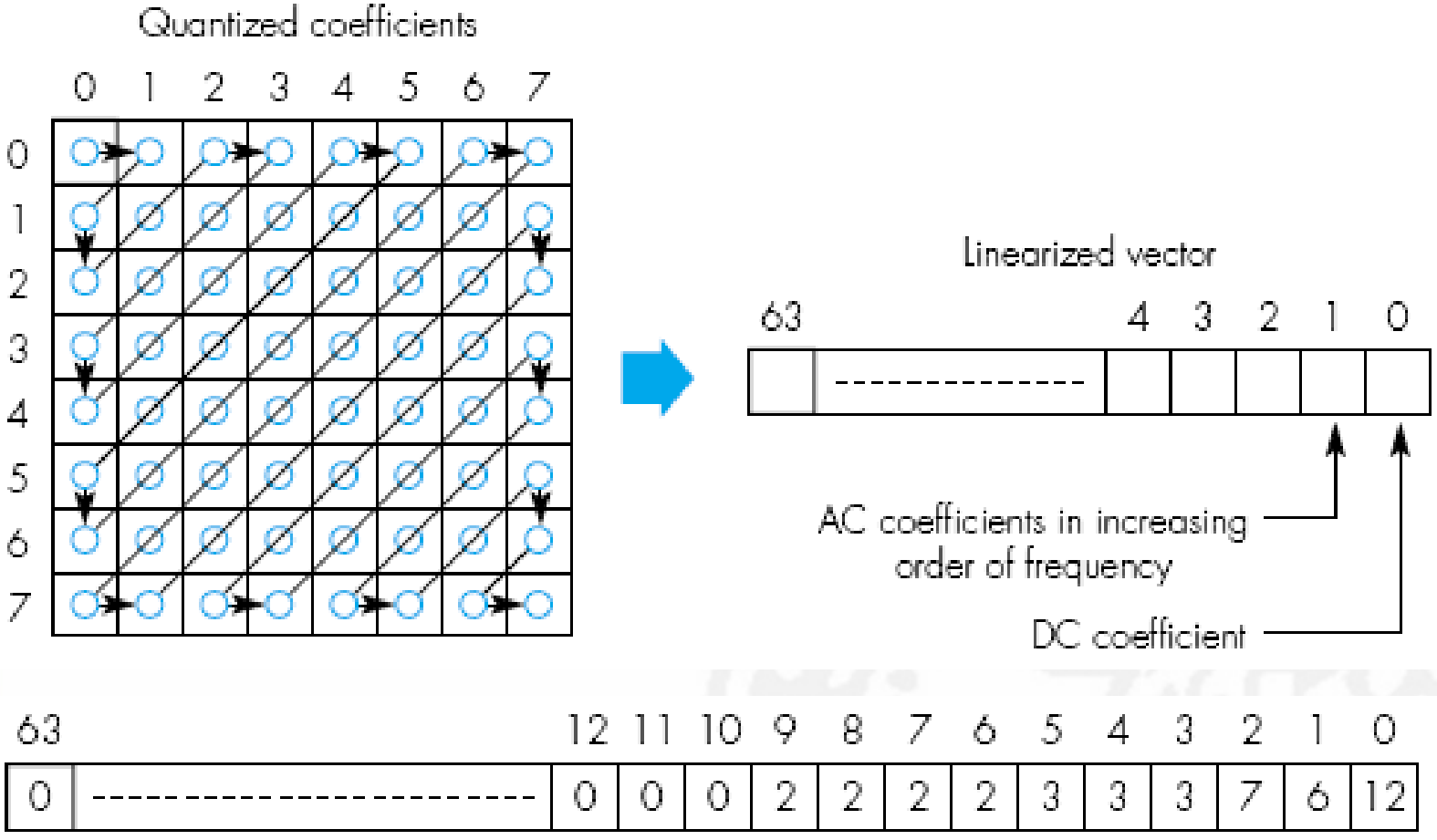
12	6	3	2	0	0	0	0
7	3	2	0	0	0	0	0
3	2	0	0	0	0	0	0
2	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

10	10	15	20	25	30	35	40
10	15	20	25	30	35	40	50
15	.20	25	30	35	40	50	60
20	25	30	35	40	50	60	70
25	30	35	40	50	60	70	80
30	35	40	50	60	70	80	90
35	40	50	60	70	80	90	100
40	50	60	70	80	90	100	110

Quantization table

Codifica di Entropia

Vettorizzazione del blocco risultato dalla quantizzazione -> Tecnica dove tail



Codifica di entropia

Il vettore viene codificato mediante RLC

I valori 1-63 vengono codificati mediante stringa di coppie di valori

(skip,value)

63		12	11	10	9	8	7	6	5	4	3	2	1	0
0	----------	0	0	0	2	2	2	2	3	3	3	7	6	12

(0,6)(0,7)(0,3)(0,3)(0,3)(0,2)(0,2)(0,2)(0,2)(0,0)

Codifica di Huffman