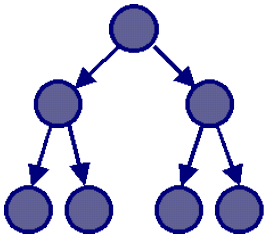


5.5 Εφαρμογή Δυαδικών Δέντρων: Κωδικοί Huffman

Τα δυαδικά δέντρα μπορούν να χρησιμοποιηθούν σε πολλά προβλήματα **κωδικοποίησης**(**encoding**) και **αποκωδικοποίησης** (**decoding**). Ένα παράδειγμα είναι η κωδικοποίηση και αποκωδικοποίηση ενός μηνύματος που μεταφέρεται με σήματα Morse, όπου κάθε χαρακτήρας αναπαρίσταται με μια σειρά από τελείες και παύλες. Για παράδειγμα, το γράμμα A συμβολίζεται ως .-, το γράμμα B ως -..., το γράμμα C ως -.-., κλπ. Όταν χρησιμοποιείται ο κώδικας Morse, το πλήθος των συμβόλων που απαιτείται για την παράσταση κάθε γράμματος δεν είναι σταθερό. Το A, για παράδειγμα θέλει δύο σύμβολα, ενώ το B και το C θέλουν από τέσσερα. Ένας τέτοιος κώδικας ονομάζεται **κώδικας μεταβλητού μήκους** (**variable-length code**). Στη συνέχεια θα ασχοληθούμε με τους **Κώδικες Huffman**, που είναι επίσης μεταβλητού μήκους κώδικες.

Η βασική ιδέα στους κώδικες μεταβλητού μήκους είναι να χρησιμοποιούνται μικρότεροι κωδικοί για τους χαρακτήρες που εμφανίζονται πιο συχνά και μεγαλύτεροι γι' αυτούς που εμφανίζονται λιγότερο συχνά. Ο σκοπός είναι να μειωθεί το αναμενόμενο μήκος του κωδικού ενός χαρακτήρα ώστε να περιοριστεί και το πλήθος των bits που απαιτούνται για την μετάδοση του κωδικοποιημένου μηνύματος.

Υποθέτουμε ότι για ένα σύνολο χαρακτήρων $C_1,C_2,...,C_n$, υπάρχει ένα σύνολο βαρών $w_1,w_2,...,w_n$, που σχετίζεται με αυτούς τους χαρακτήρες, δηλαδή w_i είναι το βάρος που αντιστοιχεί στον χαρακτήρα C_i και είναι ένα μέτρο (πιθανότητα ή σχετική συχνότητα) του πόσο συχνά εμφανίζεται αυτός ο χαρακτήρας σε μηνύματα που πρόκειται να κωδικοποιηθούν. Αν με $l_1,l_2,...,l_n$ συμβολίσουμε τα μήκη των κωδικών των χαρακτήρων $C_1,C_2,...,C_n$ αντίστοιχα, τότε το αναμενόμενο μήκος του κώδικα για καθένα από τους χαρακτήρες αυτούς είναι:

αναμενόμενο μήκος = $w_1 l_1 + w_2 l_2 + ... + w_n l_n = \sum_{i=1}^{n} w_i l_i$

Έστω, για παράδειγμα, ότι έχουμε τους χαρακτήρες A, B, C, D και E και ότι οι πιθανότητες εμφάνισης αυτών των χαρακτήρων είναι τα βάρη που φαίνονται στον ακόλουθο πίνακα:

Χαρακτήρας	Βάρος
A	0.2
B	0.15
C	0.05
D	0.15
E	0.45

Στον πίνακα που ακολουθεί φαίνεται η αναπαράσταση αυτού του συνόλου χαρακτήρων σε κώδικα Morse με τελείες και παύλες στην 2η στήλη, ενώ στην 3η στήλη οι τελείες έχουν αντικατασταθεί με 0 και οι παύλες με 1:

Χαρακτήρας	Κωδικός Morse(1)	Κωδικός Morse(2)
A	.-	01
B	-...	1000
C	-.-.	1010
D	-..	100
E	.	0

Το αναμενόμενο μήκος κώδικα για τους παραπάνω πέντε χαρακτήρες είναι:

$$2*0.2 + 4*0.15 + 4*0.05 + 3*0.15 + 1*0.45 = 2.1$$

Μια σημαντική ιδιότητα μερικών συστημάτων κωδικοποίησης είναι ότι είναι **άμεσα αποκωδικοποιήσιμα** (**immediately decodable**), δηλαδή καμία ακολουθία από bits, που αναπαριστά έναν χαρακτήρα, δεν αποτελεί πρόθεμα κάποιας μεγαλύτερης ακολουθίας bits, που να αναπαριστά έναν άλλο χαρακτήρα. Αυτό έχει ως συνέπεια, η λήψη μιας ακολουθίας από bits να αποκωδικοποιείται αμέσως στον αντίστοιχο χαρακτήρα, χωρίς να χρειάζεται να περιμένουμε επόμενα bits για να σχηματίσουμε μια μεγαλύτερη ακολουθία, που πιθανόν να αντιστοιχεί σε άλλον χαρακτήρα. Την ιδιότητα αυτή δεν την έχει ο κώδικας Morse κι αυτό φαίνεται στον παραπάνω πίνακα. Ο κωδικός του D (100) είναι πρόθεμα του αντίστοιχου για τον χαρακτήρα B (1000) και ο κωδικός του χαρακτήρα E (0) αποτελεί πρόθεμα του χαρακτήρα A (01). Μια κωδικοποίηση των χαρακτήρων A, B, C, D και E με το ίδιο μήκος κωδικών, όπως και παραπάνω, αλλά που είναι συγχρόνως άμεσα αποκωδικοποιήσιμη, φαίνεται στον πίνακα που ακολουθεί:

Χαρακτήρας	Κωδικός
A	01
B	0001
C	0000
D	001
E	1

Για την κωδικοποίηση χαρακτήρων με τρόπο ώστε να έχουν το μικρότερο αναμενόμενο μήκος και να αποκωδικοποιούνται αμέσως, μπορεί να χρησιμοποιηθεί ο παρακάτω αλγόριθμος του D. A. Huffman:

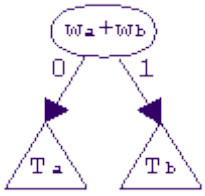
ΑΛΓΟΡΙΘΜΟΣ HUFFMAN

/*Δέχεται: Ένα σύνολο από *n* χαρακτήρες {*C1,C2,...,Cn*} και ένα σύνολο από βάρη {*w1,w2,...,wn*}, όπου *wi* είναι το βάρος του χαρακτήρα *Ci*.

Λειτουργία: Κατασκευάζει ένα δυαδικό κωδικό για το δοσμένο σύνολο χαρακτήρων, όπου το αναμενόμενο μήκος της ακολουθίας bits κάθε χαρακτήρα είναι το ελάχιστο.

Επιστρέφει: Μια συλλογή από *n* ακολουθίες bits που αναπαριστούν κωδικούς των χαρακτήρων.*/

1. Αρχικοποίησε μια λίστα από δυαδικά δέντρα ενός κόμβου που να περιέχουν τα βάρη $w_1,w_2,...,w_n$, ένα για κάθε χαρακτήρα του συνόλου $\{C_1,C_2,...,C_n\}$
2. **Για** *i* **από** 1 **μέχρι** *n*-1
   α. Βρες δύο δέντρα T_α και T_β σ' αυτήν την λίστα με ρίζες τα ελάχιστα βάρη w_α και w_β
   β. Αντικατέστησε τα δύο αυτά δέντρα με ένα δυαδικό δέντρο του οποίου η ρίζα είναι $w_\alpha+w_\beta$, τα υποδέντρα του είναι τα T_α και T_β και θέσε ετικέτες 0 και 1 στους δείκτες προς τα δύο αυτά υποδέντρα αντίστοιχα:

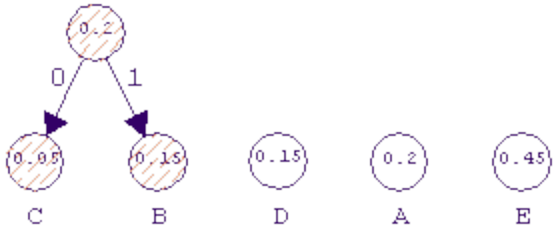


   Τέλος_επανάληψης
3. Ο κωδικός του χαρακτήρα C_i είναι η ακολουθία bits που σχηματίζεται από την διαδρομή στο τελικό δυαδικό δέντρο που ξεκινά από τη ρίζα και καταλήγει στο φύλλο C_i

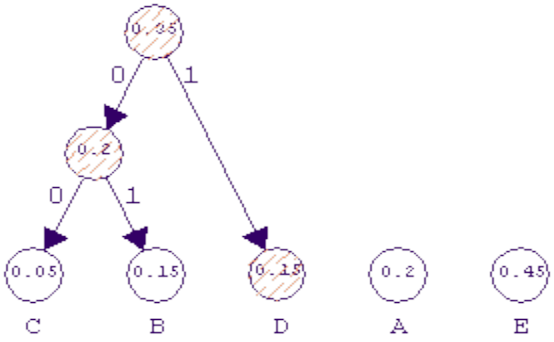
Ας εφαρμόσουμε τον αλγόριθμο Huffman στους χαρακτήρες A, B, C, D και E με βάρη αυτά που δόθηκαν παραπάνω. Ξεκινάμε κατασκευάζοντας μια λίστα από δυαδικά δέντρα ενός κόμβου, ένα για καθένα από τους πέντε χαρακτήρες:

Επειδή στο βήμα 2.α του παραπάνω αλγόριθμου αναζητούνται τα δέντρα με τα ελάχιστα βάρη, διατάσσουμε τα δέντρα ενός κόμβου σε αύξουσα διάταξη.

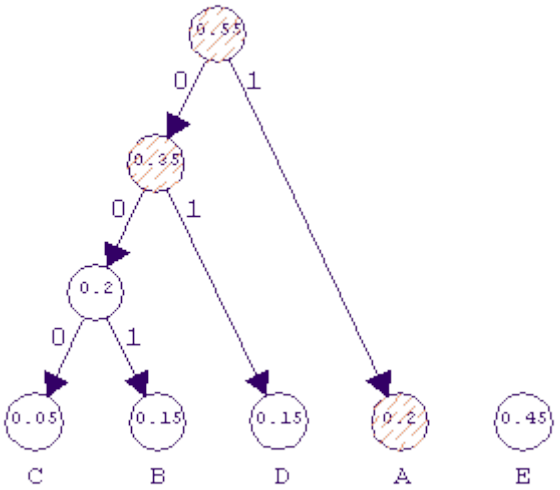
Τα δύο μικρότερα βάρη είναι αυτά που αντιστοιχούν στους χαρακτήρες C και B(εναλλακτικά μπορούμε να πάρουμε τα βάρη των C και D, γιατί οι χαρακτήρες B και D έχουν ίδιο βάρος). Επομένως, τα δύο πρώτα δέντρα που επιλέγονται είναι τα δέντρα-κόμβοι που αντιστοιχούν στους χαρακτήρες C και B και κατασκευάζεται ένα δέντρο με βάρος 0.05+0.15=0.2 όπως δείχνει το ακόλουθο σχήμα:



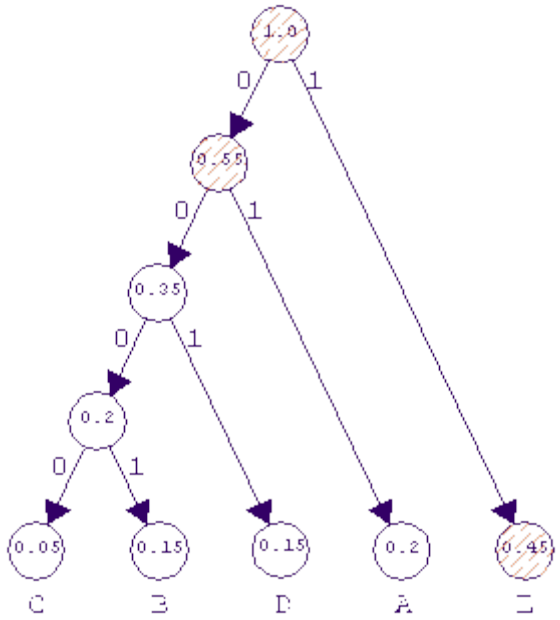
Τώρα θα διαλέξουμε πάλι τα δύο μικρότερα βάρη από τη λίστα των τεσσάρων πλέον δυαδικών δέντρων του παραπάνω σχήματος. Τα βάρη είναι τώρα 0.2, 0.15, 0.2 και0.45. Από αυτά διαλέγουμε το δέντρο με βάρος 0.15 (μικρότερο από όλα) και ένα από τα δύο δέντρα που έχουν βάρος 0.2, για να προκύψει ένα δέντρο με βάρος0.15+0.2=0.35. Έστω ότι διαλέγουμε το πρώτο, όπως φαίνεται παρακάτω:



Από τα τρία δυαδικά δέντρα της λίστας επιλέγουμε πάλι τα δύο με τα μικρότερα βάρη. Οι τιμές των βαρών είναι 0.35, 0.2 και 0.45, οπότε η μόνη μας επιλογή είναι να συνδυάσουμε τα δέντρα με βάρη 0.35 και 0.2 και να προκύψει ένα δέντρο με βάρος0.35+0.2=0.55:



Τώρα στη λίστα έχουν μείνει δύο δέντρα, τα οποία και συνδυάζονται για να προκύψει το τελικό δέντρο Huffman με βάρος 0.55+0.45=1.0 που φαίνεται παρακάτω:



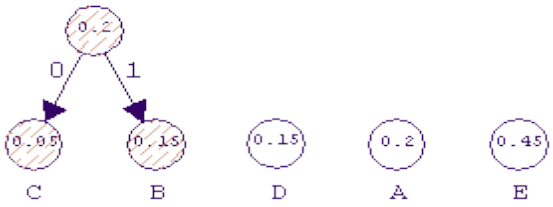
Οι κωδικοί Huffman που προκύπτουν για καθένα από τους πέντε χαρακτήρες είναι:

Χαρακτήρας	Κωδικός Huffman
A	01
B	0001
C	0000
D	001
E	1

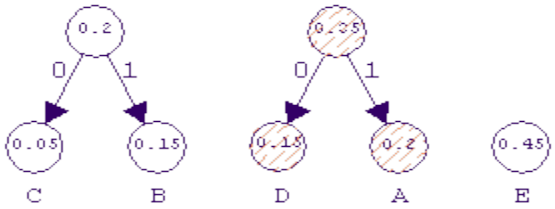
Το αναμενόμενο μήκος κώδικα για τους χαρακτήρες αυτούς είναι πάλι:

2*0.2 + 4*0.15 + 4*0.05 + 3*0.15 + 1*0.45 = 2.1

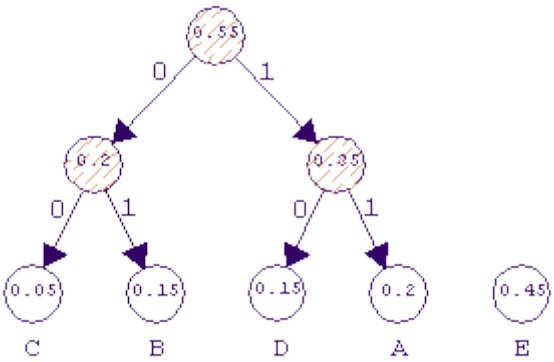
Το παραπάνω δέντρο Huffman δεν είναι το μοναδικό που θα μπορούσε να κατασκευαστεί για τους συγκεκριμένους χαρακτήρες και τα αντίστοιχα βάρη τους. Μετά από την κατασκευή του δυαδικού δέντρου με βάρος 0.2:



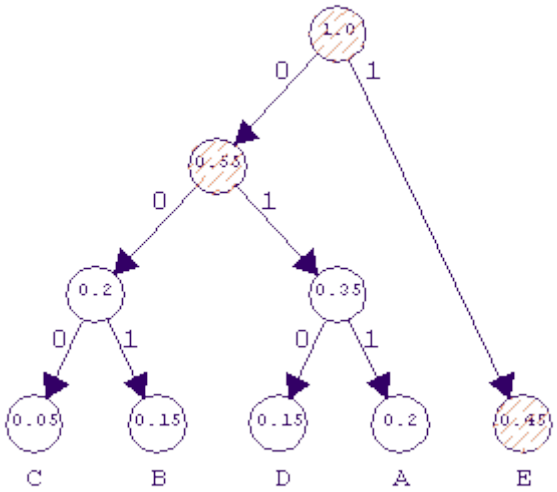
θα μπορούσαμε να επιλέξουμε τα δέντρα που αντιστοιχούν στους χαρακτήρες D και A και να κατασκευάσουμε ένα δέντρο με βάρος 0.15+0.2=0.35, όπως φαίνεται στο παρακάτω σχήμα:



Στη συνέχεια, από τη λίστα με τα τρία δέντρα, επιλέγουμε τα δύο με τα μικρότερα βάρη, οπότε προκύπτει το δέντρο που φαίνεται στο ακόλουθο σχήμα:



Τέλος, συνδυάζουμε τα δύο δέντρα με βάρη 0.55 και 0.45 και έχουμε κατασκευάσει το τελικό δέντρο Huffman, που διαφέρει από το προηγούμενο:



Οι κωδικοί Huffman που προκύπτουν από το τελευταίο σχήμα είναι:

Χαρακτήρας	Κωδικός Huffman
A	011
B	001
C	000
D	010
E	1

Σε κάθε χαρακτήρα αντιστοιχεί ένας κόμβος-φύλλο στο δέντρο Huffman και υπάρχει μια μοναδική διαδρομή από τη ρίζα προς κάθε φύλλο. Αυτό έχει ως αποτέλεσμα, καμία ακολουθία bits, που αποτελεί τον κωδικό ενός χαρακτήρα, να αποτελεί πρόθεμα μιας μεγαλύτερης ακολουθίας από bits που να αποτελεί τον κωδικό ενός άλλου χαρακτήρα. Επομένως, οι κωδικοί Huffman έχουν την ιδιότητα της άμεσης αποκωδικοποίησης και εξ αιτίας αυτής της ιδιότητάς τους είναι εύκολο να κατασκευαστεί ένας αλγόριθμος αποκωδικοποίησης:

ΑΛΓΟΡΙΘΜΟΣ ΑΠΟΚΩΔΙΚΟΠΟΙΗΣΗΣ HUFFMAN

/*Δέχεται: Ένα δέντρο Huffman και μια ακολουθία από bits που αναπαριστά ένα μήνυμα, το οποίο έχει κωδικοποιηθεί με χρήση του δέντρου Huffman.

Λειτουργία: Αποκωδικοποιεί το μήνυμα.

Επιστρέφει: Το αποκωδικοποιημένο μήνυμα.*/

1. Αρχικοποίησε έναν δείκτη *p* να δείχνει στην ρίζα του δέντρου Huffman

2. **Όσο** δεν έχει βρεθεί το τέλος του μηνύματος **επανάλαβε**

   α. Έστω ότι *x* είναι το επόμενο bit στην ακολουθία χαρακτήρων

β. **Αν** *x* = 0 **τότε**

p <- *p* NUL->*LChild*

Αλλιώς

p <- *p* NUL->*RChild*

Τέλος _αν

γ. **Αν** ο *p* δείχνει σε φύλλο **τότε**

i. Εμφάνισε το χαρακτήρα που αντιστοιχεί σ' αυτό το φύλλο

ii. Να επαναφέρεις τον δείκτη *p* ώστε να δείχνει στην ρίζα του δέντρου Huffman

Τέλος _αν

Τέλος_επανάληψης

Για να δούμε πώς εφαρμόζεται ο παραπάνω αλγόριθμος, έστω ότι θέλουμε να αποκωδικοποιήσουμε το μήνυμα:

0 0 1 1 0 1 0

που έχει κωδικοποιηθεί με χρήση του δεύτερου δέντρου Huffman που κατασκευάσαμε. Ο δείκτης p ξεκινά από τη ρίζα και ακολουθεί το μονοπάτι 001 που οδηγεί στο γράμμα B:

0 0 1 1 0 1 0
B

Στη συνέχεια, ο δείκτης p επανέρχεται και δείχνει στη ρίζα του δέντρου. Το bit 1 που ακολουθεί οδηγεί κατευθείαν στο φύλλο που αντιστοιχεί στο χαρακτήρα E:

0 0 1 **1** 0 1 0
B E

Ο δείκτης p ξεκινά πάλι από τη ρίζα του δέντρου και διαγράφει το μονοπάτι 010, που οδηγεί στο φύλλο για τον χαρακτήρα D:

0 0 1 **1** **0 1 0**
B E D

Ένα πρόγραμμα που υλοποιεί τον παραπάνω αλγόριθμο Huffman είναι το HuffmanDecoding.c.