Συστήματα και Αλγόριθμοι Πολυμέσων

Ιωάννης Χαρ. Κατσαβουνίδης

Ομιλία #4: Αρχές Επεξεργασίας Σημάτων Πολυμέσων

11 Οκτωβρίου 2005

Πανεπιστήμιο Θεσσαλίας
Τμήμα Μηχ. Η/Υ, Τηλεπ. & Δικτύων

Επανάληψη (1)

- Μετασχηματισμοί
  - Πλήρως αντιστρέψιμη διαδικασία
  - Σκοπός είναι η αποσυσχέτιση (de-correlation) του σήματος
  - Αρχή διατήρησης της ενέργειας $\| x \| = \| y \|$
  - Αλλά η κατανομή ενέργειας είναι πολύ διαφορετική μεταξύ του αρχικού σήματος και των συντελεστών του μετασχηματισμού: «κέρδος μετασχηματισμού»

Επανάληψη (2)

- Διακριτός μετασχηματισμός συνημιτόνου (DCT) 8 σημείων:

$$t_u = \frac{C_u}{2} \sum_{x=0}^{7} s_x \cos \frac{(2x+1)\pi u}{16}, u = 0..7$$

$$C_0 = \frac{1}{\sqrt{2}}, \; C_n = 1 \;\; \text{for n} = 1..7$$

Επανάληψη (3)

- «Κέρδος Μετασχηματισμού»:

$$N_1 = \frac{N}{2} + \log_2 \frac{\sigma_1}{\sqrt{\sigma_1 \sigma_2}}$$

- Π.χ. Εάν $\sigma_1 = \sigma_2 = 7.07$ πριν από το μετασχηματισμό, και N = 10 bits, E(q) = 3.125

  Εάν μετά το μετασχηματισμό $\sigma_1 = 9.5$ και $\sigma_2 = 3.12$, τότε $N_1 = 5.80$, $N_2 = 4.20$ και E`(q) = 2.15

  Το «κέρδος μετασχηματισμού» είναι 1.62 dB

Επανάληψη (4)

- Πολυδιάστατοι/διαχωρίσιμοι μετασχηματισμοί
  - Εφαρμόζονται πρώτα κατά τη μία διάσταση και εν συνεχεία κατά τη δεύτερη κ.ο.κ.
- Γρήγοροι μετασχηματισμοί
  - Η ιδέα είναι η εύρεση κοινών παραγόντων στους συντελεστές του πίνακα του μετασχηματισμού (FFT)
  - $O(N^2) \rightarrow O(N\log N)$ (DCT) $\rightarrow O(N)$ (Haar/Wavelets)

Επανάληψη (5)

- Εφαρμογές μετασχηματισμών
  - DCT(FDCT): Χρησιμοποιείται από τους codecs JPEG, MPEG1/2/4 Video (8x8), όπως επίσης και σε τροποποιημένη μορφή από τον codec MPEG1-Audio (“MP3”) – 12 ή 36 σημείων
  - Wavelets (Haar): JPEG2000
  - Hadamard: H.264

Μετασχηματισμοί σημάτων πολυμέσων

- Εφαρμόζονται (συνήθως) σε κάποιο υποσύνολο του σήματος, όπως για παράδειγμα κομμάτια 8x8 της εικόνας (JPEG, MPEG1/2/4) ή 12/36 δείγματα ήχου (MP3) ή 4x4 κομμάτια της εικόνας (H.264)
- Για να μειωθεί το φαινόμενο του σφάλματος στα όρια των κομματιών, εφαρμόζονται τεχνικές είτε αλληλοκάλυψης (MP3) είτε μετα-επεξεργασίας φιλτραρίσματος των ορίων (de-blocking filter – H.264)

Άσκηση #4

- Να χωρίσετε το πρώτο καρέ του αρχείου src13 σε υποπίνακες 8x8. Στη συνέχεια να εφαρμόσετε το διακριτό μετασχηματισμό DCT και να υπολογίσετε την ενέργεια των 64 συντελεστών για ολόκληρη την εικόνα.

Βασικές τεχνικές (αρχές) επεξεργασίας πολυμέσων

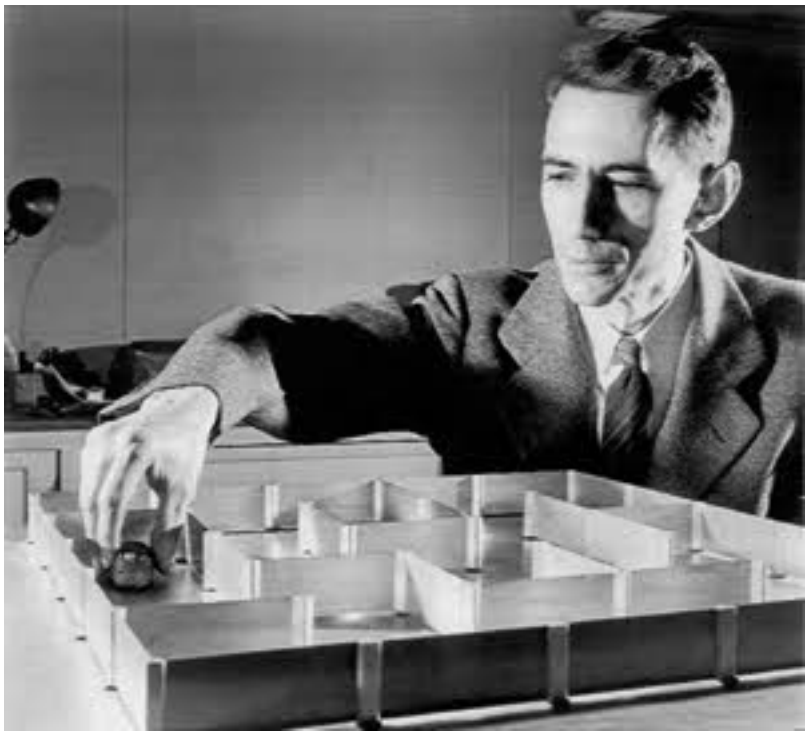
- *Δειγματοληψία*
- *Κβαντοποίηση*
- *Μετασχηματισμοί*
- Θεωρία Πληροφορίας
- Προ- και Μετά-επεξεργασία πολυμεσικών σημάτων

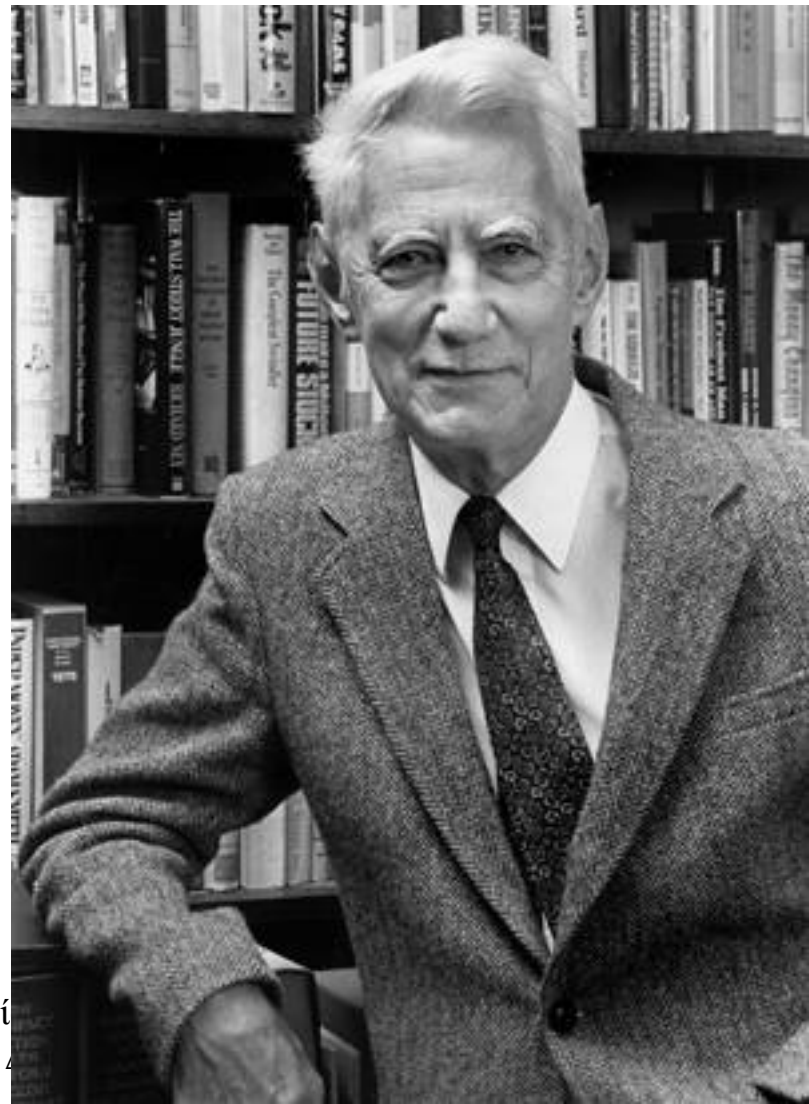
Θεωρία Πληροφορίας - Κωδικοποίησης

- Θεμελιώθηκε από τον Claude Shannon (AT&T-Bell Labs, 1948 “A Mathematical Theory of Communication”)
- Βαθειά μαθηματική θεωρία η οποία όμως εξηγεί πολύ σημαντικά προβλήματα ηλεκτρολόγων/τηλεπικοινωνιακών μηχανικών

Claude Shannon (1916-2001)

- Ο «παππούς» μας!





Θεωρία Πληροφορίας – Κωδικοποίησης (1)

- Η πληροφορία που μεταδίδει ένα σύμβολο είναι αντιστρόφως ανάλογη της πιθανότητάς του («Τι νέα;»)

$$I(p) = -\log_2 p \quad \text{(bits)}$$

- Η *εντροπία* μιας πηγής είναι η μέση τιμή της πληροφορίας των συμβόλων της

$$H(P) = -\sum_{i=0}^{N-1} p_i \log_2 p_i \quad \text{(bits)}$$

Θεωρία Πληροφορίας – Κωδικοποίησης (2)

- Κωδικοποίηση Huffman

Είσοδος	Αλφάβητο	α	β	γ	δ	ε	ζ	η	θ	ι	
	«Κόστος»	10	15	5	15	20	5	15	30	5	
Έξοδος	Κωδικός	000	010	0010	011	111	00110	110	10	00111	
	Ζυγισμένο μήκος κωδικού	10*3	15*3	5*4	15*3	20*3	5*5	15*3	30*2	5*5	=355

Θεωρία Πληροφορίας – Κωδικοποίησης (3)

- Ιδιότητες κωδικοποίησης Huffman
  - Εύκολος αλγόριθμος σχεδίασης βιβλίου κωδικών (codebook) – recursive greedy algorithm
  - Γίνεται να υλοποιηθεί «στατικά», όπου η φάση της σχεδίασης του βιβλίου κωδικών γίνεται πριν από τη μετάδοση του μηνύματος πληροφορίας και το βιβλίο παραμένει σταθερό καθ' όλη τη διάρκεια μετάδοσης.
  - Γίνεται να υλοποιηθεί «δυναμικά», δηλαδή για κάθε μήνυμα προηγείται η διαδικασία σχεδίασης του βιβλίου κωδικών, μετά η μετάδοση του βιβλίου και στο τέλος του κωδικοποιημένου μηνύματος
  - Γίνεται επίσης να ξεκινήσουμε με κάποιο βιβλίο κωδικών και να το μεταβάλουμε, προσαρμόζοντάς το για κάθε μήνυμα ("adaptive codebook").

Θεωρία Πληροφορίας – Κωδικοποίησης (3)

- Ιδιότητες κωδικοποίησης Huffman
  - Επιτυγχνάνει μέση τιμή κωδικοποιημένης πληροφορίας πολύ κοντά στην εντροπία της πηγής

$$L(P) = \sum_{i=0}^{N-1} p_i l_i < H(P) + 1 \quad \text{(bits)}$$

  - Πολύ εύκολη υλοποίηση κωδικοποίησης (look-up table) και – σχετικά – εύκολη υλοποίηση αποκωδικοποίησης (binary trees/look-up tables)

Θεωρία Πληροφορίας – Κωδικοποίησης (4)

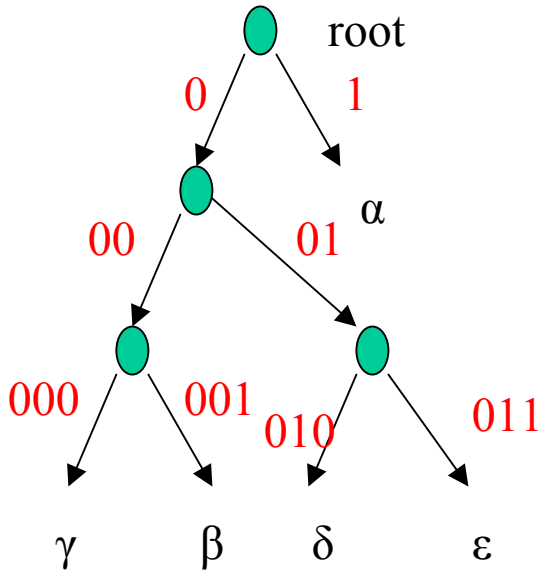
- Διαδικασία κωδικοποίησης:

Είσοδος	Αλφάβητο	α	β	γ	δ	ε	
	«Κόστος»	50	10	5	10	10	
Έξοδος	Κωδικός	1	001	000	010	011	
	Ζυγισμένο μήκος κωδικού	50*1	10*3	5*3	10*3	10*3	=155

Μήνυμα: αβγα → 1 001 000 1 → 0x91 (1 byte)

Θεωρία Πληροφορίας – Κωδικοποίησης (5)

- Διαδικασία αποκωδικοποίησης (A):




```
p = root;

while(!p->symbol)

{

        i = getbits(1);

        p = p->child[i];

}
```

Θεωρία Πληροφορίας – Κωδικοποίησης (6)

- Διαδικασία αποκωδικοποίησης (Β):

table_0[2] = {0, α};

table_1[4] = {γ, β, δ, ε};

```
i = getbits(1);
if(!(s=table_0[i]))
{
        i = getbits(2);
        s = table_1[i];
}
```

Θεωρία Πληροφορίας – Κωδικοποίησης (7)

- Διαδικασία αποκωδικοποίησης (Γ):

table[8] = {γ, β, δ, ε,
α, α, α, α};

bits[8] = { 3, 3, 3, 3,
1, 1, 1, 1};

```
i = showbits(3);
s = table[i];
advancebits(bits[i]);
```

Θεωρία Πληροφορίας – Κωδικοποίησης (8)

- int getbits(int n) : απομακρύνει n bits από τον εσωτερικό (state) buffer του συστήματος και επιστρέφει την αριθμητική τιμή των n bits.
- Π.χ.

  buffer = 0x91 (10010001); internal_bits = 8;

  getbits(1) επιστρέφει 1, internal_bits = 7;

  getbits(3) επιστρέφει 1 (001), internal_bits=4;

  getbits(3) επιστρέφει 0 (000), internal_bits=1;

  getbits(1) επιστρέφει 1, internal_bits = 0;

Θεωρία Πληροφορίας – Κωδικοποίησης (9)

- int showbits(int n) : επιστρέφει την αριθμητική τιμή n bits από τον εσωτερικό (state) buffer του συστήματος χωρίς να απομακρύνει bits
- Π.χ. buffer = 0x91 (10010001). internal_bits=8 καθόλη τη διάρκεια των παρακάτω.

  showbits(1) επιστρέφει 1

  showbits(4) επιστρέφει 9 (1001)

  showbits(7) επιστρέφει 72 (1001000)

  showbits(8) επιστρέφει 145 (0x91)

Θεωρία Πληροφορίας – Κωδικοποίησης (10)

- void advancebits(int n) : απομακρύνει n bits από τον εσωτερικό (state) buffer του συστήματος.
- Δηλαδή, i = getbits(n); ⇔ {i= showbits(n); advancebits(n);}

Θεωρία Πληροφορίας – Κωδικοποίησης (11)

- «Προβλήματα» κωδικοποίησης Huffman
  - Δεν έχει καλή απόδοση για πηγές που έχουν σύμβολα με πολύ μεγάλη πιθανότητα ($p \cong 1$)
  - Λύση: κωδικοποίηση πολλών συμβόλων ταυτόχρονα “Run Length Coding”

Θεωρία Πληροφορίας – Κωδικοποίησης (12)

- Π.χ. p(0) = 0.8, p(1) = 0.1, p(2) = 0.1
- I(0) = 0.322 bits, I(1) = I(2) = 3.322 bits
- H = 0.922 bits
- Κωδικοποίηση Huffman: 0→(1), 1→(00), 2 →(01), L = 1.2 bits («μόνο» 0.278 bits παραπάνω, αλλά 30% !!!)

Θεωρία Πληροφορίας – Κωδικοποίησης (13)

- Αν κωδικοποιήσουμε 2 σύμβολα ταυτόχρονα, τότε
  - $p(0,0) = 0.8*0.8=0.64 \Rightarrow I_0 = 0.644$ bits
  - $p(0,1) = p(1,0) = p(0,2) = p(2,0) = 0.8*0.1 = 0.08 \Rightarrow I_1 = 3.644$ bits
  - $p(1,1) = p(1,2) = p(2,1) = p(2,2) = 0.1*0.1 = 0.01 \Rightarrow I_2 = 6.644$ bits
- H = 1.844 bits (/2 σύμβολα = 0.922 bits/σύμβολο)

Θεωρία Πληροφορίας – Κωδικοποίησης (14)

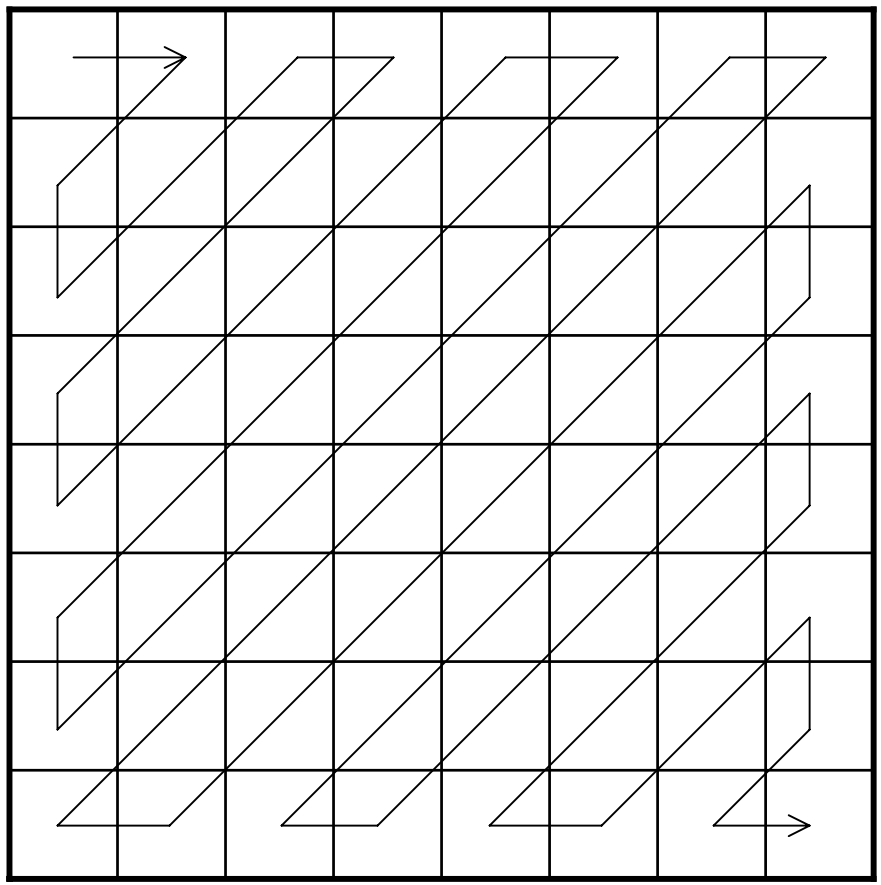
- Κωδικοποίηση Huffman:
  - (0,0)→ 1
  - (0,1)→ 010
  - (0,2)→ 0111
  - (1,0)→ 0110
  - (2,0)→ 001
  - (1,1)→ 00000
  - (1,2)→ 00001
  - (2,1)→ 00010
  - (2,2)→ 00011
  - L = 1.96 bits (+0.116 bits παραπάνω, αλλά 6.3%)

Θεωρία Πληροφορίας – Κωδικοποίησης (15)

- Κωδικοποίηση Run-Length:
  - $\underbrace{0,0,\ldots,0}_{N},L \rightarrow (N,L)$ (N: # of 0s, L≠0)
  - Π.χ. 0,0,0,1,0,0,2,1,0 → (3,1),(2,2),(0,1),EOB
  - EOB (“End of Block”) – σηματοδοτεί το τέλος του μηνύματος και κωδικοποιεί τα υπολοιπόμενα μηδενικά, καθότι ο συνολικός αριθμός συμβόλων είναι γνωστός (εικόνα-8x8 DCT: 64 σύμβολα)

Θεωρία Πληροφορίας – Κωδικοποίησης (16)

- Γραμμικοποίηση διδιάστατου πίνακα 8x8 μετά την κβαντοποίηση των συντελεστών μετασχηματισμού : zig-zag scan



Άσκηση

- Να χρησιμοποιήσετε όλες τις τεχνικές που μάθατε μέχρι τώρα (μετασχηματισμός DCT, κβαντοποίηση, zig-zag scan, κωδικοποίηση Huffman, κωδικοποίηση Run-Length) για να συμπιέσετε το 1ο καρέ του βίντεο-κλιπ src14. Συγκρίνετε τα αποτελέσματα (ποιότητα, λόγος συμπίεσης) με αυτά του στάνταρ JPEG.