Πανεπιστήμιο Ιωαννίνων

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Πληροφορικής και Τηλεπικοινωνιών

Πτυχιακή Εργασία

Μοντέλο Διανυσματικού Χώρου

Σεραφείμ Βίτκος

Υπεύθυνος Καθηγητής: Αδάμ Σταύρος

Σεπτέμβρης 2019

Ευχαριστίες

Με την ολοκλήρωση της παρούσας πτυχιακής εργασίας θα ήθελα να ευχαριστήσω θερμά τον υπεύθυνο καθηγητή μου κύριο Σταύρο Αδάμ για την πολύτιμη καθοδήγηση και βοήθεια σε όλη την διάρκεια της εργασίας. Επίσης, ένα μεγάλο ευχαριστώ στην οικογένειά μου για την αμέριστη συμπαράσταση.

Περίληψη

Η Μηχανική Μάθηση είναι ο κλάδος της επιστήμης που μελετά την δυνατότητα εκπαίδευσης του Ηλεκτρονικού Υπολογιστή να λαμβάνει αποφάσεις και να εξάγει συμπεράσματα μέσω των πληροφοριών που του διατίθενται από τον χρήστη.

Η παρούσα εργασία έχει ως σκοπό την μελέτη αλγορίθμων και προγραμμάτων υπολογιστών που επεξεργάζονται και αναλύουν έγγραφα κειμένου. Οι αλγόριθμοι αυτοί υπάγονται στην κατηγορία του Μοντέλου Διανυσματικού χώρου.

Το Μοντέλο Διανυσματικού Χώρου χρησιμοποιεί αλγεβρικές τεχνικές και μεθόδους Μηχανικής Μάθησης για να μετατρέψει τα κείμενα σε αριθμητικά δεδομένα δηλαδή στην γλώσσα που μπορεί να επεξεργαστεί ο ηλεκτρονικός υπολογιστής. Έπειτα εξάγουν αποτελέσματα ως προς το νόημα ή/και την δομή του κειμένου.

Το Μοντέλο Διανυσματικού Χώρου έχει μεγάλο εύρος χρησιμότητας όπως στις μηχανές αναζήτησης ή στο φιλτράρισμα κειμένων με ανεπιθύμητο περιεχόμενο.

Λέξεις κλειδιά: Μοντέλο διανυσματικού χώρου, Ανάκτηση πληροφορίας, Επεξεργασία εγγράφων, Μηχανική Μάθηση, Διανύσματα λέξεων, Διανύσματα εγγράφων

Abstract

Machine Learning is the scientific and technological area that studies the possibility of training a Personal Computer to be able to make decisions and come to conclusions using the information which are provided by the user.

The purpose of this essay is the research of algorithms and computer programs that are processing and analyzing text documents. These algorithms fall within Vector Space Model.

Vector Space Model uses algebraic techniques and Machine Learning methods in order to translate the documents into the language that a Personal Computer can process. Subsequently, they export results in accordance with the substance or/and the structure of the document.

Vector Space Model has a huge range of usage like in search engines or in filtering texts to isolate unwanted content.

Keywords: Vector Space Model, Information Retrieval, Document Processing, Machine Learning, Word Vectors, Document Vectors

Περιεχόμενα

Εισαγωγή

Τα μοντέλα ανάκτησης πληροφοριών είναι μία συλλογή προγραμμάτων λογισμικού που αποθηκεύουν και διαχειρίζονται πληροφορίες εγγράφων. Η ύπαρξη της πληροφορίας δεν έχει καμία ουσία χωρίς κάποιον να την διοχετεύσει. Πολλές εφαρμογές που χειρίζονται πληροφορίες στο διαδίκτυο θα ήταν εντελώς ανεπαρκείς χωρίς την υποστήριξη της τεχνολογίας των μοντέλων ανάκτησης πληροφοριών.

Στην εποχή της συνεχούς εξέλιξης της τεχνολογίας και του παγκόσμιου ιστού οι πληροφορίες είναι απεριόριστες και για να τις παραλάβει ο δέκτης πρέπει να βρεθεί ένας τρόπος μετάδοσης και επεξεργασίας. Πώς θα ήταν δυνατή η εύρεση πληροφοριών στον παγκόσμιο ιστό αν δεν υπήρχαν οι μηχανές αναζήτησης; Πώς θα μπορούσαν οι χρήστες να προστατευτούν απο το κακόβουλο περιεχόμενο που πλήτει το ηλεκτρονικό τους ταχυδρομείο χωρίς το φιλτράρισμα των ανεπιθύμητων μηνυμάτων;

Μεγάλο μέρος της εξέλιξης της τεχνολογίας ανάκτησης πληροφοριών, όπως οι μηχανές αναζήτησης και τα φίλτρα ανεπιθύμητης αλληλογραφίας, απαιτούν ένα συνδυασμό πειραματισμού και θεωρίας. Τόσο ο πειραματισμός όσο και οι αυστηροί έλεγχοι, απαιτείται να συμβαδίζουν με την αύξηση του όγκου των ιστοσελίδων και των μηνυμάτων ηλεκτρονικού ταχυδρομείου. Επιπλέον, στην πράξη απαιτείται συνεχής βελτίωση και προσαρμογή της τεχνολογίας ώστε να μειωθούν οι επιπτώσεις από ανθρώπους που προσπαθούν να χειραγωγήσουν την τεχνολογία όπως οι spammers μέσω του ηλεκτρονικού ταχυδρομείου. Ωστόσο, ο πειραματισμός πρέπει να συμβαδίζει με τη θεωρία ώστε τα νέα προβλήματα και οι νέες προκλήσεις που προκύπτουν να μπορούν να αντιμετωπιστούν. Τελικά, ποιά είναι η θεωρία της ανάκτησης πληροφοριών;

Δεν υπάρχει μια και μοναδική πειστική απάντηση σε αυτή την ερώτηση. Υπάρχουν πολλές θεωρίες, και διαφορετικά μοντέλα, κάθε ένα από τα οποία είναι χρήσιμο για την ανάπτυξη ορισμένων εργαλείων ανάκτησης πληροφοριών, αλλά όχι τόσο χρήσιμο για την ανάπτυξη άλλων. Για να κατανοήσει κανείς τη θεωρία ανάκτησης πληροφοριών, είναι σημαντικό να μελετήσει αυτά τα διαφορετικά μοντέλα.

Η οργάνωση της εργασίας έχει ως εξής:

Στο πρώτο κεφάλαιο, παρουσιάζονται και αναλύονται ορισμένα από τα πιο σημαντικά μοντέλα ανάκτησης πληροφοριών (Το Δυαδικό μοντέλο - Boolean Model και το πιθανοτικό μοντέλο ανάκτησης πληροφοριών).

Το δεύτερο κεφάλαιο επικεντρώνεται στην ανάλυση του Διανυσματικού Μοντέλου (Vector Space Model), παραθέτοντας ορισμένες σημαντικές πληροφορίες για το εν λόγω μοντέλο και τους τρόπους λειτουργίας του.

Στο τρίτο κεφάλαιο γίνεται μία προσωπική υλοποίηση εμβαθύνοντας στις τεχνικές ανάκτησης και επεξεργασίας πληροφοριών που αναφέρθηκαν στα προηγούμενα δύο κεφάλαια. Τα κείμενα τα οποία χρησιμοποιήθηκαν στην υλοποίηση αυτή επιλέχθηκαν τυχαία (Textfiles, 2019) έχοντας όμως κάποια σχετικές θεματικές ενότητες.

Τέλος, παρατίθενται τα συμπεράσματα καθώς και η άποψη του συγγραφέα για την συγκεκριμένη πτυχιακή αλλά και για περαιτέρω έρευνα.

Κεφάλαιο 1ο. Μοντέλα άντλησης και ανάκτησης πληροφοριών

Εισαγωγή

Στο κεφάλαιο αυτό γίνεται εισαγωγή στις έννοιες που απασχολούν την πτυχιακή. Ορίζονται οι έννοιες και το πλαίσιο και γίνεται ανασκόπηση της βιβλιογραφίας. Για την καλύτερη κατανόηση, χρησιμοποιούνται παραδείγματα.

1.1. Η τεχνική της ανάκτησης πληροφοριών

Η ανάκτηση πληροφοριών (Information Retrieval, IR) είναι ένα από τα κύρια αντικείμενα της επιστήμης των υπολογιστών και ασχολείται με την αναπαράσταση, την αποθήκευση και την πρόσβαση των πληροφοριών (Salton, McGill, 1983). Η διαδικασία αυτή σχετίζεται με την οργάνωση και την ανάκτηση πληροφοριών από μια τεράστια βάση δεδομένων (Yates and Neto, 1999). Η ανάκτηση πληροφοριών είναι η μεθοδολογία κατά την οποία ένας αρκετά μεγάλος όγκος δεδομένων συγκεντρώνεται, αναπαρίσταται και αποθηκεύεται με σκοπό την αποκάλυψη της γνώσης ως αποτέλεσμα του αιτήματος ή του ερωτήματος του χρήστη (Dong, Husain, Chang, 2008). Αυτή η διαδικασία διεξάγεται σε κάποια στάδια, κάποια από τα οποία αφορούν στην παρουσίαση των δεδομένων και της εξόδου και στην επιστροφή σχετικών πληροφοριών για την αξιοποίησή τους από το χρήστη. Τα ενδιάμεσα στάδια περιλαμβάνουν τις λειτουργίες αναζήτησης, φιλτραρίσματος, αντιστοίχισης και κατάταξης. Οι λειτουργίες αυτές που έχουν ως στόχο την εξαγωγή, την αποθήκευση, την επεξεργασία και την αναπαράσταση πληροφοριών μέσω ενός λογισμικού ονομάζονται μοντέλα πληροφοριών. Η επεξεργασία πληροφοριών προκύπτει σύμφωνα με το πως αντιλαμβάνεται μια τέτοια μέθοδος την πληροφορία. Ο κύριος στόχος ενός μοντέλου ανάκτησης πληροφοριών είναι η "εύρεση σχετικών πληροφοριών ή ενός εγγράφου που ικανοποιεί τις ανάγκες των χρηστών". Για την επίτευξη αυτού του στόχου, τα μοντέλα ανάκτησης πληροφοριών χρησιμοποιούν συνήθως τις εξής διαδικασίες:

- Στη διαδικασία δημιουργίας δεικτών, κατά την οποία το περιεχόμενο των εγγράφων αναδιαμορφώνεται.
- Στη διαδικασία φιλτραρίσματος, κατά την οποία όλες οι λέξεις φιλτράρονται.
- Στην αναζήτηση που είναι η κύρια διαδικασία των μοντέλων ανάκτησης πληροφοριών.

Υπάρχουν διάφορες τεχνικές για την ανάκτηση πληροφοριών που μπορούν να χρησιμοποιηθούν ανάλογα με τις ανάγκες των χρηστών. Υπάρχουν δύο στοιχειώδη μέτρα για την αξιολόγηση της ανάκτησης πληροφοριών (Yates, Neto, 1999).

- Ακρίβεια (θετική τιμή πρόβλεψης): Διαίρεση των ανακτηθέντων εγγράφων που σχετίζονται με τις ανάγκες των χρηστών.

$$\text{Ακρίβεια} = \frac{\{\sigma\chi\varepsilon\tau\iota\kappa\acute{\alpha}\ \acute{\varepsilon}\gamma\gamma\rho\alpha\varphi\alpha\}\Omega\{\alpha\nu\alpha\kappa\tau\eta\theta\acute{\varepsilon}\nu\tau\alpha\ \acute{\varepsilon}\gamma\gamma\rho\alpha\varphi\alpha\}}{\{\alpha\nu\alpha\kappa\tau\eta\theta\acute{\varepsilon}\nu\tau\alpha\ \acute{\varepsilon}\gamma\gamma\rho\alpha\varphi\alpha\}}$$

- Ανάκληση (ευαισθησία): Διαίρεση των εγγράφων που σχετίζονται με το ερώτημα χρήστη που ανακτώνται με επιτυχία.

$$\text{Ανάκληση} = \frac{\{\sigma\chi\varepsilon\tau\iota\kappa\acute{\alpha}\ \acute{\varepsilon}\gamma\gamma\rho\alpha\varphi\alpha\}\Omega\{\alpha\nu\alpha\kappa\tau\eta\theta\acute{\varepsilon}\nu\tau\alpha\ \acute{\varepsilon}\gamma\gamma\rho\alpha\varphi\alpha\}}{\{\sigma\chi\varepsilon\tau\iota\kappa\acute{\alpha}\ \acute{\varepsilon}\gamma\gamma\rho\alpha\varphi\alpha\}}$$

1.2. Μοντέλα ανάκτησης πληροφοριών

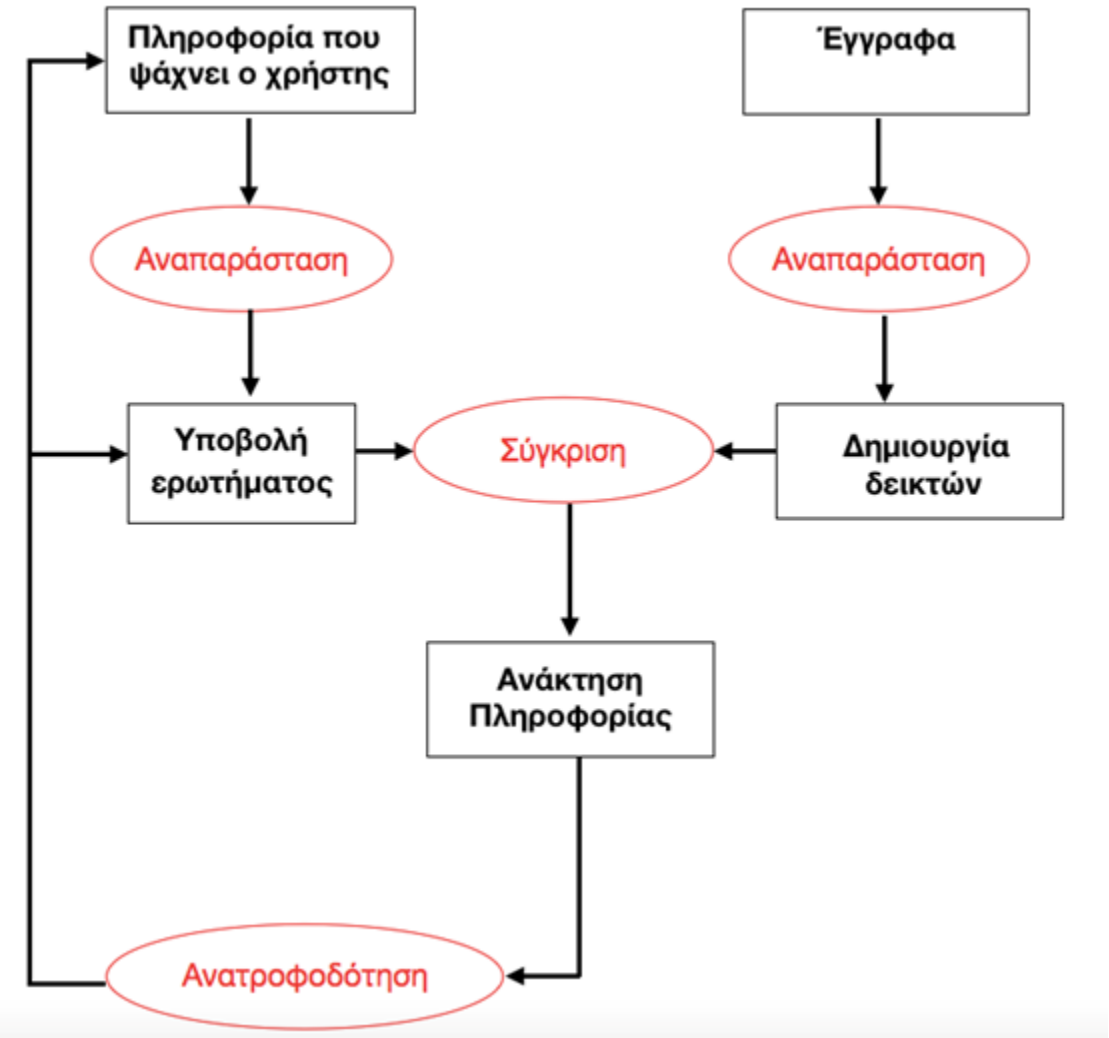
Ένα σύστημα ανάκτησης πληροφοριών, γνωστό και ως σύστημα διαχείρισης εγγράφων (Document Management System ή DMS), είναι το σύνολο των μονάδων λογισμικού που αποθηκεύει και διαχειρίζεται πληροφορίες εγγράφων (συχνά εγγράφων κειμένου), αλλά πιθανώς και πολυμέσων. Το σύστημα βοηθά τους χρήστες να βρουν τις πληροφορίες που χρειάζονται δίνοντας απάντηση σε ερωτήσεις και επιστρέφοντας την ύπαρξη και τη θέση των εγγράφων που ενδέχεται να περιέχουν τις επιθυμητές πληροφορίες. Ορισμένα έγγραφα που θα προταθούν από το σύστημα, θα ικανοποιήσουν ενδεχομένως τις απαιτήσεις του χρήστη. Αυτά τα έγγραφα ονομάζονται σχετικά έγγραφα. Ένα τέλειο σύστημα ανάκτησης θα ανακτούσε μόνο τα σχετικά έγγραφα και κανένα άσχετο έγγραφο. Ωστόσο, τα τέλεια συστήματα ανάκτησης δεν υπάρχουν, και πιθανώς να μην υπάρξουν, επειδή οι αναζητήσεις είναι ατελείς και η συνάφεια εξαρτάται από την υποκειμενική γνώμη του χρήστη. Στην πράξη, δύο χρήστες μπορούν να θέσουν το ίδιο ερώτημα σε ένα

σύστημα ανάκτησης πληροφοριών και να κρίνουν την καταλληλότητα των ανακτηθέντων εγγράφων διαφορετικά: Σε ορισμένους χρήστες τα αποτελέσματα θα καλύπτουν τις απαιτήσεις τους, ενώ σε άλλους όχι.

Υπάρχουν τρεις βασικές διαδικασίες που πρέπει να υποστηρίζει ένα σύστημα ανάκτησης πληροφοριών:

- η αναπαράσταση του περιεχομένου των εγγράφων,
- η αναπαράσταση των αναγκών του χρήστη και
- η σύγκριση των δύο αναπαραστάσεων.

Οι διαδικασίες αυτές απεικονίζονται στο Σχήμα 1.



Σχήμα 1. Διαδικασία ανατροφοδότησης

Στο σχήμα αυτό, τα τετράγωνα πλαίσια αντιπροσωπεύουν τα δεδομένα ενώ τα στρογγυλά πλαίσια αντιπροσωπεύουν διαδικασίες.

Η διαδικασία δημιουργίας δεικτών πραγματοποιείται εκτός σύνδεσης (off-line), δηλαδή ο τελικός χρήστης του συστήματος ανάκτησης πληροφοριών δεν εμπλέκεται άμεσα. Η διαδικασία δημιουργίας δεικτών οδηγεί στην αναπαράσταση του εγγράφου (Raman, Kumar, Venkatesan, 2012) και αποτελείται από τρία βασικά βήματα:

- ανάλυση του θέματος του εγγράφου,
- επιλογή όρων-δεικτών οι οποίοι μπορούν να περιγράψουν εν συντομία το θέμα (είτε διαλέγοντας λέξεις απευθείας από το κείμενο είτε δημιουργώντας όρους-συνώνυμα) και
- σύνδεση των δεικτών μέσω μιας συστηματικής σειράς προτεραιότητας ανάλογα με τις ανάγκες του κειμένου.

Συχνά, τα συστήματα ανάκτησης πλήρους κειμένου χρησιμοποιούν έναν, μάλλον τετριμμένο αλγόριθμο για την αναπαράσταση των δεικτών, για παράδειγμα έναν αλγόριθμο που προσδιορίζει λέξεις σε ένα αγγλικό κείμενο και μετατρέπει τα γράμματα σε πεζά. Η διαδικασία δημιουργίας δεικτών μπορεί να περιλαμβάνει και την πραγματική αποθήκευση του εγγράφου στο σύστημα, όμως συχνά τα έγγραφα αποθηκεύονται μόνο εν μέρει, για παράδειγμα μόνο ο τίτλος και η περίληψη, καθώς και πληροφορίες σχετικά με την πραγματική θέση του εγγράφου.

Οι χρήστες κάνουν αναζήτηση για να καλύψουν τόσο κάποιες ουσιαστικές ανάγκες καθώς επίσης και για λόγους διασκέδασης. Η διαδικασία με την οποία ο χρήστης αναπαριστά την ανάγκη του για συγκεκριμένη πληροφορία στο σύστημα, αναφέρεται ως διαδικασία διαμόρφωσης ερωτήματος. Η προκύπτουσα αναπαράσταση είναι το ερώτημα. Με μια ευρεία έννοια, η διατύπωση του ερωτήματος υποδηλώνει το διάλογο μεταξύ συστήματος και χρήστη, οδηγώντας όχι μόνο σε ένα κατάλληλο ερώτημα αλλά ενδεχομένως και στην καλύτερη κατανόηση από το χρήστη της πληροφορίας που έχει ανάγκη: Αυτό δηλώνεται από τη διαδικασία ανατροφοδότησης στο Σχήμα 1.

Η σύγκριση του ερωτήματος με τις αναπαραστάσεις -μέσω δεικτών- των εγγράφων ονομάζεται διαδικασία αντιστοίχισης. Η διαδικασία αντιστοίχισης συνήθως έχει ως αποτέλεσμα την ταξινόμηση των εγγράφων. Οι χρήστες θα χρησιμοποιήσουν αυτή την ταξινομημένη λίστα εγγράφων για να αναζητήσουν τις πληροφορίες που χρειάζονται. Τα

πλεον σχετικά έγγραφα θα βρίσκονται στην κορυφή της κατάταξης, ελαχιστοποιώντας έτσι τον χρόνο που ο χρήστης πρέπει να καταναλώσει για την ανάγνωση των εγγράφων. Οι απλοί αλλά αποτελεσματικοί αλγόριθμοι ταξινόμησης χρησιμοποιούν την κατανομή συχνοτήτων των όρων στα έγγραφα, αλλά και στατιστικά μεγέθη όπως ο αριθμός των υπερσυνδέσμων που οδηγούν στο έγγραφο. Οι αλγόριθμοι ταξινόμησης που βασίζονται σε στατιστικές προσεγγίσεις μειώνουν κατά το ήμισυ τον χρόνο που ο χρήστης πρέπει να δαπανήσει για την ανάγνωση των εγγράφων. Η θεωρία πίσω από τους αλγόριθμους ταξινόμησης είναι ένα κρίσιμο μέρος της θεωρίας ανάκτησης πληροφοριών και το κύριο θέμα αυτού του κεφαλαίου.

Η ανάπτυξη των ηλεκτρονικών βιβλιοθηκών που είχε ως αποτέλεσμα την αύξηση της χρήσης του Διαδικτύου κατέστησε τα ηλεκτρονικά δεδομένα ως τη σημαντικότερη πηγή άντλησης χρήσιμων πληροφοριών στον τομέα της έρευνας. Έτσι, τα συστήματα ανάκτησης πληροφοριών στοχεύουν στην εξεύρεση σωστών πληροφοριών από έγγραφα, άγνωστων πληροφοριών από το διαδίκτυο και κρυφών σχέσεων μεταξύ των πληροφοριών που ανακτήθηκαν.

Οι πληροφορίες μπορούν να αποθηκευτούν τόσο με δομημένο όσο και με μη δομημένο τρόπο. Για παράδειγμα, μπορεί να αποθηκευτούν με τη μορφή συστημάτων βάσεων δεδομένων που αντιπροσωπεύουν το δομημένο τρόπο, ή σαν αρχεία HTML και κειμένου που αντιπροσωπεύουν το μη δομημένο τρόπο.

Συμπερασματικά προκύπτει ότι η ανάκτηση πληροφοριών ορίζεται ως η τέχνη και η επιστήμη της αναζήτησης πληροφοριών σε έγγραφα, η αναζήτηση των ίδιων των εγγράφων, η αναζήτηση μετα-δεδομένων που περιγράφουν έγγραφα ή η αναζήτηση σε βάσεις δεδομένων για κείμενο, ήχο, εικόνες ή δεδομένα.

Ένα αποτελεματικό σύστημα ανάκτησης πληροφοριών IR πρέπει να είναι σε θέση να εκτελέσει γρήγορα και με επιτυχία την ακόλουθη διαδικασία:

- Αποδοχή του ερωτήματος του χρήστη.
- Κατανόηση των αναγκών του χρήστη με βάση το ερώτημα.
- Αναζήτηση σχετικών εγγράφων από μια βάση δεδομένων.
- Επιστροφή των εγγράφων στο χρήστη.
- Ταξινόμηση των εγγράφων με βάση την ομοιότητά τους με το ερώτημα του χρήστη.

΄Ένα μοντέλο ανάκτησης πληροφοριών είναι μια τετράδα (D,Q,F,R), όπου:

- D είναι ένα σύνολο αναπαραστάσεων των εγγράφων
- Q είναι ένα σύνολο αναπαραστάσεων των πληροφοριών που χρειάζεται ο χρήστης με βάση τα ερωτήματα.
- F είναι ένα πλαίσιο μοντελοποίησης των αναπαραστάσεων των εγγράφων, των ερωτημάτων καθώς και των μεταξύ τους σχέσεων
- R: Q → D είναι μια συνάρτηση ταξινόμησης που αντιστοιχίζει έναν πραγματικό αριθμό με ένα ερώτημα qi → Q και μια αναπαράσταση εγγράφου dj → D.

Τα κυριότερα Μοντέλα Ανάκτησης Πληροφοριών είναι:

- Το Δυαδικό Μοντέλο (Boolean Model)
- Το Διανυσματικό Μοντέλο (Vector Space Model)
- Το Πιθανοτικό Μοντέλο (Probabilistic Model)

1.2.1. Το Δυαδικό μοντέλο (Boolean Model)

Το Δυαδικό μοντέλο είναι το πρώτο μοντέλο ανάκτησης πληροφοριών και ίσως και το μοντέλο που έχει δεχθεί την περισσότερη κριτική. Το μοντέλο μπορεί να εξηγηθεί αν σκεφτεί κανείς τον όρο ενός ερωτήματος ως ένα σαφή ορισμό ενός συνόλου εγγράφων. Για παράδειγμα, ο όρος “οικονομικός”, ορίζει το σύνολο όλων των εγγράφων που σχετίζονται με τον όρο “οικονομικός”. Χρησιμοποιώντας τους τελεστές της μαθηματικής λογικής του George Boole, οι όροι ενός ερωτήματος και τα αντίστοιχα σύνολα εγγράφων μπορούν να συνδυαστούν, σχηματίζοντας νέα σύνολα εγγράφων. Ο Boole όρισε τρεις βασικούς τελεστές, το λογικό γινόμενο που ονομάζεται AND, το λογικό άθροισμα που ονομάζεται OR και τη λογική διαφορά που ονομάζεται NOT. Συνδυάζοντας όρους με τον τελεστή AND μπορεί να καθοριστεί ένα σύνολο εγγράφων που το πλήθος των στοιχείων τους είναι μικρότερο ή ίσο με το πλήθος των στοιχείων των συνόλων εγγράφων οποιωνδήποτε από τους μεμονωμένους όρους. Για παράδειγμα, το ερώτημα “κοινωνικό και οικονομικό” θα παράγει το σύνολο των εγγράφων που σχετίζονται τόσο με τον όρο “κοινωνικό” όσο και με τον όρο “οικονομικό”, δηλαδή την τομή των δύο συνόλων. Ο συνδυασμός όρων με τον τελεστή OR θα δημιουργήσει ένα σύνολο εγγράφων με το πλήθος των στοιχείων τους είναι μεγαλύτερο ή ίσο με το πλήθος των στοιχείων των συνόλων εγγράφων οποιουδήποτε από τους μεμονωμένους όρους. Έτσι, το ερώτημα “κοινωνικό ή πολιτικό” θα παράγει το σύνολο των εγγράφων που σχετίζονται είτε με τον

όρο “κοινωνικό” είτε με τον όρο “πολιτικό”, ή και με τα δύο, δηλαδή την ένωση των δύο συνόλων. Όλα αυτά απεικονίζονται στα διαγράμματα Venn του Σχήματος 2, στα οποία κάθε σύνολο εγγράφων απεικονίζεται από ένα δίσκο. Οι διασταυρώσεις αυτών των δίσκων και των συμπληρωμάτων τους χωρίζουν τη συλλογή εγγράφων σε 8 μη επικαλυπτόμενες περιοχές, οι ενώσεις των οποίων δίνουν 256 διαφορετικούς συνδυασμούς Boolean «κοινωνικών, πολιτικών και οικονομικών εγγράφων» (Wiley, 2009). Στο Σχήμα 2, τα ανακτημένα σύνολα εμφανίζονται από τις σκιασμένες περιοχές.

Σχήμα 2. Δυαδικοί συνδυασμοί συνόλων που απεικονίζονται ως διαγράμματα Venn

Ένα πλεονέκτημα του Boolean μοντέλου είναι ότι δίνει στους χρήστες μια αίσθηση ελέγχου πάνω στο σύστημα. Γίνεται αμέσως σαφές για ποιο λόγο ένα έγγραφο έχει ανακτηθεί με δεδομένο ένα ερώτημα. Εάν το προκύπτον σύνολο εγγράφων είναι είτε πολύ μικρό είτε πολύ μεγάλο, είναι άμεσα σαφές ποιοι τελεστές θα παράγουν αντίστοιχα ένα μεγαλύτερο ή μικρότερο σύνολο. Για τους μη εκπαιδευμένους χρήστες, το μοντέλο παρουσιάζει ορισμένα μειονεκτήματα. Το κύριο μειονέκτημά του είναι ότι τα ανακτηθέντα έγγραφα δεν είναι ταξινομημένα. Το μοντέλο είτε ανακτά ένα έγγραφο είτε όχι, κάτι το οποίο μπορεί να οδηγήσει ενδεχόμενα σε απογοητευτικές αποφάσεις. Για παράδειγμα, το ερώτημα “κοινωνική AND ένωση AND εργαζόμενων” φυσικά δεν θα ανακτήσει ένα έγγραφο που σχετίζεται με τους όρους πάρτυ, γενέθλια και τούρτα, αλλά δεν θα ανακτήσει επίσης ένα έγγραφο που σχετίζεται με τους όρους “ένωση” και “εργαζομένων” που

στερείται τον όρο “κοινωνική”. Σαφώς, είναι πιθανό ότι το τελευταίο έγγραφο είναι πιο χρήσιμο από το πρώτο, αλλά το μοντέλο δεν έχει κανένα μέσο για να κάνει το διαχωρισμό.

1.2.2. Το πιθανοτικό μοντέλο ανάκτησης πληροφοριών

Ενώ οι Maron και Kuhns (1960) εισήγαγαν την ταξινόμηση με βάση την πιθανότητα συσχέτισης, ο Stephen Robertson (Robertson, 1984) ήταν που θεμελίωσε την ιδέα. Διατύπωσε την αρχή της ταξινόμησης με βάση την πιθανότητα, την οποία απέδωσε στον William Cooper, ως εξής:

«Εάν η απάντηση ενός συστήματος ανάκτησης πληροφοριών σε κάθε αίτημα είναι η ταξινόμηση των εγγράφων στις συλλογές κατά φθίνουσα πιθανότητα ικανοποίησης της ανάγκης του χρήστη που υπέβαλε το ερώτημα, όπου οι πιθανότητες υπολογίζονται όσο το δυνατόν ακριβέστερα με βάση τα δεδομένα που έχουν δοθεί στο σύστημα για το σκοπό αυτό, τότε η αποτελεσματικότητα του συστήματος θα είναι η μέγιστη δυνατή βάσει αυτών των δεδομένων.»

Αυτό φαίνεται ως μια μάλλον ασήμαντη απαίτηση, δεδομένου ότι ο στόχος των συστημάτων ανάκτησης πληροφοριών είναι "να βοηθήσει τους χρήστες να βρουν τις πληροφορίες που χρειάζονται (Bello, Kaul, Britton, 2014). Ας υποτεθεί ότι ένας χρήστης εισάγει ερώτημα που περιέχει έναν μόνο όρο, για παράδειγμα τον όρο “κοινωνικό”. Εάν όλα τα έγγραφα που πληρούν τις ανάγκες του χρήστη ήταν γνωστά, θα ήταν δυνατή η διαίρεση της συλλογής σε 4 μη αλληλεπικαλυπτόμενα σύνολα εγγράφων όπως απεικονίζεται στο διάγραμμα Venn του Σχήματος 3.

Το σχήμα περιέχει πρόσθετες πληροφορίες σχετικά με το μέγεθος καθενός από τα μη αλληλεπικαλυπτόμενα σύνολα. ‘Εστω ότι η εν λόγω συλλογή έχει 10.000 έγγραφα εκ των οποίων τα 1.000 περιέχουν τη λέξη «κοινωνικό» και ότι μόνο 11 έγγραφα σχετίζονται με το ερώτημα εκ των οποίων 1 περιέχει τη λέξη "κοινωνικό".

Σχήμα 3. Διάγραμμα Venn για το πιθανοτικό μοντέλο ανάκτησης πληροφοριών

Αν ένα έγγραφο λαμβάνεται τυχαία από το σύνολο των εγγράφων που σχετίζονται με τον όρο “κοινωνικό”, τότε η πιθανότητα επιλογής ενός σχετικού εγγράφου είναι $\frac{1}{1000} = 0{,}0010$. Αν ένα έγγραφο λαμβάνεται τυχαία από το σύνολο των εγγράφων που δεν σχετίζετα με τον όρο “κοινωνικό”, τότε η πιθανότητα συνάφειας είναι μεγαλύτερη: $\frac{10}{9{,}000} = 0{,}0011$. Βάσει αυτών των στοιχείων, η καλύτερη απόδοση παρατηρήθηκε σε συστήματα που επιστρέφουν τα έγγραφα που δεν σχετίζονται με τον όρο του ερωτήματος, δηλαδή σε συστήματα που παρουσιάζουν πρώτα τα έγγραφα που είναι ανόμοια με το ερώτημα. Είναι σαφές ότι μια τέτοια στρατηγική παραβιάζει το κριτήριο ομοιότητας του Luhn.

Το κριτήριο ομοιότητας του Luhn

Ο Hans Peter Luhn ήταν ο πρώτος που πρότεινε μία στατιστική προσέγγιση στην αναζήτηση πληροφοριών (Luhn, 1957). Πρότεινε ότι για να γίνει αναζήτηση σε μία συλλογή εγγράφων, ο χρήστης θα έπρεπε αρχικά να δημιουργήσει ένα έγγραφο το οποίο να είναι παρόμοιο με τα έγγραφα που αναζητά. Ο βαθμός ομοιότητας μεταξύ της αναπαράστασης του εγγράφου που δημιούργησε και των αναπαραστάσεων των εγγράφων συλλογής, χρησιμοποιείται ως μέτρο για την ταξινόμηση των αποτελεσμάτων της αναζήτησης. Το κριτήριο ομοιότητας του Luhn συνοψίζεται ως εξής:

Όσο περισσότερο δύο αναπαραστάσεις ‘συμφωνούν’ ως προς τα δοθέντα στοιχεία, τόσο μεγαλύτερη θα είναι η πιθανότητα να αντιπροσωπεύουν παρόμοια πληροφορία.

Οι Stephen Robertson και Karen Sparck-Jones βασίζουν το πιθανοτικό μοντέλο τους στη λογική αυτή (Robertson, Jones 1976). Πρότειναν να ταξινομηθούν τα έγγραφα με βάση την πιθανότητα συσχέτισης R έχοντας την περιγραφή του περιεχομένου του εγγράφου D. Το D είναι εδώ ένα διάνυσμα δυαδικών συνιστωσών και κάθε συνιστώσα αντιπροσωπεύει έναν όρο.

Στο πιθανοτικό μοντέλο ανάκτησης πληροφοριών η πιθανότητα $\mathrm{P}\ (\mathrm{R} \mid D)$ πρέπει να ερμηνευτεί ως εξής: μπορεί να υπάρχουν πολλά, ας πούμε 10, έγγραφα που αντιπροσωπεύονται από το ίδιο D. Εάν 9 από αυτά είναι σχετικά, τότε $\mathrm{P}\ (\mathrm{R} \mid D) = 0.9$. Για να γίνει αυτό πρακτικά, χρησιμοποιούμε τον κανόνα του Bayes στις πιθανότητες: $\mathrm{P}\ (\mathrm{R} \mid D)$ / $\mathrm{P}\ (\bar{R} \mid D)$, όπου το R υποδηλώνει έλλειψη σχετικότητας. Οι πιθανότητες μας επιτρέπουν να αγνοήσουμε το $\mathrm{P}\ (D)$ στον υπολογισμό ενώ παράλληλα γίνεται μια ταξινόμηση με βάση την πιθανότητα συσχέτισης.

$$\frac{\mathrm{P}(R|D)}{\mathrm{P}(\bar{R}|D)} = \frac{\mathrm{P}(\mathrm{D}|\mathrm{R})\mathrm{P}(\mathrm{R})}{\mathrm{P}(\mathrm{D}|\bar{R})\mathrm{P}(\bar{R})} = \frac{\Pi_k \mathrm{P}(D_k|R)P(R)}{\Pi_k \mathrm{P}(D_k|\bar{R})\mathrm{P}(\bar{R})}$$

Το Dk υποδηλώνει το k-οστό όρο στο διάνυσμα του εγγράφου. Οι πιθανότητες των όρων ορίζονται όπως παραπάνω. Μια βελτιωμένη εφαρμογή του πιθανοτικού μοντέλου ανάκτησης πληροφοριών χρησιμοποιεί τους ακόλουθους τρεις μετασχηματισμούς διατήρησης διάταξης. Πρώτον, τα έγγραφα ταξινομούνται με βάση τα αθροίσματα των λογαρίθμων των λόγων [πιθανότητα επιτυχίας/πιθανότητα αποτυχίας], αντί των ίδιων των λόγων. Δεύτερον, αγνοούνται οι a priori πιθανότητες συσχέτισης $\mathrm{P}\ (\mathrm{R})$ / $\mathrm{P}\ (\bar{R})$. Τρίτον, αφαιρούμε το $\sum \log k\ (\mathrm{P}(D_k = 0|\ \mathrm{R})/(\mathrm{P}(D_k = 0|\ \bar{R}))$, δηλαδή το αποτέλεσμα του κενού εγγράφου. Με αυτό τον τρόπο, το άθροισμα για όλους τους όρους, που μπορεί να είναι εκατομμύρια, περιλαμβάνει μόνο μη μηδενικές τιμές για τους όρους που υπάρχουν στο έγγραφο.

$$D = \sum_k \left(log \frac{P(D_k = 1|R)P(D_k = 0|\bar{R})}{P(D_k = 1|\bar{R})P(D_k = 0|R)} \right)$$

Στην πράξη, οι όροι που δεν περιλαμβάνονται στο ερώτημα παραβλέπονται επίσης στην παραπάνω εξίσωση. Η πλήρης χρήση του πιθανοτικού μοντέλου ανάκτησης πληροφοριών απαιτεί δύο πράγματα: παραδείγματα σχετικών εγγράφων και μακρά ερωτήματα. Τα σχετικά έγγραφα είναι απαραίτητα για τον υπολογισμό του $P(\frac{D_k}{R})$, δηλαδή της πιθανότητας ότι το έγγραφο περιέχει τον όρο k δεδομένης της σχετικότητας. Τα μακρά ερωτήματα απαιτούνται επειδή το μοντέλο διακρίνει μόνο την παρουσία ή την απουσία όρων σε έγγραφα και ως εκ τούτου η τιμή της πιθανότητας του αποτελέσματος είναι αρκετά μικρή για σύντομα ερωτήματα. Για ένα ερώτημα μιας λέξης, ο αριθμός της διακριτής πιθανότητας είναι δύο (είτε ένα έγγραφο περιέχει τη λέξη είτε όχι), για ένα ερώτημα δύο λέξεων είναι τέσσερα (το έγγραφο περιέχει και τους δύο όρους ή μόνο τον πρώτο όρο ή μόνο τον δεύτερο, ή κανέναν), για ένα ερώτημα τριών λέξεων είναι οκτώ κλπ. Προφανώς αυτό καθιστά το μοντέλο ακατάλληλο για αναζήτηση στο διαδίκτυο, για το οποίο δεν είναι γνωστά εκ των προτέρων σχετικά έγγραφα και για τα οποία τα ερωτήματα είναι συνήθως μικρά. Ωστόσο, το μοντέλο είναι χρήσιμο για παράδειγμα σε φίλτρα ανεπιθύμητης αλληλογραφίας. Τα φίλτρα ανεπιθύμητης αλληλογραφίας συσσωρεύουν πολλά παραδείγματα σχετικών (χωρίς spam) και άσχετων (spam) εγγράφων με την πάροδο του χρόνου. Για να αποφασιστεί εάν ένα εισερχόμενο μήνυμα ηλεκτρονικού ταχυδρομείου είναι spam, μπορεί να χρησιμοποιηθεί το πλήρες κείμενο του μηνύματος ηλεκτρονικού ταχυδρομείου αντί για λίγους μόνο όρους αναζήτησης.

Κεφάλαιο 2ο. Vector Space model (Διανυσματικό Μοντέλο)

Εισαγωγή

Στο κεφάλαιο αυτό αναλύεται η λειτουργία και οι τεχνικές που περιβάλλουν το Διανυσματικό Μοντέλο, καθώς και οι περιορισμοί του.

2.1. Ορισμός Διανυσματικού Μοντέλου

Ένα από τα πιο σημαντικά μοντέλα ανάκτησης πληροφοριών είναι το Διανυσματικό Μοντέλο. Το Διανυσματικό Μοντέλο είναι ένα αλγεβρικό μοντέλο που χρησιμοποιείται για την ανάκτηση πληροφοριών. Ο λόγος που ορίζεται ως αλγεβρικό μοντέλο είναι διότι χρησιμοποιεί βασικές έννοιες και μεθοδολογίες της γραμμικής άλγεβρας. Ο σκοπός του είναι να αναπαριστά ένα έγγραφο φυσικής γλώσσας, με τη χρήση διανυσμάτων, σε έναν πολυδιάστατο χώρο.

Το Διανυσματικό Μοντέλο είναι ένας τρόπος αναπαράστασης των εγγράφων μέσω των λέξεων που περιέχουν. Η αρχή στην οποία βασίζεται το μοντέλο είναι ότι με την τοποθέτηση όρων, εγγράφων και ερωτημάτων σε ένα χώρο με τη μορφή διανυσμάτων, είναι δυνατόν να υπολογιστούν οι ομοιότητες μεταξύ των ερωτημάτων και των όρων ή των εγγράφων και να ταξινομηθούν τα αποτελέσματα σύμφωνα με την ομοιότητα μεταξύ τους.

2.2. Λειτουργία του Διανυσματικού Μοντέλου

Η λειτουργία του Διανυσματικού Μοντέλου γίνεται με τον τρόπο που περιγράφεται παρακάτω:

- Κάθε έγγραφο μετατρέπεται σε έναν πίνακα συχνότητας των λέξεων που περιέχει.
- Κάθε πίνακας μπορεί να θεωρηθεί ως ένα διάνυσμα.
- Το λεξιλόγιο δημιουργείται από όλες τις λέξεις σε όλα τα έγγραφα του συστήματος. Το λεξιλόγιο μπορεί να είναι μία λίστα ή ένας πίνακας.
- Κάθε έγγραφο και κάθε ερώτημα αναπαρίσταται σαν ένα διάνυσμα με βάση το λεξιλόγιο.
- Υπολογισμός της συνημιτονικής ομοιότητας (Cosine Similarity) μεταξύ 2 εγγράφων. Ως συνημιτονική ομοιότητα ορίζεται η μέτρηση της ομοιότητας δύο μη-

μηδενικών διανυσμάτων μέσω του υπολογισμού του συνημίτονου της γωνίας μεταξύ τους.

- Ταξινόμηση των εγγράφων με βάση την ομοιότητά τους

Τα τρία στάδια του Διανυσματικού Μοντέλου μπορούν να περιγραφούν ως εξής:

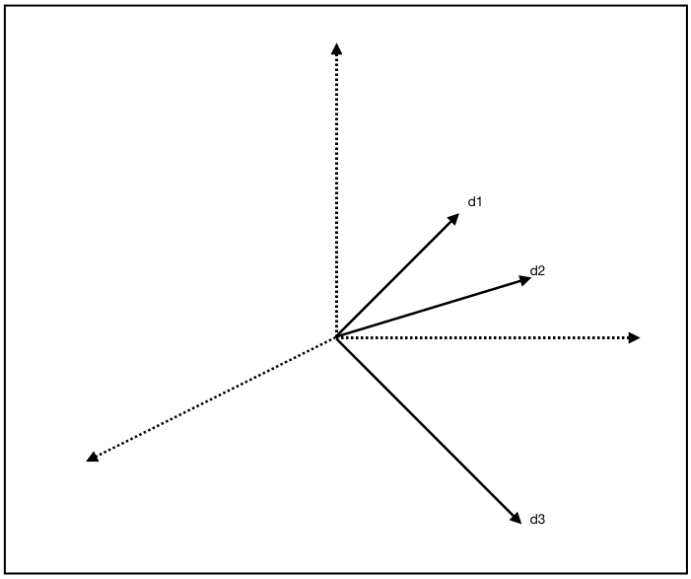
- Το πρώτο στάδιο είναι η δημιουργία δεικτών στο έγγραφο.
- Το δεύτερο στάδιο είναι η στάθμιση των κατηγοριοποιημένων όρων ώστε να ενισχυθεί η ανάκτηση των εγγράφων που σχετίζονται με το ερώτημα του χρήστη.
- Το τελευταίο στάδιο ταξινομεί το έγγραφο σε σχέση με το ερώτημα του χρήστη σύμφωνα με ένα μέτρο ομοιότητας.

Είναι προφανές ότι πολλές από τις λέξεις σε ένα έγγραφο δεν περιγράφουν το περιεχόμενο, λέξεις όπως το “είναι”. Χρησιμοποιώντας το αυτόματο ευρετήριο εγγράφων, αυτές οι μη σημαντικές λέξεις αφαιρούνται από το διάνυσμα του εγγράφου, οπότε το έγγραφο θα αντιπροσωπεύεται μόνο από λέξεις που έχουν περιεχόμενο. Η δημιουργία του ευρετηρίου μπορεί να βασιστεί στη συχνότητα των όρων. Οι όροι που έχουν είτε υψηλή είτε χαμηλή συχνότητα μέσα σε ένα έγγραφο θεωρούνται λέξεις λειτουργικές και χωρίς περιεχόμενο. Στην πράξη, ο όρος συχνότητα ήταν δύσκολο να εφαρμοστεί στη δημιουργία του ευρετηρίου. Αντ' αυτού, γίνεται χρήση μιας λίστας που κρατά συνηθισμένες λέξεις και αφαιρεί λέξεις υψηλής συχνότητας, κάτι που καθιστά τη μέθοδο αυτή γλωσσικά εξαρτημένη. Γενικά, το 40-50% του συνολικού αριθμού των λέξεων σε ένα έγγραφο καταργείται με τη βοήθεια μιας τέτοιας λίστας.

Έχουν επίσης εφαρμοστεί μη γλωσσικές μέθοδοι για τη δημιουργία του ευρετηρίου. Η δημιουργία του ευρετηρίου βασίζεται στην υπόθεση ότι υπάρχει κάποια στατιστική διαφορά στην κατανομή των λέξεων που έχουν περιεχόμενο και των λειτουργικών λέξεων. Οι λέξεις που επιλέχθηκαν ως λειτουργικές μοντελοποιούνται από την κατανομή Poisson (Kim, 2019) σε όλα τα έγγραφα, ενώ οι όροι που φέρουν περιεχόμενο δεν μπορούν να μοντελοποιηθούν. Η χρήση του μοντέλου Poisson επεκτάθηκε στο μοντέλο Bernoulli.

Το διανυσματικό Μοντέλο αναπαριστά έγγραφα σε ένα χώρο πολλών διαστάσεων, όπου κάθε όρος είναι και μια διάσταση. Χρησιμοποιείται κυρίως στη διαδικασία ανάκτησης πληροφορίας αντιπροσωπεύοντας πλήρη έγγραφα. Οι όροι θα μπορούσαν να είναι οι λέξεις που περιέχονται σε ένα έγγραφο, για παράδειγμα “γάτα”, “σκύλος”, “ποντίκι”, ενώ

τα έγγραφα είναι γραμμικοί συνδυασμοί διανυσμάτων. Έστω λοιπόν ένα έγγραφο που περιέχει μια (1) φορά τη λέξη “γάτα”, δύο (2) φορές τη λέξη “σκύλος” και μια (1) φορά τη λέξη “ποντίκι”, τότε το έγγραφο αυτό αναπαρίσταται με το διάνυσμα με συντεταγμένες (1,2,1) στον τρισδιάστατο χώρο που δημιουργείται (Σχήμα 4), όπου d1, d2, d3 τα διανύσματα των τριών εγγράφων.



Σχήμα 4. Επεξήγηση διανυσματικού μοντέλου

2.3. Εργασία με το Vector Space Model

Ο Δρ. Khalaf Khatatneh και οι συνεργάτες του (Khatatneh, 2005) έδειξαν ένα νέο μηχανισμό για τη μείωση του χώρου που χρησιμοποιείται από το σύστημα ανάκτησης πληροφοριών χρησιμοποιώντας τη δομή του ημιτελούς πίνακα. Σε αυτή τη δομή ο πίνακας περιλαμβάνει δύο υποπίνακες. Ο ένας περιέχει μια λέξη και ο άλλος ορισμένους συντελεστές. Αυτή η δομή συνεπάγεται ένα έγγραφο ανά πίνακα, επομένως η μία σειρά αποθηκεύει τον δείκτη και η δεύτερη σειρά το βάρος κάθε δείκτη όπως απεικονίζεται στο παρακάτω σχήμα:

D1	**T_1**	**T_2**	**T_3**	**……**	**……**	**T_n**
	W_{11}	**W_{12}**	**W_{13}**	**……**	**……**	**W_{ln}**

Σχήμα 5. Δομή ημιτελούς πίνακα

Χρησιμοποιώντας αυτό το νέο μηχανισμό, θα μειωθούν όσα ήταν περιττά στην κλασσική αναπαράσταση όρων και εγγράφων. Ο αριθμός των γραμμών στον πίνακα θα αυξηθεί κατά ένα με την προσθήκη νέου εγγράφου και ο αριθμός των στηλών θα αυξηθεί κατά ένα με την προσθήκη νέου όρου. Όμως, στον πίνακα, για οποιοδήποτε νέο έγγραφο που έχει προστεθεί στη συλλογή των κειμένων, ένας χωριστός πίνακας θα προστεθεί στη βάση δεδομένων και όλοι οι νέοι όροι θα προστεθούν στον νέο πίνακα.

Οι Michael W. Berry και άλλοι (Lee, Lam, 2002) παρουσίασαν στο έργο τους πώς οι θεμελιώδεις μαθηματικές έννοιες της γραμμικής άλγεβρας, όπως η έννοια του υποχώρου ή της γραμμικής ανεξαρτησίας, μπορούν να χρησιμοποιηθούν για τη διαχείριση και την δημιουργία δεικτών σε μεγάλα έγγραφα. Θεώρησαν ότι οι παραδοσιακές τεχνικές είναι άχρηστες, δεδομένου ότι κατά τη διαδικασία δημιουργίας δεικτών λαμβάνονται υπόψη όλες οι πληροφορίες στο εσωτερικό του εγγράφου, ανεξάρτητα από το πόσο σημαντικές είναι. Για παράδειγμα, οι περιλήψεις, οι κατάλογοι συγγραφέων, οι κατάλογοι λέξεων-κλειδιών κ.λπ. είναι μερικές βοηθητικές πληροφορίες οι οποίες δεν έχουν ιδιαίτερη σημασία για το περιεχόμενο του εγγράφου, μόνο οι ενδιαφερόμενοι για τη βιβλιογραφία θα μπορούσαν να βρουν τέτοια στοιχεία χρήσιμα. Επομένως, η εξαίρεση αυτών των στοιχείων από τη διαδικασία δημιουργίας δεικτών θα μειώσει την κατανάλωση χωρητικότητας και θα βελτιώσει την αποτελεσματικότητα της ανάκτησης. Οι Michael W. Berry και άλλοι (1999) στην έρευνά τους χρησιμοποίησαν τις μαθηματικές εξισώσεις του Διανυσματικού Μοντέλου για να εφαρμόσουν μια τέτοια προσέγγιση.

Υπάρχουν πολλές άλλες ενδιαφέρουσες έρευνες που αφορούν την εξόρυξη δεδομένων και τη χρήση διανυσματικών μοντέλων. Οι Czibula και Cojocar (Czibula, Cojocar, 2010) στην έρευνά τους εξήγησαν πως τα διανυσματικά μοντέλα θα μπορούσαν να χρησιμοποιηθούν στην εξόρυξη δεδομένων. Παρουσιάζουν μια νέα προσέγγιση που χρησιμοποιεί την ομαδοποίηση (clustering) στην εξόρυξη δεδομένων και προτείνουν δύο τεχνικές: μια τεχνική ομαδοποίησης βασισμένη στον αλγόριθμο k-means και μια ιεραρχική

τεχνική ομαδοποίησης. Το Διανυσματικό Μοντέλο χρησιμοποιείται για την εξεύρεση της ομοιότητας μεταξύ δύο μεθόδων και για να αποφασιστείποια από τις δύο είναι η καλύτερη.

2.4. Τεχνικές και προσεγγίσεις του Vector Space Model

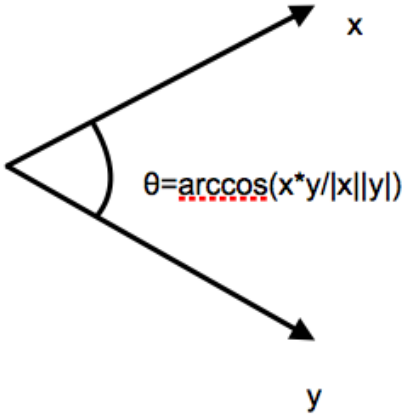
Σε αυτό το υποκεφάλαιο θα εξηγήσουμε τις τέσσερις βασικές τεχνικές του Διανυσματικού Μοντέλου και θα συγκρίνουμε αυτές τις τεχνικές. Όπως αναφέρθηκε στην εισαγωγή υπάρχουν τέσσερις βασικές τεχνικές που χρησιμοποιούνται με το Διανυσματικό Μοντέλο.

- Το εσωτερικό γινόμενο.
- Η ομοιότητα του συνημιτόνου.
- Η ομοιότητα του Jaccard.
- Η ομοιότητα του Dice.

2.4.1. Το εσωτερικό γινόμενο

Το εσωτερικό γινόμενο είναι η πρώτη τεχνική στο Διανυσματικό μοντέλο. Αυτή η τεχνική θεωρείται ως βάση για άλλες τεχνικές. Ο υπολογισμός των αποτελεσμάτων των άλλων τεχνικών εξαρτάται από τα αποτελέσματα αυτής της τεχνικής.

Ο όρος “χώρος εσωτερικού γινομένου” στα μαθηματικά είναι ένας διανυσματικός χώρος που έχει μια συμπληρωματική δομή που ονομάζεται εσωτερικό γινόμενο. Αυτή η δομή συνδέει κάθε ζεύγος διανυσμάτων στον χώρο με μια ποσότητα γνωστή ως εσωτερικό γινόμενο των διανυσμάτων. Τα εσωτερικά γινόμενα επιτρέπουν την εισαγωγή διαισθητικών γεωμετρικών σχεδιασμών, όπως για παράδειγμα το μήκος ενός διανύσματος ή η γωνία μεταξύ δύο διανυσμάτων (Wolfram, 1999). Σε ένα διανύσματικό χώρο, το εσωτερικό γινόμενο είναι ένας τρόπος πολλαπλασιασμού των διανυσμάτων (Wolfram, 1999). Το Σχήμα 6 δείχνει την ερμηνεία της γωνίας μεταξύ δύο διανυσμάτων που ορίζεται χρησιμοποιώντας το εσωτερικό γινόμενο.



Σχήμα 6. Ερμηνεία της γωνίας μεταξύ δύο διανυσμάτων καθορισμένων με χρήση ενός εσωτερικού προϊόντος

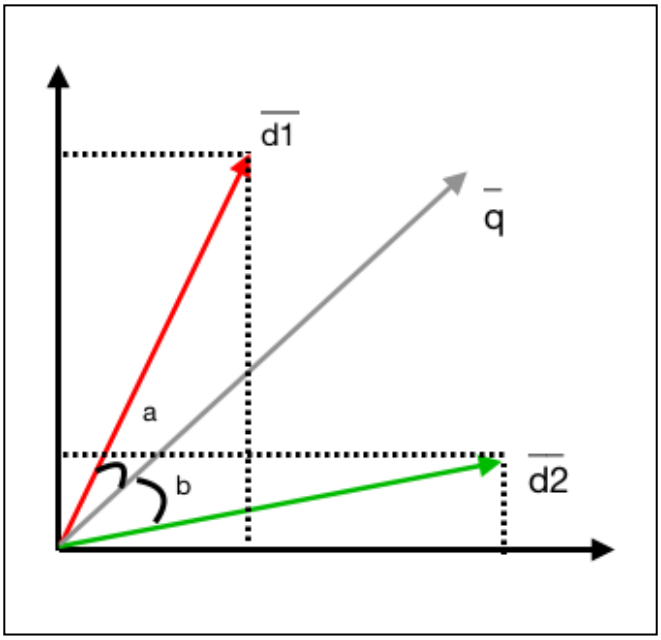
Η ιδέα εδώ είναι να απεικονιστεί κάθε έγγραφο και το ερώτημα στο σύστημα ανάκτησης ως διανύσματα και να βρούμε την ομοιότητα μεταξύ τους. Τα διανύσματα των εγγράφων τα οποία έχουν ομοιότητα με το διάνυσμα του ερωτήματος θεωρούνται συγγενικά με το ερώτημα και πιθανόν ικανοποιούν τις ανάγκες του χρήστη.

2.4.2. Η ομοιότητα του συνημιτόνου

Το συνημίτονο χρησιμοποιείται ως μέτρο για να βρεθεί η ομοιότητα μεταξύ δύο διανυσμάτων. Αυτό γίνεται με την εύρεση του συνημίτου της μεταξύ τους γωνίας (Sarwar, 2001). Έτσι, αν το εσωτερικό γινόμενο χρησιμοποιείται για την εύρεση της απόστασης μεταξύ δύο διανυσμάτων, το συνημίτονο χρησιμοποιείται για την εύρεση της γωνίας μεταξύ αυτών των διανυσμάτων.

Η διαδικασία της ομοιότητας του συνημιτόνου χρησιμοποιείται για να να αποφασιστεί ποιο έγγραφο (d1,d2) είναι περισσότερο όμοιο με τη δοθείσα ερώτηση q. Αυτή είναι και μια από τις βασικές ιδέες, στην οποία βασίζεται η λειτουργία των μηχανών αναζήτησης. Έστω λοιπόν μια ερώτηση που αναπαρίσταται με το διάνυσμα q και δύο έγγραφα που

αναπαρίστανται με τα διανύσματα d1 και d2, όπως φαίνεται στο σχήμα 7. Για να διαπιστωθεί ποιο από τα δύο έγγραφα έχει μεγαλύτερη ομοιότητα με τη ερώτηση q, αρκεί να συγκριθούν οι γωνίες a,b (ή καλύτερα τα συνημίτονα των γωνιών) που σχηματίζει το διάνυσμα της ερώτησης με καθένα από τα διανύσματα των δύο εγγράφων. Όταν το συνημίτονο είναι μικρότερο, η ομοιότητα είναι μεγαλύτερη..



Σχήμα 7. Αναπαράσταση διανυσμάτων

Η μαθηματική έκφραση που υπολογίζει το συνημίτονο της γωνίας δύο διανυσμάτων A(x1,y1) και B(x2,y2) είναι η εξής:

$$cosine\ \theta = \frac{A * B}{|A||B|} = \frac{x1'x2 + y1'y2}{(x1^2 + y1^2)^{\frac{1}{2}}\,(x2^2 + y2^2)^{\frac{1}{2}}}$$

Όπου A*B είναι το εσωτερικό γινόμενο των δύο διανυσμάτων, το οποίο είναι ένας τρόπος να πολλαπλασιάζει κανείς διανύσματα μεταξύ τους.

Για παράδειγμα μπορεί εύκολα να υπολογιστεί η ομοιότητα μεταξύ των παρακάτω συντελεστών:

A = «γάτα, σκύλος, σκύλος» = <1,2,0>

B = «γάτα, σκύλος, ποντίκι, ποντίκι» = <1,1,2>

Η απάντηση δίνεται απο τον τύπο:

$$\sigma(A,B) = \frac{1*1+2*1+0*2}{\sqrt{1^2+2^2+0^2}-b\pm\sqrt{1^2+1^2+2^2}} = \frac{3}{\sqrt{5\sqrt{6}}} \approx 0.55$$

Γενικά το συνημίτονο μιας γωνίας έχει ως σύνολο τιμών το διάστημα [-1,1], όμως αφού τα διανύσματα είναι στο πρώτο τεταρτημόριο το συνημίτονο θα έχει ως σύνολο τιμών το [0,1].

Έτσι, αν θεωρήσουμε ότι το σημείο A (x_1, y_1) αντιπροσωπεύει ένα ερώτημα και τα σημεία B (x_2, y_2), D (x_3, y_3), E (x_4, y_4), F (x_5, y_5) αντιπροσωπεύουν κάποια έγγραφα, θα πρέπει να υπολογίσουμε το συνημίτονο της γωνίας μεταξύ του A (ερώτημα) και κάθε εγγράφου και να τα ταξινομήσουμε σε φθίνουσα σειρά συνημιτόνου (cosines similarites). Αυτή η ενέργεια γίνεται για όλα τα έγγραφα (Minerazzi, 2014).

2.4.3. Η ομοιότητα του Jaccard

“Ο συντελεστής ομοιότητας Jaccard, που είναι επίσης γνωστός ως δείκτης Jaccard, είναι ένα στατιστικό μέτρο ομοιότητας μεταξύ συνόλων” (Bank, Cole, 2008). Η συνάρτηση που χρησιμοποιείται για τον υπολογισμό της ομοιότητας Jaccard φαίνεται παρακάτω (Ghosh, 2017):

$$S_{AB} = \frac{\sum_{i=1}^{N}(x_{iA}x_{iB})}{\sum_{i=1}^{N}(x_{iA})^2 + \sum_{i=1}^{N}(x_{iB})^2 - \sum_{i=1}^{N}(x_{iA}x_{iB})}$$

2.4.4. Η ομοιότητα του Dice

Η τεχνική της ομοιότητας Dice χρησιμοποιείται όπως η τεχνική της ομοιότητας Jaccard για την εύρεση της ομοιότητας μεταξύ των διανυσμάτων με τη διαφορά ότι "δίνει το διπλάσιο βάρος στις ομοιότητες" (Dice, 1945). Το μέτρο δίνεται από τον τύπο (Chu, Liu, Mao, 2002):

$$\mathrm{D}\ (\mathrm{A}, \mathrm{B}) = 2\frac{|A \cap B|}{(|A|+|B|)}$$

όπου A και B είναι τα δύο σύνολα. Πιο απλά, αυτός ο τύπος αντιπροσωπεύει το μέγεθος της ένωσης 2 σετ διαιρούμενο με το μέσο μέγεθος των δύο σετ.

2.5. Η βαρύτητα των διαφορετικών όρων

Όλες οι παραπάνω τεχνικές που αφορούν το Διανυσματικό Μοντέλο (VSM) πρέπει να υπολογίσουν κάποια τιμή που ονομάζεται "Βάρος του όρου". Αυτή η τιμή είναι κρίσιμη για την ανάκτηση των εγγράφων και την ταξινόμησή τους με βάση τη συνάφεια τους με τις ανάγκες του χρήστη.

Υπάρχουν μερικές βασικές αξίες που πρέπει να υπολογίσουμε για να πάρουμε το όρο βάρος:

- **Συχνότητα όρων (TF).** Η συχνότητα όρων (TF) είναι η τυπική έννοια της συχνότητας στη φυσική επεξεργασία γλώσσας (NLP). Χρησιμοποιείται για να υπολογίσει τις φορές που εμφανίζεται ένας όρος σε ένα έγγραφο (Jones, 1972). Η συχνότητα όρων (TF) χρησιμοποιείται ευρέως επειδή μπορεί εύκολα να υπολογιστεί για κάθε έγγραφο και δίνει καλά αποτελέσματα (Salton, Buckley, 1988).
- **Συχνότητα εγγράφου (DF).** Δεδομένου ότι συχνότητα όρων αντιπροσωπεύει τη σημασία ενός όρου στο έγγραφο, η συχνότητα εγγράφου (DF) αντιπροσωπεύει τη σημασία του όρου σε όλα τα έγγραφα, μετρά τον αριθμό των εγγράφων στα οποίο εμφανίζεται ο όρος. Η συχνότητα εγγράφου έχει πολλαπλές εφαρμογές σε πεδία όπως αυτό της Επεξεργασίας Φυσικής Γλώσσας, της ανάκτηση πληροφοριών και άλλα συναφή πεδία (Jones, 1972).

- **Αντίστροφη συχνότητα εγγράφων (IDF).** Η αντίστροφη συχνότητα εγγράφων θεωρείται ως ένα επιλεκτικό μέτρο ενός όρου στη συλλογή κειμένων. Προτάθηκε το 1972 και από τότε χρησιμοποιείται ευρέως (Jones, 1972). Το IDF στην ανάκτηση πληροφοριών χρησιμοποιείται για να διακρίνει λέξεις που έχουν την ίδια συχνότητα (Wu, Luk, 2008). Το IDF ορίζεται μαθηματικά ως $idf = log_{10}\frac{N}{n_i}$ όπου:

  N = ο αριθμός των εγγράφων στη συλλογή κειμένων και

  ni = είναι η συχνότητα του εγγράφου, ο αριθμός των εγγράφων που περιέχουν τον όρο i.
- **Το βάρος Tf-idf.** Στη συνέχεια συνδυάζουμε τη συχνότητα των όρων (η σημασία κάθε όρου στο έγγραφο (tf)) και την αντίστροφη συχνότητα των εγγράφων (η σημασία του όρου στη συλλογή κειμένου), για να παραχθεί ένα σύνθετο βάρος για κάθε όρο σε κάθε έγγραφο (Wu, Luk, 2008).

Το βάρος αυτό για κάποιον όρο t είναι:

- Υψηλότερο όταν το t συναντάται πολλές φορές μέσα σε ένα μικρό αριθμό εγγράφων
- Μικρότερο όταν ο όρος εμφανίζεται λιγότερες φορές σε ένα έγγραφο, ή εμφανίζεται σε πολλά έγγραφα.
- Ελάχιστο όταν ο όρος εμφανίζεται σε όλα τα έγγραφα.

Έτσι, αν θεωρήσουμε μια συλλογή κειμένων που αποτελείται από 50 έγγραφα, μια λέξη που εμφανίζεται σε κάθε ένα από αυτά τα έγγραφα είναι ένας εντελώς άχρηστος όρος. Επειδή δεν έχει καμία πληροφορία σχετικά με τα έγγραφα για τα οποία ενδιαφέρεται ο χρήστης.

2.6. Πλεονεκτήματα του Διανυσματικού μοντέλου

Το Διανυσματικό μοντέλο έχει κάποια πλεονεκτήματα έναντι των άλλων μοντέλων:

- Είναι ένα απλό μοντέλο βασισμένο στη γραμμική άλγεβρα.
- Τα βάρη των όρων δεν είναι δυαδικά.
- Επιτρέπει τον υπολογισμό ενός συνεχούς βαθμού ομοιότητας μεταξύ ερωτημάτων και εγγράφων.
- Επιτρέπει την ταξινόμηση των εγγράφων ανάλογα με την πιθανή τους σημασία.

- Επιτρέπει τη μερική αντιστοίχιση.

Τα περισσότερα από αυτά τα πλεονεκτήματα είναι συνέπεια της διαφοράς στην πυκνότητα των αναπαραστάσεων των εγγράφων μεταξύ των δυαδικών προσεγγίσεων (Boolean) και των TF-IDF προσεγγίσεων. Όταν χρησιμοποιούμε δυαδικά βάρη, κάθε έγγραφο βρίσκεται σε μια κορυφή σε ένα n-διαστάσεων υπερκύβο. Επομένως, οι πιθανές αναπαραστάσεις εγγράφων είναι 2n και η μέγιστη ευκλείδεια απόσταση μεταξύ των ζευγών είναι $\sqrt{n}$. Καθώς προστίθενται έγγραφα στη συλλογή, η περιοχή που ορίζεται από τις κορυφές του υπερκύβου γίνεται πυκνότερη. Σε αντίθεση με τη δυαδική προσέγγιση, όταν προστίθεται ένα έγγραφο χρησιμοποιώντας TF-IDF βάρη, οι IDFs των όρων στο νέο έγγραφο μειώνονται ενώ οι υπόλοιποι όροι αυξάνονται. Κατά μέσο όρο, καθώς προστίθενται τα έγγραφα, η περιοχή όπου βρίσκονται τα έγγραφα, διευρύνεται, ρυθμίζοντας την πυκνότητα της αναπαράστασης ολόκληρης της συλλογής. Αυτή η συμπεριφορά μοντελοποιεί το αρχικό κίνητρο του Salton και των συναδέλφων του ότι μια συλλογή εγγράφων που αντιπροσωπεύεται από μια περιοχή χαμηλής πυκνότητας θα μπορούσε να αποφέρει καλύτερα αποτελέσματα ανάκτησης πληροφοριών (Salton, Wong, Yang, 1975).

2.7. Περιορισμοί

Το Διανυσματικό μοντέλο έχει και κάποιους περιορισμούς:

Τα μεγάλα έγγραφα δεν αντιπροσωπεύονται επαρκώς επειδή έχουν πολύ μικρές τιμές ομοιότητας (μικρό εσωτερικό γινόμενο και πολλές διαστάσεις)

- Οι λέξεις-κλειδιά της αναζήτησης πρέπει να αντιστοιχούν με ακρίβεια στους όρους του εγγράφου, η τμηματική ομοιότητα κάποιων λέξεων ενδέχεται να οδηγήσει σε "ψευδή θετική αντιστοίχιση".
- Σημασιολογική ευαισθησία. Τα έγγραφα με παρόμοιο περιεχόμενο, αλλά διαφορετική ορολογία δεν θα συσχετιστούν, οδηγώντας σε μια "ψευδή αρνητική αντιστοίχιση".
- Η σειρά με την οποία εμφανίζονται οι όροι στο έγγραφο χάνεται στην αναπαράσταση με το Διανυσματικό Μοντέλο.
- Θεωρητικά οι όροι είναι στατιστικά ανεξάρτητοι.
- Η στάθμιση είναι διαισθητική αλλά όχι αυστηρά καθορισμένη.

Πολλές από αυτές τις δυσκολίες μπορούν, ωστόσο, να ξεπεραστούν με τη χρήση διαφόρων εργαλείων, συμπεριλαμβανομένων κάποιων μαθηματικών τεχνικών όπως η ανάλυση πίνακα σε ιδιαζουσες τιμές SVD (Singular Value Decomposition) και βάσεων δεδομένων με λέξεις όπως το WordNet.

2.8. Παράδειγμα εφαρμογής του Διανυσματικού Μοντέλου

Στην παράγραφο αυτή δίνεται ένα απλοποιημένο παράδειγμα του Διανυσματικού Μοντέλου (Teufe, 2014). Έστω μια πολύ μικρή συλλογή C που αποτελείται από τα ακόλουθα τρία έγγραφα:

d1: “new york times”

d2: “new york post”

d3: “los angeles times”

Μερικοί όροι εμφανίζονται σε δύο έγγραφα, μερικοί εμφανίζονται μόνο σε ένα έγγραφο. Ο συνολικός αριθμός των εγγράφων είναι N = 3. Επομένως, οι τιμές IDF για τους όρους είναι:

angles log2(3/1)=1.584

los log2(3/1)=1.584

new log2(3/2)=0.584

post log2(3/1)=1.584

times log2(3/2)=0.584

york log2(3/2)=0.584

Για όλα τα έγγραφα, υπολογίζουμε τις τιμές TF για όλους τους όρους στο C. Υποθέτουμε ότι οι λέξεις στα διανύσματα ταξινομούνται αλφαβητικά.

	angeles	los	new	post	times	york
d1	0	0	1	0	1	1
d2	0	0	1	1	0	1
d3	1	1	0	0	1	0

q	0	0	(2/2)*0.584=0.584	0	(1/2)*0.584=0.292	0

Πολλαπλασιάζουμε τις τιμές TF με τις τιμές IDF κάθε όρου, και παίρνουμε τον ακόλουθο πίνακα: (Όλοι οι όροι εμφανίστηκαν μόνο μία φορά σε κάθε έγγραφο στη μικρή συλλογή μας, οπότε η μέγιστη τιμή για κανονικοποίηση είναι 1).

	angeles	los	new	post	times	york
d1	0	0	0.584	0	0.584	0.584
d2	0	0	0.584	1.584	0	0.584
d3	1.584	1.584	0	0	0.584	0

Δεδομένου του ακόλουθου ερωτήματος: “new times”, υπολογίζουμε το διάνυσμα TF-IDF για το ερώτημα και υπολογίζουμε την τιμή κάθε εγγράφου στη συλλογή C σε σχέση με αυτό το ερώτημα, χρησιμοποιώντας το μέτρο ομοιότητας του συνημιτόνου. Κατά τον υπολογισμό των τιμών TF-IDF για τους όρους του ερωτήματος διαιρούμε τη συχνότητα με τη μέγιστη συχνότητα και πολλαπλασιάζουμε με τις τιμές IDF.

Υπολογίζουμε το μήκος του κάθε εγγράφου και του ερωτήματος:

Length of d1 = $\sqrt{(0.584^2 + 0.584^2 + 0.584^2)} = 1.011$

Length of d2 = $\sqrt{(0.584^2 + 1.584^2 + 0.584^2)} = 1.786$

Length of d3 = $\sqrt{1.584^2 + 1.584^2 + 0.584^2} = 2.316$

Length of q = $\sqrt{0.584^2 + 0.292^2} = 0.652$

Τότε, οι τιμές ομοιότητας είναι:

$$cosSim(d1,q) = \frac{(0x0 + 0x0 + 0.584x0.584 + 0x0 + 0.584x0.292 + 0.584x0)}{(1.011x0.652)} = 0.776$$

$$cosSim(d2,q) = \frac{(0x0 + 0x0 + 0.584x0.584 + 1.584x0 + 0x0.292 + 0.584x0)}{(1.786x0.652)} = 0.292$$

$$cosSim(d3,q) = \frac{(1.584x0 + 1.584x0 + 0x0.584 + 0x0 + 0.584x0.292 + 0x0)}{(2.316x0.652)} = 0.112$$

Σύμφωνα με τις τιμές ομοιότητας, η τελική σειρά με την οποία παρουσιάζονται τα έγγραφα ως αποτέλεσμα του ερωτήματος θα είναι: d1, d2, d3.

2.9. Word2vec

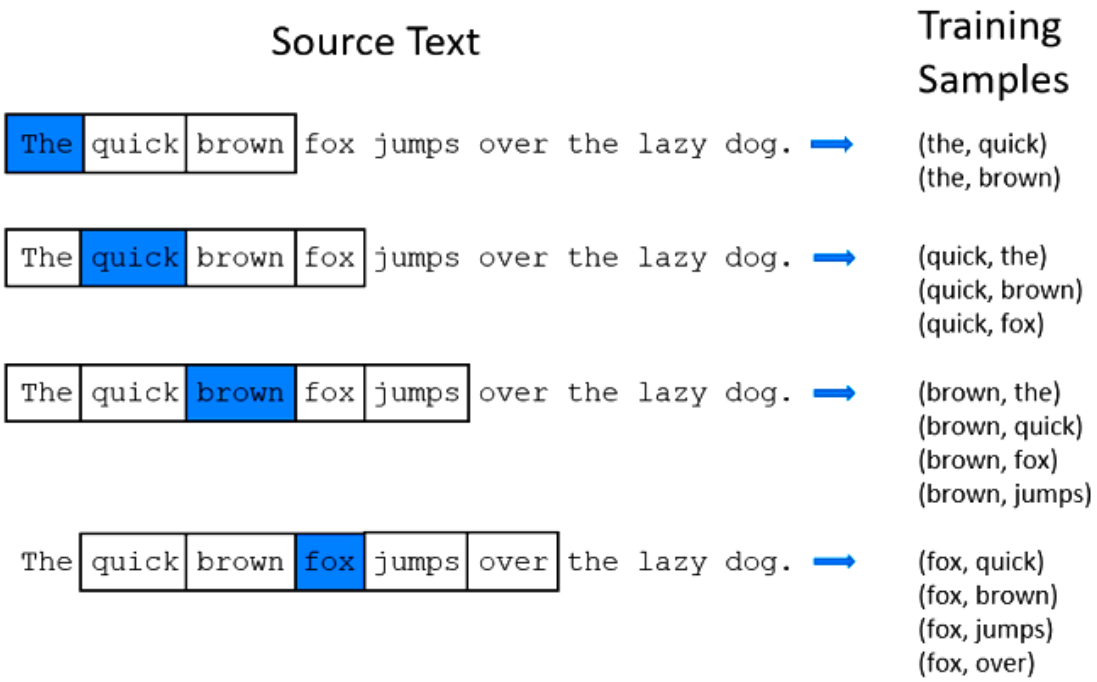
Το Word2vec είναι ένα νευρωνικό δίκτυο δύο επιπέδων που επεξεργάζεται κείμενα. Η είσοδός του είναι ένα σώμα κειμένου και η έξοδος του είναι ένα σύνολο διανυσμάτων χαρακτηριστικών για λέξεις σε αυτό το σώμα. Εισήχθη το 2013 από μια ομάδα ερευνητών με επικεφαλής τον Tomas Mikolov (Mikolov, 2014) στην Google.

Είναι ένα μοντέλο βασισμένο στις προβλέψεις και όχι στη συχνότητα. Χρησιμοποιεί την αναλυτική πρόβλεψη για να κάνει μια σταθμισμένη εικασία μιας λέξης με βάση τις γειτονικές της λέξεις. Υπάρχουν δύο παραλλαγές σε αυτό το μοντέλο:

- CBOW (Continuous Bag of Words): Αυτό το μοντέλο προσπαθεί να προβλέψει μια λέξη με βάση τις γειτονικές λέξεις.
- SkipGram: Αυτό το μοντέλο προσπαθεί να προβλέψει τις γειτονικές λέξεις κάποιας λέξης.

Το CBOW τείνει να βρει την πιθανότητα μιας λέξης να εμφανίζεται σε μια “γειτονιά”. Έτσι εξετάζει όλα τα διαφορετικά πλαίσια στα οποία μπορεί να χρησιμοποιηθεί μια λέξη. Ενώ το SkipGram μοντέλο τείνει να μάθαινει ξεχωριστά τα διαφορετικά πλαίσια.

Η έννοια της λέξης "γειτονιά" περιγράφεται καλύτερα λαμβάνοντας υπόψη μια κεντρική λέξη και ένα παράθυρο λέξεων γύρω από αυτήν. Για παράδειγμα, εάν εξετάσουμε την πρόταση "The quick brown fox jumped over the lazy dog", και ένα μέγεθος παραθύρου 2, θα έχουμε για το μοντέλο skip-gram τα ζεύγη που απεικονίζονται στο Σχήμα 8.



Σχήμα 8. Ζεύγη για το μοντέλο skip-gram που αντιπροσωπεύουν την πρόταση "The quick brown fox jumped over the lazy dog"

Αντίθετα, για το μοντέλο CBOW, εισάγουμε λέξεις μέσα στο παράθυρο (όπως το "the", "brown", "fox") και στοχεύουμε στην πρόβλεψη της λέξης "quick" (απλά αναστρέφοντας την είσοδο στη διαδικασία πρόβλεψης από το μοντέλο skip-gram).

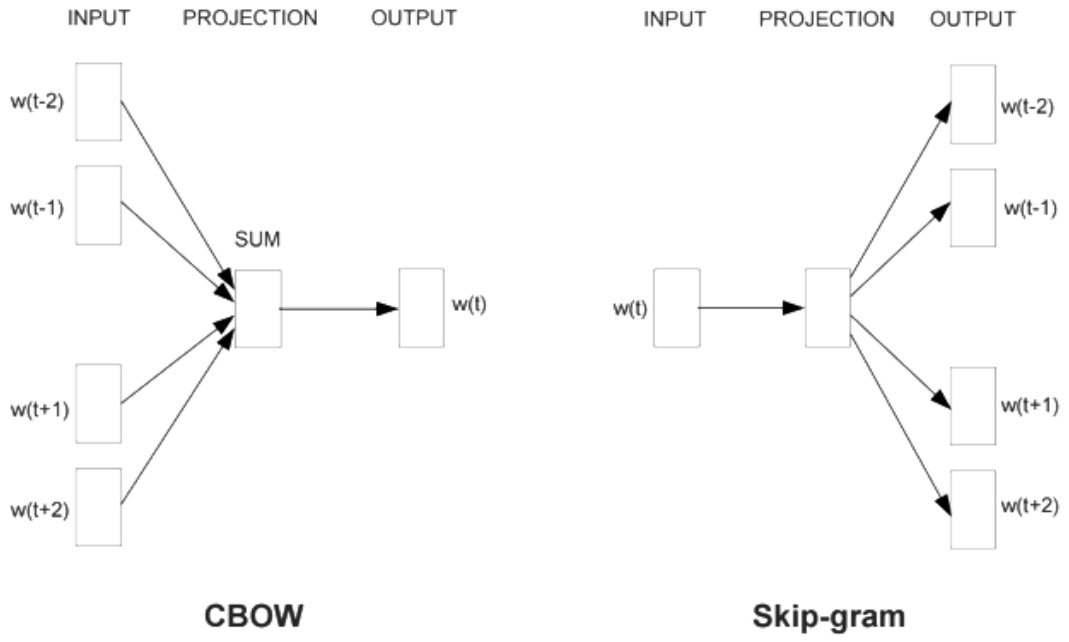
Στο Σχήμα 9, δίνεται μια απεικόνιση των μοντέλων skip-gram και CBOW.

Δεδομένου ότι το SkipGram είχε αποδειχθεί ότι έχει καλύτερες επιδόσεις από το μοντέλο CBOW, είναι προτιμότερο να χρησιμοποιείται το μοντέλο SkipGram ως επί το πλείστον. Γενικά, αν κατανοήσει κανείς το ένα μοντέλο, είναι αρκετά εύκολο να καταλάβει και το άλλο. Απλά το μόνο που πρέπει να κάνει είναι να αντιστρέψει τις εισόδους και τις προβλέψεις.

Το μοντέλο skip-gram είναι πραγματικά απλό. Το Word2Vec χρησιμοποιεί ένα κόλπο που μπορεί κανείς να δει και αλλού στη μηχανική μάθηση (machine learning). Πρόκειται να γίνει εκπαίδευση σε ένα απλό νευρωνικό δίκτυο με ένα κρυφό στρώμα για να εκτελεστεί ένα συγκεκριμένο έργο, αλλά στη συνέχεια δεν θα χρησιμοποιηθεί αυτό το νευρωνικό δίκτυο για το έργο που εκπαιδεύτηκε! Αντ' αυτού, ο στόχος στην πραγματικότητα είναι

απλώς να εντοπιστούν τα βάρη του κρυμμένου στρώματος - αυτά τα βάρη είναι στην πραγματικότητα τα “διανύσματα των λέξεων”.

Η εκπαίδευση του νευρωνικού δικτύου (McCormick,2016), γίνεται όπως στο παραπάνω σχήμα, τροφοδοτώντας το με ζεύγη λέξεων που υπάρχουν στα έγγραφα της συλλογής. Λαμβάνοντας μια συγκεκριμένη λέξη στη μέση μιας φράσης (λέξη εισόδου), μπορεί να δει κανείς τις λέξεις που βρίσκονται κοντά και να επιλέξει μια τυχαία. Το δίκτυο πρόκειται να αποφανθεί για την πιθανότητα κάθε λέξης στο λεξιλόγιο να είναι η "κοντινή λέξη" που επιλέχθηκε.



Σχήμα 9. Απεικόνιση των μοντέλων skip-gram και CBOW

Οι πιθανότητες εξόδου θα σχετίζονται με το πόσο πιθανό είναι να βρεθεί κάθε λέξη του λεξιλογίου κοντά στη λέξη εισόδου. Για παράδειγμα, εάν δοθεί στο εκπαιδευμένο δίκτυο η λέξη εισόδου "Σοβιετική", οι πιθανότητες να είναι “κοντινές λέξεις” θα είναι πολύ υψηλότερες για λέξεις όπως "Ένωση" και "Ρωσία" από ό, τι για μη σχετικές λέξεις όπως "καρπούζι" και “καγκουρό".

Το δίκτυο πρόκειται να μάθει τα στατιστικά στοιχεία ανάλογα με το πόσες φορές εμφανίζεται κάθε αντιστοίχιση. Έτσι, για παράδειγμα, το δίκτυο πιθανότατα θα πάρει πολλά περισσότερα δείγματα εκπαίδευσης ("Σοβιετική", "Ένωση") από ότι ("σοβιετικό", "καγκουρό"). Όταν τελειώσει η εκπαίδευση, αν δοθεί η λέξη "Σοβιετική" ως είσοδος, τότε θα προκύψει μια πολύ υψηλότερη πιθανότητα για τις λέξεις "Ένωση" ή "Ρωσία" από ό, τι για το "καγκουρό".

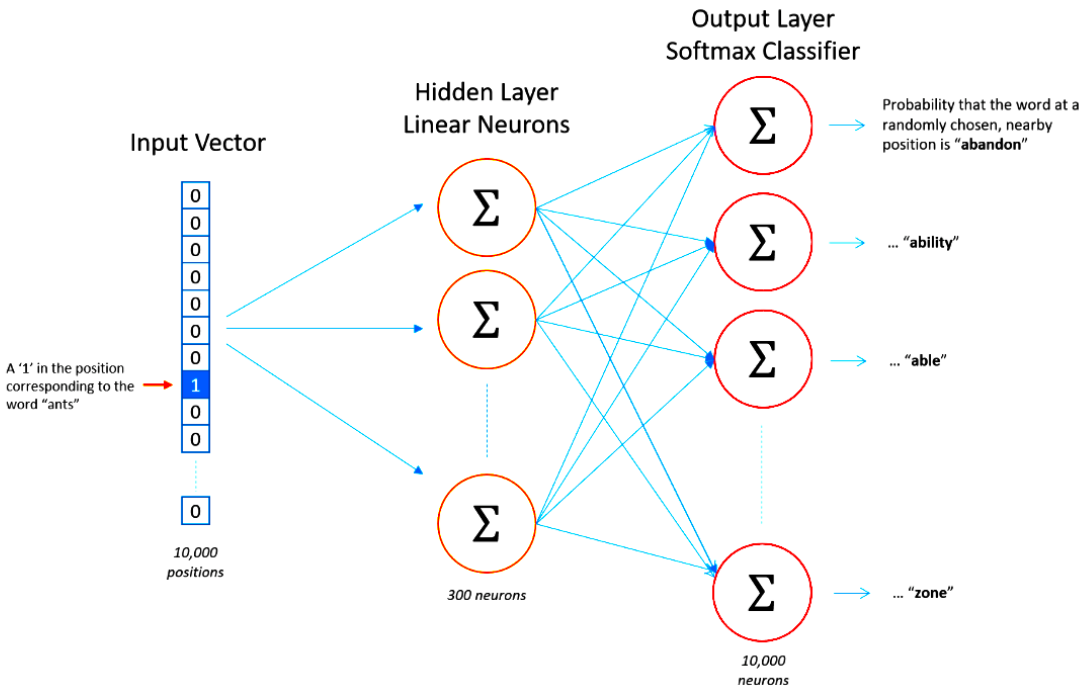
Προκύπτει λοιπόν το ερώτημα της αναπαράστασης ενός τέτοιου παραδείγματος.

Αρχικά, δεν μπορεί να τροφοδοτηθεί ένα νευρωνικό δίκτυο με μια λέξη σαν μια συμβολοσειρά κειμένου, οπότε χρειάζεται ένας τρόπος εκπροσώπησης των λέξεων στο δίκτυο. Για να γίνει αυτό, δημιουργείται πρώτα ένα λεξιλόγιο από τα έγγραφα που χρησιμοποιούνται για την εκπαίδευση του δικτύου- ας πούμε για παράδειγμα ένα λεξιλόγιο 10.000 μοναδικών λέξεων.

Μια λέξη εισόδου αναπαρίσταται ως ένα διάνυσμα. Αυτό το διάνυσμα θα έχει 10.000 συνιστώσες (μια για κάθε λέξη στο λεξιλόγιο) και θα τοποθετηθεί ένα “1” στη θέση που αντιστοιχεί στη λέξη, και “0” σε όλες τις άλλες θέσεις.

Η έξοδος του δικτύου είναι ένα απλό διάνυσμα (επίσης με 10.000 στοιχεία) που περιέχει, για κάθε λέξη στο λεξιλόγιο, την πιθανότητα μια τυχαία επιλεγμένη κοντινή λέξη να είναι η συγκεκριμένη λέξη από το λεξιλόγιο.

Στο παρακάτω Σχήμα 10 απεικονίζεται η αρχιτεκτονική του νευρωνικού δικτύου.



Σχήμα 10. Η αρχιτεκτονική του νευρωνικού δικτύου

Κατά την εκπαίδευση αυτού του δικτύου σε ζεύγη λέξεων, η είσοδος είναι ένα διάνυσμα που αντιπροσωπεύει τη λέξη εισόδου και η έξοδος της είναι επίσης ένα διάνυσμα που αντιπροσωπεύει τη λέξη εξόδου. Αλλά όταν αξιολογείται το εκπαιδευμένο δίκτυο σε μια λέξη εισόδου, το διάνυσμα εξόδου θα είναι στην πραγματικότητα μια κατανομή πιθανότητας.

Κεφάλαιο 3ο. Επίδειξη τεχνικών επεξεργασίας κειμένων και ανάκτησης πληροφοριών

Εισαγωγή

Στο παρόν κεφάλαιο θα υλοποιηθούν μερικές από τις τεχνικές που αναφέρθηκαν παραπάνω, σε μία προσπάθεια να επιδειχτεί πόσο αποτελεσματικές είναι. Τα μοντέλα που παρουσιάζονται σε αυτό το κεφάλαιο είναι:

- Bag of Words (BoW)
- Latent Semantic Analysis (LSA)

Για την υλοποίηση του LSA και BoW χρησιμοποιήθηκαν επιπλέον οι εξωτερικές βιβλιοθήκες Jama (Hicklin, Moler, 2012), που προσφέρουν ένα πακέτο για την δημιουργία και επεξεργασία πινάκων, καθώς και η JFreeChart (Viklund, 2013) για τον σχεδιασμό γραφημάτων.

3.1. Εξαγωγή χαρακτηριστικών με το Bag of Words

Το Bag of Words είναι μία μέθοδος εξαγωγής χαρακτηριστικών από αρχεία διάφορων κειμένων. Τα χαρακτηριστικά αυτά στη συνέχεια μπορούν να χρησιμοποιηθούν από αλγόριθμους μηχανικής μάθησης, όπως ο Word2Vec που παρουσιάζεται παραπάνω (Υποκεφάλαιο 2.10). Ουσιαστικά πρόκειται για μία συλλογή λέξεων η οποία αναπαριστά το κείμενο μέσω της δημιουργίας διανυσμάτων. Τα βήματα που ακολουθεί η τεχνική αυτή είναι:

1. Καθαρισμός κειμένου
2. Εξαγωγή των λέξεων
3. Δημιουργία λεξιλογίου
4. Δημιουργία Διανύσματος

Αρχικά ορίζονται οι βιβλιοθήκες της Java που πρόκειται να χρησιμοποιηθούν στον κώδικα:

```
import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Comparator;
```

```
import java.util.Hashtable;
import java.util.List;
import java.util.Map;
import java.util.Map.Entry;
```

Στη συνέχεια γίνεται ο ορισμός της κλάσης και των πεδίων που την απαρτίζουν, καθώς και της συνάρτησης δημιουργίας. Οι δομές δεδομένων που προτιμήθηκαν για την αποθήκευση και την επεξεργασία των πληροφοριών είναι οι λίστες και οι πίνακες κατακερματισμού, οι οποίοι μας επιτρέπουν να ορίσουμε σε κάθε τιμή και ένα κλειδί. Στο πεδίο vocabulary αποθηκεύεται το λεξιλόγιο των κειμένων μαζί με τον αριθμό εμφάνισης της κάθε λέξης στα κείμενα. Το πεδίο documents είναι μία λίστα όλων των λέξεων, όλων των κειμένων που το Bag Of Words έχει επεξεργαστεί. Πρόκειται για μία λίστα από πίνακες κατακερματισμού, όπου ο κάθε πίνακας αποθηκεύει κάθε λέξη που εμφανίζεται στο εκάστοτε κείμενο, καθώς και τον αριθμό συνολικών εμφανίσεων της. Οι ονομασίες των αρχείων αποθηκεύονται στη λίστα DocNames, ενώ η λίστα WeightedDoc κρατάει σαν πληροφορία το βάρος κάθε λέξης σε κάθε κείμενο. Σαν βάρος ορίζεται η βαρύτητα/σημαντικότητα που έχει κάθε λέξη στο κείμενο και θα εξεταστεί παρακάτω. Τέλος, η λίστα vectors απεικονίζει τα τελικά διανύσματα που τα παραχθούν από το Bag Of Words στο τέλος της επεξεργασίας.

```
public class BoW {

    private Hashtable<String, Double> vocabulary = new Hashtable<>();
    private List<Hashtable<String,Integer>> documents = new ArrayList<>();
    private List<String> DocNames = new ArrayList<>();
    private List<Hashtable<String,Double>> WeightedDocs = new ArrayList<>();
    private List<Integer[]> vectors = new ArrayList<>();

    public BoW() {}
```

Η ακόλουθη μέθοδος είναι υπεύθυνη για την εισαγωγή ενός κειμένου στο Bag of Words και γι’ αυτό είναι η πρώτη που «καλείται». Η διαδικασία αρχικά διαβάζει γραμμή προς γραμμή το κείμενο, αφαιρώντας τυχόν περιττούς, κατά την επεξεργασία, χαρακτήρες (όπως οι κενοί χαρακτήρες ή τα σημεία στίξης) και μετατρέποντας όλες τις λέξεις σε πεζή απεικόνιση για να μην υπάρχουν διπλότυπες. Στη συνέχεια η λέξη προστίθεται στο γενικό

λεξιλόγιο που αποτελείται από όλα τα κειμενα που έχουν εισαχθεί και στο λεξιλόγιο του κειμένου τιθεται υπό επεξεργασία.

```
public void addDoc(File file) {
    Hashtable<String,Integer> bag = new Hashtable<>();
    try (BufferedReader br = new BufferedReader(new FileReader(file))) {
    String line;
       try {
          while ((line = br.readLine()) != null) {
             line = line.trim().replaceAll("[^A-Za-z]\\s", " ").replaceAll("[^A-Za-z]", " ");
             line = line.toLowerCase();
             for (String str: line.split(" ")) {
                if (str.isEmpty() || str.length() == 1) continue;
                if (line.isEmpty()) break;
                str = str.trim().toLowerCase();
                if (!bag.containsKey(str)){
                   bag.put(str, 1);
                }
                else{
                   bag.replace(str, bag.get(str)+1);
                }
                if (!vocabulary.containsKey(str)){
                   vocabulary.put(str, (double) 1);
                }
                else{
                   vocabulary.replace(str, vocabulary.get(str)+1);
                }
             }
          }
          br.close();
       } catch (IOException e) {
          e.printStackTrace();
       }
    } catch (FileNotFoundException e1) {
       e1.printStackTrace();
    } catch (IOException e1) {
       e1.printStackTrace();
    }
    documents.add(bag);
 }
```

Μετά την ολοκλήρωση δημιουργίας του λεξιλογίου, ακολουθεί η εξαγωγή των διανυσμάτων. Όλα τα διανύσματα έχουν το ίδιο μήκος, το οποίο είναι ο συνολικός αριθμός των λέξεων στο γενικό λεξιλόγιο. Το παραγόμενο είναι ένα δυαδικό διάνυσμα όπου κάθε συνιστώσα έχει την τιμή 1 ή 0 ανάλογα με την εμφάνιση της ή μη (αντίστοιχα) στο γενικό λεξιλόγιο.

```
public void Vectorize() {
   Object[] temp = vocabulary.keySet().toArray();
   for (int j = 0; j < documents.size(); j++) {
      Integer[] vector = new Integer[vocabulary.size()];
      for (int i = 0; i < vocabulary.size(); i++) {
         if (documents.get(j).containsKey(temp[i]))
            vector[i] = 1;
         else
            vector[i] = 0;
      }
      vectors.add(vector);
   }
}
```

Αξίζει να αναφερθεί ότι το παραγόμενο δεν είναι απαραίτητο να είναι δυαδικό διάνυσμα. Σε αυτή την εκδοχή του Bag Of Words κάθε συνιστώσα του διανύσματος απεικονίζει τον αριθμό εμφάνισης της αντίστοιχης λέξης στο κείμενο. Η εκάστοτε εκδοχή έχει και την ανάλογη χρησιμότητα η οποία εξαρτάται από τις ανάγκες του χρήστη. Στη συνέχεια εξετάζονται μερικές από τις επεκτάσεις του Bag Of Words στην προσπάθεια να εξυπηρετήσει περισσότερες ανάγκες του χρήστη.

- Η μέθοδος TFIDF (Term Frequency – Inverse Document Frequency) είναι μία διαδικασία εξαγωγής θορύβου από το λεξιλόγιο, δηλαδή των λέξεων που επηρεάζουν την επεξεργασία λόγω της πολλαπλής εμφάνισης τους, χωρίς όμως να έχουν κάποια ιδιαίτερη σημασία για το κείμενο. Παράδειγμα τέτοιων λέξεων είναι τα άρθρα ή ο σύνδεσμος «και». Ο τύπος στον οποίο βασίζεται η μέθοδος είναι ο:

$$\mathrm{TFIDF}_{i,j} = \left(\frac{N_{i,j}}{N_{*,j}}\right) x\ Log\left(\frac{D}{D_i}\right)$$

όπου

- $N_{i,j}$ = Ο συνολικός αριθμός όπου μία λέξη **i** εμφανίζεται στο κείμενο **j**.
- $N_{*,j}$ = Ο συνολικός αριθμός των λέξεων στο κείμενο **j**.
- **D** = Ο συνολικός αριθμός των κειμένων.
- D_i = Ο συνολικός αριθμός των κειμένων όπου εμφανίζεται η λέξη **i**.

Μετά την διαδικασία εξαγωγής θορύβου, οι λέξεις εισάγονται στη λίστα WeightedDocs όπου κάθε λέξη εμφανίζεται σύμφωνα με την βαρύτητα που είχε στο κείμενο από το οποίο προήλθε.

```
public void TF_IDF() {
      for (int i = 0 ; i < documents.size(); i++) {
         Hashtable<String, Double> weights = new Hashtable<>(documents.get(i).size());
         for (String word:documents.get(i).keySet()) {
            int d = 0;
            for (Hashtable<String,Integer> doc : documents) {
               if (doc.containsKey(word)) d++;
            }
            double W = documents.get(i).get(word) * Math.log10((double) documents.size() / (double) d);
            weights.put(word, W);
         }
         WeightedDocs.add(weights);
      }
   }
```

- Η απόσταση Hamming είναι ένας τρόπος σύγκρισης διανυσμάτων και είναι ευρέως γνωστή, με πολλές εφαρμογές σε διάφορους τομείς της επιστήμης αλλά και στην θεωρία πληροφορίας. Ορίζεται ως ο συνολικός αριθμός των διαφορετικών συνιστωσών δύο διανυσμάτων, με τη προϋπόθεση ότι έχουν το ίδιο μήκος. Στη προκειμένη περίπτωση είναι ένας εύκολος τρόπος να γίνει σύγκριση μεταξύ δύο κειμένων. Εφόσον κάθε κείμενο απεικονίζεται σαν διάνυσμα του οποίου η κάθε συνιστώσα αναφέρεται στην ίδια λέξη, η απόσταση hamming θα είναι η διαφορά μεταξύ των λέξεων κάθε κειμένου.

```
public int getHammingDistance(int i, int j) {
   Integer[] vector1 = vectors.get(i);
   Integer[] vector2 = vectors.get(j);
   int result = 0;
   for (int b = 0; b < vector1.length; b++) {
      byte b1 = vector1[b].byteValue();
      byte b2 = vector2[b].byteValue();
      Byte b3 = (byte) (b1 ^ b2);
      if (b3.intValue() == 1) {
         result++;
      }
   }
   return result;
}
```

- Ένας άλλος τρόπος σύγκρισης δύο διανυσμάτων είναι ο υπολογισμός του συνημίτονου της γωνίας μεταξύ τους. Η ομοιότητα των δύο διανυσμάτων ορίζεται ως το συνημίτονο της γωνίας αυτής:

$$Similarity = Cos(\theta) = \frac{A\ x\ B}{||A||\ x\ ||B||}$$

Κατά συνέπεια, όσο πιο κοντά στο 1 είναι το αποτέλεσμα, τόσο πιο όμοια είναι τα δύο κείμενα. Είναι προφανές ότι όταν το συνημίτονο είναι 1 τότε τα διανύσματα συμπίπτουν, δηλαδή τα δύο κείμενα είναι ίδια.

Για να υλοποιηθεί αυτή η μέθοδος, τα δυαδικά διανύσματα δεν αρκούν. Γι' αυτό το λόγο θα πρέπει δημιουργηθούν νέα διανύσματα με γνώμονα τον συνολικό αριθμό εμφάνισης κάθε λέξης στο κείμενο. Για λόγους ταχύτητας υπολογισμού και εξοικονόμησης υπολογιστικής ισχύος, προτείνεται τα διανύσματα να περάσουν πρώτα από μία διαδικασία κανονικοποίησης. Η διαδικασία αυτή θα μετατρέψει κάθε διάνυσμα σε μοναδιαίο. Ένα μοναδιαίο διάνυσμα έχει μέτρο 1, ή || A || = 1. Ο τύπος για την μετατροπή ενός διανύσματος σε μοναδιαίο είναι:

$$u = \frac{A}{||A||}$$

Με την μετατροπή του διανύσματος **A** στο μοναδιαίο **u1** και αντίστοιχα του **B** στο μοναδιαίο **u2,** ο παραπάνω τύπος της συνημιτονικής ομοιότητας απλοποιείται σε

$$Similarity = Cos(\theta) = u1\ x\ u2$$

δηλαδή στο εσωτερικό γινόμενο των παραγόμενων μοναδιαίων.

Ακολουθεί ο κώδικας υπολογισμού της συνημιτονικής ομοιότητας.

```
public String getCosineSimilarity(int i, int j) {
    double[] vec1 = new double[vocabulary.keySet().size()];
    double[] vec2 = new double[vocabulary.keySet().size()];
    int z = 0;
    for (String word : vocabulary.keySet()) {
      if (!documents.get(i).containsKey(word) && !documents.get(j).containsKey(word)) {
        z++;
        continue;
      } else if (documents.get(i).containsKey(word) && !documents.get(j).containsKey(word)) {
        vec1[z] = (double) documents.get(i).get(word);
        vec2[z] = 0.0;
      } else if (!documents.get(i).containsKey(word) && documents.get(j).containsKey(word)) {
        vec2[z] = (double) documents.get(j).get(word);
        vec1[z] = 0.0;
      } else {
```

```
            vec1[z] = (double) documents.get(i).get(word);
            vec2[z] = (double) documents.get(j).get(word);
        }
        z++;
    }
    vec1 = normalize(vec1);
    vec2 = normalize(vec2);
    double dotProduct = 0.0;
    for (z = 0; z < vocabulary.keySet().size(); z++) {
        dotProduct += vec1[z] * vec2[z];
    }
    NumberFormat nf = new DecimalFormat("0.########");
    return nf.format(dotProduct).replace(",", ".");
}

public double[] normalize(double[] m) {
    double[] normalized = new double[m.length];
    double d = 0.0;
    for (int i = 0; i < m.length; i++) {
        d += Math.pow(m[i], 2);
    }
    double norm = Math.sqrt(d);
    for (int i = 0; i < m.length; i++) {
        normalized[i] = (m[i] / norm);
    }
    return normalized;
}
```

Η ακόλουθη συνάρτηση επιτρέπει στον χρήστη να εισάγει μία πρόταση και να πάρει ως αποτέλεσμα το κείμενο με το οποίο η πρόταση αυτή έχει την μεγαλύτερη ομοιότητα. Η σύγκριση γίνεται με την χρήση της προηγούμενης μεθόδου, της συνημιτονικής ομοιότητας.

```
public int Search(String s) {
    Hashtable<String,Integer> bag = new Hashtable<>();
    for (String str: s.split(" ")) {
        if (str.isEmpty() || str.length() == 1) return -1;
        str = str.trim().replaceAll("[^A-Za-z]\\s", " ").replaceAll("[^A-Za-z]", " ").toLowerCase();
        if (!bag.containsKey(str)){
            bag.put(str, 1);
        }
        else{
            bag.replace(str, bag.get(str)+1);
        }
        if (!vocabulary.containsKey(str)){
            vocabulary.put(str, (double) 1);
        }
        else{
            vocabulary.replace(str, vocabulary.get(str)+1);
        }
    }
```

```
        documents.add(bag);
        double similarity = 0.0;
        int result = -1;
        for (int i = 0; i < documents.size() - 1; i++) {
            String sim = getCosineSimilarity(i, documents.size() - 1);
            if (Double.valueOf(sim) > similarity) {
                similarity = Double.valueOf(sim);
                result = i;
            }
        }
        documents.remove(bag);
        return result;
}
```

Τέλος, παρατίθενται μερικές βοηθητικές μέθοδοι οι οποίες διευκολύνουν την ανάπτυξη του κώδικα και την εμφάνιση των αποτελεσμάτων:

```
public Hashtable<String, Double> GetVocab() {
    return this.vocabulary;
}


public List<Hashtable<String, Integer>> getDocs() {
    return documents;
}

public Hashtable<String, Integer> getDoc(int i) {
    return documents.get(i);
}

public Integer[] getVector(int i) {
    return vectors.get(i);
}

public List<Double> GetWeightedDoc(int i) {
    List<Double> list = new ArrayList<>(WeightedDocs.get(i).values());
    Collections.sort(list);
    return list;
}

public List<String> getDocNames() {
    return DocNames;
}
```

Εκτύπωση στην κονσόλα του λεξιλογίου του κειμένου που δίνεται σαν παράμετρος, με φθίνουσα σειρά του αριθμού εμφάνισης των λέξεων:

```
public void ShowDoc(int i){
   Hashtable<String, Integer> doc = documents.get(i);
   ArrayList<Map.Entry<?, Integer>> l = new ArrayList<Entry<?, Integer>>(doc.entrySet());
   Collections.sort(l, new Comparator<Map.Entry<?, Integer>>(){

      public int compare(Map.Entry<?, Integer> o1, Map.Entry<?, Integer> o2) {
         return o2.getValue().compareTo(o1.getValue());
      }});
   for (Entry<?, Integer> entry:l) {
      System.out.println(entry.getKey() + " : " + entry.getValue());
   }
}
```

Εκτύπωση στην κονσόλα του λεξιλογίου του κειμένου που δίνεται σαν παράμετρος, με φθίνουσα σειρά της υπολογισμένης βαρύτητας κάθε λέξης:

```
public void ShowWeightedDoc(int i){
   Hashtable<String, Double> doc = this.WeightedDocs.get(i);

   ArrayList<Map.Entry<?, Double>> l = new ArrayList<Entry<?, Double>>(doc.entrySet());
   Collections.sort(l, new Comparator<Map.Entry<?, Double>>(){

      public int compare(Map.Entry<?, Double> o1, Map.Entry<?, Double> o2) {
         return o2.getValue().compareTo(o1.getValue());
      }});
   NumberFormat nf = new DecimalFormat("0.#######");
   for (Entry<?, Double> entry:l) {
      System.out.println(entry.getKey() + " : " + nf.format(entry.getValue()));
   }
}
```

Εκτύπωση στη κονσόλα του γενικού λεξιλογίου, με φθίνουσα σειρά του αριθμού εμφάνισης των λέξεων:

```
public void ShowVocab(){

   ArrayList<Map.Entry<?, Double>> l = new ArrayList<Entry<?, Double>>(vocabulary.entrySet());
   Collections.sort(l, new Comparator<Map.Entry<?, Double>>(){

      public int compare(Map.Entry<?, Double> o1, Map.Entry<?, Double> o2) {
         return o2.getValue().compareTo(o1.getValue());
      }});
   NumberFormat nf = new DecimalFormat("0.########");
   for (Entry<?, Double> entry:l) {
      System.out.println(entry.getKey() + " : " + nf.format(entry.getValue()));
   }
}
```

Εκτύπωση στη κονσόλα των παραγόμενων δυαδικών διανυσματων για κάθε κείμενο που έχει επεξεργαστεί το μοντέλο:

```
public void ShowVectors() {
   for (int j = 0; j < vectors.size(); j++) {
      System.out.print("Doc" + j + ": [");
      for (int i : vectors.get(j)) {
         System.out.print(i);
      }
      System.out.println("]");
   }
}
```

Παρακάτω παρουσιάζονται τα αποτελέσματα του Bag of Words στη πράξη, τα αποτελέσματα του οποίου εκτυπώνονται στη κονσόλα του περιβάλλοντος που πραγματοποιήθηκε η υλοποίηση.

Αρχικά δημιουργείται ένα νέο αντικείμενο της κλάσης BoW η οποία αναπαριστά το μοντέλο Bag of Words. Στη συνέχεια εισάγονται μερικά κείμενα στο μοντέλο. Το θέμα των κειμένων φαίνεται από τον τίτλο του αρχείου, το οποίο είναι τύπου .txt. Τα κείμενα περιέχουν μία σύντομη ανάλυση για το θέμα που περιγράφουν, γύρω στις 1000 - 2000 λέξεις το καθένα. Η απόδοση του μοντέλου βασίζεται στο γενικό λεξιλόγιο, ως εκ τούτου όσο μεγαλύτερο το λεξιλόγιο τόσο καλύτερα και τα αποτελέσματα. Δίπλα στα κείμενα φαίνεται η αύξουσα αρίθμηση, καθώς μέσω αυτής της αρίθμησης θα εκτελούνται οι εντολές.

```
BoW bow = new BoW();

bow.addDoc(new File("src/Area51"));          //0
bow.addDoc(new File("src/Unix"));            //1
bow.addDoc(new File("src/Black_Holes"));     //2
bow.addDoc(new File("src/Modems"));          //3
bow.addDoc(new File("src/Pc_History"));      //4
bow.addDoc(new File("src/UFO"));             //5
```

Έχοντας εισάγει τα αρχεία, ο αλγόριθμος έχει ήδη δημιουργήσει το λεξιλόγιο και παρακάτω παρουσιάζονται μερικά από τα αποτελέσματα.

```
bow.ShowDoc(2);
```
the : 154
of : 77
black : 54
to : 53
is : 46

Τυπώνοντας τις πέντε λέξεις με την μεγαλύτερη εμφάνιση στο πρώτο έγγραφο, είναι εμφανές το πρόβλημα «θορύβου» στα δεδομένα, καθώς οι παραπάνω λέξεις δεν συμβάλλουν στο νόημα του θέματος του κειμένου. Παρόμοια αποτελέσματα παρουσιάζονται και για τα άλλα κείμενα, καθώς και για το γενικό λεξιλόγιο:

```
bow.ShowVocab();
```
the : 1410
to : 608
of : 588
and : 533
in : 378

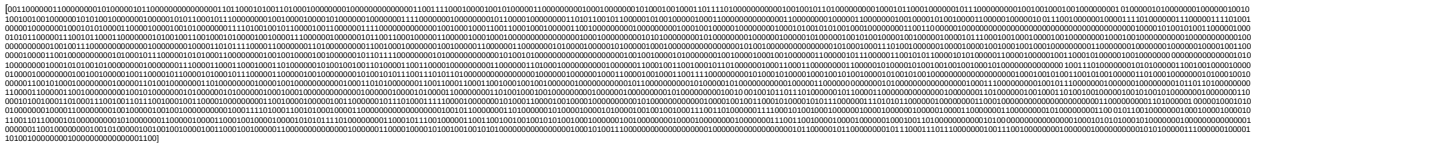
Παρόλα αυτά, με τη χρήση της μεθόδου απαλοιφής του θορύβου που παρουσιάστηκε παραπάνω, τα αποτελέσματα διευκολύνουν κατά πολύ στην απόδοση του νοήματος που κειμένου:

```
bow.TF_IDF();
bow.ShowWeightedDoc(2);
```
star : 78,8374166
black : 59,3250636
hole : 42,8458793
holes : 27,4653072
mass : 25,0846326

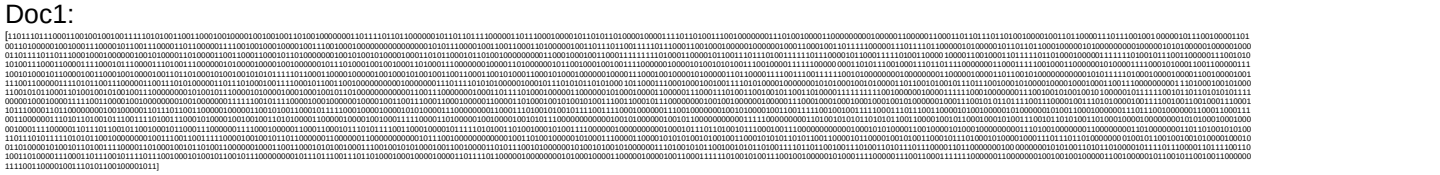
Στη συνέχεια φαίνονται τα παραγόμενα διανύσματα για το κάθε έγγραφο. Φυσικά, δεδομένου ότι το μήκος των διανυσμάτων είναι πολύ μεγάλο, ο μόνος τρόπος να πραγματοποιηθεί η εμφάνιση τους στο κείμενο της πτυχιακής είναι με μικρότερη γραμματοσειρά.

```
bow.Vectorize();
bow.ShowVectors();
```

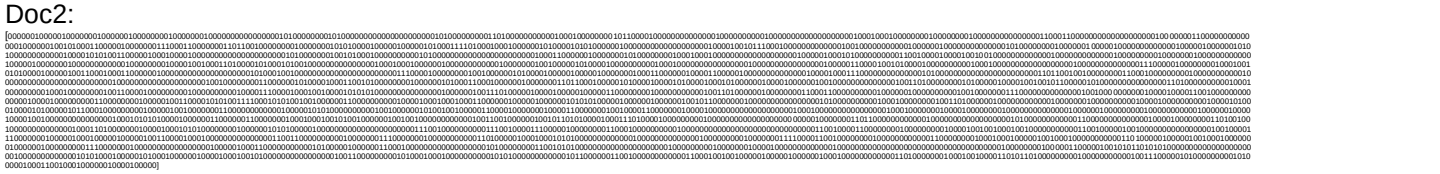
Doc0:



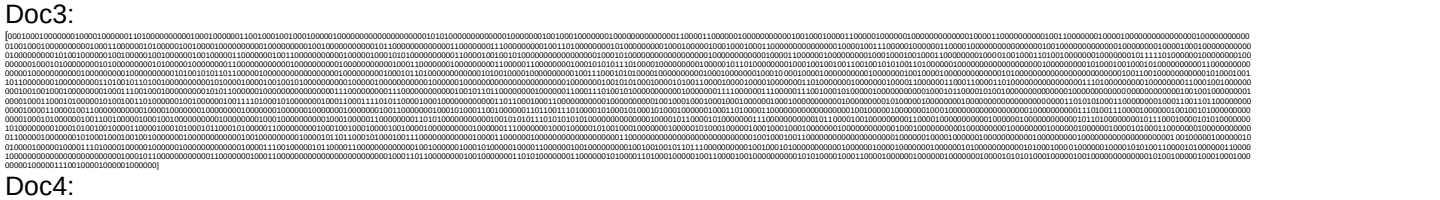
Doc1:



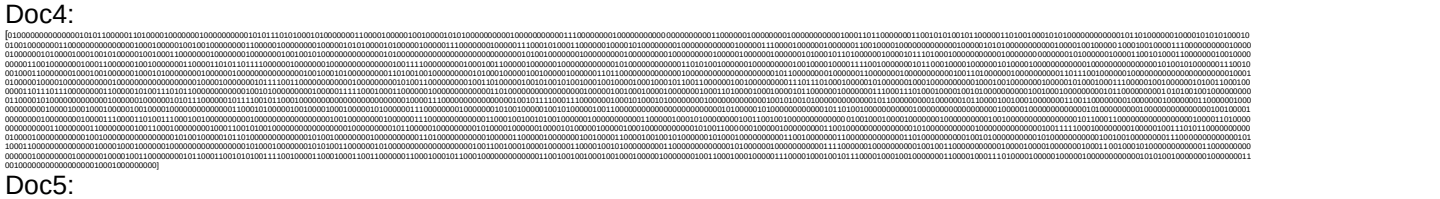
Doc2:



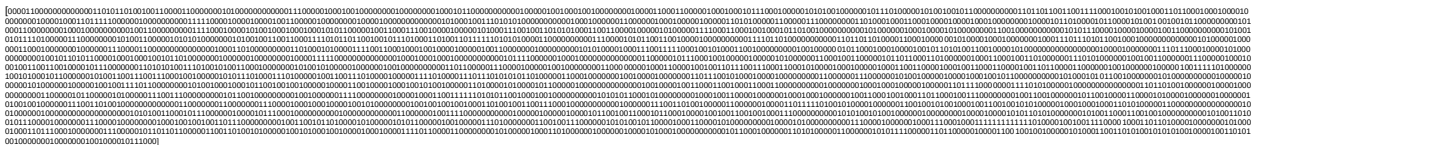
Doc3:



Doc4:



Doc5:



Παρακάτω εμφανίζονται τα αποτελέσματα των τεχνικών σύγκρισης που παρουσιάστηκαν στην υλοποίηση του μοντέλου. Αρχικά, συγκρίνοντας την απόστασης hamming μεταξύ δύο κειμένων:

```
System.out.println(bow.getHammingDistance(1,5));    // Unix VS UFO
```
2060

```
System.out.println(bow.getHammingDistance(0,5));    // Area51 VS UFO
```
1496

Όπως αναφέρθηκε παραπάνω, το αποτέλεσμα της συνάρτησης είναι το σύνολο των διαφορετικών συνιστωσών μεταξύ των δύο διανυσμάτων.

Για τα ίδια κείμενα, με τη μέθοδο της συνημιτονικής ομοιότητας τα αποτελέσματα είναι τα εξής:

```
System.out.println(bow.getCosineSimilarity(1,5));      // Unix VS UFO
0.75706529
System.out.println(bow.getCosineSimilarity(0,5));      // Area51 VS UFO
0.86887131
```

Υπενθυμίζεται ότι όσο πιο κοντά είναι το αποτέλεσμα στο 1, τόσο μεγαλύτερη η ομοιότητα μεταξύ των κειμένων.

Τέλος, μέσω της συνάρτησης **Search** που παρουσιάστηκε παραπάνω, εισάγοντας ως παράμετρο μία πρόταση, ο αλγόριθμος εμφανίζει το κείμενο με την μεγαλύτερη ομοιότητα:

```
String s = "Numerous reports of attempted abductions took place around the United States";
System.out.println(bow.Search(s));
5
UFO

String s = "unix operating systems and their evolution";
System.out.println(bow.Search(s));
1
Unix
```

Να σημειωθεί ότι οι ακολουθίες λέξεων που χρησιμοποιήθηκαν παραπάνω δεν είναι από τα κείμενα που έχουν εισαχθεί στο BoW, αλλά τυχαίες προτάσεις που επιλέχθηκαν για να ταιριάζουν στο αντικείμενο του εκάστοτε εγγράφου.

3.2. Απόδοση νοήματος κειμένου με το Latent Semantic Analysis

Η μέθοδος Latent Semantic Analysis (LSA) έχει ως σκοπό να ανακαλύψει το νόημα του κειμένου. Εάν κάθε λέξη του λεξιλογίου είχε ένα και μοναδικό νόημα, τότε το Bag of Words θα αρκούσε. Δυστυχώς όμως η γλώσσα περιέχει λέξεις με πολλαπλά νοήματα, πολύπλοκους συνδυασμούς λέξεων και ασάφειες στις οποίες ακόμη και οι ίδιοι οι άνθρωποι δυσκολεύονται να αποδώσουν νόημα. Για παράδειγμα η αγγλική λέξη «school» τις περισσότερες φορές αναφέρεται στο ίδρυμα εκπαίδευσης αλλά αν χρησιμοποιηθεί στην πρόταση «school of fish» τότε το νόημα είναι εντελώς διαφορετικό καθώς αναφέρεται σε ένα κοπάδι από ψάρια. Η μέθοδος αυτή είναι μία επέκταση του Bag of Words η οποία προήλθε από την ανάγκη της ανεύρεσης κειμένων με κριτήριο μία ακολουθία λέξεων.

Οι διαδικασία βήμα προς βήμα είναι:

1. Δημιουργία λεξιλογίου και διανυσμάτων των εγγράφων.
2. Προαιρετική χρήση της τεχνικής TFIDF(Term Frequency – Inverse Document Frequency), ανάλογα με τις ανάγκες της αναζήτησης και το μέγεθος των δεδομένων.
3. Χρήση της μαθηματικής τεχνικής SVD (Singular Value Decomposition) η οποία δίνει την δυνατότητα παραγωγής μίας αναπαράστασης του διανυσματικού χώρου, κατά προσέγγιση βαθμίδας **k**.
4. Εύρεση της βέλτιστης βαθμίδας **n < k**, η οποία μπορεί να αναπαραστήσει τον διανυσματικό χώρο, χωρίς να χαθεί σημαντική πληροφορία.
5. Σύγκριση των εγγράφων μετρώντας την διανυσματική ομοιότητα μεταξύ τους, έχοντας πρώτα κανονικοποιήσει τα διανύσματα σε μοναδιαία.

Ξεκινώντας, ορίζονται οι βιβλιοθήκες που θα χρησιμοποιηθούν από το μοντέλο:

```
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.text.DecimalFormat;
import java.text.NumberFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.List;
```

```
import Jama.Matrix;
import Jama.SingularValueDecomposition;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.labels.XYItemLabelGenerator;
import org.jfree.chart.plot.XYPlot;
import org.jfree.chart.renderer.xy.XYItemRenderer;
import org.jfree.data.xy.XYDataset;
import org.jfree.data.xy.XYSeries;
import org.jfree.data.xy.XYSeriesCollection;
import org.jfree.ui.ApplicationFrame;
import org.jfree.ui.RefineryUtilities;
```

Ακολουθεί ο ορισμός της κλάσης, η οποία κληρονομεί την κλάση ApplicationFrame διότι η παρούσα υλοποίηση εμπεριέχει γραφικό περιβάλλον, συγκεκριμένα την δημιουργία γραφήματος. Στη κλάση ορίζεται το ιδιωτικό πεδίο bow, το οποίο είναι ένα αντικείμενο της κλάση BoW, ώστε να είναι εφικτή η κλήση των μεθόδων που την περιβάλλουν. Η δομή Matrix προέρχεται από την βιβλιοθήκη Jama και επιτρέπει την δημιουργία, επεξεργασία και μερικές βασικές πράξεις μεταξύ δισδιάστατων πινάκων. Το πεδίο matrix αποθηκεύει τον διανυσματικό χώρο κατά την δημιουργία των αρχικών διανυσμάτων των εγγράφων. Το πεδίο weights περιέχει τα παραγόμενα διανύσματα μετά την διαδικασία TFIDF. Τέλος, τα WordMatrix και DocMatrix αποθηκεύουν τα διανύσματα των λέξεων και εγγράφων αντίστοιχα μετά την εξαγωγή τους από την τεχνική SDV.

```
public class LSA extends ApplicationFrame {

    private static BoW bow;
    private Matrix matrix, weights;
    private static Matrix WordMatrix, DocMatrix;


    public LSA(BoW bow) {
        super("Chart");
        LSA.bow = bow;
        makeMatrix();
        TD_IDF();
        SVD();
    }
}
```

Η παρακάτω συνάρτηση δημιουργεί τον διανυσματικό χώρο των εγγράφων που έχουν εισαχθεί προς επεξεργασία.

```
public void makeMatrix() {
    List<String> sorted = new ArrayList<>();
    for (String key:bow.GetVocab().keySet()) {
        bow.GetVocab().put(key, bow.GetVocab().get(key));
        sorted.add(key);
    }
    Collections.sort(sorted, String.CASE_INSENSITIVE_ORDER);
    double[][] tempMatrix = new double[bow.GetVocab().size()][bow.getDocs().size()];
    int i = 0;
    for (String word:sorted) {
        for (int j = 0; j < bow.getDocs().size(); j++) {
            tempMatrix[i][j] = (bow.getDoc(j).containsKey(word)) ? (double) bow.getDoc(j).get(word) : 0.0;
        }
        i++;
    }
    matrix = Matrix.constructWithCopy(tempMatrix);
}
```

Ακολουθεί η τεχνική TFIDF. Προτού όμως ο πίνακας των βαρών γίνει διαθέσιμος θα πρέπει να κανονικοποιηθεί. Κάθε στήλη θα διαχωριστεί προσωρινά σε ένα διάνυσμα και μέσω μίας διαδικασίας κανονικοποίησης θα μετατραπεί σε μοναδιαίο διάνυσμα.

```
public void TD_IDF() {
   double[][] tempMatrix = new double[bow.GetVocab().size()][bow.getDocs().size()];
   int[] Number_of_Docs_With_Word = new int[matrix.getRowDimension()];
   for (int j = 0; j < matrix.getRowDimension(); j++) {
      for (int i = 0; i < matrix.getColumnDimension(); i++) {
         Number_of_Docs_With_Word[j] += matrix.get(j, i) > 0 ? 1 : 0;
      }
   }
   for (int i = 0; i < matrix.getColumnDimension(); i++) {
      double Amount_0f_Words_In_Doc = this.DocSum(i);
      for (int j = 0; j < matrix.getRowDimension(); j++) {
         if (matrix.get(j, i) != 0) {
            double Times_of_Word_In_Doc = matrix.get(j, i);
            double weight = (Times_of_Word_In_Doc/Amount_0f_Words_In_Doc)
                        * Math.log10((double) bow.getDocs().size() / Number_of_Docs_With_Word[j]);
            tempMatrix[j][i] = weight;
         }
      }
   }
   weights = Matrix.constructWithCopy(tempMatrix);
   weights = normalize(weights);
}

public Matrix normalize(Matrix m) {

      double[][] norm = new double[m.getRowDimension()][m.getColumnDimension()];
      for (int i = 0; i < m.getColumnDimension(); i++) {
         double[] col = new double[m.getRowDimension()];
         for (int j = 0; j < m.getRowDimension(); j++) {
            col[j] = m.get(j, i);
```

```
            }
            col = normalize(col);
            for (int j = 0; j < m.getRowDimension(); j++) {
                norm[j][i] = col[j];
            }
        }
        return Matrix.constructWithCopy(norm);
}


public double[] normalize(double[] m) {
    double[] normalized = new double[m.length];
    double d = 0.0;
    for (int i = 0; i < m.length; i++) {
        d += Math.pow(m[i], 2);
    }
    double norm = Math.sqrt(d);
    for (int i = 0; i < m.length; i++) {
        normalized[i] = (m[i] / norm);
    }
    return normalized;
}
```

Στη συνέχεια φαίνεται η χρήση της τεχνικής SVD. Η πρώτη συνάρτηση εμφανίζει την ποσοστιαία σημαντικότητα της κάθε singular value. Αυτό γίνεται πιο ξεκάθαρο υψώνοντας κάθε τιμή στο τετράγωνο. Η ποσοστιαία τιμή είναι μία καλή ένδειξη για το ποιες διαστάσεις θα πρέπει να αφαιρεθούν, και να αποφασιστεί η τελική βαθμίδα του διανυσματικού χώρου. Αυτό που επιτυγχάνεται μέσω του SVD είναι να χωριστεί ο αρχικός πίνακας σε 3 επιμέρους πίνακες, οι οποίοι είναι προϊόντα του αρχικού μέσω του τύπου:

$$\boldsymbol{M} = \boldsymbol{S} \times \boldsymbol{U} \times \boldsymbol{V}$$

Ουσιαστικά, αυτοί οι πίνακες είναι μια απλουστευμένη εικόνα του διανυσματικού χώρου. Η τεχνική SVD δίνει έμφαση στις ισχυρότερες σχέσεις μεταξύ λέξεων και εγγράφων και μειώνει τον θόρυβο. Από τους επιμέρους πίνακες, ο **U** περιέχει τις συντεταγμένες κάθε λέξης στον «εννοιακό χώρο» και ο **V** περιέχει τις συντεταγμένες κάθε εγγράφου. Ο πίνακας **S** περιέχει τις τιμές που χρησιμοποιήθηκαν για την απόφαση της σημαντικότητας των διαστάσεων. Οι τιμές του πίνακα S μπορούν να χρησιμοποιηθούν και στην σύγκριση μεταξύ λέξεων και εγγράφων. Μέσω της συνάρτησης inverse() στον πίνακα DocMatrix επιτυγχάνεται μια πιό ξεκάθαρη απεικόνιση.

```
public void PrintSingularValues() {
    double[] sv = weights.svd().getSingularValues();
    System.out.print("[");
    sv = Arrays.stream(sv).map(x -> x *= x).toArray();
    double sum = Arrays.stream(sv).sum();
    for (double value : sv) {
        String s = String.valueOf((int) (value*100/sum));
        System.out.print(value==sv[sv.length-1] ? s + "%]" : s + "%, ");
    }
    System.out.println();
}


public void SVD() {
    SingularValueDecomposition s = weights.svd();
    Matrix S = s.getS(); // Sigma - Dimensional Matrix
    Matrix U = s.getU(); // coords of each word in the concept space
    Matrix V = s.getV(); // coords of each doc in the concept space

    int n = V.getRowDimension() - 2; // n = dimensions to cut, in this case 1

    DocMatrix = V.getMatrix(0, V.getRowDimension()-1, 0, n).inverse();
    WordMatrix = U.getMatrix(0, U.getRowDimension()-1, 0, n);
    S = S.getMatrix(0, n, 0, n);

    DocMatrix = normalize(DocMatrix);
    WordMatrix = normalize(WordMatrix);
}
```

Οι συναρτήσεις στη συνέχεια είναι σημαντικές για την σύγκριση των κειμένων μέσω του υπολογισμού της συνημιτονικής ομοιότητας. Όπως αναφέρθηκε και στην εισαγωγή, τα διανύσματα των κειμένων κανονικοποιούνται πρώτα σε μοναδιαία.

```
public String CompareDocs(int i, int j) {
    double dotProduct = 0.0;
    double[] m1 = new double[DocMatrix.getRowDimension()];
    double[] m2 = new double[DocMatrix.getRowDimension()];
    for (int z = 0; z < DocMatrix.getRowDimension(); z++) {
        m1[z] = DocMatrix.get(z, i);
        m2[z] = DocMatrix.get(z, j);
    }
    m1 = normalize(m1);
    m2 = normalize(m2);
    for (int z = 0; z < DocMatrix.getRowDimension(); z++) {
        dotProduct += m1[z] * m2[z];
    }
    NumberFormat nf = new DecimalFormat("0.########");
    return nf.format(dotProduct);
}
```

Η συνάρτηση Search παρέχει την δυνατότητα αναζήτησης, μέσω μίας ακολουθίας λέξεων, την καλύτερη αντιστοίχιση από τα έγγραφα που έχει επεξεργαστεί το LSA. Η αντιστοίχιση γίνεται μέσω της συνημιτονικής ομοιότητας του search query και των εγγράφων, καθώς το query αντιμετωπίζεται κι αυτό σαν ένα νέο έγγραφο και παίρνει την θέση του στον διανυσματικό χώρο. Επειδή το query αποτελείται από πολύ λιγότερες λέξεις σε σχέση με τα υπόλοιπα έγγραφα, θα πρέπει να μειωθεί η βαθμίδα του διανυσματικού χώρου στο ελάχιστο. Για τον λόγο αυτό καλείται μία δεύτερη έκδοση της μεθόδου SVD() η οποία εξυπηρετεί αυτό τον σκοπό.

```
public void Search(String s) {
        File query = new File("query");
        try {
            BufferedWriter sw = new BufferedWriter(new FileWriter(query));
            sw.append(s);
            sw.close();
        } catch (IOException e) {  }
        bow.addDoc(query);
        this.redo();
        for (int i = 0; i < bow.getDocs().size() - 1; i++) {
            System.out.println("Similarity with " + bow.getDocNames().toArray()[i]
                    + " : " + this.CompareDocs(i, bow.getDocs().size() - 1));
        }
}

public void redo() {
    matrix = null;
    weights = null;
    WordMatrix = null;
    DocMatrix = null;
    makeMatrix();
    TD_IDF();
    SVD( bow.getDocs().size() );
}

public void SVD(int n) {
    SingularValueDecomposition s = weights.svd();
    Matrix S = s.getS(); // Sigma - Dimensional Matrix
    Matrix U = s.getU(); // coords of each word in the concept space
    Matrix V = s.getV(); // coords of each doc in the concept space

    DocMatrix = V.getMatrix(0, V.getRowDimension() - 1, 0, V.getRowDimension() - n + 2).inverse();
    WordMatrix = U.getMatrix(0, U.getRowDimension() - 1, 0, V.getRowDimension() - n + 2);
    S = S.getMatrix(0, V.getRowDimension() - n + 2, 0, V.getRowDimension() - n + 2);

    DocMatrix = normalize(DocMatrix);
    WordMatrix = normalize(WordMatrix);
}
```

Για τον σχεδιασμό του Γραφήματος 1 (σελ 64) χρησιμοποιήθηκε η βιβλιοθήκη JFreeChart. (Vicklund, 2013). Παρακάτω φαίνονται οι μέθοδοι που χρησιμοποιήθηκαν. Να σημειωθεί ότι για την αποφυγή υπερβολικής συμφόρησης στο γράφημα, επιλέχθηκαν για να εμφανίζονται μόνο οι λέξεις με τη μεγαλύτερη βαρύτητα. Στο γράφημα θα απεικονίζονται δύο διαστάσεις, η δεύτερη και η τρίτη. Ο λόγος που δεν επιλέγεται η πρώτη διάσταση είναι γιατί η πληροφορία που προσφέρει δεν είναι σημαντική στην απεικόνιση. Περιέχει δηλάδη το μήκος του κάθε εγγράφου συναρτήσει του αριθμού εμφάνησης της εκάστοτε λέξης σε αυτό, κάτι το οποίο δεν συνεισφέρει στην σύγκριση εγγράφων. Πιό αναλυτικά, η πρώτη διάσταση όσον αφορά τα έγγραφα συσχετίζεται με το μήκος των εγγράφων, δηλαδή από πόσες λέξεις αποτελείται συνολικά, ενώ για τις λέξεις συσχετίζεται με τον συνολικό αριθμό εμφάνισης της κάθε λέξης στο εκάστοτε έγγραφο. Από τις υπόλοιπες διαστάσεις, συνήθως η δεύτερη και η τρίτη διάσταση συμβάλλουν περισσότερο στην απεικόνιση του διανυσματικού χώρου. Στον άξονα Χ τοποθετείται η δεύτερη διάσταση και στον άξονα Υ η Τρίτη διάσταση.

```
public void graph() {
    XYSeries WordSeries = new XYSeries("Words");
    for (int i = 0; i < WordMatrix.getRowDimension(); i++) {
       for (int j = 0; j < DocMatrix.getColumnDimension(); j++) {
          if (Math.abs((weights).get(i, j)) > 0.20) {
             WordSeries.add(WordMatrix.get(i, 1), WordMatrix.get(i, 2));
          }
       }
    }
    XYSeries DocSeries = new XYSeries("Docs");
    for (int i = 0; i < DocMatrix.getColumnDimension(); i++) {
       DocSeries.add(DocMatrix.get(1, i), DocMatrix.get(2, i));
    }
    XYSeriesCollection data = new XYSeriesCollection();
    data.addSeries(WordSeries);
    data.addSeries(DocSeries);
    JFreeChart chart = ChartFactory.createScatterPlot("Data", "2nd Dimension", "3rd Dimension", data);
    XYPlot plot = chart.getXYPlot();
    XYItemRenderer renderer = plot.getRenderer();
    MyGenerator labelgenerator = new MyGenerator();
    renderer.setBaseItemLabelGenerator(labelgenerator);
    renderer.setBaseItemLabelsVisible(true);
    plot.setRenderer(renderer);
    ChartPanel chartPanel = new ChartPanel(chart);
    chartPanel.setPreferredSize(new java.awt.Dimension(600, 600));
    setContentPane(chartPanel);
    this.pack();
    RefineryUtilities.centerFrameOnScreen(this);
```

```
        this.setVisible(true);
}



public static class MyGenerator implements XYItemLabelGenerator  {

        @Override
        public String generateLabel(XYDataset data, int arg1, int arg2) {
            List<String> words = new ArrayList<>(bow.GetVocab().keySet());
            Collections.sort(words, String.CASE_INSENSITIVE_ORDER);
            List<String> docs = new ArrayList<>(bow.getDocNames());
            if (arg1 == 0) {
                for (int i = 0; i < words.size(); i++) {
                    if (data.getXValue(arg1, arg2) == WordMatrix.get(i, 1)
                                && data.getYValue(arg1, arg2) == WordMatrix.get(i, 2)) {
                        return words.get(i);
                    }
                }
            }
            else {
                for (int i = 0; i < docs.size(); i++) {
                    if (data.getXValue(arg1, arg2) == DocMatrix.get(1,i)
                                && data.getYValue(arg1, arg2) == DocMatrix.get(2,i)) {
                        return docs.get(i);
                    }
                }
            }
            return null;
        }
}
```

Τέλος, οι βοηθητικές μέθοδοι που χρησιμοποιήθηκαν από την κλάση:

```
public void printMatrix(Matrix x) {
    if (x == matrix)
        x.print(3, 1);
    else
        x.print(1, 5);
}

public Matrix getMatrix() {
    return matrix;
}

public Matrix getWeights() {
    return weights;
}

public Matrix getWordMatrix() {
        return WordMatrix;
```

```
}

public Matrix getDocMatrix() {
        return DocMatrix;
}

private int DocSum(int col) {
        int sum = 0;
        for (int i=0; i < matrix.getRowDimension(); i++) {
            sum += matrix.get(i, col);
        }
        return sum;
}

public void ShowVocab() {
    NumberFormat nf = new DecimalFormat("0.########");
    List<String> sorted = new ArrayList<>(bow.GetVocab().keySet());
    Collections.sort(sorted, String.CASE_INSENSITIVE_ORDER);
    for (String s:sorted) {
        System.out.println(s + " : " + nf.format(bow.GetVocab().get(s)));
    }
}
```

Στη συνέχεια η λειτουργία του μοντέλου σε ένα παράδειγμα:

Σαν πρώτο βήμα δημιουργείται ένα αντικείμενο της κλάσης BoW και εισάγονται τα κείμενα προς επεξεργασία. Κατά την διαδικασία αυτή, όπως και στο προηγούμενο παράδειγμα, δημιουργείται και το λεξιλόγιο. Το θέμα κάθε κειμένου φαίνεται από τον τίτλο του.

```
BoW bow = new BoW();

bow.addDoc(new File("src/Unix"));              //0
bow.addDoc(new File("src/Unix2"));             //1
bow.addDoc(new File("src/Black_Holes"));       //2
bow.addDoc(new File("src/Modems"));            //3
bow.addDoc(new File("src/Pc_History"));        //4
bow.addDoc(new File("src/UFO"));               //5
```

Μια ματιά στις πέντε πρώτες λέξεις του λεξιλογίου αποδεικνύει την ύπαρξη του θορύβου που δημιουργούν οι λέξεις μικρής σημασίας για τα κείμενα:

```
bow.ShowVocab();
the : 1393
and : 485
to : 476
of : 474
is : 311
```

Έπειτα, δημιουργείται ένα νέο αντικείμενο της κλάσης LSA, με όρισμα το bow που ορίστηκε προηγουμένως.

```
LSA lsa = new LSA(bow);
```

Μέσω της κλήσης της συνάρτησης δημιουργίας ολοκληρώνεται και όλη η διαδικασία της κλάσης. Μέσω των βοηθητικών μεθόδων τυπώνονται οι πίνακες. Οι στήλες αντιστοιχούν στα κείμενα και οι γραμμές στις λέξεις. Οι γραμμές έχουν ταξινομηθεί με αλφαβητική σειρά λέξεων, ενώ οι στήλες με την σειρά που εισήχθησαν. Ακολουθεί ένα δείγμα:

```
System.out.println(bow.GetVocab().size());          // Vocabulary size
3289

lsa.ShowVocab();
abacus : 4
abandon : 1
abbreviation : 2
abbreviations : 2
abc : 1

lsa.printMatrix(lsa.getMatrix());
0.0  0.0  4.0  0.0  2.0  0.0
0.0  8.0  0.0  0.0  0.0  0.0
0.0  0.0  0.0  0.0  0.0  1.0
0.0  0.0  0.0  0.0  1.0  0.0
0.0  0.0  0.0  0.0  1.0  0.0

lsa.printMatrix(lsa.getWeights());
0.00000 0.00000 0.00000 0.00000 0.01160 0.00845
0.00000 0.00000 0.00000 0.02053 0.00000 0.00000
0.00000 0.00850 0.00000 0.01888 0.00000 0.00000
0.00000 0.00000 0.00000 0.02053 0.00000 0.00000
0.00000 0.00000 0.00000 0.02053 0.00000 0.00000
```

Για το συγκεκριμένο παράδειγμα επιλέχθηκε η βαθμίδα του διανυσματικού χώρου να είναι 5. Βλέποντας την σημασία κάθε διάστασης για τον διανυσματικό χώρο, η επιλογή είναι σχεδόν ξεκάθαρη. Φυσικά, σε χώρους μεγαλύτερων βαθμίδων, η επιλογή δεν είναι πάντοτε εύκολη και η εμπειρία και πολλαπλοί έλεγχοι είναι απαραίτητα συστατικά.

```
lsa.PrintSingularValues();
[21%, 17%, 16%, 16%, 16%, 11%]

lsa.printMatrix(lsa.getWordMatrix());
          Dim 1     Dim 2     Dim3     Dim4     Dim5
```

```
Word #1 0.00807 -0.00247  -0.00381  0.00207 -0.00631
Word #2 0.00825  0.00239  -0.00041 -0.00091  0.00041
Word #3 0.00303 -0.01197   0.02129 -0.01680 -0.00782
Word #4 0.00825  0.00239  -0.00041 -0.00091  0.00041
Word #5 0.00614 -0.00315  -0.00292  0.00240 -0.00550
```

```
lsa.printMatrix(lsa.getDocMatrix());
```

	Doc #1	Doc #2	Doc #3	Doc #4	Doc#5	Doc#6
Dim 1	**0.94786**	**0.96221**	**0.05814**	**0.11220**	**0.21450**	**0.14658**
Dim 2	**0.26107**	**0.24964**	**-0.54732**	**-0.39774**	**-0.40592**	**-0.55948**
Dim 3	**-0.13498**	**-0.04230**	**-0.51026**	**0.69119**	**0.39044**	**-0.31642**
Dim 4	**-0.10429**	**-0.09164**	**-0.45898**	**-0.53795**	**0.61015**	**0.34550**
Dim 5	**-0.06566**	**0.04067**	**0.47543**	**-0.24918**	**0.51429**	**-0.66784**

Σύγκριση της συνημιτονικής ομοιότητας μεταξύ των διανυσμάτων των εγγράφων:

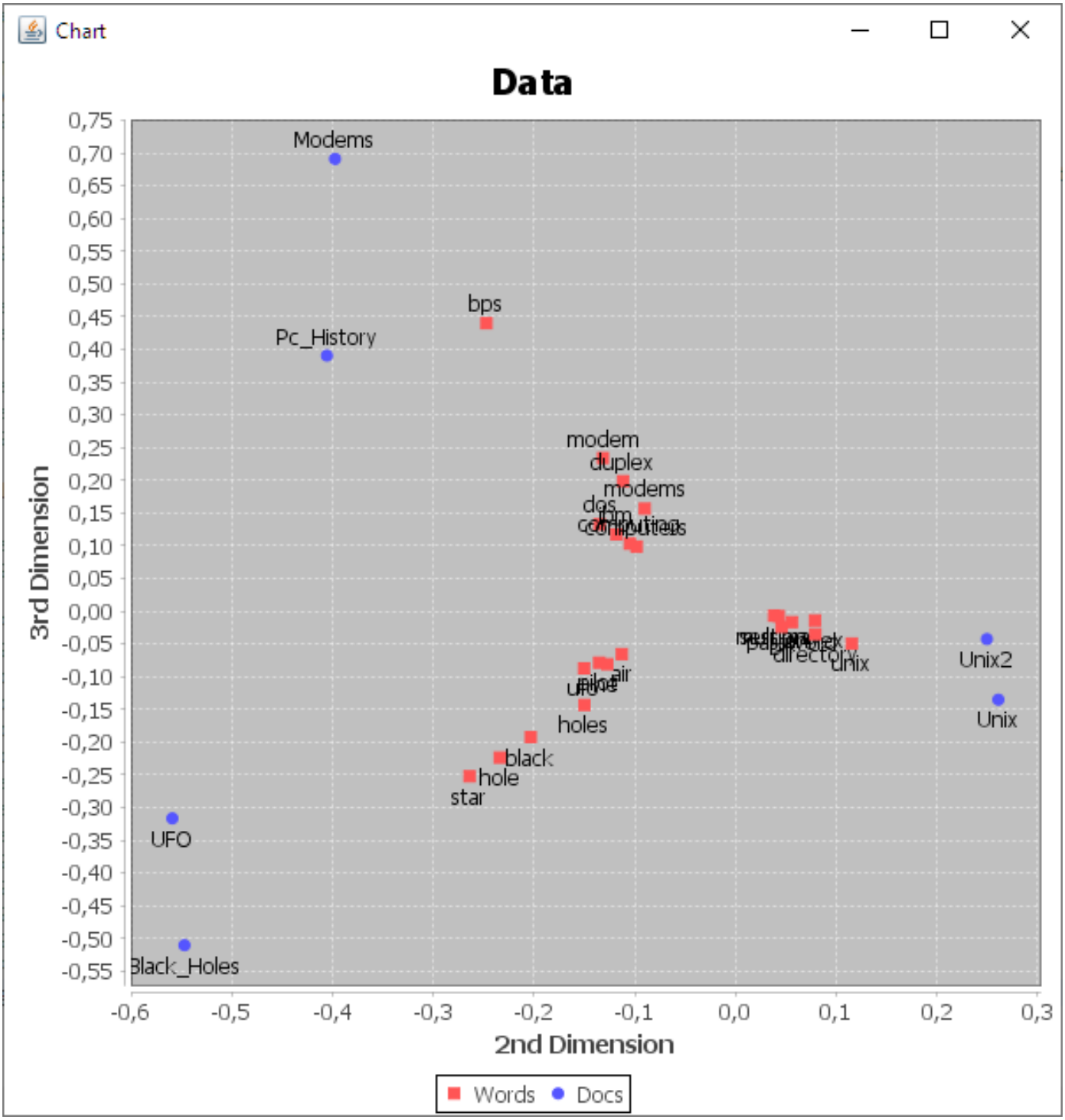
```
System.out.println(lsa.CompareDocs(0,1));        // Unix vs Unix2
0,98980363

System.out.println(lsa.CompareDocs(0,3));        // Unix vs Modems
-0,01832468
```

Με τη χρήση της μεθόδου Graph(), εμφανίζεται το Γράφημα 1.

```
lsa.graph();
```

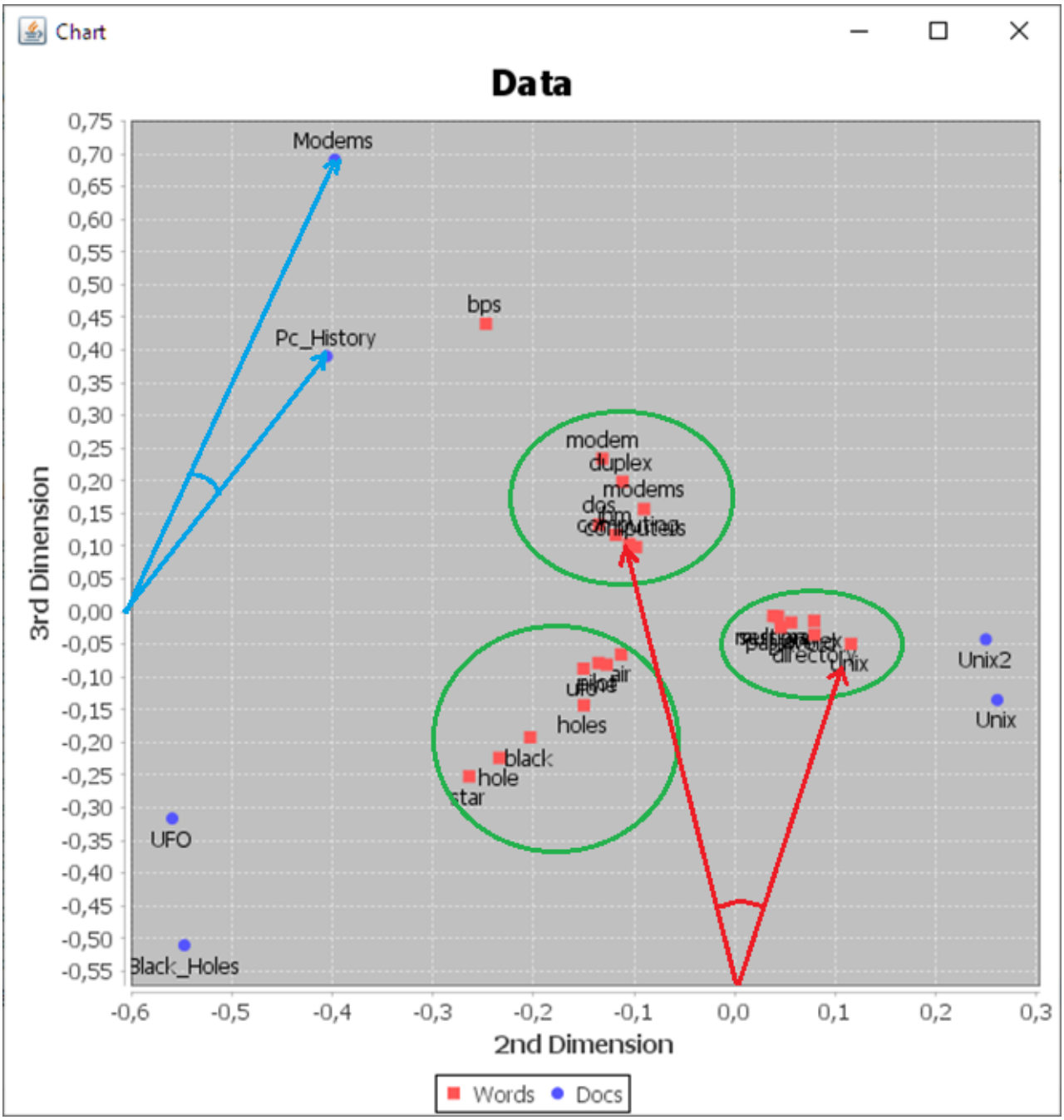
Το Γράφημα 1, παρόλο που δεν αποτελεί πλήρη απεικόνιση του διανυσματικού χώρου, προσφέρει μία γενική ιδέα της μορφής του. Η ανάλυση του γραφήματος όμως μπορεί να προσφέρει περισσότερες πληροφορίες από ότι φαίνεται εκ πρώτης όψεως.



Γράφημα 1

Σρο Γράφημα 2 φαίνεται ότι μέσω ενός αλγόριθμου ομαδοποίησης μπορούν να παρατηρηθούν συστάδες (Priy, 2019), ενώ μέσω των διανυσμάτων των εγγράφων και των λέξεων μπορεί να παρατηρηθεί η γωνιακή τους απόσταση. Φυσικά, στη

συγκεκριμένη περίπτωση σημαντικό ρόλο παίζουν και οι υπόλοιπες 3 διαστάσεις, οπότε αυτό που φαίνεται στο διάγραμμά δεν είναι η πραγματική εικόνα.



Γράφημα 2

Ενώ φαινομενικά τα διανύσματα των κειμένων Pc_History και Modems έχουν μικρή γωνιακή απόσταση, στη πραγματικότητα αν μπορούσε κανείς να δει εξ' ολοκλήρου τον διανυσματικό χώρο θα παρατηρούσε ότι η γωνιακή τους απόσταση διαφέρει αισθητά.

	Modems	Pc_History
Dim 1	**0.11220**	**0.21450**
Dim 2	**-0.39774**	**-0.40592**
Dim 3	**0.69119**	**0.39044**
Dim 4	**-0.53795**	**0.61015**
Dim 5	**-0.24918**	**0.51429**

```
System.out.println(lsa.CompareDocs(3,4));    // Modems vs Pc_History
```
-0,00099127

Αυτό αποδεικνύει πόση πληροφορία θα μπορούσε να χαθεί εάν δεν είχε επιλεχθεί η σωστή βαθμίδα για τον διανυσματικό χώρο.

Ολοκληρώνοντας το παράδειγμα, παρακάτω φαίνονται τα αποτελέσματα τριών search queries:

```
lsa.Search("regions that consume stars and have huge mass");
```
Similarity with Unix : 0,02383748
Similarity with Unix2 : 0,01255621
Similarity with Black_Holes : 0,7400015
Similarity with Modems : -0,40717364
Similarity with Pc_History : -0,22712772
Similarity with UFO : -0,0444373

```
lsa.Search("what are modems responsible for?");
```
Similarity with Unix : -0,03531217
Similarity with Unix2 : -0,06764202
Similarity with Black_Holes : -0,23353466
Similarity with Modems : 0,87140008
Similarity with Pc_History : -0,04092231
Similarity with UFO : -0,20042264

```
lsa.Search("The advent of personal computers and the concurrent Digital Revolution");
```
Similarity with Unix : -0,13234506
Similarity with Unix2 : -0,07730808
Similarity with Black_Holes : -0,30664271
Similarity with Modems : 0,10582019
Similarity with Pc_History : 0,8266139
Similarity with UFO : -0,17168566

Συμπεράσματα

Δεν υπάρχει αμφιβολία ότι οι τεχνικές ανάκτησης και επεξεργασίας πληροφοριών βασίζονται στην θεωρία δοκιμής και σφάλματος (trial and error). Η καλύτερη λύση σπάνια είναι εμφανής και η επιλογή της χρειάζεται υπομονή και επιμονή. Κυρίως όμως χρειάζεται εμπειρία.

Ειδικότερα όσον αφορά στα μοντέλα αναπαράστασης κειμένων, η καλύτερη μέθοδος σπάνια είναι η ίδια με τη προηγούμενη. Κάθε περίπτωση απαιτεί διαφορετικούς χειρισμούς και οι φορές που ο χρήστης λαμβάνει ικανοποιητικά αποτελέσματα είναι πολύ περιορισμένες. Στον κλάδο αυτό της πληροφορικής δεν αναζητείται το βέλτιστο, επειδή υπάρχει η αποδοχή ότι η ανθρώπινη γλώσσα βασίζεται στην ανθρώπινη λογική και την ασάφεια που την περιβάλλει. Έτσι πρέπει να λειτουργεί και ένα σωστό λογισμικό που έχει σαν στόχο την εξαγωγή αποτελεσμάτων και συμπερασμάτων από ανθρώπινα κείμενα. Αυτή είναι και η μαγεία της μηχανικής μάθησης, το ότι λαμβάνει υπόψιν την ασάφεια και την ατέλεια.

Τα δύο μοντέλα που υλοποιήθηκαν βασίζονται σε αυτή την αρχή, και τα αποτελέσματα το αποδεικνύουν. Κατά την σύγκριση των εγγράφων η απάντηση ποτέ δεν είναι απόλυτη. Εν αντιθέσει, είναι η πιο κοντινή προσέγγιση σε σχέση με την διαθέσιμη πληροφορία στην οποία έχουν πρόσβαση τα μοντέλα. Εξάλλου, τα αποτελέσματα δεν έχουν σαν σκοπό να δώσουν μία απόλυτη απάντηση, αλλά να καθοδηγήσουν τον χρήστη να επιλέξει από ένα εύρος απαντήσεων ή να αποκλείσει ένα εύρος πιθανών ανεπιθύμητων αποτελεσμάτων. Έχουν αναπτυχθεί ώστε να λαμβάνουν υπόψην τους και το περιθώριο σφάλματος αφού, όπως και τον άνθρωπο, τους δίνεται η δυνατότητα να αναπτύσσονται και να εκπαιδεύονται.

Συγκεκριμένα όταν η πληροφορία αφορά αρχεία κειμένου, η επεξεργασία της έχει μεγαλύτερο βαθμό δυσκολίας. Αυτό οφείλεται στο γεγονός ότι τα γραπτά κείμενα είναι προϊόν ανθρώπινο, και ο τρόπος γραφής του εξαρτάται από τις εμπειρίες, την παιδεία και την διαφορετικότητα του καθενός. Για τον λόγο αυτό πολλές φορές παρατηρείται ομοιότητα μεταξύ δύο κειμένων διαφορετικού περιεχομένου αλλά του ίδιου συγγραφέα. Στο Bag of Words (BoW) αυτό αποτελεί μεγαλύτερο πρόβλημα αφού οι συγγραφείς έχουν συγκεκριμένο τρόπο γραφής και συχνά επιλέγουν λέξεις από ένα περιορισμένο λεξιλόγιο. Λόγω αυτού του γεγονότος προέκυψε και η ανάγκη της εξέλιξης των μοντέλων

επεξεργασίας κειμένων σε πιο «έξυπνα». Το μοντέλο Latent Semantic Analysis (LSA) έχει την δυνατότητα να διακρίνει τέτοια μοτίβα και να μην τα λαμβάνει υπόψιν κατά την ανεύρεση του νοήματος του κειμένου.

Ένα εξίσου πολύπλοκο πρόβλημα, αλλά και το πιο σημαντικό είναι η ανάλυση των αποτελεσμάτων. Η συνολική αναπαράσταση των αποτελεσμάτων σε γραφήματα, ή άλλου είδους τρόπους εικονικής αναπαράστασης, δεν είναι σχεδόν ποτέ εφικτή μιας και επεκτείνονται σε πολλές διαστάσεις. Όπως παρατηρείται στο παράδειγμα για το LSA, η αναπαράσταση στις δύο διαστάσεις δεν είναι συγκεντρωτική, αλλά βοηθητική στην περαιτέρω ανάλυση.

Υπάρχουν μοντέλα που χρησιμοποιούν αυτοματοποιημένες τεχνικές ανάλυσης των αποτελεσμάτων ή τα αντιμετωπίζουν σαν νέα δεδομένα προς επεξεργασίας. Ένα παράδειγμα τέτοιων μοντέλων είναι το Word2Vec το οποίο, ανάλογα με τη διάσταση του προβλήματος, χρησιμοποιεί τα αποτελέσματα που εξάγονται από τους νευρώνες ως δεδομένα προς επεξεργασία από άλλους νευρώνες. Ακόμη, υπάρχουν τεχνικές συσχέτισης των αποτελεσμάτων όπως η ομαδοποίηση σε συστάδες η οποία αναφέρεται και στο μοντέλο του LSA. Εν κατακλείδι, οι επεκτάσεις της επιστήμης της Ανάκτησης της Πληροφορίας και γενικότερα της Μηχανικής Μάθησης έχουν μεγάλο εύρος και έχουν φτάσει στο σημείο να συμμετέχουν στην λήψη σημαντικών αποφάσεων, πολλές φορές καθοριστικών ακόμα και για την ανθρώπινη ζωή.

Για περαιτέρω έρευνα προτείνεται η μελέτη και η λεπτομερής ανάλυση τεχνολογιών μηχανικής μάθησης και ανάκτισης πληροφορίας που έχουν εφαρμογή σε διάφορους τομείς όπως ασφάλεια, υγεία, παιδεία.

Βιβλιογραφία

Alham Dr. M., Alrifai Dr. B., Khatatneh Dr. Kh., Wedyan M., 2005, Using new data - structure to implement documents vectors in vector space model in information retrieval system. Theoretical and Applied Information Technology, Vol 19. No. 1:4

Baeza-Yates R. and Ribeiro –Neto B., 1999, Modern Information Retrieval, ACM Press

Bank J. and Cole B., 2008, Calculating the Jaccard Similarity Coefficient with Map Reduce for Entity Pairs in Wikipedia, Wikipedia Similarity Team

Bello I., Kaul A., Britton J. R., 2014, Topics in contemporary mathematics, (10η έκδοση). Belmont, pp. 534

Berry M. W., Drmac Zl. and Jessup El. R., 1999, Matrices, vector-spaces, and information retrieval, Society for Industrial and Applied Mathematics, Volume 41:28

Brownlee J., 2017, A Gentle Introduction to the Bag-of-Words Model, [Πρόσβαση στις 22 Αυγούστου 2019], Διαθέσιμο από: https://machinelearningmastery.com/gentle-introduction-bag-words-model/

Czibula G., Cojocar S. Gr., 2010, A Hierarchical Clustering Based Approach in Aspect Mining, Computing and Informatics, Department of Computer Science Babe¸s-Bolyai University Vol. 29, pp. 881–900

Dice L. R., 1945, Measures of the amount of ecologic association between species, Ecology, pp 297–300, [Πρόσβαση στις 25 Ιουλίου 2018], Διαθέσιμο από: https://www.stata.com/manuals/mvmeasure_option.pdf

Dong H., Husain F.K. and Chang E., 2008, A Survey in Traditional Information Retrieval Models, IEEE International Conference on Digital Ecosystems and technologies, pp. 397-402

Ghosh J., 2017, Intelligent Data Exploration and Analysis Lab, [Πρόσβαση στις 17 Ιουλίου 2018], Διαθέσιμο απο: http://www.lans.ece.utexas.edu/ strehl/diss/node56.html

Hicklin J., Moler C., 2012, JAMA: A Java Matrix Package, [Πρόσβαση στις 25 Αυγούστου 2019], Διαθέσιμο από: https://math.nist.gov/javanumerics/jama/

Jones K. Sp., 1972, A statistical interpretation of term specificity and its application in retrieval, Journal of Documentation

Kim A., 2019, What is Poisson Distribution, [Πρόσβαση στις 23 Αυγούστου 2019], Διαθέσιμο από: https://towardsdatascience.com/poisson-distribution-intuition-and-derivation-1059aeab90d

Lee D., Lam L., 2002, Feature reduction for neural network-based text categorization. In 6th International Conference on Advanced Systems for Advanced Applications

Luhn H. P., 1957, A statistical approach to mechanized encoding and searching of literary information, Volume 1 Issue 4, pp 309-317, IBM Journal of Research and Development

Maron, M.E., Kuhns J.L., 1960, On relevance, probabilistic indexing, and information retrieval. Journal of the Association for Computing Machinery, 7: 216-244.

McCormick C., 2016, Word2Vec Tutorial, The Skip-Gram Model

Mikolov T., Chen K., Corrado Gr., Dean J., 2014, Efficient Estimation of Word Representations in Vector Space

Minerazzi, 2014, Data mining platform, [Πρόσβαση στις 28 Ιουλίου 2018], Διαθέσιμο από: http://www.miislita.com/information-retrieval-tutorial/cosine- similaritytutorial.html

PriyS., 2019, Clustering in Machine Learning, [Πρόσβαση στις 12 Αυγούστου 2019], Διαθέσιμο από: https://www.geeksforgeeks.org/clustering-in-machine-learning/

Raman S., Kumar V., and Venkatesan S. 2012, Performance Comparison of Various Information Retrieval Models Used in Search Engines, IEEE conference on communication, information and Computing Technology, Mumbai, India

Robertson S.E. Harding P., 1984, Probabilistic automatic indexing by learning from human indexers, Journal of Documentation, 40(4): 264-270

Robertson S.E., Jones K. Sp., 1976, Relevance Weighting of Search Terms, Journal of the American Society for Information Science, Volume 27 number 3, pp 129 – 146

Salton G. and Buckley C., 1988, Term-weighting approaches in automatic text retrieval, Information Processing & Management

Salton G. and McGill M.J., 1983, Introduction to Modern Information Retrieval, McGraw-Hill Book Co.

Salton G., Wong A., and Yang C.S., 1975, A Vector Space Model for Automatic Indexing, Communications of the ACM, vol. 18, nr. 11, pages 613– 620

Sarwar B. M., 2001, Cosine-based Similarity, [Πρόσβαση στις 23 Ιουλίου 2018], Διαθέσιμο από: http://www10.org/cdrom/papers/519/node12.html

Tazzyman S., DaSH, MoJ, 2019, Natural Language Processing guidance, [Πρόσβαση στις 20 Αυγούστου 2019], Διαθέσιμο από: https://moj-analytical-services.github.io/NLP-guidance/

Teufe S., 2014, Lecture 4: Term Weighting and the Vector Space Model, Information Retrieval Computer Science Tripos Part II, University of Cambridge

Textfiles, 2019, The Textfile Directory, [Πρόσβαση στις 30 Αυγούστου 2019], Διαθέσιμο από:http://textfiles.com/directory.html?fbclid=IwAR2TweXLwgn5ByOcT0R3jmkHiaBD_SEnXmbf2ONPJ9gbZw5q7hjUy8kbtPw

Viklund A., 2013, JFreeChart, [Πρόσβαση στις 20 Αυγούστου 2019], Διαθέσιμο από: http://www.jfree.org/jfreechart/download.html

Wesley W. Ch., Zhenyu L., and Wenlei M., 2002, Textual document indexing and retrieval via knowledge sources and data mining, In Communication of the Institute of Information and Computing Machinery (CIICM)

Wiley J. and Sons, 2009, Information Retrieval: Searching in the 21st Century, Wiley

Wolfram, 1999, MathWorld, [Πρόσβαση στις 20 Ιουλίου 2018], Διαθέσιμο από: http://mathworld.wolfram.com/innerproduct.html

Wu H., Luk R., Wong K. and Kwok K., 2008, Interpreting TF-IDF term weights as making relevance decisions, ACM Transactions on Information Systems

Παράρτημα

Κείμενο 1: Black_Holes

There is much more to black holes than meets the eye. In fact, your eyes, even with the aid of the most advanced telescope, will never see a black hole in space. The reason is that the matter within a black hole is so dense and has so great a gravitational pull that it prevents even light from escaping. Like other electromagnetic radiation (radio waves, infrared rays, ultraviolet radiation, X-rays, and gamma radiation), light is the fastest traveler in the Universe. It moves at nearly 300,000 kilometers (about 186,000 miles) per second. At such a speed, you could circle the Earth seven times between heartbeats. If light can't escape a black hole, it follows that nothing else can. Consequently, there is no direct way to detect a black hole. In fact, the principal evidence of the existence of black holes comes not from observation but from solutions to complex equations based on Einstein's Theory of General Relativity. Among other things, the calculations indicate that black holes may occur in a variety of sizes and be more abundant than most of us realize.

MINI BLACK HOLES

Some black holes are theorized to be nearly as old as the Big Bang, which is hypothesized to have started our Universe 10 to 20 billion years ago. The rapid early expansion of some parts of the dense hot matter in this nascent Universe is said to have so compressed less rapidly moving parts that the latter became superdense and collapsed further, forming black holes. Among the holes so created may be the submicroscopic mini-black holes. A mini-black hole may be as small as an atomic particle but contain as much mass (material) as Mount Everest. Never underestimate the power of a mini-black hole. If some event caused it to decompress, it would be as if millions of hydrogen bombs were simultaneously detonated.

HOW STARS DIE

The most widespread support is given to the theory that a black hole is the natural end product of a giant star's death. According to this theory, a star like our Sun and others we see in the sky lives as long as thermal energy and radiation from nuclear reactions in its

core provide sufficient outward pressure to counteract the inward pressure of gravity caused by the star's own great mass. When the star exhausts its nuclear fuels, it succumbs to the forces of its own gravity and literally collapses inward. According to equations derived from quantum mechanics and Einstein's Theory of General Relativity, the star's remaining mass determines whether it becomes a white dwarf, a neutron star, or black hole.

WHITE DWARFS

Stars are usually measured in comparison with our Sun's mass. A star whose remaining mass is about that of our Sun condenses to approximately the size of Earth. The star's contraction is halted by the collective resistance of electrons pressed against each other and their atomic nuclei. Matter in this collapsed star is so tightly packed that a piece the size of a sugar cube would weigh thousands of kilograms. Gravitational contraction would also have made the star white hot. It is appropriately called a white dwarf. Astronomers have detected white dwarfs in space. The first discovery was a planet-sized object that seemed to exert a disproportionately high gravitational effect upon a celestial companion, the so call dog star Sirius, which is about 2.28 times our Sun's mass. It appeared that this planet-sized object would have to be about as massive as our Sun to affect Sirius as it did. Moreover, spectral analysis indicated the star's color was white. Based upon these and other studies, astronomers concluded that they had found a white dwarf. However, it took many years after the discovery in 1914 before most scientists accepted the fact that an object thousands of times denser than anything possible on Earth could exist.

NEUTRON STARS AND SUPERNOVAS

Giant stars usually lose most of their mass during their normal lifetimes. If such a star still retains 1 1/2 to 3 solar masses after exhaustion of its nuclear fuels, it would collapse to even greater density and smaller size than the white dwarf. The reason is that there is a limit on the amount of compression electrons can resist in the presence of atomic nuclei. In this instance, the limit is breached. Electrons are literally driven into atomic nuclei, mating with protons to form neutrons and thus transmuting nuclei into neutrons. The resulting object is aptly called a neutron star. It may be only a few kilometers in diameter.

A sugar-cube size piece of this star would weigh about one-half a trillion kilograms. Sometimes, as electrons are driven into protons in atomic nuclei, neutrinos are blown outward so forcefully that they blast off the star's outer layer. This creates a supernova that may temporarily outshine all of the other stars in a galaxy. The most prominent object believed to be a neutron star is the Crab Nebula, the remnant of a supernova observed and reported by Chinese astronomers in 1504. A star-like object in the nebula blinks, or pulses, about 30 times per second in visible light, radio waves, and X and gamma rays. The radio pulses are believed to result from interaction between a point on the spinning star and the star's magnetic field. As the star rotates, this point is theorized alternately to face and be turned away from Earth. The fast rotation rate implied by the interval between pulses indicates the star is no more than a few kilometers in diameter because if it were larger, it would be torn apart by centrifugal force.

PULSARS

Radio telescopes have detected a large number of other objects which send out naturally pulsed radio signals. They were named pulsars. Like the object in the Crab Nebula, they are presumed to be rotating neutron stars. Of these pulsars, only the Vela pulsar--which gets its name because of its location in the Vela (Sails) constellation--pulses at wavelengths shorter than radio. Like the Crab pulsar, the Vela pulsar also pulses at optical and gamma ray wavelengths. However, unlike the Crab pulsar, it is not an X-ray pulsar. Aside from the mystery generated by these differences, scientists also debate the reasons for the pulses at gamma, X-ray and optical frequencies. As noted earlier, they agree on the origin of the radio pulses.

BLACK HOLES

When a star has three or more solar masses left after it exhausts its nuclear fuels, it can become a black hole. Like the white dwarf and neutron star, this star's density and gravity increase with contraction. Consequently, the star's gravitational escape velocity (speed needed to escape from the star) increases. When the star has shrunk to the Schwarzschild radius, named for the man who first calculated it, its gravitational escape velocity would be nearly 300,000 kilometers per second, which is equal to the speed of

light. Consequently, light could never leave the star. Reduction of a giant star to the Schwarzschild radius represents an incredible compression of mass and decrease in size. As an example, mathematicians calculate that for a star of 10 solar masses (ten times the mass of our Sun) after exhaustion of its nuclear fuels, the Schwarzschild radius is about 30 kilometers.

According to the Law of General Relativity, space and time are warped, or curved, by gravity. Time is theorized TO POINT INTO THE BLACK HOLE FROM ALL DIRECTIONS. To leave a black hole, an object, even light would have to go backward in time. Thus, anything falling into a black hole would disappear from our Universe. The Schwarzschild radius becomes the black hole's "event horizon", the hole's boundary of no return. Anything crossing the event horizon can never leave the black hole. Within the event horizon, the star continues to contract until it reaches a space-time singularity, which modern science cannot easily define. It may be considered a state of infinite density in which matter loses all of its familiar properties. Theoretically, it may take less than a second for a star to collapse into black hole. However, because of relativistic effects, we could never see such an event. This is because, as demonstrated by comparison of clocks on spacecraft with clocks on Earth, gravity can slow, perhaps even stop, time. The gravity of the collapsing star would slow time so much that we would see the star collapsing for as long as we watched. Once a black hole has been formed, it crushes into a singularity anything crossing its event horizon. As the black hole devours matter, its event horizon expands. This expansion is limited only by the availability of matter. Incredibly vast black holes that harbor the crushed remains of billions of solar masses are theoretically possible. Evidence that such superdense stars as white dwarfs and neutron stars do exist has supported the idea that black holes, representing what may be the ultimate in density, must also exist. Potential black holes, stars with three or more times the mass of our Sun, pepper the sky. But how can astronomers detect a black hole?

HOW BLACK HOLES MAY BE INDIRECTLY DETECTED

Scientists found indirect ways of doing so. The methods depends upon black holes being members of binary star systems. A binary star system consists of two stars comparatively near to and revolving about each other. Unlike our Sun, most stars exist in pairs. If one

of the stars in a binary system had become a black hole, the hole would betray its existence, although invisible, by its gravitational effects upon the other star. These effects would be in accordance with Newton's Law: attractions of two bodies to each other are directly proportional to the square of the distance between them. The reason is that outside of its event horizon, a black hole's gravity is the same as other objects'. Scientists also have determined that a substantial part of the energy of matter spiraling into a black hole is converted by collision, compression, and heating into X- and gamma rays displaying certain spectral characteristics. The radiation is from the material as it is pulled across the hole's event horizon, its radiation cannot escape.

WORMHOLES

Some scientists speculate that matter going into a black hole may survive. Under special circumstances, it might be conducted via passages called "wormholes" to emerge in another time or another universe. Black holes are theorized to play relativistic tricks with space and time.

NASA ORBITING OBSERVATORY OBSERVATIONS

Black hole candidates--phenomena exhibiting black hole effects--have been discovered and studied through such NASA satellites as the Small Astronomy Satellites (SAS) and the much larger Orbiting Astronomical Observatories (OAO) and High Energy Astronomical Observatories (HEAO). The most likely candidate is Cygnus X-1, an invisible object in the constellation Cygnus, the swan. Cygnus X-1 means that it is the first X-ray source discovered in Cygnus. X-rays from the invisible object have characteristics like those predicted from material as it falls toward a black hole. The material is apparently being pulled from the hole's binary companion, a large star of about 30 solar masses. Based upon the black hole's gravitational effects on the visible star, the hole's mass is estimated to be about six times of our Sun. In time the gargantuan visible star could also collapse into a neutron star or black hole or be pulled piece by piece into the existing black hole, significantly enlarging the hole's event horizon.

BLACK HOLES AND GALAXIES

It is theorized that rotating black holes, containing the remains of millions or billions of dead stars, may lie at the centers of galaxies such as our Milky Way and that vast rotating black holes may be the powerhouses of quasars and active galaxies. Quasars are believed to be galaxies in an early violent evolutionary stage while active galaxies are marked by their extraordinary outputs of energy, mostly from their cores. According to one part of the General Theory of Relativity called the Penrose Process, most of the matter falling toward black holes is consumed while the remainder is flung outward with more energy than the original total falling in. The energy is imparted by the hole's incredibly fast spin. Quiet normal galaxies like our Milky Way are said to be that way only because the black holes at their centers have no material upon which to feed. This situation could be changed by a chance break-up of a star cluster near the hole, sending stars careening into the hole. Such an event could cause the nucleus of our galaxy to explode with activity, generating large volumes of lethal gamma radiation that would fan out across our galaxy like a death ray, destroying life on Earth and wherever else it may have occurred.

BLACK HOLES AND GALACTIC CLUSTERS

Some astronomers believe that the gravity pulls of gigantic black holes may hold together vast galactic clusters such as the Virgo cluster consisting of about 2500 galaxies. Such clusters were formed after the Big Bang some 10 to 20 billion years ago. Why they did not spread randomly as the Universe expanded is not understood, as only a fraction of the mass needed to keep them together is observable. NASA's Hubble Space Telescope and AXAF Telescope, scheduled for a future Shuttle launch, will provide many more times the data than present ground and space observatories furnish and should contribute to resolving this and other mysteries of our Universe.

BLACK HOLES AND OUR UNIVERSE

Our universe is theorized to have begun with a bang that sent pieces of it outward in all directions. As yet, astronomers have not detected enough mass to reverse this expansion. The possibility remains, however, that the missing mass may be locked up in

undetectable black holes that are more prevalent than anyone realizes. If enough black holes exist to reverse the universe's expansion, what then? Will all of the stars, and galaxies, and other matter in the universe collapse inward like a star that has exhausted its nuclear fuels? Will one large black hole be created, within which the universe will shrink to the ultimate singularity? Extrapolating backward more than 10 billion years, some cosmologists trace our present universe to a singularity. Is a singularity both the beginning and end of our universe? Is our universe but a phase between singularities? These questions may be more academic than we realize. Scientists say that, if the universe itself is closed and nothing can escape from it, we may already be in a black hole.

Κείμενο 2: Modems

Not long ago, many data communicators thought that dial-up modem manufacturers had pushed transmission speeds to the limit with the introduction of 2400 bit per second (bps) modems. Recently, however, several manufacturers have creatively combined relatively mature techniques of data transmission with newer technology and have introduced 9600 bps modems. Unfortunately, a widely accepted standard for full duplex 9600 bps transmission as defined by the International Consultative Committee for Telegraphy and Telephony (CCITT) does not yet exist (the CCITT is currently considering proposals for a new 9600 bps dial-up standard). This means that today's 9600 bps modems do not offer cross-manufacturer compatibility. The CCITT HAS endorsed a half duplex and a full duplex 9600 bps standard, but to date implementations of these relatively flexible standards have been proprietary, i.e., even the "standardized" modems from different manufacturers are not compatible. All this means that modem users who want to enjoy the dream speed of 9600 bps must weigh the pros and cons of each 9600 bps technique before committing to a particular 9600 bps design. This paper was written in an effort to provide typical modem users with enough technical information and insight that they will be able to consider the new 9600 bps modems from the position of an educated consumer and not have to rely on information gleaned from sales brochures and advertisements. It should be noted that the author, Wes Cowell, is an employee of USRobotics.

THE ROAD TO 9600

High speed data communications via the dial-up phone network is limited by the available phone line bandwidth and by random channel impairments. Just as the diameter of a pipe limits its liquid flow capacity, so does the telephone channel bandwidth limit its data flow capacity. The roughly 3000-Hz available in the telephone bandwidth poses few problems for 300 bps modems, which only use about one fifth of the bandwidth. A full duplex 1200 bps modem requires about half the available bandwidth, transmitting simultaneously in both directions at 600 baud and using phase modulation to signal two data bits per baud. "Baud rate" is actually a measure of signals per second. Because each signal can represent more than one bit, the baud rate and bps rate of a modem are not necessarilly the same. In the case of 1200 bps modems, their baud rate is actually 600 (signals per second) and each signal represents two data bits. By multiplying signals per second with the number of bits represented by each signal one determines the bps rate: 600 signals per second X 2 bits per signal = 1200 bps. In moving up to 2400 bps, modem designers decided not to use more bandwidth, but to increase speed through a new signalling scheme known as quadrature amplitude modulation (QAM). In QAM, each signal represents four data bits. Both 1200 bps and 2400 bps modems use the same 600 baud rate, but each 1200 bps signal carries two data bits, while each 2400 bps signal carries four data bits: 600 signals per second X 4 bits per signal = 2400 bps. A technique known as adaptive equalization enables 2400 bps modems to adapt to phone line impairments call-by-call. Essentially, if the modem is experiencing problems with a noisy line, it looks for a "sweet spot" in the bandwidth and attempts to avoid troublesome frequencies. This technique makes 2400 bps modems more tolerant of line noise than their 1200 bps counterparts that use compromise equalization (a one-size-fits-all approach). While these advanced modulation and equalization techniques in 2400 bps modems provide for double the data rate of 1200 bps modems, they also result in a design at least four times more complex than 1200 bps modems. Which brings us to the problem of designing a 9600 bps modem. Jumping to 9600 from 2400 bps is several orders of magnitude more complicated than going to 2400 from 1200 bps. Telephone network characteristics make it highly unlikely that success will be had in extending the "data signal alphabet" (number

of bits represented by each signal) beyond four bits per signal. Instead, modem designers must increase the bandwidth that is to carry the signal, and this presents a very big problem. In fact, at speeds of 4800 bps (1200 signals per second), the transmit and receive channels must be expanded to the point where they actually begin to overlap. A 9600 bps "band" requires roughly 90 percent of the available bandwidth, making it impossible to have two-way communication without the bands interfering with each other. A helpful analogy to the problem might be to consider a two lane highway: traffic must flow in both directions simultaneously, but to carry more cars per unit of time, highway designers must either increase the number of lanes in each direction or widen the two lanes to accommodate driver error with a margin of safety. Unfortunately, these options are not available to modem designers as the available bandwidth is of a fixed size. With these considerations and limitations in mind, let's examine three basic ways to accomplish full duplex (two-way) 9600 bps communications: echo cancellation, virtual full duplex (achieved by half duplex systems), and asymmetrical frequency division.

ECHO-CANCELLATION

This method solves the problem of overlapping transmit and receive channels. Each modem's receiver must try to filter out the echo of its own transmitter and concentrate on the other modem's transmit signal. This presents a tremendous computational problem that significantly increases the complexity -- and cost -- of the modem. But it offers what other schemes don't: simultaneous two-way transmission of data at 9600 bps. The CCITT "V.32" recommendation for 9600 bps modems includes echo- cancellation. The transmit and receive bands overlap almost completely, each occupying 90 percent of the available bandwidth. Measured by computations per second and bits of resolution, a V.32 modem is roughly 64 times more complex than a 2400 bps modem. This translates directly into added development and production costs which means that it will be some time before V.32 modems can compete in the high- volume modem market. Despite the fact that V.32 is a recognized standard, it is uneconomical and unnecessarily complex for personal computer datacomm applications that simply don't require simultaneous two-way 9600 bps transmission.

HALF DUPLEX SYSTEMS (Virtual Full Duplex)

Half duplex solutions devote the entire bandwidth to 9600 bps in one direction at a time, and "ping-pong" the data flow back and forth to simulate full duplex. This is potentially the simplest scheme. Its performance is acceptable in data transfer applications that don't involve user interaction, i.e. file transfers. Even so, advanced error-control protocols that require ACKnowledgments to be sent in response to received data blocks generate a high number of "line reversals" which greatly impair overall data throughput. In short, the benefit of higher speed is so significantly compromised by line reversals in half duplex sessions that the net gain in data throughput may be marginal at best. If users want to operate in an interactive mode, their data must be sent to the remote computer, the data channel must be reversed, and then the data must be echoed back. This process results in significant turn-around delays which can be very frustrating to users. Half duplex modems of this kind are most often based on CCITT recommendation V.29 for half duplex 9600 bps transmission on the dial-up network. V.29 based data pumps used in facsimile systems are available as LSI chip sets, providing a short-cut to modem manufacturers, particularly to companies that don't develop their own modem technologies. But the major problem is that the V.29 modulation scheme has been outdated by the fact that it operates in a half duplex mode and doesn't provide good signal to noise performance. The V.32 recommendation, which operates in a full duplex mode and employs Trellis Coding Modulation offers greater throughput and a greater immunity to channel impairments. To the best of my knowledge, modems employing V.29-based modulation include products from Racal-Vadic, Comspec, Develcon, Gamma Technology, Microcomm, and Electronic Vaults, Inc. (EVI). These modems, however, are NOT mutually signal compatible -- cross-manufacturer compatibility does not exist. Another modem in the half duplex category, but not based on V.29 modulation, is the Telebit Trailblazer (R), which uses a proprietary modulation method. Trailblazer is based on a multi-carrier technique. Conceptually, the transmission channel is divided into many (512), independent, very narrow channels (think of our two-lane highway and imagine it as having 512 very narrow lanes (say, for bicycles) going in one direction and you've got a fair idea of how Trailblazer divides the bandwidth). The main advantage is that no receiver adaptive equalizer is needed because each channel is very narrow compared to the overall channel bandwidth. Further, in the

Trailblazer modulation scheme, the modulation rate in each narrow channel can be changed somewhat independently. Trailblazer is different from many other modems in that the decision to fall back to lower speeds is built into the modem protocol, rather than controlled by the user's computer port. It is claimed that in the face of channel impairments, throughput can be adapted gracefully to channel conditions. Traditional modulation systems would have to fall back in larger steps. But there are three inherent MAJOR problems: 1) The turn-around delay is very long compared to conventional modulation techniques because data must be sent in large blocks. A typed character may take several seconds to be echoed back to the system that sent it. As a result, the system fails to achieve the illusion of full duplex and is not really suited to interactive online sessions. 2) The Trailblazer receiver cannot "track" carrier "phase jitter" (phase jitter can be thought of in terms of "phase shift": think of how the whine of a race car goes from higher to lower as it passes the viewer -- the frequency of the sound is said to be "shifted" or "jittered"). Instead of cancelling out phase jitter (which is commonly encountered on long distance calls) the Trailblazer can only respond by lowering throughput to gain more immunity to phase jitter. 3) The ability to transmit at the maximum rate when subject to channel impairment is considerably less than for conventional modems. There is one notable exception: the multiple channel technique offers extremely good immunity to impulse noise because the impulse energy is distributed over narrow channels. While conventional modems can achieve similar results through special coding or filtering techniques they rarely implement such methods.

ASYMMETRICAL FREQUENCY DIVISION

When one considers the nature of most PC datacomm applications, it is realized that most applications are interactive, involving manual (typed) data entry from one end and data file transmission from the other end. Few, if any, PC users can justify using an expensive 9600 bps channel to carry their typed characters when they realize that 300 bps translates to 360 words per minute. Assuming one could type 100 words per minute, even a 100 bps transmission channel would be sufficient. On the other hand, file transfer should take advantage of the tremendous speed of the microprocessor. Serial ports are often set at data rates in excess of 19,000 bps. Considering these inherent characteristics, a

communications scheme that incorporated a high speed and a low speed channel would be best suited for most PC datacomm applications. Remembering the highway analogy (higher speeds mean wider lanes), one can see how such a method would grant modem designers a large portion of the available bandwidth for a 9600 bps channel and still leave enough room to accommodate a narrow 300 bps channel without any channel overlap. By utilizing two discreet channels, such a modem would avoid costly, complex echo-cancellation schemes. And, because the channels carry data in both directions simultaneously, the communications link is a true full duplex connection. This means that data entered at one system would be almost instantaneously echoed back -- eliminating the frustrating turn-around delay experienced in half duplex sessions. USRobotics has developed just such a modem. It passes data in one direction using the V.32 modulation technique (a very robust method that is very immune to phone line impairments) but employs only a 300 bps channel in the opposite direction so that the channels do not overlap and echo-cancellation is not necessary. The use of the high-speed channel by the two modems is based on data demand. In most applications, however, "channel swapping" will not be required. For interface elegance, the modems employ a 4K buffer that allow them to perform data rate conversion: sending and receiving speeds remain constant between the modem and the computer -- it is only in between the modems that transmitted and received data run at different speeds. For interactive sessions, users are assigned the low-speed channel while the data sent to them (long mail messages, menus, files, etc.) in the 9600 bps channel. For file transfer sessions, the data blocks that make up a file are sent in the 9600 bps channel while the corresponding ACKnowledgments are returned in the 300 bps channel. An asymmetric frequency division scheme is ideal for file transfer where large data blocks (usually several hundred bytes in length) are transmitted in the high-speed channel and the ACKs (usually only a few bytes in length) are carried in the low-speed channel. If a user switches from an interactive mode to file transfer and then back to interactive mode, the high speed channel is dynamically and automatically assigned to the system with the greatest data demand.

A BRIEF COMPARISON

Three options exist for data communicators who desire to operate at 9600 bps: 1) V.32-type modems offer a full duplex connection but do so by virtue of echo-cancellation. This technique is so complex, and has proven so difficult to employ, that the cost for such modems will remain prohibitively high and their implementation a delicate task for some time to come. 2) Half duplex modems (either V.29 or multi-carrier) offer 9600 bps but the turn-around delay inherent in half duplex links severely compromise overall throughput. This degradation of throughput, however, can be more than offset by data compression techniques assuming the modems in question support identical compression protocols and are operating on relatively "clean" phone lines. Both half duplex methods suffer disproportionate degradation on "noisy" lines: the V.29 modems must spend more and more time in line reversals as detected data errors increase, and the multi-carrier modems must sacrifice throughput to gain noise immunity. 3) Asymmetrical Frequency Division offers 9600 bps communications in a true full duplex implementation. By efficiently utilizing the available bandwidth, these modems provide users with high speed file transfer capabilities and fast response in interactive sessions. Because the transmit and receive data channels do not overlap, expensive echo-cancelling techniques are unnecessary making these modems economically efficient.

IN CONCLUSION

Until a widely recognized standard is agreed upon by the standards community, and implemented by several manufacturers, modem buyers must weigh the benefits and detriments of each 9600 bps scheme. V.32 would be best where symmetrical, full duplex, synchronous communication is desired (for example, dial-up HDLC links between multiplexers) and where the user can modify his software to accommodate non-"AT" command-driven modems. V.29 modems would be likely solutions where absolute lowest price is required and conformance to an international standard (in a very limited sense) is desired. Multi-carrier transmission schemes are well-suited to applications that require maximum one-way throughput and where circuit conditions are known to be good. This transmission method is also ideally suited for circuits where immunity to impulse noise is paramount. Users who most often work with one-way file transfers (PC-to-PC) or with

real- time applications may opt for an Asymmetrical Frequency Division scheme, which is suited equally well for either application. The elegant approach to the frequency division (avoiding overlapping bandwidths) also allows these modems to present a very economical ratio between dollars and bps. Potential high-speed-modem buyers should also consider the aspects of ease-of- use, ease-of-implementation, and downward compatibility with existing implemented standards (the CCITT's V.22bis for 2400 bps, Bell 212A for 1200 bps, and Bell 103 for 200 bps).

POST SCRIPT

Many modem users have voiced confusion and consternation about the lack of compatibility between modem manufacturers at speeds greater than 2400 bps. Modem manufacturers have embraced the Bell 212A and 103 standards for 1200 and 300 bps. In these post-divestiture days, however, Bell no longer sets modem standards in the U.S. and hence, U.S. modem manufacturers have turned to the CCITT as a definitive source for standards. The industry-wide acceptance of the CCITT's V.22bis standard for 2400 bps is the best example of this shift. The CCITT recommendations V.29 and V.32 for 9600 bps have not resulted in compatible implementations. It is important to remember that V.29 was originally developed as a four-wire full duplex leased-line modem and has since been adapted by various manufacturers to encompass half duplex dial up applications. Other problems with V.29 are that it compromises transmission speed and is poor for interactive sessions. V.32 is proving to be prohibitively complex and exceptionally difficult to implement (driving development and production costs up). Recognizing the need for an alternative to the V.32 recommendation, the CCITT has requested proposals from modem manufacturers. Presently, two proposals are being considered by the CCITT. One is the multi- carrier scheme developed and sponsored by Telebit. The other is an Asymmetrical Frequency Division scheme developed and sponsored by USRobotics.

Κείμενο 3: Pc_History

Computing or calculating by machine began in the middle east with the use of pegs or stones in trays or channels. The Babylonians developed the idea of stone or bead counters into the more modern abacus - modern in the sense that the abacus is still in

use today and in the hands of an experienced operator can calculate results faster than a computer! The beauty of the abacus is its simplicity in construction and operation. Inexpensive beads of stone or wood and a simple frame make up the abacus and the uneducated could quickly be trained in its use. In the 8th and 9th centuries we note the rise of the Arabic numeral system which slowly spread through Europe and the then civilized world. Although a superior calculating system, Arabic numerals required the user to understand the more complicated numerical theory associated with the system. By the early 1600's Napier (often associated with the development of logarithms and their practical application) introduced a series of rods which could be used for multiplication - a crude slide rule system. Soon, ever more complicated "calculating engines" or primitive mechanical computing devices appeared. One example is the complex Pascaline invented by Blaise Pascal. By 1791 the stage was set. Babbage, an English mathematician and inventor with the help of Ada Byron (daughter of lord Byron, the famous poet) developed the ideas for two mechanical calculators or "number engines." The Difference Engine was a device to solve polynomial equations by the methods of differences. The Analytical Engine (which was never built)) was designed as a general computing device. Both were mechanical in concept using gears, rods and cams to perform calculations. Unfortunately neither machine was built since the tooling and machining technology of the day was imprecise and could not construct the accurate parts needed. However the models and planning of Babbage and Byron did lead to important preliminary computing concepts still in use today. As an aside, we should note from the work of Babbage and Byron that computing even in its infancy was strongly influenced by BOTH women and men - let's face it, computing is NOT gender specific! Next we jump to the United States. By 1880 a problem had arisen with the United States census. By that time, it took 7 years to process all of the information gathered by the Census Bureau since all tabulation was done by hand on paper. It was assumed that the 1890 census might take 10 to 12 years to tabulate. Clearly a better method was needed to crunch the volume of numbers and data. A public competition was held to produce a better indexing or mechanical system to tabulate future census results. Herman Hollerith, a census employee, handily won by suggesting the use of punch cards and a form of punch card reader which tabulated the results in six weeks. Hollerith, wise in the ways of computing

devices and seeing a good opportunity went on to found the Tabulating Machine Company (later changed to IBM). Hollerith might be thus thought of as our first computer entrepreneur! The advent of World War II provided the impetus for the development of more refined computing devices. The Mark I was an electromechanical device using relays. IBM built that computer for the Navy. Later, the Colossus was built for the British and used for wartime code breaking of German radio transmissions. The ABC (Atanasoff-Berry Computer) was constructed at Iowa State and was the first fully electronic digital computer. Admiral Grace Hopper, known as "Amazing Grace" to some, was a naval officer and pioneer in the field of computer programming during the 1940's and 50's. An innovative and fundamental thinker, she recognized that computers could be used for business applications - a pioneering insight beyond the then conventional use of computers for scientific and military applications. Her programming language called "Flowmatic" later evolved into COBOL, the most common and still popular language for programming business software. She died in 1992 and is buried in Arlington National Cemetery. Computing science continued to evolve rapidly . . . Eniac was the most famous of the early computers and contained 18,000 vacuum tubes and was used by the Army for ballistics calculations. Edvac was the first stored memory computing device which did away with rewiring tasks associated with changing computer programs and represented a true computer breakthrough. This first generation of machines running from roughly 1951 through 1958 featured computers characterized by the use of radio type vacuum tubes. But the pace was increasing . . . Second generation machines such as the famous Univac were designed as true general or universal purpose machines and could process both alphabetic and numeric problems and data. Punch cards still formed the major input path to the machines of this era and all programming was done in complex low level machine language commands. By 1959 with the invention of the transistor, computers began to shrink in size and cost and operate faster and more dependably than the huge vacuum tube models. Programming languages began to feature English-like instructions rather than cumbersome machine code or assembly language. Fortran and Cobol are two modern "high level" languages developed during this period and still in use today. In many respects, the personal computer industry began in 1974 when the Intel corporation introduced a CPU integrated circuit chip named the 8080. It contained 4,500 transistors

and could address 64K of memory through a 16 bit data bus. The 8080 was the integrated circuit brain behind the early MITS Altair personal computer which fired popular interest in home and small business computing when it appeared on the July 1975 cover of Popular Electronics Magazine. The first MITS Altair contained no keyboard or monitor, only crude LED lights and tiny flip switches to facilitate programming. Four years later in 1978 Intel released the 8086 chip which had a tenfold increase in performance over the 8080 chip. When IBM began the design phase of the first desktop PC units in 1980 and 1981, they chose the cousin of the 8086, the Intel 8088 chip, to power the first PC which was designed for modest corporate use but quickly exploded in popularity due to an excellent design, spectacular keyboard and openess to upgrade by the addition of "plug in" boards and cards. Early IBM PC computers retained a link with the past by allowing the addition of a small "Baby Blue" circuit board which could run software programs based on the then dominant CPM operating system. Finally we come to the present decade . . . August 1981. Original IBM PC (personal computer) introduced. Has options for monochrome and CGA color display. Receives generally good reviews and acceptance by business users and a few home users. Original DOS version 1.0 released which supported only single sided disks (160K capacity). Later version 1.1 corrected bugs (problems) in the DOS programming code and provided double sided disks (320K capacity), and faster disk access, date and time stamping and better serial communications. August 1982. Monochrome resolution of PC screen increased with introduction of the Hercules graphics card circuit. Combined with the LOTUS 123 spreadsheet, the IBM PC was now a hot choice for corporate computing. November 1982. Compaq portable arrives. First IBM clone on the market. The IBM PC standard is growing in popularity. Clone makers start to copy the PC in earnest. Software companies such as Phoenix technologies prepare BIOS and SYS programs which run the same as the IBM BIOS program without the copyright violation which every clone computer tries to avoid. BIOS stands for basic input and output system and is the core software essential to keyboard, disk and screen input/output. The BIOS is considered legally protected IBM software code, but can be simulated (or emulated) closely by a clever programmer in an attempt to do the same job, without using exactly the same programming code. March 1983. IBM introduces the PC XT (increased memory and hard drive capability). DOS

version 2.0 released. This second DOS version includes hard drive capability, filter commands (sort, find, more), and a new floppy format system for 360K capacity per floppy. IBM bios code upgraded. October 1983. IBM PC JR released. Market disappointment for that IBM entry into the home market with the underpowered PC JR. The larger IBM PC standard is rapidly growing as the standard for personal computers and clones. March 1984. IBM PC portable introduced. Portable clones already on the market with small but growing success. August 1984. IBM PC AT machine arrives. More power, a new processor (Intel 80286). New screen display standard (EGA). Also new version of DOS 3.0. This version of DOS now takes into account the AT high density floppy drive (1.2 meg or million characters of capacity), read only files and a new disk write system for better file recovery in case of errors. Shortly thereafter, DOS 3.1 addresses file sharing. November 1985. Microsoft windows graphic display environment released. NEC multisync monitor is released. December 1985. DOS 3.2 supports the new 3.5 inch 720K diskettes. DOS now addresses up to 32MB on a single hard disk. April 1986. Older IBM PC standard model discontinued for newer models. IBM PC convertible model is released. September 1986. Compaq jumps the gun on IBM with release of new (80386) processor computer with more power than the PC AT. April 1987. IBM PS/2 models 30, 50 and 60 released. DOS 3.3 released. VGA video standard arrives. IBM blesses the new 3.5 inch minifloppy already in use on Apple Macintosh computers by offering that format on IBM machines. 1.44MB format is supported for high density floppy users. OS/2 operating system announced. August 1987. Microsoft windows version 2.0 arrives. November 1988. DOS 4.01 released which includes a shell menu interface system. This release of DOS, largely developed by IBM, generally ignored due to poor performance and large memory requirements. Many users stick with DOS 3.3. 1988 Laptop computers, smaller versions of desktop computers, are sold in large volumes. Size as well as features become issues in computer sales. 1990 Microsoft introduces Windows version 3.0 which includes a superb graphical user interface (GUI) display for the PC. Improves on earlier versions of Windows. Using software is more productive with multiple graphical software windows and the possibility of jumping between several software tasks operating on screen. But windows can only run acceptably on more expensive "high end" machines such as those containing the 386DX or 386SX chip. For

many users in small offices or home offices, Windows may not be a necessity where simple DOS applications offer affordable functionality on low priced PC's not equipped to run Windows applications. 1991 Laptop computers, portable FAX systems, and cellular portable phone technology allow computers to function anywhere on the go for a practical "portable office" concept. Still newer "palmtop" computers about the size of portable calculators now offer full IBM compatable functionality. June 1991. DOS 5.0 is released which includes excellent new features including an improved menu interface, full-screen editor which improves on the Edlin editor, some limited task- swapping abilities, unformat/unerase utility, improved Basic interpreter, and ability to load system files to High Memory on machines having at least 1MB for improved performance and increased conventional memory availability for primary applications. DOS 5.0 is seen to be a major and highly necessary update to the PC operating system. Generally receives good reviews from computer trade press. The future? Difficult to predict, but the consensus of industry observers is that the IBM PS/2 computers will migrate into the office scene while many home and home/office users will stay with older XT computers and AT models. Best entry level computer system at this time is judged by many experts to the a 386SX PC system which allows many types of software both current and future to work reliably. Prices continue to tumble on AT and XT compatibles ($400 to $500 range) and AT clones ($700 to $900 range). The operating system for AT and higher class machines (using 80286/80386 processors) is called OS/2 but requires more memory and the 80286/80386 processor found only in higher priced computers. OS/2 or Windows may slowly replace the older DOS system, but for many users of home and home/office machines not needing LAN networks (many computers talk to each other and share data), the DOS standard will live a long time. The Microsoft Windows 3.0 system may delay the acceptance of OS/2 for several years. In general expect things to happen faster, computers to become still smaller and prices to descend still further! Graphical user interfaces or GUI's will gradually become the standard so that users can point and click at small icon pictures and lists of tasks on screen to accomplish the work at hand rather than fight with terse and cryptic commands. Computing will become a standard in many small and home offices owing to the incredible power, accuracy and affordablity of personal computers. Laptop computers and even smaller palmtop computers will become new standards.

Computers and modems linked by wireless cellular radio/telephone technology allow a single computer user the power of "large office computing" on the go from anywhere in the world! Shareware software will make strong inroads into the market as users evaluate commercial "high priced" software against user support "low cost" shareware software. Tutorial finished. Be sure to order your FOUR BONUS DISKS which expand this software package with vital tools, updates and additional tutorial material for laptop users! Send $20.00 to Seattle Scientific Photography, Department LAP, PO Box 1506, Mercer Island, WA 98040. Bonus disks shipped promptly! Some portions of this software package use sections from the larger PC-Learn tutorial system which you will also receive with your order. Modifications, custom program versions, site and LAN licenses of this package for business or corporate use are possible, contact the author. This software is shareware - an honor system which means TRY BEFORE YOU BUY. Press escape key to return to menu.

Κείμενο 4: UFO

In the late afternoon of November 17, 1986, Japan Air Lines flight 1628, a Boeing 747 with a crew of three, was nearing the end of a trip from Iceland to Anchorage, Alaska. The jet, carrying a cargo of French wine, was flying at 35,000 feet through darkening skies, a red glow from the setting sun lighting one horizon and a full moon rising above the other. A little after six p.m., pilot Kenju Terauchi noticed white and yellow lights ahead, below, and to the left of his airplane. He could see no details in the darkness and assumed the lights were those of military aircraft. But they continued to pace the 747, prompting first officer Takanori Tamefuji to radio Anchorage air traffic control and ask if there were other aircraft nearby. Both Anchorage and a nearby military radar station announced that they were picking up weak signals from the 747's vicinity. Terauchi switched on the digital color cockpit weather radar, which is designed to detect weather systems, not other aircraft. His radar screen displayed a green target, a color usually associated with light rain, not the red he would have expected from a reflective solid object. Because he was sitting in the left-hand seat, Terauchi had the only unob- structed view when the lights, still in front of and below the airplane, began moving erratically,"like two bear cubs playing with each other," as the pilot later wrote in a statement for the Federal Aviation Administration. After

several minutes, the lights suddenly darted in front of the 747,"shooting off lights" that lit the cockpit with a warm glow. As the airplane passed over Eielson Air Force Base, near Fairbanks, the captain said he noticed, looming behind his airplane, the dark silhoutte of a gigantic "mothership" larger than two aircraft carriers. He asked air traffic control for permission to take his airplane around in a complete circle and then descend to 31,000 feet. Terauchi said his shadower followed him through both maneuvers. A United Airlines fight and a military C-130 were both in the area and An- chorage asked the airplanes to change course, intercept the Japanese 747, and confirm the sighting. Both airplanes flew close enough to see JAL 1628's navigation lights, alone in the night sky, before Terauchi reported that the unidentified fyling objects had disappeared. The encounter had lasted nearly 50 minutes. Because it involved an airline pilot and an unidentified flying object that had apparently been captured on radar, the JAL 1628 encounter attracted a great deal of public attention. But UFO reports from pilots--private,military and airline--are not new to the subject of "ufology." One of the best known cases was a sighting by Idaho businessman and private pilot Kenneth Arnold. Flying his single-engine airplane over Washington's Cascade Mountains on June 24,1947, Arnold spotted nine silvery, crescent-shaped objects skimming along at high speed near Mt. Rainier. They dipped as they flew,"like a saucer would if you skipped it across water," Arnold told reporters--and thus "flying saucers" entered the popular vocabulary. Pilots had reported similar unexplained aerial phenomena before, mainly in the form of the "Foo Fighters" noted by American bomber crews over Europe in World War II. But Arnold's sighting, with its accompanying front-page publicity, struck a jittery, post-Hiroshima nerve in American society and set off a barrage of similar reports. Skeptics believed that every sighting had a prosaic explanation, such as misidentification of stars, planets, or natural atmospheric phenomena. Others thought that there was more to UFOs, that they could even be visitors from other planets. Following the Arnold incident, the Air Force was given the responsibility of investigating UFO reports from the United States, first as Project Sign (also called Saucer), then Grudge, and finally Blue Book. Usually understaffed and underfunded, the Air Force program functioned more like a public relations office than a scientific investigation, according to the late astronomer J. Allen Hynek. Hynek himself, who served as a consultant to Project Blue Book from 1948 unitl it was dissolved in

December 1969, gradually changed from a skeptic into a believer. Not even skeptics can deny the subject's popular appeal. Last March, a Gallup poll found that 88 percent of its respondents had heard of UFOs. Nearly half of those polled believed UFOs were real, not figments of the imagination or misperceived natural phenomena. Nine percent of the adult population claimed to have seen one. Of these claims, pilot reports are the ones that interest Richard F. Haines, a perceptual psychologist who compiles AIRCAT, a computerized catalog that lists more than 3,000 UFO sightings by aviators over the past 40 years. Chief of the Space Human Factors Office at NASA's Ames Research Center in California Haines is the author of "Observing UFOs", a handbook of methodology for accurate observation, and the editor of "UFO Phenomena and the Behavioral Scientist", a collection of psychologically oriented essays on the subject.

SKEPTICS R US

The Committee for the Scientific Investigation of Claims of the Paranormal (CSICOP) was founded in the spring of 1976, during a meeting of the American Humanist Association in Buffalo, New York. The impetus for the group's form- ation had been provided a year earlier by the publication of "Objections to Astrology" by Paul Kurtz, professor of philosophy at the State University of New York at Buffalo. The manifesto had been signed by 186 scientists, in- cluding 18 Nobel prizewinners, who feared that the public was confusing astronomy and astrology. Today Kurtz is chairman of the loosely knit international organization, which holds annual meetings and publishes a 25,000-circulation quarterly, "The Skeptical Inquirer." The journal is devoted to articles debunking psychokinesis telepathy,clairvoyance, and other psychic claims, the Loch Ness Monster, astro- logy and UFOs. CSICOP Fellows include science writer Isaac Asimov, astronomer Carl Sagan, Nobel physicist Murray Gell-Mann, and James Randi, recent recipient of a "genius grant" awarded by the MacArthur Foundation. The UFO subcommittee is led by Philip J. Klass ("UFOs--Identified","UFOs Ex- plained",and "UFOs, the Public Decieved"), James Oberg ("UFOs & Outer Space Mysteries"),and Robert Sheaffer ("The UFO Verdict"). The subcommittee con- sists of about two dozen members who operate as an informal network, exchang- ing articles about UFOs for information and comment. Some members make them- selves available for local media appearances to counteract what

Klass calls "the popular view of UFOs as extraterrestrial spaceships." "We prefer to have skeptics, of course," says Klass, "but we don't require anyone to take an oath of allegiance saying they don't believe in flying saucers. Basically, we're a mutual education circuit" (Dennis Stacy).

AIRCAT's cases include Blue Book's declassified files as well as some Haines collected and research personally. Before joining the Space Human Factors Office, his research included interviewing pilots about what they had seen peripherally during takeoffs and landings, data that may one day lead to re- design of airplane cockpits. "I was interviewing pilot anyway," he says, "and fell naturally into the habit of asking them if they'd ever seen anything strange." Haines concentrated on pilot reports for reasons other than convenience. "They have a unique vantage point simply by being in the air," he says, "if for no other reason than if the phenomenon is between your eyes and the ground, you can calculate the slant range, and you're establishing an absolute maximum distance the object could be away. You can't do that with the object against the sky background." "Pilots also have available to them a variety of electromagnetic sensors of various kinds onboard the aircraft itself, which can possibly record some manifestations of the phenomenon, such as electromagnetic frequency and even energy content," he says. "They can control the location of their plane so that they can maneuver to gain the best vantage point, under some conditions. "Finally," says Haines, "they represent a very stable personality type with a high degree of training, motivation, and selection. If a pilot comes forward with a strange tale, I give him a lot of careful concentration because he's putting his reputation on the line and maybe his job. He's had to have thought the details out in his mind already, and perhaps eliminated a number of ex- planations before going public." He's also likely to request anonymity. Kenneth Arnold, tired of the publicity following his sighting, later commented, "If I ever see again a phenomenon of that sort, even if it's a 10-story building, I won't say a word about it." The feeling was echoed even in the Air Force. When Blue Book's predecessor, Project Grudge, conducted an informal survey of Air Force pilots in the late 1940s , one respondent said, "If a spaceship was flying wing-tip to wing-tip formation with me, I would not report it." The UFO phenomenon got its tabloid reputation at least in part because of the saucer-busting of active UFO skeptics. Foremost is the UFO panel of CSICOP, the Committee for the Scientific

Investigation of Claims of the Paranormal (see "Skeptics R Us," previous page). Led by Philip J. Klass, contributing avionics editor of "Aviation Week and Space Technology", James Oberg, an aerospace writer and a manned space operations specialist, and Robert Sheaffer, a Silicon Valley computer systems analyst, CSICOP exposes hoaxes and uncovers explanations of UFO sightings. Sheaffer doesn't agree that pilots are superior UFO observers. "The idea of pilots as super witnesses just doesn't hold," he says. "The last I heard they were human like the rest of us, and still subject to all concerns and errors of human psychology and perception. In fact, they're apt to be less worried about how bright an object is, or its angular elevation, than in keeping their plane in the air. Anyone surprised by a very brief and unexpected event is not likely to report it accurately." Haines agrees that normal perception isn't infallible. Very bright objects, for example, can appear to be much nearer than they actually are. Autokinetic or self-generated, movement of the eyeball can make distant objects like stars and planets appear to move. "Also when you're flying in a sunny, clear blue atmosphere," Haines says,"sometimes the eye can focus inaccurately, so that you're not focusing at infinity anymore, but maybe only one or two meters in front of the cockpit." Because the way we see external events depends on the body's perception of it- self in space, acceleration and inertil forces that disrupt the inner ear's delicate sense of balance can also lead to optical illusions. Still, Haines contends that many induced illusions are short-lived and cannot account for the majority of AIRCAT's cases. "If a pilot describes a disk-shaped airform with no visible means of propulsion pacing his right wing for 30 minutes, doing everything he's doing--and I have plenty of cases like that--then that's not an optical illusion, it's not a bird or balloon or meteor, it's not any of those prosaic explanations," Haines says. "We don't know what it is necessarily but we know quite clearly what it isn't." One sensational pilot-and-UFO case almost certainly had a prosaic explanation. On the afternoon of January 7, 1948, people near Godman Air Force Base at Fort Knox, Kentucky, reported an object in the sky that looked like "an ice cream cone topped with red." Captain Thomas F. Mantell, flying in command of a ferry flight of four F-51 Mustangs (P-51s had been redesignated F-51s the previous year), was asked to investigate. None of the fighters were equipped with oxy-gen, and after three dropped out of the chase Mantell continued alone. "It's directly ahead and above and still moving at about half my speed," he radioed. "The thing looks metallic

and of tremendous size. I'm going up to 20,000 feet, and if I'm no closer I'll abandon the chase." A few minutes later Mantell's airplane crashed, earning him the dubious distinction as the world's first "UFO martyr." Project Blue Book proposed that Mantell succumbed to hypoxia, or oxygen starvation, and crashed while chasing the planet Venus, but later evidence indicates he was pursuing a top-secret, high-atmosphere Skyhook balloon. The balloons, designed for upper-atmosphere research, were later used by the CIA for surveillance. At altitudes of 70,000 feet or more, the translucent plastic balloons would often be swept rapidly along by the jet stream. Mantell wasn't the last pilot to die while pursuing, or being pursued by, an alleged UFO. At 6:19 p.m. on Saturday, October 21, 1978, Frederick Valentich of Melbourne, Australia, took off from Moorabbin Airport aboard a rented Cessena 182 bound for nearby King Island. He planned to pick up a load of crayfish for his fellow officers at the Air Training Corps, where he was a flight instructor. An experienced daytime pilot with an unrestricted license and instrument rating, Valentich, 20, was relatively inexperienced at night flying. He was also a UFO enthusiast who, his father said later, had claimed a UFO sighting 10 months before his disappearance. Out of Melbourne, Valentich paralleled Cape Otway before heading over open water for King Island, where he was scheduled to land at 7:28. At 7:06 he radioed Melbourne Flight Service, asking, "Is there any known traffic in my area below 5,000 feet? Seems to be a large aircraft." Ground control asked what kind. "I cannot confirm," Valentich replied. "It has four bright lights that appear to be landing lights...[and] has just passed over me about 1,000 feet above... at the speed it's traveling are there any RAAF [Royal Australian Air Force] aircraft in the vicinity?" "Negative," answered Melbourne. "Confirm you cannot identify aircraft?" Valentich replied in the affirmative, adding three minutes later, "It's not an aircraft, it's ..." At that point there was a brief break in the recorded transmission that was later released to the Australian press. "It is flying past," Valentich continued. "It has a long shape. Cannot identify more than that... coming for me now. It seems to be stationary. I'm orbiting and the thing is orbiting on top of me. It has a green light and sort of metallic light on the outside." The pilot then informed air traffic controllers that the object had vanished. At 7:12 he was back on the air, reporting his "engine is rough-idling and coughing." Ground control asked what his intentions were; Valentich said, "Proceeding King Island. Unknown aircraft now hovering on top of me."

His radio transmission ended in a jarring 17-second metallic noise. Neither pilot nor airplane has been seen or heard from since. Some have attempted to explain away the incident as a hoax or a suicide, while others have suggested that the inexperienced night pilot, overcome by vertigo, may have turned upside down and seen the reflections of his own lights before the engine of his Cessna failed. Haines has published a book about the Valentich incident, "Melbourne Episode: Case Study of a Missing Pilot," and he is in the midst of another compiling all of AIRCAT's cases. Most are variations on ufology's two major themes: daylight disks and noturnal lights. The first involves what appears to be objects in the shape of disks, spheres, or elliptical forms. Nocturnal lights normally appear as single, continuously visible white light sources. Sometimes the lights are also detected by ground or airborne radar and less frequently, accompanied by radio static and brief engine interruption, such as that experienced by Valentich. Most sightings involve two or more witnesses and last slightly more than five minutes, long enough in most cases, says Haines, to eliminate a number of explanations, such as meteors and ballons. One case from the AIRCAT files involved a pilot--call him Captain Gray--who had logged more than 21,000 hours in a 31-year career. On July 4, 1981, he was piloting a passenger flight in a Lockheed L-1011 Tristar, cruising on automatic pilot at 37,000 feet. The flight was bound from San Francisco to New York's Kennedy Airport, approaching the eastern shore of Lake Michigan. The lake below was obscured by clouds, but ahead and above the sky was clear. Suddenly, from ahead and to the left of the aircraft, a silvery disk "splashed into view full size...like the atmosphere opened up," Gray said later. He leaned forward, blurting out, "What's that?" Appearing at first like a sombrero viewed from the top, the object rolled as it approached the airplane along an arc that carried it toward and then aburptly away from the L-1011. From the side, the disk appeared ten times wider than it was thick, with six evenly spaced, jet black portholes along its edge. A bright splash of sunlight flared off the top left end of the object. As it disappeared, seemingly in a shallow climb, Gray noticed what looked like the dark smudge of a contrail. "Did you just see anything?" Gray asked his first officer. "Yes," he replied, "a very bright light flash." The flight engineer, his view blocked, had seen nothing. The overriding question for ufologists is whether a sighting like Captain Gray's is a natural phenomenon or an object that displays evidence of in- telligence. "As a scientist I have to be cautious," says Haines.

"But when AIRCAT is made public, I think the technical-minded can read between the lines." Skeptics would disagree, "I think there are more than enough ordinary stimuli floating around to create the UFO phenomena, the UFO social event, of the past 40 years," says CSICOP's James Oberg. "Because of imperfections in human memory and perception, coincidences and so on, there'll always be a small residue of unsolved sightings. A small percent of airplane crashes, murders, and missing-person cases don't get solved either. But you don't have to invoke alien airplane saboteurs, murderers, or kidnappers to explain them." Haines retorts that Captain Gray was a skeptic before his own UFO confront- ation. But afterwards, "there was no doubt in his mind whatsoever' that what he had seen was an extraterrestrial spacecraft. Captain Terauchi of JAL flight 1628 was equally convinced that he had encount- ered an extraterrestrial craft in the skies above Alaska. Skeptics are not so sure, citing the fact that Terauchi had reported seeing UFOs on two previous occasions--and would report yet another sighting the following January, again over Alaska. (He would later explain his second Alaskan encounter as city lights reflecting off ice crystals in the clouds.) CSICOP's Philip Klass thinks that ice crystals in clouds played a significant role in the November encounter. He theorizes that moonlight reflecting off the clouds accounts for the initial sighting, and that when the crew later saw Mars and Jupiter, bright in the autumn sky, they assumed the planets were lights from the original UFO. The signal on the onboard radar, Klass believes, could have been reflected by the same ice crystals (although ice crystals, unlike rain droplets, are very poor reflectors of radar energy). The FAA analyzed the ground radar and con- cluded that they had been uncorrelated radar signals, a common phenomenon that occurs when a radar beam bounced back from an airplane to a ground station doesn't match up with a separate signal sent by the airplane's transponder. That pilots, as well as ground observers, have seen something in the skies is undeniable. The question of what they have senn has yet to be satisfactorily resolved. Maybe it never will be. It may even be irrelevant. As Jacques Valle, who has wriiten several books on the subject, once said,"It no longer matters whether UFOs are real or not, because people BEHAVE as if they were, anyway."

Κείμενο 5: Unix

Unix is an operating system, so designated because it allows a user to interface with a computer in a way that is (hopefully) easy for the user to learn and use. Unix can be known by other forms, PC-Unix, Xenix, etc., but they all basically are the same (with slight differences this file won't go into) and use the same commands. Unix is a wonderfully simple to use OS once you begin, and while this file will help you, I recommend that you find a Unix system somewhere and wander around on it to help yourself to learn. To put this more formally: The UNIX system is a set of programs that include a time-sharing operating system and a set of utility programs. The operating system has two basic parts:

1. The kernel is the program in the UNIX operating system that is responsible for most operating system functions. It schedules and manages all the work done by the computer and maintains the file system. It is always running and is invisible to users.
2. The shell is the UNIX operating system program responsible for handling all interaction between users and the computer.

It includes a powerful command language called "shell language"*. The utility programs (usually called UNIX commands) are executed through the shell, and allow users to communicate with each other, to edit and manipulate files, to write and execute programs in several programming languages, and many other things.

Part II: Recognizing a Unix system

When you connect to a Unix system you will see a message usually like "AT&T Unix: Unauthorized use will be Prosecuted!" or just "Unix System V" or the like. At the least you will see a prompt saying "login:". At this point, if possible, make sure that you are in lowercase, because if the computer det- ects that you are typing in uppercase everything you read after will be in uppercase with lowercase denoted by a \ in front of the word. This is because Unix is case sensitive, so be careful, reading lowercase is much easier than reading all uppercase and slashes. Ok, so here you are at the Unix "login:" prompt.

Part III: Logging on

At this point you must enter your login, and then, if the account (never more than 14 characters) has one, the password. Now, all Unix systems have default accounts, and unless set by the Root System Operator no passwords. This has been the means of infiltration by many the Unix hacker. There are two types of accounts in a Unix, the "super user" and the "user". The super user has access to almost everything (or everything depending on the system) and the user basically has access to the files he owns and what he can sometimes read. The default super user accounts on a unix are: ROOT MAKEFSYS MOUNTFSYS UMOUNTFSYS CHECKFSYS and sometimes ADMIN SYSADMIN. For passwords to these try things like SYSTEM, SYSMAN, SYSADMIN, ADMINISTRATOR, OPERATOR, SYSOP, etc. The default user-level accounts are: LP DAEMON TROUBLE NUUCP UUCP RJE ADM SYSADM SYNC BIN (Note: These accounts should be entered in lower case , I merely wrote them in upper case for easier reference.) After being on Unix's, I have also seen the following common accounts: USER UNIX GAMES GUEST STUDENT -on school run Unix's. The maximum length of a password is 11 characters. After doing all this you should, with luck, be in! If you couldn't hack anything out, try typing "WHO" at the login: prompt, it may list all the user accounts and you can try them until you find one without a password.

Part IV: You're in!!!

Congratulate yourself, the hardest part of Unix "hacking" is over. Ok, now that you're in you'll see a prompt which will probably look like "$" for a user account or "#" if you got lucky and got a super user account. (Quick note, to stop a unix process in action try typing ctrl-d or control backspace, these are the end of file/Stop process keys.) Ok, so you are now in. Let me give a quick lesson on Unix directories. In Unix, the root is the main directory, and it contains subdirectories which may contain subdirectories etc. In order to change to the root directory, one would type "cd /". This is because "cd" is the command "change directory" and "/" is the root directory. To change to subdirectory "Bill" contained in the root directory, you would type "cd /Bill" or, if you were in the root dir, just "cd Bill". If you wanted to access Bill's files, you'd enter "cd /Bill/files" assuming Bill had a

subdir called files where he kept his files. This is how a person would move around in a Unix sys. Graphically, it looks like this:

Root _____!!_____ !! __Bill__ !! __Files__

Part V: Basic Commands

Ok, these commands are the most useful ones that I've found and can are entered from the prompt. Command:What it does ---------- ls gives a listing of all files in a directory cat gives a dump to screen of what is contained in a file. For instance "cat phones" would show me what is in file "phones". cd change directory pwd shows what directory path you are in now ps shows system processes rm remove a file, for instance "rm phones". rmdir removes a directory, for instance "rm Bill". grep print ascii strings in a file, ie "grep phones" who shows who's on the system mail sends mail to a user, syntax mail <username> su change from 1 account to another. For instance, if you are account Bill and wish to change to account Jake (which is unpassworded) just type "su Jake" and you will change to him. If Jake has a password you will be prompted to enter it. This is useful for loggin in under a user account and switching later to a super user account. passwd allows a user to change his password. If you are a superuser you can change someone elses password by typing "passwd <account>". mkuser make a user (providing you are a super user) mkdir create a directory More Information about Commands.

The following are more of the most basic Unix commands:

cat cd chmod cp cut date echo egrep fgrep file find glossary grep help ln locate ls mail mesg mkdir mv news pr ps pwd rm sleep sort starter stty tabs tail tee time touch tty uname usage wall wc who write Using the Command: mkdir Syntax Summary: mkdir dir_name1 [dir_name2 ...] where: dir_names are simple subdirectory names, relative pathnames, or full pathnames.

Description: mkdir creates one or more new directories. If mkdir is given a simple name as an argument, the new directory will be a subdirectory of the current directory. You can make new directories anywhere in the file system by giving mkdir a complete or

relative pathname for the new directories, if you have permission to write in the directory where the new directory is to be created. Ok, those are the basic commands you will need to go around in the system.

Part VI: Useful Information

A great place to go to get information on who is on the system and what accounts you can use to get on again is contained in the file "passwd" in the "etc" directory. To look at it, cd etc, and then cat passwd. The first entry should say something like this: root:adfaBADca:0:1:Operator:/:/bin/sh what this means is that the root account has an encrypted password, has super- user capabilities (any user with a 0 in that slot is a super user) is in group 1 (relatively unimportant for this file), has a comment of Operator (this may be blank), has a home directory of / (the root) and uses the Bourne Shell, kept in the /bin directory. You will then see all the other users listed out in the same format. If you see an account followed by two colons, that means that it has no password. You want these accounts so that you can log in under them another time. If you get real lucky you may see something like this: makefsys::0:1:/bin:/bin/sh meaning that you have found a super user account with no password, a very useful item indeed. Another good place to look is the /usr/spool dir and the /usr/spool/cron/crontabs dir because if you are a super user that dir contains much that will be useful to you. In order to move up to a directory one level higher than you are presently in, type "cd ..". So to move from /Bill/files to /Bill I would just type cd .. and, assuming I started in /Bill/files I would now be in /Bill. Ok, now you can wander the system "cat"'ing around and whatnot. If a file doesn't "cat", try just typing it's name, that will execute it if you have the privileges. Try typing "admin" or "ua" if you are a superuser nad maybe you'll be able to create users or other interesting things. You may not be able to cat a file or run it because you lack access permissions. What are they? Read on!

Access Permissions

access permissions: permissions: mode: owner: owner/group/others: read/write/execute As the user of a UNIX system, you can decide who can read, write, and execute the files and directories that you own. You are usually the owner of files and directories that you

have created in your login directory and in the "subdirectories"* in your login directory. You may also own files in other peoples' directories. You control the use of your files and directories by specifying the access permissions, also called the mode, for each. You can specify different access permissions for yourself, your "group"*, and the other users of the system. Permission to read allows the user to read the contents of the file. Write permission allows the user to change the file and execute permission enables the user to execute the program within the file. ls -l prints the access permissions for each file and directory in the current directory. The sample listing below shows the mode of the file (preceded by a -), the number of "links"*, the owner, the "group ID"*, the size in characters, the date and time the file was last modified, and the "filename"*. -rwxr-x--x 1 sandy 12345 128 Oct 9 9:32 lock If this were a listing for a directory, the hyphen (-) would be replaced by the letter d. The owner of the file "lock" can read, write and execute the file, the group can read and execute it, and the others can only execute it. You can change the mode of your files and directories by using the change mode command, chmod. Other interesting places to look are in the directories assigned to the users on the Unix system, often their files will contain some useful information. Also try going into the /uucp directory or looking for any uucp dir anywhere as it may contain phone numbers to other Unix systems or other "goodies".

The *: asterisk

In the shell, an asterisk matches any "string"* of characters in a "filename"* on a command line. The command rm temp* removes all files from the current working directory that begin with the string "temp". Files like "temp", "temp1", "temp.1", and "temp.save" would all be deleted. An asterisk alone matches any filename in the current working directory except those beginning with "dot (.)"*. For example, rm * removes all the files in your directory except for the dot (.)files. Finally, typing help at the unix prompt may bring up a help manual that is usually quite well done and will help you if you are stuck or wish to explore in more depth the commands I didn't go into. Hmm, what else? I can't think of much more right now that would help you much more, in this file I think I've covered everything that should get you well on your way towards becoming a unix hacker. Once you've got this, start reading files on "Unix Shells", "Scripts", and ask around A LOT. Ah,

I just remembered something. To get help on a command, type "man <command>" or "whatis <command> " and you may find out. Also, a lot of Unix's have a built in Help feature somewhere, try to get to it.

Part VII: A Few Final Words

If you manage to get onto a Unix system, don't screw it up. Unix is a great operating system, and fun to learn on and have other people learn on. Don't become a superuser and delete everything or other things, it's just not worth it. Also, don't make a use called "Hacker" or "Shadow 1" or something, that's a blatant giveaway. Put an account a little out of the way directory, and create user level accounts if you must, and perhaps just 1 super user level. I can't think of much more to say on the basics, though I probably left some important things out....nobody's perfect. I hope you enjoyed the file and I can be found on the following boards: The Private Connection The Undergraduates Lounge Quick Shop Phreak Klass 2600 The Brewery The Works Slaughterhouse 5, Holovision Network Node 1 Spock's Brain Special Thanks to: The Prophet, for his excellent file: Unix Use and Security From the Ground Up. The End, good luck, enjoy yourself, and don't get caught!

Κείμενο 6: Unix 2

The UNIX operating system is a set of programs that act as a link between the computer and the user. The programs that allocate the system resources and coordinate all the details of the computer's internals is called the operating system or kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel. Here is a basic block diagram of a UNIX system. Spread Sheet Compilers Calculators ³ ³ ³ V V ³ عؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ؟ The Shell ³ Mail and ³ ³ عؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ؟<- Message Inventory ³ ³ UNIX system kernel ³ ³ Facilities Control --->³ ³ ³ ³ عؤؤؤؤؤؤؤؤؤؤؤؤ؟ Systems ³ ³ ³ ³ ³ ³<- Interpreters ³ ³ ³ Hardware ³ ³ ³ Formatters ->³ ³ ³ ³ ³ ³<- DBMS ³ ³ ³ ³ ³ ³ Calendar ³ ³ ³ ³ ظؤؤؤؤؤؤؤؤؤؤؤؤه Word Systems ---->³ ³ ³ ³<- Processors ³

³ ظؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ Editors ---->³ ³<- FTP ظؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ

The designers of UNIX used the following Maxims while writing the new operating system.

1. Make each program do one thing well. These simple programs would be called "tools."
2. Expect the output of every program to be the input to another program.
3. Don't stop building new "tools" to do a job. The library of tools should keep increasing.

1.1 The UNIX System

The main concept that unites all versions of UNIX is the following four basics: Kernel The kernel is the heart of the operating system. It schedules tasks and manages data storage. The user rarely interfaces with the kernel directly. This is the memory resident portion of the operating system. Shell The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. The shell will support multiple users, multiple tasks, and multiple interfaces to itself. The shell uses standard syntax for all commands. There are two popular shells currently available, the BourneShell (standard System V UNIX) and the CShell (BSD UNIX). Because separate users can use different shells at the same time, the system can appear different to different users. There is another shell known as the KornShell (named after its designer), which is popular with programmers. This ability to provide a customized user interface is one of the most powerful features of UNIX. Commands and Utilities Separate utilities can be easily combined to customize function and output. They are flexible, adaptable, portable, and modular. They use pipes and filters. There are over 200 standard commands plus numerous others provided through 3rd party software. Files and Directories The directory system supports a multilevel hierarchy. Files and directories have access protection. Files and directories are accessed through pathnames. Files support multiple name links. Removable filesystems are also supported.

1.2 File Structure

All data in UNIX is organized into files. All files are organized into directories. These directories are organized into a tree-like structure called the filesystem. The following

diagram describes the top level organization of the UNIX filesystem: / (root)
³ ³ ³ ³ ³ ³ ³ ³ عؤؤؤؤؤؤآؤؤؤؤؤؤآؤؤؤؤؤؤآؤؤءؤؤؤؤآؤؤؤؤؤآؤؤؤؤؤ؟ bin dev
etc lib tmp usr users These directories, in turn, are also organized
hierarchically. For example: / ³
³ ³ ³ عؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤآؤؤءؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ؟ dev etc usr
³ ³ ³ ³ ³ ³ ³ ³ ³ ³ عؤؤؤؤؤؤآؤؤءؤؤؤؤؤؤؤؤؤ؟ عؤؤءؤؤ؟ عؤؤءؤؤ؟ عؤؤءؤؤ؟
dsk rmt init.d rc0.d mail adm spool ³ عؤؤءؤؤؤ؟
³ ³ acct sa In this example, dev, etc, usr, and adm are directories. Directories contain other files or directories. Plain files contain text or binary data and contain no information about other files or directories. Users can make use of this same structure to organize their files. For example: / ³
³ ³ ³ عؤؤؤؤؤؤؤؤؤؤؤؤؤإؤؤؤؤؤؤؤؤؤؤؤ؟ bin users dev ³
³ ³ عؤؤؤؤؤؤؤؤؤؤؤؤءؤؤؤؤؤؤؤؤؤؤؤؤؤؤ؟ bsmith sjones ³
³ ³ ³ ³ ³ ³ عؤؤؤؤؤؤؤؤءؤؤآؤؤؤؤؤؤؤؤؤ؟ عؤؤؤؤءؤؤؤؤ؟ memos progs
physics chem history ³ ³ ³ ³ ³ عءؤؤ؟ عؤؤءؤؤ؟ عؤؤءؤؤؤ؟ عؤؤءؤؤؤ؟ عؤؤءؤؤ؟
³ ³ ³ ³ ³ ³ ³ ³ ³ ³ mfg eng c f77 mods calcs forms notes loc anc Every file has a name. A filename is composed of one to fourteen characters. Although you can use almost any character in a filename, you will avoid confusion if you choose characters from the following list.

1. upper case letters [A-Z]
2. lower case letters [a-z]
3. numbers [0-9]
4. underscore [_]
5. period [.]
6. comma [,]

The only exception is the root directory, which always uses the symbol /. No other directory or file can use this symbol. Like children of one parent, no two files in the same directory can have the same name. Files in different directories, like children of different parents, can have the same name. The filenames you choose should mean something. Too often, a directory is filled with important files with names like foobar, wombat, and

junk. A meaningless name won't help you recall the contents of a file. Use filenames that are descriptive of the contents.

1.3 UNIX System

Files In order for you to have a basic understanding of the contents of some of the system directories, here is a partial list of those directories and what files they contain: /bin This is where the executable files are located. They are available to all user. /dev These are device drivers. /etc Supervisor directory commands, configuration files, disk configuration files, reboot files, valid user lists, groups, ethernet, hosts, where to send critical messages. /lib compiler libraries /tmp scratch processes, editors, compilers, and databases /bsd Berkeley commands /mnt empty, used for disks /stand boot information /lost+found orphans go here (look here after system crash) /unix* executable, bootable kernel This is not an exhaustive list of directories that contain system information but it is intended to remove some of the mystery behind these directories and the types of files they contain.

1.4 Command Line Syntax

Users enter commands at the shell prompt. The default BourneShell prompt is the dollar sign ($). In general, the shell expects to see the following syntax:

Command Format: command options arguments

Command - This is the UNIX command. Sometimes the command is representative of the function. For example, the command to list the contents of a directory is ls. The first and third letters of the word "list" are used. Unfortunately, this is not always the case. Options - These are also known as flags. The common form is: -A where A is the abbreviation of the optional function of the command. For example, the command ls lists the contents of a directory, while the command ls -l provides a long listing and ls -C provides output in columns. Several options can be combined following one '-'; for example -CF, or they can be entered separately as -C -F. Arguments - These can be file

names, user names, or qualifiers to the command or one of its options. Example: ..... . $ls -CF sjones . ..... The UNIX command is ls list contents of directory the dash (-) indicates the options. C = Multiple-column output with entries sorted down the columns F = Put a slash (/) after each filename if that file is a directory and put an asterisk (*) after each filename that is executable. sjones = name of the directory to list (it can be a relative or absolute pathname) Example: ..... . $diff memo1 memo2 . ..... diff - differential file comparator command memo1 - filename argument memo2 - filename argument This command will tell what lines must be changed in two files to bring them into agreement. Here is another example that doesn't fit the general syntax for UNIX commands. Example: ..... . $find . -atime +7 -print . ..... find - find files . - the current working directory -atime - True if the file has been accessed in n days (n is the +7) -print - always true; causes the current path name to be printed So, this command will give a listing of all files in your current working directory that have been accessed in the past seven days. Some commands have several options and/or arguments; while others, like passwd and mail, are interactive and will prompt the user for additional input.

1.5 Correcting Mistakes

Because the shell and most other utilities do not interpret the command line (or other text) until you press the (Ret) key, you can correct typing mistakes before you press (Ret). There are two ways to correct typing mistakes. You can erase one character at a time, or you can back up to the beginning of the command line in one step. After you press (Ret), it is too late to make a correction.

1.5.1 Erasing Characters When entering characters from the keyboard, you can backspace up to and over a mistake by pressing the erase key (#) one time for each character you wish to delete. The # will appear on the screen, and the character preceding

it will be discounted. Example: ..... . $ls phajne#y . ..... In this example, the e will be ignored and ls phajny is sent to the Multimax. Multiple typos can be erased; simply press one # for each character to be erased. The erase key will back up as many characters as you wish, but it will not back up past the beginning of the line.

1.5.2 Deleting an Entire Line You can delete an entire line you are entering any time before you press (Ret) by pressing the kill key (@). When you press the @ (kill key), the cursor moves down to the next line and all the way to the left. The shell doesn't give you another prompt, but it is as though the cursor is following a prompt. The operating system does not remove the line with the mistake but instead ignores it. Now enter the command (or text) again from the start.

1.5.3 Aborting Program Execution Sometimes you may want to terminate a running program. UNIX might be performing a listing that is too long to display on your screen or for some other reason you want to terminate execution. To terminate program execution press the Delete key. The operating system sends a terminal interrupt signal to the shell. When the shell receives this signal, it displays a prompt and waits for another command.

1.5.4 Controlling Output to the Screen There are several ways to control the flow of characters to the screen as a result of executing a command. Such as: Ctrl-S - This keyboard function command will suspend the flow of characters to the screen as the result of executing a command. The screen will not continue until the keyboard function to resume output is given. Ctrl-Q - This keyboard function command will resume the output to the screen. Hold Screen - If your terminal has this key (i.e. VT200), you can press it once to stop output to the screen. To resume output to the screen, press the key again. Denver BOR MULTIMAX Each BOR Multimax 310 has four 15 Megahertz National Semiconductor 32-bit processors with 64 kilobytes of cache memory rated at 2 million instructions per second (MIPS) for a total of 8 MIPS. The main memory consists of 32 megabytes (million bytes). There can be a maximum of 14 disk drives. Each drive has a

capacity of 600 megabytes for a total capacity of 8.4 gigabytes (a gigabyte is one thousand million bytes) Connection to the Multimax is accomplished through one of several methods. Access is made through TCP/IP based Annex terminal servers. The two Annex II servers have 32 ports each and the Annex I has 16 ports. The Annex II servers will allow up to 64 users access to the two Multimax computers. The Annex I is used for access to the on-line printers. CDCnet and TELNET are other ways to gain access to the Multimaxes. Printouts are handled on a 600-line-per-minute line printer and a 10-page-per-minute laser printer. Each Multimax has a hardcopy terminal and a CRT to serve as an operator console. There are two tape drives capable of 1600 or 6250 bits per inch (bpi) on each system. There is also a cassette tape drive. Software available are FORTRAN, COBOL, C, and UNISOL (an accounting package). The database management system is INGRES by Relational Technology, Inc. PROCOMM+ will be the communication interface with IBM PC's and compatibles. The operating system for the Multimax is UMAX V. UMAX V is the name for the Encore implementation of UNIX System V.

1.6 Logging on the Annex

This sample session shows how the login process is displayed on the terminal screen and is uniform for all users. To bring the standard menu onto the screen, press the Space Bar. If you are using a PC, first start PROCOMM+. Then when you are in the Terminal-Mode Screen, press the Space Bar; and the MICOM menu will appear. NOTE: Login procedures from the regions are included in the back of this manual Sample Session: ³ ؟ؘؘؘ

name you must enter to be connected to the Annex, which is the Multimax front end processor. Some MICOM menus might not have the MAX selection; in this case, enter MAX to select the Annex. This is the same as if the menu showed the option. After entering MAX you will see something similar to the following: Sample Session: ³ عؤؤ؟ CONNECTED TO 06/011 ³ بؤؤظ
This indicates that you are connected to the port selector. Wait two seconds, press (Ret) twice, and the annex prompt will appear after a warning message. Sample Session: ³ عؤؤ؟
³ ³ Annex Command Line Interpreter * Copyright 1988 Xylogics, Inc. ³ ³
³ ³ ***WARNING***Unauthorized access to U.S. Government computers ³ ³ is punishable by fine and/or imprisonment. ***WARNING*** ³ ³ annex: ³
بؤؤظ

1.7 Logging on the Multimax

To establish a connection between the Annex and the Multimax enter the following command at the Annex prompt:
ة»ح؛
Command Format: rlogin <host> ؛؛ ؛؛ host - name of the Multimax ؛
¼بح The Denver Multimaxes have been assigned the names domax0 and domax1. The names stand for the Denver Office Multimax System 0 and 1. The domax0 is used for production of Bureau-wide applications. The domax1 is used for training and application development and it is the one to use for exercises associated with this course. To enter domax1 type: Sample Session:
³ عؤؤ؟ annex:rlogin domax1
³ بؤؤظ or
³ عؤؤ؟ annex:r domax1
³ بؤؤظ NOTE: Abbreviations are allowed for the Annex commands, the only requirement is to type

in enough characters to make it unique. When the Annex has opened communications with the selected host, the following prompt will appear: Sample Session: ³ عؤؤ؟ login: ³ ظؤؤبؤ To connect with the host, enter your login name at the prompt. Your login name is assigned to you by the system administrator and typically will be your first initial and last name, all one word with no spaces. Only 8 characters are allowed for the username so extra letters will be truncated. Sample Session: ³ عؤؤ؟ login:rharding ³ ظؤؤبؤ Once the login name has been accepted, the next prompt will be for the password. The following prompt will appear on the screen. Sample Session: ³ عؤؤ؟ Password: ³ ظؤؤبؤ Enter your password. For security reasons, the host will not display your password as you type it. Sample Session: ³ عؤؤ؟ Password: secret ³ ظؤؤبؤ Once you have entered the correct password. The login procedure will continue and the following will appear on the monitor screen. Sample Session: ³ عؤؤؤ؟ UNIX System V Release ax.2.2o ns32332 ³ ³ domax1 ³ ³ Copyright (c) 1984 AT&T ³ ³ All Rights Reserved ³ ³ ***WARNING***Unauthorized access to/use of this U.S. Government ³ ³ computer is punishable by fine and/or imprisonment. ***WARNING***³ ³ $ ³ ظؤؤبؤ At this point you are successfully signed on to the Multimax. The dollar sign ($) is the default prompt for the BourneShell.

1.8 Logging Off the Multimax

At the shell prompt $, you can logout of the Multimax using one of the following methods:

1. Enter the keyboard function command Ctrl-D.

2. Type the UNIX command exit. Once you have entered the command to logout the following will appear on the screen: Sample Session:

```
$exit
CLI: Connection closed.
annex:
```

Once you are back at the Annex prompt, you can establish another connection or logout of the Annex.

1.9 Logging Off the Annex

When the Annex prompt (annex:) appears, you can enter the command to logout of the Annex. The command to logout of the Annex is as follows:

```
Command Format: hangup
```

There is a 60 minute inactivity timeout programmed into the Annex; however, it is a waste of resources if you don't enter hangup. When you are finished with your session, be sure to enter hangup at the annex: prompt. If you don't type anything for 60 minutes, the Annex will log you out of the system and display the following message: Sample Session:

```
*** Annex Port Reset Due to Inactivity Timeout ***
Annex Command Line Interpreter
DISCONNECTED
```

When the hangup command has been entered, the following will appear on the screen: Sample Session:

```
annex: hangup
Resetting line and disconnecting.
Annex Command Line Interpreter * Copyright 1988 Xylogics
annex:
DISCONNECTED
```

1.10 Changing the Password

The following command will change the password.

Command Format: passwd ¿

¼ You will be prompted to enter the existing password (this question is skipped if you don't have a password). Next you will be prompted to enter the new password. You will then be asked to enter the new password again. This will verify that you have not made a typographical error. If the two entries are the same, the password will be changed. The new password must meet the following criteria: NOTE: Some of these items are configurable by the system administrator and these reflect the settings for the Denver Multimax only.

1. Each password must have at least six characters. Only the first eight characters are significant.

2. Each password must contain at least two alphabetic characters and at least one numeric or special character. Alphabetic characters can be upper or lower case.

3. Each password must differ from the login name and any reverse or circular shift of that login name. For comparison purposes, an upper case letter and its corresponding lower case letter are equivalent.

4. A new password must differ from the old by at least three characters. For comparison purposes, an upper case letter and its corresponding lower case character are equivalent. Passwords on the Multimax have a thirteen-week expiration period. At the end of the thirteen weeks, you will be required to change your password. Once you have changed the password, you cannot change it again for two weeks. This prevents you from immediately changing back to the old password and eliminates a possible security violation. If you try to change the password before two weeks have passed since the last change, a warning message will be displayed. Sample Session:

³ $passwd

³ ³ Changing password for teacher ³ ³ Old password: secret ³ ³

Sorry: < 2 weeks since the last change ³ ³ $ ³

NOTE: This is about as friendly as UMAX will ever get. Try to choose a password that is not easy for someone else to guess. The increasing number of computer crimes involving thefts all point to a need for protecting the system from unauthorized access. Do not use words like your birthdate, telephone number, spouse's name, child's name, etc. for passwords. Although

you may think passwords are an unnecessary nuisance, they are an important way to strengthen the security of the computer system.

1.11 On-line Manual

The major source of on-line help is in the form of documentation known as the on-line manual pages. The pages are divided into eight sections. Section 1 contains entries for UMAX user commands; the other sections describe administrative tools, library functions, games, and internal system structure and calls. To gain access to the on-line manual pages enter the following command:

Command Format: man <command> command - the UNIX command you want information about

NOTE: The name 'man' stands for manual. Example: ..... . $man ls . ..... This command will display the on-line manual pages for the ls command. The on-line manual pages entry begins with the command name and a one line summary followed by a synopsis of the command line syntax. Optional flags and arguments are enclosed by square brackets []. A detailed description of the command and all of its options and arguments follow the synopsis. The description can include helpful examples. At the conclusion of the entry, related files and commands are listed. NOTE: Most on-line manual pages will fill more than one screen. Be sure to control the output to your screen.

1.12 who and finger Commands

Once you have logged onto the Multimax, you can find out who is logged on the system with the following commands:

Command Format: who [options] options - see man pages for a complete list

The default output (no options) of the who command lists the user's login name, terminal line, and the time that the user logged in. Sample Session: ³ عؤؤ؟ ³ $who ³ ³ jwheeler ttyp0 Aug 15 10:26 ³ ³ mvlsdba rt02190 Aug 15 09:25 ³ ³ teacher rt020b0 Aug 15 11:07 ³ ³ eholderf rt021c0 Aug 15 11:03 ³ ³ dbowman rt01150 Aug 15 08:58 ³ ³ $ ³ بؤؤظ Options will display other information about the users that are currently logged onto the system. Some items available are the amount of time that has elapsed since activity occurred on that line, the process identifier (PID) of the login process, comments, and exit information. A UNIX command that provides a little more information about users that are logged in the system is the finger command. ؛ »ح==========ة Command Format: finger [options] [user1] ؛ ؛ ؛ ؛ options - see on line manual for complete list ؛ ؛ ؛ ؛ user1 - login name ح==========ب ؛¼ The finger command with no options will list the login name, full name, terminal name, write status (an asterisk (*) before the terminal name indicates that write permission is denied), idle time, login time, office location, and phone number (if known) for each user that is currently logged in the system.The UNIX operating system is a set of programs that act as a link between the computer and the user. The programs that allocate the system resources and coordinate all the details of the computer's internals is called the operating system or kernel. Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel. Here is a basic block diagram of a UNIX system. Spread Sheet Compilers Calculators ³ ³ ³ V V ³ عؤؤ؟ The Shell ³ Mail and ³ ³ عؤؤ؟<- Message Inventory ³ ³ UNIX system kernel ³ ³ Facilities Control --->³ ³ ³ ³ عؤؤ؟ Systems ³ ³ ³ ³ ³ ³<- Interpreters ³ ³ ³ Hardware ³ ³ ³ Formatters ->³ ³ ³ ³ ³ ³<- DBMS ³ ³ ³ ³ ³ ³ Calendar ³ ³ ³ ³ بؤؤظ Word Systems ---->³ ³ ³ ³<- Processors ³

³ ظؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ Editors ---->³ ³<- FTP ظؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤؤ The designers of UNIX used the following Maxims while writing the new operating system.

1. Make each program do one thing well. These simple programs would be called "tools." 2. Expect the output of every program to be the input to another program. 3. Don't stop building new "tools" to do a job. The library of tools should keep increasing.

1.1 The UNIX System

The main concept that unites all versions of UNIX is the following four basics: Kernel The kernel is the heart of the operating system. It schedules tasks and manages data storage. The user rarely interfaces with the kernel directly. This is the memory resident portion of the operating system. Shell The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. The shell will support multiple users, multiple tasks, and multiple interfaces to itself. The shell uses standard syntax for all commands. There are two popular shells currently available, the BourneShell (standard System V UNIX) and the CShell (BSD UNIX). Because separate users can use different shells at the same time, the system can appear different to different users. There is another shell known as the KornShell (named after its designer), which is popular with programmers. This ability to provide a customized user interface is one of the most powerful features of UNIX. Commands and Utilities Separate utilities can be easily combined to customize function and output. They are flexible, adaptable, portable, and modular. They use pipes and filters. There are over 200 standard commands plus numerous others provided through 3rd party software. Files and Directories The directory system supports a multilevel hierarchy. Files and directories have access protection. Files and directories are accessed through pathnames. Files support multiple name links. Removable filesystems are also supported.

1.2 File Structure

All data in UNIX is organized into files. All files are organized into directories. These directories are organized into a tree-like structure called the filesystem. The following

diagram describes the top level organization of the UNIX filesystem: / (root)
ª ª ª ª ª ª ª ª عؤؤؤؤؤآؤؤؤؤؤآؤؤؤؤؤآؤؤءؤؤؤآؤؤؤآؤؤؤؤ؟ bin dev
etc lib tmp usr users These directories, in turn, are also organized
hierarchically. For example: / ª
ª ª ª عؤؤؤؤؤؤؤؤؤؤؤؤؤؤآؤءؤؤؤؤؤؤؤؤؤؤؤؤؤؤ؟ dev etc usr
ª ª ª ª ª ª ª ª ª ª ª عؤؤؤ؟ عؤؤؤ؟ عؤؤؤؤؤآؤءؤؤؤؤؤؤؤؤؤ؟ ª ª عؤؤؤ؟
dsk rmt init.d rc0.d mail adm spool ª عؤؤؤ؟
ª ª acct sa In this example, dev, etc, usr, and adm are directories.
Directories contain other files or directories. Plain files contain text or binary data and
contain no information about other files or directories. Users can make use of this same
structure to organize their files. For example: / ª
ª ª ª عؤؤؤؤؤؤؤؤؤؤؤؤؤؤإؤؤؤؤؤؤؤؤؤؤ؟ bin users dev ª
ª ª عؤؤؤؤؤؤؤؤؤؤؤؤؤؤءؤؤؤؤؤؤؤؤؤؤ؟ bsmith sjones ª
ª ª ª ª ª ª عؤؤؤؤؤؤؤؤؤءؤؤؤآؤؤؤؤؤؤؤؤ؟ ª عؤؤؤءؤؤؤ؟ memos progs
physics chem history ª ª ª ª ª ع؟ عؤؤءؤؤ؟ عؤؤءؤؤ؟ عؤؤءؤؤ؟ عؤؤءؤؤ؟
ª ª ª ª ª ª ª ª ª mfg eng c f77 mods calcs forms notes loc anc Every file has
a name. A filename is composed of one to fourteen characters. Although you can use
almost any character in a filename, you will avoid confusion if you choose characters from
the following list.

1. upper case letters [A-Z]
2. lower case letters [a-z]
3. numbers [0-9]
4. underscore [_]
5. period [.]
6. comma [,]

The only exception is the root directory, which always uses the symbol /. No other
directory or file can use this symbol. Like children of one parent, no two files in the same
directory can have the same name. Files in different directories, like children of different
parents, can have the same name. The filenames you choose should mean something.
Too often, a directory is filled with important files with names like foobar, wombat, and

junk. A meaningless name won't help you recall the contents of a file. Use filenames that are descriptive of the contents.

1.3 UNIX System

Files In order for you to have a basic understanding of the contents of some of the system directories, here is a partial list of those directories and what files they contain: /bin This is where the executable files are located. They are available to all user. /dev These are device drivers. /etc Supervisor directory commands, configuration files, disk configuration files, reboot files, valid user lists, groups, ethernet, hosts, where to send critical messages. /lib compiler libraries /tmp scratch processes, editors, compilers, and databases /bsd Berkeley commands /mnt empty, used for disks /stand boot information /lost+found orphans go here (look here after system crash) /unix* executable, bootable kernel This is not an exhaustive list of directories that contain system information but it is intended to remove some of the mystery behind these directories and the types of files they contain.

1.4 Command Line Syntax

Users enter commands at the shell prompt. The default BourneShell prompt is the dollar sign ($). In general, the shell expects to see the following syntax:

Command Format: command options arguments

Command - This is the UNIX command. Sometimes the command is representative of the function. For example, the command to list the contents of a directory is ls. The first and third letters of the word "list" are used. Unfortunately, this is not always the case. Options - These are also known as flags. The common form is: -A where A is the abbreviation of the optional function of the command. For example, the command ls lists the contents of a directory, while the command ls -l provides a long listing and ls -C provides output in columns. Several options can be combined following one '-'; for example -CF, or they can be entered separately as -C -F. Arguments - These can be file

names, user names, or qualifiers to the command or one of its options. Example: ..... . $ls -CF sjones . ..... The UNIX command is ls list contents of directory the dash (-) indicates the options. C = Multiple-column output with entries sorted down the columns F = Put a slash (/) after each filename if that file is a directory and put an asterisk (*) after each filename that is executable. sjones = name of the directory to list (it can be a relative or absolute pathname) Example: ..... . $diff memo1 memo2 . ..... diff - differential file comparator command memo1 - filename argument memo2 - filename argument This command will tell what lines must be changed in two files to bring them into agreement. Here is another example that doesn't fit the general syntax for UNIX commands. Example: ..... . $find . -atime +7 -print . ..... find - find files . - the current working directory -atime - True if the file has been accessed in n days (n is the +7) -print - always true; causes the current path name to be printed So, this command will give a listing of all files in your current working directory that have been accessed in the past seven days. Some commands have several options and/or arguments; while others, like passwd and mail, are interactive and will prompt the user for additional input.

1.5 Correcting Mistakes

Because the shell and most other utilities do not interpret the command line (or other text) until you press the (Ret) key, you can correct typing mistakes before you press (Ret). There are two ways to correct typing mistakes. You can erase one character at a time, or you can back up to the beginning of the command line in one step. After you press (Ret), it is too late to make a correction.

1.5.1 Erasing Characters

When entering characters from the keyboard, you can backspace up to and over a mistake by pressing the erase key (#) one time for each character you wish to delete. The # will appear on the screen, and the character preceding it will be discounted. Example: ..... . $ls phajne#y . ..... In this example, the e will be ignored and ls phajny is sent to the Multimax. Multiple typos can be erased; simply press one # for each character to be erased. The erase key will back up as many characters as you wish, but it will not back up past the beginning of the line.

1.5.2 Deleting an Entire Line

You can delete an entire line you are entering any time before you press (Ret) by pressing the kill key (@). When you press the @ (kill key), the cursor moves down to the next line and all the way to the left. The shell doesn't give you another prompt, but it is as though the cursor is following a prompt. The operating system does not remove the line with the mistake but instead ignores it. Now enter the command (or text) again from the start.

1.5.3 Aborting Program Execution

Sometimes you may want to terminate a running program. UNIX might be performing a listing that is too long to display on your screen or for some other reason you want to terminate execution. To terminate program execution press the Delete key. The operating system sends a terminal interrupt signal to the shell. When the shell receives this signal, it displays a prompt and waits for another command.

1.5.4 Controlling Output to the Screen

There are several ways to control the flow of characters to the screen as a result of executing a command. Such as: Ctrl-S - This keyboard function command will suspend the flow of characters to the screen as the result of executing a command. The screen will not continue until the keyboard function to resume output is given. Ctrl-Q - This keyboard function command will resume the output to the screen. Hold Screen - If your terminal has this key (i.e. VT200), you can press it once to stop output to the screen. To resume output to the

screen, press the key again. Denver BOR MULTIMAX Each BOR Multimax 310 has four 15 Megahertz National Semiconductor 32-bit processors with 64 kilobytes of cache memory rated at 2 million instructions per second (MIPS) for a total of 8 MIPS. The main memory consists of 32 megabytes (million bytes). There can be a maximum of 14 disk drives. Each drive has a capacity of 600 megabytes for a total capacity of 8.4 gigabytes (a gigabyte is one thousand million bytes) Connection to the Multimax is accomplished through one of several methods. Access is made through TCP/IP based Annex terminal servers. The two Annex II servers have 32 ports each and the Annex I has 16 ports. The Annex II servers will allow up to 64 users access to the two Multimax computers. The Annex I is used for access to the on-line printers. CDCnet and TELNET are other ways to gain access to the Multimaxes. Printouts are handled on a 600-line-per-minute line printer and a 10-page-per-minute laser printer. Each Multimax has a hardcopy terminal and a CRT to serve as an operator console. There are two tape drives capable of 1600 or 6250 bits per inch (bpi) on each system. There is also a cassette tape drive. Software available are FORTRAN, COBOL, C, and UNISOL (an accounting package). The database management system is INGRES by Relational Technology, Inc. PROCOMM+ will be the communication interface with IBM PC's and compatibles. The operating system for the Multimax is UMAX V. UMAX V is the name for the Encore implementation of UNIX System V.

1.6 Logging on the Annex

This sample session shows how the login process is displayed on the terminal screen and is uniform for all users. To bring the standard menu onto the screen, press the Space Bar. If you are using a PC, first start PROCOMM+. Then when you are in the Terminal-Mode Screen, press the Space Bar; and the MICOM menu will appear. NOTE: Login procedures from the regions are included in the back of this manual Sample Session: ³ ؟ۇۇۇعۇ WELCOME TO THE B.O.R. NETWORK P/S:B ³ ³ SYSTEMS PRESENTLY AVAILABLE ARE: ³ ³ ³ ³ **SYSTEM** **NAME** ³ ³ ³ ³ VAX 8300'S VAX ³ ³ CYBER/CDCNET F.E. CDC ³ ³ ENCORE/UNIX MAX ³ ³ OUT DIAL OD ³ ³ ³ ³ TO SELECT

A SYSTEM, ENTER THE SYSTEM ³ ³ NAME AND CARRIAGE RETURN AT NEXT ³ ³ PROMPT. ³ ³ ³ ³ CHANNEL 08/061. ENTER RESOURCE MAX ³ ³ ³

ظ MAX is the resource name you must enter to be connected to the Annex, which is the Multimax front end processor. Some MICOM menus might not have the MAX selection; in this case, enter MAX to select the Annex. This is the same as if the menu showed the option. After entering MAX you will see something similar to the following: Sample Session: ³ ؟ CONNECTED TO 06/011 ³ ظ This indicates that you are connected to the port selector. Wait two seconds, press (Ret) twice, and the annex prompt will appear after a warning message. Sample Session: ³ ؟

³ ³ Annex Command Line Interpreter * Copyright 1988 Xylogics, Inc. ³ ³ ³ ³ ***WARNING***Unauthorized access to U.S. Government computers ³ ³ is punishable by fine and/or imprisonment. ***WARNING*** ³ ³ annex: ³

ظ

1.7 Logging on the Multimax

To establish a connection between the Annex and the Multimax enter the following command at the Annex prompt:

ح« ؛ ة

Command Format: rlogin <host> ؛ ؛ ؛ ؛ host - name of the Multimax ؛

ح¼ب The Denver Multimaxes have been assigned the names domax0 and domax1. The names stand for the Denver Office Multimax System 0 and 1. The domax0 is used for production of Bureau-wide applications. The domax1 is used for training and application development and it is the one to use for exercises associated with this course. To enter domax1 type: Sample Session: ³ ؟ annex:rlogin domax1

or

annex:r domax1

NOTE: Abbreviations are allowed for the Annex commands, the only requirement is to type in enough characters to make it unique. When the Annex has opened communications with the selected host, the following prompt will appear: Sample Session:

login:

To connect with the host, enter your login name at the prompt. Your login name is assigned to you by the system administrator and typically will be your first initial and last name, all one word with no spaces. Only 8 characters are allowed for the username so extra letters will be truncated. Sample Session:

login:rharding

Once the login name has been accepted, the next prompt will be for the password. The following prompt will appear on the screen. Sample Session:

Password:

Enter your password. For security reasons, the host will not display your password as you type it. Sample Session:

Password: secret

Once you have entered the correct password. The login procedure will continue and the following will appear on the monitor screen. Sample Session:

UNIX System V Release ax.2.2o ns32332
domax1
Copyright (c) 1984 AT&T
All Rights Reserved
WARNINGUnauthorized access to/use of this U.S. Government computer is punishable by fine and/or imprisonment. ***WARNING***
$

At this point you are successfully signed on to the Multimax. The dollar sign ($) is the default prompt for the BourneShell.

1.8 Logging Off the Multimax

At the shell prompt $, you can logout of the Multimax using one of the following methods:

1. Enter the keyboard function command Ctrl-D.
2. Type the UNIX command exit. Once you have entered the command to logout the following will appear on the screen: Sample Session: ³ عؤؤ؟ ³ $exit ³ ³ CLI: Connection closed. ³ ³ annex: ³ ³ ظؤؤ Once you are back at the Annex prompt, you can establish another connection or logout of the Annex.

1.9 Logging Off the Annex

When the Annex prompt (annex:) appears, you can enter the command to logout of the Annex. The command to logout of the Annex is as follows: ؛ »ح═══════════════════════════════════ة Command Format: hangup ؛ ح═══════════════════════════════════¼ There is a 60 minute inactivity timeout programmed into the Annex; however, it is a waste of resources if you don't enter hangup. When you are finished with your session, be sure to enter hangup at the annex: prompt. If you don't type anything for 60 minutes, the Annex will log you out of the system and display the following message: Sample Session: ³ عؤؤ؟ *** Annex Port Reset Due to Inactivity Timeout *** ³ ³ ³ ³ Annex Command Line Interpreter ³ ³ DISCONNECTED ³ ³ ظؤؤ When the hangup command has been entered, the following will appear on the screen: Sample Session: ³ عؤؤ؟ annex: hangup ³ ³ ³ ³ Resetting line and disconnecting. ³ ³ ³ ³ Annex Command Line Interpreter * Copyright 1988 Xylogics ³ ³ annex: ³ ³ DISCONNECTED ³ ³ ظؤؤ

1.10 Changing the Password

The following command will change the password. ؛ »ةححح

Command Format: passwd ؛ ححح¼ You will be prompted to enter the existing password (this question is skipped if you don't have a password). Next you will be prompted to enter the new password. You will then be asked to enter the new password again. This will verify that you have not made a typographical error. If the two entries are the same, the password will be changed. The new password must meet the following criteria: NOTE: Some of these items are configurable by the system administrator and these reflect the settings for the Denver Multimax only.

1. Each password must have at least six characters. Only the first eight characters are significant.
2. Each password must contain at least two alphabetic characters and at least one numeric or special character. Alphabetic characters can be upper or lower case.
3. Each password must differ from the login name and any reverse or circular shift of that login name. For comparison purposes, an upper case letter and its corresponding lower case letter are equivalent.
4. A new password must differ from the old by at least three characters. For comparison purposes, an upper case letter and its corresponding lower case character are equivalent. Passwords on the Multimax have a thirteen-week expiration period. At the end of the thirteen weeks, you will be required to change your password. Once you have changed the password, you cannot change it again for two weeks. This prevents you from immediately changing back to the old password and eliminates a possible security violation. If you try to change the password before two weeks have passed since the last change, a warning message will be displayed. Sample Session: ³ عؤؤؤ؟ $passwd ³ ³ Changing password for teacher ³ ³ Old password: secret ³ ³ Sorry: < 2 weeks since the last change ³ ³ $ ³ ظؤؤؤ NOTE: This is about as

friendly as UMAX will ever get. Try to choose a password that is not easy for someone else to guess. The increasing number of computer crimes involving thefts all point to a need for protecting the system from unauthorized access. Do not use words like your birthdate, telephone number, spouse's name, child's name, etc. for passwords. Although you may think passwords are an unnecessary nuisance, they are an important way to strengthen the security of the computer system.

1.11 On-line Manual

The major source of on-line help is in the form of documentation known as the on-line manual pages. The pages are divided into eight sections. Section 1 contains entries for UMAX user commands; the other sections describe administrative tools, library functions, games, and internal system structure and calls. To gain access to the on-line manual pages enter the following command:

Command Format: man <command> command - the UNIX command you want information about NOTE: The name 'man' stands for manual. Example: . $man ls . This command will display the on-line manual pages for the ls command. The on-line manual pages entry begins with the command name and a one line summary followed by a synopsis of the command line syntax. Optional flags and arguments are enclosed by square brackets []. A detailed description of the command and all of its options and arguments follow the synopsis. The description can include helpful examples. At the conclusion of the entry, related files and commands are listed. NOTE: Most on-line manual pages will fill more than one screen. Be sure to control the output to your screen.

1.12 who and finger Commands

Once you have logged onto the Multimax, you can find out who is logged on the system with the following commands:

؛ ةححث

Command Format: who [options] ؛ ؛ ؛ ؛ options - see man pages for a complete list ؛ ؛ ؛

¼بححح

The default output (no options) of the who command lists the user's login name, terminal line, and the time that the user logged in. Sample Session:

³ $who عؤؤؤ؟ ³

³ jwheeler ttyp0 Aug 15 10:26 ³ ³ mvlsdba rt02190 Aug 15 09:25 ³ ³ teacher rt020b0 Aug 15 11:07 ³ ³ eholderf rt021c0 Aug 15 11:03 ³ ³ dbowman rt01150 Aug 15 08:58 ³ ³ $ ³

Options will display بؤؤؤظ other information about the users that are currently logged onto the system. Some items available are the amount of time that has elapsed since activity occurred on that line, the process identifier (PID) of the login process, comments, and exit information. A UNIX command that provides a little more information about users that are logged in the system is the finger command.

؛ ةحح«

Command Format: finger [options] [user1] ؛ ؛ ؛ ؛ options - see on line manual for complete list ؛ ؛ ؛ ؛ user1 - login name

¼؛ بحح The finger command with no options will list the login name, full name, terminal name, write status (an asterisk (*) before the terminal name indicates that write permission is denied), idle time, login time, office location, and phone number (if known) for each user that is currently logged in the system.