ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ,
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ ΚΑΙ ΔΙΚΤΥΩΝ

ΔΙΑΤΗΡΗΣΗ ΤΗΣ ΕΜΠΙΣΤΕΥΤΙΚΟΤΗΤΑΣ ΚΑΤΑ ΤΗΝ ΕΝΟΠΟΙΗΣΗ ΤΩΝ ΔΕΔΟΜΕΝΩΝ ΣΕ ΚΑΤΑΝΕΜΗΜΕΝΕΣ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΜΗΤΡΟΓΙΑΝΝΗΣ ΒΑΣΙΛΕΙΟΣ

ΕΠΙΒΛΕΠΟΝΤΕΣ ΚΑΘΗΓΗΤΕΣ: ΒΑΣΙΛΕΙΟΣ ΒΕΡΥΚΙΟΣ
ΓΕΩΡΓΙΟΣ ΜΟΥΣΤΑΚΙΔΗΣ

Σεπτέμβριος 2005

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά:

- Τους επιβλέποντες καθηγητές κ.κ. Β. Βερύκιο και Γ. Μουστακίδη για τη συνεργασία και την υποστήριξη τους σε όλα τα στάδια της εργασίας, για την απαραίτητη καθοδήγηση που μου παρείχαν, καθώς και για τις καθοριστικές συμβουλές τους σε όλα τα κρίσιμα ζητήματα.
- Την οικογένεια μου για τη συνεχή υποστήριξη και συμπαράσταση τους καθόλη την διάρκεια της εργασίας, και ιδιαίτερα στις κρίσιμες στιγμές που αντιμετώπιζα.

Περιεχόμενα

Λίστα Πινάκων

Λίστα Σχημάτων

Αντιστοίχηση Ελληνικής Ορολογίας στην Αγγλική

Ελληνικός Όρος	Αγγλικός όρος
Ακεραιότητα των δεδομένων	Data Integrity
Αλληλουχία	Concatenation
Αλφαριθμητικός χαρακτήρας	String
Αναγνωριστής	Identifier
Ανάθεση ετικετών	Tag assignment
Ανάκτηση πληροφορίας	Information Retrieval
Ανωνυμοποίηση	Anonymization
Δημιουργία συνόλων δεδομένων	Data Set Generation
Δημόσιο ή Ασύμμετρο Κλειδί	Public Key
Διαμοίραση των δεδομένων	Data Sharing
Διασύνδεση των εγγραφών	Record Linkage
Διατήρηση της εμπιστευτικότητας κατά την εξόρυξη δεδομένων	Privacy-preserving Data Mining
Διαχωριστικό	Separator
Διερμηνευτής	Interpreter
Διπλότυπο	Duplicate
Δυναμό-σύνολο	Power-set
Εμπιστευτικότητα των πληροφοριών	Information Privacy
Εμφωλιασμένο	Nested
Ενοποίηση των δεδομένων	Data Integration
Επερώτηση	Query
Ευρετήριο	Index
Καθαρισμός και Τυποποίηση εγγραφών	Record Cleaning and Standardization
Κανόνας απόφασης	Decision rule
Κατανεμημένη Βάση Δεδομένων	Distributed Database
Κατώφλι	Threshold
Μέθοδος Δέσμης	Bind Method
Μέθοδος Ροής	Flow Method
Μεταγλωττιστής	Compiler
Μετατοπιζόμενο παράθυρο	Sliding window

Μετρητής	Counter
Μυστικό ή Συμμετρικό Κλειδί	Private or Symmetric Key
Ν-γράμμα	N-gram
Οντότητα	Entity
Πίνακας κατακερματισμού	Hash table
Συνάθροιση	Aggregation
Συνάρτηση μονόδρομου κατακερματισμού	One-way Hash function
Συνένωση των δεδομένων	Data Join
Σύνολο αντικειμένων	Item-set
Σύνοψη μηνύματος	Message Digest
Ταίριασμα των αντικειμένων	Object Matching
Ταίριασμα των σχημάτων	Schema Matching
Ταξινομημένη Ευρετηριοποίηση	Sorting Indexing
Τμηματοποίηση	Segmentation
Υπό-ρουτίνα	Subroutine
Χαρακτήρας οριοθέτησης	Delimiter

Περίληψη

Στη σύγχρονη εποχή οι διάφοροι οργανισμοί, που είτε απασχολούνται στον επιχειρηματικό τομέα, είτε δραστηριοποιούνται σε διάφορα επιστημονικά πεδία, διαχειρίζονται έναν μεγάλο όγκο πληροφοριών. Στην κατεύθυνση αυτή έχει συμβάλλει σημαντικά η ανάπτυξη της τεχνολογίας η οποία επιτρέπει την διακίνηση και ανταλλαγή των πληροφοριών μεταξύ των οργανισμών με ευκολία, ταχύτητα και ασφάλεια.

Σε αυτήν την εργασία θα γίνει μια προσπάθεια διερεύνησης των τρόπων με τους οποίους είναι δυνατόν να επιτευχθεί η ενοποίηση των πληροφοριών μεταξύ των Βάσεων Δεδομένων, διατηρώντας παράλληλα την εμπιστευτικότητα των δεδομένων. Η συγκεκριμένη διαδικασία είναι ιδιαίτερα σημαντική για την περαιτέρω ανάπτυξη του τομέα των Βάσεων Δεδομένων, αφού οι έρευνες έχουν αποδείξει ότι οι οργανισμοί θα προβούν στην ενοποίηση των δεδομένων τους μόνο αν προφυλάσσεται η εμπιστευτικότητα τους.

Μέχρι σήμερα η κοινότητα των Βάσεων Δεδομένων έχει αναπτύξει διάφορες μεθόδους που επιτυγχάνουν την ενοποίηση των πληροφοριών. Μια από τις μεθόδους αυτές η οποία εξετάζεται διεξοδικά στην εργασία είναι η διασύνδεση των εγγραφών, δηλαδή το record linkage [3, 5, 10, 13]. Η διαδικασία της διασύνδεσης των εγγραφών ουσιαστικά συγκρίνει τις εγγραφές που υπάρχουν σε δυο ή περισσότερες Βάσεις Δεδομένων και εξετάζει κατά πόσο κάποιες εγγραφές αντιστοιχούν στην ίδια οντότητα. Οι περισσότεροι τρόποι που έχουν αναπτυχθεί μέχρι στιγμής για την διασύνδεση των εγγραφών δυστυχώς δεν ασχολούνται με το ζήτημα της διατήρησης της εμπιστευτικότητας.

Προς αυτήν την κατεύθυνση οδηγείται η εργασία. Αρχικά εξετάζεται το λογισμικό Febrl [4, 33] το οποίο μπορεί να επιτύχει την διασύνδεση των εγγραφών δυο συνόλων δεδομένων. Προτού όμως ξεκινήσει η διασύνδεση των εγγραφών, οι εφαρμογές του λογισμικού που περιγράφονται στην ενότητα 3.2 «καθαρίζουν» τα δεδομένα [4], ώστε να τα φέρουν όλα στην τυποποιημένη μορφή. Παράλληλα το λογισμικό παρέχει την

δυνατότητα παραγωγής νέων τεχνητών συνόλων δεδομένων για την εξέταση και τον έλεγχο της απόδοσης διαφόρων αλγορίθμων.

Κατόπιν παρουσιάζεται η τεχνική των N-γραμμάτων [5, 13, 15] που επιτυγχάνει την διασύνδεση των εγγραφών, συγκρίνοντας την ομοιότητα των αλφαριθμητικών. Μάλιστα περιγράφεται ένα πρωτόκολλο [5] σύγκρισης της ομοιότητας εμπιστευτικών δεδομένων που συνδυάζει αυτήν την τεχνική με μεθόδους συμμετρικής κρυπτογράφησης [5, 19] και συναρτήσεις κατακερματισμού [5, 19] προκειμένου να επιτύχει την ασφαλή ενοποίηση των δεδομένων. Με βάση αυτό το πρωτόκολλο υλοποιήθηκαν ορισμένα προγράμματα που επιτυγχάνουν τον στόχο της ασφαλούς ενοποίησης των δεδομένων.

Τέλος εξετάζοντας την αποδοτικότητα των προγραμμάτων, όσον αφορά την πρακτικότητα τους και την ποιότητα των αποτελεσμάτων τους, παράγονται ορισμένα χρήσιμα συμπεράσματα για την διαδικασία της ασφαλούς διασύνδεσης των εγγραφών, και ειδικότερα της εφαρμογής της τεχνικής των N-γραμμάτων σε συνδυασμό με κρυπτογραφικές τεχνικές.

Κεφάλαιο 1

Εισαγωγή

Στη σύγχρονη εποχή η ανάγκη για απόκτηση και διαχείριση όλο και περισσότερων πληροφοριών κρίνεται επιτακτική. Οι διάφοροι οργανισμοί, σε παγκόσμια κλίμακα, προσπαθούν να έχουν στη διάθεση τους όσο το δυνατόν πιο πολλές και πιο ποιοτικές πληροφορίες στους τομείς του ενδιαφέροντος τους, διότι μόνο κατά αυτόν τον τρόπο θα έχουν την δυνατότητα να ανταπεξέλθουν στις συνθήκες του ανταγωνισμού. Προς αυτήν την κατεύθυνση έχει συνεισφέρει σημαντικά η συνεχής και αλματώδης ανάπτυξη της τεχνολογίας, χάρη στην οποία μπορούμε ανά πάσα στιγμή να έχουμε πρόσβαση σε εκατομμύρια πληροφορίες [6].

Προκειμένου να επιτευχθεί η ορθή διαχείριση του μεγάλου όγκου των πληροφοριών η κοινότητα των Βάσεων Δεδομένων στρέφεται προς την δημιουργία ειδικών τεχνικών και μεθόδων ενοποίησης και διαμοίρασης των δεδομένων των πληροφοριών [6]. Ουσιαστικά με την ενοποίηση των δεδομένων εννοούμε τον συνδυασμό [6, 31], ή αλλιώς την συνάθροιση των εγγραφών, και γενικότερα των πληροφοριών, που διαθέτουν δυο ή περισσότερες Βάσεις Δεδομένων από διαφορετικές πηγές. Το ενδιαφέρον στη διαδικασία αυτή εστιάζεται στην προσπάθεια που γίνεται ώστε τα ενοποιημένα δεδομένα να μην περιλαμβάνουν περισσότερες από μια αναφορές στην ίδια οντότητα. Η ανάγκη αυτή, η οποία είχε ήδη αρχίσει να διαφαίνεται από τα τέλη της δεκαετίας του '80, έχει γίνει πλέον ιδιαίτερα κρίσιμη σε διάφορους τομείς. Μεταξύ άλλων, είναι πολύ σημαντικό να επιτυγχάνεται η ορθή ενοποίηση των δεδομένων που διαμοιράζονται πολλές επιχειρήσεις με κοινό πεδίο δράσης, η ανταλλαγή ιατρικών δεδομένων, η κοινή χρήση των στοιχείων που έχουν στην διάθεση τους οι ερευνητές ενός επιστημονικού πεδίου, καθώς και η ανταλλαγή δεδομένων μεταξύ κυβερνητικών οργανισμών.

Κατά τα πρώτα βήματα της εξέλιξης των τεχνικών επικρατούσε η άποψη ότι δεν υπήρχε κανένας περιορισμός, και φαινόταν λογικό ότι οι πληροφορίες που διέθεταν οι Βάσεις Δεδομένων των οργανισμών μπορούσαν ελευθέρως να διαμοιραστούν [31]. Πλέον πρόσφατα έχει αναγνωριστεί το γεγονός ότι κατά την ενοποίηση των δεδομένων ανακύπτουν ιδιαίτερα ζητήματα ασφαλείας. Η κοινή χρήση πληροφοριών από διάφορες πηγές βασίζεται στην μεταξύ τους συνεργασία, και έχει ως στόχο την βέλτιστη ανάπτυξη ενός τομέα δραστηριοτήτων, και το κοινό όφελος των παραγόντων που έχουν πρόσβαση στην γνώση. Πάντοτε όμως υπάρχει ο κίνδυνος να αποκτήσει πρόσβαση στις πληροφορίες κάποιος τρίτος, με άγνωστες συνέπειες για τους οργανισμούς που ανταλλάσσουν τα δεδομένα τους. Παράλληλα ενδέχεται κάποιο από τα συνεργαζόμενα μέλη να χρησιμοποιήσει τις νέες γνώσεις που έχει στην διάθεση του υστερόβουλα, και να προβεί σε ενέργειες που θα βελτιώσουν την θέση του απέναντι στον ανταγωνισμό. Επομένως οι διάφοροι οργανισμοί από την μια πλευρά δείχνουν ιδιαίτερο ενδιαφέρον στην ενοποίηση των δεδομένων τους ώστε να αποκομίσουν διάφορα οφέλη, από την άλλη όμως το ενδιαφέρον αυτό παρεμποδίζεται από τον φόβο που επικρατεί όσον αφορά τα ζητήματα παραβίασης της εμπιστευτικότητας των πληροφοριών που προκύπτουν [6].

Η διστακτικότητα αυτή ενισχύεται ακόμη περισσότερο από το γεγονός ότι δεν παρέχεται στον τομέα αυτόν ισχυρή νομική κάλυψη. Όπως συμβαίνει γενικότερα στην επιστήμη της πληροφορικής, η νομοθεσία και τα μέτρα που λαμβάνουν τα κράτη βρίσκονται πάντα ένα βήμα πίσω από την συνεχώς αναπτυσσόμενη τεχνολογία [23]. Εκτός των άλλων, οποιαδήποτε νομικά μέτρα δεν είναι αρκετά για να παρεμποδίσουν έναν κακόβουλο χρήστη ή οργανισμό, στον οποίο η παράνομη πρόσβαση σε νέες πληροφορίες θα αποφέρει τεράστια, οικονομικά συνήθως, οφέλη. Επομένως η κοινότητα των Βάσεων Δεδομένων τίθεται αντιμέτωπη με μια νέα πρόκληση όσον αφορά το θέμα της ασφάλειας των πληροφοριών. Όπως εύλογα λοιπόν αναφέρουν οι Clifton et al. [6], κρίνεται πλέον επιτακτική η ανάγκη να αναπτυχθούν τεχνικές οι οποίες θα επιτρέπουν την ενοποίηση και κοινή χρήση των δεδομένων χωρίς διαρροές σε ζητήματα εμπιστευτικότητας.

1.1 Γενικό Υπόβαθρο

Αντικείμενο της μελέτης είναι η ενοποίηση των δεδομένων που περιέχονται σε διαφορετικές Βάσεις Δεδομένων. Στο σημείο αυτό πρέπει να δοθεί ιδιαίτερη έμφαση στο γεγονός ότι οφείλουμε να μην συγχέουμε τους όρους «διατήρηση της εμπιστευτικότητας κατά την ενοποίηση δεδομένων» [6, 31] και «διατήρηση της εμπιστευτικότητας κατά την εξόρυξη δεδομένων» [25, 26, 27]. Οι δυο αυτοί όροι είναι ιδιαίτερα συνδεδεμένοι, διότι ουσιαστικά η ενοποίηση αποτελεί το προαπαιτούμενο για να επιτευχθεί η εξόρυξη δεδομένων από πολλαπλές πηγές. Επομένως κατά την διαδικασία της εξόρυξης θεωρείται ότι έχει ήδη επιτευχθεί η ενοποίηση. Αυτό σημαίνει ότι τα ζητήματα ασφαλείας που έχει να αντιμετωπίσει η ενοποίηση των δεδομένων διαφέρουν κατά πολύ από της εξόρυξης δεδομένων, διότι κατά την ενοποίηση ο στόχος είναι να ενωθούν οι εγγραφές δύο ή περισσότερων Βάσεων Δεδομένων χωρίς να υπάρξει διαρροή ευαίσθητων πληροφοριών, ενώ κατά την εξόρυξη σε μια Βάση Δεδομένων ανακτάται η γνώση με τέτοιο τρόπο ώστε να μην παραβιάζονται τα ευαίσθητα δεδομένα.

Η ενοποίηση των δεδομένων που προέρχονται από διαφορετικές πηγές έχει να αντιμετωπίσει συγκεκριμένα προβλήματα – προκλήσεις που αφορούν την διατήρηση της εμπιστευτικότητας[6]:

- Πρέπει να συσταθεί ένα συγκεκριμένο πλαίσιο ασφάλειας το οποίο θα είναι ευπροσάρμοστο ανάλογα με την περίσταση, και παράλληλα θα είναι εύκολα κατανοητό από τα άτομα που πρόκειται να το χρησιμοποιήσουν. Στην κατεύθυνση αυτή πρέπει να αναπτυχθούν ειδικές συμβάσεις για την πρόσβαση και την διαχείριση των Βάσεων Δεδομένων με γνώμονα την εξισορρόπηση των παραμέτρων κέρδους – ρίσκου. Κατά πόσο δηλαδή συμφέρει να υπάρχει κάποιο ρίσκο διαρροής στην ασφάλεια όταν αποκομίζονται αρκετά οφέλη.
- Έπειτα ακολουθεί η τυποποίηση των εγγραφών [4]. Προκειμένου να επιτύχουμε το διαμοιρασμό των πληροφοριών, οι πηγές οφείλουν πρώτα να συμφωνήσουν στον ορισμό του προτύπου που θα ακολουθούν οι Βάσεις Δεδομένων. Πρέπει λοιπόν να αναπτυχθούν τεχνικές οι οποίες θα επιτυγχάνουν την σύσταση προτύπων διατηρώντας την εμπιστευτικότητα των πληροφοριών.

- Εφόσον έχει επιτευχθεί η τυποποίηση, ένα σημαντικό κομμάτι της ενοποίησης των πληροφοριών είναι το ταίριασμα των αντικειμένων και η συνένωση των δεδομένων. Ουσιαστικά αποτελεί σύνηθες φαινόμενο σε διαφορετικές Βάσεις Δεδομένων να υπάρχουν αναφορές στο ίδιο αντικείμενο, οι οποίες πολλές φορές είναι δύσκολο να αναγνωριστούν. Η διασύνδεση των εγγραφών είναι μια διαδικασία που είναι σε θέση να οδηγήσει στην εξάλειψη των διπλότυπων από διαφορετικές πηγές. Ο στόχος λοιπόν είναι να επιτυγχάνεται η διασύνδεση των πληροφοριών κάνοντας χρήση τεχνικών κρυπτογράφησης και ανωνυμοποίησης, ώστε να μην αποκαλύπτεται η ταυτότητα των διπλότυπων εγγραφών.
- Τέλος οι χρήστες είναι πλέον σε θέση να θέσουν διάφορες επερωτήσεις στις Βάσεις Δεδομένων. Είναι λοιπόν ιδιαίτερα σημαντικό να υπάρχουν δικλείδες ασφαλείας που θα αποτρέπουν την διαρροή πληροφοριών. Προς την επίτευξη αυτού του στόχου έχουν ήδη αναπτυχθεί ορισμένες τεχνικές οι οποίες επεξεργάζονται τις επερωτήσεις που θέτει ο εκάστοτε χρήστης και του επιστρέφουν τα αποτελέσματα που χρειάζεται αποφεύγοντας την αποκάλυψη ευαίσθητων πληροφοριών.

1.2 Εφαρμογές με Χρήση Ενοποιημένων Δεδομένων

Εν συνεχεία γίνεται μια σύντομη περιγραφή διαφόρων εφαρμογών για τις οποίες κρίνεται επιτακτική η ανάγκη να έχουν πρόσβαση σε γνώσεις που προέρχονται από ενοποιημένες Βάσεις Δεδομένων. Μάλιστα θεωρείται ότι οι συγκεκριμένες εφαρμογές προκειμένου να προβούν στην ενοποίηση των πληροφοριών τους, πρέπει οπωσδήποτε να υπάρχει η εγγύηση της διατήρησης της εμπιστευτικότητας των πηγών των δεδομένων. Γενικότερα διαχωρίζονται σε δύο κατηγορίες:

1. Κοινή χρήση επιστημονικών ερευνητικών δεδομένων.
2. Συνεργασία ανταγωνιστικών επιχειρήσεων.

1.2.1 Χρήση Επιστημονικών Ερευνητικών Δεδομένων

Οι επιστήμονες ανά τον κόσμο προκειμένου να προχωρήσουν τις έρευνες τους χρησιμοποιούν όσες πληροφορίες και γνώσεις έχουν στη διάθεση τους. Θα ήταν λοιπόν ιδιαίτερα θεμιτό να υπήρχε η δυνατότητα ανταλλαγής γνώσεων και δεδομένων μεταξύ τους.

Ιδιαίτερα στον τομέα της ιατρικής, η κοινή χρήση δεδομένων μπορεί να προσφέρει σημαντικότατη βοήθεια. Καθίσταται δυνατή η ανταλλαγή πληροφοριών που αφορούν διάφορες ασθένειες, ούτως ώστε να είναι πιο εύκολη η κατανόηση και ο τρόπος αντιμετώπισης τους. Παράλληλα επιτυγχάνεται η παρακολούθηση και ο εντοπισμός ξεσπασμάτων ασθενειών σε παγκόσμια κλίμακα. Κατά αυτόν τον τρόπο οι ιατρικές υπηρεσίες και οι υπηρεσίες ασφαλείας ανά τον κόσμο θα έχουν την δυνατότητα να ανταλλάσσουν ιδέες και απόψεις για τις τεχνικές αντιμετώπισης των καταστάσεων που απειλούν την δημόσια υγεία και ασφάλεια.

Ουσιαστικά όμως οι ερευνητές προμηθεύονται τις παραπάνω πληροφορίες από διάφορους παροχείς ιατρικής περίθαλψης, όπως είναι τα νοσοκομεία, οι οποίοι τους δίνουν τα στοιχεία των νοσηλευόμενων. Είναι λοιπόν προφανές ότι στα στοιχεία αυτά συμπεριλαμβάνονται και ευαίσθητα δεδομένα των ασθενών. Επομένως η συλλογή των στοιχείων θα πρέπει να γίνεται με τέτοιο τρόπο ώστε να αποκρύπτεται η ταυτότητα του ασθενή. Πρέπει δηλαδή να αναπτυχθούν μέθοδοι οι οποίες θα αφαιρούν από τις εγγραφές τα προσωπικά και ευαίσθητα στοιχεία των ασθενών, και θα καθιστούν αδύνατη την αντιστοίχηση προσώπων με ασθένειες [24].

1.2.2 Συνεργασία Ανταγωνιστικών Επιχειρήσεων

Η συνεργασία μεταξύ επιχειρήσεων που δραστηριοποιούνται στον ίδιο τομέα μπορεί να τους αποφέρει αρκετά οφέλη και κέρδη. Παραδείγματος χάριν αν θεωρήσουμε δυο επιχειρήσεις οι οποίες μοιράζονται μεταξύ τους ορισμένα στατιστικά στοιχεία για την

αγορά, αλλά και για τον τρόπο παραγωγής των προϊόντων τους, τότε θα μπορούν όλες να αποκομίσουν κοινά οφέλη.

Όμως, προκειμένου να ισχύσουν όλα τα προηγούμενα θα πρέπει να υπάρχει αμοιβαία εμπιστοσύνη ανάμεσα στους εμπλεκόμενους φορείς. Η συνεργασία και η ανταλλαγή δεδομένων θα πρέπει να επιτυγχάνεται με τέτοιο τρόπο ώστε να προφυλάσσεται η ασφάλεια των εταιρειών [1]. Κανείς δεν πρέπει να είναι σε θέση να λάβει ανταγωνιστικές γνώσεις οι οποίες θα του δώσουν πλεονεκτήματα έναντι των υπολοίπων, ούτε να παραβιαστούν οι νόμοι που διέπουν την επιχειρηματικότητα. Εκτός των άλλων, ενδέχεται κάποιος τρίτος, που δεν συμμετέχει στα συνεργαζόμενα μέλη, να αποκτήσει πρόσβαση στις εμπιστευτικές πληροφορίες.

Εφόσον λοιπόν οι επιχειρήσεις θελήσουν να προβούν σε τέτοιου είδους συνεργασίες, θα πρέπει πρώτα να έχουν αναπτυχθεί ορισμένες τεχνικές οι οποίες θα προφυλάσσουν την εμπιστευτικότητα των δεδομένων τόσο από τα ίδια τα συνεργαζόμενα μέλη, όσο και από την παρείσφρηση τρίτων.

1.3 Κεντρική Ιδέα Εργασίας

Στην ενότητα 1.1 , έγινε μια μικρή αναφορά στα διάφορα θέματα τα οποία καλείται να αντιμετωπίσει κάποιος που θέλει να ασχοληθεί με το ζήτημα της διατήρησης της εμπιστευτικότητας κατά την ενοποίηση των δεδομένων από διαφορετικές πηγές. Η συγκεκριμένη έρευνα εστιάζεται κυρίως στην εξέταση τεχνικών που εφαρμόζουν την διαδικασία της διασύνδεσης εγγραφών για να εξαλειφθούν τα διπλότυπα. Ουσιαστικά εστιάζουμε στα εξής ζητήματα:

- Επιτυγχάνεται θεωρητική ανάλυση και πρακτική εξέταση των αλγορίθμων, ώστε να είναι κάποιος σε θέση να αποφανθεί εάν είναι δυνατή η διατήρηση της εμπιστευτικότητας κατά την ενοποίηση των δεδομένων.
- Στην κατεύθυνση αυτή εξετάζεται η τεχνική σύγκρισης της ομοιότητας των αλφαριθμητικών με χρήση N-γραμμάτων, ώστε να επιτευχθεί η διασύνδεση των εγγραφών των κατανεμημένων Βάσεων Δεδομένων.

- Παράλληλα εξετάζεται κατά πόσο η εφαρμογή τεχνικών κρυπτογράφησης στην διαδικασία αυτή, αποτρέπει την αποκάλυψη της ταυτότητας των εγγραφών.

1.4 Συνεισφορά της Εργασίας

Η εργασία υποκινήθηκε κυρίως από την ανάγκη να ερευνηθεί ένας τομέας των Βάσεων Δεδομένων η ανάπτυξη του οποίου βρίσκεται ακόμη σε πρώιμο στάδιο. Η συνεισφορά λοιπόν της συγκεκριμένης έρευνας είναι δυνατόν να αποτυπωθεί στα παρακάτω:

- **Γενική Ιδέα**: δίδεται μια γενική ιδέα για το τι ακριβώς είναι το ζήτημα της διατήρησης της εμπιστευτικότητας κατά την ενοποίηση δεδομένων. Παράλληλα γίνεται μια μικρή παρουσίαση των τομέων όπου θεωρείται κρίσιμη η εφαρμογή του παραπάνω ζητήματος, προκειμένου να έχουν εύρυθμη λειτουργία.
- **Λογισμικό Febrl:** παρουσιάζεται η χρήσιμη πλατφόρμα λογισμικού Febrl η οποία αποτελείται από μια βιβλιοθήκη προγραμμάτων που επιτυγχάνουν την τυποποίηση και διασύνδεση των εγγραφών. Διερευνώνται οι δυνατότητες του συγκεκριμένου λογισμικού, ο τρόπος λειτουργίας του, καθώς και πώς είναι δυνατή η επέκταση της χρήσης του σε κατανεμημένες Βάσεις Δεδομένων.
- **Τεχνική Ν-γραμμάτων**: υλοποιούνται προγράμματα τα οποία είναι σε θέση να επιτύχουν την διασύνδεση των εγγραφών μεταξύ δυο διαφορετικών Βάσεων Δεδομένων χρησιμοποιώντας την τεχνική των Ν-γραμμάτων. Εξετάζεται η λειτουργικότητα της συγκεκριμένης τεχνικής ανάλογα με το μέγεθος των προς σύγκριση Βάσεων Δεδομένων, καθώς και κατά πόσο τα αποτελέσματα που δίνει βοηθούν πραγματικά στην εύρεση των διπλότυπων εγγραφών.
- **Τεχνικές Κρυπτογράφησης**: γίνεται μια παρουσίαση ορισμένων τεχνικών κρυπτογράφησης, οι οποίες εφαρμόζονται στα παραπάνω προγράμματα ώστε να επιτυγχάνεται το ζήτημα της διατήρησης της εμπιστευτικότητας των δεδομένων. Ελέγχεται επίσης κατά πόσο επηρεάζουν την αποδοτικότητα των αλγορίθμων διασύνδεσης εγγραφών και αν προσφέρουν πλήρη προστασία των δεδομένων.

1.5 Δομή της Εργασίας

Η δομή της υπόλοιπης εργασίας έχει ως εξής.

- Στο δεύτερο κεφάλαιο περιγράφονται ορισμένες χρήσιμες βασικές έννοιες που εμφανίζονται στην εργασία. Οι έννοιες αυτές αφορούν τους τρόπους με τους οποίους επιτυγχάνεται η ενοποίηση των δεδομένων, καθώς και ορισμένους κρυπτογραφικούς ορισμούς.
- Στο τρίτο κεφάλαιο γίνεται η παρουσίαση του λογισμικού Febrl. Ουσιαστικά περιγράφονται οι εφαρμογές του που παρουσίασαν ενδιαφέρον κατά την διάρκεια της έρευνας, οι οποίες είναι η δημιουργία συνόλων δεδομένων, ο καθαρισμός και τυποποίηση τους, καθώς και η διασύνδεση των εγγραφών τους.
- Στο τέταρτο κεφάλαιο παρουσιάζεται η τεχνική των N-γραμμάτων που συνιστά στην σύγκριση των δεδομένων. Αρχικά παρουσιάζεται το πρωτόκολλο που ακολουθήθηκε προκειμένου να διατηρείται η εμπιστευτικότητα των δεδομένων και στην συνέχεια τα προγράμματα που υλοποιήθηκαν για να επιτύχουν αυτόν τον σκοπό.
- Στο πέμπτο κεφάλαιο γίνεται μια προσπάθεια αξιολόγησης της τεχνικής των N-γραμμάτων. Η αξιολόγηση γίνεται τόσο από την πλευρά της ποιότητας των αποτελεσμάτων που μας παρέχει η συγκεκριμένη τεχνική, όσο και από την πλευρά της απόδοσης της. Με τον όρο απόδοση εννοείται η ταχύτητα εκτέλεσης των εφαρμογών και οι απαιτήσεις που έχουν σε υπολογιστικούς πόρους.
- Τέλος στο έκτο κεφάλαιο δίνεται ο επίλογος της εργασίας, όπου παρουσιάζονται τα τελικά συμπεράσματα, καθώς και οι κατευθύνσεις που μπορεί να ακολουθήσει κάποιος που θέλει να επεκτείνει την συγκεκριμένη έρευνα.

Κεφάλαιο 2

Βασικές Έννοιες

Εν συνεχεία θα γίνει η επεξήγηση ορισμένων βασικών εννοιών που συναντώνται στην συγκεκριμένη έρευνα. Η κατανόηση αυτών των εννοιών κρίνεται απαραίτητη προκειμένου κάποιος να είναι σε θέση να κατανοήσει το ζήτημα που πραγματεύεται η εργασία. Αρχικά περιγράφονται οι έννοιες οι οποίες σχετίζονται με την ενοποίηση των δεδομένων αυτή κάθε αυτή. Δηλαδή γίνεται μια σύντομη παρουσίαση και επεξήγηση όλων των μεθόδων και τεχνικών που χρησιμοποιήθηκαν στην εργασία ώστε να επιτύχουμε την ενοποίηση. Κατόπιν εξετάζονται διάφορες έννοιες που αφορούν την διατήρηση της εμπιστευτικότητας. Κατά κύριο λόγο, αυτές αφορούν τον τομέα της κρυπτολογίας, καθώς και την περιγραφή μεθόδων κρυπτογράφησης.

2.1 Τεχνικές Ενοποίησης Δεδομένων

Γενικότερα η κοινότητα των Βάσεων Δεδομένων έχει πλέον στην διάθεση της αρκετούς αλγορίθμους και τεχνικές με τις οποίες μπορεί να επιτύχει την ενοποίηση των δεδομένων [6]. Όπως έχει προαναφερθεί στην εισαγωγή του κεφαλαίου 1, με τον όρο ενοποίηση εννοείται ουσιαστικά η συνάθροιση των εγγραφών των συνόλων δεδομένων ώστε να εντοπιστούν οι εγγραφές που αντιστοιχούν στην ίδια οντότητα, και στην συνέχεια να απαλειφθούν οι διπλοεγγραφές. Η συνένωση των εγγραφών ενδέχεται να αφορά είτε τις εγγραφές μόνο μιας Βάσης Δεδομένων είτε περισσοτέρων. Παρατηρούμε ότι η δεύτερη περίπτωση ουσιαστικά αφορά κατανεμημένα συστήματα, και παρουσιάζει το μεγαλύτερο ενδιαφέρον. Στην συνέχεια γίνεται μια συνοπτική επεξήγηση των τεχνικών που θα μας απασχολήσουν στην εργασία.

2.1.1 Διασύνδεση Εγγραφών

Η διασύνδεση των εγγραφών [3, 5, 10] ουσιαστικά είναι η διαδικασία της σύγκρισης εγγραφών ώστε να παρθεί η απόφαση κατά πόσο αυτές ταιριάζουν, υποστηρίζοντας ότι αναπαριστούν την ίδια οντότητα, ή δεν ταιριάζουν, υποστηρίζοντας ότι αναπαριστούν τελείως διαφορετικές οντότητες. Στην περίπτωση που δεν είναι δυνατόν το σύστημα διασύνδεσης εγγραφών να πάρει κάποια απόφαση, τότε είναι απαραίτητη η ανθρώπινη παρέμβαση ώστε να αποφασισθεί αν ταιριάζουν ή όχι.

Πολλές φορές στην βιβλιογραφία η διασύνδεση των εγγραφών απαντάται με διαφορετική ορολογία. Συχνά εμφανίζεται ως «ταίριασμα δεδομένων» ή «πρόβλημα αναγνώρισης αντικειμένου». Επίσης ονομάζεται «διαδικασία συγχώνευσης/ εκκαθάρισης» και «επικάλυψη λιστών» [4]. Όποιος από τους παραπάνω ορισμούς και να χρησιμοποιείται, αναφέρεται στην ίδια διαδικασία.

Η διασύνδεση εγγραφών χρησιμοποιείται για τους εξής λόγους [4]:

- Βελτιώνει την ποιότητα και την ακεραιότητα των δεδομένων.
- Επιτρέπει την επαναχρησιμοποίηση των ήδη υπαρχόντων πηγών δεδομένων προκειμένου να πραγματοποιηθούν νέες έρευνες.
- Μειώνει το κόστος και την προσπάθεια που καταβάλλεται κατά την απόκτηση των δεδομένων για έρευνες.

Η διασύνδεση των εγγραφών, ανάλογα με την περίσταση, μπορεί να είναι από απλοϊκή ως ιδιαίτερα πολύπλοκη. Στην περίπτωση όπου για κάθε οντότητα υφίσταται ένας μοναδικός αναγνωριστής που την κάνει να ξεχωρίζει από όλες τις υπόλοιπες οντότητες και ο οποίος θα είναι ίδιος σε όλα τα σύνολα δεδομένων, τότε η διαδικασία είναι πολύ απλή και περιορίζεται στην εφαρμογή κάποιας τεχνικής που κάνει συνένωση των συνόλων. Αντίθετα, εάν δεν υπάρχει κάποιος αναγνωριστής τότε πρέπει να χρησιμοποιηθούν ορισμένες τεχνικές. Ένας πιθανός διαχωρισμός των τεχνικών αυτών, ανάλογα με την μεθοδολογία που ακολουθούν, είναι σε ντετερμινιστικές και πιθανοτικές.

Οι ντετερμινιστικές μέθοδοι έχουν στην διάθεση τους μεγάλα σύνολα που απαρτίζονται από ιδιαίτερα πολύπλοκους κανόνες που ορίζουν πότε ταιριάζουν οι εγγραφές. Με βάση αυτούς τους κανόνες κατηγοριοποιούν ένα ζευγάρι εγγραφών σε ταίριασμα, αν αντιστοιχούν στην ίδια οντότητα, ή σε μη ταίριασμα αν αντιστοιχούν σε διαφορετικές οντότητες. Από την άλλη πλευρά οι πιθανοτικές μέθοδοι χρησιμοποιούν

στατιστικά μοντέλα για να αποφανθούν για την ομοιότητα των εγγραφών. Αυτές μπορούν αν διαιρεθούν σε δυο κατηγορίες. Στις μεθόδους που βασίζονται στην «κλασική» πιθανοτική θεωρία διασύνδεσης εγγραφών, όπως αυτή επεξηγήθηκε από τους Fellegi and Sunter [11], και σε αυτές που βασίζονται στην μέγιστη εντροπία και σε νεότερες τεχνικές με χρήση μηχανών εκμάθησης [18].

Ένας τρόπος εφαρμογής της διαδικασίας της διασύνδεσης εγγραφών στις μη απλοϊκές περιπτώσεις, περιγράφεται από τα παρακάτω τρία βήματα [4].

1. Κατασκευάζονται ένα ή περισσότερα ευρετήρια, συνήθως σε μορφή μπλοκ, ώστε να ομαδοποιηθούν οι εγγραφές που ενδεχομένως ταιριάζουν. Κατά αυτόν τον τρόπο επιτυγχάνεται μείωση του πλήθους των συγκρίσεων που πρέπει να πραγματοποιηθούν.
2. Μετά την δημιουργία του ευρετηρίου, λαμβάνει χώρα η σύγκριση των εγγραφών που ανήκουν στο ίδιο μπλοκ του ευρετηρίου. Η σύγκριση γίνεται πεδίο με πεδίο, με την χρήση ειδικών συναρτήσεων, και τελικά παράγεται ένα διάνυσμα βάρους για κάθε ζευγάρι εγγραφών.
3. Τέλος τα διανύσματα βάρους εξετάζονται από έναν κατηγοριοποιητή ο οποίος θα λάβει την απόφαση για το αν το ζεύγος των εγγραφών ταιριάζει ή όχι.

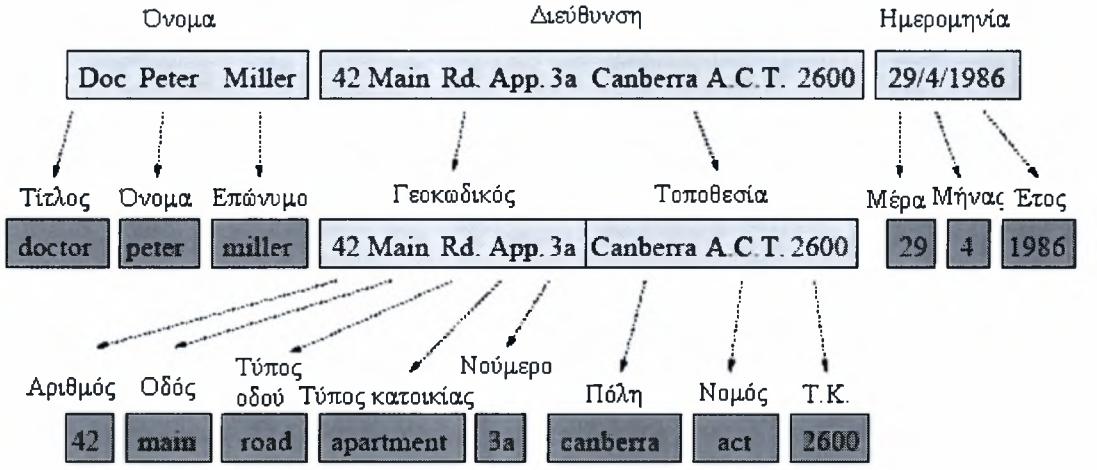
2.1.2 Καθαρισμός και Τυποποίηση Εγγραφών

Προτού ξεκινήσει η διαδικασία της διασύνδεσης των εγγραφών πρέπει πρώτα να επιτευχθεί ένας σημαντικότατος στόχος, σύμφωνα με τον οποίο όλες οι εγγραφές από όλα τα σύνολα που πρόκειται να υποβληθούν σε σύγκριση πρέπει να είναι στην τυποποιημένη μορφή. Ο σκοπός της διαδικασίας καθαρισμού και τυποποίησης εγγραφών [4] είναι να μετατρέψει την πληροφορία που είναι αποθηκευμένη στα πρωτότυπα δεδομένα σε μια καλά ορισμένη και συνεπή μορφή.

Οι προσωπικές πληροφορίες ενδέχεται να καταγράφονται σε αρκετούς διαφορετικούς τύπους, να είναι γραμμένες με άλλον τρόπο, να είναι απαρχαιωμένες, να έχουν απολέσει ορισμένα αντικείμενα ή να περιέχουν λάθη. Χαρακτηριστικό παράδειγμα αποτελεί η περίπτωση σε μια Βάση Δεδομένων να καταγράφονται στο ίδιο πεδίο το όνομα και το επώνυμο ενός ατόμου, ενώ σε μια άλλη σε δυο διαφορετικά πεδία. Η διαδικασία του καθαρισμού και της τυποποίησης των εγγραφών στοχεύει να

αντιμετωπίσει τέτοιου είδους προβλήματα. Διότι όταν όλα τα αρχικά δεδομένα είναι σε μια προκαθορισμένη τυποποιημένη μορφή, η διαδικασία διασύνδεσης εγγραφών έχει την δυνατότητα να είναι πολύ πιο ακριβής και αποτελεσματική.

Στο παρακάτω σχήμα φαίνεται ο τρόπος με τον οποίο η διαδικασία επιδρά σε μια εγγραφή. Παρατηρούμε ότι ενώ η εγγραφή είχε στην αρχή τρία συνολικά πεδία, μετατρέπεται κατάλληλα ώστε να έχει είτε οκτώ είτε δεκατέσσερα πεδία, ανάλογα με τις εκάστοτε απαιτήσεις. Κατά αυτόν τον τρόπο η συγκεκριμένη εγγραφή αποκτά τον ίδιο τύπο με μια άλλη που έχει περισσότερα πεδία, και παράλληλα, αυξάνοντας τα πεδία αυξάνεται και η ακρίβεια των αποτελεσμάτων που θα δώσει η διασύνδεση.



Σχήμα 2-1: Παράδειγμα Τυποποίησης Προσωπικών Πληροφοριών

2.1.3 Τεχνική N-Γραμμάττων

Η τεχνική των N-γραμμάτων[5,13] χρησιμοποιείται προκειμένου να επιτευχθεί η διασύνδεση των εγγραφών μεταξύ δύο ή περισσότερων Βάσεων Δεδομένων. Ουσιαστικά αποτελεί μια μέθοδο σύγκρισης της ομοιότητας μεταξύ δύο αλφαριθμητικών χαρακτήρων, ή ακόμη και μεταξύ ολόκληρων ακολουθιών τέτοιων χαρακτήρων [15]. Αυτό σημαίνει ότι η τεχνική αυτή είναι δυνατόν να χρησιμοποιηθεί τόσο για την σύγκριση των αλφαριθμητικών που περιέχονται ξεχωριστά σε κάθε πεδίο των εγγραφών, όσο και για την σύγκριση ολόκληρων κειμένων. Μάλιστα στην

περίπτωση όπου συγκρίνουμε κείμενα τότε η όλη διαδικασία ονομάζεται «ανάκτηση πληροφορίας».

Με τον όρο Ν-γράμμα εννοείται ότι ένα αλφαριθμητικό διαχωρίζεται σε ακολουθίες από Ν χαρακτήρες. Δηλαδή αν Ν=2, οπότε έχουμε ένα δι-γράμμα, το αλφαριθμητικό χωρίζεται σε ακολουθίες των δύο χαρακτήρων. Αντίστοιχα αν Ν=3 έχουμε ένα τρι-γράμμα που αποτελείται από ακολουθίες τριών χαρακτήρων. Για να γίνει το παραπάνω ευκολότερα κατανοητό θα γίνει διαχωρισμός σε δι-γράμματα της λέξεως 'Γιώργος'. Τότε υφίστανται τα εξής δι-γράμματα:

'Γιώργος' : ('γι', 'ιω', 'ωρ', 'ργ', 'γο', 'ος')

Αντίστοιχα για την περίπτωση που θέλουμε να χρησιμοποιήσουμε τα τρι-γράμματα τότε θα έχουμε:

'Γιώργος' : ('γιω', 'ιωρ', 'ωργ', 'ργο', 'γος')

Όπως φαίνεται από τα παραπάνω ένα Ν-γράμμα αποτελεί μια διανυσματική αναπαράσταση η οποία συμπεριλαμβάνει όλους τους συνδυασμούς Ν-γραμμάτων που εμφανίζονται στο αλφαριθμητικό. Ουσιαστικά κάθε διάνυσμα αποτελείται από μια διανυσματική συνιστώσα για κάθε πιθανό συνδυασμό Ν-γράμματος. Για παράδειγμα για ένα αλφάβητο που περιλαμβάνει τα γράμματα 'α' ως 'ω' και τους αριθμούς '0' ως '9' ο διανυσματικός χώρος θα είναι 34^N – διαστατός [13].

Η τεχνική των Ν-γραμμάτων ενδέχεται να εφαρμοσθεί με διάφορους τρόπους. Για παράδειγμα πολλές φορές στο διάνυσμα με τα Ν-γράμματα προστίθεται και ένα ακόμη διάνυσμα το οποίο περιλαμβάνει αριθμούς που αντιστοιχούν στο πόσες φορές εμφανίζεται το αντίστοιχο Ν-γράμμα μέσα στο αλφαριθμητικό. Στη συνέχεια της εργασίας θα επεξηγηθεί λεπτομερώς πώς ακριβώς χρησιμοποιείται στην προκείμενη έρευνα η τεχνική των Ν-γραμμάτων.

2.1.4 Φωνητικοί Αλγόριθμοι

Η σύγκριση της ομοιότητας των εγγραφών είναι δυνατόν να επιτευχθεί με την χρήση φωνητικών αλγορίθμων κωδικοποίησης [4]. Οι αλγόριθμοι αυτοί κωδικοποιούν μια ακολουθία αλφαριθμητικών και της προσθέτουν ένα συγκεκριμένο βάρος. Έπειτα βασιζόμενοι στο βάρος, κάνουν την σύγκριση των αλφαριθμητικών για να ελέγξουν

κατά πόσο ταιριάζουν ή όχι. Στη συνέχεια παρουσιάζονται ορισμένοι τέτοιοι αλγόριθμοι:

- **Soundex [18]:** Στην περίπτωση του λογισμικού Febrl ο Soundex διατηρεί τον πρώτο χαρακτήρα ενός αλφαριθμητικού ως έχει, και υπολογίζει έναν τριψήφιο κωδικό του υπολοίπου, και με βάση αυτά κάνει την σύγκριση των δεδομένων.
- **Nysiis [4]:** Σε αντίθεση με τον Soundex, υπολογίζει έναν κωδικό του αλφαριθμητικού ο οποίος δεν αποτελείται από ψηφία, αλλά μόνο χαρακτήρες, για να κάνει την σύγκριση.
- **Phonex [18]:** Αποτελεί μια παραλλαγή των δυο προηγούμενων η οποία προσπαθεί να βελτιώσει την ποιότητα της κωδικοποίησης, κάνοντας προηγουμένως επεξεργασία των αλφαριθμητικών. Η κωδικοποιημένη μορφή που προκύπτει αποτελείται από έναν χαρακτήρα αρχής, που ακολουθείται από έναν τριψήφιο αριθμό.
- **Double Metaphone [21]:** Ακολουθεί την ίδια ακριβώς διαδικασία με τον Nysiis, με την διαφορά ότι πετυχαίνει πιο σωστή ορθογράφηση μη αγγλικών λέξεων, όπως είναι οι ασιατικές και οι ευρωπαϊκές.

2.2 Κρυπτογραφικοί Ορισμοί

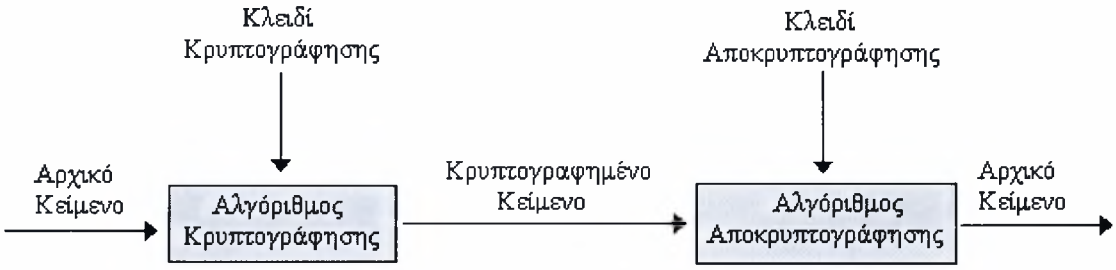
Σε αυτήν την ενότητα θα γίνει η επεξήγηση των κρυπτογραφικών μεθόδων που χρησιμοποιήθηκαν στην εργασία αυτή. Οι μέθοδοι αυτές ουσιαστικά συνέβαλαν στο τμήμα της εργασίας που είχε να κάνει με το ζήτημα της διατήρησης της εμπιστευτικότητας των πληροφοριών. Στις ενότητες 2.2.1 ως 2.2.3 θα δοθούν οι ορισμοί των μεθόδων που χρησιμοποιήθηκαν καθώς και λίγα λόγια για τον τρόπο λειτουργίας τους και την γενικότερη απόδοση τους. Στην ενότητα 4.2.2 της εργασίας θα εξεταστεί λεπτομερέστερα ο τρόπος που εφαρμόστηκαν και η απόδοση που είχαν όταν χρησιμοποιήθηκαν για τις ανάγκες της έρευνας.

2.2.1 Κρυπτογραφία

Με τον όρο κρυπτογραφία [19] εννοούμε τον επιστημονικό κλάδο ο οποίος ασχολείται με την μελέτη, τη χρήση και την ανάπτυξη τεχνικών κρυπτογράφησης και αποκρυπτογράφησης για την απόκρυψη των περιεχομένων των μηνυμάτων ή των δεδομένων, και την διευκόλυνση της ανίχνευσης κακόβουλων μετατροπών στα μηνύματα. Εν συνεχεία δίνεται η επεξήγηση των παραπάνω όρων:

- Η κρυπτογράφηση αποτελεί την διεργασία μετασχηματισμού ενός μηνύματος σε μια ακατανόητη μορφή με την χρήση ενός αλγορίθμου, ώστε αυτό να μην είναι δυνατόν να αναγνωριστεί από τρίτους εκτός από τους νόμιμους παραλήπτες.
- Η αποκρυπτογράφηση αποτελεί την διεργασία της ανάκτησης του αρχικού μηνύματος από την κρυπτογραφημένη έκδοση του που είχε παραχθεί, ώστε αυτό να επανέλθει σε κατανοητή μορφή.
- Ο κρυπτογραφικός αλγόριθμος είναι η μέθοδος μετασχηματισμού των δεδομένων σε μια μορφή που δεν επιτρέπει την αποκάλυψη τους σε τρίτους. Συνήθως ο αλγόριθμος είναι μια μαθηματική συνάρτηση.
- Το κλειδί κρυπτογράφησης είναι ένας αριθμός που χρησιμοποιείται μαζί με τον αλγόριθμο για να επιτευχθεί η κρυπτογράφηση ή η αποκρυπτογράφηση. Το μέγεθος του κλειδιού μετριέται σε bits, και ανάλογα με τον αλγόριθμο χρησιμοποιούνται κλειδιά διαφορετικών μεγεθών ώστε να επιτευχθεί το επιθυμητό επίπεδο ανθεκτικότητας.

Στο παρακάτω σχήμα απεικονίζεται ένα τυπικό σύστημα κρυπτογράφησης



Σχήμα 2-2: Τυπικό Σύστημα Κρυπτογράφησης

Γενικότερα υπάρχουν διάφοροι μέθοδοι κρυπτογραφίας. Ανάλογα με το είδος του κλειδιού που χρησιμοποιείται μπορούμε να τους διαχωρίσουμε σε μεθόδους «μυστικού ή συμμετρικού κλειδιού» και «δημοσίου ή ασύμμετρου κλειδιού». Στην πρώτη περίπτωση, η οποία εφαρμόσθηκε στην εργασία, γίνεται χρήση του ίδιου μυστικού κλειδιού τόσο για την κρυπτογράφηση όσο και για την αποκρυπτογράφηση. Ο λόγος που χρησιμοποιήθηκε η μέθοδος μυστικού κλειδιού είναι ότι προσφέρει υψηλότερο επίπεδο ασφάλειας από την μέθοδο συμμετρικού κλειδιού[19, 39].

Επίσης με βάση τον τρόπο με τον οποίο γίνεται η κρυπτογράφηση των μηνυμάτων, υπάρχουν οι μέθοδοι δέσμης και ροής. Οι μέθοδοι δέσμης μετατρέπουν το μήνυμα σε δέσμες – μπλοκ, για παράδειγμα των 128 ή 256 bits, τις οποίες κρυπτογραφούν με βάση κάποια συνάρτηση. Πρέπει να τονιστεί ότι η κρυπτογράφηση όλων των δεσμών για το ίδιο μήνυμα γίνεται με το ίδιο κλειδί. Για τον λόγο αυτό εφαρμόσθηκε μια μέθοδος δέσμης στην εργασία, ώστε να υπάρχει ένα μόνο κλειδί για όλα τα συμβαλλόμενα μέλη.

2.2.2 Κρυπτογραφικός Αλγόριθμος TwoFish

Στην εργασία, προκειμένου να επιτευχθεί η ασφάλεια των δεδομένων και η αυθεντικοποίηση των χρηστών, που υποτίθεται ότι ανταλλάσσουν τις πληροφορίες τους, χρησιμοποιήθηκε ο κρυπτογραφικός αλγόριθμος TwoFish [34]. Ο TwoFish αποτελεί έναν αλγόριθμο δέσμης που για την κρυπτογράφηση χρησιμοποιεί ένα συμμετρικό (μυστικό) κλειδί. Ο συγκεκριμένος αλγόριθμος σχεδιάστηκε στηριζόμενος στον κρυπτογραφικό αλγόριθμο AES [39] ο οποίος αποτελεί το νέο πρότυπο κρυπτογραφικού αλγορίθμου δέσμης. Ουσιαστικά ο AES, που σημαίνει Προχωρημένο Πρότυπο Κρυπτογράφησης (Advanced Encryption Standard), έχει επικυρωθεί από το NIST που αποτελεί το Ινστιτούτο Προτύπων και Τεχνολογίας των Η.Π.Α.. Αυτό δείχνει ότι ο AES, και κατά συνέπεια ο TwoFish, είναι ιδιαίτερα ανθεκτικός στις επιθέσεις και παρέχει υψηλότατο επίπεδο ασφάλειας [19, 34], και για αυτόν τον λόγο επιλέχθηκε να χρησιμοποιηθεί στην εργασία. Παρακάτω παρατίθενται τα βασικά χαρακτηριστικά του αλγορίθμου TwoFish:

- Είναι συμμετρικός αλγόριθμος δέσμης, με μέγεθος για κάθε μπλοκ 128 bits

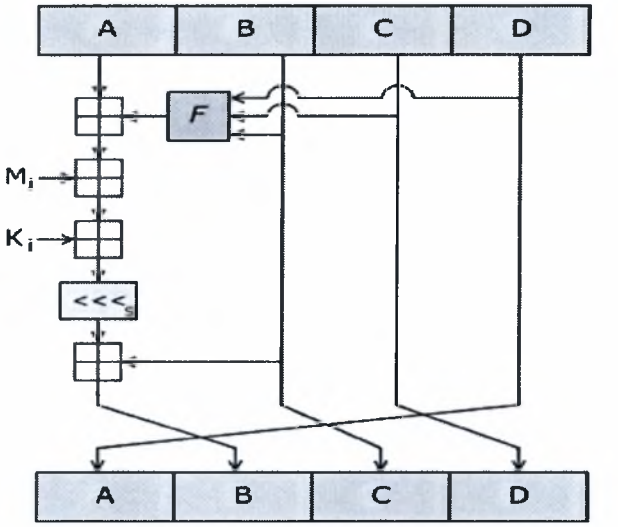
- Το κλειδί που χρησιμοποιεί μπορεί να έχει μέγεθος 128 bits, 192 bits και 256 bits.
- Σε περίπτωση που ο χρήστης που επιλέγει το μέγεθος του κλειδιού επιλέξει κάποιον άλλον αριθμό, τότε ο αλγόριθμος μόνος του το προσαρμόζει στο κοντινότερο δυνατό μέγεθος.
- Δεν χρησιμοποιούνται ποτέ αδύναμα κλειδιά.
- Η χρήση του υποστηρίζεται από πληθώρα λογισμικών προγραμμάτων, από διαφορετικά λειτουργικά συστήματα, καθώς και από διάφορα είδη τεχνολογίας επεξεργαστών.

2.2.3 Συνάρτηση Κατακερματισμού MD5

Για να επιτευχθεί ακόμη πιο υψηλό επίπεδο ασφάλειας και προστασίας των πληροφοριών χρησιμοποιήθηκε η μαθηματική συνάρτηση μονόδρομου κατακερματισμού MD5 [35, 40]. Η συνάρτηση, η οποία βασίζεται στο κρυπτογραφικό σύστημα RSA, δέχεται σαν είσοδο ένα μήνυμα οποιουδήποτε μεγέθους και στην συνέχεια παράγει σαν έξοδο μια κατακερματισμένη εκδοχή του αρχικού μηνύματος η οποία έχει μέγεθος 128 bits. Ουσιαστικά με τον όρο κατακερματισμένη, εννοείται ότι παράγεται μια σύνοψη του αρχικού μηνύματος που θα είναι πάντα 128 bits όσο μέγεθος και αν έχει το αρχικό μήνυμα.

Ο λόγος που χρησιμοποιείται η συγκεκριμένη συνάρτηση είναι για να μην έχει την δυνατότητα κάποιος τρίτος να παρέμβει στα ήδη κρυπτογραφημένα δεδομένα και να τα παραποιήσει. Αυτό βασίζεται στην έννοια της μονόδρομης συνάρτησης. Ουσιαστικά υπάρχει μια μαθηματική συνάρτηση η οποία εφαρμόζεται στο μήνυμα εισόδου και παράγει την σύνοψη του. Όμως δεν είναι δυνατόν να υπάρξει μια συνάρτηση που να εφαρμόζει την ακριβώς αντίστροφη διαδικασία, δηλαδή να δέχεται σαν είσοδο το κατακερματισμένο μήνυμα και να το επαναφέρει στην αρχική εκδοχή του. Επομένως η συγκεκριμένη διαδικασία κρίνεται αρκετά ασφαλής ως προς την μετάδοση μη παραποιημένων δεδομένων.

Στο παρακάτω σχήμα παρουσιάζεται μια διαδικασία της συνάρτησης του MD5:



Σχήμα 2-3: Μια Διαδικασία του αλγορίθμου MD5

Ο αλγόριθμος MD5 αποτελείται από 64 τέτοιες διαδικασίες, οι οποίες είναι οργανωμένες σε 4 γύρους των 16 διαδικασιών. Το F αποτελεί μια μη γραμμική συνάρτηση. Σε κάθε γύρο χρησιμοποιείται μια μόνο τέτοια συνάρτηση. Το M_i δηλώνει μια δέσμη μεγέθους 32 bits της αρχικής έκδοσης και το K_i μια σταθερά μεγέθους κι αυτή 32 bits, η οποία είναι διαφορετική για κάθε διαδικασία. Το σύμβολο $<<<_s$ δηλώνει μια μετατόπιση των bit προς τα αριστερά κατά 's' θέσεις, και μπορεί να παίρνει διαφορετικές τιμές ανάλογα με την διαδικασία. Τέλος το σύμβολο $\boxplus$ δηλώνει την πρόσθεση που γίνεται κάθε φορά, πάντα σε modulo 2^{32}.

Κεφάλαιο 3

Λογισμικό Febrl

Το πρωτότυπο λογισμικό Febrl [4, 33] έχει σχεδιαστεί προκειμένου να αναλαμβάνει τον καθαρισμό και την τυποποίηση εγγραφών, την εξεύρεση και διαγραφή των διπλότυπων σε ένα σύνολο δεδομένων, καθώς και την διασύνδεση των εγγραφών δυο ή περισσοτέρων Βάσεων Δεδομένων. Όλες οι παραπάνω εφαρμογές επιτυγχάνονται με την χρήση συγκεκριμένων πιθανοτικών μοντέλων. Αρχικά το Febrl σχεδιάστηκε για να χρησιμοποιηθεί από ερευνητές που ασχολούνται με βιοϊατρικά δεδομένα, παρόλα αυτά είναι δυνατόν να εφαρμοστεί και σε οποιαδήποτε άλλη εφαρμογή. Αυτό φαίνεται και από την ονομασία του, που αποτελεί τα αρχικά των λέξεων «Ελευθέρως Επεκτάσιμη Διασύνδεση Βιοϊατρικών Εγγραφών».

Στο παρόν κεφάλαιο γίνεται μια παρουσίαση και επεξήγηση των δυνατοτήτων που παρέχει το Febrl στους ερευνητές, συγκεκριμένα όσες εφαρμογές παρουσίασαν ενδιαφέρον στην εργασία. Όλες οι εφαρμογές του λογισμικού είναι υλοποιημένες στην γλώσσα προγραμματισμού 'Python' [36]. Αξίζει να τονισθεί, ότι ένα σημαντικό πλεονέκτημα του λογισμικού είναι ότι ένας χρήστης έχει την δυνατότητα να επέμβει στον πηγαίο κώδικα των προγραμμάτων του, ή ακόμη να προσθέσει δικά του προγράμματα, ώστε να μεταβάλει την χρησιμότητα του ανάλογα με τις απαιτήσεις του. Επιγραμματικά αναφέρονται οι εφαρμογές του Febrl οι οποίες μας απασχόλησαν:

- Δημιουργία συνόλων δεδομένων με τυχαίες εγγραφές, για την εξέταση της λειτουργικότητας και αποτελεσματικότητας των προγραμμάτων.
- Καθαρισμός και Τυποποίηση δεδομένων με χρήση πιθανοτικών μοντέλων και κανόνων απόφασης.
- Εξάλειψη διπλότυπων και διασύνδεση εγγραφών με χρήση πιθανοτικών μοντέλων και ενός ειδικού κατηγοριοποιητή.

3.1 Δημιουργία Συνόλων Δεδομένων

Ένα σημαντικό πλεονέκτημα του Febrl είναι ότι παρέχει στους χρήστες την δυνατότητα να δημιουργήσουν τα δικά τους σύνολα δεδομένων ανάλογα με τις απαιτήσεις τους. Ως γνωστόν οι διαδικασίες της εξάλειψης των διπλότυπων και της διασύνδεσης των εγγραφών, προκειμένου να παράγουν ορατά αποτελέσματα και να εξεταστεί η αποτελεσματικότητα τους, πρέπει να εφαρμοστούν σε δεδομένα για τα οποία τα αποτελέσματα τις διασύνδεσης τους είναι γνωστά εκ των προτέρων. Δυστυχώς λόγο της εμπιστευτικότητας και της ασφάλειας των πληροφοριών, δεν υπάρχουν διαθέσιμα στο ευρύ κοινό πραγματικά σύνολα δεδομένων που να έχουν αυτή την ιδιότητα. Επομένως δεν είναι εφικτός ο αυτόματος έλεγχος των αλγορίθμων και των διαδικασιών της διασύνδεσης εγγραφών, εκτός αν είναι δυνατή η παραγωγή τεχνητών συνόλων δεδομένων.

Το λογισμικό Febrl παρέχει την δυνατότητα παραγωγής τεχνητών συνόλων δεδομένων. Το πρόγραμμα που επιτελεί αυτή τη διαδικασία είναι το 'generate.py'. Η συγκεκριμένη εφαρμογή είναι σε θέση να δημιουργήσει σύνολα δεδομένων που περιλαμβάνουν στα πεδία τους ονόματα, διευθύνσεις, ημερομηνίες, τηλεφωνικούς αριθμούς και προσωπικούς αριθμούς ταυτοποίησης όπως είναι το Α.Φ.Μ.. Όλα τα προηγούμενα είναι διαθέσιμα σε διάφορους πίνακες που έχει το Febrl και προέρχονται από την ανάμιξη πραγματικών δεδομένων από τηλεφωνικούς καταλόγους. Επί προσθέτως, για έναν μεγάλο αριθμό ονομάτων υπάρχουν διαθέσιμες πάρα πολλές διαφορετικές εκδοχές προφοράς τους, όπως επίσης και συνήθη τυπογραφικά λάθη που συμβαίνουν.

Η δημιουργία των συνόλων δεδομένων επιτυγχάνεται σε δυο στάδια. Στο πρώτο δημιουργούνται οι πρωτότυπες εγγραφές, ενώ στο δεύτερο δημιουργούνται οι διπλότυπες τους με βάση τυχαίες αλλαγές που υφίστανται οι πρωτότυπες. Κάθε εγγραφή που δημιουργείται χαρακτηρίζεται από ένα μοναδικό αναγνωριστικό, από το οποίο φαίνεται αν είναι πρωτότυπη ή όχι. Αν είναι διπλότυπη φαίνεται ότι είναι το αντίγραφο μιας συγκεκριμένης πρωτότυπης εγγραφής, με βάση το γεγονός ότι έχουν τον ίδιο κωδικό, απλά στην πρωτότυπη έχει την κατάληξη 'org' από το 'original', ενώ στην διπλότυπη έχει κατάληξη 'dup' από το 'duplicate'. Η εφαρμογή δίνει την

δυνατότητα στον χρήστη να ορίσει μόνος του τις παραμέτρους της δημιουργίας του συνόλου δεδομένων. Αυτές φαίνονται παρακάτω:

- Ο χρήστης ελέγχει τον αριθμό των λαθών που συμβαίνουν ανά πεδίο ανά εγγραφή.
- Ορίζει τον αριθμό των πρωτότυπων εγγραφών, καθώς και αυτόν των διπλότυπων.
- Καθορίζει πόσες διπλότυπες εγγραφές θα παραχθούν για κάθε μια πρωτότυπη.
- Ορίζει το πιθανοτικό μοντέλο που θα χρησιμοποιηθεί για τα λάθη που θα εισαχθούν στις διπλότυπες εγγραφές.

Τα παραπάνω δίνονται σαν παράμετροι κάθε φορά που ο χρήστης εκτελεί το πρόγραμμα 'generate.py'. Για παράδειγμα όταν δώσει στο command line την εντολή:

'python generate.py set1.csv 500 500 1 1 1 poisson'

τότε θα δημιουργηθεί το σύνολο δεδομένων set1.csv που θα έχει 1000 εγγραφές, από τις οποίες οι 500 θα είναι διπλότυπες, ακριβώς μια για κάθε πρωτότυπη. Κάθε διπλότυπη εγγραφή θα έχει μια αλλαγή σε κάθε ένα πεδίο της, και θα χρησιμοποιηθεί το πιθανοτικό μοντέλο Poisson [4].

Αφού οριστούν οι παραπάνω παράμετροι από τον χρήστη, τότε δημιουργούνται τα δεδομένα. Τα λάθη που εμφανίζονται στις διπλότυπες εγγραφές γίνονται τυχαία από το πρόγραμμα με βάση τα παρακάτω:

- Αν για το πρωτότυπο αλφαριθμητικό υπάρχει κάποια άλλη εκδοχή διαφορετικής προφοράς, τότε χρησιμοποιείται αυτή.
- Εισάγεται ένας νέος χαρακτήρας σε κάποιο τυχαίο σημείο του αλφαριθμητικού.
- Διαγράφεται ένας τυχαίος χαρακτήρας.
- Αντικαθίσταται ένας τυχαίος χαρακτήρας με κάποιον άλλον.
- Αντικαθίσταται ολόκληρο το αλφαριθμητικό από κάποιο άλλο.
- Γίνεται ανταλλαγή θέσεων μεταξύ δυο τυχαίων χαρακτήρων.
- Εισάγεται ένα κενό στο πεδίο, ώστε να διασπαστεί το αλφαριθμητικό.
- Αντίστοιχα διαγράφεται ένα κενό, αν υπάρχει.
- Ορίζεται ένα πεδίο της εγγραφής να είναι κενό.

- Αν ένα πεδίο είναι κενό, τότε εισάγεται μια νέα τιμή σε αυτό.
- Ανταλλάσσονται οι τιμές δυο διαφορετικών πεδίων στην ίδια εγγραφή.

Τα αρχεία συνόλων δεδομένων που παράγονται είναι σε μορφή αρχείου '.csv'. Τα αρχεία αυτά ουσιαστικά είναι αρχεία κειμένου που οι τιμές τους διαχωρίζονται με ένα κόμμα (,) και είναι προσβάσιμα τόσο από έναν επεξεργαστή κειμένου όπως είναι η εφαρμογή 'WordPad' τόσο και από την εφαρμογή 'Excel' με την οποία διαφαίνεται πιο καθαρά ο διαχωρισμός των εγγραφών και των πεδίων.

Για τις ανάγκες της εργασίας χρησιμοποιήθηκαν σύνολα δεδομένων που παράχθηκαν από την συγκεκριμένη εφαρμογή. Στην εικόνα που παρουσιάζεται στην συνέχεια φαίνεται ένα μέρος ενός συνόλου με εγγραφές που χρησιμοποιήθηκε. Παρατηρούμε ότι κάθε εγγραφή έχει δεκατέσσερα πεδία. Αυτά είναι ο κωδικός της εγγραφής, το όνομα, το επώνυμο, ο αριθμός της διεύθυνσης, η διεύθυνση, μια δεύτερη διεύθυνση, η περιοχή, το ταχυδρομικός κώδικας, η πολιτεία, η ημερομηνία γέννησης, η ηλικία, ο τηλεφωνικός αριθμός, ο αριθμός ασφαλίσεως και άλλος ένας αναγνωριστικός αριθμός. Συνολικά στην εικόνα παρουσιάζονται δεκαπέντε εγγραφές.

rec_id	given_name	surname	street_num	address_1	address_2	suburb	postcode	state	date_of_birth	age	phone_numb	soc_sec_	blocking_r
rec-17-dup-0	zane	tallanrpda	9	van raalte place	colooli village	alice springs	6442	tas		33	07 12605642	6256444	3
rec-2-org	matthew	carbone	53	jinka street		birkdale	4802	sa	19600428	31	03 29741455	3391176	0
rec-5-dup-0	nicholas	mercornlla	82	fitchett street	tatonga	toorak	2560	qld	19061026	22	02 43222792	8111523	1
rec-18-org	mitchell	spark	375	gingana street		ballarat	5000	nsw	19150402	31	04 53279081	6584750	3
rec-9-org	desi	webb	7	morrison street		warilla	850	nt	19860830	30	04 28564183	1047636	6
rec-5-org	nicholas	mercorella	84	fitchett street	taronga	toorak	2560	qld	19061026	22	02 43222792	8111523	1
rec-7-org	lucas	nguyen	14	clive steele avenue		penneshaw	3044	nsw			02 70491925	9293749	6
rec-0-org	eliza	wilkins	8	melrose drive		duncraig	5582	vic	19241002	30	02 70809290	7882097	5
rec-12-dup-0	grgurovic	jscob	3	dumas street	flt 61	namvor	2747	vic	19141004	29	02 43559497	3622021	3
rec-3-dup-0	madelyn	macirnn	3			croydon north	3034	vic	19970126	33	07 06911506	1891909	3
rec-0-dup-0	eliza	wilkins	8	melrosehdvive	mount flagstone	duncraig	6582	vic	19245992	30	02 70809290	7882097	5
rec-7-dup-0	luc	nguyen	1			penneskhaw	3044				02 70491925	9293749	6
rec-6-org	ayden	mcelhinney	26	oxley street	glen arthur	tallai	2360	qld	19341118	27	07 09750426	5081160	6
rec-13-dup-0	logan	sribaf	91	maloney street	kimbe		3021	tas	19177041		02 07964279	6715716	6
rec-11-org	dylan	dolan	7	wisdom street		raby	3150	qld	19610913		07 17284241	8636877	8

Σχήμα 3-1: Παράδειγμα συνόλου δεδομένων της εφαρμογής 'generate.py'

Τα δεδομένα του συνόλου προέρχονται από τηλεφωνικούς καταλόγους της Αυστραλίας από όπου προέρχονται οι δημιουργοί του Febrl, καθώς και από το G-NAF [20, 42], μια

Βάση Δεδομένων της Αυστραλίας που βασίζεται σε Γεωγραφικά Συστήματα Πληροφοριών 'GIS' [20], για την εύρεση διευθύνσεων.

3.2 Εφαρμογές Καθαρισμού και Τυποποίησης

Όπως έχει προαναφερθεί στην ενότητα 2.1.2, για να επιτευχθεί η σύγκριση των εγγραφών των Βάσεων Δεδομένων, πρέπει αυτές να είναι στην τυποποιημένη μορφή. Το λογισμικό Febrl παρέχει την δυνατότητα του καθαρισμού και της τυποποίησης των εγγραφών μέσω της εφαρμογής 'standardization.py' [33] η οποία χρησιμοποιεί και ορισμένες ρουτίνες των εφαρμογών 'address.py', 'date.py', 'name.py' και 'phonenum.py'. Οι ρουτίνες των παραπάνω εφαρμογών περιέχουν συναρτήσεις που συνιστούν στον καθαρισμό και την τυποποίηση των αλφαριθμητικών στα οποία αντιστοιχούν. Δηλαδή για παράδειγμα η 'address.py' στις διευθύνσεις και η 'phonenum.py' στους τηλεφωνικούς αριθμούς. Όπως φαίνεται και από την ονομασία των εφαρμογών, το Febrl είναι σε θέση να επεξεργαστεί ονόματα, διευθύνσεις, ημερομηνίες και τηλεφωνικούς αριθμούς.

Προκειμένου να γίνει η επεξεργασία των ονομάτων και των διευθύνσεων ώστε να έρθουν σε τυποποιημένη μορφή ακολουθούνται τα εξής τρία βήματα [4]:

- Καθαρισμός
- Ανάθεση ετικετών
- Τμηματοποίηση

3.2.1 Βήμα 1: Καθαρισμός

Το πρώτο βήμα είναι να γίνει ο καθαρισμός των δεδομένων. Το πρόγραμμα δέχεται σαν είσοδο μια ακολουθία αλφαριθμητικών που μπορεί να είναι είτε ένα όνομα είτε μια διεύθυνση. Αρχικά όλα τα γράμματα στην ακολουθία μετατρέπονται σε πεζά, και στη συνέχεια χρησιμοποιείται μια λίστα επιδιορθώσεως, με την οποία ελέγχονται ξεχωριστά τα αλφαριθμητικά της ακολουθίας ώστε ορισμένα από αυτά να αντικατασταθούν με άλλα που είναι σε κανονική μορφή. Για παράδειγμα αν σε ένα πεδίο εμφανιζόταν η λέξη «ωσγνωστόν» τότε αυτή θα αντικαθιστούταν από την σωστή

φράση «ως γνωστόν». Γίνεται κατανοητό ότι επιδιορθώνονται λέξεις και εκφράσεις που είναι δυνατόν να εντοπιστούν οι σωστές εκδοχές τους στην λίστα επιδιορθώσεως. Οι καταχωρήσεις που έχει η λίστα αποτελούνται από λέξεις ή φράσεις που ενδέχεται να εντοπιστούν στα δεδομένα που ελέγχονται, και τις αντίστοιχες τους που πρόκειται αν τις αντικαταστήσουν. Στην παρακάτω εικόνα φαίνεται ένα παράδειγμα της λίστας επιδιορθώσεως με αγγλικές εκφράσεις:

Original	Replacement
' knownas '	' known as '
' a.k.a. '	' known as '
' aka '	' known as '
' babyof '	' baby of '
' b/o '	' baby of '
' b.o. '	' baby of '
' n/a '	' '
' na '	' '
'['	'—'
'('	'—'
':'	' '

Σχήμα 3-2: Παράδειγμα Λίστας Επιδιορθώσεως

Είναι προφανές ότι η έξοδος του προγράμματος καθαρισμού θα είναι ένα μια τελείως νέα ακολουθία με αλφαριθμητικά, στην οποία κάθε αλφαριθμητικό που βρέθηκε στη λίστα επιδιορθώσεως έχει αντικατασταθεί.

3.2.2 Βήμα 2: Ανάθεση Ετικετών

Αφού γίνει ο καθαρισμός των αλφαριθμητικών που περιλαμβάνονται σε ένα πεδίο, το επόμενο βήμα είναι να τα διαχωρίσουμε σε μια λίστα που περιέχει ονόματα, αριθμούς και πιθανά διαχωριστικά. Για παράδειγμα αν ένα πεδίο έχει την φράση "doctor peter paul miller", τότε αυτή διαχωρίζεται στη λίστα ['doctor', 'peter', 'paul', 'miller']. Κατόπιν σε κάθε στοιχείο της λίστας ανατίθεται μια ετικέτα ανάλογα με το τι αντιπροσωπεύει. Δηλαδή ορίζεται ότι το συγκεκριμένο στοιχείο είναι κόμμα, τελεία ή παύλα. Στην περίπτωση των λέξεων ανατίθεται ετικέτα που ορίζει αν είναι όνομα,

επώνυμο, τίτλος, ταχυδρομικός κώδικας, χώρα κ.ο.κ.. Στον παρακάτω πίνακα φαίνονται ορισμένοι τύποι στοιχείων και οι αντίστοιχες ετικέτες τους.

Τύπος Στοιχείου	Ετικέτα
Επώνυμο	SN
Αντρικό όνομα	GM
Γυναικείο όνομα	GF
Τίτλος	TI
Χώρα	CR
Περιοχή	LN
Ταχυδρομικός Κώδικας	PC
Αριθμός	NU
Στοιχείο με γράμματα και αριθμούς	AN
Άγνωστο στοιχείο	UN
Κόμμα	CO

Πίνακας 3-1: Αντιστοίχηση Στοιχείων με Ετικέτες

Πρέπει να σημειωθεί ότι είναι πιθανό ένα στοιχείο να αντιστοιχεί σε περισσότερες από μια ετικέτες, οπότε του ανατίθενται όλες όσες του αντιστοιχούν.

Το αποτέλεσμα που δίνει η ρουτίνα που επεξεργάζεται τα καθαρισμένα αλφαριθμητικά είναι μια λίστα με στοιχεία και μια λίστα με τις αντίστοιχες ετικέτες τους. Για παράδειγμα για την φράση που αναφέρθηκε στην αρχή της παραγράφου το αποτέλεσμα θα ήταν:

Λίστα στοιχείων: [‘doctor’, ‘peter’, ‘paul’, ‘miller’]

Λίστα ετικετών: [‘TI’, ‘GM/SN’, ‘GM’, ‘SN’]

Όπως φαίνεται το στοιχείο ‘peter’ έχει δυο ετικέτες διότι ενδέχεται να είναι είτε αντρικό όνομα είτε επώνυμο.

3.2.3 Βήμα 3: Τμηματοποίηση

Στο τελευταίο βήμα γίνεται ο διαχωρισμός των στοιχείων σε πεδία, αν χρειάζεται, ώστε να ολοκληρωθεί η διαδικασία της τυποποίησης. Η ρουτίνα που κάνει την τμηματοποίηση δέχεται σαν είσοδο την λίστα με τα στοιχεία και την λίστα με τις ετικέτες τους. Τότε ελέγχει την ετικέτα κάθε στοιχείου και το τοποθετεί στο αντίστοιχο πεδίο, ώστε κάθε εγγραφή να πάρει την τυποποιημένη μορφή. Για παράδειγμα στο σχήμα 3-1 οι εγγραφές είναι σε τυποποιημένη μορφή. Επομένως ένα στοιχείο που έχει ετικέτα SN θα τοποθετηθεί στο τρίτο πεδίο (surname), που περιέχει τα επώνυμα.

3.3 Απαλοιφή Διπλότυπων και Διασύνδεση Εγγραφών

Όπως έχει αναφερθεί στην ενότητα 2.1.1 η διαδικασία της διασύνδεσης των εγγραφών απαιτεί αρχικά την ομαδοποίηση των εγγραφών σε ευρετήρια, την σύγκριση τους ώστε να παραχθεί ένα διάνυσμα βάρους, και την εξέταση του διανύσματος από έναν κατηγοριοποιητή ο οποίος θα αποφανθεί για το αν ταιριάζουν οι εγγραφές. Το λογισμικό Febrl έχει υλοποιημένες τις κατάλληλες διαδικασίες για να φέρει σε πέρας την παραπάνω διαδικασία.

3.3.1 Εφαρμογή Ευρετηριοποίησης

Το πρώτο βήμα είναι η οργάνωση των εγγραφών σε ευρετήρια ώστε να μειωθούν αισθητά οι μεταξύ τους συγκρίσεις. Το Febrl διαθέτει αρκετές μεθόδους δημιουργίας ευρετηρίων, οι οποίες είναι υλοποιημένες στην εφαρμογή 'indexing.py' [33]. Είναι ιδιαίτερα σημαντικό να τονισθεί ότι όταν γίνεται η σύγκριση των εγγραφών δύο διαφορετικών συνόλων δεδομένων, πρέπει η δημιουργία των ευρετηρίων και των δύο συνόλων να γίνει με την ίδια ακριβώς μέθοδο, αλλιώς θα καταστεί αδύνατη η σύγκριση τους.

Η πιο απλή μέθοδος ευρετηριοποίησης είναι η 'direct' [33] η οποία απλά θεωρεί τις τιμές των πεδίων ως τις μεταβλητές του ευρετηρίου. Μια άλλη μέθοδος είναι η 'truncate' [33] η οποία με βάση έναν αριθμό που ορίζεται ως παράμετρος, οργανώνει

τα αλφαριθμητικά των πεδίων σε ευρετήρια, σπάζοντας αυτά που ξεπερνούν σε μήκος τον αριθμό. Εκτός αυτών, υπάρχουν και οι μέθοδοι που βασίζονται στην κωδικοποίηση των τιμών των πεδίων με βάση έναν φωνητικό αλγόριθμο, ο οποίος αφού κωδικοποιήσει τις τιμές στη συνέχεια τις οργανώνει σε ευρετήρια. Οι φωνητικοί αλγόριθμοι που χρησιμοποιούνται είναι οι 'Soundex', 'Phonex', 'Nyssis' και 'Double-Metaphone'.

Οποιαδήποτε μέθοδος δημιουργίας των ευρετηρίων εφαρμοσθεί, θα χρησιμοποιήσει μια από τις δύο διαφορετικές υπάρχουσες δομές.

- **Μπλοκ Ευρετηριοποίηση [10, 29, 30]:** Στην πρώτη περίπτωση οι εγγραφές οργανώνονται σε μπλοκ. Στο ίδιο μπλοκ συμπεριλαμβάνονται εγγραφές που έχουν την ίδια τιμή σε κάποιο πεδίο η οποία προηγουμένως έχει οριστεί ως μεταβλητή μπλοκ. Στην συνέχεια η σύγκριση γίνεται μόνο μεταξύ των εγγραφών που ανήκουν στο ίδιο μπλοκ. Για παράδειγμα μπορεί ένας χρήστης να ορίσει να ομαδοποιούνται στο ίδιο μπλοκ όλες οι εγγραφές οι οποίες εμφανίζουν σε κάποιο πεδίο τους το αλφαριθμητικό "john".
- **Ταξινομημένη Ευρετηριοποίηση [14]:** Η περίπτωση αυτή αποτελεί μια βελτιωμένη εκδοχή της προηγούμενης. Τα μπλοκ που έχουν δημιουργηθεί ταξινομούνται αλφαβητικά με βάση την καθορισμένη μεταβλητή μπλοκ. Στην συνέχεια ένα μετατοπιζόμενο παράθυρο μετακινείται μεταξύ των μπλοκ και η σύγκριση γίνεται ανάμεσα στις εγγραφές που βρίσκονται στα μπλοκ του παραθύρου. Το μέγεθος του παραθύρου, δηλαδή πόσο μπλοκ θα περιλαμβάνει, καθορίζεται κάθε φορά από τον χρήστη. Η λογική αυτής της δομής βασίζεται στο γεγονός ότι οι εγγραφές που ανήκουν σε κοντινά μπλοκ ενδέχεται να ταιριάζουν. Παρακάτω παρουσιάζεται ένα παράδειγμα όπου μέσα στα μπλοκ φαίνεται ο υποτιθέμενος κωδικός της εγγραφής. Συνολικά υπάρχουν τέσσερα μπλοκ τα οποία ταξινομούνται ως εξής:

  a123: [4, 12, 25, 32]
  a129: [1, 15, 18, 24]
  b111: [3, 23]
  b125: [6, 27, 30]

  Αν ορισθεί το μέγεθος του παραθύρου ως τρία, τότε θα γίνει η σύγκριση των εγγραφών που ανήκουν στα μπλοκ:

[a123, a129, b111]: [1, 3, 4, 12, 15, 18, 23, 24, 25, 32]
[a129, b111, b125]: [1, 3, 6, 15, 18, 23, 24, 27, 30]

3.3.2 Εφαρμογή Σύγκρισης Εγγραφών

Αφού οργανωθούν οι εγγραφές σε ευρετήρια, τότε γίνεται η σύγκριση τους, διαδικασία που αποτελεί το σημαντικότερο κομμάτι της διασύνδεσης εγγραφών. Το Febrl διαθέτει αρκετές συναρτήσεις σύγκρισης των εγγραφών οι οποίες είναι υλοποιημένες στην εφαρμογή 'comparison.py' [33]. Αυτές ελέγχουν ξεχωριστά τα πεδία των εγγραφών και ανάλογα με την συνάρτηση, μπορούν να συγκρίνουν αλφαριθμητικά, αριθμούς, ημερομηνίες, ηλικίες και χρονικές περιόδους.

Κάθε συνάρτηση συγκρίνει τις εγγραφές και παράγει τα διανύσματα βάρους τα οποία ουσιαστικά περιλαμβάνουν τιμές που δείχνουν κατά πόσο δυο εγγραφές ταιριάζουν ή όχι. Τα διανύσματα βάρους παράγονται από την χρήση ορισμένων πιθανοτικών μοντέλων τα οποία δίνουν κάποια ποσοστά (τιμές από 0 ως 1) ομοιότητας των εγγραφών. Παρακάτω αναφέρονται ενδεικτικά ορισμένες από τις συναρτήσεις που είναι υλοποιημένες στο λογισμικό:

- Υπολογισμός βάρους βασισμένος στην συχνότητα του πεδίου.
- Ακριβής σύγκριση ολόκληρης της τιμής του πεδίου ή τμημάτων της.
- Προσεγγιστική σύγκριση της τιμής του πεδίου.
- Σύγκριση κωδικοποιημένης τιμής πεδίου.
- Αριθμητική σύγκριση με ποσοστιαία ανθεκτικότητα.
- Αριθμητική σύγκριση με απόλυτη ανθεκτικότητα.

3.3.3 Κατηγοριοποίηση

Το τελευταίο στάδιο της διασύνδεσης εγγραφών είναι η εξέταση των διανυσμάτων βάρους από έναν κατηγοριοποιητή ο οποίος θα αποφανθεί για την ομοιότητα τους. Το λογισμικό Febrl χρησιμοποιεί δύο είδη κατηγοριοποιητών για να επιτύχει αυτόν τον σκοπό.

- **Κατηγοριοποιητής Fellegi & Sunter [11]:** Αποτελεί έναν ιδιαίτερα απλοϊκό κατηγοριοποιητή ο οποίος αποτιμά όλα τα βάρη που απαρτίζουν ένα διάνυσμα βάρους ενός ζεύγους εγγραφών προσθέτοντας τα, και χρησιμοποιώντας δυο κατώφλια, αποφασίζει αν οι εγγραφές ταιριάζουν, δεν ταιριάζουν ή πρέπει να παρεμβληθεί ο ανθρώπινος παράγοντας για να αποφασίσει. Όπως γίνεται κατανοητό όταν η αποτίμηση των βαρών είναι κάτω από το μικρότερο κατώφλι τότε οι εγγραφές δεν ταιριάζουν. Όταν είναι πάνω από το μεγαλύτερο κατώφλι τότε ταιριάζουν, ενώ αν είναι ανάμεσα στα δύο κατώφλια τότε δεν μπορεί να αποφανθεί οπότε χρειάζεται η ανθρώπινη παρέμβαση.
- **Εύκαμπτος Κατηγοριοποιητής [4]:** Η διαφορά του από τον προηγούμενο είναι ότι χρησιμοποιεί διάφορες μεθόδους για την αποτίμηση του συνολικού βάρους ενός ζεύγους εγγραφών. Ενδέχεται από ένα διάνυσμα βάρους να εξεταστεί μόνο το βάρος με την ελάχιστη ή μέγιστη τιμή, το άθροισμα όλων ή μερικών βαρών, το γινόμενο τους, ή ο μέσος όρος τους. Παράλληλα είναι δυνατόν να γίνει συνδυασμός των παραπάνω μεθόδων. Δηλαδή για παράδειγμα στην τελική αποτίμηση να συμπεριληφθεί το άθροισμα των μισών βαρών και ο μέσος όρος των υπολοίπων. Παρόμοια με τον τρόπο λειτουργίας του προηγούμενου, ο κατηγοριοποιητής βασίζεται στην χρήση δύο κατωφλίων για να πάρει την τελική απόφαση ομοιότητας.

Κεφάλαιο 4

Εφαρμογή της Τεχνικής Σύγκρισης Εμπιστευτικών Δεδομένων

Σε αυτό το κεφάλαιο θα γίνει η παρουσίαση των τεχνικών μεθόδων που υλοποιήθηκαν στα πλαίσια αυτής της εργασίας προκειμένου να εφαρμοστεί η τεχνική σύγκρισης της ομοιότητας των εγγραφών διαφορετικών Βάσεων Δεδομένων, διατηρώντας παράλληλα την εμπιστευτικότητα των δεδομένων. Αρχικά θα γίνει η περιγραφή του πρωτοκόλλου πάνω στο οποίο βασίστηκε η υλοποίηση των προγραμμάτων. Εν συνεχεία θα λάβει χώρα η πλήρης επεξήγηση των προγραμμάτων που υλοποιήθηκαν, η μεθοδολογία που ακολουθήθηκε, καθώς και ο τρόπος με τον οποίο συνδέονται τα προγράμματα μεταξύ τους.

Στο σημείο αυτό αξίζει να τονισθεί ότι οι P. Ravikumar et al [22] έχουν αναπτύξει ένα δικό τους πρωτόκολλο για την σύγκριση της ομοιότητας εμπιστευτικών δεδομένων, το οποίο βασίζεται, από την πλευρά της ασφάλειας, στις ίδιες κρυπτογραφικές αρχές με το πρωτόκολλο που περιγράφεται στην αμέσως επόμενη παράγραφο. Η διαφορά τους έγκειται στο γεγονός ότι χρησιμοποιούν στοχαστικές κλιμακωτές διεργασίες για τον υπολογισμό μετρικών αποστάσεων, όπως είναι η Ευκλείδεια απόσταση [7 ,8, 22], προκειμένου να υπολογίσουν την ομοιότητα των δεδομένων. Όμως το συγκεκριμένο πρωτόκολλο εφαρμόζεται για την ανάκτηση πληροφοριών [8], και όχι για την διασύνδεση των εγγραφών δυο Βάσεων Δεδομένων αυτή κάθε αυτή. Αυτό σημαίνει ότι μπορεί να διαβάσει ένα κείμενο, και να ανακαλύψει, με βάση των υπολογισμό των μετρικών, πόσες φορές εμφανίζεται στο κείμενο ένα συγκεκριμένο αλφαριθμητικό.

4.1 Πρωτόκολλο Σύγκρισης Εγγραφών

Η ενότητα αυτή παρουσιάζει το πρωτόκολλο σύμφωνα με τους T. Churches et al [5], με βάση το οποίο δυο ή και περισσότερες Βάσεις Δεδομένων μπορούν να προβούν σε σύγκριση της ομοιότητας των δεδομένων τους, με την βοήθεια μιας τρίτης έμπιστης πηγής.

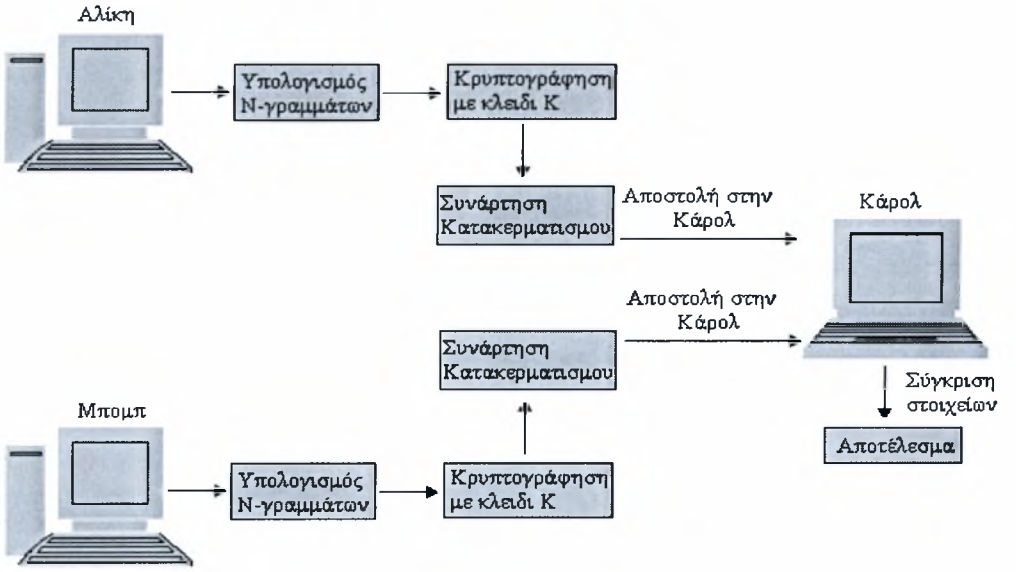
Θεωρείται ότι υπάρχει η Αλίκη η οποία έχει στην κατοχή της μια Βάση Δεδομένων Α. Η Α περιέχει μια ή περισσότερες εγγραφές r_1, r_2 κ.ο.κ., κάθε μια από τις οποίες έχει κάποιον αριθμό από πεδία $r_{1.1}$, $r_{1.2}$ κ.ο.κ., που συμπεριλαμβάνουν εμπιστευτικά αλφαριθμητικά όπως είναι ονόματα και διευθύνσεις. Η ανάγκη για εμπιστευτικότητα οφείλεται στο γεγονός ότι τα περιεχόμενα των πεδίων ενδέχεται να οδηγούν στην αναγνώριση προσώπων ή στην αποκάλυψη εμπορικών ανταγωνιστικών πληροφοριών. Επίσης θεωρείται ότι υπάρχει ο Μπομπ ο οποίος έχει στην κατοχή του μια παρόμοια με της Αλίκης Βάση Δεδομένων, που αντίστοιχα περιέχει εμπιστευτικά πεδία εγγραφών όπως τα $rec_{1.1}$, $rec_{1.2}$.

Ο στόχος της Αλίκης και του Μπομπ είναι να προσδιορίσουν κατά πόσο οι πληροφορίες που περιέχονται στα $r_{1.1}$ ταιριάζουν με αυτές των $rec_{1.1}$, χωρίς να αποκαλύπτουν ο ένας στον άλλον ποιες είναι οι πραγματικές πληροφορίες. Για τον λόγο αυτό υπάρχει η τρίτη έμπιστη πηγή, η οποία θα ονομάζεται Κάρολ. Θεωρείται δεδομένο ότι η Κάρολ είναι υπεύθυνη ώστε:

- Να ακολουθήσει επακριβώς το πρωτόκολλο.
- Να μην αποκαλύψει πληροφορίες σε άλλες πηγές εκτός από όπου επιτρέπεται με βάση το πρωτόκολλο.
- Να μην προσπαθήσει να προσδιορίσει τις πηγαίες πληροφορίες της Αλίκης και του Μπομπ.

Σε καμία περίπτωση δεν θεωρείται δεδομένο ότι η Αλίκη και ο Μπομπ εμπιστεύονται ο ένας τον άλλον. Τέλος πρέπει να αναφερθεί ότι για όλες τις μεταφορές δεδομένων μεταξύ των τριών μελών έχει συμφωνηθεί να χρησιμοποιηθεί κρυπτογράφηση συμμετρικού κλειδιού ώστε να επιτυγχάνεται η ασφάλεια των δεδομένων και η αυθεντικοποίηση των μελών.

Προτού γίνει η παρουσίαση του πρωτοκόλλου, παρατίθεται το παρακάτω σχήμα στο οποίο διαφαίνεται συνοπτικά η όλη διαδικασία που ακολουθείται.



Σχήμα 4-1: Αναπαράσταση Πρωτοκόλλου Σύγκρισης Εγγραφών

Στη συνέχεια επεξήγονται τα βήματα του πρωτοκόλλου όπως τα ορίζουν οι T. Churches et al [5]:

1. Αρχικά η Αλίκη και ο Μπομπ αποφασίζουν από κοινού για το ποιο θα είναι το μυστικό κλειδί κρυπτογράφησης, το οποίο θα γνωρίζουν μόνον αυτοί οι δυο, και καθορίζουν ποιον συγκεκριμένο αλγόριθμο κρυπτογράφησης θα εφαρμόσουν στα δεδομένα τους. Επίσης ορίζουν την μονόδρομη μαθηματική συνάρτηση κατακερματισμού, όπως είναι η MD5, που πρόκειται να χρησιμοποιήσουν. Παράλληλα συμφωνούν να εφαρμόσουν στα δεδομένα τους ένα τυποποιημένο πρωτόκολλο καθαρισμού των αλφαριθμητικών, ώστε οι πληροφορίες που περιέχονται στις εγγραφές τους να είναι στην ίδια τυποποιημένη μορφή.
2. Έπειτα η Αλίκη υπολογίζει μια ταξινομημένη λίστα με N-γράμματα για κάθε αλφαριθμητικό που περιέχεται στα πεδία $rec_{1.1}$. Αξίζει να σημειωθεί ότι τα διπλότυπα N-γράμματα απαλείφονται, οπότε στην λίστα κάθε ένα εμφανίζεται μια και μοναδική φορά. Κατόπιν η Αλίκη υπολογίζει όλες τις πιθανές υπό-

λίστες με μήκος μεγαλύτερο το μηδενός, για κάθε λίστα Ν-γραμμάτων. Ουσιαστικά υπολογίζει το δυναμό-σύνολο των Ν-γραμμάτων αφαιρώντας το κενό σύνολο. Για παράδειγμα αν ένα πεδίο περιείχε το αλφαριθμητικό 'Peter', και ήθελε να υπολογίσει τα δι-γράμματα τότε θα παραγόταν η λίστα δι-γραμμάτων ("er", "et", "pe", "te"), και το δυναμό-σύνολο:

("er"), ("et"), ("pe"), ("te"),

("er", "et"), ("er", "pe"), ("er", "te"), ("et", "pe"), ("et", "te"), ("pe", "te"),

("er", "et", "pe"), ("er", "et", "te"), ("er", "pe", "te"), ("et", "pe", "te"),

("er", "et", "pe", "te")

Παρατηρούμε ότι αφού η λίστα περιέχει 4 δι-γράμματα τότε το μήκος των υπό-λιστών δι-γραμμάτων που υπολογίζονται κυμαίνεται από 1 έως 4. Εκτός των άλλων, συμπεραίνουμε ότι αν μια λίστα Ν-γραμμάτων περιέχει 'a' Ν-γράμματα, τότε ο συνολικός αριθμός των υπό-λιστών που θα απαρτίζουν το δυναμό-σύνολο θα είναι $2^a - 1$. Έπειτα η Αλίκη κρυπτογραφεί κάθε μια από τις υπολογισμένες υπό-λίστες Ν-γραμμάτων, χρησιμοποιώντας το προκαθορισμένο κλειδί και αλγόριθμο, και συνοψίζει το αποτέλεσμα εφαρμόζοντας την μονόδρομη συνάρτηση κατακερματισμού. Παράλληλα δημιουργεί μια κρυπτογραφημένη έκδοση του κλειδιού αναγνώρισης κάθε εγγραφής. Στη συνέχεια στέλνει στην Κάρολ αυτά που υπολόγισε προηγουμένως.

3. Ομοίως με την Αλίκη, ο Μπομπ φέρνει σε πέρας την ίδια διαδικασία που περιγράφηκε στο βήμα 2, για τις δικές του εγγραφές, και στέλνει τα αποτελέσματα στην Κάρολ.
4. Τέλος η Κάρολ, αφού έχει στην διάθεση της τα αποτελέσματα και των δυο, καθορίζει την τομή των συνόλων των τιμών των κρυπτογραφημένων Ν-γραμμάτων που της έχουν αποσταλεί. Κατόπιν για κάθε τιμή των Ν-γραμμάτων που ανήκουν στην τομή, για κάθε μοναδικό ζευγάρι των κλειδιών αναγνώρισης των εγγραφών, υπολογίζει την βαθμολογία σύγκρισης Ν-γραμμάτων. Η βαθμολογία δίνεται από τον τύπο:

$$\left|\text{βαθμολογία} = \frac{\text{αριθμός\_κοινών\_N - γραμμάτων}}{0.5 * \left(\text{αριθμός\_N - γραμμάτων\_στο\_r}_{1,1} + \text{αριθμός\_N - γραμμάτων\_στο\_rec}_{1,1}\right)}\right|$$

Στη συνέχεια για κάθε μοναδικό ζευγάρι κλειδιών αναγνώρισης εγγραφών, από όλες τις βαθμολογίες που έχουν υπολογιστεί επιλέγει την μέγιστη. Είναι προφανές ότι οι τιμές της βαθμολογίας κυμαίνονται από 0 ως 1, με 1 να σημαίνει ότι οι κρυπτογραφημένες τιμές ταιριάζουν επακριβώς.

Αφού έχουν ολοκληρωθεί και τα τέσσερα βήματα του πρωτοκόλλου, ουσιαστικά έχουν παραχθεί τα αποτελέσματα της σύγκρισης. Τότε η Κάρολ θα παραδώσει τις βαθμολογίες μαζί με τους κωδικούς αναγνώρισης εγγραφών στην Αλίκη και τον Μπομπ, οι οποίοι στην συνέχεια θα αποφασίσουν σε ποιες ενέργειες θα προβούν. Ενδεχομένως, αντί να τους δώσει όλα τα αποτελέσματα, μπορεί να έχει οριστεί από πριν ένα κατώφλι βαθμολογίας, και να τους παραθέτει όλες τις βαθμολογίες που το ξεπερνούν, ώστε να αποφεύγεται η μετάδοση μεγάλου όγκου πληροφοριών. Αξίζει να τονισθεί ότι λόγω της κρυπτογράφησης, ο όγκος των δεδομένων που μεταδίδονται καθόλη την διάρκεια της διαδικασίας είναι πολύ μεγαλύτερος από ότι θα ήταν.

4.2 Επεξήγηση Προγραμμάτων

Στην ενότητα αυτή παρουσιάζονται τα προγράμματα που υλοποιήθηκαν έτσι ώστε να ολοκληρωθεί η εφαρμογή που βασίζεται στο πρωτόκολλο που περιγράφθηκε προηγουμένως. Όλα τα προγράμματα είναι γραμμένα στην γλώσσα προγραμματισμού Perl [37]. Ο λόγος που επιλέχθηκε αυτή η γλώσσα προγραμματισμού αντί κάποιας άλλης όπως οι ευρέως χρησιμοποιούμενες C και Java, είναι ότι η σύνταξη της είναι απλούστερη και, όσον αφορά τα προγράμματα που επρόκειτο να υλοποιηθούν, είναι πιο πρακτική και λειτουργική. Διότι η Perl έχει την δυνατότητα να επιτυγχάνει τη σύγκριση δύο ή περισσότερων αλφαριθμητικών με χρήση απλών εντολών, σε αντίθεση με τις άλλες γλώσσες προγραμματισμού που χρειάζεται να υλοποιηθεί πολύπλοκος κώδικας. Για παράδειγμα με χρήση μόνο της εντολής 'split' χωρίζει μια ακολουθία αλφαριθμητικών κάθε φορά που διαβάζει το στοιχείο που δόθηκε ως παράμετρος με την εντολή, και στη συνέχεια μπορούμε να συγκρίνουμε τα διαχωρισμένα αλφαριθμητικά. Συνολικά για τις ανάγκες της εργασίας υλοποιήθηκαν τέσσερα προγράμματα:

1. **Πρόγραμμα create_ngrams** : Υπολογίζει τα N-γράμματα για κάθε πεδίο των εγγραφών μιας Βάσης Δεδομένων.
2. **Πρόγραμμα encr_ngrams** : Κρυπτογραφεί τα N-γράμματα που έχουν παραχθεί από το προηγούμενο πρόγραμμα
3. **Πρόγραμμα sort_encrypted** : Κάνει την ταξινόμηση των κρυπτογραφημένων N-γραμμάτων.
4. **Πρόγραμμα comp_ngrams** : Δέχεται σαν είσοδο δυο αρχεία με N-γράμματα και κάνει την σύγκριση τους.

Εκτός αυτών των προγραμμάτων, χρησιμοποιήθηκε και μια βιβλιοθήκη προγραμμάτων [38] που έχει υλοποιηθεί για την Perl και μπορεί να την χρησιμοποιήσει ελεύθερα κάθε χρήστης.

4.2.1 Πρόγραμμα Υπολογισμού N-Γραμμάτων

Η εφαρμογή χρησιμοποιεί αρχικά το πρόγραμμα 'create_ngrams.plx'. Το συγκεκριμένο πρόγραμμα διαβάζει ένα αρχείο και παράγει τελικά ένα αρχείο με τα N-γράμματα όλων των πεδίων του αρχείου που διάβασε, εκτός του πεδίου που περιέχει τον κωδικό της εκάστοτε εγγραφής. Τα αρχεία εισόδου που χρησιμοποιήθηκαν είναι τα 'inputA.txt' και 'inputB.txt'. Αυτά δημιουργήθηκαν χρησιμοποιώντας την εφαρμογή 'generate.py' του λογισμικού Febrl, και οι περισσότερες εγγραφές τους είναι παρεμφερείς ώστε στη συνέχεια να γίνει η σύγκριση τόσο των κοινών στοιχείων, όσο και των μη κοινών. Αντιστοίχως, μετά το πέρας της εκτέλεσης του προγράμματος εξάγονται τα αρχεία 'outngramA.txt' και 'outngramB.txt'.

Το αρχείο εισόδου αποτελείται από έναν κωδικό της εγγραφής, δηλαδή το κλειδί αναγνώρισης της, και διάφορα πεδία τα οποία χωρίζονται μεταξύ τους με κόμμα ελληνικό (,). Ενδεικτικά ορισμένα από αυτά περιλαμβάνουν ονόματα, διευθύνσεις, τηλέφωνα. Για όλα τα πεδία κάθε εγγραφής, εκτός από αυτό που περιλαμβάνει τον κωδικό, δημιουργείται ένα ενδιάμεσο αρχείο που ονομάζεται 'output.txt'. Το αρχείο αυτό περιέχει εγγραφές με στήλες τον κωδικό, ένα ερωτηματικό (;), τον αύξοντα αριθμό (α/α) του πεδίου, ένα κενό και το πεδίο. Δηλαδή για κάθε πεδίο δημιουργείται μία εγγραφή. Επομένως για r εγγραφές και n πεδία δημιουργούνται rxn εγγραφές. Από

κάθε τέτοια εγγραφή κατόπιν παράγονται τα Ν-γράμματα του πεδίου στο αρχείο outngram. Στον παρακάτω πίνακα διακρίνεται ένα τμήμα του αρχείου 'output.txt' το οποίο αναφέρεται στα πεδία μιας εγγραφής.

Κωδικός	;	α/α	Τιμή Πεδίου
r-5	;	1	nicholas
r-5	;	2	mercorella
r-5	;	3	84
r-5	;	4	fitchettstreet
r-5	;	5	taronga
r-5	;	6	toorak
r-5	;	7	2560
r-5	;	8	qld
r-5	;	9	19061026
r-5	;	10	22
r-5	;	11	0243222792
r-5	;	12	8111523
r-5	;	13	1

Πίνακας 4-1: Παράδειγμα μορφής αρχείου output.txt

Το πρόγραμμα παίρνει σαν παραμέτρους το όνομα του αρχείου εισόδου, το όνομα του αρχείου εξόδου και το μήκος των Ν-γραμμάτων, δηλαδή τον αριθμό Ν. Στην αρχή ανοίγει το αρχείο εισόδου και διαβάζει μία μία τις εγγραφές του. Τότε τις καθαρίζει από τα κενά, μετατρέπει όλους τους χαρακτήρες των πεδίων σε πεζά ή κεφαλαία (πεζά στην περίπτωση μας) και χρησιμοποιώντας ως χαρακτήρα οριοθέτησης το κόμμα (,) σπάει τα πεδία του σε ένα πίνακα. Από αυτόν τον πίνακα γράφονται στο ενδιάμεσο αρχείο 'output' οι εγγραφές με την μορφή: 'κωδικός ; α/α πεδίο'.

Κατόπιν κλείνουν τα αρχεία και πλέον ανοίγουν σαν είσοδος το ενδιάμεσο αρχείο που έχει δημιουργηθεί και σαν έξοδος το 'outngram'. Δημιουργούνται τα Ν-γράμματα

του πεδίου σε πίνακα, γίνεται η ταξινόμηση τους και διαγράφονται τα πολλαπλά έτσι ώστε κάθε ένα να εμφανίζεται μια ακριβώς φορά όπως προαναφέρθηκε στο δεύτερο βήμα του πρωτοκόλλου που επεξηγείται στην ενότητα 4.1. Έπειτα δημιουργείται ένας πίνακας κατακερματισμού που περιέχει N-γράμματα για κλειδιά και αριθμούς 1,2,3,... αντίστοιχα για τιμές. Καλείται η υπό-ρουτίνα 'powerset_iterate' η οποία έχει ως παράμετρο μια μεταβλητή αναφοράς στον πίνακα κατακερματισμού. Αυτή κάνοντας χρήση αναδρομής δημιουργεί μια δομή υποσυνόλων των αντίστοιχων N-γραμμάτων, δηλαδή το δυναμό-σύνολο όπως έχει προαναφερθεί στο δεύτερο βήμα του πρωτοκόλλου που επεξηγείται στην ενότητα 4.1. Μετά καλείται η υπό-ρουτίνα 'sos_as_string' η οποία πρόκειται να δημιουργήσει σαν αλφαριθμητικά τις υπολίστες των N-γραμμάτων, εκτός του κενού συνόλου.

Όλες οι παραπάνω υπολίστες γράφονται στο αρχείο 'outngram' με τη μορφή: 'κωδικός ; α/α N-γράμμα'. Τα πεδία που είναι κενά και αυτά που ο αριθμός των χαρακτήρων τους είναι μικρότερος ή ίσος με τον αριθμό N που έχουμε δηλώσει στην αρχή, γράφονται όπως είναι στην εγγραφή. Δηλαδή αυτό που γίνεται είναι να μεταφέρεται αυτούσιο το πεδίο. Στον παρακάτω πίνακα παρουσιάζεται ένα παράδειγμα. Θεωρείται ότι το πεδίο 2 του κωδικού r.256 είναι 'George'. Τότε παράγονται τα ακόλουθα N-γράμματα : ('ge', 'eo', 'or', 'rg', 'ge'), και μετά την ταξινόμηση και την δημιουργία των υπολιστών έχουμε τις παρακάτω τελικές εγγραφές στο αρχείο outngram:

κωδικός	;	α/α	N-γράμμα
r-256	;	2	eo
r-256	;	2	eoge
r-256	;	2	eogeor
r-256	;	2	eogeorrg
r-256	;	2	eogerg
r-256	;	2	eoor
r-256	;	2	eoorrg
r-256	;	2	eorg
r-256	;	2	ge
r-256	;	2	geor
r-256	;	2	georrg
r-256	;	2	gerg
r-256	;	2	or
r-256	;	2	orrg
r-256	;	2	rg

Πίνακας 4-2: Παράδειγμα μορφής αρχείου outngram

Τέλος προκειμένου να γίνει η σύγκριση των αρχείων με τα N-γράμματα των δύο αρχείων που χρησιμοποιήθηκαν σαν είσοδοι, δημιουργούνται και δύο αρχεία κλειδιών που περιέχουν μόνο τους κωδικούς σε ταξινομημένη σειρά. Παράλληλα τα αρχεία εξόδου output που έχουν τα N-γράμματα είναι ταξινομημένα κατά κωδικό, α/α πεδίου και το ίδιο το N-γράμμα. Οι ταξινομήσεις αυτές γίνονται με βάση τα αλφαριθμητικά. Στον παρακάτω πίνακα φαίνεται τμήμα του αρχείου με τα ταξινομημένα κλειδιά του συνόλου δεδομένων input1.csv.

rec-0-org
rec-1-org
rec-10-org
rec-11-org
rec-12-org
rec-13-org
rec-14-org
rec-15-org
rec-16-org
rec-17-org

Πίνακας 5-3: Παράδειγμα μορφής αρχείου skeyinput1.csv

4.2.2 Πρόγραμμα Κρυπτογράφησης N-Γραμμάτων

Αφού έχουν παραχθεί τα αρχεία που περιλαμβάνουν τα N-γράμματα, η εφαρμογή συνεχίζει με την κρυπτογράφηση τους. Το πρόγραμμα της κρυπτογράφησης, το 'encr_ngrams', χρησιμοποιεί κρυπτογράφηση με υποδομή συμμετρικού κλειδιού και κατόπιν την μονόδρομη μαθηματική συνάρτηση κατακερματισμού.

Όπως έχει προαναφερθεί στην ενότητα 2.2.2, εφαρμόστηκε ο αλγόριθμος κρυπτογράφησης TwoFish. Συγκεκριμένα χρησιμοποιήθηκε το module Crypt::Twofish, που υλοποιήθηκε από τον Nishant Kakani [38], το οποίο αποτελεί μια Perl επέκταση του αλγόριθμου και υποστηρίζει τις παρακάτω μεθόδους:

1. **Κρυπτογράφηση**: Δέχεται σαν είσοδο το κλειδί, το μήκος του κλειδιού και το κείμενο προς κρυπτογράφηση και παράγει την κρυπτογραφημένη εκδοχή.
2. **Αποκρυπτογράφηση**: Δέχεται σαν είσοδο το κλειδί, το μήκος του κλειδιού, το κρυπτογραφημένο κείμενο και το μήκος του, και τελικά επαναφέρει το αρχικό κείμενο.
3. **Λάθος**: Σε περίπτωση που συμβεί λάθος κατά την διάρκεια μιας διαδικασίας μας δείχνει το τελευταίο λάθος που έγινε.

Με τον όρο Perl επέκταση, εννοείται ότι τα συγκεκριμένα modules αποτελούν ένα είδος βιβλιοθήκης προγραμμάτων τα οποία αποθηκεύονται στην Perl και ο χρήστης έχει την δυνατότητα να υλοποιήσει δικά του προγράμματα, όπως έγινε στην παρούσα εργασία, στα οποία θα καλεί και θα χρησιμοποιεί τις συναρτήσεις των προγραμμάτων της βιβλιοθήκης τα οποία έχουν υλοποιηθεί από άλλους.
Από τις συναρτήσεις αυτές στο πρόγραμμα χρησιμοποιήθηκε μόνον η πρώτη.

Ο αλγόριθμος της μαθηματικής συνάρτησης μονόδρομου κατακερματισμού που εφαρμόστηκε είναι ο MD5. Το module που χρησιμοποιήθηκε είναι το Digest::MD5 το οποίο αποτελεί επίσης μια Perl επέκταση του MD5, και υλοποιήθηκε από τον Gisle Aas [38]. Από τις διάφορες συναρτήσεις που έχει, για την υλοποίηση του προγράμματος χρησιμοποιήθηκε η md5_hex. Η μέθοδος αυτή επιστρέφει μια σύνοψη του μηνύματος που δέχεται σαν είσοδο σε δεκαεξαδική φόρμα. Το μήκος του επιστρεφόμενου μηνύματος, που ουσιαστικά είναι κάθε N-γράμμα, είναι 32 και περιέχει χαρακτήρες από το σύνολο '0' έως '9' και 'a' έως 'f'.

Το πρόγραμμα διαβάζει το αρχείο με τα N-γράμματα και το διαχωρίζει με ερωτηματικά (;) και κενά ώστε να πάρει ξεχωριστά κάθε N-γράμμα. Κάθε N-γράμμα αποτελεί το κείμενο εισόδου προς κρυπτογράφηση που δίνεται σε μια υπορουτίνα cryptr. Αυτή παίρνει σαν παραμέτρους το κλειδί, το μήκος του κλειδιού και το N-γράμμα. Στο σημείο αυτό καλείται η συνάρτηση κρυπτογράφησης του module της TwoFish, που ονομάζεται Encipher($key,$keylength,$plaintext) και κατόπιν η συνάρτηση md5_hex($ciphertext) του module της MD5. Μετά επιστρέφει την σύνοψη, που αναφέρεται ως $digest, η οποία γράφεται σε ένα αρχείο κρυπτογραφημένων δεδομένων με τον κωδικό και το πεδίο σε κάθε εγγραφή. Παραδείγματος χάριν η μορφή του θα είναι:

'r_{224};1 284a37d79dbdd91e48e0df99ab8ff50a'

Το πρόγραμμα λόγω της κρυπτογράφησης είναι ιδιαίτερα απαιτητικό σε μνήμη, αφού η Perl χρησιμοποιεί διερμηνευτή και όχι μεταγλωττιστή. Για τον λόγο αυτό η ταξινόμηση γίνεται σε άλλο πρόγραμμα, που ονομάζεται 'sort_encr_ngrams.plx'. Αυτό συμβαίνει γιατί όπως έχει προαναφερθεί στην ενότητα 4.2.1, το πρόγραμμα σύγκρισης θέλει τα πεδία ταξινομημένα.

4.2.3 Πρόγραμμα Σύγκρισης Ν-Γραμμάτων

Το τελευταίο κομμάτι της εφαρμογής είναι η σύγκριση των κρυπτογραφημένων Ν-γραμμάτων που έχουν παραχθεί. Αυτό επιτυγχάνεται με το πρόγραμμα 'comp_ngrams.plx', το οποίο προφανώς είναι σε θέση να συγκρίνει και μη κρυπτογραφημένα Ν-γράμματα.

Το πρόγραμμα ξεκινά με το διάβασμα των αρχείων με τους κωδικούς. Ουσιαστικά αυτό που κάνει είναι να εφαρμόζει όλους τους δυνατούς συνδυασμούς κωδικών που παίρνει από τα δύο αρχεία. Έπειτα με οδηγό κάθε ζεύγος κωδικών συγκρίνει τα αρχεία που περιέχουν τα Ν-γράμματα. Συγκεκριμένα παίρνει σαν παραμέτρους (command line arguments) τα δύο αρχεία προς σύγκριση 'outngramA.txt' και 'outngramB.txt', τα δύο αρχεία που περιέχουν τους κωδικούς, το όνομα του αρχείου στο οποίο θα αποθηκευτούν τα αποτελέσματα σαν output, και τον αριθμό των πεδίων κάθε εγγραφής. Η σύγκριση γίνεται κατά α/α πεδίου καταρχάς, και κωδικού του πρώτου αρχείου με όλους τους κωδικούς του δεύτερου αρχείου στη συνέχεια. Για τον λόγο αυτό εφαρμόζεται μία επαναληπτική διαδικασία for από 1 έως n, όπου 'n' είναι ο αριθμός των πεδίων της εγγραφής, στην οποία λαμβάνει χώρα η όλη διαδικασία της σύγκρισης όπως περιγράφεται στη συνέχεια.

Διαβάζονται τα δύο αρχεία εισόδου που περιέχουν τους κωδικούς, και υλοποιείται μια αλληλουχία του κωδικού και του α/α του πεδίου που υπάρχει από τον επαναληπτικό βρόχο for (1..n). Με βάση αυτό δημιουργείται ένα μοναδικό κλειδί κωδικός-α/α πεδίου για κάθε αρχείο. Στη συνέχεια το πρώτο αρχείο με τους κωδικούς διαβάζεται με μια επαναληπτική διαδικασία while και το δεύτερο όμοια με ένα while εμφωλιασμένο στο πρώτο. Αυτό συμβαίνει έτσι ώστε να επιτευχθούν όλοι οι δυνατοί συνδυασμοί κωδικών.

Κατόπιν χρησιμοποιώντας το παραπάνω κλειδί, γίνεται αναζήτηση των αντίστοιχων εγγραφών στα αρχεία που περιέχουν τα Ν-γράμματα. Εν συνεχεία αυτά διαβάζονται με επαναληπτικό βρόχο while ο οποίος είναι και αυτός εμφωλιασμένος στις παραπάνω επαναληπτικές διαδικασίες. Η σύγκριση γίνεται με τον τρόπο που περιγράφεται ακολούθως. Υπάρχει ένα κλειδί του πρώτου αρχείου με τους κωδικούς και ένα κλειδί του δεύτερου αρχείου. Εντοπίζονται οι εγγραφές στο πρώτο αρχείο με τα Ν-γράμματα που το κλειδί τους είναι ίσο με το κλειδί του πρώτου αρχείου με τους κωδικούς. Παρόμοια εντοπίζονται και οι εγγραφές του δεύτερου αρχείου με Ν-

γράμματα σύμφωνα με τα κλειδιά του δεύτερου αρχείου με τους κωδικούς. Στο σημείο αυτό πρέπει να σημειωθεί ότι, όταν διαβάζονται όλες οι παραπάνω εγγραφές κλείνουν τα αρχεία με τα N-γράμματα, και αυτά ανοίγουν πάλι με ένα επόμενο ζεύγος κλειδιών. Δηλαδή αν υφίστανται N πεδία, K εγγραφές κωδικών στο πρώτο αρχείο και L εγγραφές κωδικών στο δεύτερο αρχείο, τότε η προσπέλαση στα δύο αρχεία με τα N-γράμματα γίνεται NxKxL φορές.

Στην συνέχεια στα δύο αρχεία που περιέχουν τα N-γράμματα, έχοντας τις εγγραφές ταξινομημένες αλφαριθμητικά, δηλαδή κατά κλειδί και N-γράμμα, υπάρχει η δυνατότητα να γίνει η σύγκριση για ένα ζεύγος κλειδιών κάθε φορά με τον ακόλουθο τρόπο. Αν το N-γράμμα, έστω A, του πρώτου αρχείου ταυτίζεται με το N-γράμμα, έστω B, του δεύτερου αρχείου, τότε καταμετράται με έναν μετρητή, έστω 'cntc'. Αν ισχύει 'A < B' τότε διαβάζεται το πρώτο αρχείο μέχρι να ισχύει 'A ≥ B'. Αν ισχύει το ίσο '=' τότε μετράται (cntc++), αλλιώς όταν ισχύει το μεγαλύτερο '>' διαβάζεται το δεύτερο αρχείο. Αυτό συμβαίνει μέχρι αντίστοιχα να βρεθεί 'B ≥ A' οπότε και εφαρμόζεται η ίδια διαδικασία. Δηλαδή διαβάζονται κατά προτεραιότητα τα N-γράμματα που έχουν την μικρότερη τιμή. Επίσης κάθε φορά που διαβάζεται μια εγγραφή, τότε αυτή καταμετράται με έναν μετρητή cntA για το πρώτο αρχείο και cntB για το δεύτερο. Όταν τελειώσουν οι εγγραφές με το συγκεκριμένο κλειδί στο ένα αρχείο, τότε αυτό κλείνει, αλλά συνεχίζεται η ανάγνωση στο άλλο αρχείο καταμετρώντας τις υπόλοιπες εγγραφές. Αυτό γίνεται μέχρι να βρεθεί ένα άλλο κλειδί, οπότε κλείνει και αυτό και υπολογίζεται το ποσοστό από τον τύπο που δίνεται στην περιγραφή του πρωτοκόλλου.

Δηλαδή θεωρούμε ένα πεδίο 'n_field' το οποίο ενδέχεται να αντιστοιχεί για παράδειγμα σε ένα επίθετο ή τηλέφωνο. Τότε θα είναι 'cntA' τα N-γράμματα του πρώτου αρχείου, έστω με κωδικό r_i, και 'cntB' τα N-γράμματα του δεύτερου αρχείου, έστω με κωδικό rec_j. Αν τα κοινά τους N-γράμματα είναι το 'cntc', τότε για κάθε ζεύγος (r_i, rec_j) για το πεδίο 'n_field', υπολογίζεται το ποσοστό, έστω 'ngram_score', σύμφωνα με τον τύπο:

$$\left| \text{ngram\_score} = \frac{\text{cntc}}{0.5 * (\text{cntA} + \text{cntB})} \right|$$

Τέλος γράφονται όλες αυτές οι τιμές για κάθε πεδίο και κάθε ζεύγος κωδικών σε ένα αρχείο αποτελεσμάτων. Επίσης υπάρχει η δυνατότητα, να χρησιμοποιηθεί μια μεταβλητή m που να αντιστοιχεί στην τιμή κατωφλίου, έτσι ώστε μόνο οι τιμές με 'ngram_score ≥ m' να εγγράφονται στο αρχείο αποτελεσμάτων. Προφανώς εάν m = 0 τότε γράφονται όλοι οι δυνατοί συνδυασμοί κωδικών των δύο αρχείων προς σύγκριση και για όλα τα πεδία. Στον παρακάτω πίνακα διακρίνεται ένα τμήμα του αρχείου αποτελεσμάτων. Σε αυτό περιέχονται οι κωδικοί των εγγραφών που συγκρίνονται, ο αριθμός των κοινών τους N-γραμμάτων, ο αριθμός των N-γραμμάτων που έχει κάθε εγγραφή, καθώς και η ποσοστιαία βαθμολογία με βάση τον τύπο.

Κωδικός 1ης Εγγραφής	Κωδικός 2ης Εγγραφής	Κοινά N-γράμματα	N-γράμματα 1ης Εγγραφής	N-γράμματα 2ης Εγγραφής	Ngram score
r-2241	rec-2241	127	127	127	1
r-2241	rec-2561	1	127	127	0.007874015748031
r-2241	rec-2811	3	127	2047	0.002759889604416
r-2241	rec-3761	0	127	15	0
r-2561	rec-2241	1	127	127	0.007874015748031
r-2561	rec-2561	127	127	127	1
r-2561	rec-2811	1	127	2047	0.000919963201472
r-2561	rec-3761	3	127	15	0.042253521126761
r-2811	rec-2241	3	2047	127	0.002759889604416
r-2811	rec-2561	1	2047	127	0.000919963201472
r-2811	rec-2811	1023	2047	2047	0.499755740107474
r-2811	rec-3761	0	2047	15	0

Πίνακας 4-4: Παράδειγμα αρχείου αποτελεσμάτων 'result.txt'

Κεφάλαιο 5

Αξιολόγηση Εφαρμογής

Στο παρόν κεφάλαιο θα γίνει η αξιολόγησης της εφαρμογής που υλοποιεί την τεχνική της συνένωσης εμπιστευτικών δεδομένων με βάση την χρήση N-γραμμάτων και κρυπτογραφικών μεθόδων για να επιτύχει την διασύνδεση των εγγραφών. Η συγκεκριμένη αξιολόγηση αφορά τόσο την πρακτικότητα της εφαρμογής, όσο και την ποιότητα της. Με τον όρο πρακτικότητα ουσιαστικά εννοείται ο χρόνος που χρειάζεται για να τρέξουν τα προγράμματα της εφαρμογής, καθώς και τους υπολογιστικούς πόρους που χρειάζεται. Όσον αφορά την ποιότητα, θα ελεγχθεί κατά πόσο τελικά η τεχνική των N-γραμμάτων είναι σε θέση να εφαρμόσει με επιτυχία την διασύνδεση των εγγραφών, και ποια είναι η απόδοση της ανάλογα με τα αλφαριθμητικά που συγκρίνονται.

Πριν γίνει η παρουσίαση της αξιολόγησης πρέπει να γίνει αναφορά στο υπολογιστικό σύστημα που χρησιμοποιήθηκε για τις ανάγκες της έρευνας. Συγκεκριμένα χρησιμοποιήθηκε ένας προσωπικός υπολογιστής με επεξεργαστή Intel Pentium 4 στα 2.0 GHz, με μνήμη RAM 512 Mbyte. Το λειτουργικό σύστημα του υπολογιστή είναι Windows XP, παρά όλα αυτά τα προγράμματα της εφαρμογής έτρεξαν στο λογισμικό Cygwin το οποίο ουσιαστικά προσομοιώνει το περιβάλλον του λειτουργικού συστήματος Unix.

5.1 Αξιολόγηση Πρακτικότητας Εφαρμογής

Όσον αφορά την πρακτικότητα της εφαρμογής θα παρουσιαστούν ορισμένα στατιστικά μεγέθη. Αυτά αφορούν τους χρόνους που χρειάζονται για να τρέξουν τα προγράμματα

σε διάφορες περιπτώσεις, όπως είναι το διαφορετικό πλήθος των εγγραφών προς σύγκριση, καθώς και το μήκος των N-γραμμάτων που δημιουργούνται. Παράλληλα για τις ίδιες περιπτώσεις θα ελεγχθεί ο χώρος στον σκληρό δίσκο που καταλαμβάνουν τα διάφορα αρχεία που παράγονται κατά την εφαρμογή των προγραμμάτων.

Με το λογισμικό Febrl παράχθηκαν συνολικά 7 διαφορετικά ζεύγη συνόλων δεδομένων. Τα σύνολα δεδομένων παράχθηκαν κατά ζεύγη ώστε η σύγκριση των N-γραμμάτων να γίνεται για υποτιθέμενες Βάσεις Δεδομένων με ίδιο πλήθος εγγραφών. Το πρώτο ζεύγος που παράχθηκε είχε 4 εγγραφές για κάθε σύνολο, το δεύτερο ζεύγος είχε 8 εγγραφές, το τρίτο είχε 20 εγγραφές, το τέταρτο είχε 40 εγγραφές, το πέμπτο είχε 80 εγγραφές, το έκτο είχε 160 εγγραφές και το έβδομο 1000 εγγραφές. Σε όλες τις περιπτώσεις κάθε εγγραφή έχει τον προκαθορισμένο αριθμό πεδίων ο οποίος, όπως έχει αναφερθεί στην ενότητα 3.1, είναι 13 πεδία μη συμπεριλαμβανομένου του κωδικού της εγγραφής. Το μέγεθος των αρχείων αυτών είναι ιδιαίτερα μικρό. Στην πρώτη περίπτωση δεν ξεπερνά καν το 1 Kbyte, ενώ στην τελευταία με τις 1000 εγγραφές είναι περίπου 229 Kbytes. Επίσης όσον αφορά τον χρόνο που χρειάστηκε για να δημιουργηθούν τα σύνολα δεδομένων είναι περίπου ο ίδιος, κυμαίνεται γύρω στα 40 δευτερόλεπτα, εκτός από την τελευταία περίπτωση η οποία έκανε σχεδόν τον διπλάσιο χρόνο. Αυτό συμβαίνει διότι το Febrl χρειάζεται κάποιο χρονικό διάστημα της τάξης των 38 δευτερολέπτων για να συνδεθεί με τα αρχεία που έχουν τα ονόματα, τις διευθύνσεις κ.λ.π., και ελάχιστο χρόνο για να δημιουργήσει κάθε εγγραφή. Όλα τα προηγούμενα παρουσιάζονται συνοπτικά στον παρακάτω πίνακα.

Πλήθος Εγγραφών Συνόλου	Χρόνος Δημιουργίας σε sec.	Μέγεθος σε bytes
4	39,26	565
8	39,40	997
20	39,98	2.310
40	40,94	4.650
80	40,60	9.330
160	42,02	18.400
1000	70,43	229.000

Πίνακας 5-1: Στατιστικά Στοιχεία Παραγωγής Συνόλου Εγγραφών

Εν συνεχεία θα αναλυθούν ορισμένα στοιχεία που αφορούν την δημιουργία των Ν-γραμμάτων, την κρυπτογράφηση τους, καθώς και την σύγκριση. Μεγαλύτερη βάση δόθηκε στην εξέταση περιπτώσεων με δι-γράμματα. Ένα ιδιαίτερα σημαντικό συμπέρασμα είναι ότι η γλώσσα προγραμματισμού Perl, κατά την δημιουργία των Ν-γραμμάτων, δεν έχει την δυνατότητα να τα αποθηκεύσει ένα-ένα στο δίσκο, αλλά όλα μαζί. Αυτό έχει σαν αποτέλεσμα κατά την επεξεργασία εκατομμυρίων Ν-γραμμάτων να παρουσιάζεται πρόβλημα υπερχείλισης μνήμης. Συγκεκριμένα η Perl, ανεξαρτήτως του υπολογιστικού συστήματος, μπορεί να χρησιμοποιήσει μέχρι περίπου 261 Mbytes μνήμης RAM. Επομένως κατά την επεξεργασία ενός μεγάλου πλήθους Ν-γραμμάτων, όπου χρειάζεται περισσότερη μνήμη, δημιουργείται σφάλμα παραβίασης του μέγιστου επιτρεπτού ορίου χρήσης μνήμης.

Το παραπάνω πρόβλημα εμφανίστηκε στις περιπτώσεις όπου το σύνολο είχε από 80 εγγραφές και πάνω. Μάλιστα εξετάσθηκε και η περίπτωση δημιουργίας μέχρι και έξι-γραμμάτων, λόγω του μικρότερου πλήθους τους από τα δι-γράμματα, αλλά και πάλι εμφανίστηκε το ίδιο σφάλμα. Όσον αφορά τις περιπτώσεις όπου ήταν δυνατή η σύσταση των Ν-γραμμάτων, ιδιαίτερο ενδιαφέρον παρουσίασε το παρακάτω γεγονός. Σε σύνολα δεδομένων με ίδιο πλήθος εγγραφών, υπάρχουν μεγάλες αποκλίσεις όσον αφορά το πλήθος των Ν-γραμμάτων που παράγονται. Χαρακτηριστικό παράδειγμα αποτελεί η περίπτωση των συνόλων με 40 εγγραφές, όπου το ένα παρήγαγε 1.133.824 δι-γράμματα, ενώ το άλλο παρήγαγε 797.186 δι-γράμματα. Οι αποκλίσεις αυτές οφείλονται στο γεγονός ότι ενδέχεται να διαφέρουν κατά πολύ τα μεγέθη των τιμών που περιέχουν τα πεδία των εγγραφών των διαφορετικών συνόλων δεδομένων. Εκτός των άλλων πολλές εγγραφές σε ορισμένα πεδία, όπως η δεύτερη διεύθυνση, δεν έχουν καν κάποια τιμή. Επίσης πρέπει να αναφερθεί ότι ο χρόνος δημιουργίας των Ν-γραμμάτων αυξάνει όσο αυξάνεται το πλήθος των Ν-γραμμάτων, όπως και το μέγεθος των αρχείων που τα περιλαμβάνουν, το οποίο κυμαίνεται από μερικά Kbytes ως αρκετά Mbytes.

Όσον αφορά την εφαρμογή του αλγορίθμου κρυπτογράφησης και της συνάρτησης κατακερματισμού στα αρχεία με τα Ν-γράμματα, πρέπει να αναφερθεί ότι επιτυγχάνεται σχετικά γρήγορα. Αυτό αφορά και τα δυο στάδια, δηλαδή την δημιουργία των συνόψεων των κρυπτογραφημένων αρχείων, και την ταξινόμηση τους. Μάλιστα τα αρχεία που παράγονται, για μικρό αριθμό εγγραφών έχουν πολύ μεγαλύτερο μέγεθος, σχεδόν διπλάσιο. Αυτό οφείλεται στην συνάρτηση

κατακερματισμού η οποία για κάθε πεδίο κάθε εγγραφής δημιουργεί ίδιου μήκους συνόψεις.

Το πρόγραμμα που εφαρμόζει την σύγκριση των δεδομένων χρειάζεται μεγάλο χρονικό διάστημα για να επιτύχει την σύγκριση, το οποίο αυξάνει εκθετικά όσο μεγαλώνει το πλήθος των N-γραμμάτων. Μάλιστα στην περίπτωση των συνόλων με 4 εγγραφές χρειάζεται αρκετά δευτερόλεπτα, ενώ όταν πρέπει να συγκρίνει δεδομένα με 40 εγγραφές χρειάζεται μερικές ώρες. Αυτό οφείλεται στο γεγονός ότι κάθε ένα πεδίο κάθε εγγραφής ξεχωριστά, πρέπει να συγκριθεί με όλα τα αντίστοιχα του πεδία των εγγραφών του άλλου συνόλου. Φυσικά η σύγκριση αυτή γίνεται μεταξύ δεκάδων N-γραμμάτων που καθυστερούν ακόμη περισσότερο την διαδικασία.

Όλα τα προηγούμενα συμπεριλαμβάνονται συνοπτικά στον πίνακα που ακολουθεί. Είναι προφανές ότι η σύγκριση έγινε μεταξύ των δυο συνόλων με ίδιο πλήθος εγγραφών οπότε ο χρόνος σύγκρισης είναι ο ίδιος για κάθε ένα από τα παρακάτω ζεύγη.

Πλήθος Εγγραφών Συνόλου	**Πλήθος δι-γραμμάτων**	**Χρόνος σε sec δημιουργίας δι-γραμμάτων**	**Μέγεθος αρχείων δι-γραμμάτων**	**Χρόνος σε sec κρυπτογράφησης**	**Μέγεθος κρυπτογραφημενων αρχείων**	**Χρόνος σε sec σύγκρισης**
4	20.674	2	527Kb	1	930Kb	78
4	37.104	3	959Kb	2	1.669Kb	78
8	34.738	2	872Kb	2	1.564Kb	662
8	30.468	2	733Kb	2	1.387Kb	662
20	81.284	5	2.014Kb	3	3.664Kb	Πάνω από 1 ώρα
20	145.654	10	3.818Kb	6	6.551Kb	Πάνω από 1 ώρα
40	1.133.824	97	34.110Kb	37	57.238Kb	Αρκετές ώρες
40	797.186	41	23.648Kb	30	36.221Kb	Αρκετές ώρες
80	Λάθος μνήμης	----------	----------	----------	----------	----------
160	Λάθος μνήμης	----------	----------	----------	----------	----------
1000	Λάθος μνήμης	----------	----------	----------	----------	----------

Πίνακας 5-2: Στατιστικά Στοιχεία για τα δι-γράμματα

Εκτός των άλλων εξετάσθηκαν και περιπτώσεις επεξεργασίας Ν-γραμμάτων μεγαλύτερου μήκους από 2. Συγκεκριμένα για το σύνολο με τις 40 εγγραφές δημιουργήθηκαν τρι-γράμματα, τέτρα-γράμματα, πέντα-γράμματα και έξι-γράμματα. Η εξέταση μεγαλύτερου μήκους δεν θα είχε πλέον νόημα αφού θα ήταν πολύ λίγα τα αλφαριθμητικά με μήκος μεγαλύτερο του 6. Όπως είναι λογικό καθώς αυξάνεται το μήκος των Ν-γραμμάτων μειώνεται αισθητά το πλήθος τους που παράγεται, καθώς και το μέγεθος των αρχείων και ο χρόνος επεξεργασίας. Παρά όλα αυτά το συνολικό πλήθος των Ν-γραμμάτων προς σύγκριση είναι αρκετά μεγάλο, οπότε ο χρόνος που διαρκεί η σύγκριση ξεπερνά κατά πολύ την μια ώρα, στην καλύτερη περίπτωση. Στον παρακάτω πίνακα παρουσιάζονται τα προηγούμενα στοιχεία, τα οποία αφορούν το ένα σύνολο με τις 40 εγγραφές.

Πλήθος Εγγραφών Συνόλου	**Μήκος Ν-γραμμάτων**	**Πλήθος Ν-γραμμάτων**	**Χρόνος σε sec δημιουργίας Ν-γραμμάτων**	**Μέγεθος αρχείων Ν-γραμμάτων**
40	2	1.133.824	97	34.110Kb
40	3	707.520	59	26.016Kb
40	4	394.972	31	16.814Kb
40	5	230.242	17	11.079Kb
40	6	115.130	8	5.979Kb

Πίνακας 5-3: Στατιστικά Στοιχεία διαφορετικού μήκους Ν-γραμμάτων του ίδιου Συνόλου Εγγραφών

5.2 Αξιολόγηση Ποιότητας Εφαρμογής

Στην ενότητα αυτή παρουσιάζονται ορισμένα στοιχεία που αφορούν την ποιότητα της εφαρμογής. Δηλαδή κατά πόσο η τεχνική των Ν-γραμμάτων είναι σε θέση να εφαρμόσει την διαδικασία της διασύνδεσης των εγγραφών με επιτυχία. Προκειμένου να επιτευχθεί αυτό εξετάσθηκαν σύνολα δεδομένων με συγκριμένες τιμές σε κάποια πεδία. Θεωρήθηκε ότι αυτές οι τιμές μπορούν να αποτελέσουν ενδεικτικό κριτήριο της ποιοτικής απόδοσης της εφαρμογής.

Συγκεκριμένα εξετάσθηκαν οι τιμές του πεδίου «επώνυμο» των συνόλων δεδομένων, αλλά δεν θα υπάρξει διαφορά αν κάποιος εξετάσει ένα άλλο πεδίο. Χρησιμοποιήθηκαν δυο διαφορετικά ονόματα που αντιπροσωπεύουν τις κατηγορίες των μεγάλων αλφαριθμητικών και των σχετικά μικρών. Για κάθε μια εγγραφή στο ένα σύνολο δεδομένων που περιείχε το προς εξέταση αλφαριθμητικό, δημιουργήθηκαν εγγραφές στο άλλο σύνολο δεδομένων με παραλλαγές των συγκεκριμένων αλφαριθμητικών. Οι παραλλαγές που εφαρμόστηκαν βασίστηκαν σε αυτές που δημιουργεί και το λογισμικό Febrl, και ήταν:

- Προσθήκη ενός χαρακτήρα σε κάποιο σημείο του αλφαριθμητικού.
- Αφαίρεση ενός χαρακτήρα από οποιοδήποτε σημείο του αλφαριθμητικού.
- Αλλαγή ενός χαρακτήρα του αλφαριθμητικού με κάποιον άλλον.
- Αλλαγή ενός χαρακτήρα με κάποιον άλλον, και αφαίρεση ενός άλλου χαρακτήρα από το αλφαριθμητικό.

Στους παρακάτω πίνακες διακρίνονται συνοπτικά τα ονόματα και οι παραλλαγές τους, καθώς και τα αποτελέσματα που έδωσε η σύγκριση τους για N = 2,3,4,5. Δηλαδή το πλήθος των N-γραμμάτων κάθε αλφαριθμητικού, το πλήθος των κοινών N-γραμμάτων και η ποσοστιαία βαθμολογία της σύγκρισης.

Επώνυμο ($1^{η}$ Εγγραφή)	Παραλλαγή ($2^{η}$ Εγγραφή)	Κοινά N-γράμματα	N-γράμματα $1^{ης}$ Εγγραφής	N-γράμματα $2^{ης}$ Εγγραφής	Ngram score
mitrogiannis	mitrogiannis	2047	2047	2047	1
mitrogiannis	mitrogianis	1023	2047	1023	0.666449511400651
mitrogiannis	mitrogianniv	1023	2047	2047	0.499755740107474
mitrogiannis	mitrogiannisi	2047	2047	4095	0.66655812438945
mitrogiannis	mikrogiannis	511	2047	2047	0.249633610161212
petrou	petrou	31	31	31	1
petrou	petrov	15	31	31	0.483870967741935
petrou	petrous	31	31	63	0.659574468085106
petrou	petro	15	31	15	0.652173913043478
petrou	pirou	3	31	15	0.130434782608696

Πίνακας 5-4: Συγκριτικά αποτελέσματα για δι-γράμματα (N = 2)

Επώνυμο (1η Εγγραφή)	Παραλλαγή (2η Εγγραφή)	Κοινά N-γράμματα	N-γράμματα 1ης Εγγραφής	N-γράμματα 2ης Εγγραφής	Ngram score
mitrogiannis	mitrogiannis	1023	1023	1023	1
mitrogiannis	mitrogianis	255	1023	511	0.332464146023468
mitrogiannis	mitrogianniv	511	1023	1023	0.499511241446725
mitrogiannis	mitrogiannisi	1023	1023	2047	0.666449511400651
mitrogiannis	mikrogiannis	127	1023	1023	0.124144672531769
petrou	petrou	15	15	15	1
petrou	petrov	7	15	15	0.466666666666667
petrou	petrous	15	15	31	0.652173913043478
petrou	petro	7	15	7	0.636363636363636
petrou	pirou	1	15	7	0.090909090909091

Πίνακας 5-5: Συγκριτικά αποτελέσματα για τρι-γράμματα (N = 3)

Επώνυμο (1η Εγγραφή)	Παραλλαγή (2η Εγγραφή)	Κοινά N-γράμματα	N-γράμματα 1ης Εγγραφής	N-γράμματα 2ης Εγγραφής	Ngram score
mitrogiannis	mitrogiannis	511	511	511	1
mitrogiannis	mitrogianis	63	511	255	0.164490861618799
mitrogiannis	mitrogianniv	255	511	511	0.499021526418787
mitrogiannis	mitrogiannisi	511	511	1023	0.666232073011734
mitrogiannis	mikrogiannis	63	511	511	0.123287671232877
petrou	petrou	7	7	7	1
petrou	petrov	3	7	7	0.428571428571429
petrou	petrous	7	7	15	0.636363636363636
petrou	petro	3	7	3	0.6
petrou	pirou	0	7	3	0

Πίνακας 5-6: Συγκριτικά αποτελέσματα για τέτρα-γράμματα (N = 4)

Επώνυμο (1η Εγγραφή)	Παραλλαγή (2η Εγγραφή)	Κοινά N-γράμματα	N-γράμματα 1ης Εγγραφής	N-γράμματα 2ης Εγγραφής	Ngram score
mitrogiannis	mitrogiannis	255	255	255	1
mitrogiannis	mitrogianis	31	255	127	0.162303664921466
mitrogiannis	mitrogianniv	127	255	255	0.498039215686275
mitrogiannis	mitrogiannisi	255	255	511	0.66579634464752
mitrogiannis	mikrogiannis	31	255	255	0.12156862745098
petrou	petrou	3	3	3	1
petrou	petrov	1	3	3	0.333333333333333
petrou	petrous	3	3	7	0.6
petrou	petro	1	3	1	0.5
petrou	pirou	0	3	1	0

Πίνακας 5-6: Συγκριτικά αποτελέσματα για πέντα-γράμματα (N = 5)

Εξετάζοντας τους πίνακες που παρατίθενται πιο πάνω συνάγονται ορισμένα ενδιαφέροντα συμπεράσματα τα οποία συνοψίζονται παρακάτω:

- Κάθε φορά που αυξάνεται το μήκος των N-γραμμάτων κατά ένα, υποδιπλασιάζεται το πλήθος τους. Αυτό αποδεικνύεται από τον τύπο «πλήθος = $2^a - 1$», που αναφέρεται στην ενότητα 4.1.
- Η ποσοστιαία βαθμολογία και για τις δυο κατηγορίες αλφαριθμητικών φαίνεται να είναι ιδιαίτερα μικρή για τέτοιου είδους λάθη. Η μέγιστη βαθμολογία που εμφανίζεται όταν έχει γίνει προσθήκη ενός χαρακτήρα στο τέλος του αλφαριθμητικού είναι 0.665 περίπου, την στιγμή που θα περίμενε κανείς ότι η ελάχιστη βαθμολογία για δυο παρόμοια αλφαριθμητικά θα ήταν μεγαλύτερη του 0.7.
- Στην περίπτωση των δι-γραμμάτων και για τις δυο κατηγορίες η περίπτωση προσθήκης ή αφαίρεσης ενός χαρακτήρα δίνει παρόμοια αποτελέσματα, τα οποία κυμαίνονται κοντά στην μέγιστη βαθμολογία.
- Γενικότερα για όλα τα N-γράμματα, στην περίπτωση όπου γίνεται προσθήκη ενός χαρακτήρα η ποσοστιαία βαθμολογία παραμένει στα ίδια επίπεδα.
- Όσον αφορά την κατηγορία μεγάλων αλφαριθμητικών όταν αφαιρείται ένας χαρακτήρας, υπάρχει μια μείωση του ποσοστού όσο αυξάνει το N. Συγκεκριμένα ενώ για N = 2, έχει το μέγιστο ποσοστό (0.66), για N = 3

υποδιπλασιάζεται, και για N = 4,5 μειώνεται στο 0.16 περίπου. Αντίθετα στην δεύτερη κατηγορία αλφαριθμητικών το ποσοστό παραμένει στα ίδια επίπεδα όσο κι αν αυξάνει το N.

- Η εναλλαγή ενός ακριανού χαρακτήρα έχει σαν αποτέλεσμα να διαφέρουν τα μισά N-γράμματα, οπότε η ποσοστιαία βαθμολογία κυμαίνεται κοντά στην βάση (0.5). το ποσοστό αυτό είναι παρόμοιο σε όλα τα δυνατά μήκη N-γραμμάτων για την πρώτη κατηγορία.
- Στην κατηγορία σχετικά μικρών αλφαριθμητικών, κατά την εναλλαγή ενός ακριανού χαρακτήρα, το ποσοστό μειώνεται αισθητά όσο αυξάνει το N. Από 0.48 που είναι για N = 2, γίνεται 0.33 για N = 5.
- Όταν εναλλάσσεται ένας μόνο χαρακτήρας που δεν είναι στην άκρη τότε τα αποτελέσματα είναι απογοητευτικά, με ποσοστά που κυμαίνονται κάτω από 0.24. Δηλαδή για ένα σύνηθες ορθογραφικό λάθος, δεν υπάρχει περίπτωση η εφαρμογή να συνδέσει τις δύο εγγραφές.
- Όταν γίνονται δυο ορθογραφικά λάθη στο ίδιο αλφαριθμητικό τότε η εφαρμογή δεν μπορεί να συνδέσει τις εγγραφές. Διότι στην καλύτερη περίπτωση το ποσοστό ομοιότητας που δίνει είναι περίπου 0.13, ενώ καθώς αυξάνεται το N, υπάρχει η περίπτωση να μην ταιριάζει κανένα N-γράμμα.

Τέλος αξίζει να σημειωθεί ότι στην διαδικασία της σύγκρισης δεν παίζει κάποιο ρόλο η εφαρμογή των κρυπτογραφικών μεθόδων στα δεδομένα. Δηλαδή τα αποτελέσματα της σύγκρισης είναι πανομοιότυπα είτε αυτή εφαρμόζεται σε κρυπτογραφημένες εγγραφές, είτε σε μη κρυπτογραφημένες.

5.3 Συμπεράσματα

Με βάση την προηγούμενη αξιολόγηση, τόσο σε επίπεδο πρακτικότητας, όσο και σε επίπεδο ποιότητας των αποτελεσμάτων, συνάγεται το συμπέρασμα ότι η εφαρμογή της τεχνικής των N-γραμμάτων παρουσιάζει ιδιαίτερες αδυναμίες.

Από την μια πλευρά παρατηρείται ότι η συγκεκριμένη τεχνική συνήθως δίνει μικρά ποσοστά ομοιότητας των εγγραφών, με αποτέλεσμα πολλές φορές να μην

επιτυγχάνεται η επιβεβαίωση της ομοιότητας τους. Βέβαια στην περίπτωση εγγραφών που αναφέρονται σε διαφορετικές οντότητες η ποσοστιαία βαθμολογία είναι ακόμη πιο μικρή, σχεδόν μηδενική. Επομένως είναι δυνατόν, με την παρέμβαση του χρήστη, να ελεγχθεί κατά πόσο ένα σχετικά μικρό ποσοστό ομοιότητας ενδέχεται να αφορά εγγραφές που αναφέρονται στην ίδια οντότητα. Παρόλα αυτά, όπως έγινε φανερό προηγουμένως στην ενότητα 5.2 ακόμη και ένα μικρό και σύνηθες τυπογραφικό λάθος μπορεί να έχει σαν αποτέλεσμα το μη ταίριασμα δύο εγγραφών.

Εκτός των άλλων, κατά την επεξεργασία των εγγραφών παράγεται ιδιαίτερα μεγάλος όγκος N-γραμμάτων. Αρκεί μόνο να σκεφτεί κανείς ότι 4 εγγραφές παράγουν πάνω από 20.000 διαφορετικά N-γράμματα, ενώ 40 εγγραφές παράγουν πάνω από 1.000.000 διαφορετικά N-γράμματα. Είναι προφανές ότι οι Βάσεις Δεδομένων των οργανισμών που έχουν σκοπό να διαμοιραστούν τις πληροφορίες τους, απαρτίζονται από χιλιάδες, ίσως και εκατομμύρια εγγραφές. Επομένως για ένα τέτοιο σύνολο δεδομένων παράγονται δισεκατομμύρια N-γράμματα, όπου η αποθήκευση και η σύγκριση τους μπορεί να λάβει χώρα μόνο σε πολύ ισχυρά υπολογιστικά συστήματα.

Κεφάλαιο 6

Επίλογος

Μια από τις σπουδαιότερες προκλήσεις που καλείται να αντιμετωπίσει η κοινότητα των Βάσεων Δεδομένων, αλλά και η κοινωνία της πληροφορίας γενικότερα, είναι το ζήτημα της διατήρησης της εμπιστευτικότητας των πληροφοριών. Ιδιαίτερο ενδιαφέρον παρουσιάζει η επίτευξη αυτού του στόχου όταν επιθυμούμε να ενοποιήσουμε τα δεδομένα που προέρχονται από πολλαπλές Βάσεις Δεδομένων.

Η διατήρηση της εμπιστευτικότητας κατά την ενοποίηση των δεδομένων από πολλαπλές πηγές αποτελεί ένα αρκετά σύγχρονο ερευνητικό πεδίο όπου διαγράφονται πολλές προοπτικές. Μέχρι στιγμής η έρευνα στο συγκεκριμένο πεδίο είναι σε πρώιμο στάδιο, όμως έχουν τεθεί οι βάσεις ώστε στο άμεσο μέλλον να αναπτυχθούν πολλές μέθοδοι που θα είναι ικανές να επιτύχουν τον στόχο της διατήρησης της εμπιστευτικότητας. Έχει ήδη γίνει ένας πρωταρχικός διαχωρισμός σε τομείς που αφορούν διαφορετικές εκφάνσεις της ενοποίησης των δεδομένων, και αναπτύσσονται αλγόριθμοι και μεθοδολογίες για κάθε έναν. Μάλιστα στο πεδίο της ενοποίησης των πληροφοριών αυτής κάθε αυτής, εδώ και μεγάλο χρονικό διάστημα έχουν αναπτυχθεί πολλές μέθοδοι. Επομένως το ενδιαφέρον παρουσιάζεται στην διερεύνηση τρόπων με τους οποίους θα υλοποιηθεί το ζήτημα της διατήρησης της ασφάλειας στις ήδη υπάρχουσες μεθόδους.

Ένας τομέας που παρουσιάζει ιδιαίτερο ενδιαφέρον είναι η διασύνδεση των εγγραφών. Η παρούσα εργασία ασχολήθηκε σχεδόν αποκλειστικά με την διερεύνηση και ανάπτυξη μεθόδων που θα επιτύχουν την διασύνδεση των εγγραφών διατηρώντας παράλληλα την εμπιστευτικότητα τους. Προς αυτήν την κατεύθυνση εξετάσθηκε το λογισμικό Febrl το οποίο έχει στην διάθεση του αρκετές μεθόδους και αλγορίθμους που επιτυγχάνουν την διασύνδεση των εγγραφών.

Παράλληλα παρουσιάσθηκε ένα πρωτόκολλο το οποίο εφαρμόζει την τεχνική των N-γραμμάτων σε συνδυασμό με κρυπτογραφικές μεθόδους ώστε να επιτύχει τον παραπάνω στόχο. Μετά την υλοποίηση του συγκεκριμένου πρωτοκόλλου έγιναν φανερές αρκετές αδυναμίες του. Παρά όλα αυτά υπό ορισμένες προϋποθέσεις το πρωτόκολλο μπορεί να θεωρηθεί επιτυχημένο, ιδιαίτερα στον τομέα της προστασίας των ευαίσθητων πληροφοριών, και είναι δυνατόν να εφαρμοστεί κάτω από τις κατάλληλες συνθήκες.

Στο σημείο αυτό αξίζει να γίνει αναφορά σε ορισμένα στοιχεία που δεν υλοποιήθηκαν στην παρούσα εργασία, τα οποία όμως συνδέονται άμεσα με αυτήν, και μπορούν να αποτελέσουν την μελλοντική επέκταση της. Ένα πλεονέκτημα της γλώσσας προγραμματισμού Perl είναι η επεκτασιμότητα της, δηλαδή η δυνατότητα που έχει να συνδυάζονται τα προγράμματα της με κώδικα από άλλες γλώσσες προγραμματισμού. Δυστυχώς στα πλαίσια της εργασίας, παρότι έγινε ενδελεχής έρευνα, δεν ήταν δυνατή η εύρεση ενός τρόπου με τον οποίο θα συνδυάζονταν σωστά τα προγράμματα που υλοποιήθηκαν, με τον κώδικα του Febrl που είναι γραμμένος σε Python. Επομένως κάποιος θα μπορούσε να επεκτείνει τις δυνατότητες του Febrl, μια προτροπή που την δίνουν οι ίδιοι οι δημιουργοί του, ώστε να χρησιμοποιεί και N-γράμματα για την σύγκριση της ομοιότητας των δεδομένων, καθώς και τις κρυπτογραφικές μεθόδους για την διασφάλιση της εμπιστευτικότητας.

Επί προσθέτως θα ήταν ενδιαφέρουσα μια μετατροπή της τεχνικής που χρησιμοποιεί N-γράμματα για την σύγκριση της ομοιότητας των δεδομένων, ώστε να μειωθεί το πλήθος των παραγόμενων N-γραμμάτων. Για παράδειγμα το πρόγραμμα θα μπορούσε να ελέγχει το μέγεθος των αλφαριθμητικών, και ανάλογα με το αν είναι μεγαλύτερα από κάποιο ορισμένο μέγεθος, αντί να τα σπάει σε δι ή τρι-γράμματα, να τα διαχωρίζει σε μεγαλύτερο μήκος N-γραμμάτων.

Παράλληλα θα μπορούσε να ελεγχθεί κατά πόσο η εφαρμογή των P. Ravikumar et al [22] θα μπορούσε να υλοποιηθεί με τέτοιον τρόπο ώστε να επιτυγχάνει την διασύνδεση των εγγραφών από διαφορετικές Βάσεις Δεδομένων, και όχι μόνο την ανάκτηση πληροφορίας.

Συνοψίζοντας, καταλήγουμε στο συμπέρασμα ότι σιγά σιγά αναπτύσσονται αρκετές μέθοδοι που είναι σε θέση να επιτύχουν την διατήρηση της εμπιστευτικότητας των πληροφοριών, κατά την ενοποίηση των δεδομένων. Πλέον, ο έλεγχος τους υπό πραγματικές συνθήκες θα αποδείξει αν είναι δυνατόν να προφυλάξουμε την

ευαισθησία των πληροφοριών σε τέτοιο βαθμό ώστε να μην αποτρέπεται η επιτυχής ενοποίηση τους.

Βιβλιογραφία

[1] M. Atallah, H. Elmongui, V. Deshpande, and L. Schwarz, "Secure supply-chain protocols", in *IEEE International Conference on E-Commerce,* Newport Beach, California, June 24-27 2003, pp. 293-302.

[2] N. Bruno, L. Gravano, and A. Marian, "Evaluating Top-k Queries over Web-Accessible Databases", in ACM Transactions on Database Systems TODS archive, volume 29, issue 2, June 2004, pp. 319-362.

[3] P. Christen, "Probabilistic Data Generation for Deduplication and Data Linkage", in *Proceedings of The sixth IDEAL 2005 International Conference,* Brisbane, Australia, July 6-8 2005, pp. 109-116.

[4] P. Christen, and T. Churches, "Febrl: Release 0.3", Documentation, Australian National University, April 6 2005.

[5] T. Churches, and P. Christen, "Blind Data Linkage using n-gram Similarity Comparisons" in *Proceedings of the eighth PAKDD 2004 Conference,* Sydney, Australia, May 26-28 2004, pp. 121-126.

[6] C. Clifton, A. Doan, A. Elmagarmid, M. Kantarcioglu, G. Schadow, D. Suciu, and J. Vaidya, "Privacy-Preserving Data Integration and Sharing", in *Proceedings of DMKD 2004 Conference.* Paris, France, June 13 2004, pp. 19-26.

[7] W. Cohen, "The WHIRL Approach to Integration: An Overview", in *Proceedings of the AAAI-98 Workshop on AI and Information Integration*, AAAI Press, 1998

[8] W. Cohen, P. Ravikumar, and S. E. Fienberg, "A Comparison of String Distance Metrics for Name-Matching Tasks", in *Proceedings of IJCAI 2003 Workshop on Information Integration on the Web,* August 9-10 2003, Acapulco, Mexico, pp. 73-78.

[9] I. Dinur, and K. Nissim, "Revealing Information while Preserving Privacy", in *Proceedings of PODS 2003 Conference.* San Diego, California, USA, June 9-12 2003, pp. 202-210.

[10] M. Elfeky, V. Verykios, and A. Elmagarmid, "TAILOR: A record linkage toolbox", in *Proceedings of the 18th International Conference on Data Engineering ICDE 2002,* San Jose, California, Feb. 2002.

[11] I. Fellegi, and A. Sunter, "A Theory for Record Linkage", in Journal of the American Statistical Society, 1969.

[12] B. Gedik, and L. Liu, "A Customizable k-Anonymity Model for Protecting Location Privacy", in Georgia Institute of Technology – CERCS-04-15 Technical Reports.

[13] L. Gravano, P. G. Ipeirotis, H.V. Jagadish, N. Koudas, S. Muthukrishnan, L. Pietarinen, and D. Srivastava, "Using q-grams in a DBMS for Approximate String Processing", in IEEE Data Engineering Bulletin 24(4), 2001, pp 145-154.

[14] M. Hernandez and S. Stolfo, "The Merge/Perge problem for Large Databases", in *Proceedings of the SIGMOD Conference,* San Jose, 1995

[15] J. A. Hylton, "Identifying and Merging Related Bibliographic Records", in MIT Technical Report TR-678, June 1996.

[16] I. F. Ilyas, W.G. Aref, and A. K. Elmagarmid, "Supporting top-k join queries in relational databases", in *Proceedings of the 29th International Conference on Very Large Databases, VLDB 2003,* Berlin, Germany, pp. 754-765.

[17] M. Kantarcioglu, J. Jin, and C. Clifton, "When do Data Mining Results Violate Privacy?", in *Proceedings of KDD 2004 Conference,* Seattle, Washington, USA, August 22-25 2004, pp. 599-604.

[18] A.J Lait, and B. Randell, "An Assessment of Name Matching Algorithms", Technical Report, Department of Computing Science, University of Newcastle upon Tyne, UK 1993.

[19] G. Pagalos, and I. Maurides, "Security in Informational Systems and Networks", 2002, pp. 187-215.

[20] Paull, "D.L.: A geocoded National Address File for Australia: The G-NAF What, Why, Who and When?", PSMA Australia Limited, Griffith, ACT, Australia, 2003.

[21] L. Philips, "The Double-Metaphone Search Algorithm", C/C++ User's Journal, vol. 18 no. 6, June 2000.

[22] P. Ravikumar, W. Cohen, , and S. E. Fienberg, "A Secure Protocol for Computing String Distance Metrics", in *Proceedings of Workshop on Privacy and Security Aspects of Data Mining ICDM 2004.*

[23] D. Struck, "Don't store my data, Japanese tell government", International Herald Tribune, p.1, August 24-25 2002.

[24] F. Tsui, J. Espino, V. Dato, P. Gesteland, J. Hutman, and M. Wagner, "Technical description of RODS: A real-time public health surveillance system", J AM Inform Association, vol. 10, no. 5, pp. 399-408, September 2003.

[25] J. Vaidya, and C. Clifton, "Privacy-Preserving Association Rule Mining in Vertically Partitioned Data", in *The Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Edmonton, Alberta, Canada, July 23-26 2002, pp. 639-644.

[26] J. Vaidya, and C. Clifton, "Privacy-Preserving k-means Clustering over Vertically Partitioned Data", in *The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* Washington, DC, August 24-27 2003, pp. 206-213.

[27] J. Vaidya, and C. Clifton, "Privacy-Preserving Naïve Bayes Classifier for Vertically Partitioned Data".

[28] V. Verykios, A. Elmagarmid, M. Elfeky, M. Cochinwala and S. Dalal, "On the Completeness and Accuracy of the Record Matching Process", in *Proceedings of the MIT Conference on Information Quality*, Boston, MA, October 2000.

[29] V. Verykios, A. Elmagarmid, and E. Houstis, "Automating the Approximate Record-Matching Process", Information Sciences, vol. 126, July 2000.

[30] V. Verykios, G. Moustakides, and M. Elfeky, "A Bayesian decision model for cost optimal record matching", *The Very Large Data Bases Journal,* vol. 12, no. 1, pp. 28-40, May 2003.

[31] L. Xiong, S. Chitti, and L. Liu, "Topk Queries across Multiple Private Databases", in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems ICDCS 2005,* pp. 145-154.

[32] S. Zhong, Z. Yang, and R. N. Wright, "Privacy-Enhancing k-Anonymization of Customer Data", in *Proceedings of the 24th ACM SIGMOD 2005 symposium on Principles of database systems,* Baltimore, Maryland, USA, pp. 139-147.

[33] The Febrl Project Web site. http://datamining.anu.edu.au/projects/linkage.html.

[34] The TwoFish Cryptographic Algorithm. http://www.twofish.org.

[35] The MD5 Message-Digest Algorithm. http://www.faqs.org/rfcs/rfc1321.html.

[36] The Python programming language Web site. http://www.python.org.

[37] The Perl programming language Web site. http://www.perl.org.

[38] The Perl Programs Library and Documentation Web site. http://www.cpan.org.

[39] The AES cryptosystem Web site. http://www.cryptosystem.net/aes/

[40] The RSA Security System. http://www.rsasecurity.com.

[41] The U.S. Bureau of the Census Web site. http://www.census.gov/

[42] The G-NAF Web site. http://www.g-naf.com.au/

Παράρτημα Α

Στο συνοδευτικό DVD περιέχονται όλα τα προγράμματα και το λογισμικό που χρησιμοποιήθηκαν κατά την έρευνα. Μάλιστα τα προγράμματα που υλοποιήθηκαν και περιγράφονται στο κεφάλαιο 5, παρατίθενται και στο παράρτημα Β.

Συγκεκριμένα στον φάκελο \Programs συμπεριλαμβάνονται τα perl αρχεία που υλοποιήθηκαν για την εφαρμογή του πρωτοκόλλου της τεχνικής των Ν-γραμμάτων. Τα αρχεία αυτά είναι:

create_ngrams.plx : Υπολογίζει τα Ν-γράμματα για κάθε πεδίο των εγγραφών μιας Βάσης Δεδομένων.

encr_ngrams.plx : Κρυπτογραφεί τα Ν-γράμματα που έχουν παραχθεί από το προηγούμενο πρόγραμμα

sort_encrypted.plx : Ταξινομεί τα κρυπτογραφημένα Ν-γράμματα με βάση τον κωδικό εγγραφής.

comp_ngrams.plx : Δέχεται σαν είσοδο δυο αρχεία με Ν-γράμματα και κάνει την σύγκριση τους.

Προκειμένου κάποιος να εφαρμόσει το πρωτόκολλο πρέπει να εκτελέσει τα παραπάνω προγράμματα με την συγκεκριμένη σειρά δίνοντας στον kernel τις παρακάτω εντολές.

1. perl create_ngrams.plx inputA.csv outputA.txt N

Δέχεται σαν παραμέτρους το πρώτο σύνολο δεδομένων, το αρχείο με τα Ν-γράμματα, και το μήκος των Ν-γραμμάτων.

2. perl create_ngrams.plx inputB.csv outputB.txt N

Ομοίως για το δεύτερο σύνολο δεδομένων. Το Ν πρέπει να είναι ίδιο και για τις δύο περιπτώσεις.

3. perl encr_ngrams.plx outputA.txt 16 a123456789123456

Δέχεται σαν παραμέτρους το αρχείο με τα N-γράμματα, το μήκος του κλειδιού και το κλειδί κρυπτογράφησης.

4. perl encr_ngrams.plx outputB.txt 16 a123456789123456

Ομοίως για το αρχείο με τα N-γράμματα του δεύτερου συνόλου.

5. perl sort_encrypted.plx croutputA.txt
6. perl sort_encrypted.plx croutputB.txt

Δέχεται σαν παράμετρο τα κρυπτογραφημένα αρχεία.

7. perl comp_ngrams.plx scroutputA.txt scroutputB.txt skeyinputA.txt skeyinputB.txt results.txt 13

Δέχεται σαν παραμέτρους τα δύο ταξινομημένα κρυπτογραφημένα αρχεία με τα N-γράμματα, τα δυο ταξινομημένα αρχεία με τα κλειδιά των εγγραφών, το αρχείο όπου θα αποθηκευτούν τα αποτελέσματα και τον αριθμό των πεδίων που έχει κάθε εγγραφή.

Στον φάκελο \Software περιλαμβάνονται τα αρχεία για την εγκατάσταση των γλωσσών προγραμματισμού Perl και Python καθώς και το λογισμικό Cygwin. Η γλώσσα Python χρειάζεται για να λειτουργήσει το Febrl. Κανονικά τα προγράμματα εκτελούνται σε περιβάλλον Linux/Unix. Παράλληλα όμως είναι δυνατόν να εκτελεστούν και σε περιβάλλον Windows με την χρήση του λογισμικού Cygwin. Οι εντολές για την χρήση του είναι ίδιες με αυτές που χρησιμοποιούνται στο Unix. Η εγκατάσταση των παραπάνω είναι απλή, αλλά σε περίπτωση δυσκολίας υπάρχουν τα αρχεία readme.txt που διαθέτουν οδηγίες και διάφορες πληροφορίες.

Στον φάκελο \Febrl υπάρχει ένα συμπιεσμένο αρχείο που περιλαμβάνει το λογισμικό Febrl. Αυτό για να λειτουργήσει πρέπει να αποσυμπιεστεί μέσα στον φάκελο που είναι εγκατεστημένη η γλώσσα προγραμματισμού Python. Στο φάκελο υπάρχει και το αρχείο febrldoc-0.3.pdf που αποτελεί τον οδηγό για την χρησιμοποίηση των διαφόρων εφαρμογών του λογισμικού.

Στον φάκελο \Crypto υπάρχουν τα modules του αλγορίθμου κρυπτογράφησης και της συνάρτησης κατακερματισμού. Αυτά για να χρησιμοποιηθούν πρέπει να εγκατασταθούν μέσα στο Cygwin.

Παράρτημα Β

Στη συνέχεια παρατίθεται ο κώδικας των προγραμμάτων που υλοποιήθηκαν.

Πρόγραμμα create_ngrams.plx:

```
#!/usr/bin/env perl
#use diagnostics;
use strict;

my $time1=time;
my $cnt=0;
my $numrecs = 0;
# The mask cache for the powerset_iter().
my @_powerset_iterate_mask = ( );

print "input file: $ARGV[0], output file: $ARGV[1], ngram= $ARGV[2]";
my $infile = $ARGV[0];
my $ngramfile = $ARGV[1];
my $ngram = $ARGV[2];
my $outkey = "key";
        $outkey .= "$infile";
open (INFILE, $infile);
open (OUTFILE, ">output.txt");
field)
open (KEYFILE, ">$outkey");
        while (<INFILE>)
        {

        if ($numrecs == 0) { $numrecs++; next;}
    s/\s*//g;
        $_=lc;
        my @recArray = split(/,\s*/);
    my $i;
```

```
        for ($i = 1; $i < $#recArray+1; $i++)
        {
                if ( $recArray[$i] ne ""){
                print OUTFILE "$recArray[0];$i $recArray[$i]\n";}

                else {
                print OUTFILE "$recArray[0];$i  \n";}
                }
                print KEYFILE "$recArray[0]\n";

        $numrecs++;
                }
                print "\nread :$numrecs records\n";
close INFILE;
close OUTFILE;
close KEYFILE;
                open (INFILE, "output.txt");
                open (OUTFILE, ">sortoutput.txt");
                print OUTFILE sort  <INFILE>;
close INFILE;
close OUTFILE;
                my $outkey1 = "s";
                $outkey1 .= "$outkey";
                open (INKEYFILE, $outkey);
                open (OUTKEYFILE, ">$outkey1");
                print OUTKEYFILE sort <INKEYFILE>;
close INKEYFILE;
close OUTKEYFILE;



open (INFILE, "sortoutput.txt");
open (OUTNGRAM, ">$ngramfile");
                while (<INFILE>)
                {
        my @fieldsArray = split /\s/;
                my $reckey = $fieldsArray[0];
                if (not defined $fieldsArray[1]) {
                print OUTNGRAM "$reckey  \n";
                $cnt++;
        next;}

                if ( length($fieldsArray[1]) < $ngram+1 ) {
```

```
        print OUTNGRAM "$reckey $fieldsArray[1]\n";
         $cnt++;
         next;}


    my @fieldngram = "";
    my @fieldattr = split /\s*/, $fieldsArray[1];
         my $len = $#fieldattr+1;
         my $l = 0;
         my $j;
for ($j = 0; $j < $len-$ngram+1; $j++)
         {
         my $k = $l;
         while($k<$ngram+$l) {
        $fieldngram[$j] .= "$fieldattr[$k++]";
                                                }
                $l++;
        }
         my @sortfieldngram = "";
         @sortfieldngram = sort  @fieldngram;

         my @remsortfieldngram = "";
         my $i1=0;
         my $i2=0;
         for ($i1 = 0; $i1 < $#sortfieldngram ; $i1++)
         {
         if ($sortfieldngram[$i1] eq $sortfieldngram[$i1+1]){ next; }
         $remsortfieldngram[$i2++] = $sortfieldngram[$i1];
         }
         if ($i1 == $#sortfieldngram ){
         $remsortfieldngram[$i2++] = $sortfieldngram[$i1];}
         if ( $#remsortfieldngram < 1 ) {
         print OUTNGRAM "$reckey $remsortfieldngram[0]\n";$cnt++; next;}


        my %hash = ($remsortfieldngram[0] => 1);
        my $i3;
        for ($i3=1; $i3 < $#remsortfieldngram+1; $i3++) {
        %hash = (%hash, $remsortfieldngram[$i3] => $i3+1);}
        my $a = \%hash;
        my $pi = powerset_iterate( $a );
        my $psvar = sos_as_string( $pi );
        $psvar =~ s/^{{+//;
        $psvar =~ s/^}{+//;
```

```
            $psvar =~ s/}}+$//;
            $psvar =~ s/{}+//;
            my @psArray = split /} {*/, $psvar;
            my @pssorted = sort @psArray;
            foreach (@pssorted) {
            print OUTNGRAM "$reckey $_\n";

            $cnt++;
                                                    }
            print  "$reckey \n";     #write key kai ngram


            }
            print "total writen ngrams ", $cnt,  "\n";
            close INFILE;
        close OUTNGRAM;
my $time2=time;
my $time = $time2 - $time1;
print "total time in secs $time";

sub powerset_iterate {
   my $set = shift;

   my @keys     = keys   %{ $set };
   my @values   = values %{ $set };
   my $nmembers = @keys;
   my $nsubsets = 1 << $nmembers;
   my ( $i, $j, $powerset, $subset );

   if ( $nmembers >  @_powerset_iterate_mask ) {
      for ( $j = @_powerset_iterate_mask; $j < $nmembers; $j++ ) {
       push( @_powerset_iterate_mask, 1 << $j );
      }
   }

   for ( $i = 0; $i < $nsubsets; $i++ ) {
      $subset = { };
      for ( $j = 0; $j < $nmembers; $j++ ) {
         $subset-> { $keys[ $j ] } = $values[ $j ]
            if $i & $_powerset_iterate_mask[ $j ];
      }
      $powerset-> { $subset } = $subset;
   }
```

```
    return $powerset;
}

sub sos_as_string ($;$) {
    my ( $set, $string ) = @_;

    $$string .= '{';

    my $i;

    foreach my $key (sort keys %{ $set } ) {
        $$string .= " if $i++;
        if ( ref $set-> { $key } ) {
            sos_as_string( $set-> { $key }, $string );
        } else {
            $$string .= $key;
        }
    }

    return $$string .= '}';
}
```

Πρόγραμμα encr_ngrams.plx

```
#!/usr/bin/env perl
#use warnings;
#use strict;
use Crypt::Twofish;
use Digest::MD5 qw(md5 md5_hex);

my $time1=time;
my $numrecs = 0;
print "input file: $ARGV[0], keylength: $ARGV[1], key: $ARGV[2] ";
my $infile = $ARGV[0];
my $keylength = $ARGV[1];
my $key = $ARGV[2];
my $outcrypt = "cr";
            $outcrypt .= "$infile";

open (INFILE, $infile);
open (OUTFILE, ">$outcrypt");
            while (1)
            {
```

```
            my $line= readline INFILE;
            $_=$line;
            my @recArray = split(/;|\s/);
      my $plaintext=$recArray[2];
            my $ciphertext = cryptr($key,$keylength,$plaintext);
            print OUTFILE "$recArray[0];$recArray[1] $ciphertext\n";
      $numrecs++;
            if (eof INFILE == 1){print "\nlast line, $line";
            last;}

            }
                  print "\nread :$numrecs records\n";
close INFILE;
close OUTFILE;
my $time2=time;
my $time = $time2 - $time1;
print "total time in secs $time";

sub cryptr {
my $key=shift;
my $keylength=shift;
my $plaintext=shift;
      my $ciphertext = Encipher($key,$keylength,$plaintext);
      my $digest = md5_hex($ciphertext);
return $digest;
}
```

Πρόγραμμα sort_encrypted.plx

```
#!/usr/bin/env perl
#use warnings;
use strict;

my $time1=time;
print "input file: $ARGV[0] ";
my $infile = $ARGV[0];

my $outcrypt = "s";
            $outcrypt .= "$infile";

            open (INGRAM, "$infile");
            open (OUTCRYPTFILE, ">$outcrypt");
```

```
        print OUTCRYPTFILE sort <INGRAM>;

close INGRAM;
close OUTCRYPTFILE;
my $time2=time;
my $time = $time2 - $time1;
print "\ntotal time in secs $time";
```

Πρόγραμμα comp_ngrams.plx

```
#!/usr/bin/env perl
#use warnings;
use strict;

my $time1=time;
my $m=0;
my $cntA=0;
my $cntB=0;
my $cntc=0;
my $numrecsA = 0;
my $numrecsB = 0;
my $keynfA;
my $keynfB;
my $knA;
my $knB;
my $fgramA;
my $fgramB;

print "enter outngramA outngramB sortkeyinputA sortkeyinputB results nofld \n";
my $outngramA = shift;
my $outngramB = shift;
my $fkeyA = shift;
my $fkeyB = shift;
my $results = shift;
my $nofields = shift;
my $ifl;
open (RESULTS, ">$results");
        for ($ifl = 1; $ifl <= $nofields; $ifl++)
        {
open (KEY_A, $fkeyA);
   while (<KEY_A>)
        {
```

```
            chomp;
            $keynfA = $_;
            $keynfA .= $ifl;

open (KEY_B, $fkeyB);
OUTB:
   while (<KEY_B>)
            {
            chomp;
            $keynfB = $_;
            $keynfB .= $ifl;

open (NGRAM_A, $outngramA);
open (NGRAM_B, $outngramB);
NGRA:
   while (<NGRAM_A>)
            {
     $numrecsA++;
            my @file = split(/;|\s/);
            $knA =$file[0] . $file[1];
            $fgramA = $file[2];
            next if ($knA lt $keynfA);
            if ($knA eq $keynfA) {
            $cntA++;
            while (<NGRAM_B>)
                {
            $numrecsB++;
            my @file = split(/;|\s/);
            $knB =$file[0] . $file[1];
            $fgramB = $file[2];
            next if ($knB lt $keynfB);
            if ($knB eq $keynfB) {
            $cntB++;
INNGRA:
            next if ($fgramA gt $fgramB);
            if ($fgramA eq $fgramB) {
            $cntc++;
            next NGRA;}
            else {
            while (<NGRAM_A>)
                {
            $numrecsA++;
            my @file = split(/;|\s/);
            $knA =$file[0] . $file[1];
```

```
$fgramA = $file[2];
if ($knA eq $keynfA) {
$cntA++;
goto INNGRA;          }
else {
while (<NGRAM_B>)
    {
$numrecsB++;
my @file = split(/;|\s/);
$knB =$file[0] . $file[1];
if ($knB eq $keynfB) {
$cntB++;
next;}
my $scorengr = 2 * $cntc/($cntA + $cntB);

if ($scorengr >= $m)
{print RESULTS "$keynfA $keynfB $cntc $cntA $cntB $scorengr\n";      }
$cntA = 0;
$cntB = 0;
$cntc = 0;
close NGRAM_A;
close NGRAM_B;
next OUTB;
}
}
}
}
}
else {
while (<NGRAM_A>)
    {
$numrecsA++;
my @file = split(/;|\s/);
$knA =$file[0] . $file[1];
if ($knA eq $keynfA) {
$cntA++;}
        }
my $scorengr = 2 * $cntc/($cntA + $cntB);

if ($scorengr >= $m)
{print RESULTS "$keynfA $keynfB $cntc $cntA $cntB $scorengr\n";}
$cntA = 0;
$cntB = 0;
$cntc = 0;
```

```
close NGRAM_A;
close NGRAM_B;
next OUTB;
}
}
while (<NGRAM_A>)
    {
$numrecsA++;
my @file = split(/;|\s/);
$knA =$file[0] . $file[1];
if ($knA eq $keynfA) {
$cntA++;
next;}
}
}
else {
        while (<NGRAM_B>)
                {
                $numrecsB++;
                my @file = split(/;|\s/);
                $knB =$file[0] . $file[1];
                if ($knB eq $keynfB) {
                $cntB++;
                next;}
                my $scorengr = 2 * $cntc/($cntA + $cntB);

                if ($scorengr >= $m)
                {print RESULTS "$keynfA $keynfB $cntc $cntA $cntB
$scorengr\n";}
                $cntA = 0;
                $cntB = 0;
                $cntc = 0;
          close NGRAM_A;
                close NGRAM_B;
                next OUTB;
                }
        }
}
while (<NGRAM_B>)
        {
        $numrecsB++;
        $cntB++;
        }
my $scorengr = 2 * $cntc/($cntA + $cntB);
```

```
            if ($scorengr >= $m)
            {print RESULTS "$keynfA $keynfB $cntc $cntA $cntB $scorengr\n";}
            $cntA = 0;
            $cntB = 0;
            $cntc = 0;
            close NGRAM_A;
            close NGRAM_B;
    }
            print "read B: $numrecsB, field $ifl";
            print " read A: $numrecsA, field $ifl \n";
            $numrecsA=0;
            $numrecsB=0;
            close KEY_B;
    }
            $numrecsA=0;
            $numrecsB=0;
            close KEY_A;
    }
            close RESULTS;
my $time2=time;
my $time = $time2 - $time1;
print "total time in secs $time";
```