Università degli Studi di Bergamo

DIP. DI INGEGNERIA DELL'INFORMAZIONE E METODI MATEMATICI

RETI INTERNET MULTIMEDIALI

Codifica Numerica & Trasformate Discrete

CODIFICA NUMERICA

Introduzione

Introduzione alla Codifica Numerica

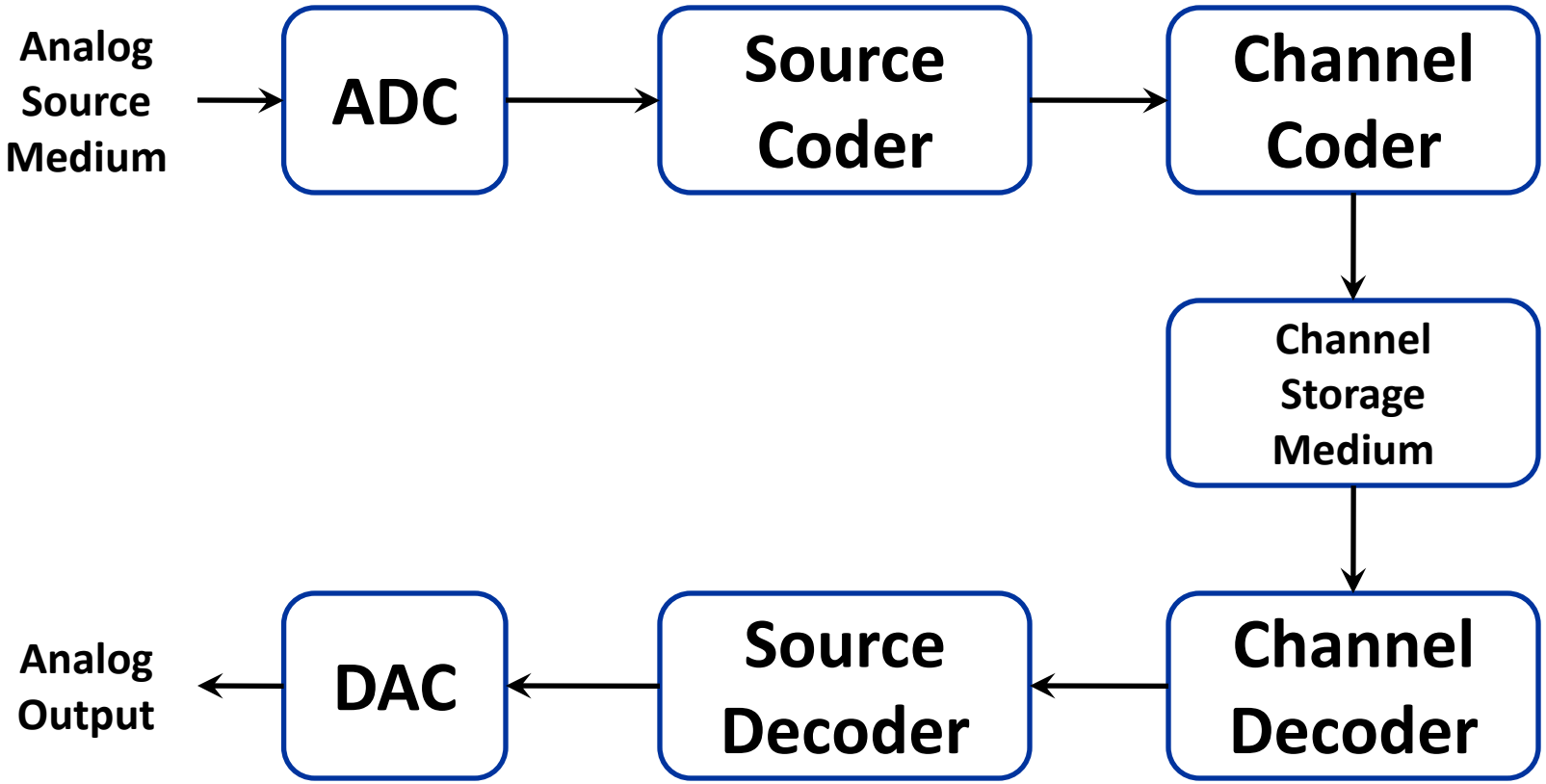
- La codifica numerica viene utilizzata in numerosi settori dell'ICT
- Può essere:
  - Lossless: senza perdita del contenuto informativo originario
  - Lossy: con perdita
- Parte del contenuto informativo può essere intenzionalmente perso con la codifica, per ridurre la dimensione del contenuto codificato

Introduzione alla Codifica Numerica

- La codifica viene eseguita in diversi punti della trasmissione di contenuti multimediali
  - Source coding: il contenuto originale viene compresso in un file (o stream)
  - Channel coding: il file (o stream) è convertito in un'opportuno segnale per la trasmissione
- I due tipi di codifica hanno obiettivi opposti
  - Source coding: comprimere i dati (riduce la ridondanza dell'informazione)
  - Channel coding: garantire una certa affidabilità (aumenta la ridondanza dell'informazione)

Introduzione alla Codifica Numerica

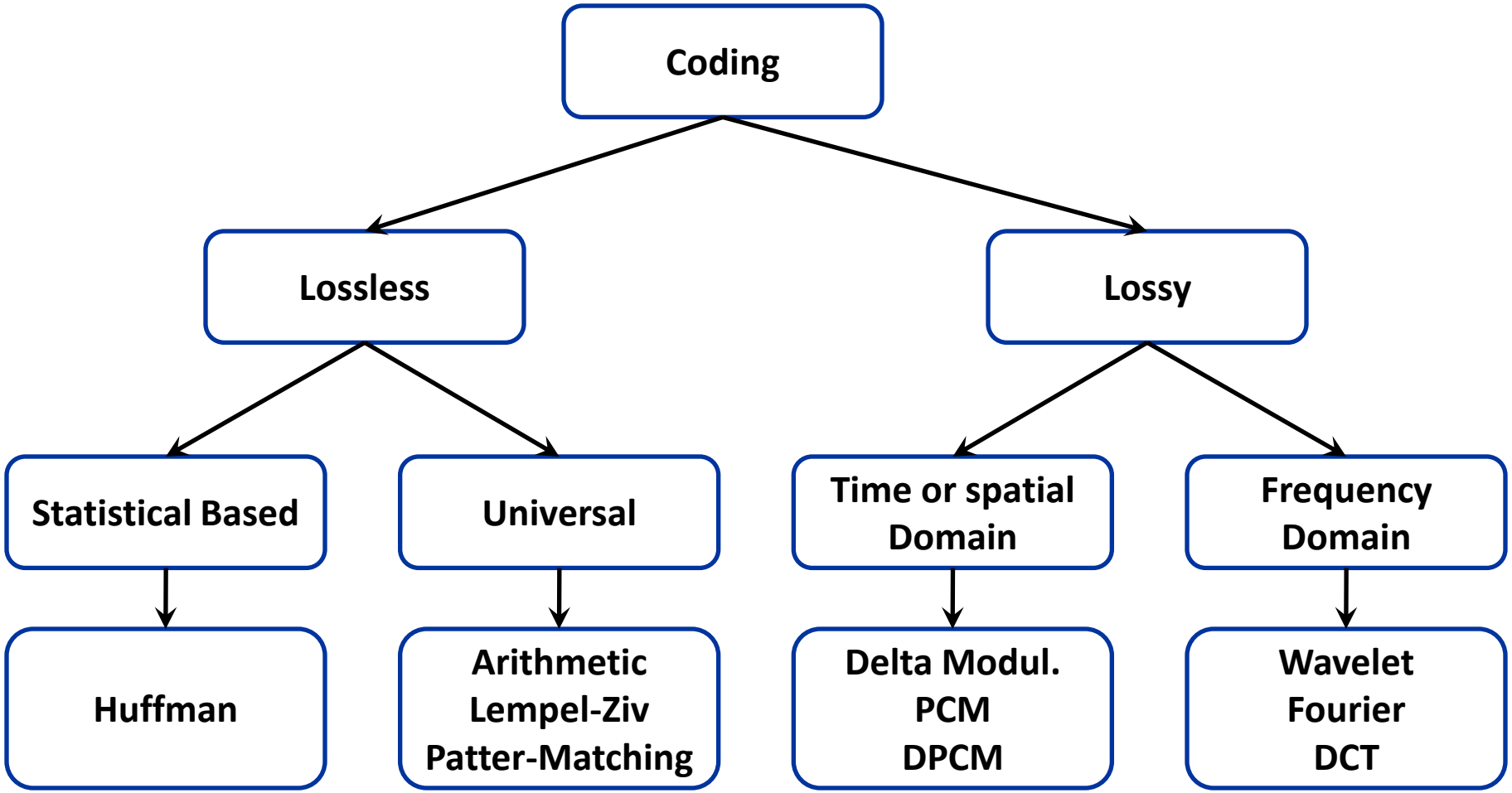
- Fasi di codifica, trasmissione e decodifica di contenuti multimediali



Introduzione alla Codifica Numerica

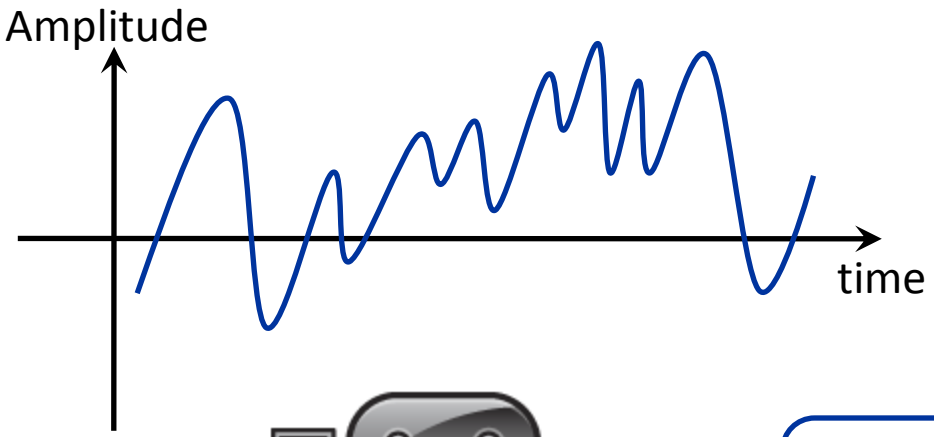
- La codifica utilizzata e i parametri utilizzati sono scelti considerando:
  - Efficienza (rapporto di compressione)
  - Ritardo
  - Complessità
  - Qualità dell'informazione decodificata
- Un ulteriore criterio per contenuti multimediali riguarda il bit-rate
  - Constante (CBR)
  - Variabile (VBR)

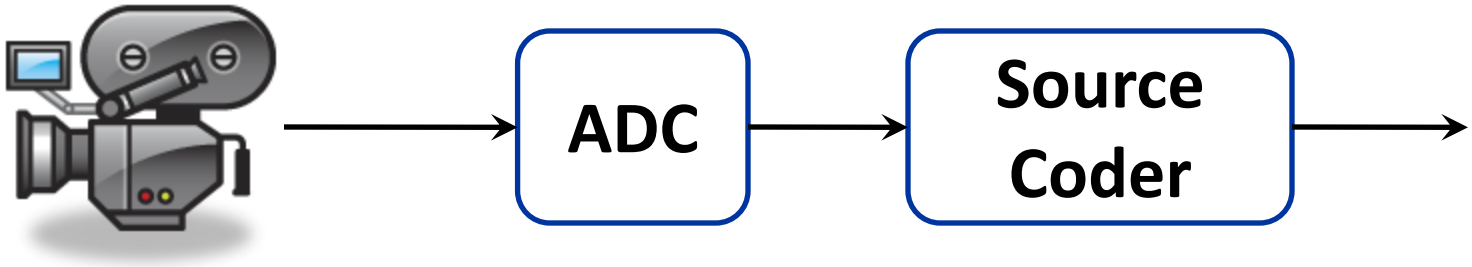
Introduzione alla Codifica Numerica



Introduzione alla Codifica Numerica

- Codifica temporale
  - il segnale in ingresso varia nel tempo
  - Es: voce, video

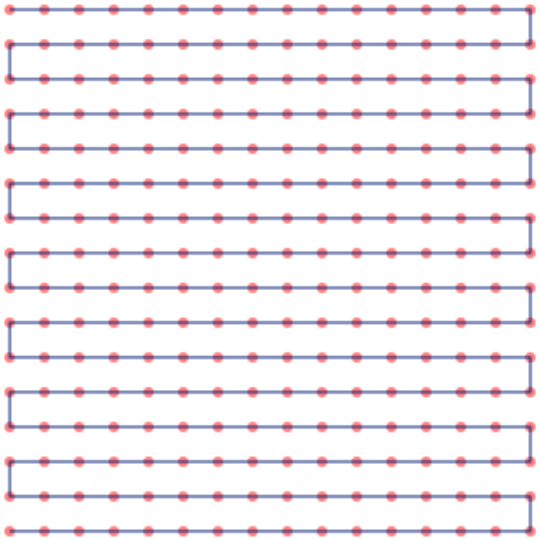




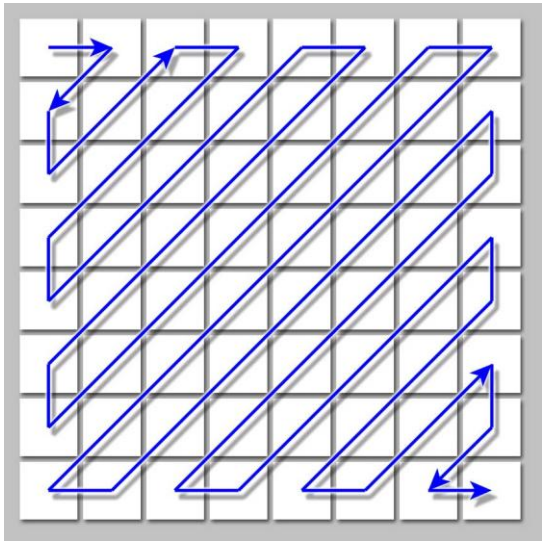
Introduzione alla Codifica Numerica

- Codifica spaziale:
  - il segnale varia in una o più dimensioni dello spazio
  - La sequenza spaziale (il percorso) utilizzata dalla codifica viene definita per convenienza a priori

Linear



Zig-Zag



Introduzione alla Codifica Numerica

- Per poter misurare le prestazioni degli algoritmi di codifica utilizzati nella compressioni di contenuti multimediali è necessario introdurre alcuni concetti fondamentali della teoria dell'informazione
  - Informazione
  - Entropia
- Tali concetti permettono di calcolare la quantità di informazione di una sorgente

Informazione

- Assunzioni fondamentali:
  - L'informazione di un messaggio/evento è tanto maggiore quanto meno probabile è il messaggio/evento
  - L'informazione di una coppia di messaggi indipendenti è la somma delle rispettive quantità di informazione
- Nota: solo il contenuto sintattico è di interesse ai fini della misura di informazione (non quello semantico)

Informazione

- Sia x_i un messaggio generato dalla sorgente X con probabilità $P(x_i)$
- L'informazione del messaggio x_i è definita nel seguente modo:

$$I(x_i) = \log \frac{1}{P(x_i)} = -\log P(x_i)$$

- La base del logaritmo identifica l'unità di misura

Entropia

- L'Entropia della sorgente X è definita come il valore atteso dell'informazione della sorgente X:

$$H(X) = \sum_{i=1}^{N} P(x_i) \log \frac{1}{P(x_i)} = -\sum_{i=1}^{N} P(x_i) \log\left(P(x_i)\right)$$

- L'entropia rappresenta perciò una misura
  - del contenuto informativo atteso della sorgente
  - dell'incertezza media della sorgente
- Se la base del logaritmo è 2 l'entropia viene misurata in bits (logaritmo naturale ➔ nats)

Entropia

- Si consideri una variabile casuale X in {0,1}

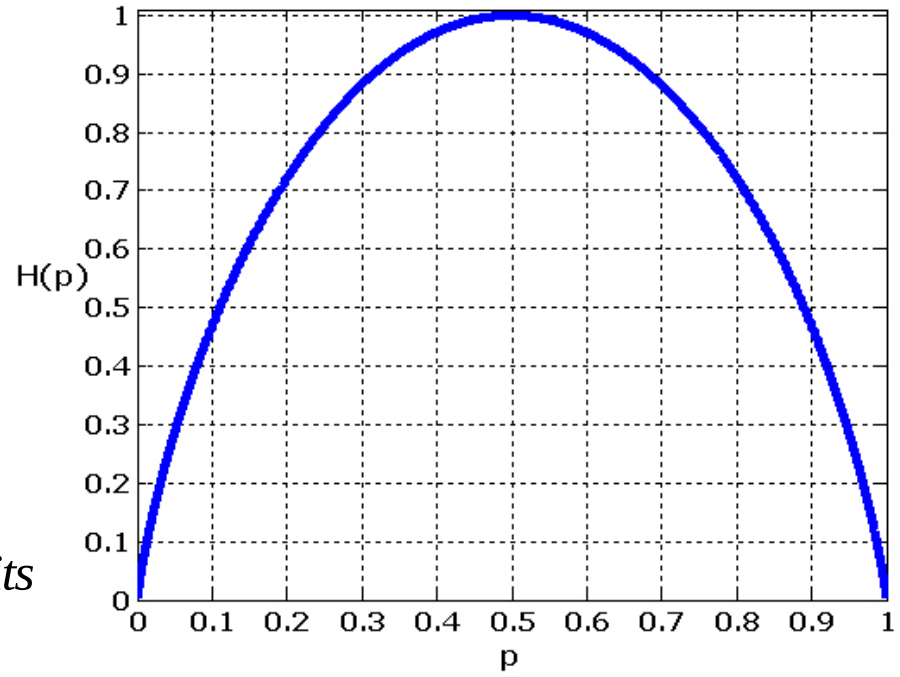
$$X = \begin{cases} 1 & \text{con prob. } p \\ 0 & \text{con prob. } 1-p \end{cases}$$

$p = 0.5$

$H(X) = -(0.5\log 0.5 + 0.5\log 0.5) = 1 \quad bit$

$p = 0.9$

$H(X) = -(0.1\log 0.1 + 0.9\log 0.9) = 0.469 \quad bits$



CODIFICA NUMERICA

Codifica Lossless

Lossless Entropy Coding

- Tecniche di codifica basate su modelli statistici basati sull'occorrenza dei simboli di input
  - Esempio Classico: Codice Morse
- Dato un insieme di simboli in ingresso sono generate parole di codice tali da minimizzare la lunghezza attesa delle parole di codice
- L'obiettivo è quindi quello di ridurre simboli ridondanti e ottenere un rate medio di informazione simile all'entropia della sorgente

Codifica di Huffman

- Utilizzato negli algoritmi di compressione di tipo Lossless (senza perdita di informazione)
- L'idea è quella di creare un albero binario in cui la somma della lunghezza dei percorsi per raggiungere le foglie è minima
- L'algoritmo costruisce l'albero a partire dalle foglie (bottom-up), che raprpesentano i simboli generati dalla sorgente

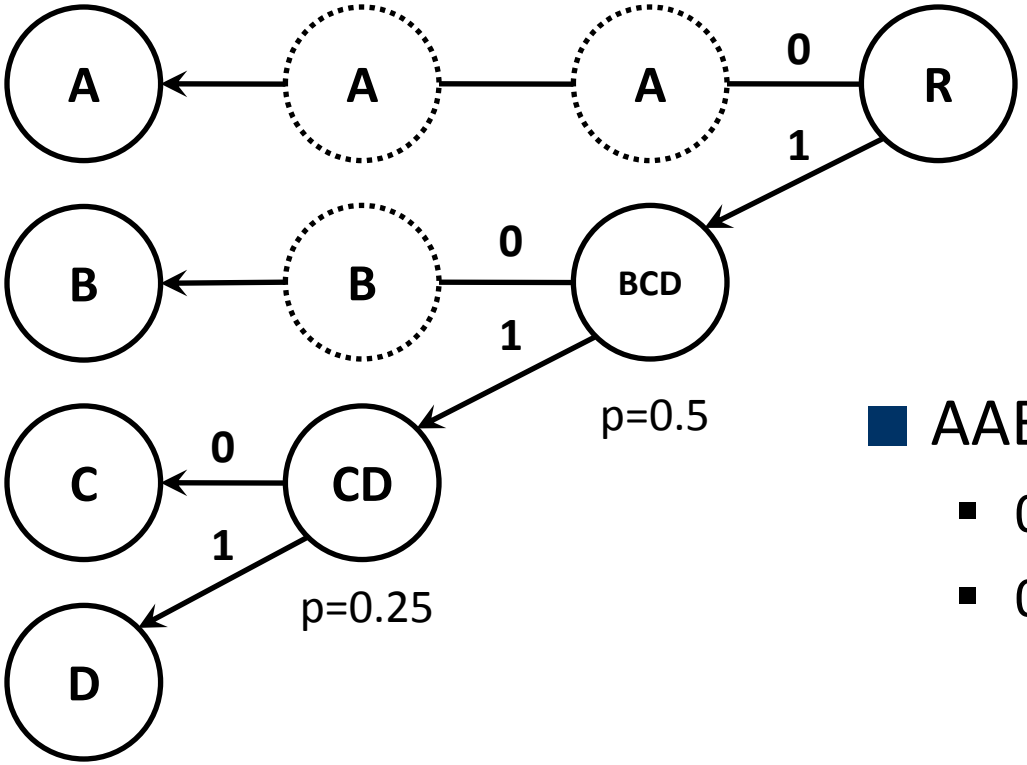
Codifica di Huffman

- Algoritmo:
  1. Ordina i simboli in ordine decrescente rispetto alle loro probabilità
  2. Combina gli ultimi due simboli in un nuovo simbolo. Ripeti il passo (1) finché la nuova lista contiene solo 2 simboli (figli della radice)
  3. A partire dal nodo radice assegna al ramo sx ‘0’ e al dx ‘1’
- Il codice di ogni simbolo è uguale alla sequenza di ‘1’ e ‘0’ del percorso che unisce la radice con la foglia che rappresenta il simbolo

Codifica di Huffman

- Esempio:

A	B	C	D
0.5	0.25	0.125	0.125



	Huffman	Binary
A	0	00
B	10	01
C	110	10
D	111	11

- AABAABCD è codificato:
  - 00100010110111 (14 bits)
  - 0000010000011011 (16 bits)

Codifica di Huffman

- Esistono casi in cui la codifica di Huffman non determina univocamente la lunghezza delle parole di codice, a causa delle scelte arbitrarie fra coppie di probabilità minime
  - Per esempio nel caso di una sorgente con probabilità {0.4, 0.2, 0.2, 0.1, 0,1} è possibile ottenere parole di codice di lunghezza [2, 2, 2, 3, 3] e [1, 2, 3, 4, 4]
- Sarebbe meglio avere parole di codice la cui lunghezza ha varianza minima
  - Soluzione più efficiente in quanto necessita di buffer di lunghezza minima per il trasmittente e per il ricevente

Codifica di Huffman

- Per alfabeti di sorgente di dimensione ridotta una codifica efficiente richiede blocchi di dimensione relativamente elevata
- Uno degli svantaggi del codificatore di Huffman è che la soluzione relativa a blocchi di dimensione k non può essere dedotta a da quella relativa a blocchi di dimensione k-1.
- Il codificatore di Huffman richiede infatti il calcolo completo delle probabilità di tutte le possibili sequenze di lunghezza k e la costruzione dell'albero relativo
- Più adatti appaiono gli schemi di codifica che consentono l'estensione a lunghezze di blocco via via crescenti di soluzioni relative a lunghezze di blocco inferiori.

Codifica Aritmetica

- Algoritmo di codifica:
  1. Inizializza l'**intervallo corrente** a [0;1).
  2. Per ogni carattere della sequenza:
     1. **Suddividi** l'intervallo corrente **in sottointervalli**, uno per ciascun simbolo dell'alfabeto. **L'ampiezza** di ciascun sottointervallo è **proporzionale alla probabilità** (stimata) che il carattere corrente sia uguale al simbolo corrispondente all'intervallo.
     2. **Seleziona** come intervallo corrente il **sottointervallo** corrispondente al **carattere corrente**.
- La lunghezza finale dell'intervallo corrente è uguale al prodotto delle probabilità dei singoli caratteri della sequenza codificata

Codifica Aritmetica

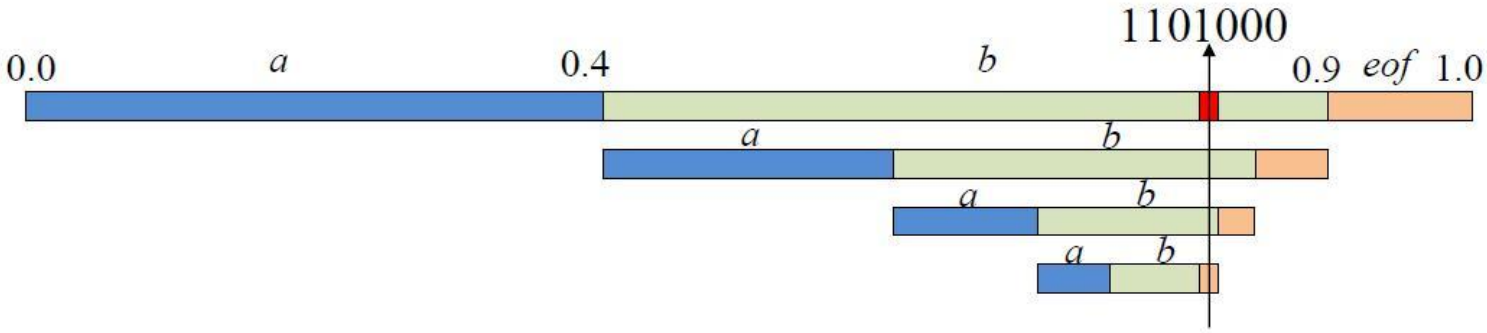
- Esempio (codifica)

A	B	C
0.4	0.5	0.1

- Input: BBBC

Intervallo Corrente	A	B	C	Input	Azione
[0,1)	[0, 0.4)	**[0.4, 0.9)**	[0.9, 1)	B	Dividi
[0.4, 0.9)	[0.4, 0.6)	**[0.6, 0.85)**	[0.85, 0.9)	B	Dividi
[0.6, 0.85)	[0.6, 0.7)	**[0.7, 0.825)**	[0.825, 0.85)	B	Dividi
[0.7, 0.825)	[0.7, 0.75)	[0.75, 0.812)	**[0.812, 0.825)**	C	Stop
[0.812, 0.825)					

Codifica Aritmetica



- L’intervallo finale è [0.812,0.825), che è approssimato in rappresentazione binaria [0.11010000, 0.11010011)
- L’intervallo è identificato dal suo lower bound. Si può quindi trasmettere la sola stringa 1101000.

Codifica Aritmetica

- Il decodificatore ha bisogno delle probabilità dei simboli codificati, in quanto esegue simili operazioni a quelle della codifica
- Algoritmo di decodifica
  1. Inizializza l'intervallo corrente a [0, 1)
  2. Per ciascuno degli N caratteri da decodificare:
     1. Suddividi l'intervallo corrente in sottointervalli, uno per ciascun simbolo dell'alfabeto. L'ampiezza di ciascun sottointervallo è proporzionale alla probabilità (stimata) che il carattere corrente sia uguale al simbolo corrispondente all'intervallo
     2. Seleziona il carattere corrispondente al sottointervallo in cui cade il valore x e seleziona tale sottointervallo come intervallo corrente

Codifica Aritmetica

- Esempio (decodifica)

A	B	C
0.4	0.5	0.1

- Input: 0.11010000 ➔ 0.812

Intervallo Corrente	A	B	C	Input	Output
[0,1)	[0, 0.4)	**[0.4, 0.9)**	[0.9, 1)	0.8	B
[0.4, 0.9)	[0.4, 0.6)	**[0.6, 0.85)**	[0.85, 0.9)	0.81	B
[0.6, 0.85)	[0.6, 0.7)	**[0.7, 0.825)**	[0.825, 0.85)	0.812	B
[0.7, 0.825)	[0.7, 0.75)	[0.75, 0.812)	**[0.812, 0.825)**	0.812	C

Codifica Lempel Ziv

- Algoritmo di codifica indipendente dalla statistica della sorgente
  - Codificatori di sorgente universali
- Algoritmo di compressione adattativo basato su dizionario
- Si costruisce automaticamente un dizionario che comprenda le stringhe viste fino ad adesso nel messaggio da comprimere
- Il testo precedente costituisce infatti un ottimo dizionario dato che generalmente condivide lo stesso linguaggio e stile della restante arte del messaggio da codificare

Codifica Lempel Ziv

- Il dizionario non deve essere inviato o memorizzato
- Il decodificatore lo costruisce utilizzando lo stesso meccanismo utilizzato dal codificatore
- Esistono numerosi varianti dell'algoritmo
  - Rappresentazione dei puntatori
  - Lunghezza massima oltre la quale i puntatori non possono più fare riferimento
  - Brevetti

Codifica Lempel Ziv

- Algoritmo di codifica:

1. La sequenza d'ingresso é scomposta in blocchi di lunghezza variabile, chiamati frasi
2. Ogni volta che un blocco di lettere differisce da uno dei blocchi precedenti per l'ultima lettera esso viene inserito nel dizionario
3. Tutte le frasi sono riportate nel dizionario unitamente all'informazione sulle locazioni in cui ciascuna frase compare
4. Nel codificare quindi una nuova frase, l'algoritmo registra nel dizionario la locazione da cui essa inizia e la nuova lettera

Codifica Lempel Ziv

- Il messaggio compresso consiste in una serie di triplette <P, L, C>
  - P (puntatore) indica quanto indietro guardare nel testo già decodificato
  - L = la lunghezza della frase
  - C = carattere successivo nel messaggio di input

Codifica Lempel Ziv

- Esempio 1 (codifica)

  Alfabeto = {x, y}

  Input: x y xx yxy xxyy

  Output: <0,0,x> <0,0,y> <2,1,x> <3,2,y> <5,3,y>
- Esempio 1 (decodifica)

  Alfabeto = {x, y, z}

  Output: <0,0,x> <0,0,y> <2,1,z> <2,1,x> <5,3,z> <6,3,z> <5,2,z>

  Input: x y xz xx yxzz xxyz zxz

Codifica Lempel Ziv

- Esempio 2

  Input:

  1010110100100111010100001100111010110001101 1

  Scomposizione in frasi

  1, 0, 10, 11, 01, 00, 100, 111, 010, 1000, 011, 001, 110, 101, 10001, 1011
- La parola di codice è ottenuta concatenando:
  - la locazione del dizionario che contiene la frase precedente che coincide fino al penultimo carattere (0000 locazione della parola vuota)
  - Il nuovo carattere

	Locazione dizionario	Contenuto dizionario	Parola di codice
1	0001	1	0000**1**
2	0010	0	0000**0**
3	0011	10	000**10**
4	0100	11	0001**1**
5	0101	01	0010**1**
6	0110	00	0010**0**
7	0111	100	0011**0**
8	1000	111	0100**1**
9	1001	010	0101**0**
10	1010	1000	0111**0**
11	1011	011	0101**1**
12	1100	001	0110**1**
13	1101	110	0100**0**
14	1110	101	0011**1**
15	1111	10001	1010**1**
16		1011	1110**1**

CODIFICA NUMERICA

Trasformate Discrete

Trasformate Discrete

- Le trasformate discrete sono utilizzate per rappresentare nel dominio delle frequenze un segnale campionato nel dominio del tempo o dello spazio
- Il segnale è rappresentato come somma pesata di funzioni base
  - La sequenza di coefficienti permette di ricreare il segnale originario
  - Coefficienti “non utili” ai fini dell’utilizzo del segnale originario possono essere filtrati (es. componenti non percettibili dall’uomo)

Discrete Fourier Transform (DFT)

- Rappresentazione di una sequenza di campioni di segnale nel dominio delle frequenze
- E’ utilizzata per l’analisi di segnali a tempo discreto composti da un numero finito di campioni
- Il segnale è la somma di N funzioni sinusoidali (funzioni base)

Discrete Fourier Transform (DFT)

- DFT in una dimensione di un segnale X(n) composto da N campioni:

$$X(n) = \frac{1}{N}\sum_{k=0}^{N-1} x(k) e^{\left(\frac{-j2\pi nk}{N}\right)} \qquad n = 0,1,2\ldots, N-1$$

$$= \frac{1}{N}\sum_{k=0}^{N-1} x(k)\left[\cos\left(2\pi nk\right) - j\sin\left(2\pi nk\right)\right]$$

- Inverse DFT (IDFT) 1-D:

$$x(k) = \sum_{n=0}^{N-1} X(n) e^{\left(\frac{j2\pi nk}{N}\right)} \qquad k = 0,1,2\ldots, N-1$$

Discrete Fourier Transform (DFT)

- DFT in due dimensioni con NxM campioni:

$$X(n) = \frac{1}{NM} \sum_{k=0}^{N-1} \sum_{h=0}^{M-1} x(k,h) e^{\left(\frac{-j2\pi nk}{N}\right)} e^{\left(\frac{-j2\pi mk}{M}\right)} \qquad \begin{matrix} n = 0,1,2\ldots, N-1 \\ m = 0,1,2\ldots, M-1 \end{matrix}$$

- DCT 2-D inversa:

$$x(k,h) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} X(n,m) e^{\left(\frac{j2\pi nk}{N}\right)} e^{\left(\frac{j2\pi mk}{M}\right)} \qquad \begin{matrix} k = 0,1,2\ldots, N-1 \\ h = 0,1,2\ldots, M-1 \end{matrix}$$

DFT - Proprietà

- **Linearità**:

  Se x(k) e y(k) sono due successioni di valori reali di lunghezza N, le cui rispettive DFT sono X(n) e Y(n), allora

$$x(k) \xrightarrow{DFT} X(n) \quad y(k) \xrightarrow{DFT} Y(n)$$

$$ax(k) + by(k) \xrightarrow{DFT} aX(n) + bY(n)$$

- **Periodicità:**

  x(k) e X(n) sono periodiche di periodo N

DFT - Proprietà

- **Simmetria**:

  Se x(k) è una successione di valori reali di lunghezza N, la cui DFT è X(n), allora

$$X(n) = \overline{X}(-n)$$

  Dalla relazione precedente segue

$\Re(\{X(n)\})$ Funzione pari

$\Im(\{X(n)\})$ Funzione dispari

DFT - Proprietà

- **Simmetria**:

Segnale in Ingrsso	DFT
Pari $x(k)=x(-k)$	Pari $X(n)=X(-n)$
Dispari $x(k)=-x(-k)$	Dispari $X(n)=-X(-n)$
Reale e Pari $x(k)=\bar{x}(k)=x(-k)$	Reale e Pari $X(n)=\bar{X}(n)=X(-n)$
Reale e Dispari $x(k)=\bar{x}(k)=-x(-k)$	Immaginaria e Dispari $X(n)=-\bar{X}(n)=-X(-n)$

DFT - Proprietà

- **Traslazione**:

  Se x(k) è una successione di valori reali di lunghezza N, la cui DFT è X(n), allora

$$x(k) \xrightarrow{DFT} X(n)$$
$$x(k-k_0) \xrightarrow{DFT} e^{-jnk_0} X(n)$$
$$e^{-jn_0k} x(k) \xrightarrow{DFT} X(n-n_0)$$

  con k_0 e n_0 costanti

Discrete Fourier Transform (DFT)

- Il calcolo della DFT è molto dispendioso dal punto di vista computazionale
- Può essere eseguito in modo efficiente utilizzando una delle molte versioni dell'algoritmo FFT (Fast Fourier Transform)
  - Decimazione nel tempo: due sommatorie
    - Somma delle componenti x(k) con k pari
    - Somma delle componenti x(k) con k dispari
  - Decimazione nelle frequenze
    - Calcolo delle componenti in frequenza X(n) con n pari
    - Calcolo delle componenti in frequenza X(n) con n dispari

Fast Fourier Transform (FFT)

- Algoritmo Fast Fourier Transform (FFT)

Discrete Cosine Transform (DCT)

- Trasformata molto utilizzata dagli algoritmi di compressione audio e video
- Utilizza come funzioni base cosinusoidi
- Come altre trasformate la DCT tenta di decorrelare i dati che compongono l'immagine
- I coefficienti ottenuti dalla trasformata possono essere codificati indipendentemente senza perdere efficienza di compressione

Discrete Cosine Transform (DCT)

- DCT in una dimensione di un segnale:

$$X(n) = \alpha(n) \sum_{k=0}^{N-1} x(k) \cos\left(\frac{\pi(2k+1)n}{2N} \right) \quad n = 0,1,2\ldots, N-1$$

- Inverse DCT 1-D:

$$x(k) = \sum_{n=0}^{N-1} \alpha(n) X(n) \cos\left(\frac{\pi(2k+1)n}{2N} \right) \quad k = 0,1,2\ldots, N-1$$

- Multiplier:

$$\alpha(n) = \begin{cases} \sqrt{\frac{1}{N}} & n = 0 \\ \sqrt{\frac{2}{N}} & n \neq 0 \end{cases}$$

Discrete Cosine Transform (DCT)

- Il primo componente (n=0) è chiamato anche DC coefficient e rappresenta una “media” della sequenza di campioni

$$X(0) = \sqrt{\frac{1}{N}} \sum_{k=0}^{N-1} x(k)$$

- I restatanti componenti sono detti coefficienti AC
- DC e AC per motivi storici (analisi di segnali in circuiti elettrici)

Discrete Cosine Transform (DCT)

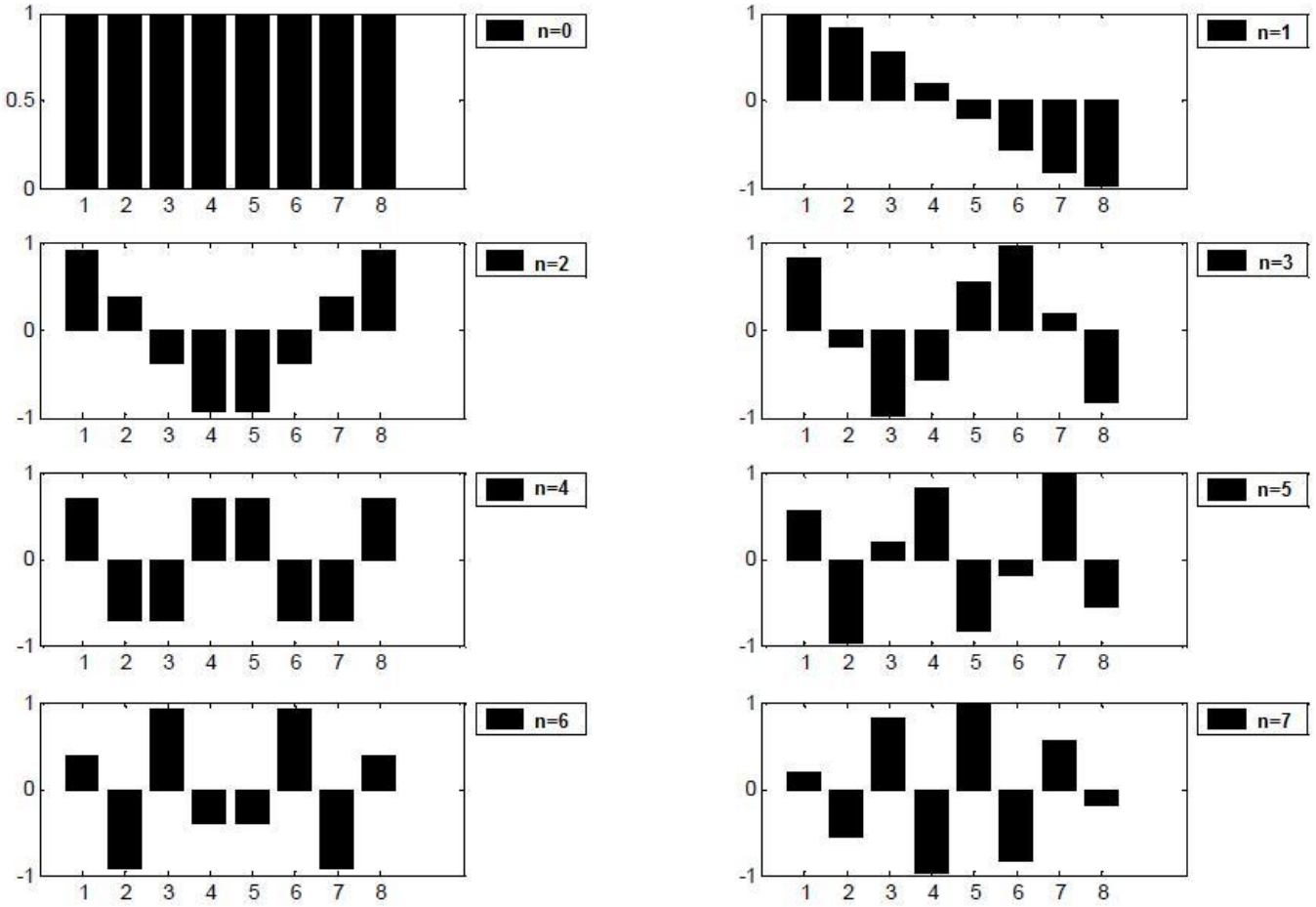
- Waveforms WF(n) per n=1, 2,…, N-1 sono chiamate Cosine Basis Functions

$$WF(n) = \sum_{k=0}^{N-1} \cos\left(\frac{\pi(2k+1)n}{2N}\right) \quad n = 1,2\ldots, N-1$$

- Sono funzioni ortogonali:
  - la moltiplicazione di due diverse WF, seguita dalla somma di tutti i campioni è nulla
  - la moltiplicazione di una WF con se stessa, seguita dalla somma di tutti i campioni è un valore scalare

Discrete Cosine Transform (DCT)

- Basi ortogonali WF(n) della 1-D DCT (N=8)



Discrete Cosine Transform (DCT)

- DCT in due dimensioni con NxM campioni:

$$X(n,m) = \alpha(n)\alpha(m)\sum_{k=0}^{N-1}\sum_{h=0}^{M-1} x(k,h)\cos\left(\frac{\pi(2k+1)n}{2N}\right)\cos\left(\frac{\pi(2h+1)m}{2M}\right) \quad \begin{matrix} n = 0,1,2\ldots,N-1 \\ m = 0,1,2\ldots,M-1 \end{matrix}$$

- DCT 2-D inversa:

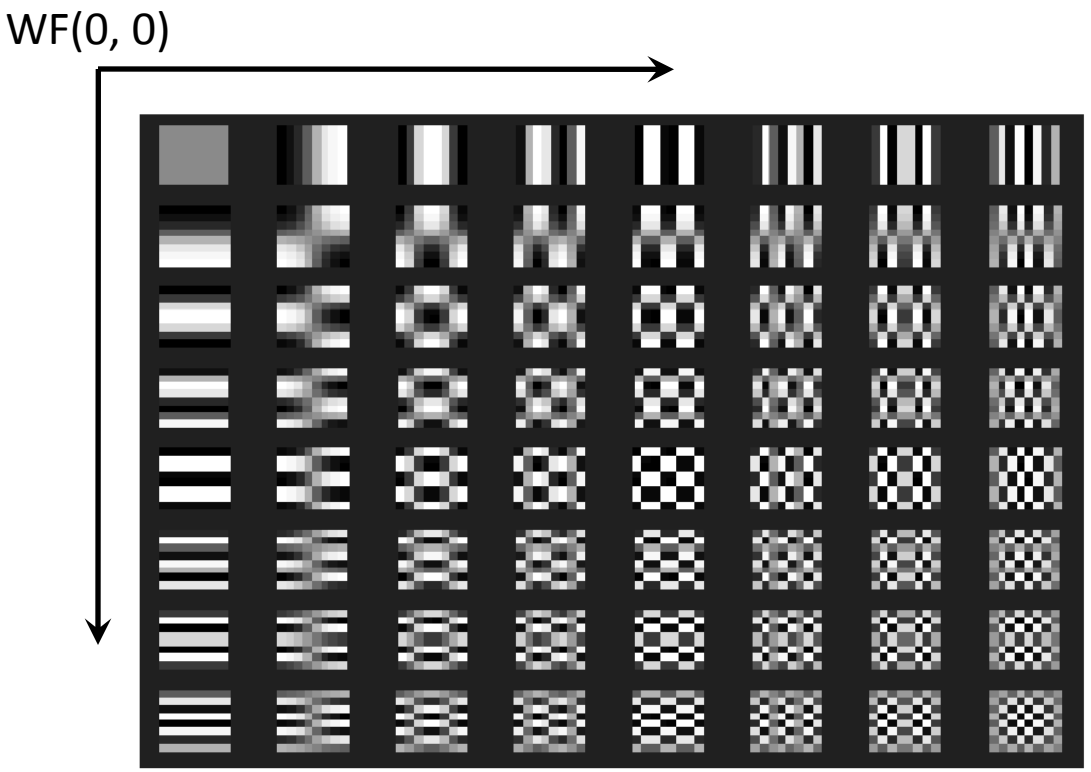
$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{M-1} \alpha(u)\alpha(v)C(u,v)\cos\left(\frac{\pi(2x+1)u}{2N}\right)\cos\left(\frac{\pi(2y+1)v}{2M}\right) \quad \begin{matrix} x = 0,1,2\ldots,N-1 \\ y = 0,1,2\ldots,M-1 \end{matrix}$$

- Multiplier:

$$\alpha(n) = \begin{cases} \sqrt{\frac{1}{N}} & n = 0 \\ \sqrt{\frac{2}{N}} & n \neq 0 \end{cases}$$

Discrete Cosine Transform (DCT)

- Basi ortogonali WF(n,m) della 2-D DCT (N=8, M=8)
  - Grigio: zero
  - Nero: ampiezza negativa
  - Bianco: ampizza positiva



Wavelet Compression

- Promettente alternativa alle classiche trasformate introdotte in precedenza
- Le basi che compongono il segnale sono compressioni o dilatazioni di una funzione “madre” o wavelet (a differenza delle armoniche dell’analisi di Fourier)
- Rispetto alla trasformata di Fourirer:
  - Ha il vantaggio di rappresentare meglio immagini in cui vi sono discontinuità o bruschi cambiamenti
  - Si possono utilizzare diversi tipi di wavelet, invece delle sole sinusoidi

Wavelet Decomposition

- Le funzioni che formano le basi nella analisi wavelet sono definite come:

$$\phi_{s,k}(t) = 2^{-\frac{s}{2}}\phi(2^{-s}t - k)$$

$\phi(t)$ ⟵ **Analyzing Wavelet**

s ⟵ **Scale Index**

k ⟵ **Location Index**

- Gli indici s e k permettono di dilatare e traslare la Analyzing Wavelet

Wavelet Decomposition

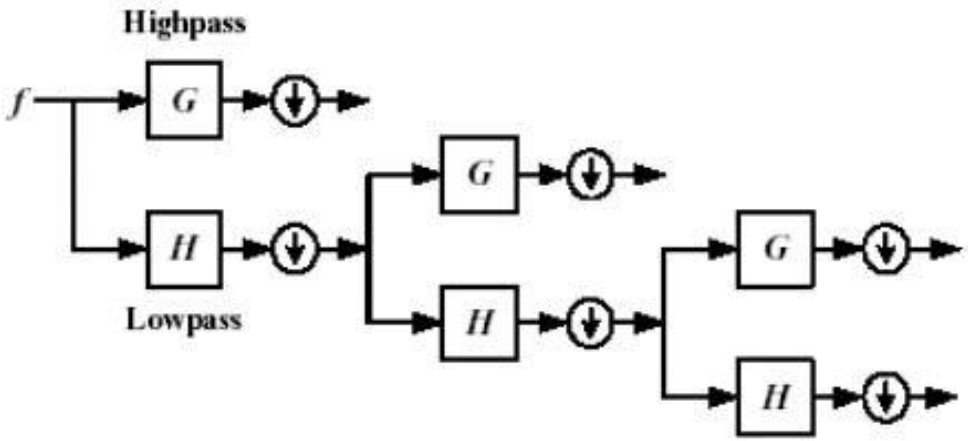
- Le funzioni base sono ottenute da dilatazioni e traslazioni della *Analyzing Wavelet* e sono tra loro ortogonali
- Discrete Wavelet Transform (DWT)

$$d_{s,k} = \sum_{s=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} x(t)\phi_{s,k}(t)$$

$d_{s,k}$ ⟵ **Coefficienti della DWT**

Wavelet Decomposition

- Discrete Wavelet Transform (DWT) 2-D (per immagini)
  - Albero composto da una successione di filtri a low-pass e high-pass
  - A livello L il segnale di riferimento ha una risoluzione ridotta di un fattore 2^{2L} rispetto a segnale in ingresso



Wavelet Decomposition

- Haar Wavelet Decomposition