ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ,

ΤΜΗΜΑ ΤΕΧΝΟΛΟΓΙΑΣ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΒΕΣ 04: ΣΥΜΠΙΕΣΗ ΚΑΙ ΜΕΤΑΔΟΣΗ ΠΟΛΥΜΕΣΩΝ

Ακαδημαϊκό Έτος 2007 – 2008, Χειμερινό Εξάμηνο

13 Νοεμβρίου 2007

Φροντιστηριακή Άσκηση 3: (I) Συμπίεση κειμένου με τον αλγόριθμο LZW

(II) Προβλεπτική κωδικοποίηση εικόνων

(III) Αριθμητική Κωδικοποίηση

I. Συμπίεση κειμένου με τον αλγόριθμο LZW

Δίνονται τα παρακάτω τμήματα ψευδοκώδικα τα οποία επιδεικνύουν τη λειτουργία της συμπίεσης και αποσυμπίεσης σύμφωνα με τον αλγόριθμο κατά LZW

LZW Compression	LZW Decompression

BEGIN
s = next input character;
while (~EOF)
{
c = next input character;
if s+c exists in the dictionary
s = s+c;
else
{
output the code for **s**;
add string **s+c** to the dictionary;
s = c;
}
}
output the code for **s**;
END
	BEGIN
	s = NIL;
	while (~EOF)
	{
	k = next input code;
	entry = dictionary entry for **k**;
	/* exception handler */
	if (**entry** ==NULL) **entry = s+s[0]**;
	
	output **entry**;
	if (**s** != NIL)
	add string **s+entry[0]** to the dictionary;
	s = entry;
	}
	END

(1) Εφαρμόστε τον αλγόριθμο συμπίεσης για την ακολουθία ABABBABCABABBA και συμπληρώστε τους πίνακες I.1 και I.2. Θεωρήστε ότι το αρχικό λεξικό περιέχει μόνο τις εξής εγγραφές:

String	Code	Codeword
A	1	0000
B	2	0001
C	3	0010

(2) Υπολογίστε το βαθμό συμπίεσης που επιτυγχάνεται στο παραπάνω παράδειγμα.

(3) Εφαρμόστε τον αλγόριθμο αποσυμπίεσης για τους κωδικούς 1 2 4 5 2 3 4 6 1 και συμπληρώστε τους πίνακες Ι.3 και Ι.4. Θεωρήστε ότι το αρχικό λεξικό είναι γνωστό στο δέκτη. Συγκρίνετε τους πίνακες Ι.2 και Ι.4.

(4) Εξηγήστε τη γραμμή ψευδοκώδικα που ακολουθεί το σχόλιο /* exception handler */ αποσυμπιεστή.

(5) Γράψτε τις ανάλογες συναρτήσεις (m-files) σε Matlab που υλοποιούν το συμπιεστή και αποσυμπιεστή LZW.

ΙΙ. Προβλεπτική κωδικοποίηση εικόνων

Συνεχίστε από το βήμα (1) της προηγούμενης άσκησης (έχετε επομένως φορτώσει στο χώρο εργασίας της Matlab την εικόνα '*pouf3.tif*').

(1) Αφού μετατρέψτε την εικόνα σε μονοδιάστατο διάνυσμα (π.χ. *g=f(:)*), υπολογίστε τις διαφορές ανάμεσα σε γειτονικά pixel της εικόνας (μπορείτε να χρησιμοποιήσετε την εντολή *diff* – βεβαιωθείτε ότι έχετε μετατρέψει το διάνυσμα *g* σε double) δημιουργώντας το διάνυσμα *d* των διαφορών των pixels.

(2) Βρείτε πόσες διακριτές τιμές περιέχει η εικόνα διαφορών (το διάνυσμα *d* αντιστοιχεί στην πραγματικότητα στην εικόνα διαφορών) υπολογίζοντας το ιστόγραμμα του διανύσματος *d* (εντολή *hist*).

(3) Με βάση τη μορφή των ιστογραμμάτων των ερωτημάτων ΙΙ.(3) και ΙΙΙ.(2) ποια από τις εικόνες *f* και *d* συμπιέζεται περισσότερο;

(4) Υπολογίστε την εντροπία του διανύσματος *d* (δηλαδή της εικόνας διαφορών).

(5) Εφαρμόστε κωδικοποίηση μήκους διαδρομής για το διάνυσμα *d*. Κάθε μη μηδενική τιμή του *d* στα διαστήματα [-127 -1], [1 127] κωδικοποιείται ως προσημασμένος ακέραιος. Ο κωδικός 00000000 χρησιμοποιείται ως σύμβολο που δηλώνει ακολουθία μηδενικών ακολουθούμενη από το πλήθος των συνεχόμενων μηδενικών. Δώστε την κωδικοποίηση των πρώτων 26 τιμών του διανύσματος *d*.

(6) Υπολογίστε τη συμπίεση που επιτυγχάνεται με την πιο πάνω τεχνική.

(7) Κάτω από ποιες προϋποθέσεις η ανωτέρω τεχνική είναι χωρίς απώλειες;

(8) Περιγράψτε τη μεθοδολογία αποκωδικοποίησης της εικόνας.

Παράδειγμα: Σύμφωνα με τα παραπάνω η ακολουθία 3 0 0 0 0 0 -12 0 0 0 0 0 0 0 8 8 θα κωδικοποιηθεί ως:

3 !5 -12 !6 8 8 (ο κωδικός για το ! είναι ο 00000000)

III. Αριθμητική κωδικοποίηση

Δίνονται τα παρακάτω τμήματα ψευδοκώδικα τα οποία επιδεικνύουν τη λειτουργία της Αριθμητικού Κωδικοποιητή. Γράψτε της ανάλογες συναρτήσεις (m-files) σε Matlab που υλοποιούν τον κωδικοποιητή. Υποθέστε ότι ο κωδικοποιητής διαθέτει ένα αρχείο με της πιθανότητες των συμβόλων και τα αντίστοιχα διαστήματα (το $ είναι το σύμβολο τερματισμού της ακολουθίας).

Symbol	Probability	Range
A	0.2	[0 0.2)
B	0.1	[0.2 0.3)
C	0.2	[0.3 0.5)
D	0.05	[0.5 0.55)
E	0.3	[0.55 0.85)
F	0.05	[0.85 0.9)
$	0.1	[0.9 1.0)

Ελέγξτε την αποκωδικοποίηση της συμβολοσειράς CDEF$.

Estimation of Low and High values	Generating Codewrods

BEGIN
low=0.0; high=1.0; range=1.0;
while (symbol !=terminator)
{
get(symbol);
low = low + range*Range_low(symbol);
high = low + range*Range_high(symbol);
range = high – low;
}
output a code so that low <= code < high
END
	BEGIN
	code=0;
	k=1;
	while (value(code) < low)
	{
	assign 1 to the k-th binary fraction bit;
	if (value(code) > high)
	replace the k-th bit by 0;
	k = k+1;
	}
	END

Απαντήσεις

Πίνακας Ι.1: Κωδικοποίηση συμβολοσειράς ABABBABCABABBA

s	c	Output Code	Output Codeword

Πίνακας Ι.2: Ενημερωμένο λεξικό

String	Code	Codeword

Πίνακας Ι.3: Αποκωδικοποίηση κωδικών 1 2 4 5 2 3 4 6 1

s	k	Output string

Πίνακας Ι.4: Λεξικό δέκτη

String	Code	Codeword

Ι.(2): Βαθμός συμπίεσης:

Ι.(4): Η σημασία της εντολής

if (**entry** ==NULL) **entry = s+s[0]**;

II.(1): Εντολή υπολογισμού διανύσματος διαφοράς:

III.(2): Αριθμός διακριτών τιμών διαφοράς:

Ιστόγραμμα:

III.(3): Καλύτερα συμπιέζεται η εικόνα επειδή:

III.(4): Η =

III.(5): Κωδικοποίηση πρώτων 26 τιμών του *d*:

III.(6): Βαθμός Συμπίεσης:

III.(7): Προϋποθέσεις για να είναι η τεχνική χωρίς απώλειες:

III.(8): Περιγραφή αποκωδικοποίησης: