Εργασία
Ψηφιακών Τηλεπικοινωνιών

Κυριακόπουλος Αριστομένης[1]

Πανεπιστήμιο Πατρών
Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών και Πληροφορικής
up1054307@upnet.gr

Περίληψη Σκοπός της εργασίας είναι η δημιουργία συναρτήσεων για την κατανόηση της κωδικοποίησης Huffman, της κωδικοποίησης ΠΤΜ και την μελέτη απόδοσης ομόδηνου ζωνοπερατού συστήματος Μ-ΦΣΚ. Παρουσιάζονται παρακάτω τα αποτελέσματα της προσπάθειας μου.

1 Ερώτημα 1 - Κωδικοποίηση Huffman

Η κωδικοποίηση Huffman αποτελεί μια βέλτιστη κωδικοποίηση διακριτών πηγών και χρησιμοποιείται για την συμπίεση δεδομένων χωρίς χάσιμο πληροφορίας. Παρακάτω παρατίθεται οι δικές μου θεμελιώσεις των 3 βασικών συναρτήσεων.

1.1 Κωδικοποίηση πηγής κώδικα

Huffman Dictionary Η συνάρτηση huffmanDict λαμβάνει ως είσοδο 2 διανύσματα, το 1 με τους αγγλικούς χαρακτήρες και το άλλο με την πιθανότητα του κάθε γράμματος από το αρχείο cvxopt.txt. Αρχικά ταξινομούνται όλες οι πιθανότητες σε αύξουσα σειρά και λαμβάνονται οι δύο χαρακτήρες με την μικρότερη πιθανότητα να εμφανιστούν. Αυτοί εισέρχονται σε ένα κοινό κόμβο με το άθροισμα των πιθανοτήτων τους και εισέρχονται ξανά στα αντίστοιχα διανύσματά τους. Αυτό γίνεται μέχρι η πιθανότητες να φτάσει στο 1. Για το δοθέν κείμενο η έξοδος της συνάρτησης είναι η εξής:

```
Columns 1 through 8

  {'a'   }    {'b'     }    {'c'    }    {'d'    }    {'e'  }    {'f'     }    {'g'     }    {'h'    }
  {'0110'}    {'101111'}    {'10001'}    {'10110'}    {'110'}    {'100100'}    {'101110'}    {'10011'}

Columns 9 through 16

  {'i'   }    {'j'       }    {'k'      }    {'l'    }    {'m'    }    {'n'   }    {'o'   }    {'p'    }
  {'0101'}    {'111110011'}    {'11111000'}    {'01000'}    {'10000'}    {'0111'}    {'0000'}    {'11110'}

Columns 17 through 24

  {'q'       }    {'r'   }    {'s'   }    {'t'   }    {'u'     }    {'v'     }    {'w'     }    {'x'     }
  {'111110010'}    {'1010'}    {'1110'}    {'0001'}    {'010010'}    {'100101'}    {'01000110'}    {'01001111'}

Columns 25 through 27

  {'y'     }    {'z'     }    {' '  }
  {'111111'}    {'1111101'}    {'001'}
```

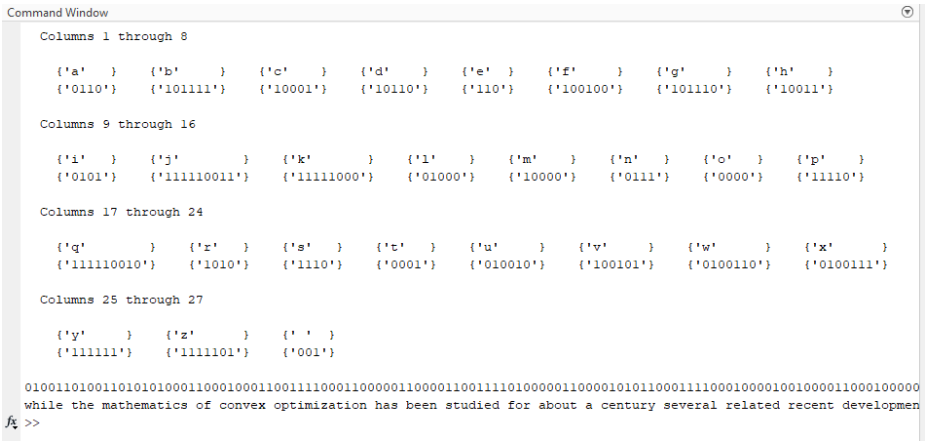
Εικ 1: Huffman Dictionary Output.

Huffman Encoding Η συνάρτηση huffmanEnc λαμβάνει ως είσοδο το αρχείο που θέλουμε να κωδικοποιηθεί και το dictionary που δημιουργήθηκε από την συνάρτηση huffmanDict. Όσο το αρχείο εισόδου δεν είναι κενό, συγκρίνει έναν προς έναν τους χαρακτήρες με την αντίστοιχη κωδικοποίησή τους και δημιουργεί ένα διάνυσμα εξόδου με την ακολουθία από bit που το αναπαριστούν. Τα αρχικά ψηφία φαίνοντα παρακάτω και μπορούν να επαληθευτούν από το dictionary.

```
Command Window
  Columns 1 through 8

    {'a'   }    {'b'     }    {'c'    }    {'d'    }    {'e'  }    {'f'     }    {'g'     }    {'h'    }
    {'0110'}    {'101111'}    {'10001'}    {'10110'}    {'110'}    {'100100'}    {'101110'}    {'10011'}

  Columns 9 through 16

    {'i'   }    {'j'       }    {'k'      }    {'l'    }    {'m'    }    {'n'   }    {'o'   }    {'p'    }
    {'0101'}    {'111110011'}    {'11111000'}    {'01000'}    {'10000'}    {'0111'}    {'0000'}    {'11110'}

  Columns 17 through 24

    {'q'       }    {'r'   }    {'s'   }    {'t'   }    {'u'     }    {'v'     }    {'w'     }    {'x'     }
    {'111110010'}    {'1010'}    {'1110'}    {'0001'}    {'010010'}    {'100101'}    {'01000110'}    {'01001111'}

  Columns 25 through 27

    {'y'     }    {'z'     }    {' '  }
    {'111111'}    {'1111101'}    {'001'}

0100110100110101010001100010001100111110001100000110000110011110100000110000101011000111100010000100100001100010000
```

Εικ 2: Huffman Encoding Output.

Huffman Decoding Η συνάρτηση huffmanDec λαμβάνει ως είσοδο το κωδικοποιημένο διάνυσμα όπως επίσης και το dictionary που έχει δημιοργηθεί. Διαβάζοντας έναν έναν τα bit εισόδου, δημιουργεί ένα προσωρινό λεξιλόγιο με σκοπό να ταυτοποιήσει την ακολουθία από bit που αντιστοιχεί στο κάθε γράμμα. Όταν το εντοπίσει, το γράμμα εισέρχεται στο vector εξόδου. Παρακάτω φαίνονται τα αποτελέσματα όλων των συναρτήσεων:



Εικ 3: Huffman Decoding Output.

1.2 Αποδοτικότητα κώδικα

Για να υπολογίσουμε την αποδοτικότητα του κώδικα, πρέπει πρώτα να υπολογιστούν η εντροπία του κώδικα όπως επίσης και το μέσο μήκος κώδικα. Ο τύπος της εντροπίας είναι:

$$\sum iPx(i) * log(Px(i)) \tag{1}$$

και ο τύπος του μέσου μήκους κώδικα είναι:

$$\sum iPx(i) * Length(i)) \tag{2}$$

Υπολογίζοντας αυτά μπορούμε να βρούμε την αποδοτικότητα του κώδικα αφού αποτελεί απλώς μια διαίρεση μεταξύ εντροπίας και μήκους. Για τον δικό μου κώδικα τα αποτελέσματα φαίνονται παρακάτω:

```
while the mathematics of convex optimization has been s

entropy =

    4.1347


avgLenCode =

    4.1630


efficiency =

    0.9932

>>
```

Εικ 4: Huffman Efficiency

Όπως φαίνεται η αποδοτικότητα του κώδικα είναι στο 99% το οποίο είναι και ένα αποτέλεσμα που περιμέναμε αφού οι πιθανότητες που χρησιμοποιήσαμε προέρχονται από το κείμενο που κωδικοποιούμε.

1.3 Κωδικοποίηση με πιθανότητες συμβόλων αγγλικής γλώσσας

Χρησιμοποιώντας το αρχείο frequencies.txt για τις πιθανότητες των συμβόλων λαμβάνουμε το εξής αποτέλεσμα:

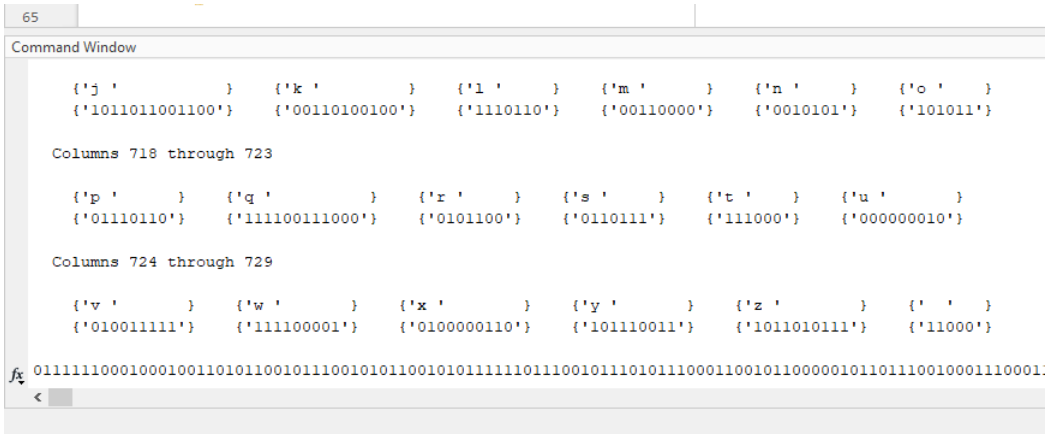
```
entropy =

    4.1679


avgLenCode =

    4.1875


efficiency =

    0.9953

>>
```

Εικ 5: Huffman Efficiency

Παρατηρούμε ότι το μέσο μήκος κώδικα είναι λίγο μεγαλύτερο, το οποίο οφείλεται στο γεγονός ότι οι δοσμένες πιθανότητες είναι γενικευμένες για την αγγλική γλώσσα και δεν είναι προσαρμοσμένες πάνω στο δοθέν κείμενο.

1.4 Δεύτερη τάξη επέκταση πηγής A

Αρχικά δημιουργούμε την δεύτερη τάξη επέκταση πηγής A, η οποία είναι όλοι οι συνδυασμοί γραμμάτων μεταξύ τους και υπολογίζουμε τις πιθανότητες κάθε συνόλου 2 χαρακτήρων πολλαπλασιάζοντας τις πιθανότητες του καθενός. Στο τέλος καταλήγουμε με 729 συνδυασμούς γραμμάτων (δηλαδή $26\hat{2}$). Η τελική κωδικοποιηση φαίνεται παρακάτω μαζί με τα τελευταία ζεύγη του λεξιλογίου:



Εικ 6: Huffman Second Order

Παρατηρούμε ότι παρόλο που το μέσο μήκος κώδικα έχει διπλασιαστεί, το οποίο είναι και ένα αποτέλεσμα που περιμέναμε λόγω της αύξησης των χαρακτήρων από 27 σε 729, η απόδοση της κωδικοποίησης είναι μεγαλύτερη και η συμπίεση του κειμένου είναι μεγαλύτερη. Επίσης η εντροπία μια διακριτής πηγής με την αντίστοιχη n τάξης επέκτασής της ακολουθεί τον τύπο:

$$H(\Phi n) = nH(\Phi) \tag{3}$$

Το οποίο επαληθεύεται με τα δεδομένα μας, αφού η εντροπία της δεύτερης τάξης επέκταση της πηγής A είναι διπλάσια.