ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΜΑΤΙΚΗΣ

ΑΝΑΠΤΥΞΗ ΠΡΟΓΡΑΜΜΑΤΟΣ ΕΝΤΟΠΙΣΜΟΥ ΛΟΓΟΚΛΟΠΩΝ ΣΕ ΕΛΛΗΝΙΚΑ ΔΟΚΙΜΙΑ

Πτυχιακή εργασία

Αλέξανδρος Μπάγιος

Αθήνα, 2017



ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΜΑΤΙΚΗΣ

Τριμελής Εξεταστική Επιτροπή

Γεώργιος Δημητρακόπουλος

Καθηγητής, Τμήμα Πληροφορικής και Τηλεματικής, Χαροκόπειο Πανεπιστήμιο

Μάρα Νικολαΐδου

Καθηγητής, Τμήμα Πληροφορικής και Τηλεματικής, Χαροκόπειο Πανεπιστήμιο

Χρύσα Σοφιανοπούλου

Καθηγητής, Τμήμα Πληροφορικής και Τηλεματικής, Χαροκόπειο Πανεπιστήμιο

Ο Αλέξανδρος Μπάγιος

δηλώνω υπεύθυνα ότι:

1) Είμαι ο κάτοχος των πνευματικών δικαιωμάτων της πρωτότυπης αυτής εργασίας και από όσο γνωρίζω η εργασία μου δε συκοφαντεί πρόσωπα, ούτε προσβάλει τα πνευματικά δικαιώματα τρίτων.

2) Αποδέχομαι ότι η ΒΚΠ μπορεί, χωρίς να αλλάξει το περιεχόμενο της εργασίας μου, να τη διαθέσει σε ηλεκτρονική μορφή μέσα από τη ψηφιακή Βιβλιοθήκη της, να την αντιγράψει σε οποιοδήποτε μέσο ή/και σε οποιοδήποτε μορφότυπο καθώς και να κρατά περισσότερα από ένα αντίγραφα για λόγους συντήρησης και ασφάλειας.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή της πτυχιακής μου εργασίας κο Γεώργιο Δημητρακόπουλο για την πολύ καλή συνεργασία που είχαμε αλλά και τη μεγάλη υπομονή που έδειξε σε όλη τη διάρκεια της εργασίας. Επίσης, θα ήθελα να ευχαριστήσω τα μέλη της τριμελής επιτροπής, κα Μάρα Νικολαΐδου και κα Χρύσα Σοφιανοπούλου, για το ενδιαφέρον που δείξανε στην αξιολόγηση της προσπάθειάς μου αλλά και κατά τη διάρκεια των σπουδών μου. Τέλος, οφείλω ένα μεγάλο ευχαριστώ στην οικογένειά μου για τη στήριξη και τη συμπαράσταση που μου προσφέρει, όπως και στους φίλους μου που με στήριξαν και συνεχίζουν να με στηρίζουν στην προσωπική και ακαδημαϊκή σταδιοδρομία μου.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ....10
ABSTRACT....11
ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ....12
ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ....13
1.ΕΙΣΑΓΩΓΗ....15
1.1.ΚΙΝΗΤΡΟ ΕΡΓΑΣΙΑΣ....15
1.2.ΣΚΟΠΟΣ ΕΡΓΑΣΙΑΣ....16
1.3.ΔΟΜΗ ΕΡΓΑΣΙΑΣ....18
2.ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΑΝΑΣΚΟΠΙΣΗ....20
2.1.CONTEXT....20
2.2.CONTEXT MATCHING....23
2.3.ΠΕΡΙΓΡΑΦΗ ΣΥΣΤΗΜΑΤΟΣ....27
2.3.1.ΕΠΕΞΕΡΓΑΣΙΑ ΕΙΣΟΔΟΥ....28
2.3.2.ΑΝΑΖΗΤΗΣΗ ΣΤΟ ΕΣΩΤΕΡΙΚΟ ΑΠΟΘΕΤΗΡΙΟ....29
2.3.3.ΑΝΑΖΗΤΗΣΗ ΣΤΟ ΔΙΑΔΙΚΤΥΟ....30
2.4.ΔΩΡΕΑΝ ΛΟΓΙΣΜΙΚΟ/ΛΟΓΙΣΜΙΚΟ ΑΝΟΙΧΤΟΥ ΚΩΔΙΚΑ ΣΤΟΝ ΕΝΤΟΠΙΣΜΟ ΛΟΓΟΚΛΟΠΩΝ....31
3.ΠΡΑΚΤΙΚΟ ΜΕΡΟΣ....35
3.1.ΒΑΣΙΚΑ ΤΜΗΜΑΤΑ....35
3.1.1.ΕΙΣΟΔΟΣ ΚΑΙ ΕΞΑΓΩΓΗ ΚΕΙΜΕΝΟΥ....39
3.1.2.ΑΝΑΖΗΤΗΣΗ ΦΡΑΣΗΣ ΣΤΗ ΒΑΣΗ REPO....42
3.1.3.ΑΝΑΖΗΤΗΣΗ ONLINE....45
3.1.3.1.GOOGLE SEARCH....48
3.1.3.2.BING SEARCH....49
3.1.3.3.YAHOO SEARCH....50
3.1.3.4.ΕΝΗΜΕΡΩΣΗ ΤΗΣ ΒΑΣΗΣ REPO....51
3.2.ΕΡΜΗΝΕΙΑ ΑΠΟΤΕΛΕΣΜΑΤΩΝ....53
4.ΣΥΜΠΕΡΑΣΜΑΤΑ....54
4.1.ΑΠΟΔΕΙΞΗ ΩΣ ΠΡΟΣ ΤΗΝ ΤΕΧΝΙΚΗ CONTEXT-MATCHING....55
4.2.ΑΠΟΔΕΙΞΗ ΩΣ ΠΡΟΣ ΤΟΝ ΕΝΤΟΠΙΣΜΟ ΛΟΓΟΚΛΟΠΩΝ....57
4.3.ΒΕΛΤΙΩΣΕΙΣ....59
5.ΒΙΒΛΙΟΓΡΑΦΙΑ....63
6.ΠΑΡΑΡΤΗΜΑ....65

ΠΕΡΙΛΗΨΗ

Συστήματα σχεδιασμένα ώστε η λειτουργία τους να βασίζεται στην αξιοποίηση της 'πλαισιακής' γνώσης είναι αρκετά και συναντώνται σε διαφορετικούς θεματικούς κλάδους. Με την εκμετάλλευση του τρόπου λειτουργίας αυτού του είδους αρχιτεκτονικής γίνεται να υλοποιηθούν συστήματα τα οποία παρουσιάζουν στοιχεία επαναληπτικότητας κατά την εκτέλεσή τους. Ένα πεδίο εφαρμογής αυτής της αρχιτεκτονικής είναι ο εντοπισμός λογοκλοπών. Αναγνωρίζοντας την επαναληπτικότητα στον τρόπο ταύτισης όμοιων φράσεων μεταξύ πηγών, η εφαρμογή ενός αλγορίθμου αναγνώρισης πλαισίου μπορεί να αποφέρει θετικά αποτελέσματα ως προς την ποιότητα των αποτελεσμάτων που προκύπτουν αλλά και την απόδοση του συστήματος που εκτελεί μία ανάλογη διαδικασία. Ενώ υπάρχει ένας αξιόλογος αριθμός ηλεκτρονικών εργαλείων για τον εντοπισμό λογοκλοπών, πολλά από τα οποία παρέχουν κάποιες υπηρεσίες δωρεάν, ο αριθμός των εφαρμογών ανοιχτού λογισμικού είναι μικρός. Επιπροσθέτως, εφαρμογές για τον εντοπισμό λογοκλοπών που να απευθύνονται συγκεκριμένα σε ελληνικά δοκίμια σπανίζουν.

Αντικείμενο της πτυχιακής εργασίας είναι η υλοποίηση μίας εφαρμογής ανοιχτού κώδικα για τον εντοπισμό λογοκλοπών προσαρμοσμένη στην ανάλυση ελληνικών κειμένων με τη χρήση ενός αλγορίθμου ταύτισης πλαισίου. Προκειμένου να λειτουργήσει τελικά αυτό το σύστημα, θα πρέπει να υλοποιηθούν βασικές διεπιφάνειες για την εξαγωγή κειμένου, αναζήτηση φράσεων στο διαδίκτυο και αποθήκευση εγγραφών στη μνήμη του συστήματος. Με την υλοποίηση του προγράμματος για την εξυπηρέτηση των βασικών αναγκών μπορούμε να καταλήξουμε σε συμπεράσματα τόσο για την ποιότητα της ανάλυσης στον τομέα των λογοκλοπών, όσο και για την απόδοση του συγκεκριμένου αλλά κατά επέκταση και κάθε συστήματος που ακολουθεί αυτό το μοντέλο.

Στο σύστημα που αναπτύχθηκε στη γλώσσα Java, υλοποιούνται τεχνικές αναγνώρισης κειμένου και εξαγωγής κειμένου, όπως επίσης και μέθοδοι αναζήτησης φράσεων στο διαδίκτυο χρησιμοποιώντας τις υπηρεσίες των μηχανών αναζήτησης Google, Bing και Yahoo. Συμπεριλαμβάνονται επίσης διαδικασίες για την εξαγωγή κειμένου από διαδικτυακές σελίδες καθώς και τη διατήρηση και ενημέρωση ενός εσωτερικού αποθετηρίου της μορφής μίας σχεσιακής βάσης δεδομένων.

Λέξεις κλειδιά: πλαίσιο, αναγνώριση πλαισίου, αναγνώριση κειμένου, αλγόριθμος αναγνώρισης πλαισίου, λογοκλοπές, μηχανή αναζήτησης, web crawler, web scraping

ABSTRACT

Systems designed to function based on taking advantage of contextual knowledge are common and can be encountered in many different thematic areas. Systems that appear repetitiveness among their tasks could benefit from this kind of architecture. Plagiarism detection techniques could be implemented based on this architectural model. By recognizing the way similar phrases are matched with text from authentic sources, the application of a Context Matching Algorithm could be proved to be a beneficial asset regarding the quality of the results as well as system performance during execution of this procedure. There are plenty of remarkable online tools to handle plagiarism detection and some of them are offering their services for free. However, only a few of them are of open source. In addition to this, it is hard to find applications for plagiarism detection specifically designed for Greek sources.

Subject of this thesis is the implementation of an open source application to handle plagiarism detection specifically for Greek sources. In order for this system to function properly, basic interfaces should be implemented to deal with text extraction, online text searching and storing records in system memory. By developing the program with basic functionality we would be able to analyze the results and reach to valuable conclusions regarding quality on plagiarism detection and system performance. These results are important for considering whether the described model could be applied on more generic systems.

In the context of this computer application which is developed in Java, functions that handle text recognition are implemented. Among others, in the computer program's source code, the following procedures are implemented: text extraction, online text searching by using services provided by known search engines (Google, Bing, Yahoo), web scraping in order to get text from various online sources and configuration/updating of an internal repository designed on a relational database model.

Keywords: context, context matching, text recognition, context matching algorithm, plagiarism, search engine, web crawler, web scraping

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

Εικόνα 2.1.1: Διαδικασία αναγνώρισης DSNPM (Σαατσάκης, 2009).....22
Εικόνα 2.3.1: Διάγραμμα ροής.....27
Εικόνα 2.4.1: Αποτελέσματα εκτέλεσης του ArticleChecker.....32
Εικόνα 2.4.2: Παρουσίαση σύγκρισης του DupliChecker.....33
Εικόνα 2.4.3: Αποτέλεσμα σύγκρισης του WCopyFind.....34
Εικόνα 3.1.1: Υπογραφές μεθόδων κλάσης Control.....37
Εικόνα 3.1.2: Πίνακες της βάσης repo.....38
Εικόνα 3.1.3: Δημιουργία της βάσης repo.....38
Εικόνα 3.1.1.1: Περιεχόμενα του αρχείου crawl.properties.....39
Εικόνα 3.1.1.2: Εκτέλεση του WebCrawler.jar από τη γραμμή εντολών των Windows.....40
Εικόνα 3.1.1.3: Μενού προγράμματος.....40
Εικόνα 3.1.1.4: Επιλογή 'help'.....40
Εικόνα 3.1.2.1: Υπογραφές μεθόδων κλάσης Repo.....43
Εικόνα 3.1.2.2: Αναζήτηση φράσης στα κείμενα του αποθετηρίου.....44
Εικόνα 3.1.3.1: Παράδειγμα αναζήτησης αυτούσιας φράσης και απλής αναζήτησης.....46
Εικόνα 3.1.3.1.1: Εισάγοντας ψευδο-τυχαία χρονική καθυστέρηση 48
Εικόνα 3.1.3.2.1: Παράδειγμα ίδιας περιγραφής σε μορφές JSON και XML.....49
Εικόνα 3.1.3.3.1: Κώδικας σε Java για την διεξαγωγή σύνθετης αναζήτησης Yahoo.....50
Εικόνα 3.1.3.4.1: Κείμενο που ανήκει σε tag <p> ιστοσελίδας (IBM, 2016).....51
Εικόνα 3.1.3.4.2: Κείμενο που ανήκει σε tag <li> ιστοσελίδας (IBM, 2016).....51

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

CMA	Context Matching Algorithm
JSON	JavaScript Object Notation
MS	Microsoft
OCR	Optical Character Recognition
PDF	Portable Document Format
URL	Uniform Resource Locator

1. ΕΙΣΑΓΩΓΗ

1.1. ΚΙΝΗΤΡΟ ΕΡΓΑΣΙΑΣ

Διανύουμε μία περίοδο όπου η πρόσβαση στην πληροφορία είναι πλέων δεδομένη και απλούστατη. Με μεγάλη ευκολία κάποιος μπορεί να αποκτήσει γνώσεις σχετικά με οποιοδήποτε επιστημονικό τομέα με μία συνήθως αρκετά απλή αναζήτηση στο internet. Το γεγονός αυτό έχει ωθήσει στη ραγδαία ανάπτυξη του επιστημονικού υπόβαθρου και προφανώς την δημιουργία ολοένα και περισσότερων και πιο ενημερωμένων γνωστικών αποθετηρίων. Με αυτόν τον τρόπο εξελίσσονται σταθερά όλες οι επιστήμες, καθώς συνεχώς δημιουργούνται νέες πηγές γνώσης. Καθώς αυτή η διαδικασία προχωρεί, η μεγάλη ευκολία στην πρόσβαση πληροφορίας μπορεί πολύ εύκολα να καταχραστεί. Μία πολύ σημαντική μορφή τέτοιας περίπτωσης αποτελεί η παρουσίαση μέρους – αν όχι του συνόλου – κάποιας πληροφορίας ως προϊόν κάποιου προσώπου διαφορετικού του αυθεντικού πνευματικού ιδιοκτήτη. Με αυτόν τον τρόπο ορίζουμε ως λογοκλοπή τη χρήση αυτούσιου μέρους του λόγου ενός τρίτου προσώπου χωρίς αυτή να συνοδεύεται από την απαραίτητη αναγνώριση του πνευματικού ιδιοκτήτη. Όπως αναφέρθηκε νωρίτερα, σε μία εποχή όπου οι πηγές πληροφόρησης είναι πρακτικά αμέτρητες, καταλαβαίνει κανείς πως ο πειρασμός οικειοποίησης δεδομένων είναι αρκετά μεγάλος και συνεπώς, τέτοιες ενέργειες έχουν μείζων αντίκτυπο την παραβίαση της δημοσιογραφικής δεοντολογίας καθώς και της ακαδημαϊκής λειτουργίας εξετάσεων και εργασιών.

Για το σκοπό του εντοπισμού λογοκλοπών έχουν αναπτυχθεί αρκετά ηλεκτρονικά εργαλεία, κάποια από αυτά με ιδιαίτερη ακρίβεια στα αποτελέσματα που σχηματίζουν. Καθώς ο μεγαλύτερος όγκος πληροφορίας βρίσκεται σε πηγές που έχουν συνταχθεί στα αγγλικά, αντίστοιχα τα περισσότερα εργαλεία εντοπισμού λογοκλοπών έχουν υλοποιηθεί με τη βασική παράμετρο της γλώσσας. Το γεγονός ότι εφαρμογές ανοιχτού λογισμικού οι οποίες απευθύνονται σε ελληνικά δοκίμια σπανίζουν, αποτέλεσε το κίνητρο της εργασίας και την υλοποίηση μίας εφαρμογής όπου ο κώδικας να είναι διαθέσιμος προς όλους, να μπορεί να αναλύσει κείμενα των βασικών μορφών, να είναι δωρεάν και να απευθύνεται σε σύνολο ελληνικών πηγών.

Καθώς η ανάλυση ενός κειμένου ως προς την ύπαρξη τμημάτων που θα μπορούσαν να χαρακτηριστούν ως λογοκλοπή είναι το ένα από τα δύο αντικείμενα της εργασίας, το δεύτερο αντικείμενο είναι η εφαρμογή ενός αλγορίθμου ο οποίος θα βασίζεται στην αναγνώριση και ταύτιση τμημάτων της εξεταζόμενης πηγής, με τα διαθέσιμα αρχεία που μπορεί να εντοπίσει αναζητώντας στο internet. Αυτή η ιδιότητα χαρακτηρίζει τον *Context Matching Algorithm (CMA)* ο οποίος έχει εφαρμοστεί. Συνοπτικά, η κεντρική ιδέα με την οποία λειτουργεί ο CMA είναι η δημιουργία μοτίβων από εξεταζόμενες τιμές όπου αποσκοπεί στην αναγνώρισή τους σε μελλοντικές αναζητήσεις.

Τέλος, μεγάλο μέρος της εργασίας αφορά τις τεχνολογικές προδιαγραφές και περιγραφές τεχνικών χαρακτηριστικών που ενσωματώθηκαν στο πρόγραμμα που αναπτύχθηκε. Παρουσιάζονται ηλεκτρονικά εργαλεία τρίτου προσώπου που χρησιμοποιήθηκαν και τεχνικές που εφαρμόστηκαν για την επίλυση διαφόρων προκλήσεων.

1.2. ΣΚΟΠΟΣ ΕΡΓΑΣΙΑΣ

Σκοπός της εργασίας είναι η εφαρμογή ενός αλγορίθμου CMA στον εντοπισμό λογοκλοπών σε ελληνικά κείμενα. Στο πλαίσιο της έρευνας του Τμήματος Πληροφορικής και Τηλεματικής του Χαροκόπειου Πανεπιστημίου, έχει αναπτυχθεί μία εφαρμογή ανοιχτού κώδικα η οποία εξετάζει ένα κείμενο και χρησιμοποιώντας τον κατάλληλο αλγόριθμο αναζητεί αναφορές σε κείμενα τα οποία βρίσκει online για να αναλύσει κατά πόσο θα μπορούσε το αρχικό κείμενο να εμφανίζει ποσοστά ομοιότητας ώστε τελικά να χαρακτηριστεί ως λογοκλοπή.

Το σύστημα το οποίο έχει αναπτυχθεί αναλύει ένα κείμενο που δέχεται σαν είσοδο και το οποίο έχει συνταχθεί στα ελληνικά προσπαθώντας να ταυτίσει το περιεχόμενό του με ηλεκτρονικά διαθέσιμα αρχεία ώστε τελικά να εμφανίσει σε τι ποσοστό υπάρχει ομοιότητα και με ποια από αυτά. Η υλοποίηση της εφαρμογής έχει γίνει σε Java, ενώ παράλληλα γίνεται χρήση βάσης δεδομένων, συνεπώς υπάρχουν κομμάτια σε SQL. Αρχικά το σύστημα σπάει το κείμενο που θα ελέγξει σε φράσεις τις οποίες σε επόμενο βήμα θα αναζητήσει μία-προς-μία για να εντοπίσει πιθανή ταύτιση με άλλα κείμενα. Βασική λειτουργία της εφαρμογής είναι η αναζήτηση στο internet για πηγές με όμοιες φράσεις με το εξεταζόμενο κείμενο

χρησιμοποιώντας κάποια μηχανή αναζήτησης μεταξύ των Google, Bing και Yahoo. Για το σκοπό αυτό, έχει αναπτυχθεί ένας μηχανισμός ανάλυσης αρχείων της μορφής απλού κειμένου (αρχεία απλού κείμενου όπως της επέκτασης TXT), ως και σύνθετης μορφής (DOC/DOCX, PDF) καθώς και χειρισμός ιστότοπων (HTM/HTML, PHP, JSP) για την εξαγωγή κειμένου. Το σύστημα στηρίζεται σε μία σχεσιακή βάση δεδομένων (MySQL) στην οποία αποθηκεύει κάθε άρθρο στο οποίο έχει εντοπίσει κάποια ομοιότητα. Η κύρια ιδέα στην οποία έχει αναπτυχθεί το σύστημα είναι η αναζήτηση στη βάση δεδομένων για ομοιότητες πριν από τη διαδικτυακή αναζήτηση. Η αναζήτηση στη βάση δεδομένων είναι πολύ πιο γρήγορη από την αναζήτηση στο διαδίκτυο, οπότε με αυτό τον τρόπο σε περίπτωση που αναγνωριστεί κάποια ομοιότητα πριν από την αναζήτηση στο internet γίνεται σημαντική εξοικονόμηση πόρων και αποφεύγεται μία τελικά περιττή ενέργεια.

Για τη δημιουργία του λογισμικού αρχικά μελετήθηκαν παρόμοια ηλεκτρονικά εργαλεία ως προς τα διάφορα χαρακτηριστικά τους. Εξετάστηκε η μορφή εισόδου που δέχονται, ο τρόπος με τον οποίο λειτουργούν κατά προσέγγιση και η φύση των αποτελεσμάτων που αντλούν. Σε πιο χαμηλό επίπεδο, έγινε ανάλυση προγράμματος ανοιχτού κώδικα που έχει αναπτυχθεί σε C/C++ ώστε να αναγνωριστεί ο αλγόριθμος που χρησιμοποιεί. Στη συνέχεια μελετήθηκε η φύση του αλγορίθμου που χρησιμοποιήθηκε. Όπως αναφέρθηκε παραπάνω, επιλέχτηκε ένας αλγόριθμος ομοιότητας περιεχομένου. Για την αναζήτηση στο διαδίκτυο μέσω Java χρησιμοποιήθηκαν οι γνωστές μηχανές αναζήτησης Google, Bing και Yahoo, ενώ για την αλληλεπίδρασή τους με τη Java χρησιμοποιήθηκαν διαφορετικοί μηχανισμοί και διεπιφάνειες. Διαφορετικές τεχνικές επίσης εφαρμόστηκαν στον τρόπο αναγνώρισης κειμένου ώστε να είναι δυνατή η διαχείριση των γνωστότερων μορφών αρχείων κειμένου. Τέλος, χρησιμοποιήθηκε βάση δεδομένων MySQL ώστε να αποθηκεύει αρχεία που προκύπτουν κατά την εκτέλεση του προγράμματος και να αποτελέσει τη βάση από την οποία διεξάγονται τα αποτελέσματα.

1.3. ΔΟΜΗ ΕΡΓΑΣΙΑΣ

Η εργασία έχει δομηθεί ως εξής:

Το 2ο κεφάλαιο αποτελεί τη βιβλιογραφική ανασκόπηση. Εκεί περιγράφονται με μεγαλύτερη λεπτομέρεια οι βασικοί όροι που θα χρησιμοποιηθούν σε όλη την έκταση της εργασίας. Ορισμοί που θα μας απασχολήσουν είναι οι: *context*, *context-matching*, *pattern-matching* και *text recognition*. Εκεί γίνεται η ανάλυση κάποιων κύριων εννοιών για την κατανόηση της βασικής αρχής της πτυχιακής εργασίας. Πιο αναλυτικά, ορίζεται η έννοια *context* με μία πιο εκτεταμένη προσέγγιση καθώς και με χαρακτηριστικά παραδείγματα. Αναλύεται επίσης τι σημαίνει *pattern-matching* καθώς και πως συσχετίζεται με το *context-matching*, ποιες γνωστές εφαρμογές χρησιμοποιούν παρόμοια τεχνική, τι άλλες τεχνικές υπάρχουν για την αναγνώριση κειμένου (*text recognition*) και τέλος, παρουσιάζονται κάποιες εφαρμογές που διανέμονται δωρεάν αλλά και ανοιχτού λογισμικού για την αναγνώριση κειμένου και εντοπισμού λογοκλοπών. Εκεί γίνεται η ανάλυση του τρόπου λειτουργίας κάποιων από αυτών.

Το 3ο κεφάλαιο είναι το πρακτικό μέρος. Όροι που αναλύθηκαν στο 2ο κεφάλαιο θα χρησιμοποιηθούν για την περιγραφή της εφαρμογής που έχει αναπτυχθεί. Θα αναλυθούν αρκετές τεχνικές λεπτομέρειες ως προς την υλοποίηση του προγράμματος, την παραμετροποίηση και το σχεδιασμό του (αρχιτεκτονική). Σε αυτό το κεφάλαιο αναφέρεται το σύνολο του επιμέρους έτοιμου λογισμικού το οποίο χρησιμοποιήθηκε, είναι διαθέσιμο δωρεάν και εφαρμόσθηκε στην αναζήτηση σε ιστοσελίδες μέσω Java και εξαγωγή κειμένου από διαφορετικά είδη αρχείων. Σε αυτό το σημείο γίνεται λόγος και για την φύση των αποτελεσμάτων και αναφέρονται κάποια τεχνικά προβλήματα που προέκυψαν κατά την ανάπτυξη της εφαρμογής όπως επίσης και τρόποι που αντιμετωπίστηκαν και μπορούν να ξεπεραστούν. Επίσης, εκεί υπάρχουν χαρακτηριστικές εικόνες από παραδείγματα εκτέλεσης, αποτελέσματα, παραμετροποιήσεις και ερμηνείας τελικών αποτελεσμάτων.

Το 4ο μέρος της εργασίας είναι αυτό που περιέχει συμπεράσματα που προέκυψαν από αυτή την εργασία. Όλοι οι παράγοντες που συντέλεσαν στην υλοποίηση της εφαρμογής θα συνυπολογιστούν ώστε να αξιολογηθεί κατά πόσο μία προσέγγιση όπως αυτή που επιλέχτηκε θα είναι αποτελεσματική, σε τι ποσοστό τα αποτελέσματα προκύπτουν με λογικό τρόπο και

ποια θα είναι τα πλεονεκτήματα και φυσικά τα μειονεκτήματα παρόμοιων υλοποιήσεων σε σχέση με άλλες μεθόδους. Τέλος, αναφέρονται τρόποι και τεχνικές που θα μπορούσαν να εφαρμοστούν μελλοντικά και να συντελέσουν στη βελτίωση του προγράμματος ως προς την απόδοση και την αξιοπιστία των αποτελεσμάτων.

2. ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΑΝΑΣΚΟΠΙΣΗ

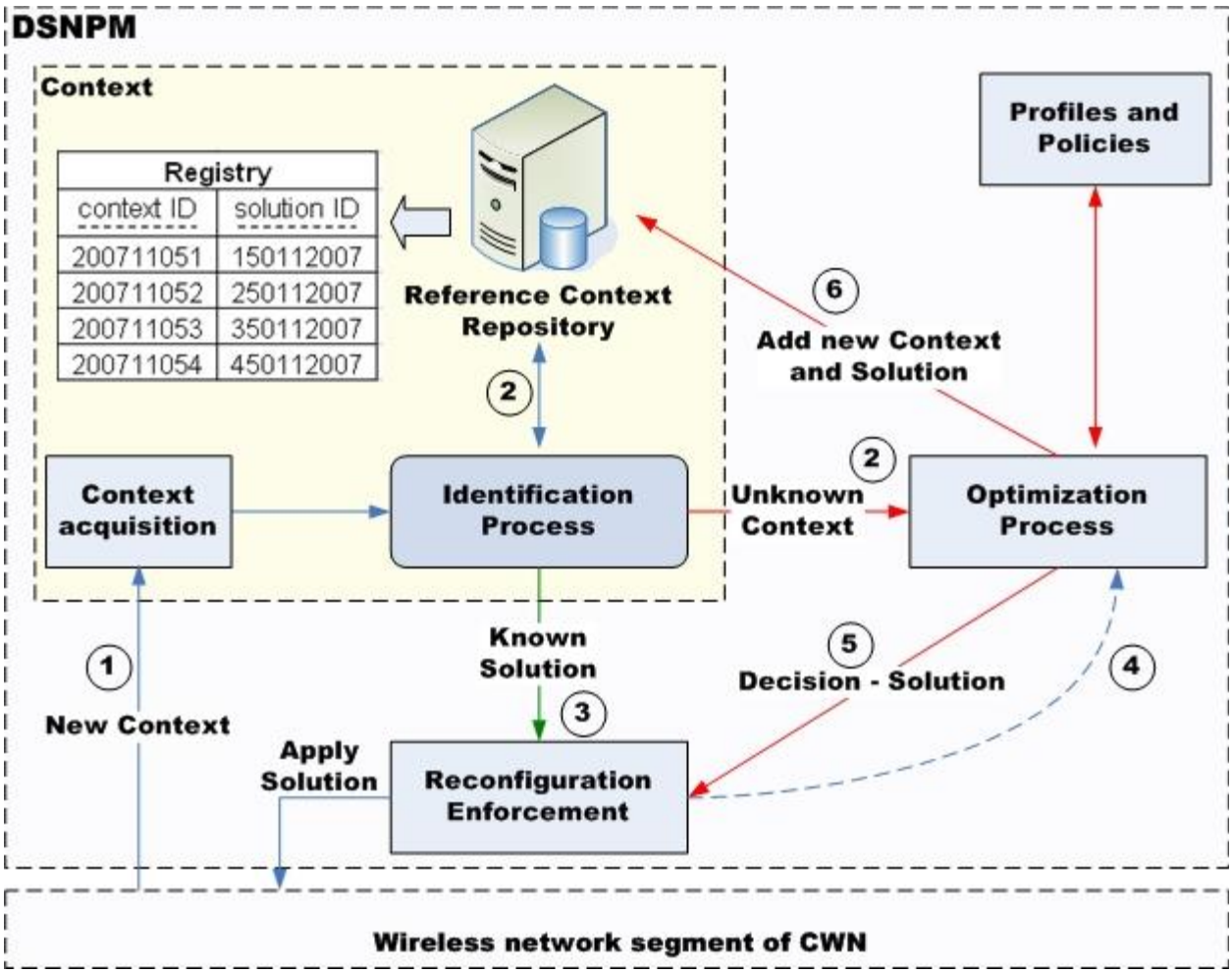
Σε αυτό το κεφάλαιο αναλύονται οι βασικές έννοιες που αποτελούν το σύνολο της ορολογίας που χρησιμοποιείται στην εργασία. Αρχικά ορίζεται η έννοια context με διαφορετικούς τρόπους και παραθέτονται παραδείγματα για την καλύτερη κατανόηση αλλά και την επισήμανση της ευρύτητας των πεδίων που μπορεί να οριστεί. Γίνεται αναφορά στους όρους ταύτιση ομοιότητας (context-matching) και ταύτιση μοτίβου (pattern-matching*)* και εξηγείται η κοντινή σημασία τους, ενώ αναλύεται και ο όρος αναγνώριση κειμένου (text recognition). Επίσης περιγράφεται η τεχνική που χρησιμοποιήθηκε στο λογισμικό που υλοποιήθηκε για να εξυπηρετήσει αυτό το σκοπό. Στο τέλος αυτού του μέρους βρίσκονται αναφορές σε ηλεκτρονικά εργαλεία εντοπισμού λογοκλοπών, περιγράφονται μερικά χαρακτηριστικά παραδείγματα εκτέλεσης και εξετάζονται κάποια από τα πλεονεκτήματα και τα μειονεκτήματα μερικών.

2.1. CONTEXT

Ο όρος *context* ορίζει ένα γενικό πλαίσιο που αποτελείται από μονάδες ή στοιχεία τα οποία συνολικά περιγράφουν μία κατάσταση. Οι μονάδες αυτές στο σύνολο συντελούν στο να παραμετροποιήσουν ένα ευρύ πλαίσιο προσδίδοντας σε αυτό, μεμονωμένα η κάθε μία, τα δικά της χαρακτηριστικά. Η κατανόησή του μπορεί να αλλάξει τελείως την μεθοδολογία με την οποία κάποιος εξάγει αποτελέσματα από ένα δείγμα, καθώς επίσης μπορεί να βοηθήσει σημαντικά στο να αναδείξει τι αντιπροσωπεύει κάθε αριθμός/παράμετρος μίας ευρύτερης εφαρμογής και πώς θα πρέπει να ερμηνευθεί. Σε κάθε έρευνα ή μελέτη, είναι πολύ σημαντικό να γνωρίζουμε τι ακριβώς αντιπροσωπεύει το δείγμα που αναλύουμε πριν προβούμε στην ερμηνεία των αποτελεσμάτων και προκύψουν βιαστικά και κατά επέκταση εσφαλμένα συμπεράσματα. Χωρίς κατάλληλη γνώση για το ευρύτερο πλαίσιο τα δεδομένα ουσιαστικά είναι άχρηστα και κάθε πιθανό σενάριο εξαγωγής αποτελεσμάτων δεν θα είναι τεκμηριωμένο. Θα πρόκειται για χρήση των δεδομένων ως απλές τιμές, αγνοώντας το αν αντιπροσωπεύουν κάτι όπως κάποια παράμετρο, κάποιο βαθμό δυσκολίας για έναν έλεγχο, ένα συντελεστή

επιτάχυνσης μίας διαδικασίας, ένα σημείο που καθιστά ακόμη και αδύνατη την ολοκλήρωση κάποιου πειράματος.

Στον τομέα των κινητών συσκευών, οι Wei Liu, Xue Li και Daoli Huang, οι οποίοι εξετάζουν την ιδιότητα *context-awareness*, ορίζουν ως *context* το περιβάλλον ενός χρήστη υπολογιστικού υλικού όπως η τοποθεσία ή ο τεχνικός εξοπλισμός, παράμετροι οι οποίοι συνεχώς αλλάζουν (Liu, Li, & Huang, 2011). Σε μία αντίστοιχη προσέγγιση παρουσιάζεται η ιδέα ότι η 'πλαισιακή' γνώση (contextual knowledge) μπορεί να χρησιμοποιηθεί για την ταχύτερη και αποτελεσματικότερη αναδιοργάνωση ενός δικτύου και την προσαρμογή του στις διαφορετικές συνθήκες του περιβάλλοντος (Σαατσάκης, 2009). Πιο συγκεκριμένα, για ένα δίκτυο κινητής τηλεφωνίας η 'πλαισιακή' γνώση προκύπτει από την αναγνώριση των απαραίτητων παραμέτρων μέσα σε ένα γενικό πλαίσιο. Οι απαραίτητες παράμετροι μπορεί να αφορούν διάφορες καταστάσεις συνδρομητών που εξυπηρετούνται από το δίκτυο, όπως για παράδειγμα συνδρομητές που βρίσκονται στη μέση μίας κλήσης ή χρησιμοποιούν πακέτα δεδομένων για τη σύνδεσή τους σε κάποιο κοινωνικό δίκτυο. Ένα δείγμα τεχνικών αναγνώρισης πλαισίου παρουσιάζεται από τους M. J. Van Sinderen, A. T. Van Halteren, M. Wegdam, H. B. Meeuwissen και E. Henk Eertink στο άρθρο του IEEE Communications Magazine (Sinderen, Halteren, Wegdam, Meeuwissen, & Eertink, 2006). Με την εφαρμογή των κατάλληλων τεχνικών και σε συνδυασμό με ένα μηχανισμό άντλησης αποτελεσμάτων, περιγράφεται η αρχιτεκτονική DSNPM (Dynamic Self-Organizing Network Planning and Management). Η αρχιτεκτονική αυτή προσδίδει σε ένα δίκτυο τη δυνατότητα να “αποφασίζει” μέχρι και μακροπρόθεσμα σχετικά με τον τρόπο που θα αναδιοργανώσει κάποιο τμήμα του ανάλογα με το εύρος των διαφορετικών αρχικών καταστάσεων που πρέπει να εξυπηρετήσει κάθε φορά. Σε συνδυασμό με την εφαρμογή τεχνικών βελτιστοποίησης της λειτουργίας και την ενίσχυση της αρχιτεκτονικής με τεχνικές εκμάθησης, επιτυγχάνεται βελτιστοποίηση της λειτουργικότητας (Nolte, et al., 2010). Στην Εικόνα 2.1.1 παρουσιάζεται ο τρόπος με τον οποίο το δίκτυο της αρχιτεκτονικής DSNPM χειρίζεται την επαναληπτική συμπεριφορά του περιβάλλοντος του δικτύου.



Εικόνα 2.1.1: Διαδικασία αναγνώρισης DSNPM (Σαατσάκης, 2009)

Η διαδικασία αναγνώρισης που ορίστηκε λειτουργεί ως εξής:

Αρχικά, το δίκτυο δέχεται ένα νέο χαρακτηριστικό πλαίσιο σαν είσοδο το οποίο πρέπει να εξυπηρετήσει. Σαν πρώτο βήμα, χρησιμοποιούνται μέθοδοι για την ανάγνωση του πλαισίου. Αφού το πλαίσιο έχει αναλυθεί, αρχίζει μία διαδικασία αναγνώρισης της εισόδου. Η διαδικασία αυτή χρησιμοποιεί ένα αποθετήριο το οποίο περιέχει πληροφορίες σχετικά με το ποιος τρόπος διαμόρφωσης χρησιμοποιήθηκε σε κάθε προηγούμενη είσοδο. Σε περίπτωση που το πλαίσιο της εισόδου αναγνωριστεί σαν κάποιο που έχει εξυπηρετηθεί στον παρελθόν, τότε ανακαλείται ο τρόπος διαμόρφωσης μέσω του αποθετηρίου και το δίκτυο διαμορφώνεται άμεσα στην κατάσταση που είχε εξυπηρετήσει το παρόμοιο πλαίσιο εισόδου. Στην περίπτωση που το πλαίσιο εισόδου δεν αναγνωρίζεται ως κάποιο που έχει ήδη εξυπηρετηθεί, τότε ξεκινάει η διαδικασία βελτιστοποίησης ώστε να βρεθεί μία νέα μέθοδος η οποία θα εφαρμοστεί στην εξυπηρέτηση της νέας εισόδου. Η μέθοδος αυτή χρησιμοποιείται στη διαμόρφωση του δικτύου αλλά επίσης εισάγεται στο αποθετήριο ων ένας ακόμα τρόπος διαμόρφωσης του δικτύου της συγκεκριμένης περίπτωσης. Με αυτό τον τρόπο το αποθετήριο

παραμένει ενημερωμένο και σε μελλοντική αναγνώριση της ίδιας κατάστασης από το δίκτυο, θα γίνει άμεσα χρήση της ήδη γνωστής διαμόρφωσης. Έτσι το δίκτυο κερδίζει το χρόνο που θα περνούσε μέχρι να επεξεργαστεί ξανά η ίδια κατάσταση.

2.2. CONTEXT MATCHING

Η μεθοδολογία που παρουσιάστηκε στο προηγούμενο παράδειγμα, μπορεί αναλογικά να μεταφραστεί σε μία αντίστοιχη μεθοδολογία για την εφαρμογή στην αναγνώριση όμοιων τμημάτων μεταξύ δύο κειμένων. Σαν αποτέλεσμα, συγκρίνοντας δύο κείμενα και αναγνωρίζοντας ένα μεγάλο αριθμό από ομοιότητες, μπορεί να τεθεί το θέμα της αξιολόγησης της γνησιότητας ενός κειμένου και σήμανση λογοκλοπής. Για να προσεγγίσουμε περισσότερο το θέμα της λογοκλοπής κειμένου, χρειάζεται να είναι ξεκάθαρος ο τρόπος που αρμόζει να εφαρμοσθεί ώστε να χαρακτηριστούν τμήματα κειμένων ως όμοια.

Ένα κείμενο αποτελείται από ένα σύνολο λέξεων οι οποίες συνολικά οργανώνονται σε φράσεις, προτάσεις, παραγράφους και κεφάλαια. Συγκρίνοντας δύο κείμενα, σίγουρα κάποιος θα εντοπίσει ένα μεγάλο σύνολο από λέξεις που επαναλαμβάνονται αρκετές φορές και στα δύο από αυτά. Είναι ξεκάθαρο πως ο χαρακτηρισμός ενός κειμένου ως προϊόν λογοκλοπής για το λόγο ότι περιέχει αρκετές λέξεις που υπάρχουν και σε κάποια άλλη αυθεντική πηγή είναι λάθος. Για την προσθήκη ουσιαστικής λειτουργικότητας σε μία διαδικασία εντοπισμού λογοκλοπής δεν αρκεί να εντοπιστούν κάποιες λέξεις, αλλά να γίνει αναγνώριση όμοιων τμημάτων. Αυτό επιτυγχάνεται με την οργάνωση των λέξεων σε φράσεις. Με την αναζήτηση αυτούσιων φράσεων σε ένα κείμενο λαμβάνουμε περισσότερο ουσιαστική πληροφορία. Στην περίπτωση που εντοπιστεί ένας σημαντικός αριθμός όμοιων αυτούσιων φράσεων σε δύο κείμενα είναι δυνατό να ακολουθήσει μία εκτενέστερη ανάλυση για να διαπιστωθεί η ύπαρξη ή όχι λογοκλοπής.

Το πρώτο βήμα για τον εντοπισμό φράσεων, είναι η αναγνώριση κειμένου (text-recognition). Οι Rohini Salunke, Dipali Badhe, Vanita Doke, Yogeshwari Raykar, Prof. Bhushan S. Thakar ορίζουν την έννοια αναγνώρισης χαρακτήρων ως τη διαδικασία ανίχνευσης και τμηματοποίησης χαρακτήρων από μία εικόνα και της μετατροπής τους σε μορφή που μπορεί

να χειριστεί το υπολογιστικό σύστημα, όπως για παράδειγμα σε ASCII (Salunke, Badhe, Doke, & Raykar, 2015). Κάθε κείμενο που υπόκειται σε έλεγχο μπορεί να ανήκει σε αρχείο διαφόρων τύπων. Αρχεία που ενδέχεται να περιέχουν κείμενο μπορεί να είναι μορφής απλού κειμένου (plaintext) όπως οι γνωστοί τύποι με επέκταση TXT, μπορεί επίσης να είναι αρχεία που δημιουργήθηκαν με κάποιο πρόγραμμα επεξεργασίας κειμένου όπως για παράδειγμα του Microsoft Word με χαρακτηριστική επέκταση DOC και DOCX. Άλλη μία γνωστή μορφή αρχείων κειμένου είναι εκείνα της μορφής PDF (Portable Document Format), όπως επίσης κείμενο μπορεί να περιέχεται σε αρχεία τύπου εικόνας. Για την ακρίβεια, είναι αρκετά συχνό αρχεία PDF να περιέχουν το κείμενο σε μορφή εικόνας, όπου η κάθε σελίδα αντιμετωπίζεται σαν εικόνα και όχι σαν κείμενο που να μπορεί ο χρήστης να επιλέξει, να αντιγράψει ή να αναζητήσει. Σε περίπτωση που το αρχείο που περιέχει το κείμενο είναι σε κάποια γνωστή μορφή αρχείων κειμένου, τότε το υπολογιστικό σύστημα "γνωρίζει" πως να το χειριστεί άμεσα και η εξαγωγή του κειμένου είναι εύκολη και ταχύτατη. Διαφορετικά, το κείμενο χρειάζεται να εξαχθεί από την εικόνα. Αυτό επιτυγχάνεται με διάφορες τεχνικές, μία από τις οποίες είναι η οπτική αναγνώριση χαρακτήρων (Optical Character Recognition, OCR). Παρόμοιες τεχνικές περιγράφονται επιγραμματικά στο άρθρο *The State of the Art in Text Recognition Techniques* (Salunke, Badhe, Doke, & Raykar, 2015). Η εφαρμογή κάποιας παρόμοιας τεχνικής επιβαρύνει το σύστημα με περισσότερο όγκο εκτέλεσης. Αφού ολοκληρωθεί η εξαγωγή κειμένου, είναι πλέον δυνατή η διαχείρισή του ανάλογα με τις ανάγκες του προγράμματος.

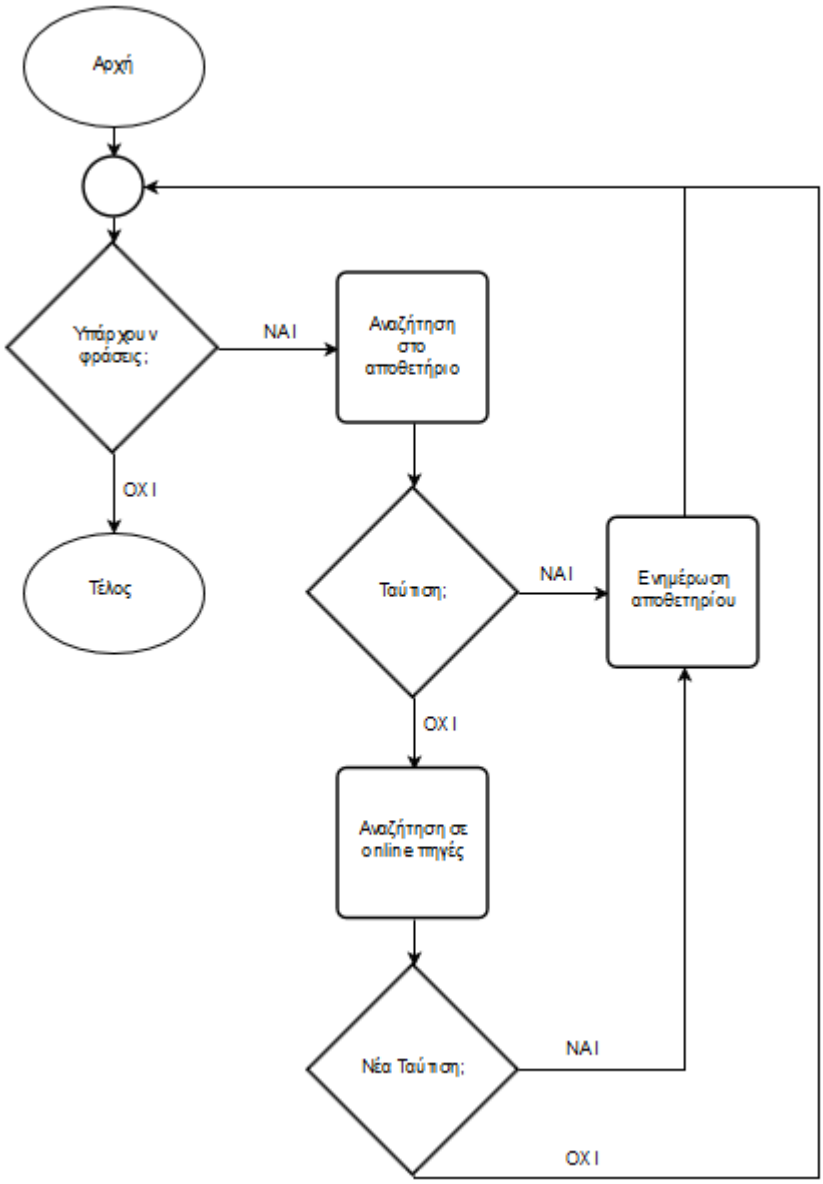
Αφού το κείμενο έχει αναγνωρισθεί από το υπολογιστικό σύστημα μπορεί να διαχωριστεί σε λέξεις και φράσεις. Ο διαχωρισμός ενός κειμένου σε φράσεις μπορεί να είναι μία αρκετά απλή διαδικασία, όμως αυτό δε σημαίνει πως δεν είναι ένας αξιόλογος μηχανισμός που μπορούμε να χρησιμοποιήσουμε σαν αρχικό εργαλείο στον εντοπισμό λογοκλοπών. Σε αυτό το σημείο ο εντοπισμός λέξεων ανάγεται σε εντοπισμό φράσεων. Υποθέτουμε πως έχουμε ένα σύστημα που θα αναλύσει δύο κείμενα: ένα κείμενο το οποίο θα πρέπει να το αξιολογήσει ως αυθεντικό ή όχι (A) και ένα αυθεντικό κείμενο από μία γνωστή πηγή (B). Για να γίνει εφαρμογή μίας τεχνικής εντοπισμού φράσεων στο κείμενο, αρχικά το κείμενο προς αξιολόγηση (A) αναλύεται, εφαρμόζοντας κάποια τεχνική αναγνώρισης κειμένου αν αυτό χρειαστεί, ώστε το σύστημα να σχηματίσει ένα σύνολο φράσεων. Στη συνέχεια, το σύστημα θα αναζητήσει τις φράσεις που σχημάτισε στο αυθεντικό κείμενο (B). Είναι λογικό πως αν το σύστημα εντοπίσει αρκετές αυτούσιες φράσεις, τα δύο κείμενα θα παρουσιάζουν ένα σημαντικό ποσοστό ομοιότητας.

Στο παραπάνω παράδειγμα, η αναζήτηση κειμένου και πιο συγκεκριμένα η αναζήτησης αυτούσιας φράσης σε ένα κείμενο είναι μία εφαρμογή τεχνικής ταύτισης μοτίβου (pattern-matching). Στην επιστήμη των υπολογιστών, ο όρος ταύτιση μοτίβου αναφέρεται στον έλεγχο ότι μια ακολουθία από στοιχεία περιέχει συστατικά ενός μοτίβου (Wikipedia, 2017). Αυτό γίνεται αντιληπτό στο προηγούμενο παράδειγμα αν θεωρήσουμε πως το κείμενο γνωστής πηγής (Β) είναι η ακολουθία από στοιχεία, δηλαδή από φράσεις και κατά επέκταση χαρακτήρες, ενώ το μοτίβο που μας απασχολεί είναι το κείμενο προς έλεγχο (Α) και τα στοιχεία που το αποτελούν είναι οι φράσεις του που αναζητούμε στο αυθεντικό (Β). Με αυτό τον τρόπο έχουμε εφαρμόσει μία πρώιμης μορφής τεχνική ταύτισης μοτίβου. Χρησιμοποιώντας τους ορισμούς και τις έννοιες που αναπτύχθηκαν, παραθέτουμε την ακόλουθη υπόθεση. Έστω ένα κείμενο που θα περάσει από έλεγχο για λογοκλοπή. Το κείμενο αυτό αποτελείται από ένα σύνολο λέξεων που σχηματίζουν φράσεις. Αναλύοντας το κείμενο σε φράσεις, έχουμε πλέον ένα σύνολο φράσεων. Το σύστημα που διεξάγει τον έλεγχο, δέχεται σαν είσοδο μία φράση την οποία αναζητά στο διαδίκτυο για να εντοπίσει ομοιότητες με διάφορες πηγές. Αν υποθέσουμε πως το σύστημα αναγνωρίζει την κάθε φράση σαν μία κατάσταση που πρέπει να εξυπηρετήσει, μπορούμε να το φέρουμε κοντά στη λογική ερμηνείας του γενικότερου πλαισίου (context). Μία φράση αποτελεί μία κατάσταση που το σύστημα πρέπει να εξυπηρετήσει, όπως ακριβώς ο αλγόριθμος που εφαρμόστηκε στην αρχιτεκτονική DSNPM στην Εικόνα 2.1.1. Εξυπηρέτηση της κατάστασης θα είναι η εύρεση της προς αναζήτηση φράσης στο διαδίκτυο. Για τη βελτίωση της απόδοσης του συστήματος και την ολοκλήρωση της παρομοίωσης των δύο περιπτώσεων, εισάγουμε στο σύστημα που διεξάγει την ανάλυση ένα εσωτερικό αποθετήριο στο οποίο θα αποθηκεύει τα ευρήματα προηγούμενων αναζητήσεων. Με αυτή την προσθήκη, μία αναζήτηση φράσης θα περνάει πρώτα από το αποθετήριο αναζητήσεων. Αν βρεθεί στο αποθετήριο όμοια φράση με αυτή που αναζητείται, τότε επιτυγχάνονται δύο σημαντικά πράγματα: βελτιώνεται η απόδοση της αναζήτησης λόγω του μειωμένου όγκου εγγραφών που κρατάει το αποθετήριο το οποίο επιστρέφει αποτελέσματα άμεσα και βελτιώνεται η ποιότητα των αποτελεσμάτων. Αυτό γίνεται επειδή όταν έχουμε ταύτιση φράσης με κείμενο του αποθετηρίου, σημαίνει πως παραπάνω από μία φράσεις του κειμένου που ελέγχεται έχουν ταυτιστεί με το κείμενο της συγκεκριμένης πηγής, οπότε αυξάνεται η πιθανότητα να αποτελεί κρούσμα λογοκλοπής.

Συνοψίζοντας, έχουμε περιγράψει ένα σύστημα το οποίο αντλεί πληροφορία σχετικά με την 'πλαισιακή' γνώση. Κάτι τέτοιο αποτελεί μία εφαρμογή τεχνικής ταύτισης πλαισίου (context-matching) επειδή βασίζεται στην αναζήτηση μοτίβων (ταύτιση φράσεων σε κείμενο) και τα αποτελέσματα που παράγονται προκύπτουν με βάση το context που αναλύεται κάθε φορά, αν είναι κάτι ήδη γνωστό και έχει αναγνωριστεί, ή κάτι πρωτόγνωρο το οποίο όμως το σύστημα θα είναι σε θέση να αναγνωρίσει σε μελλοντική αναφορά.

2.3. ΠΕΡΙΓΡΑΦΗ ΣΥΣΤΗΜΑΤΟΣ

Το σύστημα που αναπτύχθηκε στα πλαίσια της εργασίας, είναι αρκετά όμοιο με το σύστημα που περιγράφεται στο παράδειγμα της προηγούμενης παραγράφου. Στην Εικόνα 2.3.1, παρουσιάζεται η βασική αρχιτεκτονική του συστήματος.



Εικόνα 2.3.1: Διάγραμμα ροής

Το πρόγραμμα έχει σχεδιαστεί ώστε να δέχεται ένα κείμενο και να κάνει αναζήτηση των φράσεων σε πηγές από το διαδίκτυο έτσι ώστε να εμφανίσει εκείνες με τα μεγαλύτερα ποσοστά ταύτισης. Βασική παράμετρος είναι το εσωτερικό αποθετήριο που διατηρεί με

αποτελέσματα προηγούμενων αναζητήσεων και λειτουργεί ως μία ταχύτατη μνήμη (cache) στην κάθε αναζήτηση.

2.3.1. ΕΠΕΞΕΡΓΑΣΙΑ ΕΙΣΟΔΟΥ

Το κείμενο που ορίζεται σαν είσοδος μπορεί να είναι διάφορων μορφών. Υποστηριζόμενες μορφές είναι αρχεία τύπου απλού κειμένου (plaintext, TXT), κείμενα που αναπτύχθηκαν σε κάποιο πρόγραμμα επεξεργασίας κειμένου (DOC, DOCX) καθώς και κείμενα μορφής PDF (Portable Document Format). Στα αρχεία απλού κειμένου και εκείνα που αναπτύχθηκαν σε προγράμματα επεξεργασίας κειμένου, η εξαγωγή του κειμένου γίνεται μέσω της Java σχετικά απλά. Στα αρχεία της μορφής PDF, η εξαγωγή κειμένου είναι αρκετά πιο πολύπλοκη. Επειδή δεν είναι ξεκάθαρο κάθε αρχείο PDF τι κωδικοποίηση έχει καθώς και αν διατηρεί ενσωματωμένες λειτουργίες όπως για παράδειγμα κάποιες γραμματοσειρές (custom fonts) ο χειρισμός τους με ένα γενικό τρόπο δεν είναι εφικτός. Επίσης, πολλά αρχεία της μορφής PDF διατηρούν αποθηκευμένο το κείμενο σε μορφή εικόνας (embedded image). Σε αυτή την περίπτωση, δεν είναι δυνατή καμία τεχνική εξαγωγής κειμένου και τέτοιες περιπτώσεις εξυπηρετούνται με μία τεχνική OCR.

Αφού το κείμενο έχει εξαχθεί από το αρχείο της εισόδου, γίνεται μία αρχική επεξεργασία με σκοπό να διαχωριστούν οι λέξεις του κειμένου. Με τη χρήση αρκετών κανονικών εκφράσεων (regular expressions) το κείμενο χωρίζεται σε λέξεις οι οποίες τελικά σχηματίζουν φράσεις. Οι λίστα με τις φράσεις είναι αυτή που θα χρησιμοποιηθεί από το πρόγραμμα. Όσο υπάρχουν φράσεις, αυτές θα αναζητάει το πρόγραμμα αρχικά στο εσωτερικό αποθετήριο (repository) και σε περίπτωση που δεν βρει σε αυτό κάποια ταύτιση, θα συνεχίζει με την αναζήτησή τους στο διαδίκτυο.

2.3.2. ΑΝΑΖΗΤΗΣΗ ΣΤΟ ΕΣΩΤΕΡΙΚΟ ΑΠΟΘΕΤΗΡΙΟ

Το πρώτο βήμα στην αναζήτηση είναι η αναζήτηση στο εσωτερικό αποθετήριο. Το αποθετήριο ουσιαστικά αποτελεί τη βάση δεδομένων του προγράμματος. Είναι σχεδιασμένο να κρατάει κάθε κείμενο που έχει παρουσιάσει κάποια ταύτιση στο παρελθών, καθώς και τον αριθμό των ταυτίσεων που έχει σημειώσει το κάθε κείμενο. Όταν μία φράση αναζητηθεί στο repository και σημειωθεί ταύτιση, τότε το κείμενο που παρουσίασε την ταύτιση ενημερώνεται αυξάνοντας τον αριθμό ταυτίσεων που έχει παρουσιάσει. Η αναζήτηση γίνεται με τα περιεχόμενα του αποθετηρίου να είναι ταξινομημένα με βάση των αριθμό των εντοπισμένων φράσεων σε φθίνουσα σειρά. Με αυτό τον τρόπο ένα κείμενο που έχει εντοπιστεί να περιέχει περισσότερες φράσεις από κάποιο άλλο της βάσης δεδομένων, είναι πιο πιθανό να σημειώσει νέα ταύτιση, το οποίο το μετατρέπει σε καλύτερο υποψήφιο υποκείμενο λογοκλοπής.

Στην περίπτωση που η αναζήτηση στο εσωτερικό αποθετήριο δεν επιστρέψει κάποιο αποτέλεσμα, ακολουθεί η αναζήτηση στο διαδίκτυο.

2.3.3. ΑΝΑΖΗΤΗΣΗ ΣΤΟ ΔΙΑΔΙΚΤΥΟ

Σε αυτό το σημείο, το σύστημα αναζητάει κάποια φράση η οποία δεν υπάρχει καθόλου στη βάση δεδομένων που διατηρεί. Πρόκειται για ένα νέο context. Σε αυτό το βήμα, γίνεται αναζήτηση της φράσης στο διαδίκτυο. Η φράση αναζητείται αυτούσια ώστε να μην εντοπιστούν ανεπιθύμητα αποτελέσματα, όπως κείμενα που περιέχουν τις λέξεις της φράσης σε διαφορετικά σημεία αλλά πέρα από τη μορφή που θέλουμε. Επειδή κάποια φράση μπορεί να είναι πολύ γενική, δεδομένου ότι έχει μήκος τριών (3) έως πέντε (5) λέξεων, και να υπάρχει σε μία πληθώρα κειμένων, η αναζήτηση περιέχει και κάποιες λέξεις-κλειδιά που προσδιορίζονται στην είσοδο και παραμετροποιούν την αναζήτηση. Με αυτό τον τρόπο φιλτράρονται τα αποτελέσματα ώστε να κόβονται εκείναι που δεν έχουν τελικά άμεση σχέση με το περιεχόμενο που μας ενδιαφέρει. Η αναζήτηση γίνεται σε κάποια μηχανή αναζήτησης μεταξύ των Google, Bing και Yahoo και τα ευρήματα που χειρίζεται το πρόγραμμα είναι αρχεία των μορφών που ήδη αναφέρθηκε (DOC, DOCX, PDF) με την προσθήκη ιστοσελίδων. Αναζήτηση δεν γίνεται συγκεκριμένα σε ηλεκτρονικά/ψηφιακά αποθετήρια επιστημονικών άρθρων και ακαδημαϊκών εργασιών διότι τα αποτελέσματα που προκύπτουν από αντίστοιχη αναζήτηση είναι ελλειπή, κάτι που αναλύεται περισσότερο στην επόμενη θεματική ενότητα.

Αφού εντοπιστεί ταύτιση της φράσης σε κείμενο κάποιας πηγής μέσω του διαδικτύου, ακολουθεί η ενημέρωση του αποθετηρίου. Το κείμενο που βρέθηκε διαβάζεται από το πρόγραμμα με τεχνικές που περιγράφηκαν στο πρώτο βήμα και εισάγεται στο αποθετήριο. Με αυτό τον τρόπο θα συμπεριληφθεί στην πρώτη αναζήτηση που θα γίνει πριν τη διαδικτυακή και θα είναι αυτόματα πιο εύκολο να αναγνωριστούν και επόμενες φράσεις. Οπότε αν κάποιες φράσεις αποτελούν προϊόν λογοκλοπής, με αυτό το χειρισμό το πρόγραμμα θα έχει μία αρκετά καλύτερη πιθανότητα να ταυτίσει αυτές τις φράσεις στο σωστό κείμενο και τα αποτελέσματα θα είναι αρκετά πιο αξιόπιστα από μία απλή αναζήτηση στο internet. Στην περίπτωση που μία φράση δεν ταυτιστεί ούτε στο αποθετήριο αλλά ούτε και στο internet, η φράση αυτή αγνοείται και το πρόγραμμα συνεχίζει με την επόμενη φράση.

Όταν τελειώσουν οι φράσεις προς αναζήτηση, η εκτέλεση του προγράμματος τερματίζει. Το τελικό αποτέλεσμα είναι η διαμόρφωση του εσωτερικού αποθετηρίου ώστε να περιέχει όλη

την πληροφορία που χρειάζεται ο χρήστης για να ερμηνεύσει τα αποτελέσματα. Πιο συγκεκριμένα, πόσες ταυτίσεις έχουν εντοπιστεί σε κάθε κείμενο της βάσης δεδομένων και ποιο από αυτά έχει παρουσιάσει το μεγαλύτερο αριθμό. Με μία αρκετά περιορισμένη παραμετροποίηση, ο χρήστης μπορεί να επιλέξει το πλήθος των αποτελεσμάτων πουτελικά θα εμφανιστούν.

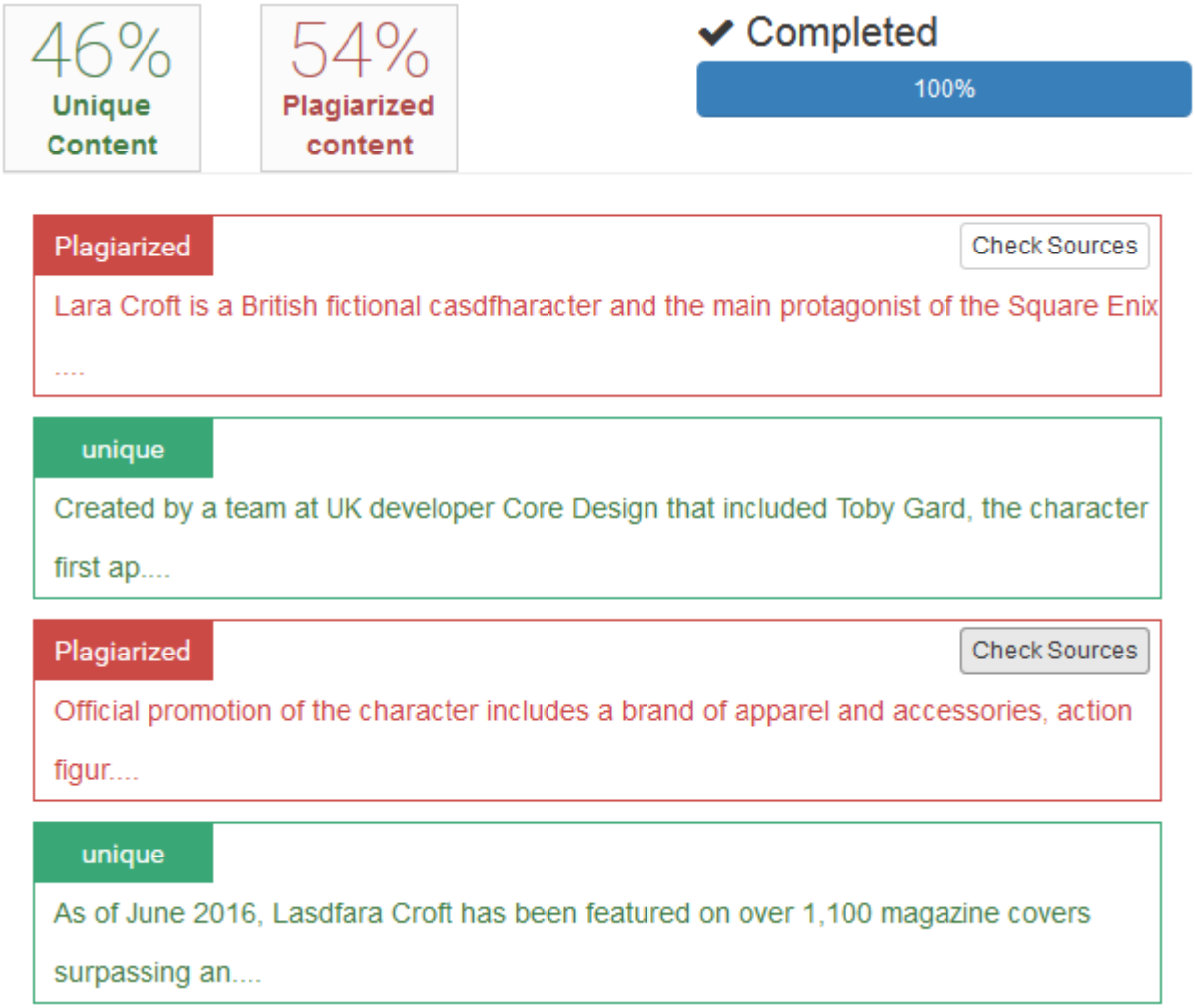
2.4. ΔΩΡΕΑΝ ΛΟΓΙΣΜΙΚΟ/ΛΟΓΙΣΜΙΚΟ ΑΝΟΙΧΤΟΥ ΚΩΔΙΚΑ ΣΤΟΝ ΕΝΤΟΠΙΣΜΟ ΛΟΓΟΚΛΟΠΩΝ

Υπάρχουν αρκετά εργαλεία για τον εντοπισμό λογοκλοπών. Αρκετά είναι επί πληρωμή και άλλα παρέχουν υπηρεσίες δωρεάν. Κάποια από τα πιο γνωστά εργαλεία εντοπισμού λογοκλοπών είναι τα εξής:

Grammarly: Πρόκειται για ένα εργαλείο το οποίο αναπτύχθηκε για γραμματικό έλεγχο κειμένου. Το Grammarly παρέχει τις υπηρεσίες του μέσω του ιστότοπου που διατηρεί αλλά μπορεί επίσης να εγκατασταθεί σαν add-on στο πρόγραμμα πλοήγησης στο διαδίκτυο firefox. Μπορεί επίσης να εγκατασταθεί ως add-on στη σουίτα προϊόντων Microsoft Office αλλά και ως εφαρμογή σταθερού υπολογιστή των Windows. Μεταξύ των λειτουργιών που παρέχει είναι και ο έλεγχος για λογοκλοπές. Είναι ένα αρκετά διαδεδομένο εργαλείο που χρησιμοποιείται για διάφορους ελέγχους. Για να το χρησιμοποιήσει κάποιος στην κύρια λειτουργία του, δηλαδή το γραμματικό έλεγχο, χρειάζεται εγγραφή. Η υπηρεσία εντοπισμού λογοκλοπών απαιτεί συνδρομή.

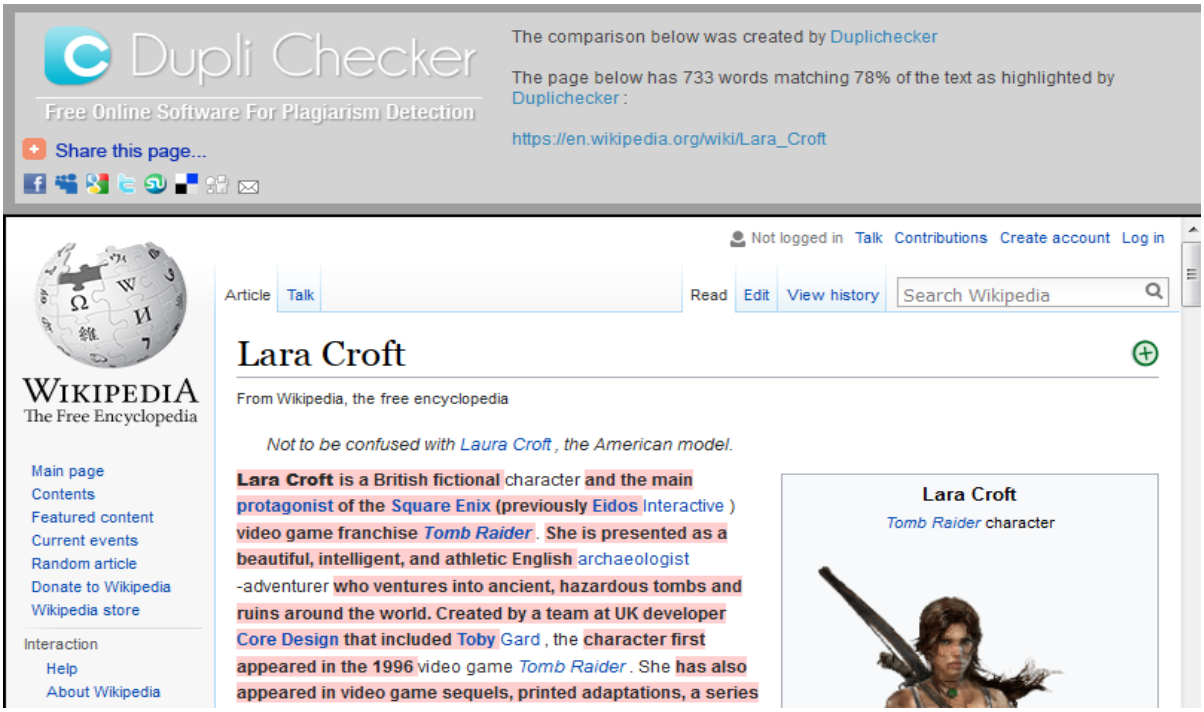
ArticleChecker: Η υπηρεσία αναζήτησης λογοκλοπών παρέχεται δωρεάν μέσω του ArticleChecker. Με όριο τις χίλιες (1000) λέξεις ανά αναζήτηση, το εργαλείο αυτό αποτελεί μία αξιόλογη δωρεάν επιλογή με γρήγορη εξαγωγή αποτελεσμάτων. Το ArticleChecker εξυπηρετεί ένα μικρό αριθμό αναλύσεων, χωρίς να ζητήσει από το χρήστη να εγγραφεί. Με την εγγραφή του χρήστη, η οποία είναι γίνεται δωρεάν, μπορεί να υποστηρίξει μέχρι 500 αναζητήσεις. Είναι μία υπηρεσία που δέχεται για είσοδο ένα κείμενο που ο χρήστης πληκτρολογεί σε μία περιοχή

κειμένου και παρέχεται μέσω WEB. Μελετώντας τον τρόπο λειτουργίας του ArticleChecker, θα παρατηρήσει κάποιος πως από τον τρόπο που παρουσιάζονται τα αποτελέσματα, το εργαλείο φαίνεται "σπάει" το κείμενο σε φράσεις ανάλογα με το μήκος του κειμένου. Σε ένα κείμενο με το μέγιστο αριθμό λέξεων (1000), το ArticleChecker χώρισε το κείμενο σε φράσεις των δύο (2) προτάσεων τις οποίες στη συνέχεια έλεγξε για ομοιότητες στο διαδίκτυο. Σε μία δοκιμή για την ερμηνεία των αποτελεσμάτων αλλά και του τρόπου λειτουργίας, πραγματοποιήθηκε το εξής πείραμα. Δόθηκε για είσοδο ένα κείμενο παρμένο από το site Wikipedia. Το κείμενο τροποποιήθηκε ώστε να περιέχει μερικές παραπάνω λέξεις σε διάφορα σημεία κάποιων προτάσεων με τυχαίο τρόπο. Ουσιαστικά το κείμενο της Wikipedia παρέμεινε ίδιο με διαφορά τις μερικές λέξεις που προστέθηκαν σε τυχαία σημεία. Το αποτέλεσμα ήταν να εντοπισθεί 54% του περιεχομένου ως κρούσμα λογοκλοπής. Το ποσοστό αυτό αν και απέχει αρκετά από το πραγματικό ποσοστό ομοιότητας των κειμένων, παραμένει ένα υψηλό ποσοστό ομοιότητας.



Εικόνα 2.4.1: Αποτελέσματα εκτέλεσης του ArticleChecker

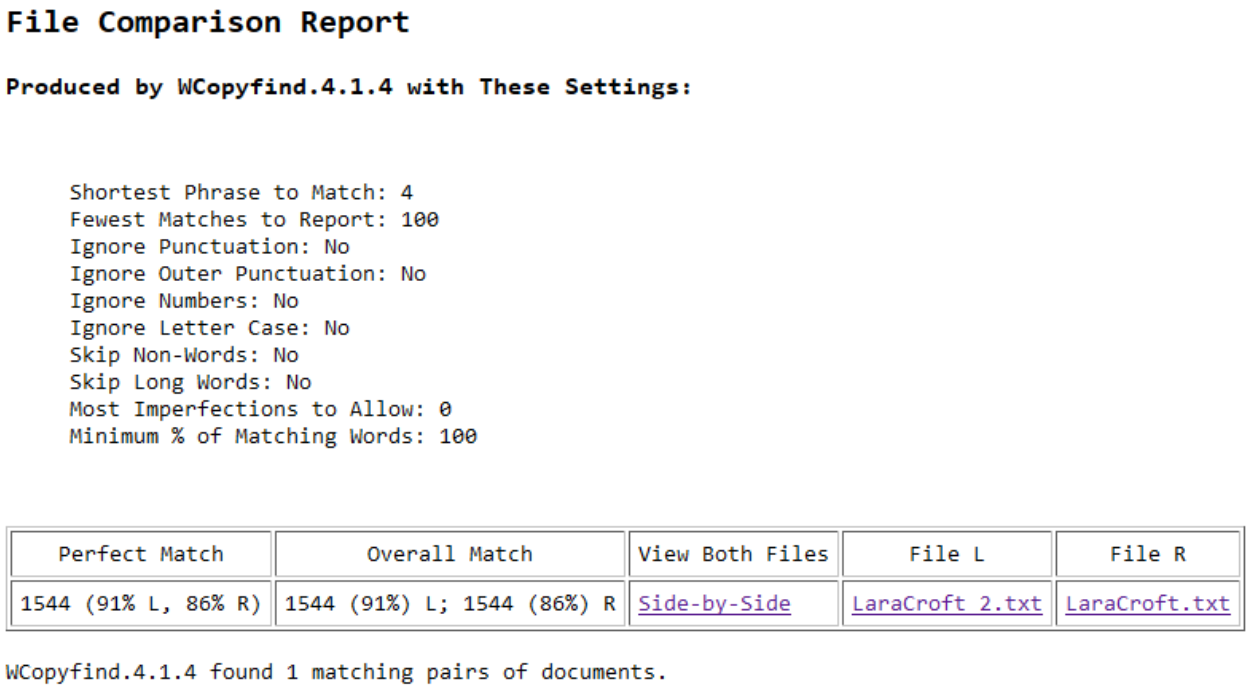
DupliChecker: Το DupliChecker είναι ένα website το οποίο λειτουργεί σαν εργαλείο εντοπισμού λογοκλοπών. Δέχεται είσοδο είτε κείμενο που ο χρήστης πληκτρολογεί εκείνη τη στιγμή, είτε αρχεία των μορφών TXT και DOCX, είτε ακόμα και URL. Μετά την αναζήτηση/ανάλυση παρουσιάζονται άρθρα τα οποία βρέθηκαν να έχουν ομοιότητα. Κάθε αναζήτηση έχει το ανώτατο όριο των χιλίων (1000) λέξεων. Παρέχεται δωρεάν εγγραφή για τη χρήση των υπηρεσιών. Συγκριτικά με το προηγούμενο παράδειγμα, το DupliChecker εντόπισε 78% ομοιότητα μεταξύ του ελαφρώς τροποποιημένου κειμένου και της αυθεντικής πηγής, ποσοστό αρκετά πιο ρεαλιστικό από εκείνο που προέκυψε από την ανάλυση του ArticleChecker. Τα αποτελέσματα εμφανίζονται σε μορφή λίστας, όπου επιλέγοντας τη σελίδα που περιέχει ομοιότητα, ανοίγει μία καινούργια καρτέλα στον περιηγητή που είναι σε χρήση και εμφανίζονται μαρκαρισμένα σημεία της σελίδας που είναι όμοια με το κείμενο που ελέγχθηκε, όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 2.4.2: Παρουσίαση σύγκρισης του DupliChecker

Viper: Το λογισμικό αυτό εγκαθίσταται offline, χρειάζεται δηλαδή εγκατάσταση στον τοπικό υπολογιστή του κάθε χρήστη. Διανέμεται απολύτως δωρεάν, αλλά για να το χρησιμοποιήσει κάποιος πρέπει να δημιουργήσει ένα λογαριασμό, με ένα όνομα χρήστη και ένα κωδικό πρόσβασης τα οποία θα πρέπει να συμπληρώνει κάθε φορά που κάνει χρήση του Viper. Πρόκειται για ένα αρκετά παραμετροποιήσιμο εργαλείο στον εντοπισμό λογοκλοπών.

WcopyFind: Στον τομέα του ανοιχτού λογισμικού, δεν θα βρει κάποιος ποικιλία σε εργαλεία εντοπισμού λογοκλοπών. Ένα από τα διαθέσιμα εργαλεία είναι το WCopyFind. Είναι ένα πρόγραμμα που η λειτουργία του περιλαμβάνει τη σύγκριση δύο (2) ή περισσότερων αρχείων κειμένου και εντοπισμό ομοιοτήτων. Αρχεία συμβατά προς έλεγχο είναι της μορφής DOC, DOCX, TXT, PDF, HTML. Το WCopyFind δεν υποστηρίζει την αναζήτηση ομοιοτήτων σε αρχεία στο διαδίκτυο, αλλά μόνο σύγκριση με αρχεία τοπικά. Αφού τα απαραίτητα αρχεία επεξεργαστούν και τελειώσουν οι συγκρίσεις, τα αποτελέσματα παρουσιάζονται σε μορφή HTML. Το πρόγραμμα έχει υλοποιηθεί στη γλώσσα C. Στο παράδειγμα με το ArticleChecker, το ποσοστό ομοιότητας που αναγνώρισε το WCopyFind σε σύγκριση του αυθεντικού κειμένου με την ελαφρώς τροποποιημένη έκδοσή του, είναι 91% ταύτιση.



File Comparison Report

Produced by WCopyfind.4.1.4 with These Settings:

Shortest Phrase to Match: 4
Fewest Matches to Report: 100
Ignore Punctuation: No
Ignore Outer Punctuation: No
Ignore Numbers: No
Ignore Letter Case: No
Skip Non-Words: No
Skip Long Words: No
Most Imperfections to Allow: 0
Minimum % of Matching Words: 100

Perfect Match	Overall Match	View Both Files	File L	File R
1544 (91% L, 86% R)	1544 (91%) L; 1544 (86%) R	Side-by-Side	LaraCroft 2.txt	LaraCroft.txt

WCopyfind.4.1.4 found 1 matching pairs of documents.

Εικόνα 2.4.3: Αποτέλεσμα σύγκρισης του WCopyFind

Πρόκειται για αρκετά υψηλό και ακόμα πιο ρεαλιστικό ποσοστό, κάτι που σημαίνει πως το WCopyFind λειτουργεί περισσότερο σε επίπεδο λέξεων, παρά σε επίπεδο προτάσεων. Αυτό επαληθεύεται και από τη μελέτη του πηγαίου κώδικα του προγράμματος, ο οποίος είναι διαθέσιμος.

3. ΠΡΑΚΤΙΚΟ ΜΕΡΟΣ

Σε αυτή τη θεματική ενότητα περιγράφεται λεπτομερώς όλο το τεχνικό μέρος της υλοποίησης του προγράμματος *WebCrawler*. Αναλύεται ξανά ο τρόπος λειτουργίας και παρουσιάζονται τεχνικές πληροφορίες σχετικά με την υλοποίηση διαφόρων σημείων, αναφέρονται έτοιμα εργαλεία που χρησιμοποιήθηκαν τα οποία είναι διαθέσιμα δωρεάν και όπως επίσης και το σύνολο των επιμέρους βιβλιοθηκών που ενσωματώθηκαν στη λειτουργία του προγράμματος.

3.1. ΒΑΣΙΚΑ ΤΜΗΜΑΤΑ

Το πρόγραμμα έχει αναπτυχθεί σε Java και η εκτελέσιμη μορφή του είναι ως Java™ Archive (*WebCrawler.jar*). Ο λόγος που επιλέχθηκε η Java είναι το ότι πρόκειται για μία αντικειμενοστρεφείς γλώσσα που τη χαρακτηρίζει η ευελιξία στη χρήση βιβλιοθηκών για διάφορες επιπρόσθετες λειτουργίες. Σημαντικός παράγοντας είναι η ύπαρξη αναλυτικών εγχειριδίων (documentation) και εύκολη πρόσβαση σε παραδείγματα εφαρμογής διαφόρων τεχνικών. Χρησιμοποιήθηκε το JDK 1.8.0_111 (32bit). Το πρόγραμμα κάνει χρήση ενός εσωτερικού αποθετηρίου, μίας βάσης δεδομένων. Επιλέχθηκε η χρήση βάσης δεδομένων για τη μεγιστοποίηση της απόδοσης πολλαπλών αναζητήσεων σε πλήθος κειμένων και η ελαχιστοποίηση όγκου που δεσμεύουν κείμενα που εισάγονται στο αποθετήριο. Για βάση δεδομένων επιλέχθηκε η MySQL (MySQL Community Server 5.7.13), για το λόγο ότι είναι μία αρκετά συνηθισμένη βάση δεδομένων και δεν είναι ιδιαίτερα απαιτητική σε υλικό/λογισμικό. Επίσης, αν και ακολουθεί το σχεσιακό μοντέλο βάσης δεδομένων, απέχει αρκετά από την αυστηρή οργάνωση και παραμετροποίηση που απαιτούν άλλες σχεσιακές βάσεις δεδομένων όπως της Oracle. Πρόκειται για μία αρκετά απλή βάση ανοιχτού λογισμικού όταν χρησιμοποιείται στην ανάπτυξη ανοιχτού λογισμικού και από μη-κερδοσκοπικές οργανώσεις. Οι απαιτήσεις του προγράμματος δεν είναι τέτοιες ώστε να μην καλύπτονται από τη MySQL.

Συνολικά, η εφαρμογή μπορεί να χωριστεί στα παρακάτω τμήματα:

Application Layer: Πρόκειται για το σκελετό του προγράμματος. Περιέχει το σύνολο των κλάσεων που καλύπτουν τις βασικές λειτουργίες της εφαρμογής. Εδώ ανήκουν οι κλάσεις που

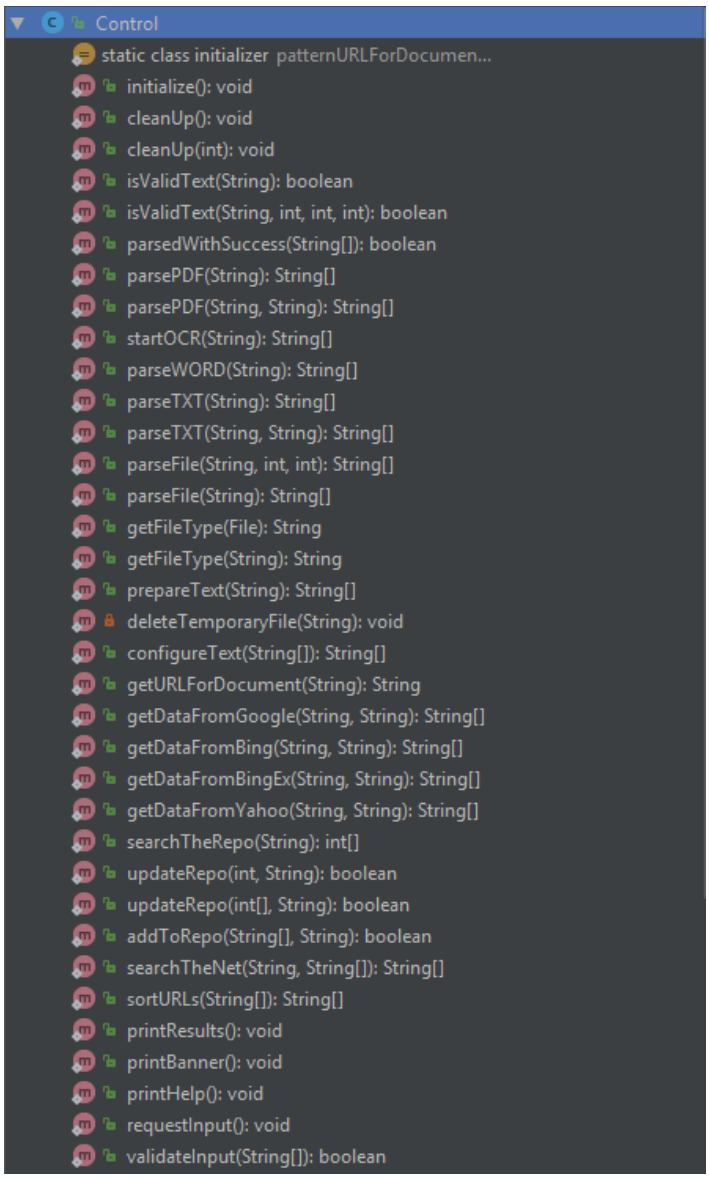
χειρίζονται την εξαγωγή κειμένου από διαφόρων ειδών αρχεία, την αναζήτηση στο αποθετήριο όπως και στο διαδίκτυο για ομοιότητες, αλλά και κλάσεις που έχουν βοηθητικές λειτουργίες όπως η επεξεργασία κειμένου και η διατήρηση εγγραφών ιστορικού (tracing) που είναι αρκετά χρήσιμα στη διαδικασία εξέτασης της ροής εκτέλεσης του προγράμματος. Οι βασικές κλάσεις είναι υλοποιημένες στα αρχεία Base.java, Control.java, Snitch.java και ResultRecord.java.

Η κλάση Base περιέχει τη ροή εκτέλεσης του προγράμματος, μαζί με μερικά παραδείγματα των επιμέρους λειτουργιών.

Στην κλάση Snitch υλοποιείται το αντικείμενο που χειρίζεται το tracing. Είναι μεγάλης σημασίας η υλοποίηση να γίνεται με τρόπο ώστε να επιλέγεται καθολικά η κωδικοποίηση UTF-8.

Η κλάση ResultRecord αποτελεί μία δομή αποθήκευσης των τελικών αποτελεσμάτων. Ένα αντικείμενο ResultRecord έχει πληροφορία σχετικά με τις ταυτίσεις που έχουν σημειωθεί σε ένα κείμενο.

Η κλάση Control υλοποιεί το σύνολο των συναρτήσεων που εκτελούνται από το πρόγραμμα. Σε αυτό το αρχείο βρίσκονται και χειρίζονται οι βασικές σταθερές του προγράμματος. Στην Control υλοποιούνται οι συναρτήσεις σχετικά με την αρχικοποίηση του προγράμματος, εξαγωγή κειμένου, ανάλυση κειμένου, αναζήτηση μέσω των γνωστών μηχανών αναζήτησης και εμφάνιση αποτελεσμάτων. Οι μέθοδοι της κλάσης Control είναι οι παρακάτω:



Εικόνα 3.1.1: Υπογραφές μεθόδων κλάσης Control

Database Layer: Περιγράφεται το εσωτερικό αποθετήριο. Στο επίπεδο της MySQL έχει δημιουργηθεί μία βάση δεδομένων με όνομα repo (repository). Η repo αποτελείται από τους εξής πίνακες:

- DOCUMENTS: Αποθηκεύει κάθε νέο κείμενο που εντοπίζεται από το διαδίκτυο. Κάθε κείμενο χαρακτηρίζεται από ένα μοναδικό αύξοντα αριθμό (doc_id), μία διεύθυνση μορφής URL (doc_url), καθώς και ολόκληρο το κείμενο που “διαβάστηκε” από το πρόγραμμα (doc_text).
- HITS: Ο πίνακας αυτός αποθηκεύει πληροφορίες που σχετίζονται με ταύτιση φράσης σε κάποια σελίδα/κείμενο. Χρησιμοποιεί ένα αύξοντα αριθμό (hit_id) για να αναγνωρίζει

το κάθε εύρημα, το χαρακτηριστικό αριθμό του κείμενου που εντοπίστηκε η ομοιότητα (hit_doc_id, αναφορά στη στήλη doc_id του πίνακα DOCUMENTS) και αποθηκεύει επίσης τη φράση η οποία βρέθηκε να είναι όμοια στο συγκεκριμένο κείμενο (hit_text).

Στην Εικόνα 3.1.2 παρουσιάζονται οι πίνακες που αναφέρθηκαν παραπάνω.

```
mysql> describe DOCUMENTS;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| doc_id   | int(11)      | NO   | PRI | NULL    | auto_increment |
| doc_url  | varchar(500) | NO   |     | NULL    |                |
| doc_text | text         | NO   |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
3 rows in set (0.04 sec)


mysql> describe HITS;
+------------+--------------+------+-----+---------+----------------+
| Field      | Type         | Null | Key | Default | Extra          |
+------------+--------------+------+-----+---------+----------------+
| hit_id     | int(11)      | NO   | PRI | NULL    | auto_increment |
| hit_doc_id | int(11)      | NO   | MUL | NULL    |                |
| hit_text   | varchar(200) | NO   |     | NULL    |                |
+------------+--------------+------+-----+---------+----------------+
3 rows in set (0.01 sec)
```

Εικόνα 3.1.2: Πίνακες της βάσης repo

Για να έχει τη δυνατότητα να υποστηρίξει και να εμφανίσει κατάλληλα τους ελληνικούς χαρακτήρες, η βάση repo έχει δημιουργηθεί με την εντολή που φαίνεται στην Εικόνα 3.1.3.

```
mysql> CREATE DATABASE repo DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
```

Εικόνα 3.1.3: Δημιουργία της βάσης repo

Με αυτό τον τρόπο δηλώνεται ρητά πως η repo θα χειρίζεται τις εγγραφές της χρησιμοποιώντας κωδικοποίηση UTF-8. Η βάση δεδομένων επικοινωνεί με τη Java μέσω της κλάσης Repo.java, η οποία χειρίζεται τη δημιουργία της βάσης και των πινάκων, την καταχώρηση εγγραφών στους πίνακες καθώς και την εκτέλεση αναζητήσεων κατά την εκτέλεση του προγράμματος.

Crawl.properties: Πρόκειται για ένα αρχείο στο οποίο ο χρήστης συμπληρώνει κάποιες βασικές παραμέτρους που χρειάζονται κατά την εκτέλεση του προγράμματος, όπως το όνομα χρήστη ή το κλειδί υπηρεσίας Bing Web Search, όπως θα αναλυθεί και παρακάτω.

Στα επόμενα μέρη αναλύονται τεχνικά τα βήματα εκτέλεσης του προγράμματος.

3.1.1. ΕΙΣΟΔΟΣ ΚΑΙ ΕΞΑΓΩΓΗ ΚΕΙΜΕΝΟΥ

Το πρόγραμμα ξεκινάει με κάποιους αρχικούς ελέγχους. Βασική προϋπόθεση της ομαλής εκτέλεσης του προγράμματος είναι η ύπαρξη του αρχείου παραμέτρων *crawl.properties* στην ίδια τοποθεσία με το *WebCrawler.jar*. Το αρχείο αυτό θα πρέπει να είναι σωστά συμπληρωμένο από το χρήστη. Οι παράμετροι που συμπληρώνονται στο αρχείο *crawl.properties* είναι οι εξής:

```
; Windows user
current_user =

; MySQL root user
root_user =

; MySQL root user password
root_pass =

; Configuration: how many words will be forming a phrase
; (optimal values: 3 to 4)
words_per_phrase =

; Configuration: how many web pages will be returned per search
; (optimal values: 1 to 2)
docs_added_per_search =

; Configuration: how many web pages will be searched at once
; (optimal values: 6-10)
urls_fetched_per_search =

; Bing Web Search API Key
bing_api =

; Number of result records to be presented when execution completes
number_of_results = 10

; Level of logging
; 0 -> ERROR & INFORMATION & DEBUG
; 1 -> ERROR & INFORMATION
; 2 -> ERROR
log_level =

; Home of Tesseract-OCR utility
; (NOTE: use DOUBLE BACKSLASH on paths!)
tesseract_home =

; Home of Xpdf suite
; (NOTE: use DOUBLE BACKSLASH on paths!)
xpdf_home =

; Configuration: keywords to include on your searches to filter out the results
; (Good for more accurate results)
; (NOTE: use SPACE to separate keywords)
; (If not used, insert the comment ";" symbol in front of the line)
keywords =
```

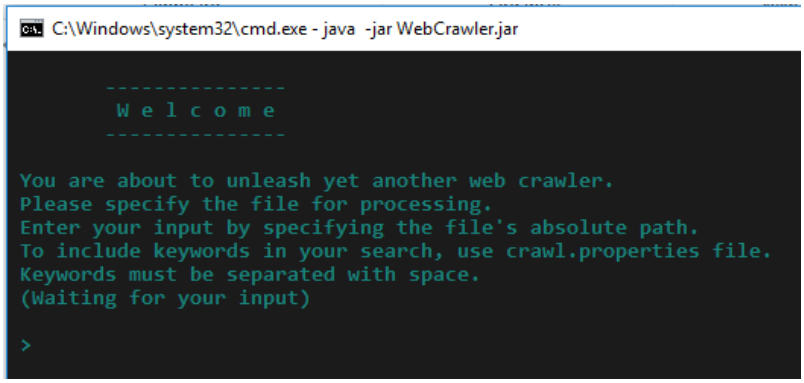
Εικόνα 3.1.1.1: Περιεχόμενα του αρχείου crawl.properties

Αφού το αρχείο έχει συμπληρωθεί κατάλληλα, ο χρήστης μπορεί να εκτελέσει το *WebCrawler.jar* ως εξής:



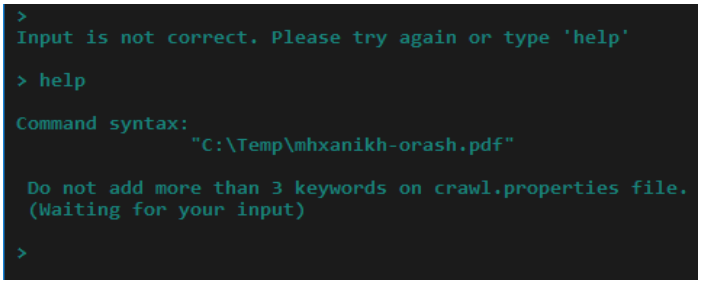
Εικόνα 3.1.1.2: Εκτέλεση του WebCrawler.jar από τη γραμμή εντολών των Windows

Αφού οι αρχικοί έλεγχοι ολοκληρωθούν επιτυχώς, θα εμφανιστεί το κεντρικό μενού του προγράμματος:



Εικόνα 3.1.1.3: Μενού προγράμματος

Στην περίπτωση που το αρχείο που προσδιορίσει ο χρήστης δεν είναι έγκυρο, τυπώνεται το αντίστοιχο μήνυμα, ενώ επίσης δίνεται η βοήθεια της επιλογής 'help'.



Εικόνα 3.1.1.4: Επιλογή 'help'

Ανάλογα με τον τύπο του αρχείου που επιλέγεται, θα εφαρμοστεί και η αντίστοιχη τεχνική ανάγνωσης και εξαγωγής κειμένου. Την εξαγωγή του κειμένου ακολουθεί μία επεξεργασία ώστε να δημιουργηθούν φράσεις κάποιων λέξεων. Οι φράσεις αυτές είναι εκείνες που το πρόγραμμα θα αναζητήσει στα επόμενα βήματα.

Το πρώτο βήμα της εκτέλεσης είναι η ανάγνωση και εξαγωγή του κειμένου που πρόκειται να ελεγχθεί για ομοιότητες. Οι τύποι αρχείων που υποστηρίζονται είναι των επεκτάσεων TXT, DOC/DOCX και PDF. Αρχικά καλείται μία γενική συνάρτηση για την ανάγνωση κειμένου ενώ η ίδια εσωτερικά ανάλογα με την επέκταση του αρχείου καλεί αντίστοιχα τη σωστή συνάρτηση που θα χειριστεί την εξαγωγή του κειμένου. Τα αρχεία TXT διαβάζονται εύκολα μέσω της Java. Αρχεία με επέκταση DOC/DOCX χειρίζονται σε ξεχωριστή μέθοδο, η οποία χρησιμοποιεί τη βιβλιοθήκη Apache POI (υλοποίηση ενός Java API για κείμενα που αναπτύχθηκαν σε προγράμματα της Microsoft όπως το MS WORD).

Σε αντίθεση με τους προηγούμενους τύπους αρχείων, για τα αρχεία της μορφής PDF η διαδικασία εξαγωγής κειμένου δεν είναι απλή. Ένα αρχείο PDF εκτός του ότι δεν δηλώνει ξεκάθαρα την κωδικοποίηση που έχει εφαρμόσει, μπορεί να ενσωματώνει παραμέτρους όπως προσαρμοσμένες γραμματοσειρές (custom fonts). Πολλά αρχεία PDF επίσης αντί για κείμενο που μπορεί ο χρήστης να χειριστεί κάνοντας αντιγραφή ή απλά μαρκάροντας, ενσωματώνουν το περιεχόμενο σε μορφή εικόνας. Σε αυτή την περίπτωση μέχρι και το μαρκάρισμα κειμένου είναι αδύνατο. Για τους λόγους που αναφέρθηκαν, η διαδικασία εξαγωγής κειμένου από ένα αρχείο PDF γίνεται πιο πολύπλοκη και δεν περιορίζεται στην κλήση μίας μονάχα μεθόδου. Αρχικά, γίνεται η χρήση της βιβλιοθήκης της Apache PDFBox. Αν η προσπάθεια δεν έχει επιτυχία, στη συνέχεια χρησιμοποιείται το εκτελέσιμο αρχείο pdftotext.exe. Το αρχείο αυτό είναι ένα πρόγραμμα από τη σουίτα Xpdf, μίας ανοιχτού λογισμικού εφαρμογής για τη διαχείριση αρχείων PDF. Το pdftotext έχει τη δυνατότητα να διαβάσει σε κωδικοποίηση UTF-8 με την κατάλληλη παράμετρο ρύθμισης. Σε περίπτωση που και αυτή η προσπάθεια δεν έχει επιτυχία, γίνεται εφαρμογή τεχνικής OCR. Σε μία ξεχωριστή συνάρτηση, η λειτουργία της βασίζεται στα προγράμματα convert.exe και tesseract.exe. Το convert είναι ένα πρόγραμμα της σουίτας ImageMagick, ενός συνόλου εφαρμογών που είναι διαθέσιμες δωρεάν και

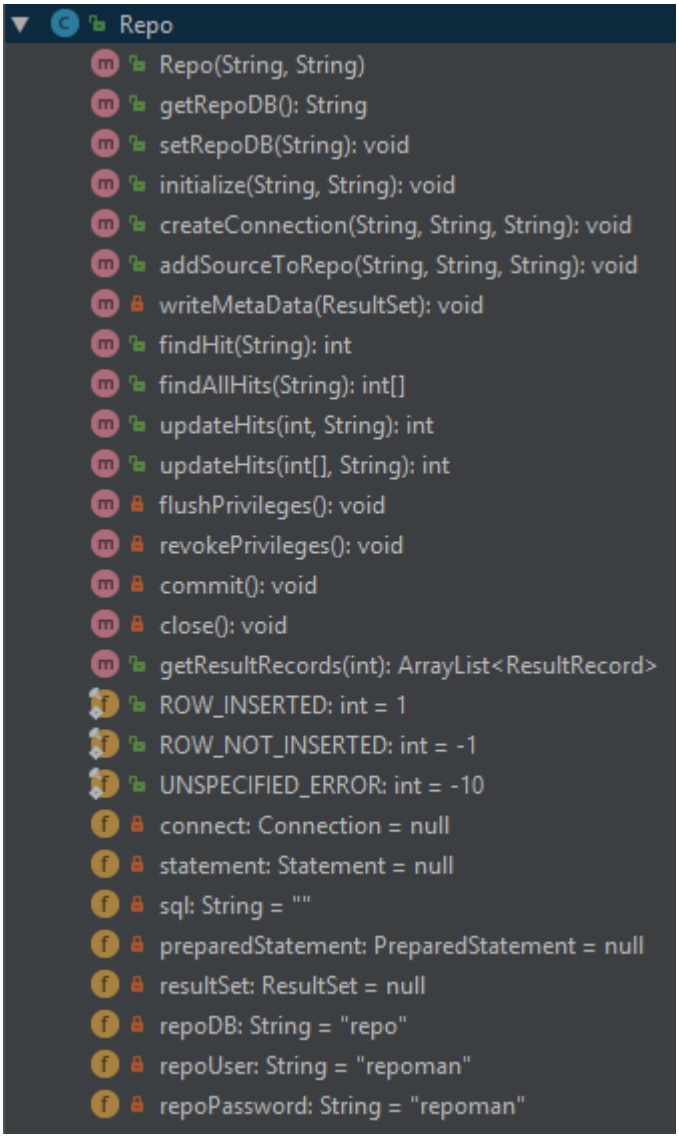
χρησιμεύουν στη δημιουργία, επεξεργασία, σύνθεση και μετατροπή εικόνων. Πιο συγκεκριμένα, το convert είναι μία εφαρμογή που χειρίζεται τη μετατροπή εικόνας σε διαφορετικές μορφές. Στην περίπτωση του PDF αρχείου, γίνεται η χρήση του convert ώστε να μετατραπεί το PDF σε εικόνα υψηλής ευκρίνειας επέκτασης TIFF. Στη συνέχεια η εικόνα που δημιουργήθηκε θα επεξεργαστεί από το εκτελέσιμο αρχείο tesseract.exe. Το tesseract είναι μία ανοιχτού λογισμικού μηχανή OCR. Με αυτό τον τρόπο η εικόνα που δημιουργήθηκε θα "αναγνωσθεί" ώστε να γίνει η εξαγωγή του κειμένου. Η εικόνα που προκύπτει από το convert πρέπει να είναι υψηλής ευκρίνειας για να είναι όσο το δυνατόν πιο ακριβής η ανάλυση από το tesseract. Ουσιαστικά η εικόνα διαβάζεται τμηματικά ώστε να γίνει ταύτιση σχημάτων με γνωστά ελληνικά γράμματα και να δημιουργηθεί τελικά το κείμενο. Πρόκειται για μία διαδικασία απαιτητική και σε χρόνο αλλά και σε πόρους του συστήματος.

Αφού το κείμενο εξαχθεί από το αρχείο, ακολουθεί η επεξεργασία του ώστε να χωριστεί αρχικά σε λέξεις και τελικά σε φράσεις. Ο αριθμός των λέξεων που αποτελούν μία φράση είναι παραμετροποιήσιμος. Με τις κατάλληλες μεθόδους, η τροποποιήσεις προκύπτουν από τη χρήση 'κανονικών' εκφράσεων (regular expressions). Μετά την εφαρμογή αυτών των μεθόδων το κείμενο έχει μετατραπεί σε μία λίστα από φράσεις. Κάθε φράση θα αναζητηθεί αρχικά στο repo και τελικά στο διαδίκτυο αν κάτι τέτοιο χρειαστεί.

3.1.2. ΑΝΑΖΗΤΗΣΗ ΦΡΑΣΗΣ ΣΤΗ ΒΑΣΗ REPO

Η αναζήτηση στο αποθετήριο αποτελεί τη βασική αρχή στην υλοποίηση του αλγορίθμου ταύτισης πλαισίου (CMA). Το αποθετήριο θα χτίζεται διαρκώς είτε με την καταχώρηση νέων κειμένων που θα διευρύνουν το σύνολο τιμών των καταστάσεων που θα ελέγχονται, είτε με την ενημέρωση κειμένων που ήδη είναι καταχωρημένα σε περίπτωση που εμφανίζουν ξανά ταύτιση. Σε αυτό το βήμα γίνεται έλεγχος σε κάθε φράση για να διαπιστωθεί αν υπάρχει ταύτιση με κάποιο γνωστό κείμενο.

Η επικοινωνία της βάσης δεδομένων με τη Java, όπως αναφέρθηκε παραπάνω, χειρίζεται από την κλάση Repo. Στην Εικόνα 3.1.2.1 παρουσιάζονται οι μέθοδοι που είναι υλοποιημένες στην κλάση Repo.



Εικόνα 3.1.2.1: Υπογραφές μεθόδων κλάσης Repo

Μετά από το χωρισμό των λέξεων σε φράσεις, ξεκινάει το αρχικό στάδιο ελέγχου του προγράμματος. Κάθε φράση θα αναζητηθεί στο αποθετήριο πριν χρειαστεί να αναζητηθεί στο διαδίκτυο. Ο τρόπος που γίνεται η αναζήτηση έχει προσαρμοστεί έτσι ώστε να ενημερωθούν όλα τα κείμενα τα οποία παρουσιάζουν ομοιότητα. Με αυτό τον τρόπο αναδεικνύονται κείμενα με περισσότερες ταυτίσεις. Αυτό μεταφράζεται στο γεγονός ότι κάποιο κείμενο που συγκεντρώνει μεγαλύτερο αριθμό ταυτίσεων στον εντοπισμό ομοιοτήτων σε σχέση με άλλα,

αποτελεί καλύτερη υποψήφια πηγή - κρούσμα λογοκλοπής. Όταν εντοπιστεί μία φράση σε ένα ή περισσότερα κείμενα στο αποθετήριο, τότε ενημερώνεται ο πίνακας HITS αυξάνοντας την τιμή της στήλης hit_count κατά 1 στα κείμενα όπου βρέθηκε η ταύτιση (χρήση του ξένου κλειδιού hit_doc_id).

```
preparedStatement = connect.prepareStatement( sql: "SELECT doc_id FROM documents WHERE doc_text LIKE ?");
preparedStatement.setString( parameterIndex: 1,  x: "%" + phrase + "%");
resultSet = preparedStatement.executeQuery();
```

Εικόνα 3.1.2.2: Αναζήτηση φράσης στα κείμενα του αποθετηρίου

Από τον τρόπο χρήσης της βάσης δεδομένων, είναι προφανές πως όσο πιο πολύ χρόνο εκτελείται το πρόγραμμα και όσες πιο πολλές αναζητήσεις κάνει, τόσο περισσότερο έγκυρο γίνεται στο να εντοπίσει αποτελέσματα ομοιότητας. Οι πρώτες αναζητήσεις ουσιαστικά χρησιμεύουν στο να ετοιμάζουν το περιβάλλον αναζητήσεων, με αρκετά κείμενα να εισέρχονται στο αποθετήριο λόγω του μη εντοπισμού ομοιοτήτων ως προς τις φράσεις των κειμένων. Όσο περισσότερα κείμενα εισέρχονται στο αποθετήριο, τόσες περισσότερες ταυτίσεις γίνονται μελλοντικά, λειτουργώντας σε μία αντιστρόφως ανάλογη σχέση.

Θα μπορούσε να συμπεράνει κανείς πως τα περισσότερα κείμενα μεταφράζονται σε μεγαλύτερη σπατάλη χώρου στο δίσκο του συστήματος. Με τη MySQL και γενικότερα χρησιμοποιώντας μία βάση δεδομένων, ο χώρος που καταλαμβάνει ένα κείμενο είναι αρκετά μικρότερος από όσο θα νόμιζε κάποιος. Για παράδειγμα, σε μία βάση που έχουν αποθηκευτεί περίπου 1600 κείμενα τουλάχιστον μεσαίου χαρακτηρισμού ως προς την ποσότητα, ο χώρος στο δίσκο που χρειάζεται για να τα ελέγχει είναι λιγότερος των 300MB.

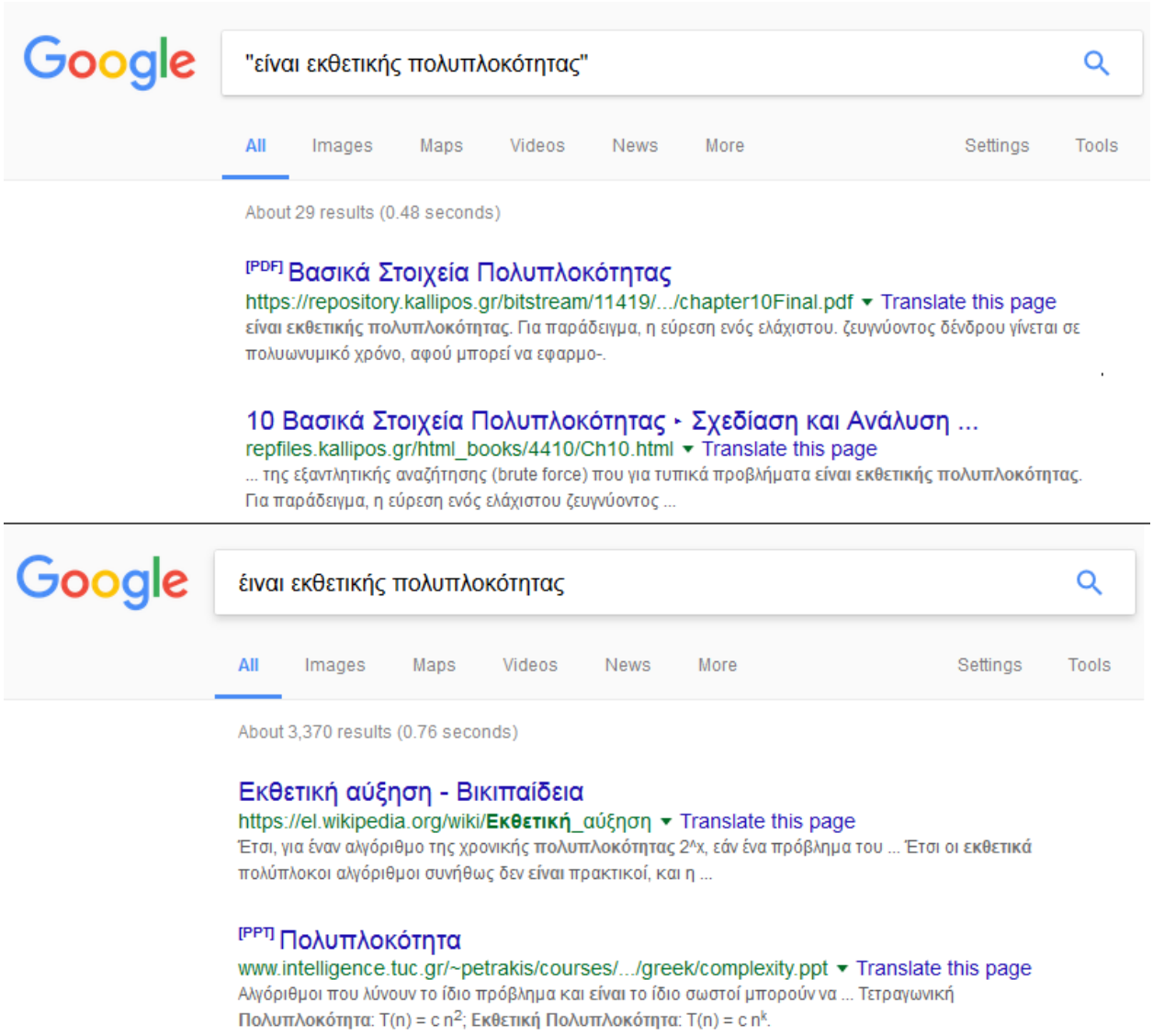
3.1.3. ΑΝΑΖΗΤΗΣΗ ONLINE

Η αναζήτηση στο αποθετήριο δεν είναι επιτυχείς, οπότε σαν τελικό βήμα διεξάγεται αναζήτηση στο διαδίκτυο. Η διαδικτυακή αναζήτηση διεξάγεται με επιτυχία στην πλειοψηφία των περιπτώσεων. Για κάθε φράση επιλέγεται τυχαία η μηχανή αναζήτησης που θα χρησιμοποιηθεί. Όταν βρεθεί κάποιο κείμενο που να παρουσιάζει ομοιότητα, πριν την επεξεργασία της επόμενης φράσης, το κείμενο εισάγεται στη βάση δεδομένων του συστήματος ώστε να είναι διαθέσιμο σε επόμενες γρήγορες αναζητήσεις. Ουσιαστικά σε αυτό το σημείο το πρόγραμμα “μαθαίνει” το συγκεκριμένο κείμενο (κατάσταση) και σε μελλοντική αναζήτηση μπορεί να το “αναγνωρίσει” (εφαρμόσει) σε κάποια φράση. Με αυτό τον τρόπο τεκμηριώνεται η περιγραφή του συστήματος ως προς την ιδιότητα ταύτισης πλαισίου (context-matching).

Για την αναζήτηση των φράσεων στο διαδίκτυο, αρχικά αναλύθηκε η περίπτωση χρήσης ελληνικών ψηφιακών αποθετηρίων στην αναζήτηση για ελληνικά δοκίμια, η οποία δεν επιλέχθηκε. Στη συνέχεια εξετάστηκε και τελικώς εφαρμόστηκε η χρήση γνωστών μηχανών αναζήτησης. Πρόκειται για της υπηρεσίες που παρέχουν οι Google, Bing και Yahoo, ενώ επίσης εξετάστηκαν και οι μηχανές αναζήτησης DuckDuckGo και StartPage.

Με την αναζήτηση σε ηλεκτρονικά αποθετήρια από την αρχή το πρόγραμμα θα είχε πρόσβαση σε ένα πολύ μεγάλο όγκο ελληνικών επιστημονικών κειμένων. Ο λόγος που δεν επιλέχτηκε η αναζήτηση σε ηλεκτρονικά/ψηφιακά αποθετήρια είναι το περιορισμένο επίπεδο της αναζήτησης που παρέχεται. Συγκεκριμένα, τα APIs για την επικοινωνία με τα αποθετήρια όπως το Helios, το Εθνικό Αρχείο Διδακτορικών Διατριβών και το openarchives.gr δεν επιτρέπουν την αναζήτηση στο αυτούσιο κείμενο. Όταν πρόκειται για έλεγχο κειμένου, η αναζήτηση που εκτελείται αφορά μονάχα λέξεις κλειδιά και όχι αυτούσιες φράσεις. Επίσης, ο χρήστης έχει πρόσβαση σε meta-data της κάθε εγγραφής και όχι σε ολόκληρο το αυθεντικό κείμενο, συνεπώς δεν γίνεται να προκύψουν αποτελέσματα ρεαλιστικά με αυτό τον τρόπο.

Για την ευρύτερη φύση αποτελεσμάτων και την πρόσβαση σε απεριόριστο υλικό αναζήτησης, εξετάστηκε η χρήση των γνωστών μηχανών αναζήτησης. Όλες οι γνωστές μηχανές αναζήτησης υποστηρίζουν την επιλογή προηγμένης αναζήτησης. Εισάγοντας ειδικά σύμβολα και λέξεις κλειδιά στην είσοδο του κειμένου, η κάθε μηχανή αναγνωρίζει και εφαρμόζει ειδικές λειτουργίες. Για παράδειγμα, η αναζήτηση κειμένου που περικλείεται σε σύμβολα “” αναζητείται αυτούσια σε κείμενα.



Εικόνα 3.1.3.1: Παράδειγμα αναζήτησης αυτούσιας φράσης και απλής αναζήτησης

Η συγκεκριμένη τεχνική εφαρμόζεται σε όλες τις αναζητήσεις του προγράμματος και είναι βασικότατη στη γενικότερη ιδέα της ταύτισης ομοιοτήτων και κατά επέκταση εντοπισμού λογοκλοπών. Σε συνδυασμό με την αναζήτηση αυτούσιας φράσης, στην αναζήτηση συμπεριλαμβάνονται και οι λέξεις-κλειδιά που ο χρήστης έχει ορίσει κατά την επιλογή του κειμένου. Με αυτό τον τρόπο, φράσεις που θα μπορούσαν να εντοπίζονται αυτούσιες σε

πηγές που δεν έχουν θεματολογική σχέση με το κείμενο που αναλύεται, δεν θα περιλαμβάνονται στα αποτελέσματα. Έτσι εφαρμόζεται ένα είδους φίλτρου, προσδιορίζοντας μία κατηγοριοποίηση με βάση τη θεματολογία και η ποιότητα των αποτελεσμάτων που προκύπτουν είναι σημαντικά βελτιωμένη.

Μελετώντας την υλοποίηση του συστήματος της εργασίας, γίνεται κατανοητό πως ο αριθμός των ηλεκτρονικών αναζητήσεων είναι αρκετά μεγάλος. Αυτό σημαίνει πως θα εκτελείται υψηλός αριθμός αναζητήσεων σε αρκετά μικρό χρονικό διάστημα. Αυτή η μεθοδολογία εντάσσεται στα πλαίσια του όρου web scraping. Οι ενέργειες που εκτελούνται στα πλαίσια του web scraping, είτε πρόκειται για αναζητήσεις είτε για αντιγραφή αρχείων και γενικότερα για πρόσβαση σε πληροφορίες και παροχές υπηρεσιών, ξεκινάνε από ένα λογισμικό που ορίζεται ως web crawler. Κατά αυτό τον ορισμό, χρησιμοποιείται προσαρμοσμένο λογισμικό (crawler) στην ανάκτηση δεδομένων από διάφορες σελίδες του διαδικτύου. Είναι μία τεχνική που μπορεί να αποδειχτεί ιδιαίτερα χρήσιμη, αφού με την κατάλληλη παραμετροποίηση ένας χρήσης αποκτά άμεση πρόσβαση σε τεράστιο όγκο χρήσιμης πληροφορίας. Από την άλλη μεριά, παρόμοιου τύπου λογισμικό κάνει κατάχρηση των υπηρεσιών που παρέχονται από διάφορες ιστοσελίδες. Αν αυτό συμβαίνει σε μαζικό βαθμό, αποτέλεσμα είναι μέχρι και μείωση της ποιότητας παροχής υπηρεσιών (QoS). Επιπροσθέτως, υπάρχουν νομικά εμπόδια για την εφαρμογή τεχνικών web scraping. Πολλές υπηρεσίες εφαρμόζουν συγκεκριμένες τεχνικές για την αντιμετώπιση και απαγόρευση αυτών των ενεργειών. Μερικά συστήματα βασίζονται στη διατήρηση δεδομένων σχετικά με τη σύνδεση του χρήστη (session) και κάνουν χρήση των cookies του προγράμματος περιήγησης ώστε να καταγράφουν τις δραστηριότητες του χρήστη. Άλλος ένας τρόπος αντιμετώπισης κατάχρησης υπηρεσιών, υλοποιείται με την απαγόρευση πρόσβασης στα δεδομένα των διαχειριστών (servers) στην περίπτωση που εντοπίσουν κάποια χαρακτηριστικά, όπως είναι η ασυνήθιστη κίνηση και υψηλός βαθμός λήψης σε πολύ μικρό διάστημα ειδικά από μία συγκεκριμένη IP. Ένα άλλο κριτήριο είναι ο εντοπισμός εκτέλεσης επαναληπτικών κινήσεων, κάτι που προσδίδεται με την αλγοριθμική φύση του λογισμικού (crawler) και δεν χαρακτηρίζει τον τρόπο που χρησιμοποιεί τις υπηρεσίες ο άνθρωπος (ScrapeHero, 2014). Το πρόγραμμα έχει υλοποιηθεί ώστε να χρησιμοποιεί τις μηχανές αναζήτησης Google, Yahoo και Bing όπως ήδη αναφέρθηκε. Η χρήση αποκλειστικά μίας μηχανής αναζήτησης παρουσιάζει τεχνικούς περιορισμούς.

3.1.3.1. GOOGLE SEARCH

Για τη χρήση των υπηρεσιών της Google, η διασύνδεση γίνεται με αντικείμενα jsoup. Η jsoup είναι μία ανοιχτού κώδικα βιβλιοθήκη της Java σχεδιασμένη να εξάγει και να τροποποιεί δεδομένα αποθηκευμένα σε αρχεία της μορφής HTML. Συνοπτικά, η φράση προς αναζήτηση προωθείται στο Google και τα αποτελέσματα χειρίζονται σαν αντικείμενα jsoup τα οποία ελέγχονται τελικά από τη Java.

Η μηχανή αναζήτησης της Google κάνει απαγόρευση υπηρεσιών στην περίπτωση που αναγνωρίσει κάποιο χαρακτηριστικό που να εντάσσεται στα πλαίσια του web scraping. Πιο συγκεκριμένα, απαγορεύει από το λογισμικό (crawler) να χρησιμοποιήσει υπηρεσίες με το να απαιτεί έναν έλεγχο CAPTCHA για τη συνέχιση της παροχής υπηρεσιών. Σε ένα παράδειγμα υλοποίησης με τη χρήση μονάχα των υπηρεσιών της Google, αυτό αποτελεί πρόβλημα. Πιο συγκεκριμένα, μετά από κάποιο αριθμό αναζητήσεων μέσω του προγράμματος, μελλοντικές αναζητήσεις αποτυγχάνουν. Αυτό συμβαίνει επειδή η μηχανή αναζήτησης της Google αντιμετωπίζει το σύστημα το οποίο στέλνει πολλά requests σαν bot και αποκλείει την παροχή υπηρεσιών προς εκείνο. Χαρακτηριστικά exceptions που αντιμετωπίζονται σε επίπεδο κώδικα είναι τα παρακάτω:

- *org.jsoup.HttpStatusException: HTTP error fetching URL. Status=503, URL=...*

  503: Service Unavailable

- *org.jsoup.HttpStatusException: HTTP error fetching URL. Status=403, URL=...*

  403: Forbidden

Για το λόγο αυτό, έχουν εφαρμοστεί συγκεκριμένες τεχνικές. Με την προσθήκη “τυχαιότητας” στο ρυθμό των αναζητήσεων, η περίπτωση να αναγνωριστούν αλγοριθμικά κινήσεις που εμφανίζουν χαρακτηριστικά επαναληπτικότητας μειώνεται. Τυχαιότητα εισάγεται με τη χρήση καθυστέρησης μεταξύ των αναζητήσεων.

```
// a fair amount of waiting time between searches for 0-0.7 seconds
timeout = genTimeout.nextInt( bound: 701);
```

Εικόνα 3.1.3.1.1: Εισάγοντας ψευδο-τυχαία χρονική καθυστέρηση

Τυχαιότητα εφαρμόζεται και στην επιλογή της μηχανής αναζήτησης που θα επιλεχθεί για την αναζήτηση κάθε φράσης.

3.1.3.2. BING SEARCH

Η μηχανή αναζήτησης της Microsoft, παρέχει ένα ξεκάθαρο API για την ενσωμάτωση στον πηγαίο κώδικα και την πρόσβαση στην υπηρεσία αναζήτησης. Προϋπόθεση της χρήσης του API είναι η εγγραφή του χρήστη. Ο χρήστης πρέπει να δημιουργήσει ένα κλειδί από το Microsoft Azure. Η δημιουργία λογαριασμού είναι δωρεάν. Η υπηρεσία που χρειάζεται είναι το Bing Search API, πιο συγκεκριμένα Bing Web Search API, μία από τις υπηρεσίες των Cognitive Services. Το κλειδί για την υπηρεσία που παρέχει το Bing Web Search API και δημιουργείται δωρεάν εξυπηρετεί έναν πεπερασμένο αριθμό αναζητήσεων, ενώ παράλληλα λήγει με τη συμπλήρωση ενός (1) μήνα από την ημέρα δημιουργίας του κλειδιού. Λόγο του περιορισμένου αριθμού αναζητήσεων, οι χρήση των υπηρεσιών μόνο της αναζήτησης Bing δεν εξυπηρετεί απόλυτα τη λειτουργία του συστήματος.

Προγραμματιστικά, τα αποτελέσματα αναζητήσεων επιστρέφονται και χειρίζονται από τη Java σε μορφή JSON. JSPON (JavaScript Object Notation) είναι μία ελαφριά μορφή εσωτερικής ανταλλαγής δεδομένων. Είναι βασισμένη σε ένα υποσύνολο της γλώσσας JavaScript και πρόκειται για μορφή κειμένου τελείως ανεξάρτητη της γλώσσας του προγράμματος (Introducing JSON, 2013). Θα μπορούσε να συγκριθεί με την περιγραφική μορφή της XML.

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateNewDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Close", "onclick": "CloseDoc()"}
    ]
  }
}}

The same text expressed as XML:

<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

Εικόνα 3.1.3.2.1: Παράδειγμα ίδιας περιγραφής σε μορφές JSON και XML

3.1.3.3. YAHOO SEARCH

Η χρήση της μηχανής αναζήτησης Yahoo υλοποιείται με τη χρήση της βιβλιοθήκης jsoup. Δεν χρησιμοποιείται κάποιο προσαρμοσμένο API για την αλληλεπίδραση και χρήση υπηρεσιών της Yahoo. Μέσα από τον κώδικα του προγράμματος, αξιοποιείται η μορφή της σύνθετης αναζήτησης που παρέχει η μηχανή αναζήτησης. Αφού δημιουργηθεί η μορφή του κειμένου προς αναζήτηση, όπως φαίνεται στην Εικόνα 3.1.3.3.1). Μερικές από τις επιμέρους παραμέτρους που μπορούν να προσδιοριστούν αποτελούν η κωδικοποίηση (στην Εικόνα 3.1.3.3.1 έχει επιλεχθεί η UTF-8), η αναζήτηση σε συγκεκριμένο domain, η μορφή του αρχείου (για παράδειγμα HTML ή PDF), η γλώσσα του κειμένου και η χώρα.

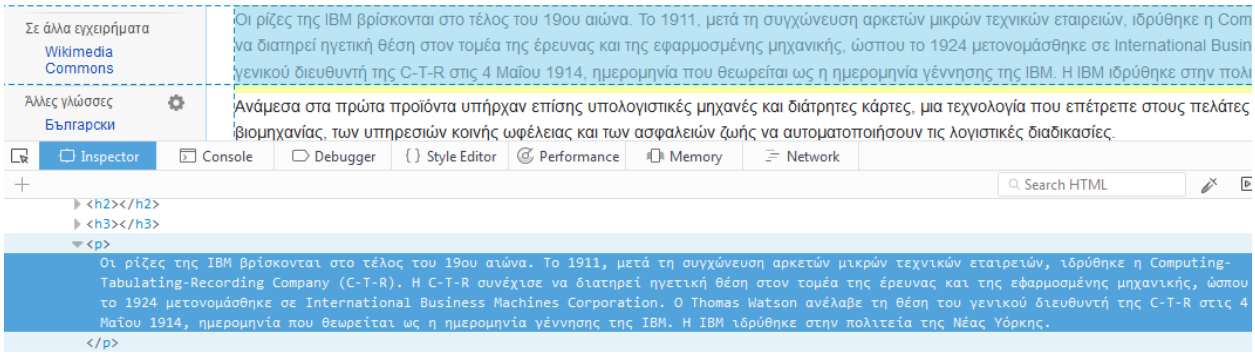
```
url = "https://uk.search.yahoo.com/search?n=" + URLS_RETURNED_PER_SEARCH +
        "&ei=UTF-8" +
        "&va_vt=any" +
        "&vo_vt=any" +
        "&ve_vt=any" +
        "&vp_vt=any" +
        "&vst=0" +
        "&vf=all" +
        "&vm=i" +
        "&fl=0" +
        "&fr=sfp" +
        "&p=" + keywords + "+\"\"" + phrase + "\"\"" +
        "&vs=";
Document doc2 = Jsoup.connect(url).timeout(5000).get();
Elements elements = doc2.getElementsByClass( className: "title").select("a[href]");
```

Εικόνα 3.1.3.3.1: Κώδικας σε Java για την διεξαγωγή σύνθετης αναζήτησης Yahoo

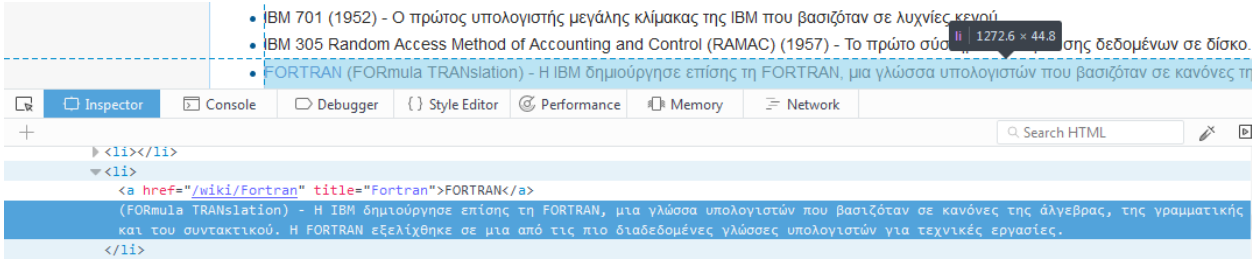
Αφού αποσταλεί το αίτημα αναζήτησης, εξάγεται από τα αποτελέσματα το URL όπως φαίνεται στο παραπάνω παράδειγμα, επιλέγοντας τα στοιχεία τύπου “a[href]” τα οποία είναι τα links των πηγών που εντοπίστηκαν αποτελέσματα.

3.1.3.4. ΕΝΗΜΕΡΩΣΗ ΤΗΣ ΒΑΣΗΣ REPO

Μετά από τον εντοπισμό της πηγής που περιέχει την επιθυμητή φράση, ακολουθεί η ενημέρωση της βάσης δεδομένων repo με το νέο κείμενο. Ανάλογα με τη μορφή της πηγής που εντοπίστηκε, ακολουθείται διαφορετικός τρόπος χειρισμού. Στην περίπτωση που η πηγή είναι ένα αρχείο γνωστής μορφής κειμένου μεταξύ αυτών που παρουσιάστηκαν στο Κεφάλαιο 3.1.1, το αρχείο αποθηκεύεται τοπικά και το κείμενο εξάγεται με την κατάλληλη τεχνική. Αντίστοιχα, όταν η πηγή είναι μία ηλεκτρονική σελίδα η εξαγωγή κειμένου εφαρμόζεται άμεσα με χρήση της βιβλιοθήκης jsoup. Τα πεδία που έχουν ενδιαφέρον ως προς το κείμενο που περιέχουν είναι εκείνα που βρίσκονται στα CSS tags <p> και <li>. Το βασικό κείμενο των ηλεκτρονικών σελίδων στην πλειοψηφία ανήκει σε αυτά τα tags. Στις παρακάτω εικόνες απεικονίζονται παραδείγματα κειμένου που ανήκουν στα tags <p> και <li> σε μία ιστοσελίδα (“IBM”, el.wikipedia.org).



Εικόνα 3.1.3.4.1: Κείμενο που ανήκει σε tag <p> ιστοσελίδας (IBM, 2016).



Εικόνα 3.1.3.4.2: Κείμενο που ανήκει σε tag <li> ιστοσελίδας (IBM, 2016).

Την εξαγωγή κειμένου ακολουθεί η ενημέρωση της βάσης δεδομένων με το νέο κείμενο. Το κείμενο εισάγεται στο repo σαν νέα εγγραφή του πίνακα DOCUMENTS, ενώ επίσης καταχωρείται μία νέα εγγραφή στον πίνακα HITS με τον μοναδικό αριθμό του νέου κειμένου και τη φράση στην οποία εμφάνισε ομοιότητα.

3.2. ΕΡΜΗΝΕΙΑ ΑΠΟΤΕΛΕΣΜΑΤΩΝ

Κατά την εκτέλεση του προγράμματος, η βάση δεδομένων (repo) μεγαλώνει σταδιακά και ενημερώνεται με νέες ταυτίσεις. Τόσο η εισαγωγή νέων κειμένων, όσο και η ενημέρωση των ήδη υπάρχων σε περίπτωση ταυτίσεων με νέες φράσεις, συντελεί στη διαμόρφωση της βάσης δεδομένων σε ένα συνεχώς πιο έγκυρο αποθετήριο αναφοράς. Οι πρώτες φράσεις που θα αναλυθούν δεν αναμένεται να ταυτιστούν άμεσα με κείμενα της βάσης δεδομένων. Είναι λογικό οι πρώτες εκτελέσεις να λειτουργούν ως αρχικοποίηση του repo και την προετοιμασία για μελλοντικές ταυτίσεις. Όταν η εκτέλεση του προγράμματος ολοκληρωθεί, ο χρήστης μπορεί να αναλύσει τον τρόπο που έχει διαμορφωθεί η βάση δεδομένων. Το *WebCrawler* παρουσιάζει στο χρήστη έναν πίνακα με τα 10 (προεπιλογή) κείμενα που εντοπίστηκαν οι περισσότερες ταυτίσεις. Χρησιμοποιώντας SQL εντολές, ο χρήστης μπορεί να αποκτήσει πληροφορίες για τα κείμενα που παρουσίασαν τις περισσότερες ομοιότητες σε σχέση με το κείμενο που ελέγχθηκε, ποιες φράσεις ταυτίστηκαν και σε ποια κείμενα, καθώς και στατιστικά στοιχεί όπως το πόσα κείμενα έχουν εισαχθεί στο αποθετήριο και με τι σειρά εμφανίστηκαν οι ταυτίσεις.

4. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η εφαρμογή του θεωρητικού μοντέλου που παρουσιάστηκε παραπάνω είναι γεγονός. Σε αξιόλογο βαθμό, το μοντέλο που υλοποιήθηκε μπορεί να μελετηθεί ώστε να να προκύψουν συμπεράσματα με ενδιαφέρον σε διαφορετικούς τομείς.

4.1. ΑΠΟΔΕΙΞΗ ΩΣ ΠΡΟΣ ΤΗΝ ΤΕΧΝΙΚΗ CONTEXT-MATCHING

Η προσέγγιση 'πλαισιακής' ταύτισης, θεωρητικά προσδίδει στο σύστημα την ιδιότητα της αξιόλογης λειτουργίας ελαχιστοποιώντας το κόστος βελτιστοποίησης όταν αντιμετωπίζονται καταστάσεις όμοιες με προηγούμενες, που έχουν ήδη αναλυθεί και βελτιστοποιηθεί. Η δημιουργία αυτού του είδους μνήμης είναι καθοριστική σε αυτό το αποτέλεσμα. Στην πράξη αυτή η θεωρία επαληθεύεται. Παρατηρείται πως η εκμετάλλευση αυτού του είδους μνήμης παρουσιάζει εμφανή αποτελέσματα. Σε ένα παράδειγμα εκτέλεσης, εξετάστηκε ένα προσαρμοσμένο κείμενο το οποίο είχε επεξεργαστεί με σκοπό να αποτελείται από ένα αρχικό κείμενο κάποιας πηγής (της σελίδας Wikipedia) τροποποιημένο σε μεγάλη έκταση με εμβόλιμα τοποθετημένες φράσεις από διαφορετικές πηγές. Το κείμενο αναλύθηκε και χωρίστηκε σε 1210 φράσεις. Από τις 1210 φράσεις που αναζητήθηκαν, οι 781 από αυτές ταυτίστηκαν με κείμενα του repo, ενώ για τις υπόλοιπες 429 χρειάστηκε να διεξαχθεί διαδικτυακή αναζήτηση και να εισάγουν νέα κείμενα στο repo. Τελικά η βάση δεδομένων διατήρησε πληροφορίες για 283 κείμενα (μερικές φράσεις δεν είναι δυνατό να βρεθούν ούτε στο διαδίκτυο, η απόκλιση είναι λογική) ενώ εξυπηρέτησε 781 άμεσες αναζητήσεις. Αυτό σημαίνει πως το σημαντικό ποσοστό του 64% των συνολικών αναζητήσεων εξυπηρετήθηκαν με τη γρήγορη λύση που ήταν γνωστή από το σύστημα.

Από την άλλη μεριά, όταν το κείμενο που ελέγχεται δεν αποτελεί τελικά προϊόν λογοκλοπής και δεν εμφανίζει ιδιαίτερη ταύτιση με άλλα κείμενα, τα αποτελέσματα δεν είναι της ίδιας τάξης. Ελέγχοντας ένα κείμενο 1491 φράσεων, μόνο οι 264 εξυπηρετήθηκαν από το αποθετήριο. Για τις υπόλοιπες 1227 φράσεις χρειάστηκε να γίνει διαδικτυακή αναζήτηση, ενώ επίσης η βάση δεδομένων που χρησιμοποιήθηκε ξεπέρασε το διπλάσιο μέγεθος της βάσης δεδομένων από το προηγούμενο παράδειγμα. Ένα ποσοστό της τάξης του 21% δεν χαρακτηρίζει ένα ιδιαίτερα αποδοτικό σύστημα δεδομένου ότι το κόστος αναζητήσεων είναι αρκετά μεγάλο και πρέπει να αποφευχθεί.

Αυτός ο τρόπος ερμηνείας που παρουσιάστηκε, επεκτείνεται στην αρχιτεκτονική διάφορων υπολογιστικών συστημάτων. Όπως και το παράδειγμα στην Εικόνα 2.1.1 του δεύτερου μέρους της εργασίας, η αρχιτεκτονική αυτή βελτιστοποιεί το χρόνο που το σύστημα χρειάζεται για να

εξυπηρετήσει μία κατάσταση, ενώ ταυτόχρονα οι απαιτήσεις λειτουργίας όσο αφορά τους πόρους του συστήματος διατηρούνται χαμηλά, αν όχι στο ελάχιστο. Βασική προϋπόθεση για να λειτουργήσει αυτό το μοντέλο είναι η μελέτη κατά πόσο θα μπορούσε πραγματικά το σύστημα να επωφεληθεί από την εφαρμογή. Για να κριθεί χρήσιμη η εφαρμογή στο σύστημα θα πρέπει να έχουν αναγνωριστεί περιπτώσεις που να χαρακτηρίζονται από επαναληπτικότητα και να αποτελούν αξιόλογο ποσοστό στη συνολική λειτουργία του συστήματος.

4.2. ΑΠΟΔΕΙΞΗ ΩΣ ΠΡΟΣ ΤΟΝ ΕΝΤΟΠΙΣΜΟ ΛΟΓΟΚΛΟΠΩΝ

Στον εντοπισμό λογοκλοπών τα συμπεράσματα που προκύπτουν από τη χρήση του προγράμματος εκτείνονται σε δύο περιοχές: αλγοριθμικά και τεχνικά.

Σχετικά με τον αλγόριθμο που χρησιμοποιήθηκε, τα αποτελέσματα είναι θετικά. Το πρόγραμμα είναι σε θέση να αναγνωρίζει ταυτίσεις και να εφαρμόζει ενός τύπου ταξινόμηση με βάση ποια κείμενα παρουσιάζουν τις περισσότερες ταυτίσεις με εκείνο που ελέγχεται. Χρησιμοποιώντας τεχνική που περιστρέφεται γύρω από την 'πλαισιακή' αναγνώριση επιτυγχάνεται η ταξινόμηση των αποτελεσμάτων καθώς επίσης η δυνατότητα της δυναμικής διαμόρφωσης του αποθετηρίου. Το γεγονός ότι το αποθετήριο χτίζεται δυναμικά μπορεί να επηρεάσει σημαντικά την ποιότητα των αποτελεσμάτων που διεξάγονται και να χρήσει το πρόγραμμα πιο αξιόπιστο.

Όσο αφορά τα αποτελέσματα που προκύπτουν βάσει των τεχνικών παραμέτρων, επικρατεί μία ουδετερότητα. Η προσέγγιση που ακολουθήθηκε στην ανάλυση του κειμένου και πιο συγκεκριμένα στο χωρισμό σε λέξεις και τελικά σε φράσεις, είναι κάτι το οποίο φαίνεται να λειτουργεί ως ένα πρώτο και βασικό μέρος. Πιο συγκεκριμένα, χωρίζοντας τις προτάσεις σε λέξεις και δημιουργώντας φράσεις, υπάρχουν πολλές περιπτώσεις όπου η τελευταία λέξη μίας πρότασης ανήκει στην ίδια φράση με την πρώτη λέξη της επόμενης πρότασης. Μία φράση όπως αυτή, δύσκολα θα παρουσιάσει έγκυρη ταύτιση με κάποια διαδικτυακή πηγή εκτός και αν στο άρθρο που ελέγχεται έχει γίνει χρήση διαδοχικών αυτούσιων προτάσεων. Είναι κάτι που δεν χαρακτηρίζεται συνηθισμένο. Επίσης, η εισαγωγή κειμένου στο αποθετήριο είναι μία διαδικασία που χρήζει βελτίωση. Παρατηρούνται κείμενα από ιστοσελίδες που αγνοούνται κατά την εξαγωγή του κειμένου, με αποτέλεσμα σε μελλοντική αναζήτηση στο αποθετήριο, να μην υπάρχουν μέρη κειμένου το οποίο όμως βρίσκεται στο αποθετήριο και θα μπορούσε να παρουσιάσει ταύτιση. Παρά τις τεχνικές ατέλειες που υπάρχουν, το πρόγραμμα που αναπτύχθηκε θα μπορούσε να χαρακτηριστεί πως συνεισφέρει στην έρευνα εντοπισμού λογοκλοπών σε ελληνικά δοκίμια.

Επιστρέφοντας στο προηγούμενο παράδειγμα, το κείμενο που αποτελεί προϊόν λογοκλοπής, από τις 781 ταυτίσεις που συνολικά εντοπίστηκαν, εμφάνισε ταύτιση 766 φράσεων (από τις 1210 συνολικές φράσεις) σε μία συγκεκριμένη αυθεντική πηγή. Το ποσοστό ομοιότητας είναι 63%, κάτι σίγουρα που θα μπορούσε να θεωρηθεί ως ανησυχητικό.

4.3. ΒΕΛΤΙΩΣΕΙΣ

Όπως ήδη αναφέρθηκε, αρκετά μέρη του προγράμματος και του αλγορίθμου που χρησιμοποιείται μπορούν να βελτιωθούν με διαφορετικές τεχνικές. Με την εφαρμογή μερικών βελτιώσεων μπορεί να παρατηρηθεί πολύ μεγάλη βελτίωση τόσο στην αποδοτικότητα εκτέλεσης (ταχύτητα, εξοικονόμηση πόρων συστήματος), όσο και στην ποιότητα εντοπισμού λογοκλοπής.

Ένα τμήμα που χρήζει βελτίωση, είναι ο τρόπος ανάλυσης του αρχικού κείμενου. Το μοντέλο που ακολουθείται, όπως αναλύθηκε, είναι ο χωρισμός προτάσεων σε λέξεις και στη συνέχεια σε φράσεις. Η αναζήτηση γίνεται ξεχωριστά για κάθε φράση που δημιουργήθηκε. Αυτό το μοντέλο είναι αρκετά γενικό. Εξυπηρετεί στον εντοπισμό μεμονωμένων μικρών τμημάτων αλλά έχοντας μεγάλο αριθμό μικρών φράσεων, οι αναζητήσεις που γίνονται είναι πάρα πολλές. Αυτό το μοντέλο θα μπορούσε να τροποποιηθεί ώστε να δημιουργούνται φράσεις με ένα τρόπο πιο κοντά στην αναγνώριση μοτίβου. Ένα παράδειγμα λειτουργίας ενός άλλου μοντέλου είναι το εξής:

Κάθε πρόταση θα χειρίζεται ξεχωριστά. Η φράση θα δημιουργείται με την συμπλήρωση τριών λέξεων με αριθμό χαρακτήρων μεγαλύτερο των τεσσάρων (4). Σε περίπτωση που μία πρόταση χωριστεί και περισσεύουν κάποιες λέξεις, αυτές θα αγνοούνται. Για παράδειγμα, παρακάτω αναλύεται η ακόλουθη φράση πρώτα με τον τρόπο που εφαρμόζεται στο πρόγραμμα και ύστερα με τη θεωρητικά βελτιωμένη μορφή:

"*Μερικά συστήματα είναι ικανά να αναπαράγουν ακόμη και τις πληροφορίες που δεν είναι κείμενο σε ένα έγγραφο, όπως εικόνες, στήλες, γραμμές, γωνίες κτλ. Το 1929 ...*" ("Οπτική Αναγνώριση Χαρακτήρων", Wikipedia, 2017) .

Το κείμενο θα χωριστεί στις οκτώ (8) φράσεις: "Μερικά συστήματα είναι", "ικανά να αναπαράγουν", "ακόμη και τις", "πληροφορίες που δεν", "είναι κείμενο σε", "ένα έγγραφο όπως", "εικόνες στήλες γραμμές", "γωνίες κτλ Το".

Χρησιμοποιώντας τον αλγόριθμο που περιγράφηκε, προκύπτουν οι ακόλουθες τέσσερις (4) φράσεις: "Μερικά συστήματα είναι", "ικανά να αναπαράγουν ακόμη", "και τις πληροφορίες που δεν είναι κείμενο", "σε ένα έγγραφο όπως εικόνες στήλες".

Με αυτό τον τρόπο, οι αναζητήσεις είναι πιο ακριβείς στα αποτελέσματα που επιστρέφουν και ο αριθμός των αναζητήσεων μειώνεται σημαντικά. Βελτιώνοντας αυτό το σημείο με μία πιο μελετημένη υλοποίηση, θα αυξηθεί η ποιότητα της αναζήτησης αλλά και η αποδοτικότητα χρήσης των πόρων του συστήματος.

Το τμήμα που χειρίζεται την εξαγωγή κειμένου από μία ιστοσελίδα και την εισαγωγή του κειμένου στο αποθετήριο χρειάζεται επίσης βελτίωση. Όταν πρόκειται να πραγματοποιηθεί εξαγωγή κειμένου από ιστοσελίδα, το πρόγραμμα κάνει χρήση των CSS tags της σελίδας ώστε να κάνει εξαγωγή του κειμένου που βρίσκει στα tags <p> και <li> (Εικόνα 3.1.3.4.1, Εικόνα 3.1.3.4.2). Τα tags <p> και <li> επιλέχθηκαν ως βασικά tags κειμένου. Αυτή η παραδοχή δεν είναι ιδιαίτερα ακριβείς. Πολλές φορές κείμενο που εντοπίζεται ως ταύτιση δεν ανήκει σε ένα από αυτά τα tags. Από την αναζήτηση στο internet το πρόγραμμα βρίσκει το κείμενο το οποίο στη συνέχεια επιχειρεί να αποθηκεύσει. Με την επεξεργασία μόνο κειμένου των <p> και <li> tags χάνεται η βασική πληροφορία η οποία τελικά δεν αποθηκεύεται στο repo. Σε μελλοντική ταύτιση, πληροφορία η οποία θα έπρεπε να βρίσκεται στο repo, δεν βρίσεκται πραγματικά και θα γίνει ξανά διαδικτυακή αναζήτηση. Η τεχνική εξαγωγής δεδομένων από ιστοσελίδα πρέπει να αναθεωρηθεί. Αυτή η βελτίωση θα ενισχύσει σημαντικά την ποιότητα λειτουργίας, ελαχιστοποιώντας – αν όχι εκμηδενίζοντας τις περιττές αναζητήσεις στο διαδίκτυο.

Ο αλγόριθμος που χρησιμοποιείται έχει σχηματιστεί γύρω από την ιδέα της 'πλαισιακής' ταύτισης. Για αποδοτικότερη λειτουργία του προγράμματος μπορούν να επανασχεδιαστούν μεμονωμένα κομμάτια ώστε να παραμετροποιηθεί το σύστημα κατάλληλα. Για παράδειγμα, αν το σύστημα επιλέγει να αποθηκεύει τις δύο (2) πρώτες σελίδες που εντοπίζουν ταύτιση, μπορεί να προκύψει καλύτερη πιθανότητα σε επόμενη αναζήτηση στο αποθετήριο να βρεθεί κάποιο όμοιο κείμενο που έχει ήδη αναγνωριστεί. Ένα άλλο αρκετά χρονοβόρο μέρος της λειτουργίας του προγράμματος είναι η περίπτωση εφαρμογής τεχνικής OCR για την εξαγωγή του κειμένου. Πρόκειται για μία αρκετά απαιτητική διαδικασία η οποία από το πρόγραμμα χρησιμοποιείται μόνο σε σπάνιες περιπτώσεις (όταν δηλαδή αποτύχουν οι προηγούμενες προσπάθειες εξαγωγής κειμένου από αρχείο PDF). Η αντικατάσταση της διαδικασίας OCR ή συγκεκριμένα αυτής της μεθόδου που εφαρμόζεται στο πρόγραμμα, είναι κάτι που μπορεί να επιταχύνει σημαντικά τη συνολική εκτέλεση.

Τέλος, πρέπει να σημειωθεί πως το κομμάτι που χειρίζεται την αναζήτηση στο διαδίκτυο με τη χρήση μηχανών αναζήτησης των Google, Bing και Yahoo, δεν αξιολογείται ως προς την απόδοση, ορθότητα εφαρμογής και ακρίβειας αποτελεσμάτων.

5. ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Liu, W., Li, X., & Huang, D. (2011). A survey on context awareness. *2011 International Conference on Computer Science and Service System (CSSS)*. doi:10.1109/csss.2011.5972040
2. Saatsakis, A., & Demestichas, P. (2009). Context Matching for Realizing Cognitive Wireless Network Segments. *Wireless Personal Communications, 55*(3), 407-440. doi:10.1007/s11277-009-9807-z
3. Sinderen, M. V., Halteren, A. V., Wegdam, M., Meeuwissen, H., & Eertink, E. (2006). Supporting context-aware mobile applications: an infrastructure approach. *IEEE Communications Magazine,44*(9), 96-104. doi:10.1109/mcom.2006.1705985
4. Nolte, K., Kaloxylos, A., Tsagkaris, K., Rosowski, T., Stamatelatos, M., Galani, A., . . . Koenig, W. (2010). The E3 architecture: enabling future cellular networks with cognitive and self-x capabilities. *International Journal of Network Management, 21*(5), 360-383. doi:10.1002/nem.762
5. Salunke, R., Badhe, D., Doke, V., & Raykar, Y. (2015). The State of the Art in Text Recognition Techniques. Ijarcce, 442-444. doi:10.17148/ijarcce.2015.42100
6. Pattern matching. (2017, May 27). Retrieved from https://en.wikipedia.org/wiki/Pattern_matching
7. Nattel, L. (n.d.). Your writing, at its best. Retrieved from https://www.grammarly.com/1
8. Check Your Essays and get the A You Deserve. (n.d.). Retrieved from http://www.articlechecker.com/
9. What can I use it for? (n.d.). Retrieved from https://www.duplichecker.com/?utm_campaign=elearningindustry.com&utm_source=%2Ftop-10-free-plagiarism-detection-tools-for-teachers&utm_medium=link
10. Viper Plagiarism Scanner. (n.d.). Retrieved from https://www.scanmyessay.com/?utm_campaign=elearningindustry.com&utm_source=%2Ftop-10-free-plagiarism-detection-tools-for-teachers&utm_medium=link

11. WCopyfind. (n.d.). Retrieved from http://plagiarism.bloomfieldmedia.com/wordpress/software/wcopyfind/

12. Data as a Service (DaaS) - we scrape websites and give you clean data. (n.d.). Retrieved from https://www.scrapehero.com/

13. Introducing JSON. (n.d.). Retrieved from http://www.json.org/

14. Οπτική Αναγνώριση Χαρακτήρων. (2017, March 02). Retrieved from https://el.wikipedia.org/wiki/%CE%9F%CF%80%CF%84%CE%B9%CE%BA%CE%AE_%CE%91%CE%BD%CE%B1%CE%B3%CE%BD%CF%8E%CF%81%CE%B9%CF%83%CE%B7_%CE%A7%CE%B1%CF%81%CE%B1%CE%BA%CF%84%CE%AE%CF%81%CF%89%CE%BD

6. ΠΑΡΑΡΤΗΜΑ

Ο πηγαίος κώδικας του προγράμματος WebCrawler είναι διαθέσιμος στο GitHub:

https://github.com/ba055482/WebCrawler