ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Αυτόματη Θεματική Κατηγοριοποίηση και Σημασιολογική Διεύρυνση Ερωτημάτων για Μηχανή Αναζήτησης με Οντολογίες

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΑΜΑΛΙΑΣ Δ. ΚΟΥΡΤΗ

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2008

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Αυτόματη Θεματική Κατηγοριοποίηση και Σημασιολογική Διεύρυνση Ερωτημάτων για Μηχανή Αναζήτησης με Οντολογίες

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

ΑΜΑΛΙΑΣ Δ. ΚΟΥΡΤΗ

Επιβλέπων : Ιωάννης Βασιλείου
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 9η Ιουλίου 2008.

.....	.....	.....
Ιωάννης Βασιλείου Καθηγητής Ε.Μ.Π.	Τίμος Σελλής Καθηγητής Ε.Μ.Π.	Νεκτάριος Κοζύρης Αναπλ. Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2008

.....

ΑΜΑΛΙΑ Δ. ΚΟΥΡΤΗ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Σκοπός της παρούσας διπλωματικής είναι η επέκταση του συστήματος GoNTogle, μίας μηχανής αναζήτησης που συνδυάζει λέξεις κλειδιά και σημασιολογία και αναπτύχθηκε στο Εργαστήριο ΣΒΓΔ.

Για να είναι χρηστικό το σύστημα αναζήτησης του GoNTogle, απαιτείται η ύπαρξη σημασιολογικών επισημειώσεων επί των εγγράφων μιας συλλογής ή τμημάτων τους, ως προς τους κόμβους μιας οντολογίας. Επειδή η χειροκίνητη δημιουργία αυτών των χαρακτηρισμών είναι έργο επίπονο, υλοποιήθηκε υποσύστημα αυτόματου σημασιολογικού χαρακτηρισμού εγγράφων με χρήση τεχνικών μηχανικής μάθησης. Το σύστημα προτείνει στο χρήστη τους πιο υποσχόμενους χαρακτηρισμούς για κάθε κείμενο, μαθαίνει από τα λάθη του και επιτυγχάνει υψηλή απόδοση.

Ένα άλλο συχνό πρόβλημα που παρατηρείται σε αναζητήσεις, είναι η επιστροφή μη επαρκούς αριθμού αποτελεσμάτων, για παράδειγμα αν το ερώτημα αναζήτησης είναι υπερβολικά περιοριστικό. Για την αντιμετώπιση αυτού του προβλήματος προτείνουμε τεχνικές επέκτασης του ερωτήματος στο σημασιολογικό άξονα, υπό το γενικό τίτλο «αναζήτηση σημασιολογικής γειτονίας». Με τις μεθόδους αυτές ο χρήστης μπορεί να εξερευνήσει αποδοτικά τον χώρο των σημασιολογικά συναφών αποτελεσμάτων σχετικά με το αρχικό του ερώτημα.

Επίσης υλοποιήθηκαν επεκτάσεις του συστήματος GoNTogle, που αφορούν στην ευρετηριοποίηση εγγράφων. Τέλος, δημιουργήθηκε οντολογία για την κατηγοριοποίηση επιστημονικών δημοσιεύσεων Πληροφορικής βασισμένη στην κατάταξη ACM. Πειράματα με πραγματικά δεδομένα με χρήση αυτής της οντολογίας αναδεικνύουν την αποδοτικότητα των μεθόδων μας.

Λέξεις Κλειδιά: Κατηγοριοποίηση, μηχανική μάθηση, k-NN, αναζήτηση εγγύτητας, οντολογία, σημασιολογικός χαρακτηρισμός

Abstract

This thesis extends the GoNTogle system, a search engine combining keywords and semantics developed at KDBSL.

For search to be effective in GoNTogle, documents or fragments thereof need to be semantically marked up wrt. concepts in an ontology. Manual creation of such annotations is an assiduous task, thus an automatic semantic annotation subsystem was developed using machine learning techniques. The system suggests the most promising concepts for every document, learns from its mistakes and achieves high performance.

Another frequent problem in the context of search is an insufficient number of results being returned, for instance in cases of an overly restrictive query. To alleviate this problem we propose techniques for extending the query along the semantic axis, under the general title of "semantic proximity search". Our methods allow for the effective exploration of results semantically affine to the initial query.

GoNTogle was also extended wrt. indexing capacities. Finally, an ontology for classifying scientific publications in Computer Science was created, based on the ACM classification system. Experiments with real data and this ontology demonstrate the effectiveness of our approach.

Keywords: Classification, machine learning, k-NN, proximity search, semantic annotation

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλους τους ανθρώπους που βοήθησαν στην εκπόνηση αυτής της διπλωματικής εργασίας.

Ευχαριστώ τον Καθηγητή κ. Ιωάννη Βασιλείου και τους συνεπιβλέποντες Θοδωρή Δαλαμάγκα και Άντζελα Δημητρίου, για την ανάθεση της διπλωματικής και για την υπομονή και την πολύτιμη βοήθειά τους. Επίσης, ευχαριστώ τον Καθηγητή κ. Τιμολέοντα Σελλή που υπήρξε για εμένα ισχυρό πρότυπο και με ενέπνευσε να ακολουθήσω τη συγκεκριμένη επιστημονική κατεύθυνση.

Ευχαριστώ πολύ την οικογένειά μου για την υποστήριξη που μου παρείχε τα χρόνια των σπουδών μου.

Τέλος, θα ήθελα να ευχαριστήσω θερμά τον αγαπητό συνάδελφο και φίλο Αλβέρτο Άντζελ, χωρίς την αμέριστη συμπαράσταση του οποίου δε θα είχε έρθει εις πέρας η διπλωματική αυτή.

Πίνακας περιεχομένων

1

Εισαγωγή

Η πρόοδος της επιστήμης των υπολογιστών και της πληροφορικής και η αλματώδης ανάπτυξη του διαδικτύου έδωσαν τη δυνατότητα αποθήκευσης πολύ μεγάλου όγκου δεδομένων και πληροφοριών. Η «πληροφοριακή συμφόρηση» που δημιουργήθηκε, γέννησε την ανάγκη εύρεσης μηχανισμών ικανών για αποδοτική αξιοποίηση της πληθώρας των δεδομένων αυτών, με γρήγορο εντοπισμό και αποδοτική σταχυολόγηση των ζητούμενων πληροφοριών. Αποτέλεσμα ήταν η ανάπτυξη του «Σημασιολογικού Ιστού» (Semantic Web).

Αν θεωρήσουμε ότι ο Σημασιολογικός Ιστός αποτελεί το «μακρόκοσμο», θα μπορούσε να υποτεθεί ότι ο σκληρός δίσκος ενός οικιακού χρήστη αποτελεί ένα «μικρόκοσμο», στον οποίο εντοπίζονται τα προβλήματα και οι ανάγκες του πραγματικού κόσμου. Η αύξηση της χωρητικότητας, δίνει τη δυνατότητα αποθήκευσης μεγάλου πλήθους αρχείων, η αναζήτηση των οποίων μετατρέπεται σε επίπονη διαδικασία, μιας και θεματικά παρόμοια αρχεία μπορεί να έχουν αποθηκευτεί σε διαφορετικές περιοχές του δίσκου. Η αναζήτηση δε μόνο με λέξεις κλειδιά, θα αποφέρει πιθανώς και αποτελέσματα άσχετα με τις προσδοκίες του χρήστη. Η ανάγκη αυτή οδήγησε στη δημιουργία του συστήματος GoNTogle 1.0, μία μηχανή αναζήτησης εγγράφων δηλαδή, που συνδυάζει την κλασσική αναζήτηση με λέξεις κλειδιά, με τη σημασιολογική αναζήτηση στις έννοιες-κατηγορίες μίας οντολογίας.

Αν όμως το μέγεθος της συλλογής εγγράφων είναι πολύ μεγάλο, ο σημασιολογικός χαρακτηρισμός τους καταλήγει να είναι μια επίπονη διαδικασία για το χρήστη. Η αυτόματη ή υποβοηθούμενη οργάνωσή των αρχείων κατά θεματικές περιοχές, θα διευκόλυνε πολύ το

χρήστη. Αυτή την ανάγκη έρχεται να καλύψει το GoNTogle 2.0 με ένα νέο υποσύστημα αυτόματου σημασιολογικού χαρακτηρισμού.

Επίσης, παρατηρείται συχνά σε αναζητήσεις, ότι επιστρέφεται μικρός αριθμός αποτελεσμάτων όταν πρόκειται πχ για υπερβολικά περιοριστικό ερώτημα. Το GoNTogle 2.0 υλοποιεί δύο μεθόδους εμπλουτισμού των αποτελεσμάτων σημασιολογικής αναζήτησης, στα πλαίσια της αναζήτησης σημασιολογικής γειτονίας.

1.1 Αντικείμενο της διπλωματικής

Η εργασία αυτή έχει στόχο να επεκτείνει το υπάρχον σύστημα αναζήτησης GoNTogle 1.0 προσθέτοντάς του νέες λειτουργικότητες. Ειδικότερα, η συνεισφορά της παρουσιάζεται ακολούθως.

Η διπλωματική πρόσθεσε στο σύστημα GoNTogle 1.0 νέες λειτουργίες, με την πλέον βασική να είναι η αυτόματη κατηγοριοποίηση εγγράφων σε κατηγορίες μιας οντολογίας, δηλαδή ο αυτόματος σημασιολογικός χαρακτηρισμός.. Το σύστημα, γνωστό πλέον ως GoNTogle 2.0 χρησιμοποιεί τεχνικές μηχανικής μάθησης για να «μαθαίνει» από τα λάθη του και να προσαρμόζεται στις συνήθειες του χρήστη.

Επιπλέον, στο GoNTogle 2.0, υλοποιήθηκαν τεχνικές διεύρυνσης των ερωτημάτων στο σημασιολογικό άξονα, υπό το γενικό τίτλο «αναζήτηση σημασιολογικής γειτονίας». Με τις τεχνικές αυτές ο χρήστης μπορεί να εξερευνήσει αποδοτικά τον χώρο των σημασιολογικά συναφών αποτελεσμάτων σχετικά με το αρχικό του ερώτημα. Έτσι, αντιμετωπίζουμε ένα συχνό πρόβλημα που παρατηρείται στις αναζητήσεις, το οποίο είναι η επιστροφή μη επαρκούς αριθμού αποτελεσμάτων, για παράδειγμα αν το ερώτημα αναζήτησης είναι υπερβολικά περιοριστικό.

Για την αξιολόγηση των μεθόδων που υλοποιήθηκαν, δημιουργήθηκε μια πλήρης οντολογία κατάταξης επιστημονικών δημοσιεύσεων Πληροφορικής, βασισμένη στο σύστημα κατάταξης της ACM. Πειράματα με χρήση πραγματικών δεδομένων (πρακτικών συνεδρίων και αυτής της οντολογίας) αναδεικνύουν την αποτελεσματικότητα των μεθόδων μας.

Τέλος, υλοποιήθηκαν επιπλέον επεκτάσεις του συστήματος GoNTogle 1.0, κυρίως στο υποσύστημα ευρετηριοποίησης εγγράφων.

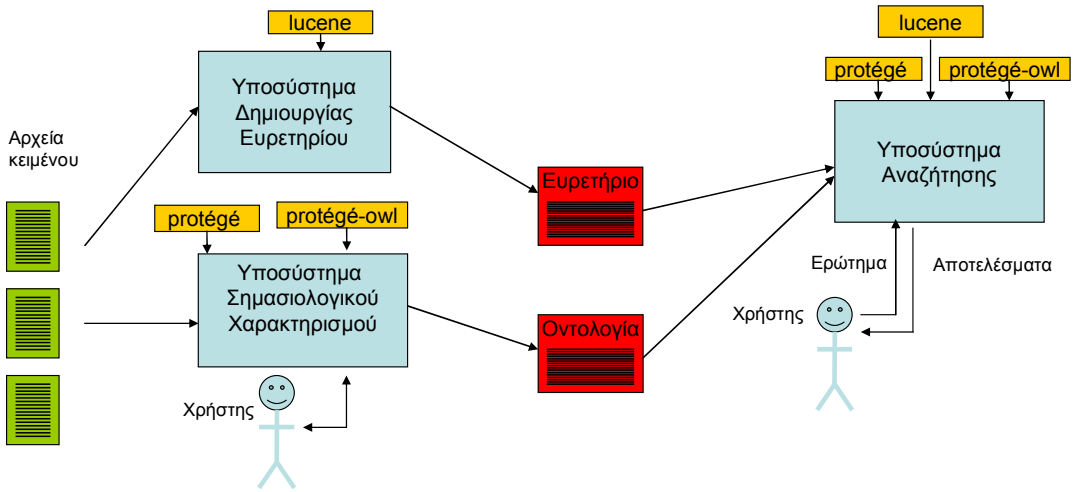
1.1.1 Περιγραφή του συστήματος GoNTogle 1.0

Το GoNTogle 1.0 είναι μια μηχανή αναζήτησης, που αναπτύχθηκε στο Εργαστήριο Συστημάτων Βάσεων Γνώσεων και Δεδομένων, για την παροχή προχωρημένων δυνατοτήτων αναζήτησης, με συνδυασμό λέξεων-κλειδιών και σημασιολογίας. Με την χρήση του

συστήματος αυτού, μπορούν να απαντηθούν ερωτήματα όπως «Ποια έγγραφα αναφέρονται σε ‘αποθήκες’ και ‘καθαρισμό’, και θεματικά εμπίπτουν στην περιοχή των Βάσεων Δεδομένων;», που είναι δύσκολο να απαντηθούν με κλασσικά παραδείγματα αναζήτησης.

Για την επίτευξη του στόχου αυτού, το GoNTogle 1.0 χρησιμοποιεί οντολογίες, δηλαδή, ιεραρχικές κατηγοριοποιήσεις εννοιών μιας γνωστικής περιοχής. Ο χρήστης έχει την δυνατότητα να αποδώσει θεματική κατηγορία σε κάθε έγγραφο, αντιστοιχίζοντάς το σε έναν ή περισσότερους κόμβους μιας οντολογίας. Κατόπιν, είναι δυνατή η αναζήτηση του εγγράφου αυτού και με θεματικά κριτήρια.

Το αρχιτεκτονικό σχέδιο του συστήματος GoNTogle 1.0 παρουσιάζεται στο παρακάτω σχήμα:



Σχήμα 1 Αρχιτεκτονικό σχέδιο GoNTogle 1.0

1.1.2 GoNTogle 2.0 - Αυτόματη Θεματική Κατηγοριοποίηση

Η χειρωνακτική κατάταξη των εγγράφων σε θεματικές κατηγορίες, που απαιτείται για την μετέπειτα αναζήτησή τους από το GoNTogle, είναι έργο επίπονο και χρονοβόρο. Για να χαρακτηρίσει θεματικά μια συλλογή εγγράφων, ο χρήστης πρέπει, για κάθε έγγραφο, να εξετάσει αφενός το περιεχόμενό του και αφετέρου την οντολογία, προκειμένου να επιλέξει τον κόμβο της οντολογίας που χαρακτηρίζει καλύτερα το θέμα του εγγράφου. Η πολυπλοκότητα της διαδικασίας αυτής αυξάνεται καθώς μεγαλώνει το μέγεθος της οντολογίας ή/και της συλλογής εγγράφων.

Για την αντιμετώπιση του προβλήματος αυτού, αναπτύχθηκε ένα σύστημα Αυτόματης Θεματικής Κατηγοριοποίησης εγγράφων, με βάση το περιεχόμενό τους. Το σύστημα

χρησιμοποιεί τεχνικές Μηχανικής Μάθησης, και προτείνει στο χρήστη τους καταλληλότερους κόμβους μιας οντολογίας για κάθε έγγραφο ή τμήμα αυτού.

Η αρχή που χρησιμοποιείται είναι η εξής (k-Nearest Neighbour): το λεκτικό περιεχόμενο του εγγράφου συγκρίνεται με αυτό των εγγράφων ενός επισημειωμένου συνόλου εκπαίδευσης, και το έγγραφο τοποθετείται στην κατηγορία όπου ανήκουν τα πιο όμοια έγγραφα με αυτό. Συγκεκριμένα, υπολογίζεται η ομοιότητα του προς κατηγοριοποίηση εγγράφου με τα έγγραφα εκπαίδευσης, και στη συνέχεια τα k πιο όμοια έγγραφα "ψηφίζουν" για την κατηγορία κατάταξης αυτού. Η ψηφοφορία μπορεί να γίνει είτε με ισοβαρείς ψήφους, είτε με ψήφους σταθμισμένες ανάλογα με την ομοιότητα των εγγράφων. Οι δύο αυτές παραλλαγές υλοποιήθηκαν και αξιολογήθηκαν πειραματικά.

Ως δεδομένα εκπαίδευσης, το σύστημα χρησιμοποιεί το σύνολο των ήδη κατηγοριοποιημένων εγγράφων. Έτσι εξασφαλίζεται η καλή λειτουργία του συστήματος, ανεξαρτήτως της οντολογίας που χρησιμοποιείται, το δε σύστημα βελτιώνεται και μαθαίνει από τα λάθη του και τις συνήθειες του χρήστη.

1.1.3 GoNTogle 2.0 - Σημασιολογική Διεύρυνση Ερωτημάτων

Συχνά μια αναζήτηση δεν επιστρέφει επαρκή αριθμό αποτελεσμάτων στο χρήστη, για παράδειγμα, αν το ερώτημα αναζήτησης είναι υπερβολικά περιοριστικό (πχ αναζήτηση εγγράφων σχετικών με κάποιο 'συνέδριο' στη 'Λευκάδα', στην θεματική περιοχή Επιστήμη). Για την αντιμετώπιση αυτού του προβλήματος προτείνονται τεχνικές διεύρυνσης του ερωτήματος στο σημασιολογικό άξονα, υπό το γενικό τίτλο «αναζήτηση σημασιολογικής γειτονίας».

Οι μέθοδοι αυτές επιτρέπουν στο χρήστη να εξερευνήσει το χώρο των σημασιολογικά συναφών αποτελεσμάτων αναζήτησης, διευρύνοντας τους σημασιολογικούς περιορισμούς του ερωτήματος. Στο προηγούμενο παράδειγμα, ένα έγγραφο για ένα 'συνέδριο' στη 'Λευκάδα', που ανήκει όμως στην θεματική υποπεριοχή Μουσική Πληροφορική (Music Computing), μπορεί να ικανοποιεί την πρόθεση του χρήστη.

Συγκεκριμένα, προτείνονται δύο μέθοδοι σημασιολογικής διεύρυνσης ερωτημάτων, την "Αναζήτηση Επόμενης Γενιάς" (ΑΕΓ), που επιστρέφει έγγραφα από κατηγορίες που προοδευτικά εξειδικεύουν τις κατηγορίες του ερωτήματος, και την "Αναζήτηση Στενής Γειτονίας" (ΑΣΓ), που επιστρέφει έγγραφα από κατηγορίες σημασιολογικά συσχετισμένες με αυτές του ερωτήματος, βαθμολογημένα ανάλογα με την σημασιολογική αυτή συνάφεια.

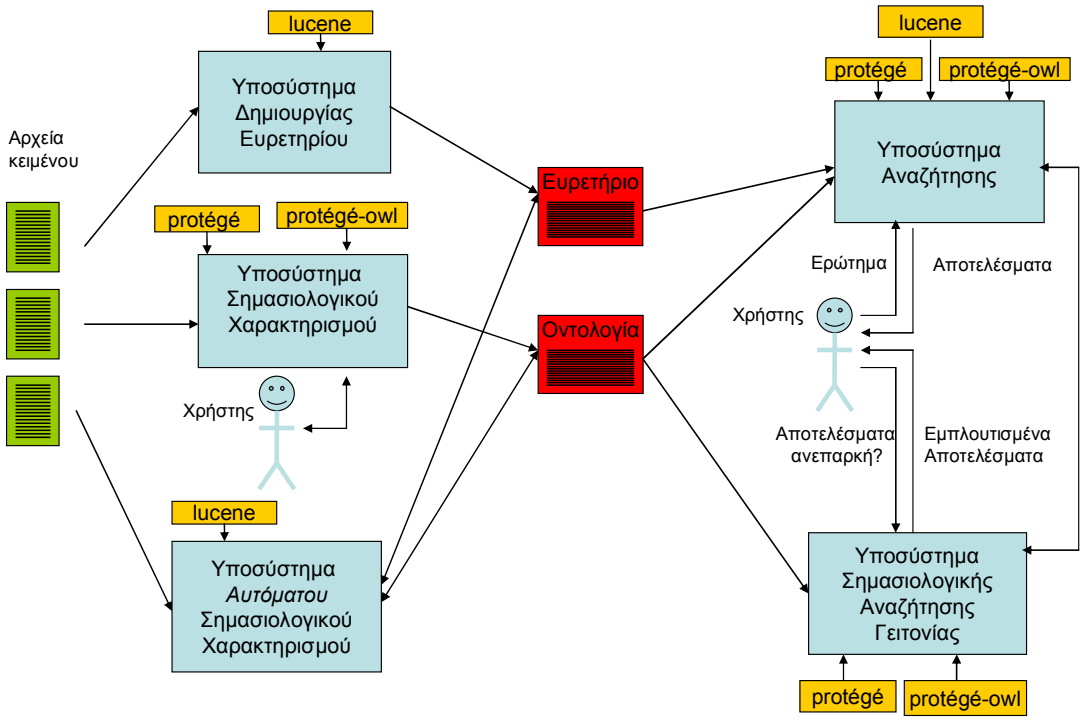
Στο προηγούμενο παράδειγμα, έστω ότι η αναζήτηση εγγράφων 'συνέδριο' στη 'Λευκάδα', στην θεματική περιοχή Επιστήμη δεν επιστρέφει αρκετά αποτελέσματα στο χρήστη. Με την ΑΕΓ, θα επιστραφούν στο χρήστη σε πρώτη φάση έγγραφα για 'συνέδριο'

στη ‘Λευκάδα’ στις θεματικές περιοχές π.χ. Ανθρωπιστικές Επιστήμες, Θετικές Επιστήμες, κλπ, ενώ αν ο χρήστης ζητήσει περισσότερα αποτελέσματα, θα του επιστραφούν έγγραφα για ‘συνέδριο’ στη ‘Λευκάδα’ στις θεματικές περιοχές π.χ. Πληροφορική, Φυσική, κλπ.. Αν ζητήσει περαιτέρω αποτελέσματα, από τις θεματικές περιοχές Μουσική Πληροφορική, Πυρηνική Φυσική, κλπ, κ.ο.κ.. Με την ΑΣΓ, θα επιστραφούν στο χρήστη έγγραφα για ‘συνέδριο’ στη ‘Λευκάδα’ στις θεματικές περιοχές Ανθρωπιστικές Επιστήμες, Θετικές Επιστήμες, κλπ, και Πληροφορική, Φυσική, κλπ, με βαθμολογίες ανάλογες της σχετικότητας της θεματικής τους περιοχής, με αυτήν που αρχικά ζητήθηκε (Επιστήμη).

Με τις μεθόδους αυτές ο χρήστης μπορεί να εξερευνήσει αποδοτικά τον χώρο των σημασιολογικά συναφών αποτελεσμάτων σχετικά με το αρχικό του ερώτημα.

1.1.4 Αρχιτεκτονικό Σχέδιο GoNTogle 2.0

Το συνολικό σύστημα GoNTogle 2.0, έτσι όπως διαμορφώνεται με την προσθήκη των παραπάνω λειτουργιών φαίνεται στο παρακάτω σχήμα:



Σχήμα 2 Αρχιτεκτονικό σχέδιο GoNTogle 2.0

1.1.5 Πειραματική Αξιολόγηση

Για την αξιολόγηση των μεθόδων μας, δημιουργήθηκε μια πλήρης οντολογία κατάταξης επιστημονικών δημοσιεύσεων Πληροφορικής, βασισμένη στο σύστημα κατάταξης

της ACM[1]. Εκτελέστηκαν πειράματα με χρήση πραγματικών δεδομένων (πρακτικών συνεδρίων και αυτής της οντολογίας). Σε αυτά, εξετάστηκε ποσοτικά η ικανότητα των αλγορίθμων αυτόματης θεματικής κατηγοριοποίησης, στον εντοπισμό της σωστής κατηγορίας μιας δημοσίευσης, με βάση το κείμενό της. Επίσης, αξιολογήθηκαν ποιοτικά οι μέθοδοι σημασιολογικής διεύρυνσης ερωτημάτων.

Τα πειράματά αυτά αναδεικνύουν την αποτελεσματικότητα των μεθόδων μας.

1.2 Οργάνωση του τόμου

Η οργάνωση του υπόλοιπου τόμου έχει ως εξής:

Στο κεφάλαιο 2 γίνεται μια μικρή περιγραφή του συστήματος GoNTogle 1.0.

Το κεφάλαιο 3 αφορά τον αυτόματο σημασιολογικό χαρακτηρισμό εγγράφων. Αρχικά αναφέρονται σχετικές εργασίες και γίνεται μια εισαγωγή στο θεωρητικό υπόβαθρο που χρησιμοποιήθηκε. Στη συνέχεια, ακολουθεί η περιγραφή της λειτουργικότητας του συστήματος "Auto Annotation", καθώς και στοιχεία του κώδικα.

Το κεφάλαιο 4 αφορά την αναζήτηση σημασιολογικής γειτονίας. Περιγράφει τις δύο μεθόδους αναζήτησης που υλοποιήθηκαν «Αναζήτηση Επόμενης Γενιάς» και «Αναζήτηση Στενής Γειτονίας», το θεωρητικό υπόβαθρο που τις ενέπνευσε και στοιχεία του κώδικά τους.

Στο κεφάλαιο 5 γίνεται ο πειραματικός έλεγχος των υποσυστημάτων που υλοποιήθηκαν, ενώ στο κεφάλαιο 6 αναφέρονται οι τροποποιήσεις και οι προσθήκες που αφορούν το αρχικό σύστημα GoNTogle 1.0.

Τέλος, στο κεφάλαιο 7 παρουσιάζεται συνοπτικά η εργασία και γίνεται παράθεση των συμπερασμάτων στα οποία καταλήγει. Επίσης αναφέρονται μελλοντικές επεκτάσεις της.

1 Το σύστημα κατάταξης επιστημονικών δημοσιεύσεων Πληροφορικής CCS (Computing Classification System) της ACM (Association for Computing Machinery).

2

GoNTogle 1.0

2.1 Περιγραφή

Το GoNTogle 1.0 [Για06] είναι ένα σύστημα αναζήτησης εγγράφων στο σκληρό δίσκο υπολογιστών το οποίο συνδυάζει την κλασσική αναζήτηση με λέξεις κλειδιά, με σημασιολογική αναζήτηση στις έννοιες-κατηγορίες μίας οντολογίας.

Δίνει τη δυνατότητα στο χρήστη να ευρετηριοποιεί έγγραφα του σκληρού δίσκου και βάσει αυτού του ευρετηρίου, να εκτελεί αναζήτηση εγγράφων με χρήση λέξεων-φράσεων κλειδιών. Επιπρόσθετα, υπάρχει η δυνατότητα σημασιολογικού χαρακτηρισμού τμήματος ή ολόκληρου του κειμένου εγγράφων, χρησιμοποιώντας τους κόμβους-κατηγορίες μίας οντολογίας που θα έχει φορτώσει στο σύστημα, δημιουργώντας στιγμιότυπα των κλάσεών της. Για τη σημασιολογική αναζήτηση ο χρήστης περιηγείται στις κλάσεις της οντολογίας και επιλέγει στιγμιότυπα - χαρακτηρισμένα έγγραφα, προς εμφάνιση. Οι οντολογίες που χρησιμοποιούνται μπορούν να τροποποιηθούν από το χρήστη, προσθαφαιρώντας κλάσεις, στιγμιότυπα και ιδιότητες.

Επιπλέον, ο χρήστης μπορεί να εκτελέσει συνδυαστική αναζήτηση, δηλαδή ταυτόχρονη αναζήτηση με λέξεις κλειδιά και περιήγηση στην οντολογία, αναζήτηση στα αποτελέσματα προηγούμενης αναζήτησης και αναζήτηση με σύνθετα επερωτήματα στην οντολογία.

Η παρουσίαση των αποτελεσμάτων περιλαμβάνει το όνομα των εγγράφων, τους κόμβους της οντολογίας που συμμετέχουν στην αναζήτηση και με τους οποίους έχουν χαρακτηριστεί τα

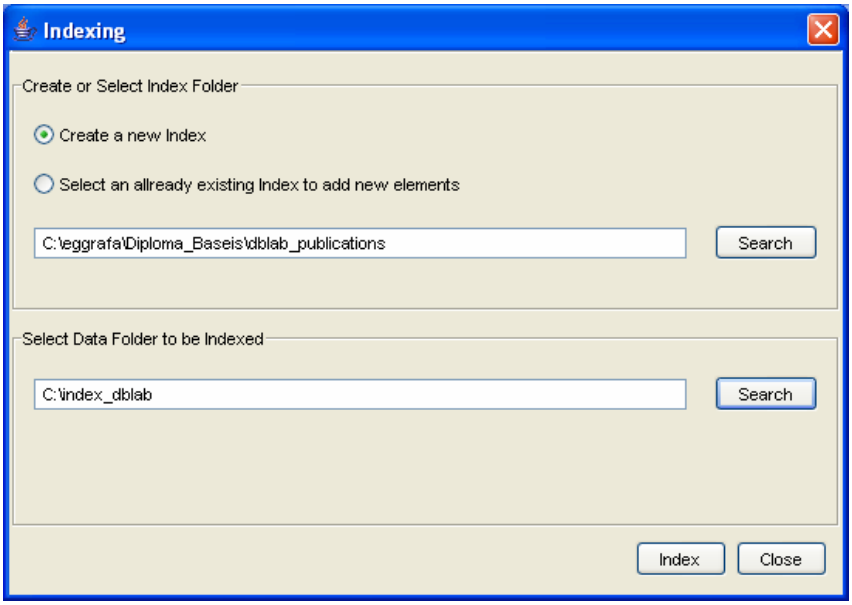
έγγραφα, καθώς και το σκορ που τα κατατάσσει ανάλογα με το κατά πόσο προσεγγίζουν τα κριτήρια αναζήτησης. Ο χρήστης μπορεί να αναγνώσει τα έγγραφα των αποτελεσμάτων, να δει τη σημασιολογική πληροφορία που μεταφέρουν, αλλά και να εντοπίσει κείμενα παρόμοιου σημασιολογικού χαρακτηρισμού.

2.2 Λειτουργικότητα

Το σύστημα χωρίζεται σε τρία υποσυστήματα: Δημιουργίας ευρετηρίου, Σημασιολογικού χαρακτηρισμού και Αναζήτησης.

2.2.1 Δημιουργία ευρετηρίου

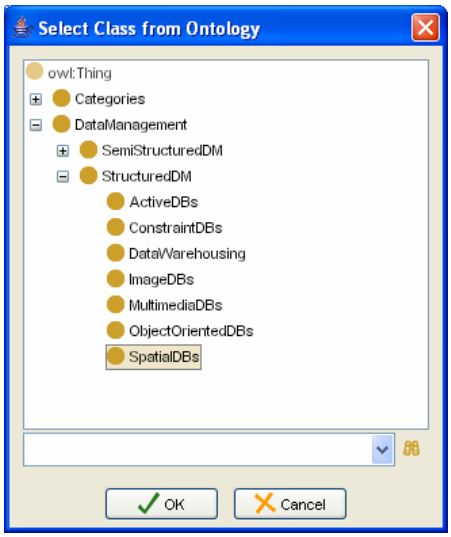
Μέσω του παρακάτω πλαισίου είναι δυνατή η δημιουργία (ή τροποποίηση με προσθήκη νέων αρχείων) ευρετηρίου πάνω στο οποίο θα εκτελείται αναζήτηση με λέξεις κλειδιά. Οι τύποι εγγράφων που είναι μπορεί να ευρετηριοποιήσει το σύστημα είναι txt, rtf, doc, pdf, xls, html.



Σχήμα 3 Πλαίσιο δημιουργίας ευρετηρίου

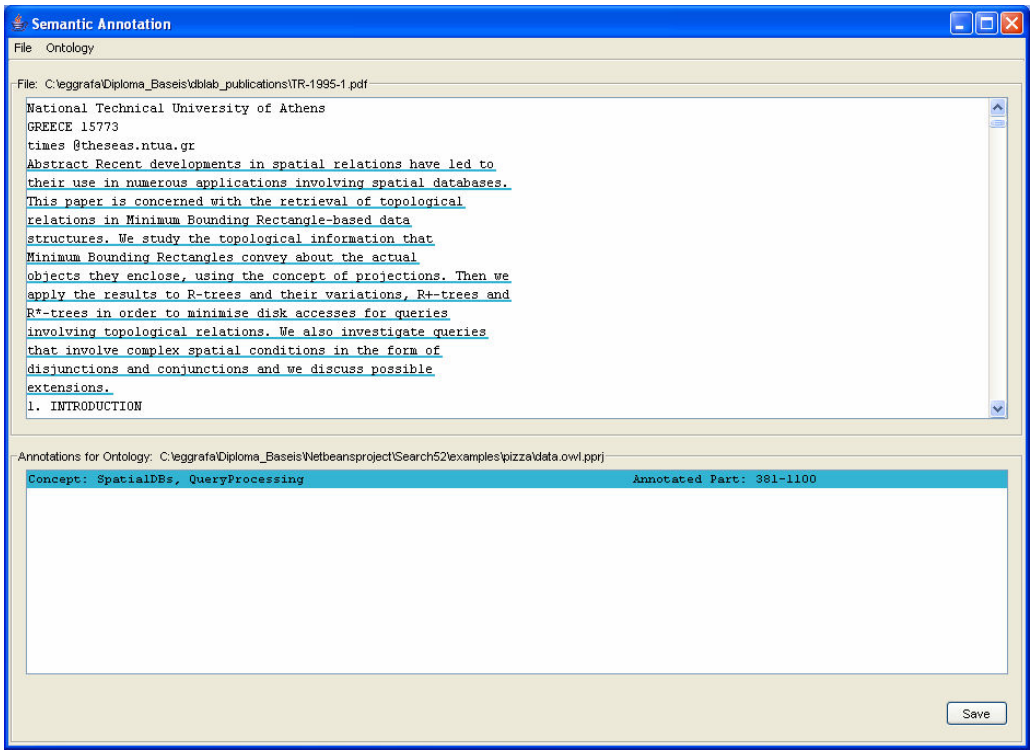
2.2.2 Σημασιολογικός χαρακτηρισμός

Ο σημασιολογικός χαρακτηρισμός γίνεται από το πλαίσιο “Semantic Annotation”, αφού φορτωθεί η επιθυμητή οντολογία τύπου .owl και αφού ανοιχθεί το προς χαρακτηρισμό έγγραφο. Για χαρακτηρισμό τμήματος του κειμένου επιλέγεται το επιθυμητό κομμάτι και με δεξί κλικ εμφανίζεται το κουμπί επιλογής της δυνατότητας. Εμφανίζεται ένας διάλογος που περιέχει την ιεραρχία της οντολογίας, απ’ όπου ο χρήστης επιλέγει κλάσεις της για να χαρακτηριστεί το κομμάτι του κειμένου.



Σχήμα 4 Επιλογή κλάσης για σημασιολογικό χαρακτηρισμό

Το αποτέλεσμα του σημασιολογικού χαρακτηρισμού φαίνεται στο παρακάτω σχήμα.



Σχήμα 5 Αποτελέσματα σημασιολογικού χαρακτηρισμού τμήματος εγγράφου.

Αν, αντί για κομμάτι του κειμένου, θέλουμε να χαρακτηρίσουμε ολόκληρο το έγγραφο, τότε επιλέγουμε File->Annotate και ακολουθούμε την ίδια διαδικασία. Ο χρήστης μπορεί να επιλέξει το στιγμιότυπο που δημιουργήθηκε και να το επεξεργαστεί. Μπορεί επίσης να επεξεργαστεί την οντολογία προσθαφαιρώντας κλάσεις, ιδιότητες και στιγμιότυπα. Οι αλλαγές που έγιναν στην οντολογία αποθηκεύονται με το πάτημα του κουμπιού “save”.

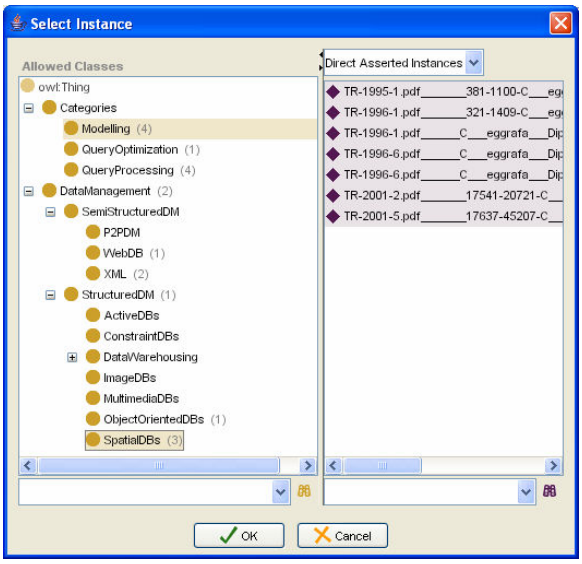
2.2.3 Αναζήτηση

2.2.3.1 Αναζήτηση με λέξεις κλειδιά

Για την απλή αναζήτηση με λέξεις κλειδιά, ο χρήστης εισάγοντας το ερώτημά του, το οποίο μπορεί να περιέχει πολύπλοκους τελεστές, πχ "title:TR* AND –spatial" (δηλαδή αρχεία με τίτλο που ξεκινά από "TR" και δεν περιέχουν τη λέξη "spatial").

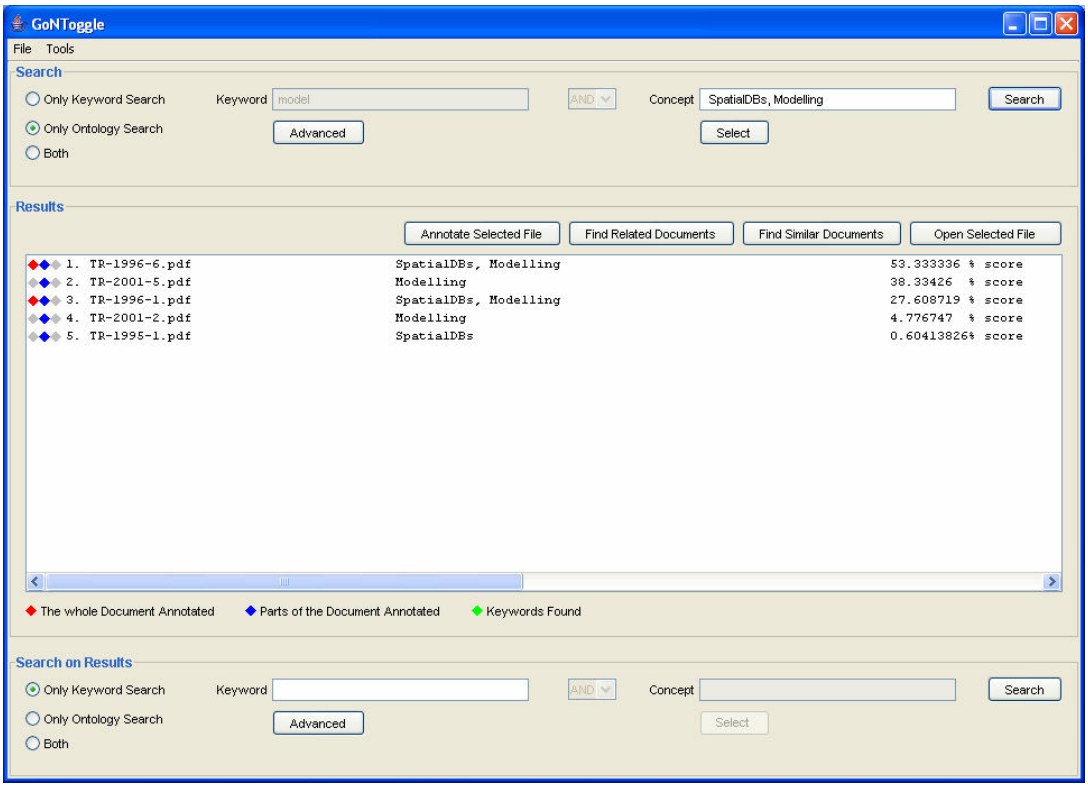
2.2.3.2 Σημασιολογική αναζήτηση

Ο χρήστης φορτώνει την επιθυμητή οντολογία και με το κουμπί "Select" εμφανίζεται πλαίσιο με την ιεραρχία της οντολογίας, απ' όπου επιλέγονται οι κόμβοι αναζήτησης. Στα δεξιά του παραθύρου εμφανίζονται τα στιγμιότυπα των εκάστοτε επιλεγμένων κλάσεων (οι οποίες φαίνονται αριστερά).



Σχήμα 6 Διάλογος σημασιολογικής αναζήτησης

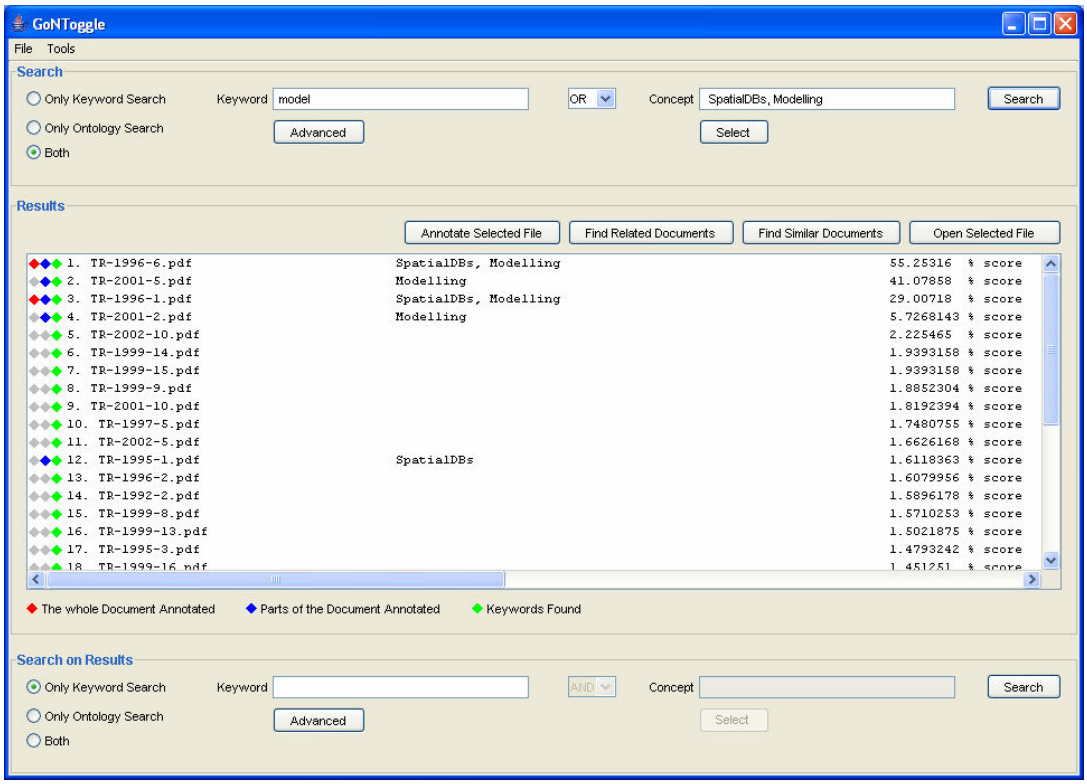
Στα αποτελέσματα, ο κόκκινος ρόμβος συμβολίζει ότι το συγκεκριμένο έγγραφο, για κάποια από τις κλάσεις, έχει χαρακτηριστεί στο σύνολό του, ενώ ο μπλε, ότι έχουν χαρακτηριστεί τμήματά του. Δίπλα από το όνομα του κάθε εγγράφου εμφανίζονται και οι κλάσεις της οντολογίας που του αντιστοιχούν για τη συγκεκριμένη αναζήτηση.



Σχήμα 7 Αποτελέσματα σημασιολογικής αναζήτησης

2.2.3.3 Συνδυαστική αναζήτηση

Επιπρόσθετα, δίνεται η δυνατότητα συνδυαστικής αναζήτησης (με λέξεις κλειδιά και οντολογία), η οποία προσφέρει δύο επιλογές: Με AND προκύπτει η τομή των αποτελεσμάτων των δύο αναζητήσεων, ενώ με OR προκύπτει η ένωσή τους. Στη συνδυαστική αναζήτηση, το τελικό σκορ προκύπτει, συνυπολογίζοντας το σκορ που θα προέκυπτε αν η αναζήτηση γινόταν μόνο με λέξεις κλειδιά, με το σκορ που θα προέκυπτε μόνο από τη σημασιολογική αναζήτηση.



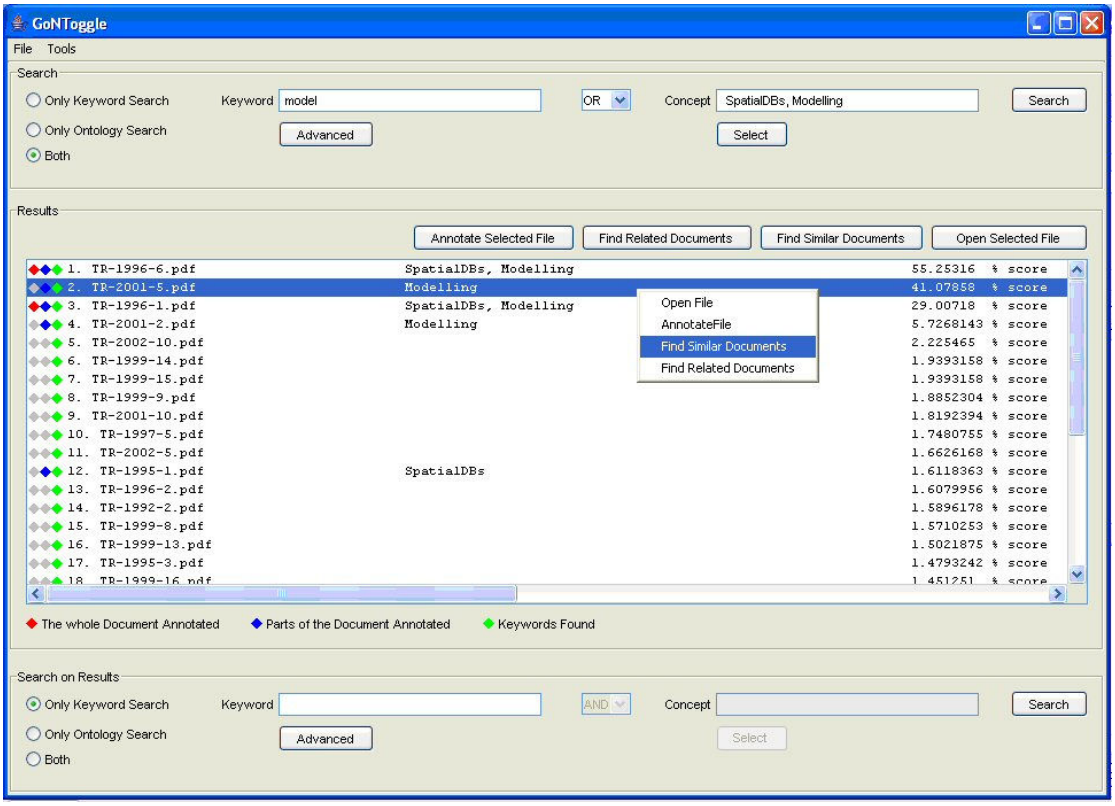
Σχήμα 8 Αποτελέσματα συνδυαστικής αναζήτησης με OR

Ο πράσινος ρόμβος στα αποτελέσματα πληροφορεί για το ποια έγγραφα περιέχουν τις λέξεις κλειδιά της αναζήτησης.

Από τα αποτελέσματα μιας αναζήτησης είναι δυνατό το άνοιγμα ενός εγγράφου, οπότε εμφανίζεται ένα νέο πλαίσιο με το κείμενό του και υπογραμμισμένα τα σημασιολογικά κομμάτια του με διαφορετικά χρώματα, ανάλογα με την κλάση χαρακτηρισμού. Επιπλέον, κάτω από το κείμενο υπάρχει μία λίστα με στιγμιότυπα που αντιστοιχούν στα χαρακτηρισμένα κομμάτια.

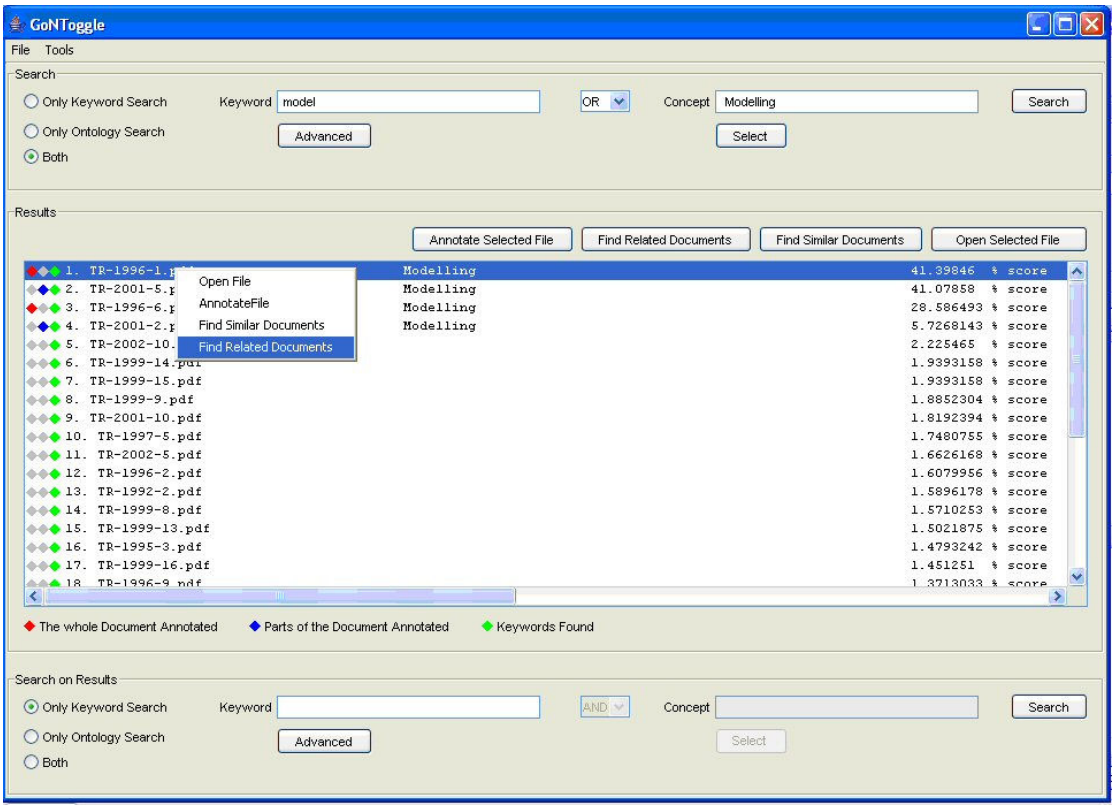
Επιπροσθέτως, είναι δυνατή η επιλογή ενός στιγμιοτύπου και πατώντας "Find Documents Related by Properties", να εξεταστεί ποιες από τις ιδιότητές το συσχετίζουν με άλλα στιγμιότυπα. Έτσι, επιλέγοντας την ιδιότητα isRelatedTo και στη συνέχεια το στιγμιότυπο που αυτή περιέχει, εμφανίζεται στο πλαίσιο το κείμενο ενός άλλου εγγράφου, το οποίο σχετίζεται σημασιολογικά με το προηγούμενο, με τη συγκεκριμένη ιδιότητα.

Μία επιπλέον δυνατότητα σημασιολογικής αναζήτησης προσφέρεται με την επιλογή "Find Similar Documents". Με αυτήν την επιλογή, εμφανίζεται ένας διάλογος σημασιολογικής αναζήτησης, ο οποίος όμως, αντί να περιέχει όλη την ιεραρχία των κλάσεων της οντολογίας, περιέχει μόνο τις κλάσεις της προηγούμενης αναζήτησης οι οποίες χαρακτηρίζουν το επιλεγμένο έγγραφο, πάνω στο οποίο εκτελείται αυτή η λειτουργία.



Σχήμα 9 Σημασιολογική αναζήτηση παρόμοιων εγγράφων με το επιλεγμένο κείμενο

Με την επιλογή “Find Related Documents” εμφανίζεται ένας διάλογος σημασιολογικής αναζήτησης, ο οποίος όμως, αντί να περιέχει όλη την ιεραρχία των κλάσεων της οντολογίας, περιέχει μόνο τις κλάσεις οι οποίες χαρακτηρίζουν οποιοδήποτε στιγμιότυπο του επιλεγμένου εγγράφου μέσα στην οντολογία. Έτσι, ο χρήστης μπορεί να επιλέξει στιγμιότυπα μόνο από αυτές τις κλάσεις.



Σχήμα 10 Σημασιολογική αναζήτηση σχετικών εγγράφων με το επιλεγμένο κείμενο

Τέλος, ο χρήστης μπορεί να εκτελεστεί αναζήτηση πάνω στα αποτελέσματα προηγούμενης αναζήτησης.

3

Αυτόματος σημασιολογικός χαρακτηρισμός

3.1 Περιγραφή

Το GoNTogle 1.0 είναι ένα σύστημα που καλύπτει ανάγκες αναζήτησης του χρήστη δίνοντας τη δυνατότητα σημασιολογικής αναζήτησης. Για το σκοπό αυτό απαιτείται από το χρήστη να χαρακτηρίσει σημασιολογικά τα έγγραφά του. Η διαδικασία αυτή είναι χρονοβόρα, ειδικά όταν πρόκειται για συλλογές με μεγάλο πλήθος εγγράφων.

Για να αντιμετωπιστεί αυτό το πρόβλημα, προσθέσαμε στο GoNTogle τη δυνατότητα αυτόματης σημασιολογικής επισήμανσης των εγγράφων. Το σύστημα προτείνει στο χρήστη αυτόματα τους καταλληλότερους κόμβους-κατηγορίες της οντολογίας, για το χαρακτηρισμό ενός εγγράφου ή τμήματός του. Με τον τρόπο αυτό διευκολύνεται η πραγματοποίηση του σημασιολογικού χαρακτηρισμού, επιτρέποντας στο χρήστη να επισημάνει σημασιολογικά μεγάλο όγκο εγγράφων του.

3.2 Σχετικές εργασίες

Όσον αφορά το αυτόματο σύστημα σημασιολογικού χαρακτηρισμού, έχουν προταθεί αρκετές εργασίες επάνω σε παρεμφερή προβλήματα.

Στο [KPO+04] προτείνεται το σύστημα KIM, το οποίο συνδυάζει αναφορές από τη θεμελιώδη οντολογία (upper level ontology) KIMO που χρησιμοποιείται για την κατηγοριοποίηση και το συσχετισμό οντοτήτων (named entities), τη βάση γνώσης

(knowledge base) περιγραφής οντοτήτων και συνήθη αναζήτηση με λέξεις κλειδιά, για την παροχή προχωρημένων δυνατοτήτων αναζήτησης. Το σύστημα αυτό πραγματοποιεί σημασιολογική επισήμανση οντοτήτων που περιέχονται σε κείμενο (Named Entity Recognigtion - NER), χωρίς να προσδίδει σημασιολογικό χαρακτηρισμό του κειμένου ή τμημάτων του. Δηλαδή, δεν χαρακτηρίζει ένα κείμενο βάσει του εννοιολογικού του περιεχομένου. Για παράδειγμα, ένα κείμενο που αναφέρεται στην προσωπική ζωή ποδοσφαιριστών (οπότε θα χαρακτηριζόταν σημασιολογικά με την κατηγορία πχ «Κοσμικά»), στο KIMO μπορεί να αναζητηθεί σημασιολογικά μόνο μέσω των «Ποδοσφαιριστών». Πολλά άλλα συστήματα κάνουν χρήση NER όπως πχ τα [CMB+02] και [Reu08].

Στο [CFV07] προτείνεται ένα σύστημα αναζήτησης που πραγματοποιεί αναζήτηση συνδυάζοντας οντολογίες και αναζήτηση με λέξεις κλειδιά. Επίσης προτείνει τεχνικές ημιαυτόματου σημασιολογικού χαρακτηρισμού κειμένων. Εντούτοις, οι τεχνικές αυτές βασίζονται σε κάποιες ευριστικές και επεξεργασία φυσικής γλώσσας. Μη χρησιμοποιώντας τεχνικές μηχανικής μάθησης, το σύστημα δεν έχει τη δυνατότητα βελτίωσης της απόδοσής του, καθώς εμπλουτίζεται η επισημειωμένη συλλογή εγγράφων.

Το [SDS06] χρησιμοποιεί k-NN Classifiers για την κατάταξη εγγράφων σε κόμβους μιας οντολογίας, όπως και η παρούσα εργασία – εντούτοις ο στόχος στην [SDS06] είναι η εξατομικευμένη ιεράρχηση αποτελεσμάτων μετα-αναζήτησης στον παγκόσμιο ιστό.

Στην [CHS04] προτείνεται μέθοδος κατηγοριοποίηση εγγράφων σε οντολογία , η οποία γίνεται με τεχνικές μη επιβλεπόμενης μηχανικής μάθησης, και άρα αδυνατεί να εκμεταλλευτεί τη γνώση από τα ήδη επισημασμένα έγγραφα.

Το [HM06] πραγματοποιεί μη επιβλεπόμενο αυτόματο σημασιολογικό χαρακτηρισμό κειμένων, χωρίς όμως την χρήση οντολογίας. Το [KN03] περιγράφει τρόπους κατηγοριοποίησης εγγράφων ως σχετικών με κάποια ευρεία σημασιολογική κατηγορία ή μη, χρησιμοποιώντας τεχνικές μηχανικής μάθησης, αλλά αφορά δυαδική κατηγοριοποίηση (Binary Classification). Το [DC00] εξετάζει ζητήματα διάκρισης μεταξύ γενικών ή ειδικότερων κατηγοριών που ανακύπτουν κατά την αυτόματη κατηγοριοποίηση εγγράφων.

Τέλος, όσον αφορά το GoNTogle 1.0, έχουν προταθεί διάφορες σχετικές εργασίες που συνδυάζουν τόσο αναζήτηση με λέξεις κλειδιά, όσο και σημασιολογική αναζήτηση με χρήση οντολογιών [KPO+04], [CFV07].

3.3 Θεωρητικό υπόβαθρο

Οι τεχνικές για τον αυτόματο σημασιολογικό χαρακτηρισμό έχουν πλούσιο θεωρητικό υπόβαθρο, που άπτεται πολλών περιοχών της Επιστήμης Υπολογιστών, όπως Μηχανική

Μάθηση, Ανάκτηση πληροφορίας και Οντολογίες. Ακολουθούν βασικές εισαγωγικές έννοιες και παρουσίαση των περιοχών αυτών.

3.3.1 Εισαγωγικές έννοιες

Γνώση είναι το σύνολο της οργανωμένης **πληροφορίας**, η επεξεργασία της οποίας οδηγεί τον άνθρωπο στην κατανόηση των φαινομένων και των αρχών του κόσμου που τον περιβάλλει, δίνοντάς του τα εφόδια για την αποδοτική λήψη αποφάσεων και επίλυση προβλημάτων.

Η πρόοδος της επιστήμης των υπολογιστών και της πληροφορικής και η αλματώδης ανάπτυξη του διαδικτύου έδωσαν τη δυνατότητα αποθήκευσης πολύ μεγάλου όγκου δεδομένων και πληροφοριών. Η «πληροφοριακή συμφόρηση» που δημιουργήθηκε, γέννησε την ανάγκη εύρεσης μηχανισμών ικανών για αποδοτική αξιοποίηση της πληθώρας των δεδομένων αυτών, με γρήγορο εντοπισμό και αποδοτική σταχυολόγηση των ζητούμενων πληροφοριών. Αποτέλεσμα ήταν η ανάπτυξη του «**Σημασιολογικού Ιστού**» (Semantic Web [LHL01]) και του τομέα της «**Ανακάλυψη Γνώσης**» (Knowledge Discovery), με σκοπό την ανάκτηση πληροφορίας από τα δεδομένα και την εκμαίευση και αναπαράσταση του νοήματος των πληροφοριών αυτών, με απώτερο στόχο την εξαγωγή **γνώσης**.

Για την ανάπτυξη τόσο του Σημασιολογικού Ιστού, όσο και της Ανακάλυψης Γνώσης, συνέβαλλαν διάφορες τεχνολογίες της πληροφορικής, που περιλαμβάνουν διαδικασίες για την αυτόματη εύρεση χρήσιμων δομών και προτύπων, που παράγουν γνώση από τα δεδομένα. Δομικά στοιχεία για τη δημιουργία των δομών και των προτύπων αυτών, αποτελούν η **Μηχανική Μάθηση** (Machine Learning) και η **Ανάκτηση Πληροφορίας** (Information Retrieval).

Για τη σημασιολογική αναπαράσταση της γνώσης, μία από τις βασικότερες τεχνολογίες που αναπτύχθηκαν και σχετίζεται με την Τεχνητή Νοημοσύνη, είναι η **Αναπαράσταση Γνώσης** (knowledge representation) μέσω **Οντολογιών**.

3.3.2 Μηχανική μάθηση

Η μηχανική μάθηση (machine learning) [Bur01] είναι ένας από τους ευρύτερους τομείς της Τεχνητής Νοημοσύνης, του κλάδου, δηλαδή, της Επιστήμης Υπολογιστών που ασχολείται με την ευφυή συμπεριφορά μη φυσικών μηχανικών οντοτήτων (υπολογιστικών μηχανών). Η «μηχανική μάθηση» αναφέρεται στην ικανότητα που αποκτούν τα υπολογιστικά συστήματα με την παροχή δεδομένων και τη χρήση κατάλληλων αλγορίθμων, στη λήψη αποφάσεων για την πραγματοποίηση ενεργειών. Έχει ένα μεγάλο φάσμα εφαρμογών, όπως σε μηχανές αναζήτησης, επεξεργασία φυσικής γλώσσας, αναγνώριση προτύπων, αναγνώριση φωνής, όραση υπολογιστών, ιατρική διάγνωση, βιοπληροφορική κά..

Ο όρος «μάθηση» έγκειται στο γεγονός ότι τα συστήματα αυτά δεν είναι στατικά, αλλά αλληλεπιδρώντας με το περιβάλλον δράσης τους, αυτοτροφοδοτούνται με νέα γνώση για τον τρόπο που πρέπει να ενεργούν. Έτσι «μαθαίνουν» και βελτιώνονται.

Σε τι όμως συνίσταται μάθηση ενός συστήματος και πώς εξασφαλίζεται η βελτίωση της συμπεριφοράς του; Το σύστημα καλείται να δραστηριοποιηθεί σε ένα συγκεκριμένο γνωστικό πεδίο. Ο σχεδιαστής του συστήματος λοιπόν, πρέπει αρχικά να βρει έναν τρόπο αναπαράστασης των εννοιών του κόσμου που αντιπροσωπεύει το πεδίο αυτό, πχ μια γλώσσα αναπαράστασης. Στη συνέχεια, ένα σύνολο κανόνων οι οποίοι εφαρμοζόμενοι επί των εννοιών αυτών, οδηγούν σε μετασχηματισμό των αρχικών αναπαραστάσεων μέσω γενικεύσεων, εξειδικεύσεων και λοιπόν τροποποιήσεων. Δημιουργείται έτσι ένας ευριστικός μηχανισμός, που επιτυγχάνει τη μάθηση του συστήματος καθοδηγώντας το σε κάθε βήμα δράσης του. Το ποσοστό, λοιπόν, κατανόησης του χώρου του προβλήματος, ο βαθμός απεικόνισης όλων των βασικών παραμέτρων για τη σχεδίαση και αναπαράστασή του, καθώς και η ύπαρξη ή όχι αρχικής γνώσης και δεδομένων εκπαίδευσης του συστήματος, είναι οι παράγοντες που καθορίζουν την αποτελεσματικότητά του.

3.3.2.1 Βασικές κατηγορίες μηχανικής μάθησης

Η μηχανική μάθηση μπορεί να διακριθεί σε τρεις κατηγορίες :

- Επιβλεπόμενη μάθηση (Supervised learning):

  Στην επιβλεπόμενη μάθηση δημιουργείται μια συνάρτηση που αντιστοιχεί δεδομένα εισόδου σε επιθυμητές εξόδους. Μία μορφή της επιβλεπόμενης μάθησης είναι η κατηγοριοποίηση (classification), όπου δίνονται στο σύστημα παραδείγματα αντικειμένων (δεδομένα εκπαίδευσης) που ανήκουν σε μια εκ των προτέρων γνωστή κατηγορία και αυτό πρέπει να εντοπίσει τα κοινά χαρακτηριστικά των αντικειμένων αυτών. Από τη διαδικασία αυτή εξάγονται κανόνες για το πώς θα κατηγοριοποιηθεί ένα άγνωστο αντικείμενο.

  Γνωστοί αλγόριθμοι επιβλεπόμενης μάθησης είναι, για παράδειγμα, ο αλγόριθμος «Απαλοιφής Υποψηφίων» και ο αλγόριθμος κατασκευής «Δένδρων Απόφασης».

- Μη επιβλεπόμενη μάθηση (Unsupervised learning):

  Στη μη επιβλεπόμενη μάθηση δε δίνονται στο σύστημα αρχικές κατηγορίες ούτε δεδομένα εκπαίδευσης. Αντίθετα, το σύστημα προσπαθεί να εντοπίσει κάποια κοινά χαρακτηριστικά και συσχετίσεις σε ένα σύνολο αντικειμένων, ώστε είτε να τα τοποθετήσει σε κατηγορίες που έχει δημιουργήσει μόνο του, χωρίς να του υποδειχθεί αν υπάρχουν τέτοιες κατηγορίες ή πόσες είναι σε αριθμό, είτε να βγάλει κάποια συμπεράσματα υπό μορφή προτύπων, για τα αντικείμενα αυτά.

Μία διαδικασία που χρησιμοποιείται στη μη επιβλεπόμενη μάθηση είναι η «συσταδοποίηση» (clustering), σύμφωνα με την οποία ένα σύνολο ετερογενών δεδομένων ομαδοποιείται με βάση ένα βαθμό ομοιότητας που παρατηρείται μεταξύ των δεδομένων αυτών. Οι ομάδες που δημιουργούνται δεν είναι προκαθορισμένες και επαφίεται στο αναλυτή των αποτελεσμάτων να ερμηνεύσει τη σημασία που έχει κάθε ομάδα που δημιουργείται. Για παράδειγμα, οι ομάδες διατροφικών συνηθειών ασθενών υποδεικνύουν το συσχετισμό της διατροφής με διάφορες παθήσεις.

Άλλη διαδικασία που χρησιμοποιείται είναι η μάθηση μέσω «κανόνων συσχέτισης» (associations rules). Με τη χρήση των κανόνων συσχέτισης προκύπτουν πρότυπα που έχουν τη μορφή «αν Χ τότε Υ», όπου Χ και Υ είναι εκφράσεις που συσχετίζουν αντικείμενα.

Γνωστοί αλγόριθμοι που χρησιμοποιούνται σε εφαρμογές μη επιβλεπόμενης μάθησης είναι k-Μέσων (k-Means), SOM (Self-Organizing Map) και ART (Adaptive resonance theory).

- Ενισχυτική μάθηση (Reinforcement learning):

Πρόκειται για μάθηση μέσω επιβράβευσης. Χρησιμοποιείται κυρίως σε νοήμονες πράκτορες (intelligent agents) οι οποίοι είναι οντότητες που αντιλαμβάνονται το περιβάλλον μέσα στο οποίο βρίσκονται με τη βοήθεια αισθητήρων, είναι μέρος του περιβάλλοντος αυτού, κάνουν συλλογισμούς για το περιβάλλον και δρουν πάνω σ' αυτό με τη βοήθεια μηχανισμών δράσης για την επίτευξη κάποιων στόχων. Ο στόχος των αλγορίθμων ενισχυτικής μάθησης είναι παρατηρώντας τα αποτελέσματα μιας αλληλουχίας ενεργειών, να βρουν μία πολιτική που να χαρτογραφεί τις καταστάσεις του περιβάλλοντος, σε ενέργειες που οφείλουν να πράττουν οι πράκτορες όταν βρεθούν στις καταστάσεις αυτές. Η μάθηση του συστήματος έγκειται στο ότι υπάρχει επιβράβευση του πράκτορα ανάλογα με το πόσο σωστός είναι ο τρόπος με τον οποίο επιλέγει να μεταβεί από μία κατάσταση σε μία άλλη.

Η διαφορά της ενισχυτικής μάθησης από την επιβλεπόμενη μάθηση είναι ότι στην πρώτη δεν χρησιμοποιείται προηγούμενη γνώση.

Οι ποιο γνωστοί αλγόριθμοι ενισχυτικής μάθησης είναι η «μάθηση Q» (Q learning), και ο αλγόριθμος SARSA (State-Action-Reward-State-Action).

3.3.3 Αυτόματη κατηγοριοποίηση κειμένων

Μία σημαντική εφαρμογή της μηχανικής μάθησης είναι η κατηγοριοποίηση εγγράφων (document classification/categorization), που αποτελεί μία από τις βασικές εργασίες εξόρυξης δεδομένων και ασχολείται με την ταξινόμηση ηλεκτρονικών εγγράφων σε μία ή περισσότερες κατηγορίες βάσει του περιεχομένου τους. Για παράδειγμα, η καταχώρηση ενός κειμένου

σχετικό με το χορό, θα μπορούσε να γίνει σε μία κατηγορία από τις ακόλουθες: «Τέχνες», «Πολιτιστικές εκδηλώσεις», «Αθλητισμός», «Διάσημοι χορευτές» κλπ. Όλες αυτές οι κατηγορίες είναι εύλογες, ανάλογα με το νόημα του κειμένου.

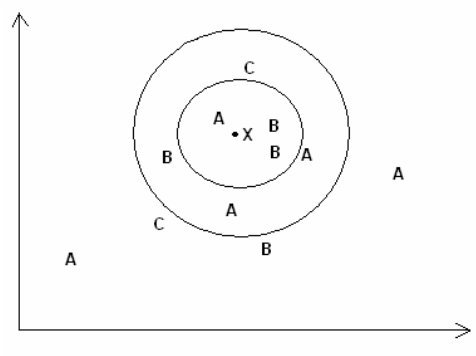
Ανήκει στις εφαρμογές επεξεργασίας φυσικής γλώσσας και χρησιμοποιείται με απώτερο στόχο την Ανάκτηση πληροφορίας, δηλαδή την εύρεση ηλεκτρονικών εγγράφων σχετικών με μία ερώτηση.

Για την αυτόματη ταξινόμηση κειμένων σε κατηγορίες μπορεί να χρησιμοποιηθεί είτε επιβλεπόμενη, είτε μη επιβλεπόμενη κατηγοριοποίηση. Στην επιβλεπόμενη κατηγοριοποίηση ένα άγνωστο κείμενο ανατίθεται σε κάποιες προκαθορισμένες κατηγορίες βάσει του περιεχομένου του και συγκριτικά με άλλα κείμενα που είναι γνωστή η κατηγορία στην οποία ανήκουν. Στην μη επιβλεπόμενη κατηγοριοποίηση, τα έγγραφα ταξινομούνται σε μη προκαθορισμένες κατηγορίες και χωρίς να υπάρχει αρχική γνώση.

Οι βασικές μέθοδοι που χρησιμοποιούνται για την κατηγοριοποίηση κειμένων είναι: Μέθοδος Bayes, Δέντρα Αποφάσεων, Νευρωνικά Δίκτυα, SVM (Support Vector Machines), k-Nearest Neighbor, Διανυσματικά μοντέλα (Vector Space Model) κά.

3.3.3.1 Αλγόριθμος k-Nearest Neighbor (k-NN)

Ο αλγόριθμος k-NN λόγω της απλότητάς του και της ευρείας εφαρμοσιμότητάς του είναι ένας από τους δημοφιλέστερους αλγορίθμους μηχανικής μάθησης. Χρησιμοποιείται για κατηγοριοποίηση αντικειμένων, βασιζόμενος στην παραδοχή ότι τα δεδομένα εκπαίδευσης μπορούν να αναπαρασταθούν ως σημεία σε κάποιο μετρικό χώρο, με διαστάσεις που σχετίζονται με τις παραμέτρους του προβλήματος. Ένα νέο αντικείμενο τοποθετείται στο χώρο αυτό ως νέο σημείο. Η κατηγοριοποίησή του γίνεται βάσει των αποτελεσμάτων ψηφοφορίας γειτονικών του αντικειμένων για τα οποία είναι γνωστή η κατηγορία που ανήκουν. Το νέο αντικείμενο τοποθετείται στην κατηγορία που πλειοψηφεί. Το k είναι ένας μικρός θετικός ακέραιος, που δηλώνει τον αριθμό των γειτόνων που θα ληφθούν υπόψη. Στο παράδειγμα του παρακάτω σχήματος, το άγνωστο αντικείμενο X ανατίθεται στην κατηγορία B αν k=3 (εσωτερικός κύκλος), ενώ για k=7 ανατίθεται στην κατηγορία A (εξωτερικός κύκλος).



Σχήμα 11

Η τιμή του k διαδραματίζει σημαντικό ρόλο στην αποδοτικότητα του αλγορίθμου. Η τιμή που επιλέγεται εξαρτάται από τα δεδομένα του προβλήματος (πχ μία τιμή από 5 έως 10 είναι συνήθης επιλογή για δεδομένα λίγων διαστάσεων). Υπάρχουν τεχνικές για τον υπολογισμό του ενδεδειγμένου k, η γνωστότερη από τις οποίες είναι η «διασταυρωμένη επικύρωση» (cross validation) [Koh95].

Η απόσταση των k κοντινότερων γειτόνων υπολογίζεται με κάποια μετρική απόστασης όπως πχ η Ευκλείδια απόσταση (L2), η απόσταση Manhattan (L1), το Μέτρο επικάλυψης (Overlap metric) ή το Τροποποιημένο Μέτρο Διαφοράς Τιμών (Modified Value Difference Metric MVDM) [Μαλ05].

Ο συνήθης τρόπος που ακολουθείται για την ψηφοφορία είναι αυτός όπου οι ψήφοι όλων των γειτόνων είναι ισοβαρείς. Σ' αυτό τον τρόπο ψηφοφορίας δε λαμβάνεται υπ' όψη η απόσταση των γειτόνων από το εξεταζόμενο αντικείμενο, κάτι που θα ήταν ίσως επιθυμητό, αφού οι πλησιέστεροι γείτονες, πιθανώς, το προσεγγίζουν περισσότερο. Μία μέθοδος ψηφοφορίας που συνυπολογίζει την απόσταση των γειτόνων χρησιμοποιείται από τον λεγόμενο distance-weighted k-NN αλγόριθμο [Dud76], δίνοντας ένα βάρος σε κάθε ψήφο ανάλογο του αντιστρόφου της απόστασης.

Ο αλγόριθμος k-NN χρησιμοποιείται πολύ συχνά για κατηγοριοποίηση εγγράφων. Στην περίπτωση αυτή τα βήματα εκτέλεσης του αλγορίθμου είναι τα εξής:

- Φάση εκπαίδευσης του αλγορίθμου: Αποθηκεύονται τα έγγραφα εκπαίδευσης, των οποίων είναι γνωστή η κατηγορία στην οποία ανήκουν, με τη μορφή του διανυσματικού μοντέλου.
- Φάση κατηγοριοποίησης: Υπολογίζεται η διανυσματική αναπαράσταση του άγνωστου προς κατηγοριοποίηση εγγράφου. Υπολογίζεται η απόστασή του από όλα τα δεδομένα εκπαίδευσης και επιλέγονται τα k κοντινότερα. Το άγνωστο έγγραφο κατατάσσεται στην κατηγορία που πλειοψηφεί μεταξύ των k που επιλέχθηκαν.

3.3.4 Ανάκτηση πληροφορίας (IR)

Ανάκτηση πληροφορίας (Information Retrieval) είναι η επιστήμη που ασχολείται με την εύρεση εγγράφων, εύρεση πληροφορίας στο περιεχόμενο εγγράφων, εύρεση μεταδεδομένων σχετικά με έγγραφα, τους, καθώς και ανάκτηση πληροφορίας από βάσεις δεδομένων και τον Παγκόσμιο Ιστό (WWW). Η *αυτόματη* ανάκτηση πληροφορίας, ασχολείται με την εύρεση ηλεκτρονικών εγγράφων σχετικών με μια ερώτηση. Στις πιο συχνά χρησιμοποιούμενες προσεγγίσεις για την εύρεση απαντήσεων σε τέτοια ερωτήματα, τα έγγραφα χαρακτηρίζονται από *λέξεις κλειδιά*, ενώ η ερώτηση είναι μια σύζευξη λέξεων. Αυτή η προσέγγιση χρησιμοποιείται σήμερα από όλες τις μηχανές αναζήτησης (search engines) στο διαδίκτυο, δίνει όμως πολλά αποτελέσματα άσχετα με την αναζητούμενη πληροφορία. Μία ιδανική περίπτωση θα ήταν η μηχανή αναζήτηση να μπορεί να κάνει νοηματική επεξεργασία των εγγράφων, η δε ερώτηση να διατυπώνεται σε φυσική γλώσσα. Θα ήταν επιθυμητό σε αυτή την περίπτωση η μηχανή αναζήτησης να μην επιστρέφει κείμενα που απλά περιέχουν τις λέξεις κλειδιά, αλλά κείμενα το νόημα των οποίων ταιριάζει με την ερώτηση που υποβάλλεται, ανεξάρτητα από τις λέξεις που χρησιμοποιήθηκαν ή που περιέχει το κείμενο. Θα θέλαμε δηλαδή να μπορεί η μηχανή αναζήτησης να κατηγοριοποιεί τα έγγραφα βάσει του νοηματικού περιεχομένου τους (βλ και [BKB+02]).

Ακολούθως, περιγράφονται μερικές βασικές λειτουργίες ενός συστήματος ανάκτησης πληροφορίας στα πλαίσια της αυτόματης κατηγοριοποίησης κειμένων.

3.3.4.1 Προεπεξεργασία κειμένου

Πρώτο βήμα για την αυτόματη κατηγοριοποίηση κειμένων είναι η εύρεση ενός τρόπου αναπαράστασης του περιεχομένου τους. Οι λέξεις που περιέχονται σε ένα κείμενο, είναι στην ουσία τα δομικά στοιχεία που το περιγράφουν. Οι λέξεις αυτές υπόκεινται λεξικολογική ανάλυση, ώστε να εντοπιστούν ποιες από αυτές και σε ποια μορφή αποτελούν τους ιδανικότερους περιγραφείς (descriptors) του κειμένου.

Αρχικά αφαιρούνται οι κοινές λέξεις (stopwords), αυτές δηλαδή που ενώ εμφανίζονται σε μεγάλη συχνότητα, δεν προσδίδουν χρήσιμη πληροφορία για το περιεχόμενο του εγγράφου (πχ άρθρα). Στη συνέχεια, πραγματοποιείται στελέχωση των λέξεων (stemming), δηλαδή αφαιρείται η κατάληξη των λέξεων ώστε να παραχθεί η ρίζα τους (πχ μετά τη στελέχωση οι λέξεις «ταξίδι», «ταξιδεύω», «ταξιδιώτης» μετατρέπονται σε «ταξιδ»). Για τη στελέχωση έχουν αναπτυχθεί διάφοροι αλγόριθμοι (κυρίως για την αγγλική γλώσσα) με πιο γνωστό τον αλγόριθμο Porter [Por80].

Ως αποτέλεσμα της λεξικολογικής διαδικασίας, παράγεται ένα σύνολο ελάχιστων λεκτικών μονάδων (tokens), που αποτελούν τους περιγραφείς του κειμένου. Και μάλιστα, ο όγκος των λεκτικών μονάδων αυτών είναι κατά πολύ μικρότερος του αρχικού κειμένου.

3.3.4.2 Ανάθεση βάρους σημαντικότητας λέξεων

Η διαδικασία αυτή (weighting) σκοπό έχει να αναθέσει ένα βάρος σε κάθε λέξη-περιγραφέα που προέκυψε από την παραπάνω ανάλυση, το οποίο απεικονίζει τη σημαντικότητα που έχει κάθε λέξη σε ένα έγγραφο. Η σημαντικότητα αυτή εξαρτάται τόσο από τη συχνότητα εμφάνισης μιας λέξης σε ένα έγγραφο, όσο και από τη συχνότητα εμφάνισής της στο ευρετήριο της συλλογή όλων των εγγράφων (που αποτελούν τον «κόσμο» απ' όπου θα προκύψουν τα στατιστικά εμφάνισης των λέξεων). Για την παραγωγή των συμπερασμάτων αυτών σημαντικό ρόλο έπαιξαν οι η παρατηρήσεις του George Zipf [Zip49] οι οποίες διαμόρφωσαν και το ομώνυμο νόμο: Η σημαντικότητα μίας λέξης σε ένα έγγραφο είναι αντιστρόφως ανάλογη της συχνότητας εμφάνισης της σ' αυτό.

Συνυπολογίζοντας τα παραπάνω, αναπτύχθηκαν διάφοροι τρόποι υπολογισμού των βαρών οι οποίοι υπακούν στον εξής κανόνα: «Αν μια λέξη εμφανίζεται σε πολλά έγγραφα δεν πρέπει να θεωρείται σημαντικότερη από μία λέξη που εμφανίζεται σε λίγα και ένα έγγραφο που περιέχει πολλές εμφανίσεις μιας λέξης δεν πρέπει να θεωρείται λιγότερο σημαντικό από ένα άλλο με λίγες εμφανίσεις» [WMB99].

Η πιο γνωστή μέθοδος υπολογισμού των βαρών είναι η **tf-idf**, και δίνεται από τη σχέση

$$w_{ij} = tf_{ij} \cdot idf_i ,$$

όπου w_{ij} είναι το βάρος του όρου i για ένα έγγραφο j , tf_{ij} είναι η συχνότητα εμφάνισης του όρου i στο έγγραφο και idf_j (inverse document frequency of term i) είναι η αντίστροφη εμφάνιση εγγράφων της συλλογής που περιέχουν τον όρο i.

Συνήθως τα tf_{ij} και idf_j κανονικοποιούνται ώστε να αποφευχθούν μεροληψίες που οφείλονται πχ στο μέγεθος των εγγράφων. Από τις συνηθέστερα χρησιμοποιούμενες σχέσεις υπολογισμού τους είναι οι ακόλουθες:

$$tf_{ij} = \frac{f_{ij}}{\max_k \{f_{kj}\}},$$

όπου f_{ij} είναι η συχνότητα εμφάνισης του όρου i στο έγγραφο j και $\max_k \{f_{kj}\}$ είναι το μέγιστο πλήθος εμφανίσεων ενός όρου στο έγγραφο j.

$$idf_i = \log_2 (\frac{N}{df_i}),$$

όπου N είναι το πλήθος των εγγράφων της συλλογής.

3.3.4.3 Διανυσματικό μοντέλο αναπαράστασης εγγράφων

Υπάρχουν πολλά μοντέλα που χρησιμοποιούνται από τους αλγόριθμους κατηγοριοποίησης για την αναπαράσταση εγγράφων. Τα κλασσικότερα από αυτά είναι το Boolean (ένα κείμενο περιέχει ή δεν περιέχει μία λέξη), το Διανυσματικό (Vector space model) (όπου παρέχεται η σημαντικότητα κάθε λέξης στο κείμενο), ενώ υπάρχουν και άλλα όπως το Πιθανοκρατικό, το Ασαφές (Fuzzy model), το μοντέλο Λανθάνοντος σημασιολογικού ευρετηριασμού (Latent semantic indexing) μοντέλα δομημένου κειμένου, νευρωνικού δικτύου κά.

Το πιο συχνά χρησιμοποιούμενο μοντέλο αναπαράστασης εγγράφων, από αλγορίθμους κατηγοριοποίησης είναι το Διανυσματικό μοντέλο (Vector space model). Πρόκειται για διανυσματικό τρόπο αναπαράστασης του κειμένου ενός εγγράφου.

Κάθε έγγραφο d_j αναπαρίσταται από ένα διάνυσμα $d_j=(w_{1,j},\ldots,w_{t,j})$, όπου $w_{i,j} \in [0,1]$ είναι το βάρος της λέξης k_i για το κείμενο d_j και $K=\{k_1,\ldots,k_t\}$ το σύνολο όλων των λέξεων που σχηματίζουν το ευρετήριο της συλλογής εγγράφων, η οποία θα αποτελέσει τον «κόσμο» απ' όπου θα προκύψουν τα στατιστικά εμφάνισης των λέξεων.

Για τον προσδιορισμό της ομοιότητας κειμένων μέσω του διανυσματικού μοντέλου χρησιμοποιούνται κυρίως δύο μέθοδοι: η μέθοδος «Εσωτερικού Γινομένου» (Inner Product) και η «Ομοιότητα Συνημιτόνου» (Cosine Similarity) [Τζι08].

Με τη μέθοδο εσωτερικού γινομένου, η ομοιότητα δύο εγγράφων που παριστάνονται με τα διανύσματα d και q, ορίζεται από το εσωτερικό τους γινόμενο, που δίνεται από την παρακάτω σχέση:

$$sim(d_j, q) = \vec{d_j} \cdot \vec{q} = \sum_{i=1}^{t} w_{ij} \cdot w_{iq} \ ,$$

Όπου w_{ij} το βάρος του όρου i στο έγγραφο j, w_{iq} το βάρος του όρου i στο έγγραφο q και t το πλήθος των διαφορετικών όρων στη συλλογή.

Το Μέτρο Ομοιότητας Συνημίτονου υπολογίζεται από τη σχέση:

$$\cos(d_j, q) = \frac{\sum_{i=1}^{t} (w_{ij} \cdot w_{iq})}{\sqrt{\sum_{i=1}^{t} w_{ij}^2 \cdot \sum_{i=1}^{t} w_{iq}^2}}$$

Η δεύτερη μέθοδος, επειδή λαμβάνει υπ' όψη τα σχετικά μεγέθη των διανυσμάτων, γενικά δίνει καλύτερα αποτελέσματα, γι' αυτό και χρησιμοποιείται πιο συχνά στην πράξη.

3.3.4.4 Αξιολόγηση αποτελεσμάτων – Κλασσικές μετρικές

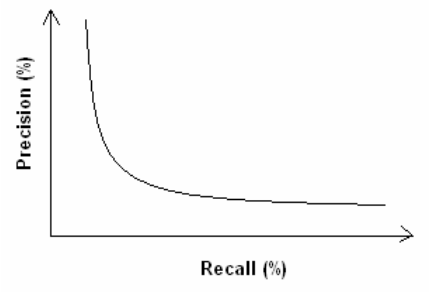
Ο συνηθέστερος τρόπος αξιολόγησης της αποδοτικότητας ενός συστήματος ανάκτησης είναι υπολογίζοντας πόσα από τα σωστά έγγραφα ανακτήθηκαν και σε ποια θέση ταξινομήθηκαν.

Τα μεγέθη που χρησιμοποιούνται είναι η ακρίβεια P (precision) η οποία εκφράζει την ικανότητα ανάκτησης μόνο συναφών εγγράφων και η ανάκληση R (recall), που εκφράζει την ικανότητα ανάκτησης όλων των συναφών εγγράφων της συλλογής. Υπολογίζονται από τις παρακάτω σχέσεις:

$$\text{P} = \frac{\text{Αριθμός συναφών εγγράφων που ανακτήθηκαν}}{\text{Αριθμός όλων των εγγράφων που ανακτήθηκαν}}$$

$$\text{R} = \frac{\text{Αριθμός συναφών εγγράφων που ανακτήθηκαν}}{\text{Αριθμός όλων των συναφών εγγράφων της συλλογής}}$$

Τα δύο αυτά μεγέθη έχουν αρνητική συσχέτιση. Για τον συσχετισμό τους χρησιμοποιούνται οι καμπύλες ανάκλησης-ακρίβειας (recall-precision), στις οποίες στην ουσία απεικονίζεται η ακρίβεια για διάφορες τιμές ανάκλησης. Μία τέτοια καμπύλη φαίνεται στο παρακάτω σχήμα:



Σχήμα 12

Ένας ιδανικός αλγόριθμος θα ανακτούσε όλα τα συναφή έγγραφα και μόνο αυτά. Σ' αυτή την περίπτωση η ακρίβεια θα ήταν 100% για κάθε επίπεδο ανάκλησης και η καμπύλη ανάκλησης-ακρίβειας θα ήταν μία οριζόντια γραμμή.

Υπάρχουν και μέτρα της αποτελεσματικότητας ενός συστήματος που συνδυάζουν τις P και R σε ένα μέγεθος. Για παράδειγμα, το μέτρο F-Measure που υπολογίζει τον αρμονικό τους μέσο από τη σχέση:

$$F = \frac{2PR}{P+R}.$$

Υψηλή τιμή του F προκύπτει για υψηλές τιμές P και R.

Παραλλαγή του F-Measure είναι το E-Measure που δίνεται από τη σχέση:

$$E = \frac{(1+\beta^2)PR}{\beta^2 P + R},$$

Όπου για β>1 ενισχύεται η ακρίβεια, για β<1 η ανάκληση, ενώ για β=1 προκύπτει το μέτρο F.

3.3.4.5 Αξιολόγηση αποτελεσμάτων – Μετρικές για διαδραστικό σύστημα

Οι ανωτέρω μετρικές είναι καθιερωμένες και κοινά αποδεκτές, εντούτοις αδυνατούν να περιγράψουν πλήρως τη συμπεριφορά ενός αλληλεπιδραστικού συστήματος, όπως το σύστημα Αυτόματου σημασιολογικού χαρακτηρισμού του GoNTogle 2.0. Πράγματι, το σύστημα προτείνει έναν αριθμό αποτελεσμάτων, ο χρήστης εξετάζει μερικά από αυτά, και τελικά αποφασίζει αν και ποιο θα δεχθεί. Θα ήταν λάθος να ερμηνεύονταν τα προτεινόμενα αποτελέσματα ως αποφάσεις κατηγοριοποίησης ενός εγγράφου σε πολλαπλές κατηγορίες. Για να αναδείξουμε τη συμπεριφορά αυτή, χρησιμοποιούμε την εξής παραλλαγή μετρικών, παρορμώμενοι από τις κλασσικές μετρικές.

Η **Τροποποιημένη Ακρίβεια** μετράει τον λόγο του αριθμού των εγγράφων στα οποία το σωστό αποτέλεσμα ανήκε στα προτεινόμενα, προς τον συνολικό αριθμό εγγράφων.

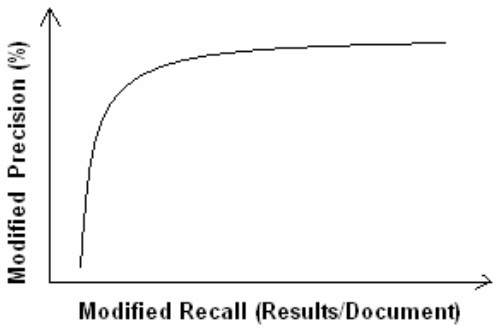
$$\text{Τροπ/νη Ακρίβεια} = \frac{\text{Αριθμός εγγράφων με ένα τουλ. σωστό σημασιολογικό χαρ/μό}}{\text{Αριθμός εγγράφων των οποίων επιχειρήθηκε ο χαρ/μός}}$$

Η δε **Τροποποιημένη Ανάκληση** μετράει τον λόγο των αποτελεσμάτων που προτείνονται στο χρήστη, προς τον αριθμό των εγγράφων για τα οποία ζητήθηκαν αποτελέσματα.

$$\text{Τροπ/νη Ανάκληση} = \frac{\sum_{\forall \text{έγγραφο i}} \text{Αριθμός προτεινόμενων αποτελεσμάτων για το i}}{\text{Αριθμός εγγράφων των οποίων επιχειρήθηκε ο χαρ/μός}}$$

Έτσι, για παράδειγμα, αν σε κάποιο πείραμα παρατηρήθηκε Τροπ/νη Ακρίβεια 0.9 και Τροπ/νη Ανάκληση 2.1 , αυτό σημαίνει ότι, χρησιμοποιώντας το σύστημα Αυτόματου σημασιολογικού χαρακτηρισμού, ο χρήστης μπόρεσε να χαρακτηρίσει σωστά το 90% των εγγράφων που επιχείρησε, και για να το κάνει αυτό, χρειάστηκε να εξετάσει 2.1 προτεινόμενα αποτελέσματα ανά έγγραφο, κατά μέσο όρο.

Η αναμενόμενη μορφή της καμπύλης Τροποποιημένης Ακρίβειας / Τροποποιημένης Ανάκλησης είναι αυτή που φαίνεται στο Σχήμα 13 σε αντίθεση με αυτήν της Ακρίβειας / Ανάκλησης, λόγω της διαφορετικής σημασιολογίας που έχει η Τροποποιημένη Ανάκληση.



Σχήμα 13

3.3.5 Οντολογίες

Ο όρος «οντολογία» (ontology) είναι όρος δανεισμένος από τη φιλοσοφία και εκφράζει έναν τρόπο αναπαράστασης των εννοιών ενός τομέα του πραγματικού κόσμου κατά τέτοιο τρόπο, ώστε αφενός να είναι αντιληπτή και κοινώς αποδεκτή από τον άνθρωπο, αφετέρου να είναι τυπικά ορισμένη για να μπορεί να χρησιμοποιηθεί αποδοτικά από μηχανικά συστήματα. Οι οντολογίες χρησιμοποιούνται σε διάφορες επιστήμες όπως στην Επιστήμη Υπολογιστών, την Οικονομική Επιστήμη και τη Βιοϊατρική, όταν απαιτείται ένα μοντέλο απεικόνισης ενός γνωστικού πεδίου ικανό να χρησιμοποιηθεί από υπολογιστές.

Ευρεία χρήση οντολογιών γίνεται από το Σημασιολογικό Ιστό, με στόχο τη σημασιολογική αναπαράσταση εγγράφων και αρχείων πολυμέσων (πχ εικόνες, βίντεο) η οποία θα επιτρέψει σε διάφορες δικτυακές εφαρμογές (πχ μηχανές αναζήτησης) να αποκτήσουν κάποια «ευφυΐα» προσεγγίζοντας την ανθρώπινη σκέψη.

Οι μηχανές αναζήτησης του διαδικτύου βασίζονται σε αναζήτηση με λέξεις κλειδιά που περιέχονται στα έγγραφα και τις ιστοσελίδες. Έτσι τα αποτελέσματα που επιστρέφουν αφενός περιέχουν πολλά έγγραφα που ενώ περιέχουν τις λέξεις αναζήτησης, τα περισσότερα είναι άσχετα με την αναζητούμενη πληροφορία, αφετέρου δε συμπεριλαμβάνονται επιθυμητά έγγραφα που δεν περιέχουν τις λέξεις (πχ περιέχουν συνώνυμά τους). Οι χρήση οντολογιών από μηχανές αναζήτησης, θα μπορούσε να επικεντρώσει την αναζήτηση στο εννοιολογικό περιεχόμενο των εγγράφων.

Οι οντολογίες διακρίνονται σε οντολογίες θεματικής περιοχής (domain ontology) και θεμελιώδεις (upper-level or foundation ontology). Οι οντολογίες θεματικής περιοχής προσδιορίζονται από λεξιλόγιο το οποίο έχει το συγκεκριμένο νόημα που τους προσδίδει η θεματική περιοχή. Για παράδειγμα η λέξη «μάρκα» έχει διαφορετική σημασία όταν χρησιμοποιείται για το καζίνο και διαφορετική για να περιγράψει πχ μάρκα αυτοκινήτων. Οι θεμελιώδεις οντολογίες μοντελοποιούν αντικείμενα χρησιμοποιώντας ένα «γλωσσάρι» (glossary) ένα λεξικό δηλαδή, που περιέχει μία λίστα όρων με τις εξηγήσεις τους των εννοιών που σχετίζονται με ένα συγκεκριμένο θεματικό τομέα. Γνωστά παραδείγματα τέτοιων οντολογιών είναι η SUMO[NP01] και η DOLCE [GGM+02].

Οι βασικές συνιστώσες που μοντελοποιούνται σε μία οντολογία είναι οι ακόλουθες:

- *Στιγμιότυπα* (individuals), δηλαδή τα αντικείμενα του κόσμου που μοντελοποιεί η οντολογία.
- *Κλάσεις* (classes), οι οποίες απεικονίζουν κατηγορίες στις οποίες ταξινομούνται τα στιγμιότυπα. Οι κλάσεις μπορεί να περιέχουν αντικείμενα ή υποκλάσεις. Πχ η κλάση «Θηλαστικά» απεικονίζει μια κατηγορία της οντολογίας «Ζώα», η οποία χρησιμοποιείται για την κατηγοριοποίηση όλων των ζώων.

- *Ιδιότητες* (attributes) που προσδίδουν χαρακτηριστικά στα αντικείμενα
- *Σχέσεις* (relations) που καθορίζουν πώς συσχετίζονται οι κλάσεις και τα αντικείμενα αναμεταξύ τους.
- Συναρτήσεις, περιορισμοί και κανόνες που εμπλέκουν τα παραπάνω.

3.3.5.1 Γλώσσες οντολογιών

Έχουν αναπτυχθεί πολλές γλώσσες για την κατασκευή οντολογιών (πχ OWL, DAML+OIL [1], KIF [2]) καθώς και γραφικά περιβάλλοντα ανάπτυξης (πχ Protégé [7], Swoop [3], Ontolingua [4], OntoEdit [5], JOE [6]). Οι περισσότερες γλώσσες βασίζονται σε τεχνολογίες όπως η XML (Extenseble Markup Language) και η RDF (Resource Description Framework), δηλαδή χρησιμοποιούν δενδρικές δομές εμφωλιασμένων ετικετών όπως στην XML και αναπαράσταση σημασιολογίας που κωδικοποιούνται μέσω τριάδων της RDF.

3.3.5.2 OWL

Η γλώσσα OWL (Web Ontology Language) [GH04] είναι η πιο εκφραστική γλώσσα οντολογιών που έχει αναπτυχθεί για το Σημασιολογικό Ιστό. Μπορεί να θεωρηθεί ως μια συλλογή RDF τριάδων (δηλαδή τριάδων της μορφής (Οντότητα1, Κατηγόρημα, Οντότητα2) πχ (Λιλίκα, Γράφει, Κώδικα)), στις οποίες όμως προσδίδει μια ιδιαίτερη σημασιολογία.

Τα βασικά δομικά στοιχεία της είναι τα ακόλουθα:

Class: Δημιουργία μίας κλάσης η οποία θα περιλαμβάνει στιγμιότυπα που ανήκουν στην κατηγορία που ορίζει η κλάση. Επίσης επιτρέπεται ο ορισμός ιεραρχίας υποκλάσεων χρησιμοποιώντας το στοιχείο rdfs:subClassOf.

Individual: Αφορά τα στιγμιότυπα των κλάσεων.

rdfs:Property: Εκφράζει ιδιότητες που ισχύουν για τα στιγμιότυπα (ObjectProperty) ή σχέση τιμής δεδομένου με ένα στιγμιότυπο (DatatypeProperty). Επίσης είναι δυνατή η δημιουργία ιεραρχίας ιδιοτήτων με το στοιχείο rdfs:subPropertyOf. Με το rdfs:domain ορίζεται το πεδίο ορισμού μίας ιδιότητας καθορίζοντας σε ποιων κλάσεων τα στιγμιότυπα μπορεί να εφαρμοστεί.. Κατ' ανάλογο τρόπο ορίζεται το πεδίο τιμών μιας ιδιότητας rdfs:range.

Η γλώσσα OWL χωρίζεται σε τρία επίπεδα εκφραστικότητας: OWL Lite, OWL DL και OWL Full. Στο GoNTogle χρησιμοποιείται το επίπεδο εκφραστικότητας OWL Lite.

3.4 Ανάλυση και σχεδίαση

Η ανάγκη που μας οδήγησε στην ανάπτυξη του υποσυστήματος αυτόματου σημασιολογικού χαρακτηρισμού εγγράφων, ήταν ότι έπρεπε να μπορεί ο χρήστης να κατηγοριοποιεί το μεγάλο όγκο εγγράφων του με λίγο κόπο. Στο αρχικό σύστημα, για να κατηγοριοποιηθεί ένα έγγραφο έπρεπε (i) ανοιχθεί η ιεραρχία της οντολογίας και (ii) να επιλεγεί ο κατάλληλος κόμβος-κατηγορία. Η διαδικασία αυτή έχει τα εξής μειονεκτήματα: αφενός όταν χρησιμοποιείται οντολογία μεγάλου μεγέθους, απαιτείται πολύς χρόνος για την ανεύρεση του επιθυμητού κόμβου, αφετέρου ο χρήστης πρέπει να θυμάται το περιεχόμενο του εγγράφου ώστε να επιλέξει ποιος κόμβος ταιριάζει σημασιολογικά με το περιεχόμενο αυτό. Είναι φανερό, ότι η διαδικασία αυτή είναι επίπονη, πόσο μάλλον όταν πρόκειται για την κατηγοριοποίηση μεγάλου όγκου εγγράφων.

Έτσι λοιπόν, ήταν αναγκαίο να αναπτυχθεί ένα υποσύστημα, το οποίο χρησιμοποιώντας έναν αλγόριθμο μηχανική μάθησης, να μπορεί να αποφασίζει αυτόματα ποιος κόμβος αντιστοιχεί σημασιολογικά σε κάθε έγγραφο προς κατηγοριοποίηση. Για την εκπαίδευση του αλγορίθμου αυτού, απαιτείται ένα σύνολο εγγράφων εκπαίδευσης (training data set), για κάθε κόμβο της εκάστοτε οντολογίας.

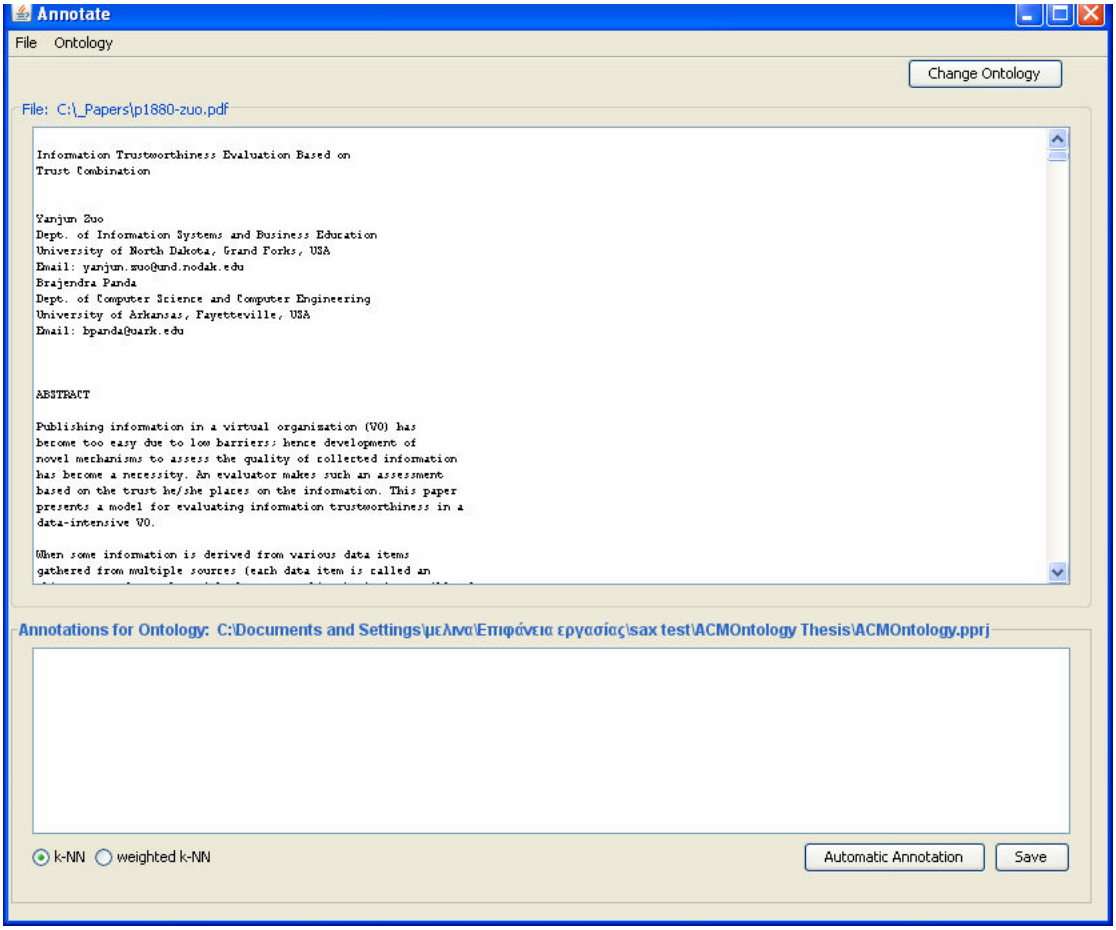
Δεδομένου ότι το GoNTogle δεν κατασκευάστηκε για να λειτουργεί με μία μόνο συγκεκριμένη οντολογία, προέκυψε το ζήτημα της εξασφάλισης του κατάλληλου συνόλου τέτοιων εγγράφων, για τους κόμβους *οποιασδήποτε* οντολογίας εισαχθεί στο σύστημα (είτε αυτή έχει κατασκευαστεί από τρίτους, είτε από τον ίδιο το χρήστη). Για παράδειγμα, έστω ότι ο χρήστης χρησιμοποιεί μία έτοιμη οντολογία που συνοδεύεται από το αντίστοιχα έγγραφα εκπαίδευσης. Δύο προβλήματα μπορεί να προκύψουν: Αφενός αν ο χρήστης προσθέσει νέους κόμβους στην οντολογία, θα πρέπει να εξασφαλίζονται δεδομένα εκπαίδευσης γι' αυτούς. Αφετέρου μπορεί ο χρήστης να αποδίδει διαφορετικό νόημα στο περιεχόμενο κάποιας κατηγορίας (κατά την υποκειμενική του κρίση), οπότε να διαφωνεί με το δοθέν σύνολο εγγράφων εκπαίδευσης αυτής. Τέλος, επιθυμητό θα ήταν το σύστημα να μπορεί να βελτιώνεται καθώς αυξάνεται η αποθηκευμένη σε αυτό γνώση και να μαθαίνει από τα λάθη του.

Η ιδέα αυτή υιοθετήθηκε για την υλοποίηση του Auto Annotation ως εξής: Από τη στιγμή που ο χρήστης χαρακτηρίζει σημασιολογικά ένα έγγραφο, το έγγραφο αυτό μπορεί πλέον να συμμετέχει στα έγγραφα εκπαίδευσης του αντίστοιχου κόμβου της οντολογίας. Έτσι, το σύστημα *μαθαίνει* κάθε φορά που κατηγοριοποιείται ένα νέο έγγραφο, με αποτέλεσμα να χτίζεται σταδιακά το σύνολο των δεδομένων εκπαίδευσης ολόκληρης της οντολογίας και κατ' επέκταση να βελτιώνονται οι επιδόσεις του συστήματος.

Ένα μικρό μειονέκτημα της προσέγγισης αυτής είναι το γεγονός ότι ο χρήστης αρχικά θα καταφύγει στην χειρωνακτική κατηγοριοποίηση, ώστε να αποκτήσει το σύνολο εκπαίδευσης τα πρώτα του έγγραφα. (Για την εξάλειψη αυτής της ατέλειας, προτείνεται μία λύση στο κεφάλαιο 7.2, ως μελλοντική επέκταση του συστήματος.)

3.4.1 Λειτουργικότητα Auto Annotation

Όταν ο χρήστης επιθυμεί να κατηγοριοποιήσει ένα έγγραφο, ανοίγει το πλαίσιο σημασιολογικού χαρακτηρισμού "Annotate", επιλέγει το προς κατηγοριοποίηση έγγραφο, το κείμενο του οποίου προβάλλεται στην αντίστοιχη περιοχή του πλαισίου και φορτώνει την επιθυμητή οντολογία.

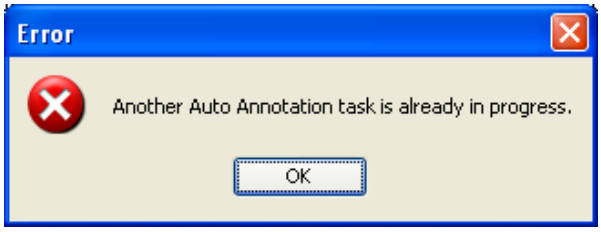


Σχήμα 14

Η διαδικασία του αυτόματου χαρακτηρισμού ξεκινάει πατώντας το κουμπί "Automatic Annotation".

Το σύστημα εξασφαλίζει ότι μόνο μία ενέργεια αυτόματου χαρακτηρισμού μπορεί να πραγματοποιείται κάθε φορά. Αν ο χρήστης πατήσει ξανά το κουμπί "Automatic Annotation"

προτού ολοκληρωθεί η προηγούμενη εργασία Auto Annotation, εμφανίζεται το ακόλουθο μήνυμα λάθους:



Σχήμα 15

Το πάτημα του κουμπιού "Automatic Annotation" έχει ως αποτέλεσμα την άντληση του κειμένου του επιλεχθέντος εγγράφου. Ακολουθεί λεξικογραφική ανάλυση του κειμένου από το Lucene για τη δημιουργία των ελάχιστων λεκτικών μονάδων (tokens), με μετατροπή των κεφαλαίων γραμμάτων σε πεζά, αφαίρεση τόνων, διαγραφή των κοινών λέξεων (stop words) και τέλος στελέχωση, δηλαδή αφαίρεση των καταλήξεων των λέξεων ώστε να παραχθεί η ρίζα τους (stemming). Στη συνέχεια, δημιουργείται το διάνυσμα συχνοτήτων των λεκτικών μονάδων του κειμένου (tf vector).

Ακολούθως, κατασκευάζεται ένα ερώτημα (query) στο οποίο, εισάγονται οι όροι (terms) του παραπάνω διανύσματος, αφού τεθεί σε κάθε έναν απ' αυτούς, ως βάρος για την αναζήτηση, η συχνότητα εμφάνισης του όρου στο κείμενο. Τέλος, πραγματοποιείται αναζήτηση στα ευρετηριοποιημένα (indexed) έγγραφα, των πιο όμοιων από αυτά με το ερώτημα αναζήτησης.

Η αναζήτηση επιστρέφει, ως αποτέλεσμα, έγγραφα ταξινομημένα σύμφωνα με το βαθμό ομοιότητάς τους.

3.4.1.1 Βαθμός ομοιότητας

Ο υπολογισμός του βαθμού ομοιότητας ενός ερωτήματος q με ένα έγγραφο d γίνεται με την ακόλουθη συνάρτηση που χρησιμοποιεί το Lucene [GH05], η οποία υπολογίζει μια παραλλαγή της απόστασης συνημιτόνου (cosine similarity):

$$score(q,d) = \sum_{t \in q} tf_{t \in d} \cdot idf^2(t) \cdot w_{t \in q} \cdot lengthNorm(t \in d) \cdot coord(q,d) \cdot queryNorm(\sum_{t \in q}(idf(t) \cdot w_{t \in q})^2)$$

Όπου:

$tf = \sqrt{freq}$	Κανονικοποιημένη συχνότητα εμφάνισης (freq) του όρου t στο έγγραφο d
$idf = \log(\frac{numDocs}{docFreq + 1}) + 1$	Αντίστροφη συχνότητα εμφάνισης (inverse document frequency) του όρου t σε πλήθος docFreq εγγράφων συγκριτικά με το συνολικό αριθμό numDocs εγγράφων της συλλογής

$w_{t \in q}$	Βάρος του όρου t στο q
$lengthNorm(t \in d)$	Παράγοντας κανονικοποίησης που εξαρτάται από το αντίστροφο του συνολικού πλήθους των όρων t του εγγράφου d
$coord(q,d)$	Παράγοντας που εξαρτάται από το πλήθος των όρων του ερωτήματος q που εμφανίζονται στο έγγραφο d
$queryNorm = \frac{1}{\sqrt{\sum_{t \in q}(idf(t) \cdot w_{t \in q})^2}}$	Παράγοντας κανονικοποίησης που σχετίζεται με τα βάρη w των όρων του q

3.4.1.2 Αλγόριθμος μηχανικής μάθησης

Επί των ταξινομημένων αποτελεσμάτων που επιστρέφει η αναζήτηση, εφαρμόζεται ο αλγόριθμος μηχανικής μάθησης του συστήματος, k-NN (k-Nearest Neighbor).

Επιλέγονται k το πλήθος έγγραφα με το μεγαλύτερο βαθμό ομοιότητας, τα οποία συνάμα είναι σημασιολογικά χαρακτηρισμένα από κόμβους της οντολογίας.

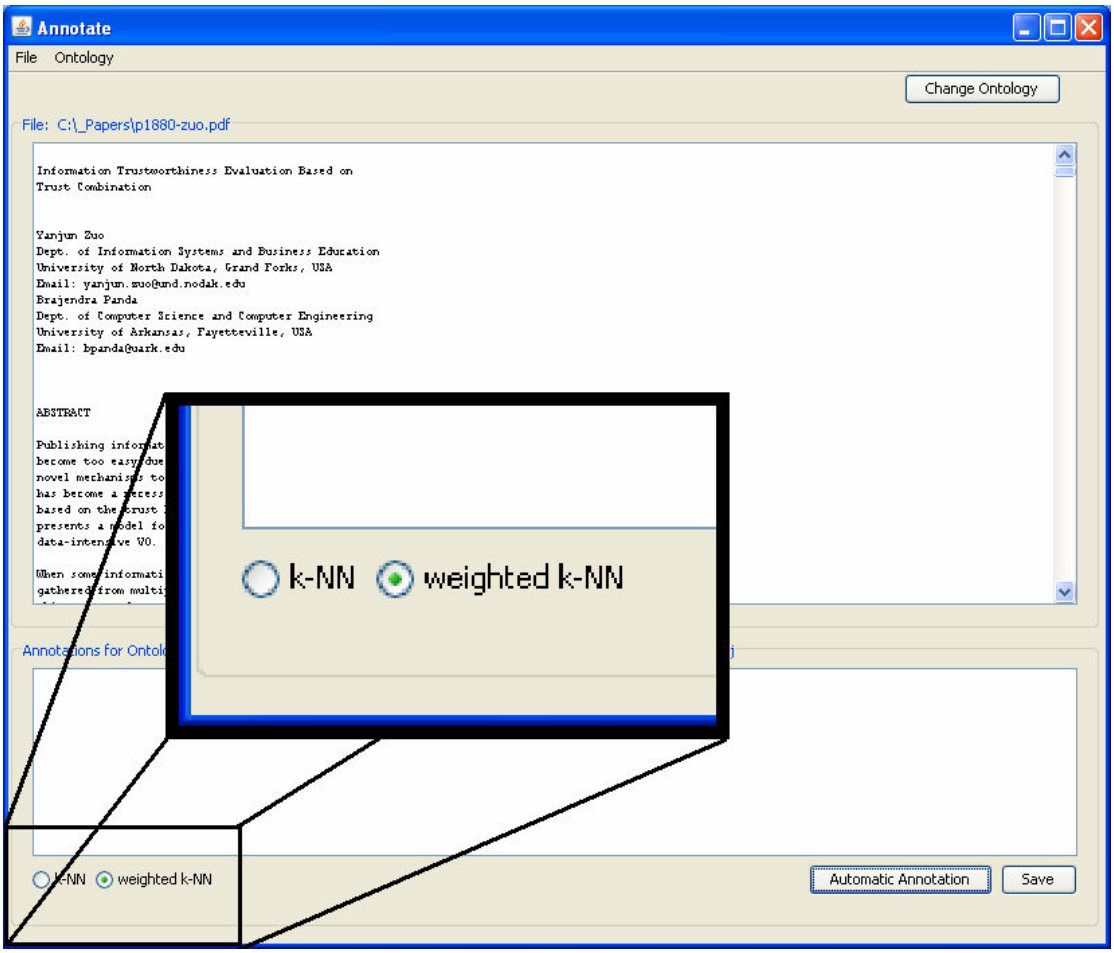
Στη συνέχεια, διεξάγεται ψηφοφορία μεταξύ των k επικρατέστερων εγγράφων για να εκλεγούν οι πλειοψηφίσαντες κόμβοι-κατηγορίες με τους οποίους είναι χαρακτηρισμένα τα έγγραφα αυτά.

Για την ψηφοφορία έχουν υλοποιηθεί δύο τρόποι:

- Απλός k-NN: Όλες οι ψήφοι είναι ισοβαρείς
- Σταθμισμένος (weighted) k-NN: Η βαρύτητα της ψήφου κάθε εγγράφου εξαρτάται από το βαθμό ομοιότητας του με το ερώτημα αναζήτησης, σύμφωνα με την ακόλουθη σχέση:

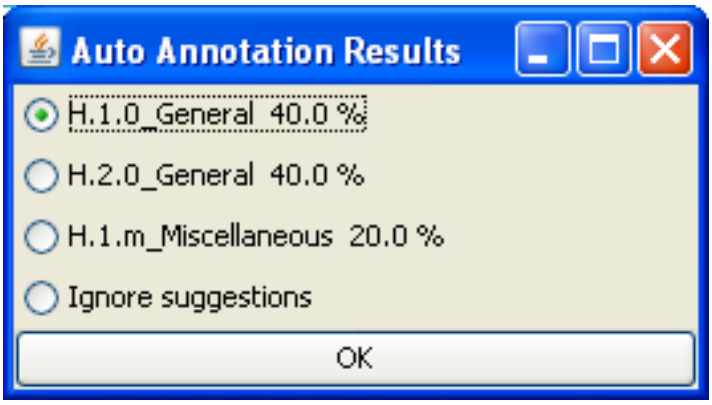
$$\text{Βάρος ψήφου εγγραφου (i)} = \frac{\text{Βαθμός ομοιότητας εγγράφου i}}{\text{Άθροισμα βαθμών ομοιότητας όλων των αποτελεσμάτων}}$$

Ένας έμπειρος χρήστης έχει τη δυνατότητα να επιλέξει το κανόνα ψηφοφορίας.

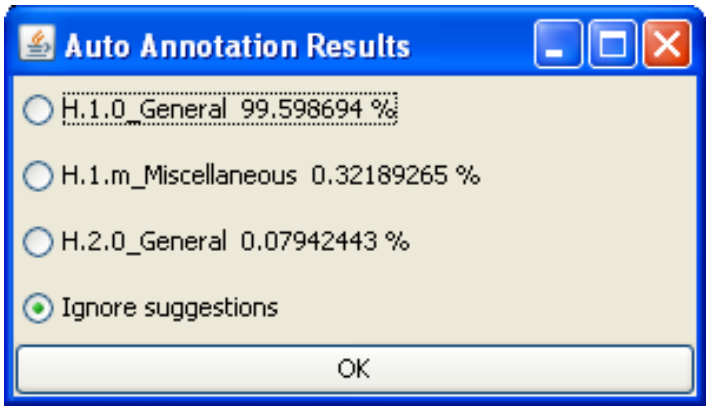


Σχήμα 16: Επιλογή τρόπου ψηφοφορίας

Οι m πλειοψηφίσαντες κόμβοι (όπου $m \leq k$) εμφανίζονται στο χρήστη σε φθίνουσα κατάταξη βάσει των σκορ τους (ή αλφαβητικά σε περίπτωση ισοψηφίας) όπως φαίνεται στα παρακάτω πλαίσια για τους δύο αλγορίθμους.

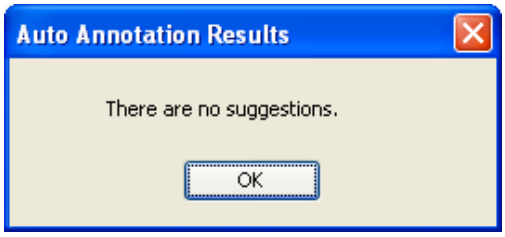


Σχήμα 17 Αποτελέσματα απλού k-NN



Σχήμα 18 Αποτελέσματα του Σταθμισμένου k-NN

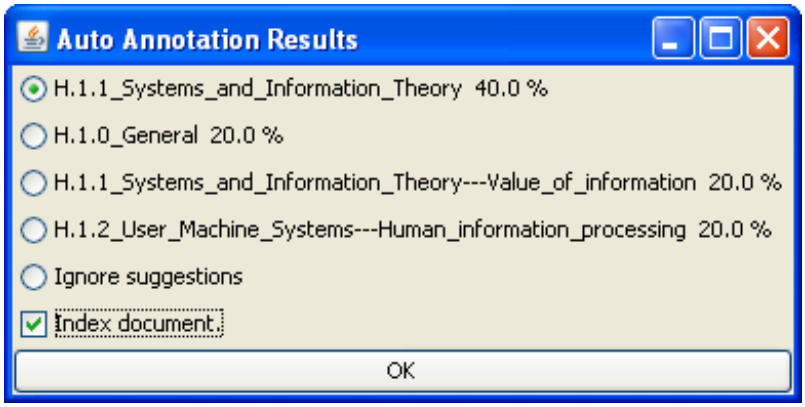
Αν δεν υπάρχουν προτεινόμενοι κόμβοι εμφανίζεται το ακόλουθο μήνυμα:



Σχήμα 19

3.4.1.3 Εκπαίδευση του συστήματος

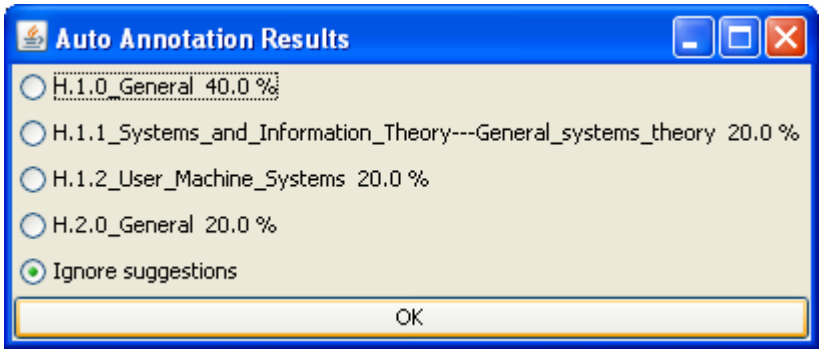
Ο χρήστης μπορεί είτε να επιλέξει μία από τις προτεινόμενες κατηγορίες για το χαρακτηρισμό του εγγράφου, ή να τις αγνοήσει. Στην περίπτωση που επιλέξει μία από τις κατηγορίες που του προτείνονται, και αν το έγγραφο δεν ανήκει στο ευρετήριο, ο χρήστης έχει την επιλογή να το προσθέσει στο ευρετήριο και άρα στο σύνολο των εγγράφων εκπαίδευσης του συστήματος, ενεργοποιώντας την αντίστοιχη επιλογή.



Σχήμα 20

Αν η διαδικασία του αυτόματου χαρακτηρισμού πραγματοποιηθεί για ένα χαρακτηρισμένο έγγραφο που συμμετέχει ήδη στα δεδομένα εκπαίδευσης, δεν εμφανίζεται στο πλαίσιο των

αποτελεσμάτων η επιλογή προσθήκης του στο ευρετήριο, όπως φαίνεται στο παρακάτω πλαίσιο:



Σχήμα 21 Auto Annotation σε ήδη χαρακτηρισμένο έγγραφο

Αν το σημασιολογικά χαρακτηρισμένο έγγραφο χαρακτηριστεί επιπλέον με νέο κόμβο, τότε αποτελεί έγγραφο εκπαίδευσης για κάθε κατηγορία στην οποία συμμετέχει.

3.4.1.4 Αυτόματος χαρακτηρισμός τμήματος του εγγράφου

Ο αυτόματος χαρακτηρισμός μπορεί να πραγματοποιηθεί για τμήμα ενός εγγράφου, επιλέγοντας το επιθυμητό τμήμα του κειμένου και πατώντας το κουμπί Auto Annotation.

Στην εκπαίδευση του συστήματος δε συμμετέχουν χαρακτηρισμένα τμήματα εγγράφων, αλλά μόνο ολόκληρα χαρακτηρισμένα έγγραφα.

3.4.2 Οντολογία ACM

Για τις ανάγκες του Auto Annotation, ήταν απαραίτητο να χρησιμοποιηθεί μία οντολογία, πρακτικού μεγέθους για το μέσο χρήστη του συστήματος, η οποία να παρέχει ευρεία κάλυψη μίας θεματικής περιοχής, και να μπορεί να αναπαριστά τόσο τις γενικότερες, όσο και τις πιο εξειδικευμένες έννοιες της περιοχής αυτής.

Επιπλέον, θα έπρεπε να είναι εύκολη η ανεύρεση επαρκούς πλήθους εγγράφων που θα αποτελούσαν τα σύνολα εκπαίδευσης και ελέγχου του συστήματος (training/testing data set).

3.4.2.1 Περιγραφή ACMOntology

Για την επίδειξη των δυνατοτήτων του αρχικού συστήματος GoNTogle, χρησιμοποιήθηκε μια μικρή οντολογία 19 κόμβων , που αφορούσε στη θεματική περιοχή των Βάσεων Δεδομένων, για την κατηγοριοποίηση των δημοσιεύσεων του Εργαστηρίου Συστημάτων Βάσεων Γνώσεων και Δεδομένων του ΕΜΠ.

Κινούμενοι λοιπόν στην ίδια θεματική περιοχή, κατασκευάσαμε την οντολογία ACMOntology, η οποία βασίστηκε στο σύστημα CCS (Computing Classification System) που αναπτύχθηκε από την ACM (Association for Computing Machinery) [ACM98], για την

κατηγοριοποίηση ερευνητικών κειμένων που ανήκουν στον ευρύτερο τομέα της Επιστήμης Υπολογιστών. Πρόκειται για μια δενδρική ιεραρχική κατηγοριοποίηση τεσσάρων επιπέδων, που αποτελείται από 11 βασικές κατηγορίες.

Βασικό πλεονέκτημα για την επιλογή της οντολογίας αυτής, ήταν το μεγάλο πλήθος ήδη κατηγοριοποιημένων εγγράφων, τα οποία αποτέλεσαν το σύνολο για την εκπαίδευση και τον έλεγχο του συστήματος. Τα έγγραφα αυτά βρίσκονται στην ψηφιακή βιβλιοθήκη ACM DL[2], και αποτελούνται από παραπομπές ή πλήρη κείμενα που δημοσιεύτηκαν σε περιοδικά της ACM, ενημερωτικά άρθρα και πρακτικά συνεδρίων.

3.4.2.2 Υλοποίηση ACMOntology

Για την κατασκευή της οντολογίας κατασκευάστηκε πρόγραμμα μετάφρασης του xml αρχείου της τελευταίας έκδοσης του ACM Computing Classification System (1998), κατά τέτοιο τρόπο ώστε να είναι εύκολη η μετατροπή και ενδεχόμενων μελλοντικών εκδόσεών του.

Η οντολογία που παράχθηκε είναι τύπου OWL, έχει μέγιστο βάθος κόμβων 4 και αποτελείται συνολικά από 1463 κόμβους. Επιπροσθέτως, η οντολογία εμπλουτίστηκε με τις παρακάτω σχέσεις για το συσχετισμό των στιγμιοτύπων της:

`IsExtendedVersionOf` ↔ `hasExtendedVersion`

`isAbstractOf` ↔ `has Abstract`

`isCitedBy` ↔ `cites`

`isRelatedTo`[3]

3.5 Υλοποίηση

Το GoNTogle 1.0 είναι υλοποιημένο σε γλώσσα προγραμματισμού Java, και κάνει χρήση αρκετών επιπλέον βιβλιοθηκών πχ Protégé, Lucene κλπ. Το GoNTogle 2.0 ως επέκταση του GoNTogle 1.0 έχει γραφτεί στο ίδιο περιβάλλον.

Ακολούθως περιγράφουμε σε υψηλό επίπεδο τις βασικές προσθήκες και επεκτάσεις που έγιναν στον κώδικα του GoNTogle 1.0, ώστε να υλοποιηθούν οι παραπάνω λειτουργικότητες.

---

[2] http://portal.acm.org/dl.cfm

[3] Η ομώνυμη ετικέτα που χρησιμοποιείται στο xml κείμενο του ACS CCS για το συσχετισμό παρόμοιων σημασιολογικά κατηγοριών της ιεραρχίας, μοντελοποιήθηκε με το annotation property `RelatedClasses`.

3.5.1 Κλάση AutomaticAnnotator

Πρόκειται για τη βασική κλάση που πραγματοποιεί τον αλγόριθμο κατηγοριοποίησης. Οι βασικότερες μέθοδοί της είναι οι ακόλουθες:

3.5.1.1 Μέθοδος annotate

Καλείται από την κλάση AutoAnnotatorResults με όρισμα το έγγραφο προς κατηγοριοποίηση και το κείμενό του και επιστρέφει τους προτεινόμενους κόμβους-κατηγορίες και το σκορ τους. Ακολούθως παρατίθενται τα βασικά τμήματα του κώδικα της μεθόδου, με σχόλια σχετικά με τη λειτουργία της:

```
static LinkedHashMap<OWLNamedClass,Float> annotate(String text, File
searchFile2){
```

Καλώντας την findSimilarDocs βρίσκουμε τα πιο όμοια έγγραφα με το σκορ τους και τα αποθηκεύουμε ταξινομημένα κατά σκορ στον πίνακα similarDocs. Στη συνέχεια διαμορφώνουμε το όνομα των εγγράφων σύμφωνα με τον τρόπο που αποθηκεύονται στα στιγμιότυπα στην οντολογία, καλώντας την μέθοδο fixPathName.

```
PathAndScore[] similarDocs = findSimilarDocs(text);
fixPathName(similarDocs);
```

Από το protégé, εντοπίζονται οι κατηγορίες στις οποίες συμμετέχει κάθε έγγραφο. Τα k καλύτερα έγγραφα, τα οποία όμως είναι σημασιολογικά χαρακτηρισμένα στην οντολογία, τοποθετούνται στον πίνακα similarDocClasses.

```
OWLNamedClass[] similarDocClasses = new OWLNamedClass[K];
LinkedHashMap<Integer,ClassAndScore> classAndScores =
                  new LinkedHashMap<Integer,ClassAndScore>();
OWLModel owlModel = (OWLModel) AnnotateJDialog.kb;
int m = 0;
for(int i=0;i<similarDocs.length;i++){
      OWLIndividual ind =
            owlModel.getOWLIndividualsimilarDocs[i].path);
```

```
String className = (String) ind.getPropertyValue(owlModel.

getOWLDatatypeProperty("rdfs:comment"));
className   =   className.substring(className.indexOf("\n\n")+2,
                                    className.indexOf("\n\n\n"));
similarDocClasses[m]=owlModel.getOWLNamedClass(className);
ClassAndScore cs = new ClassAndScore(owlModel.
            getOWLNamedClass(className),similarDocs[i].score);
classAndScores.put(m,cs);
m++;
  if( m < K ){
        continue;
 }else break;
}
```

Κατόπιν, καλείται η μέθοδος που πραγματοποιεί την ψηφοφορία, ανάλογα με την επιλογή του χρήστη (απλός k-NN ή σταθμισμένος k-NN), και επιστρέφονται στην κλάση AutoAnnotatorResults τα αποτελέσματα, δηλαδή οι επικρατέστεροι κόμβοι με το σκορ τους.

```
if(AnnotateJDialog.classifierFlag == 1){
      return weightedKNN(classAndScores);
} else
      return simpleKNN(classAndScores);
}
```

3.5.1.2 Μέθοδος findSimilarDocs

Καλείται από την μέθοδο annotate για να υπολογίσει το βαθμό ομοιότητας του εγγράφου προς κατηγοριοποίηση, με όλα τα έγγραφα εκπαίδευσης της οντολογίας. Η μέθοδος επιστρέφει τα έγγραφα ταξινομημένα κατά σκορ. Ακολουθούν τα σημαντικότερα μέρη του κώδικα σχολιασμένα.

```
public static PathAndScore[] findSimilarDocs(String queryText){
. . .
    try {
      IndexReader reader = IndexReader.open(index);
```

```
Searcher searcher = new IndexSearcher(reader);
Analyzer analyzer = new StandardAnalyzer();
```

Κατασκευάζεται το vector του προς κατηγοριοποίηση κειμένου, αφού υποβληθεί λεξικολογική ανάλυση από το Lucene. Η μεταβλητή tf είναι τύπου HashMap και περιέχει τους όρους (tokens) του κειμένου, με τη συχνότητα εμφάνισής τους.

```
HashMap<String,Integer> tf = new HashMap<String,Integer>();
TokenStream tokens = analyzer.tokenStream("contents",new
                                        StringReader(queryText));
while(true){
    Token t = tokens.next();
    if(t==null) break;
        String term = t.termText();
    if(tf.containsKey(term)){
        tf.put(term,tf.get(term)+1);
    }else {
        tf.put(term,1);
    }
}
```

Στη συνέχεια δημιουργείται το ερώτημα query για την ανάκτηση των εγγράφων κατά βαθμό ομοιότητας. Κάθε όρος του vector εισάγεται στο query, έχοντας ως βάρος τη συχνότητα εμφάνισής του.

```
BooleanQuery query= new BooleanQuery();
for(Entry<String,Integer> term: tf.entrySet()){
      TermQuery tq = new TermQuery(newTerm
                        ("contents",term.getKey()));
      tq.setBoost(term.getValue());
      query.add(tq,false,false);
 }
```

Ακολούθως, εκτελείται το ερώτημα και τα αποτελέσματα ταξινομούνται και επιστρέφονται. Αν δεν υπάρχουν αποτελέσματα, επιστρέφεται κενός πίνακας.

```
        Hits hits = searcher.search(query);
        PathAndScore[] result = new PathAndScore[hits.length()];
        for (int i = 0; i < hits.length(); i++) {
              float score = hits.score(i);
              String path = hits.doc(i).get("path");
              result[i]=new PathAndScore(path, score);
         }
. . .
         return result;
    } catch (Exception e){
          e.printStackTrace();
          return new PathAndScore[0];
    }
}
```

3.5.1.3 Μέθοδος simpleKNN

Η μέθοδος αυτή πραγματοποιεί την ψηφοφορία σύμφωνα με τον κλασσικό τρόπου του αλγόριθμου k-NN. Οι k επικρατέστεροι κόμβοι ψηφίζουν για να βρεθούν ποιοι πλειοψηφούν. Επιστρέφονται οι $m \leq k$ πλειοψηφίσαντες κόμβοι, με το ποσοστό των ψήφων που έλαβαν επί τοις εκατό.

3.5.1.4 Μέθοδος weightedKNN

Η μέθοδος αυτή πραγματοποιεί ψηφοφορία, στην οποία κάθε ψήφος βαραίνει ανάλογα με το βαθμό ομοιότητας των k επικρατέστερων εγγράφων, με το προς κατηγοριοποίηση έγγραφο. Η ψήφος κάθε κόμβου ισούται με το πηλίκο του βαθμού ομοιότητας δια του αθροίσματος των βαθμών ομοιότητας των k κόμβων. Επιστρέφονται οι $m \leq k$ κόμβοι, με το ποσοστιαίο επί τοις εκατό σκορ που προέκυψε από την ψηφοφορία.

3.5.2 Κλάση AutoAnnotatorResults

Η κλάση αυτή δέχεται και εμφανίζει τα αποτελέσματα του αλγορίθμου κατηγοριοποίησης, και ανάλογα με τις επιλογές του χρήστη πραγματοποιεί σημασιολογικό χαρακτηρισμό του προς κατηγοριοποίηση κειμένου με τον επιλεχθέντα κόμβο, καθώς και ευρετηριοποίησή του. Τα βασικά μέρη του κώδικα παρουσιάζονται σχολιασμένα ακολούθως:

3.5.2.1 Μέθοδος annotateText

Η μέθοδος αυτή καλέι την μέθοδο annotate της κλάσης AutomaticAnnotator, με όρισμα το προς κατηγοριοποίηση έγγραφο και της επιστρέφονται οι επικρατέστεροι κόμβοι της οντολογίας με το σκορ τους.

```
public static void annotateText(String text, File searchFile1,
AnnotateJDialog currentWindow){
    searchFile = searchFile1;
     LinkedHashMap<OWLNamedClass,Float> annotatorSuggestions =
                        AutomaticAnnotator.annotate(text,searchFile);
     String[] classNamesAndScores = new
                                  String annotatorSuggestions.size()];
     OWLNamedClass[] suggestedClasses = new
                          OWLNamedClass[annotatorSuggestions.size()];
     int i=0;
     for(Entry<OWLNamedClass,Float> entry :
                                    annotatorSuggestions.entrySet()){
         if(entry != null){
             classNamesAndScores[i]= entry.getKey().getName() +"   "+
                                              entry.getValue() +" %";
             suggestedClasses[i]= entry.getKey();
             i++;
         }
     }
```

Καλείται η μέθοδος checkIfIndexed που ελέγχει αν το έγγραφο είναι ευρετηριοποιημένο, ώστε να ενεργοποιήσει την εμφάνιση της επιλογής για ευρετηριοποίηση:

```
     checkIfIndexed(searchFile1);
```

Στη συνέχεια καλείται η μέθοδος getChoice η οποία θα δημιουργήσει τη διαπροσωπεία με τα αποτελέσμα (παράθυρο AutoAnnotationResults). Η μέθοδος formatStringArray διαμορφώνει τη σειρά εμφάνισης των αποτελεσμάτων, ώστε οι ισοψηφίσαντες κόμβοι να εμφανίζονται κατά αλφαβητική σειρά.

```
    getChoice(formatStringArray(classNamesAndScores),
                                    suggestedClasses,currentWindow);
}
```

3.5.2.2 Μέθοδος getChoice

Η μέθοδος αυτή εμφανίζει τα προτεινόμενα αποτελέσματα στο χρήστη, σε ξεχωριστό παράθυρο. Όταν ο χρήστης επιλέξει κάποιο απ' αυτά (ή κανένα), η αρμόδια μέθοδος χειρισμού γεγονότων θα αναλάβει να πραγματοποιήσει τον ζητούμενο σημασιολογικό χαρακτηρισμό. Τα σημαντικότερα τμήματα του κώδικα είναι τα ακόλουθα:

```
public static void getChoice(final String[] suggestions1, final
OWLNamedClass[] suggestionNodes,final AnnotateJDialog currentWindow)
{
```

Αν δεν υπάρχουν προτάσεις ή το έγγραφο δεν περιείχε κείμενο, ο χρήστης ενημερώνεται κατάλληλα, και επανενεργοποιείται η δυνατότητα χρήσης του κουμπιού αυτόματου σημασιολογικού χαρακτηρισμού

```
        if(suggestions1.length == 0){
            JOptionPane.showMessageDialog(null," There are no
suggestions.","Auto Annotation Results",JOptionPane.PLAIN_MESSAGE);
            AnnotateJDialog.autoAnnotationFlag = 0;
        }else{
```

Διαφορετικά, εμφανίζεται το παράθυρο.

```
      Runnable a = new Runnable() {
                public void run() {
                    JFrame frame = new JFrame("Auto Annotation
Results");
                    JComponent newContentPane = new
AutoAnnotatorResults ( suggestions1, suggestionNodes,
frame,currentWindow);
                    newContentPane.setOpaque(true);
                    frame.setContentPane(newContentPane);
```

```
                frame.pack();
                frame.setVisible(true);
            }
        };
        javax.swing.SwingUtilities.invokeLater(a);
    }
}
```

3.5.3 Κλάση AnnotateJDialog

Η κλάση αυτή είναι η βασική κλάση του GoNTogle 1.0 που αφορά το σημασιολογικό χαρακτηρισμό κειμένων και την επεξεργασία οντολογιών. Σ' αυτή την κλάση προστέθηκαν μέθοδοι που αφορούν τον αυτόματο σημασιολογικό χαρακτηρισμό.

3.5.3.1 Μέθοδος autoAnnotButtonActionPerformed

Η μέθοδος αυτή καλείται όταν πατηθεί το κουμπί "Automatic Annotation". Εξασφαλίζει ότι έχει φορτωθεί οντολογία, έχει ανοιχτεί έγγραφο και ότι μόνο μία ενέργεια Auto Annotation λαμβάνει χώρα κάθε χρονική στιγμή. Επίσης καλεί τη συνάρτηση annotateText της κλάσης AutoAnnotatorResults με όρισμα το κείμενο του εγγράφου που πρόκειται να χαρακτηριστεί (ολόκληρο ή τμήμα του).

3.5.3.2 Μέθοδος saveAnnotation

Αυτή η μέθοδος χρησιμοποιείται για να χαρακτηρίσει σημασιολογικά το τρέχον έγγραφο με τον κόμβο που επιλέχθηκε από το Auto Annotation.

Αν επιχειρηθεί η διαδικασία Auto Annotation σε έγγραφο που είναι ήδη χαρακτηρισμένο, εμφανίζεται μήνυμα λάθους "The selected text is already annotated."

4

Αναζήτηση σημασιολογικής γειτονίας

4.1 Περιγραφή

Συχνά σε αναζητήσεις, ανεξαρτήτως φύσης, επιστρέφεται στο χρήστη μη επαρκής αριθμός αποτελεσμάτων. Αυτό μπορεί να οφείλεται, π.χ., σε υπερβολικά περιοριστικό ερώτημα του χρήστη, και έχει ως επακόλουθο ο χρήστης να αδυνατεί να εντοπίσει κάποιο σχετικό έγγραφο μέσα στα αποτελέσματα.

Για την αντιμετώπιση αυτού του προβλήματος, προτείνουμε τεχνικές που επεκτείνουν ένα ερώτημα, επεκτείνοντάς το στον σημασιολογικό άξονα. Συγκεκριμένα, με τον όρο «Αναζήτηση σημασιολογικής γειτονίας», αναφερόμαστε σε δύο νέες δυνατότητες αναζήτησης που υλοποιήθηκαν, για τον εμπλουτισμό των αποτελεσμάτων της απλής σημασιολογικής αναζήτησης, γενικεύοντας το οντολογικό κομμάτι του ερωτήματος, αναζητώντας έγγραφα συγγενών. Η πρώτη δυνατότητα επιτρέπει προοδευτική ανάκτηση εγγράφων, από κόμβους της οντολογίας που βρίσκονται στο αμέσως επόμενο βάθος σε σχέση με τους κόμβους αρχικής αναζήτησης, ενώ η δεύτερη πραγματοποιεί αναζήτηση στους εγγύτερους απογόνους των κόμβων αυτών.

4.2 Θεωρητικό υπόβαθρο

Το πρόβλημα της σημασιολογικής επέκτασης ενός ερωτήματος σε γειτονία κόμβων σχετίζεται με το γνωστό πρόβλημα αναζήτησης εγγύτητας σε γράφους (proximity search on

graphs) [LCV+01], [KSI+08], [GSW05]. Ο στόχος της αναζήτησης εγγύτητας σε γράφους είναι ο εντοπισμός ενός συνόλου "κοντινών" κόμβων ενός γράφου, που ως σύνολο πληρούν τα κριτήρια μιας αναζήτησης (όταν, για παράδειγμα, ο αριθμός των μεμονωμένων κόμβων που πληρούν τα κριτήρια αναμένεται μικρός). Μολονότι η αναζήτηση σημασιολογικής γειτονίας είναι πρόβλημα διαφορετικό, η υπάρχουσα βιβλιογραφία αναζήτησης εγγύτητας σε γράφους έχει να προτείνει δύο μοντέλα που μπορούν να χρησιμοποιηθούν επιτυχώς στην αναζήτηση σημασιολογικής γειτονίας, και τους οποίους περιγράφουμε ακολούθως.

4.2.1 Μοντέλο σταδιακής εξάπλωσης

Στο [GSW05] προτείνεται ένα σύστημα αναζήτησης εγγράφων, δομημένων ως γράφων. Τα αποτελέσματα μιας αναζήτησης ανακτώνται και αξιολογούνται ανά γενιές, με βάση την απόστασή τους από κόμβους που σχετίζονται με το ερώτημα της αναζήτησης. Η διαίσθηση πίσω απ' αυτό το μοντέλο είναι ότι κόμβοι της ίδιας γενιάς (οι έχοντες, δηλαδή, την ίδια απόσταση από τους κόμβους του ερωτήματος) έχουν παρόμοια σχετικότητα με το ερώτημα. Υιοθετώντας τη διαίσθηση αυτή, ακολούθως προτείνουμε την Αναζήτηση Επόμενης Γενιάς.

4.2.2 Μοντέλο περιορισμένης εξαπλούμενης ενεργοποίησης

Το μοντέλο περιορισμένης εξαπλούμενης ενεργοποίησης (constrained spreading activation) [CK87] έχει χρησιμοποιηθεί ευρέως στην αναζήτηση εγγύτητας σε γράφους [KSI+08], [LCV+01]. Σε αυτό το μοντέλο, θεωρείται ότι οι κόμβοι ενός γράφου που σχετίζονται με το ερώτημα μιας αναζήτησης λαμβάνουν μια μονάδα "ενεργοποίησης". Ακολούθως, με βάση κάποιους κανόνες που διέπουν την μετάδοσή της, η ενεργοποίηση μεταδίδεται στους υπόλοιπους κόμβους του γράφου. Για παράδειγμα, μπορεί ο κάθε κόμβος να χάνει ένα ποσοστό μ, της ενεργοποίησής του, το οποίο ισοκατανέμεται σε όλους τους γειτονικούς του κόμβους, και η τελική ενεργοποίηση ενός κόμβου είναι το μέγιστο όλων των ενεργοποιήσεων που έχει λάβει. Το τελικό σκορ ενός κόμβου είναι η τελική ενεργοποίησή του. Επιπλέον, για να υπάρχει μια ελάχιστη εγγυημένη σχετικότητα του κάθε αποτελέσματος με το ερώτημα, κόμβοι με ενεργοποίηση κάτω από ένα κατώφλι, δεν την μεταδίδουν. Διάφοροι άλλοι κανόνες μπορούν να χρησιμοποιηθούν, ο καθένας οδηγώντας σε διαφορετική σημασιολογία. Εμείς ακολούθως προτείνουμε την Αναζήτηση Στενής Γειτονίας, υιοθετώντας μια υλοποίηση του μοντέλου όπου:

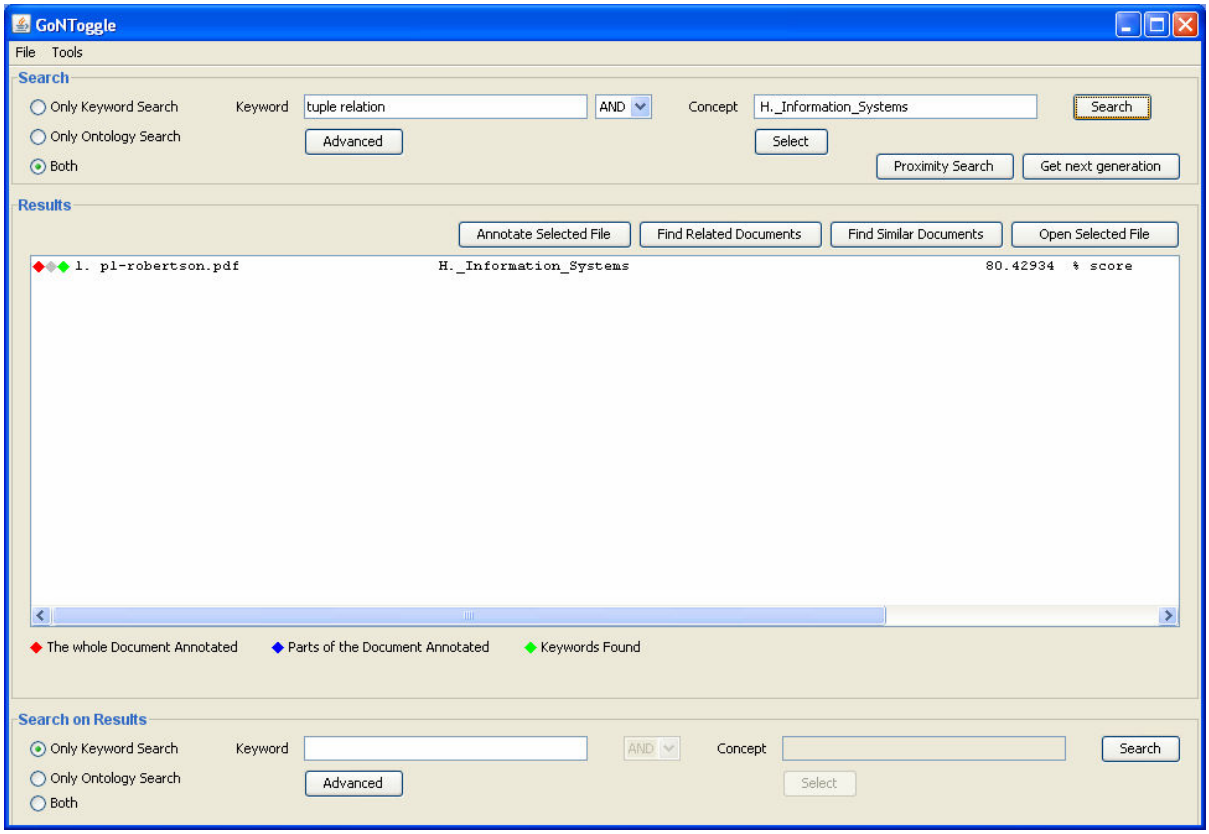
- κάθε κόμβος της οντολογίας που είναι σχετικός με το σημασιολογικό ερώτημα λαμβάνει μοναδιαία ενεργοποίηση
- κάθε κόμβος μεταδίδει ένα ποσοστό μ της ενεργοποίησής του σε **κάθε** κόμβο-παιδί του

- η τελική ενεργοποίηση ενός κόμβου είναι το μέγιστο όλων των ενεργοποιήσεων που έχει λάβει[4], και
- ενεργοποίηση κάτω του κατωφλίου μ^2 δεν μεταδίδεται (αυτό αντιστοιχεί στις δύο πρώτες γενιές, με βάση το μοντέλο σταδιακής εξάπλωσης - ο αριθμός αυτός επιλέχθηκε έπειτα από εμπειρικές παρατηρήσεις).

4.3 Ανάλυση και σχεδίαση

Η αναζήτηση σημασιολογικής γειτονίας υλοποιήθηκε για να εμπλουτίσει τα αποτελέσματα μιας απλής σημασιολογικής αναζήτησης, είτε στην περίπτωση που τα αρχικά αποτελέσματα δεν ικανοποιούν το χρήστη, οπότε γενικεύει επικεντρώνοντας την έρευνά του στους εγγύτερους απογόνους των κόμβων αρχικής αναζήτησης (Αναζήτηση Στενής Γειτονίας), είτε όταν ο χρήστης επιθυμεί να ανακτήσει έγγραφα κατά φθίνουσα σειρά σχετικότητας ως προς το βάθος κατανομής των κόμβων (Αναζήτηση Επόμενης Γενιάς).

Ας δούμε ένα παράδειγμα απλής αναζήτησης και τα αποτελέσματα που προκύπτουν από τις δύο αυτές υλοποιήσεις.



Σχήμα 22 Αποτελέσματα απλής σημασιολογικής αναζήτησης

[4] Στην περίπτωσή μας όπου ο γράφος της οντολογίας είναι δέντρο, ούτως ή άλλως κάθε κόμβος μπορεί να λάβει ενεργοποίηση το πολύ από έναν άλλο κόμβο (πατρικό κόμβο).

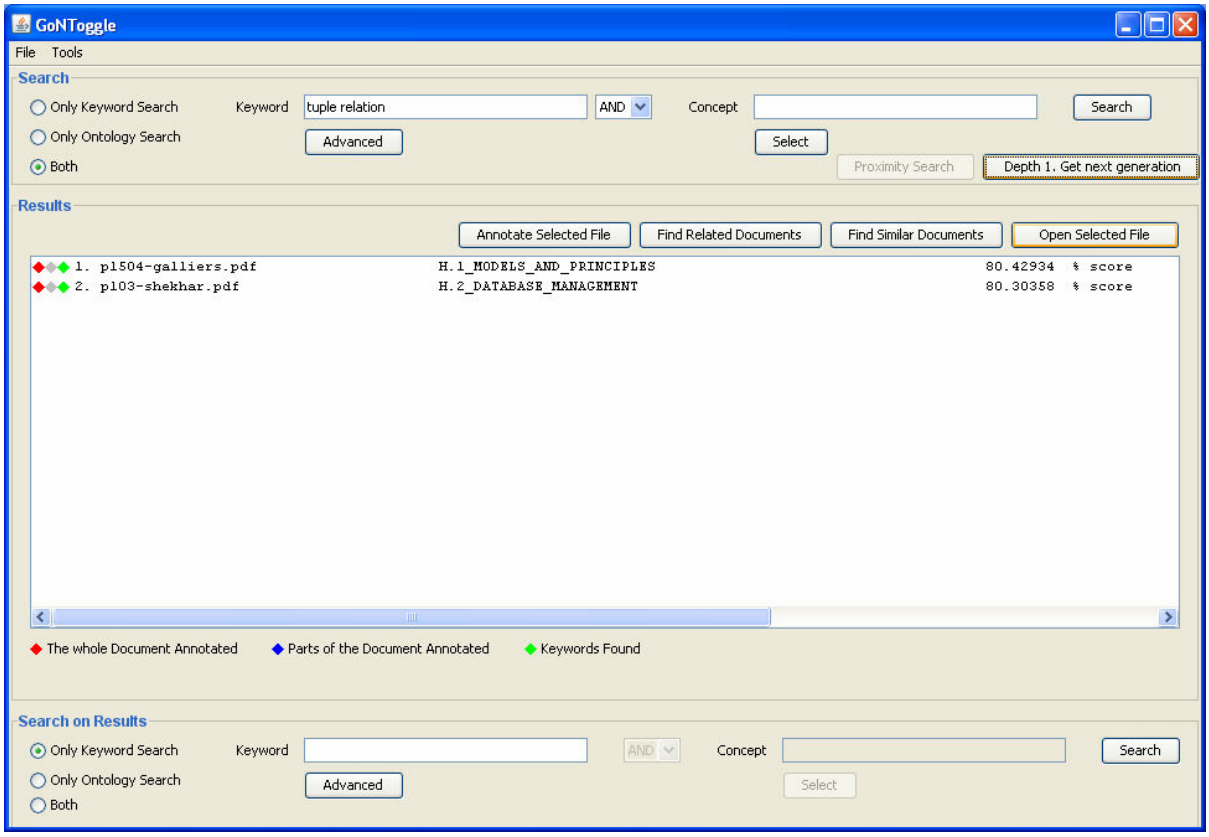
4.3.1 Αναζήτηση Επόμενης Γενιάς

Ο χρήστης επιλέγει την Αναζήτηση Επόμενης Γενιάς (ΑΕΓ) όταν ψάχνει έγγραφα που ναι μεν προσεγγίζουν εννοιολογικά τις κατηγορίες αρχικής αναζήτησης, αλλά απέχουν σαφώς συγκεκριμένο βάθος από τους αρχικούς κόμβους.

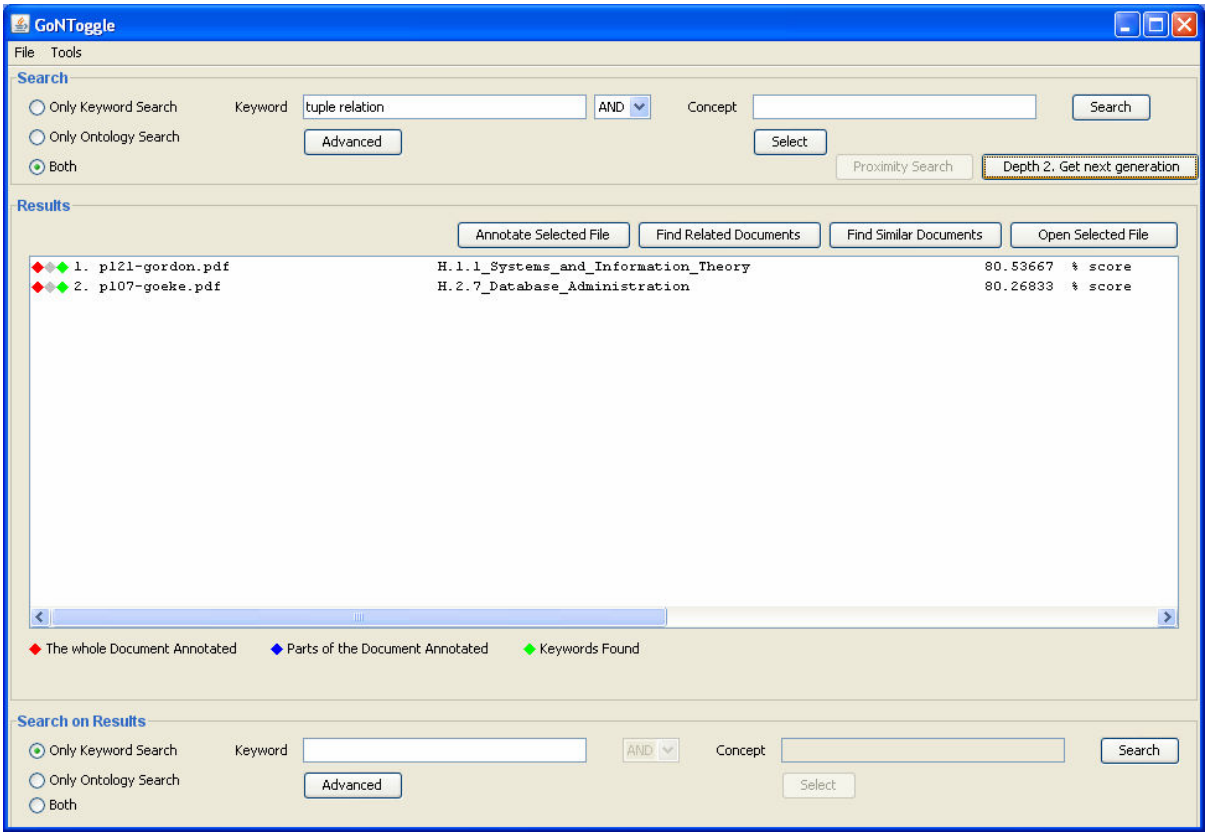
Η πραγματοποίηση της ΑΕΓ γίνεται πατώντας το κουμπί "Get Next Generation", το οποίο ενεργοποιείται μετά από απλή σημασιολογική αναζήτηση. Το πάτημα του κουμπιού αυτού έχει ως αποτέλεσμα την ανάκτηση εγγράφων από όλους τους κόμβους της οντολογίας που είναι απόγονοι πρώτου βαθμού συγγένειας των αρχικών κόμβων αναζήτησης. Κάθε νέο πάτημα το κουμπιού "Get Next Generation", επαναλαμβάνει την ίδια διαδικασία θέτοντας ως κόμβους αναζήτησης τους κόμβους-παιδιά της προηγούμενης, μέχρις ότου να μην υπάρχουν άλλοι απόγονοι.

Όσο περισσότεροι κόμβοι απ' αυτούς που συμμετέχουν στο τρέχον βήμα της ΑΕΓ χαρακτηρίζουν σημασιολογικά ένα έγγραφο (και ανάλογα φυσικά και με το μέγεθος και το ποσοστό χαρακτηρισμού του κειμένου), τόσο θεωρούμε πιο γενικό το περιεχόμενο του εγγράφου αυτού και άρα πιο πιθανό να προσεγγίζει νοηματικά τους πατρικούς κόμβους αναζήτησης (δηλαδή τους κόμβους αναζήτησης του προηγούμενου βήματος).

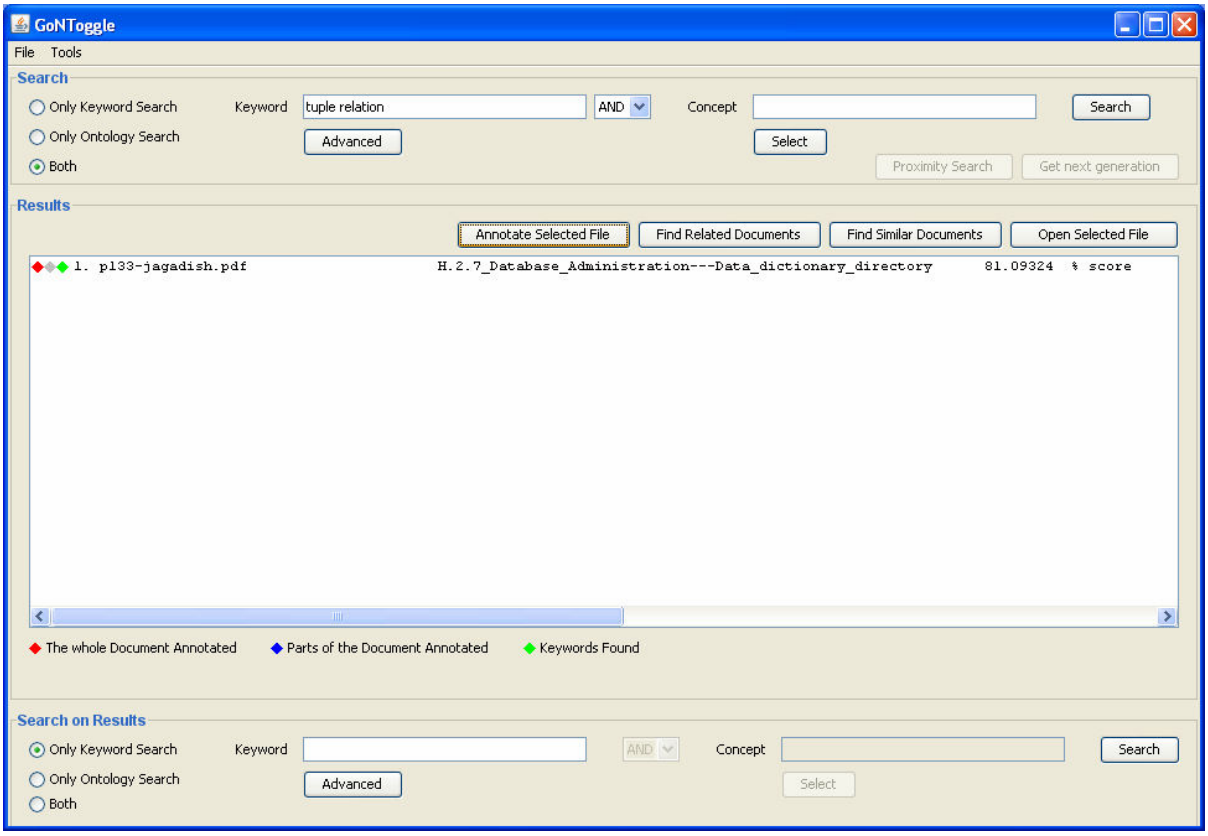
Ακολουθούν τα αποτελέσματα που προκύπτουν από την εφαρμογή της ΑΕΓ, μετά από την απλή αναζήτηση που αναφέρθηκε παραπάνω:



Σχήμα 23 Αποτελέσματα πρώτης γενιάς



Σχήμα 24 Αποτελέσματα δεύτερης γενιάς



Σχήμα 25 Αποτελέσματα τρίτης γενιάς (δεν υπάρχουν άλλες γενιές)

4.3.2 Αναζήτηση Στενής Γειτονίας

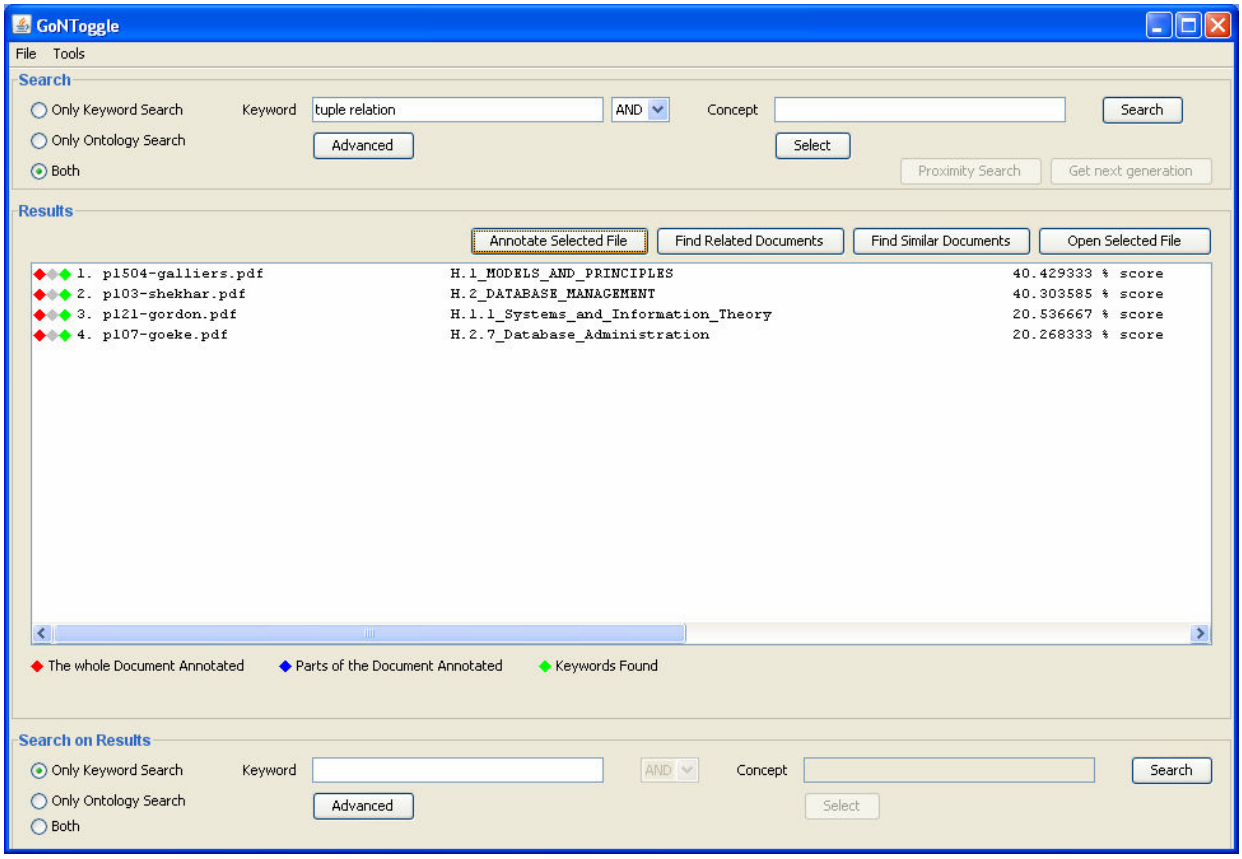
Στην Αναζήτηση Στενής Γειτονίας (ΑΣΓ) καταφεύγει ο χρήστης στην περίπτωση που τον ενδιαφέρει να βρει έγγραφα όσο τον δυνατόν εννοιολογικά πλησιέστερα στις αρχικές κατηγορίες αναζήτησης. Επειδή το πιθανότερο είναι τα έγγραφα αυτά να βρίσκονται στο κοντινό περιβάλλον των αρχικών κόμβων αναζήτησης, εξετάζουμε τους κόμβους πρώτης και δεύτερης γενιάς (παιδιά και εγγόνια). Δεν επεκτείνουμε την έρευνα σε έγγραφα που ανήκουν σε γενιές κόμβων μεγαλύτερες από τη δεύτερη, γιατί θεωρούμε ότι απομακρύνονται σημασιολογικά αρκετά από τις αρχικές κατηγορίες.

Η ΑΣΓ ξεκινά με το πάτημα του κουμπιού "Proximity Search", που και σ' αυτή την περίπτωση ενεργοποιείται μετά από απλή αναζήτηση στην οντολογία. Τα αποτελέσματα που εμφανίζονται στο χρήστη, προκύπτουν από την ακόλουθη διαδικασία:

Αρχικά πραγματοποιείται αναζήτηση στους κόμβους πρώτης γενιάς και μετά γίνεται αναζήτηση στους κόμβους δεύτερης γενιάς. Στη συνέχεια, υποδιπλασιάζεται το σκορ των εγγράφων-παιδιών, υποτετραπλασιάζεται το σκορ των εγγράφων-εγγονών, πραγματοποιείται η ένωση των δύο συνόλων αποτελεσμάτων και τα τελικά αποτελέσματα εμφανίζονται στο χρήστη ταξινομημένα κατά σκορ.

Η βαθμολόγηση των αποτελεσμάτων γίνεται κατά τέτοιο τρόπο, ώστε να πριμοδοτούνται τα έγγραφα-παιδιά (ως πιο σχετικά), για να προηγούνται των εγγράφων-εγγονών στην κατάταξη. Είναι βέβαια πιθανό να δούμε έγγραφα-εγγόνια να εμφανίζονται πριν από έγγραφα- παιδιά. Αυτό είναι επιθυμητό και αναμενόμενο. Πχ αν ένα έγγραφο Α είναι χαρακτηρισμένο κατά 1% με ένα κόμβο παιδί και κατά 99% με έναν άσχετο κόμβο, ενώ ένα έγγραφο Β κατά 100% με ένα κόμβο εγγόνι, τότε το έγγραφο Β προσεγγίζει περισσότερο εννοιολογικά τους αρχικούς κόμβους αναζήτησης, άρα πρέπει να εμφανίζεται πριν από το Α στην κατάταξη. Τέλος, αν ένα έγγραφο εμφανίζεται και στα δύο σύνολα (δηλαδή είναι χαρακτηρισμένο με κόμβους πρώτης και δεύτερης γενιάς), τότε εμφανίζεται στα αποτελέσματα με το μεγαλύτερο εκ των δύο σκορ.

Η εφαρμογή της ΑΣΓ μετά την προαναφερθείσα απλή αναζήτηση, δίνει τα ακόλουθα αποτελέσματα:



Σχήμα 26 Αποτελέσματα ΑΣΓ

4.4 Υλοποίηση

Ακολούθως περιγράφουμε σε υψηλό επίπεδο τις βασικές προσθήκες και επεκτάσεις που έγιναν στον κώδικα του GoNTogle, ώστε να υλοποιηθούν οι παραπάνω λειτουργικότητες.

4.4.1 Κλάση NewJFrame

4.4.1.1 Μέθοδος proximitySearchActionPerformed

Η μέθοδος αυτή καλείται όταν πατηθεί το κουμπί “Proximity Search”, το οποίο ενεργοποιείται μετά από απλή σημασιολογική αναζήτηση ή συνδυασμένη αναζήτηση. Παρατίθενται τα σημαντικότερα τμήματα του κώδικα για αναζήτηση στην οντολογία:

```
private void proximitySearchActionPerformed(
                                        java.awt.event.ActionEvent evt) {
. . .
```

Αρχικά απενεργοποιείται η δυνατότητα ΑΕΓ και στη συνέχεια καλείται η σημασιολογική αναζήτηση καλώντας τη μέθοδο navigateOntologyProxSearch, αναζητώντας αποτελέσματα

στους κόμβους-παιδιά των κόμβων της αρχικής απλής αναζήτησης (aClsNextGenerNProximity):

```
nextGenerationButton.setEnabled(false);
navigateOntologyProxSearch(aClsNextGenerNProximity);
. . .
```

Τα τρέχοντα αποτελέσματα (currentData) που επιστρέφονται με το σκορ τους, αποθηκεύονται στο Vector currentDataChildren.

```
Vector currentDataChildren = new Vector();
 currentDataChildren.addAll(currentData);
```

Στη συνέχεια, αποθηκεύονται στο vector currentSearchClasses οι τρέχοντες κόμβοι-παιδιά και κόμβοι εγγόνια, ώστε να χρησιμοποιηθούν από τον renderer για την εμφανιση των αποτελεσμάτων.

```
currentSearchClasses.clear();
currentSearchClasses.addAll(children);
currentSearchClasses.addAll(grandChildren);
```

Ακολούθως, ξανακαλείται η μέθοδος navigateOntologyProxSearch αναζητώντας αποτελέσματα στους κόμβους-παιδιά των κόμβων της προηγούμενης αναζήτησης, δηλαδή αναζητώντας αποτελέσματα στους κόμβους- εγγόνια της αρχικής αναζήτησης. Τα αποτελέσματα, αφού διαμορφωθεί το σκορ τους τοποθετούνται στο vector currentDataGrandchildren.

```
navigateOntologyProxSearch(children);
currentData.removeAllElements();
. . .
Vector currentDataGrandchildren  = new Vector();
currentDataGrandchildren.addAll(currentData);
```

Υποδιπλασιάζεται το σκορ των παιδιών, ενώ υποτετραπλασιάζεται το σκορ των εγγονών. Στη συνέχεια εξετάζεται αν υπάρχει κάποιο αποτέλεσμα εγγράφου και στις δύο λίστες και αν ναι, το σκορ του αποτελέσματος με το μικρότερο σκορ τίθεται ίσο με μηδέν.

```
for(int k=0; k<currentDataChildren.size(); k++){
      Object value[] = (Object []) currentDataChildren.get(k);
      value[12] = ((Float)value[12])/2;
      currentDataChildren.set(k,value);
}
for(int k=0; k<currentDataGrandchildren.size(); k++){
      Object value[]=(Object[])currentDataGrandchildren.get(k);
      value[12] = ((Float)value[12])/4;
      currentDataGrandchildren.set(k,value);
}
for(int c=0; c<currentDataChildren.size(); c++){
      Object valueC[] = (Object []) currentDataChildren.get(c);
      for(int g=0; g<currentDataGrandchildren.size(); g++){
            Object valueG[] = (Object [])
                              currentDataGrandchildren.get(g);
             if(valueC[0].equals(valueG[0])){
                  if((Float)valueC[12] > (Float)valueG[12]){
                        valueG[12] = new Float(0);
                  }else{
                        valueC[12] = new Float(0);
                  }
            }
      }
}
```

Τα δυο σύνολα συνενώνονται στο currentData. Από αυτό, αφαιρούνται τα αποτελέσματα με μηδενικό σκορ, ενώ τα υπόλοιπα διατάσσονται και αποκτούν αύξουσα αρίθμηση:

```
currentData.clear();
currentData.addAll(currentDataChildren);
currentData.addAll(currentDataGrandchildren);
ArrayList temp = new ArrayList();
for(Object o : currentData){
      Object value[] = (Object [])o;
```

```
            if((Float)value[12] != 0){
                  temp.add(o);
            }
      }
      currentData.clear();
      currentData.addAll(temp);
      for (int m = 0; m < currentData.size() - 1; m++){
            for (int n = m; n < currentData.size(); n++) {
                  Object value1[] = (Object []) currentData.get(m);
                  Object value2[] = (Object []) currentData.get(n);
                  if (((Float) value1[12]).floatValue() <
                                    ((Float) value2[12]).floatValue()){
                        currentData.set(m, value2);
                        currentData.set(n, value1);
                  }
            }
      }
      for (int m = 0; m < currentData.size(); m++){
            Object value[] = (Object []) currentData.get(m);
            value[8] = new Integer(m + 1);
            currentData.set(m, value);
      }
```

Τέλος τα τελικά αποτελέσματα παρουσιάζονται στο χρήστη και απενεργοποιείται το κουμπί "Proximity Search".

```
      jList2.setListData(currentData);
      ListCellRenderer renderer = new ComplexCellRenderer();
      jList2.setCellRenderer(renderer);
   }
}
proximitySearchButton.setEnabled(false);
```

Στην περίπτωση της συνδυαστικής αναζήτησης και ανάλογα με το αν εχει επιλεγεί σύζευξη ή διάζευξη των αποτελεσμάτων των δύο αναζητήσεων (αναζήτηση με λέξεις και αναζήτηση

στην οντολογία), δημιουργείται η τομή ή η ένωση των αποτελεσμάτων με κατάλληλη διαμόρφωση του τελικού σκορ και της τελικής κατάταξης των αποτελεσμάτων

4.4.1.2 Μέθοδος navigateOntologyProxSearch

Η μέθοδος αυτή καλείται από την proximitySearchActionPerformed, με όρισμα αναζήτησης τα αποτελέσματα της προηγούμενης αναζήτησης (αρχικών κόμβων ή κόμβων παιδιών των αρχικών) και αναζητά αποτελέσματα των κόμβων παιδιών του ορίσματός της. Στις static μεταβλητές children και grandchildren, απόθηκεύονται οι ομώνυμοι τρέχοντες κόμβοι.

```
private void navigateOntologyProxSearch(ArrayList prevClas) {
      ArrayList previousClasses = new ArrayList();
      previousClasses.addAll(prevClas);
      children.clear();
      grandChildren.clear();
```

Για κάθε κλάση prevClas, αποθηκεύονται οι υποκλάσεις της, βάθους ένα, στη συλλογή rdfSubClCollection1 και στο vector children. Ελέγχεται αν υπάρχουν υποκλάσεις βάθους δύο (εγγόνια) και αν υπάρχουν αποθηκεύονται στη συλλογή rdfSubClCollection2 και στο vector grandchildren.

```
      for(int j=0; j<previousClasses.size(); j++){
            RDFSClass rdfCl = (RDFSClass)previousClasses.get(j);
            Collection<RDFSClass> rdfSubClCollection1 =
                                    rdfCl.getNamedSubclasses(false);
            for(RDFSClass a : rdfSubClCollection1){
                  children.add(a);
                  if(a.getSubclassCount()>0){
                  Collection<RDFSClass> rdfSubClCollection2 =
                                          a.getNamedSubclasses(false);
                  for(RDFSClass b : rdfSubClCollection2){
                        grandChildren.add(b);
                  }
            }
      }
```

Στη συνέχεια, για κάθε κλάση-παιδί ανακτάται το σύνολο στιγμιοτύπων της και τοποθετείται στο totalIndividualList.

```
ArrayList totalndividualList = new ArrayList();
for(RDFSClass a  : children){
      Collection<RDFIndividual> oInst = a.getInstances(false);
      for(RDFIndividual ind : oInst){
            totalndividualList.add(ind);                }
      }
```

Ακολούθως, για κάθε στιγμιότυπο ανακτώνται από το πεδίο rdfs:comment, οι πληροφορίες για το όνομα του εγγράφου, το μονοπάτι του στο δίσκο, το μέγεθος του κειμένου, τη λίστα κλάσεων σημασιολογικού χαρακτηρισμού, τις πληροφορίες τμηματικού ή ολικού χαρακτηρισμού του εγγράφου, ώστε να δημιουργηθεί αντικείμενο που θα περιγράφει το στιγμιότυπο. Τα αντικείμενα αυτά αποθηκεύονται στο Vector hitsOwlData, απ' όπου θα χρησιμοποιηθούν από τη μέθοδο proximitySearchActionPerformed.

4.4.1.3 Μέθοδος nextGenerationActionPerformed

Η μέθοδος αυτή καλείται κάθε φορά που ο χρήστης πατάει το κουμπί "Get next generation", το οποίο ενεργοποιείται μετά από απλή ή συνδυαστική αναζήτηση στην οντολογία. Αρχικά απενεργοποιείται το κουμπί "Proximity Search" και τίθεται η σημαία που ειδοποιεί τον renderer ότι πρόκειται για αναζήτηση γειτονίας. Μετά, καλείται η συνάρτηση navigateOntologyGetNextGeneration με όρισμα τους κόμβους της προηγούμενης αναζήτησης (aClsNextGenerNProximity για το πρώτο πάτημα του κουμπιού, children για ανάκτηση επόμενων γενιών).

```
private void nextGenerationActionPerformed(java.awt.event.ActionEvent
evt) {
      proximitySearchButton.setEnabled(false);
      rendererFlag = 1;
. . .
      if(depth == 0){
            navigateOntologyGetNextGeneration(
                                          aClsNextGenerNProximity);
            currentSearchClasses.clear();
            currentSearchClasses.addAll(children);
```

```
        } else{
              navigateOntologyGetNextGeneration(children);
              currentSearchClasses.clear();
              currentSearchClasses.addAll(children);
        }
```

Η υλοποίηση της μεδόδου είναι παρόμοια με αυτή της μεθόδου proximitySearchActionPerformed, με τη διαφορά ότι για όσο οι τρέχοντες κόμβοι αναζήτησης έχουν απογόνους, καλείται η μέθοδος navigateOntologyGetNextGeneration η οποία επιστρέφει τα στιγμιότυπα των απογόνων. Τα αποτελέσματα αυτά παρουσιάζονται στο χρήστη, με σκορ που προκύπτει από τον συνήθη υπολογισμό σημασιολογικής ή συνδυαστικής αναζήτησης.

4.4.1.4 Μέθοδος navigateOntologyGetNextGeneration

Η μέθοδος αυτή καλείται από την nextGenerationActionPerformed, για να εκτελέσει την αναζήτηση στιγμιοτύπων κόμβων απογόνων. Η λειτουργία είναι παρόμοια με αυτή της navigateProximitySearch. Επίσης σε κάθε πάτημα του κουμπιού "Get next generation" εμφανίζεται το βάθος που βρίσκονται τα τρέχοντα αποτελέσματα.

```
private void navigateOntologyGetNextGeneration(ArrayList prevClas) {
. . .
      if(grandChildren == null || grandChildren.isEmpty()){
      depth = 0;
      nextGenerationButton.setText("Get next generation");
      nextGenerationButton.setEnabled(false);
      ontFlag = 0;
      }else{
            depth++;
            nextGenerationButton.setText(" Depth "+depth +". Get next
                                                      generation");
      }
. . .
}
```

4.4.1.5 Μέθοδος relatedClassActionPerformed

Η μέθοδος αυτή χρησιμοποιείται για την εμφάνιση των αποτελεσμάτων στο χρήστη και έχει τροποποιηθεί κατάλληλα, ώστε να υποστηρίζει την εμφάνιση των αποτελεσμάτων αναζήτησης γειτονίας.

Τέλος, έγιναν οι απαραίτητες τροποποιήσεις του κώδικα ώστε να είναι δυνατή η λειτουργία «αναζήτησης στα αποτελέσματα», σε αποτελέσματα αναζήτησης γειτονίας.

5

Πειραματικός έλεγχος

5.1 Αυτόματος σημασιολογικός χαρακτηρισμός

5.1.1 Στόχοι

Το σύστημα αυτόματου σημασιολογικού χαρακτηρισμού έχει ως στόχο την διευκόλυνση του χρήστη στο χαρακτηρισμό μεγάλου πλήθους εγγράφων. Για να είναι αποτελεσματικό, είναι απαραίτητο ο επιθυμητός κόμβος χαρακτηρισμού τις περισσότερες φορές να βρίσκεται εντός των προτεινόμενων. Επίσης, σημαντικό είναι να μην παρουσιάζονται υπερβολικά πολλά αποτελέσματα στο χρήστη - ιδανικά θα πρέπει εξετάζοντας τις λίγες κορυφαίες προτάσεις του συστήματος, ο χρήστης να μπορεί να βρει τη σωστή κατηγορία. Ακόμη, είναι ευκταίο για την καλύτερη διεπαφή με το χρήστη, το σκορ των προτεινόμενων κόμβων να συμβαδίζει με την πιθανότητα να είναι σωστοί[5] (πχ. ένας κόμβος που προτείνεται πρώτος στην κατάταξη με σκορ 90% να είναι περισσότερο πιθανό να είναι σωστός, απ' ότι ένας κόμβος που προτείνεται επίσης πρώτος, αλλά με ποσοστό 30%). Κατ'αυτόν τον τρόπο, ο διαχειριστής του συστήματος μπορεί να αποφασίσει να μην προτείνονται αποτελέσματα με πολύ χαμηλές βαθμολογίες, και ο χρήστης μπορεί να αποφασίσει να μην εξετάσει τα προτεινόμενα αποτελέσματα, εάν οι βαθμολογίες τους είναι γενικά χαμηλές.

[5] Αυτό αντανακλά τη θεμελιώδη «αρχή της συνέπειας» από τον τομέα της Αλληλεπίδρασης Ανθρώπου-Υπολογιστή (Human-Computer Interaction, HCI) [WBL04]

Ακόμη, όσον αφορά την ορθότητα των προτεινόμενων αποτελεσμάτων, διακρίνουμε τρια επίπεδα ορθότητας: Ιδανικά, ένα προτεινόμενο αποτέλεσμα, θα αντιπροσωπεύει την πραγματική κατηγορία του κειμένου για το οποίο προτείνεται (αυστηρή ορθότητα). Εάν κάτι τέτοιο δεν συμβαίνει, είναι εντούτοις προτιμότερο να αντιπροσωπεύει μια γενικότερη κατηγορία αυτής (π.χ. το κείμενο πραγματεύεται Τεχνικές Πολυδιάστατης Ευρετηριοποίησης και το σύστημα προτείνει την κατηγορία Τεχνικές Ευρετηριοποίησης) (ορθότητα γενίκευσης). Τέλος, εάν και αυτό δεν συμβαίνει, είναι πάλι προτιμότερο κάποια προτεινόμενη κατηγορία να πρόκειται για συγγενή κατηγορία της πραγματικής, και συγκεκριμένα για «αδελφική» κατηγορία στο δέντρο της οντολογίας (π.χ., στο ανωτέρω παράδειγμα, το σύστημα προτείνει την κατηγορία Τεχνικές Μονοδιάστατης Ευρετηριοποίησης) (ορθότητα συνάφειας).

Λαμβάνοντας τις ανωτέρω παρατηρήσεις υπ'όψιν, για να μετρήσουμε την αποδοτικότητα της μεθόδου μας, χρησιμοποιούμε τις εξής μετρικές:

Κατ'αρχήν, χρησιμοποιούμε τις κλασσικές μετρικές Ακρίβειας και Ανάκλησης (βλ. παράγραφο 3.3.4.4). Ως αποτέλεσμα του αλγορίθμου, θεωρούμε το πρώτο προτεινόμενο αποτέλεσμα, εφόσον αυτό έχει βαθμολογία άνω ενός κατωφλίου (διαφορετικά, υποθέτουμε ότι το αποτέλεσμα δεν προτείνεται στο χρήστη, ή ότι ο χρήστης το αγνοεί). Μεταβάλλοντας το κατώφλι, μπορούμε να παρατηρήσουμε διαφορετικά ζεύγη τιμών Ακρίβειας και Ανάκλησης του συστήματος. Για τον καθορισμό του «σωστού» αποτελέσματος, χρησιμοποιούνται εναλλάξ και οι τρεις ορισμοί της ορθότητας που δόθηκαν προηγουμένως.

Επίσης, χρησιμοποιούνται οι παραλλαγές τους για αλληλεπιδραστικό σύστημα (βλ. παράγραφο 3.3.4.5). Ως αποτελέσματα του αλγορίθμου θεωρούμε όσα προτεινόμενα αποτελέσματα έχουν βαθμολογία πάνω από το εκάστοτε κατώφλι, όπως προηγουμένως. Μεταβάλλοντας το κατώφλι, μπορούμε να παρατηρήσουμε διαφορετικά ζεύγη τιμών Τροπ/νης Ακρίβειας και Τροπ/νης Ανάκλησης του συστήματος. Για τον καθορισμό του «σωστού» αποτελέσματος, χρησιμοποιούνται εναλλάξ και οι τρεις ορισμοί της ορθότητας που δόθηκαν προηγουμένως.

5.1.2 Πειραματική αξιολόγηση

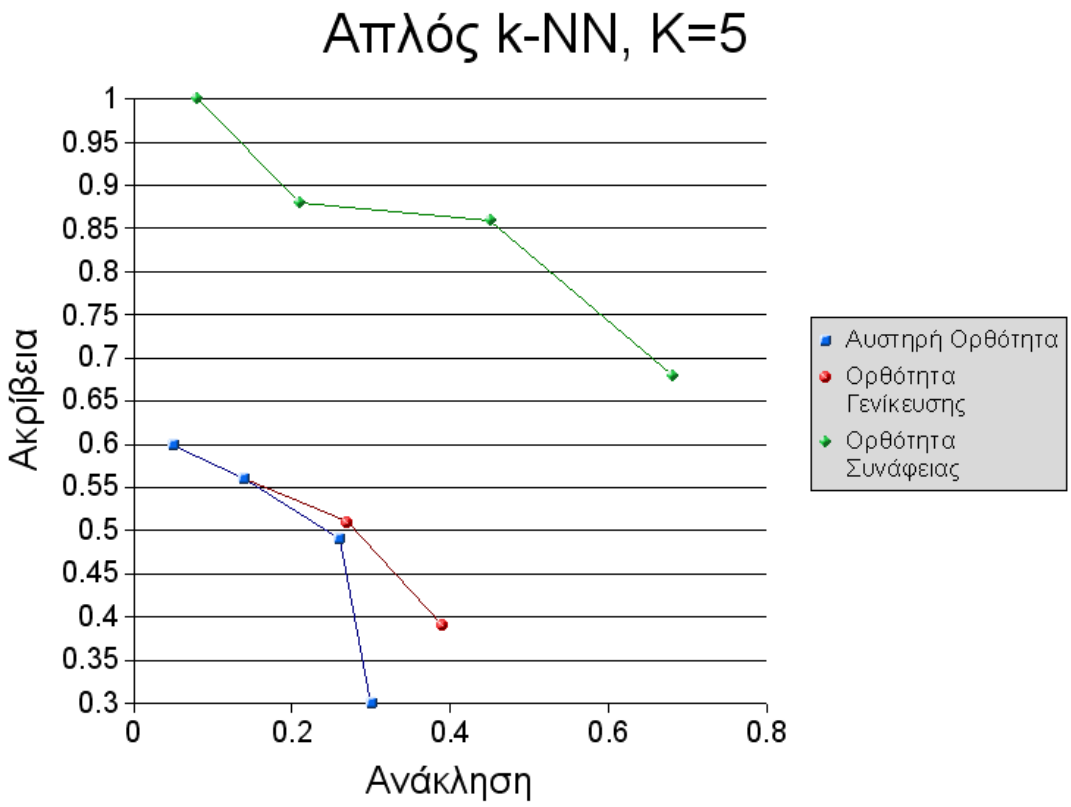
Για τις ανάγκες της αξιολόγησης, χρησιμοποιήθηκε την οντολογία ACM (βλ κεφ 3.4.2), καθώς και δημοσιεύσεις στον τομέα της επιστήμης υπολογιστών που ανακτήθηκαν από πρακτικά συνεδρίων, από την ηλεκτρονική βιβλιοθήκη της ACM [9]. Από κάθε δημοσίευση, εξήχθη το κείμενο, από το οποίο απομονώθηκαν οι κατηγορίες στις οποίες την κατέταξαν οι συγγραφείς της. Επίσης, για να εξασφαλιστεί ότι η εκπαίδευση του σύστηματος θα γίνει με βάση το περιεχόμενο των εγγράφων, από το κείμενό τους αφαιρέθηκε το όνομα και ο κωδικός αριθμός των κατηγοριών αυτών. Από αυτές, την ορισμένη από τους συγγραφείς ως

βασική κατηγορία (primary category) θεωρούμε ως κατηγορία του κειμένου. Δεν συμπεριλάβαμε στην αξιολόγηση κείμενα που δεν περιλάμβαναν ορισμένη κατηγορία.

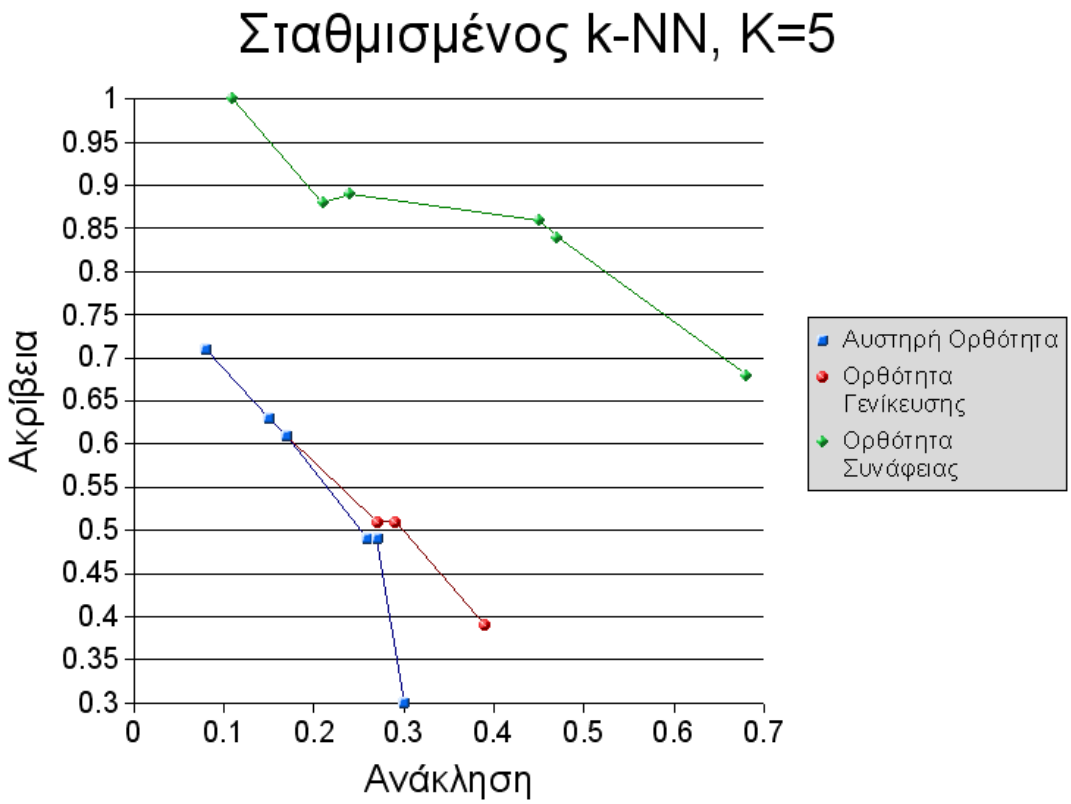
Συγκεκριμένα, χρησιμοποιήθηκαν 500 δημοσιεύσεις ως σύνολο δεδομένων εκπαίδευσης, και εκτελέστηκε η λειτουργία του Αυτόματου Σημασιολογικού Χαρακτηρισμού πάνω σε σύνολο 66 διαφορετικών δημοσιεύσεων.

Επίσης, για να εξετάσουμε την επιρροή των παραμέτρων του αλγορίθμου στην ποιότητα των παραγόμενων αποτελεσμάτων, μεταβάλλαμε τις παραμέτρους του k-NN αλγορίθμου - συγκεκριμένα, το είδος του αλγορίθμου: απλός, σταθμισμένος (weighted), και την τιμή της παραμέτρου k.

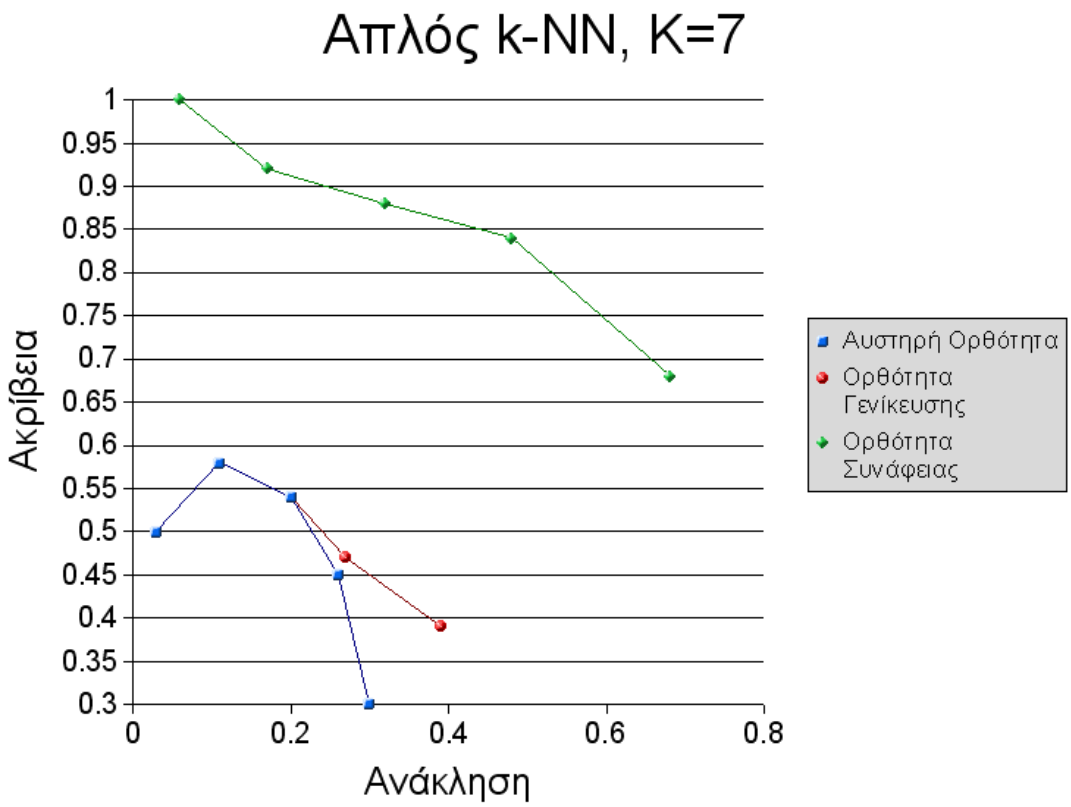
Ακολούθως παρατίθενται γραφήματα Ανάκλησης/Ακρίβειας και Τροποποιημένης Ανάκλησης/Τροποποιημένης Ακρίβειας, και ακολουθεί σχολιασμός τους.



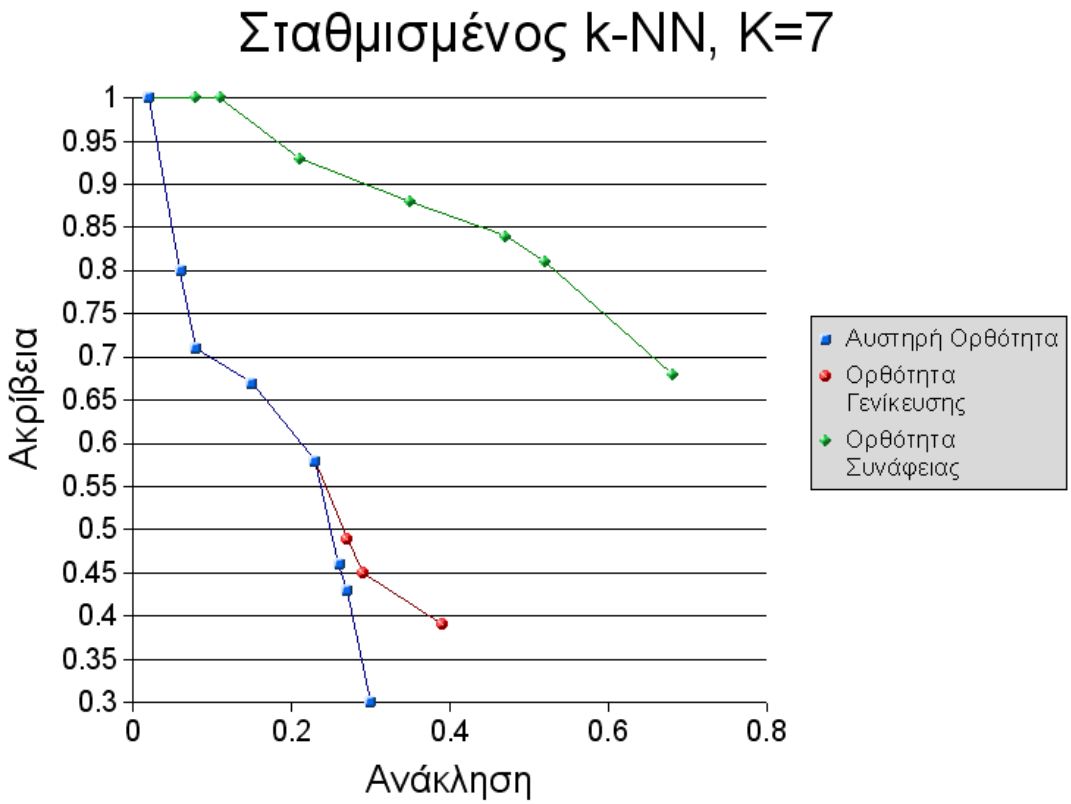
Σχήμα 27



Σχήμα 28

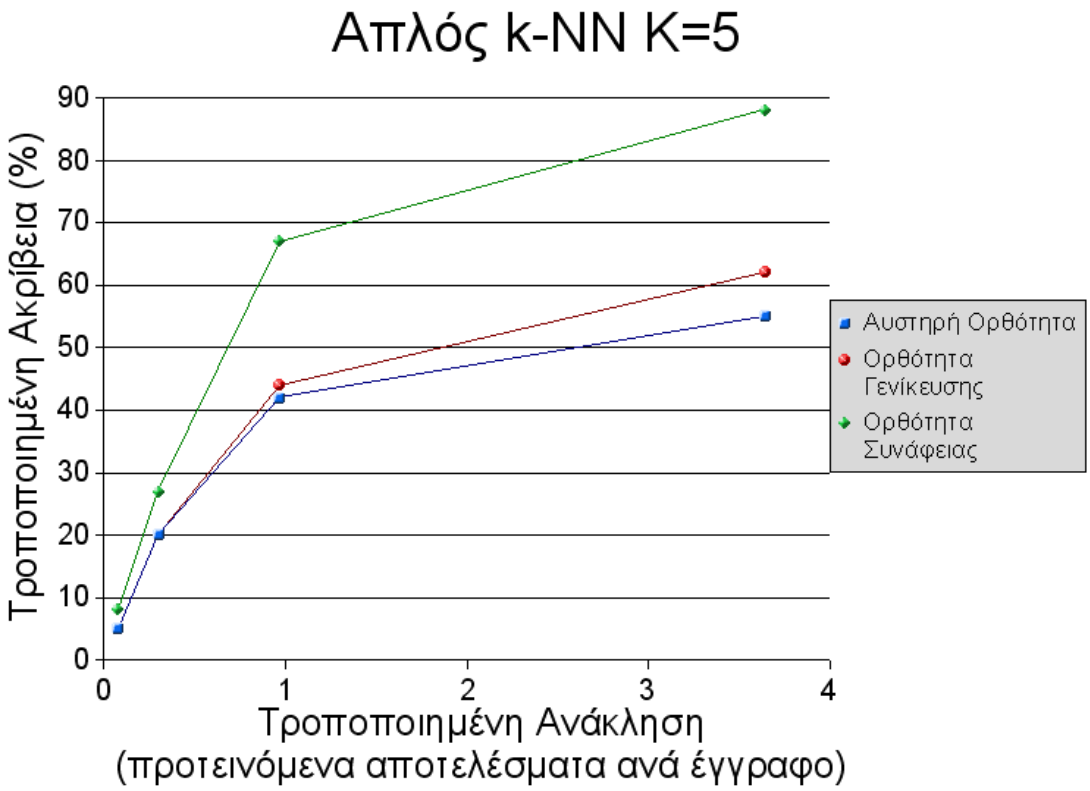


Σχήμα 29

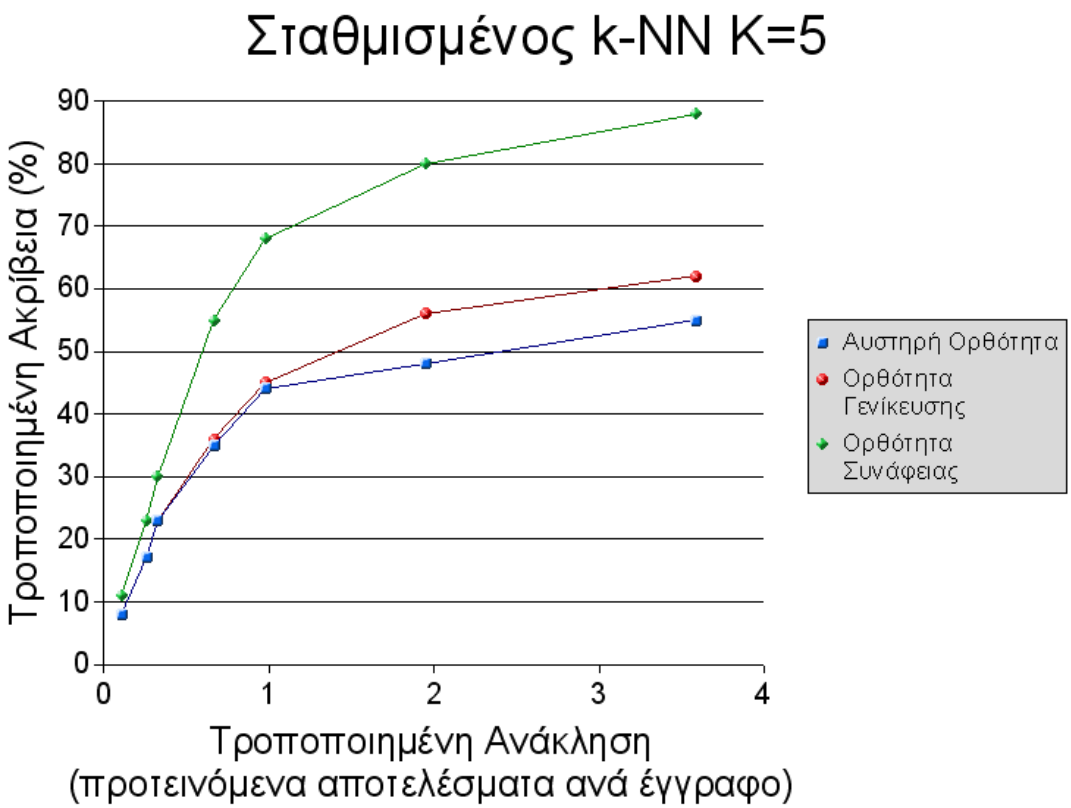


Σχήμα 30

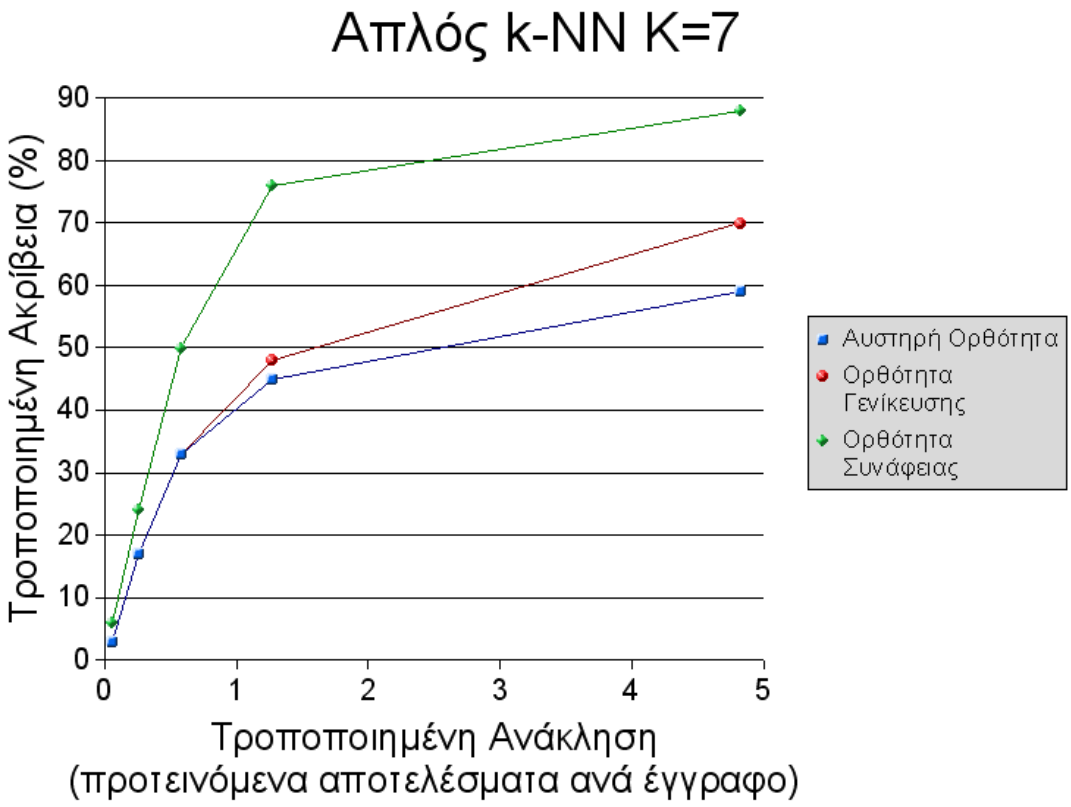
Από τα αποτελέσματα αυτά παρατηρούμε τα εξής: Κατ'αρχήν, το σύστημα παρουσιάζει ένα αναμενόμενο ισοζύγιο Ακρίβειας / Ανάκλησης. Επίσης, η απόδοσή του δεν επηρρεάζεται σημαντικά από την τιμή του K που χρησιμοποιείται, επομένως ο καθορισμός του από το χρήστη ή το διαχειριστή του συστήματος, για κάθε περίπτωση χρήσης, δεν κρίνεται σκόπιμος. Μολαταύτα, η απόδοση του συστήματος βελτιώνεται αισθητά όταν χρησιμοποιείται ο Σταθμισμένος k-NN αλγόριθμος που υλοποιήσαμε, αντί του απλού. Αυτό είναι αναμενόμενο, δεδομένης της τάσης του Απλού k-NN να ευνοεί λιγότερο σχετικά έγγραφα εκπαίδευσης.



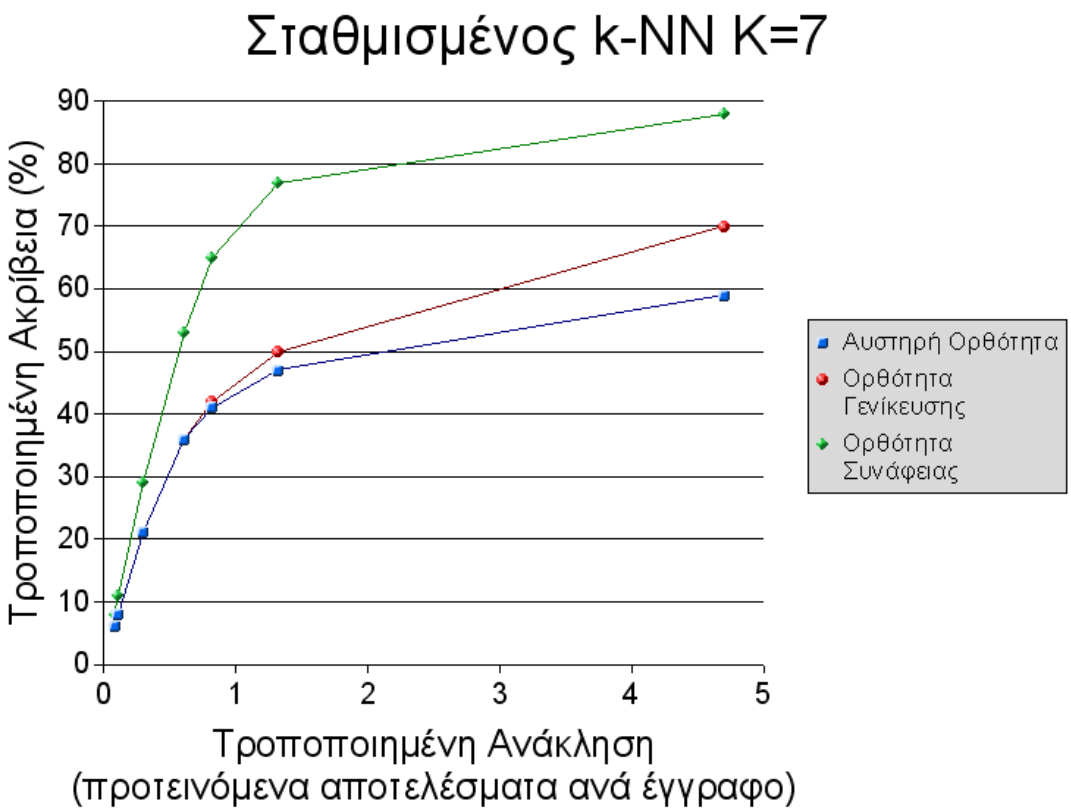
Σχήμα 31



Σχήμα 32



Σχήμα 33

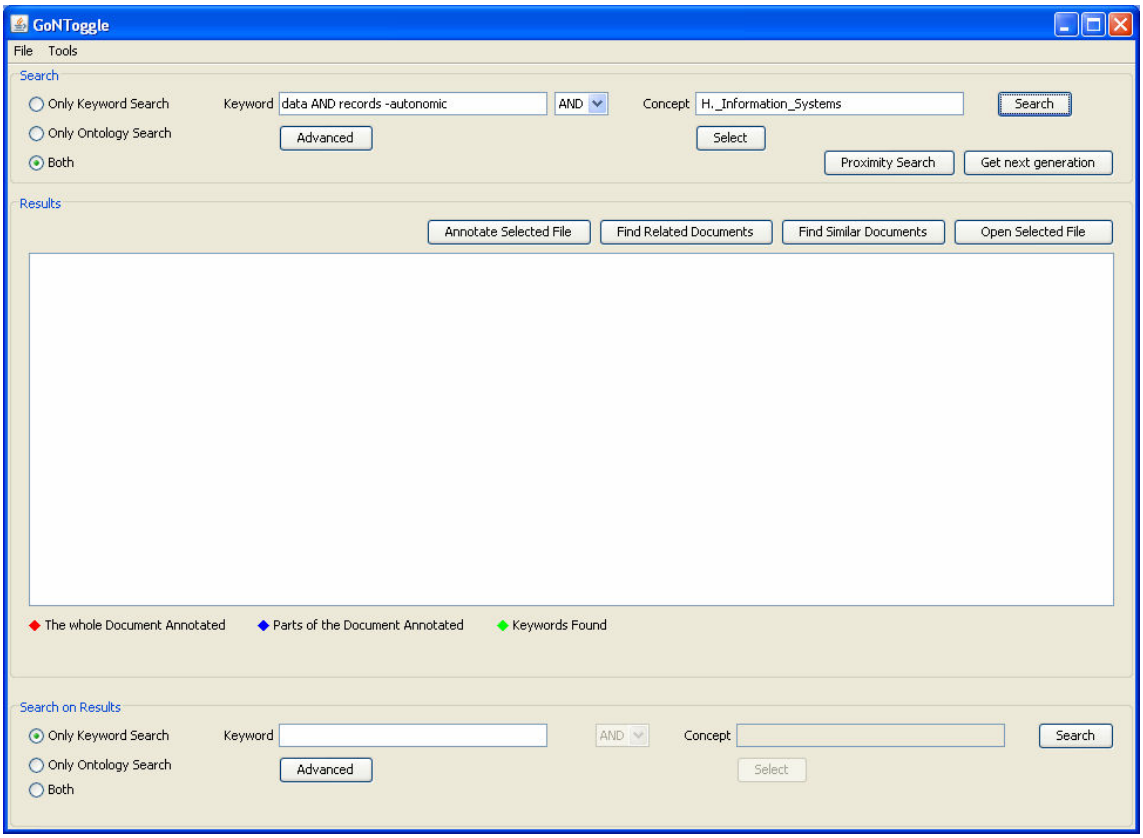


Σχήμα 34

Από τα αποτελέσματα αυτά, μπορεί κανείς να παρατηρήσει ότι το σύστημά παρουσιάζει στην πράξη πολύ καλή απόδοση. Ο χρήστης μπορεί να χαρακτηρίσει σωστά 60-90% των εγγράφων του (ανάλογα με την αυστηρότητα της ορθότητας που επιθυμεί), εξετάζοντας λιγότερες από 4 προτεινόμενες κατηγορίες ανά έγγραφο. Κατά την ερμηνεία αυτών των αριθμών, ας συνυπολογιστεί το γεγονός ότι η κατηγοριοποίηση επιστημονικών δημοσιεύσεων είναι έργο δύσκολο: Το περιεχόμενο των δημοσιεύσεων συχνά αναφέρεται σε πολλαπλές κατηγορίες, διαφορετικές μεταξύ τους. Επίσης, αρκετοί κόμβοι στην οντολογία ACM, βρίσκονται εννοιολογικά πολύ κοντά, λόγω της εξειδίκευσης της οντολογίας. Αυτός είναι και ένας λόγος που το σύστημα παρουσιάζει σαφώς βελτιωμένη συμπεριφορά για λιγότερο αυστηρούς ορισμούς «ορθότητας» αποτελεσμάτων.

5.2 Αναζήτηση σημασιολογικής γειτονίας

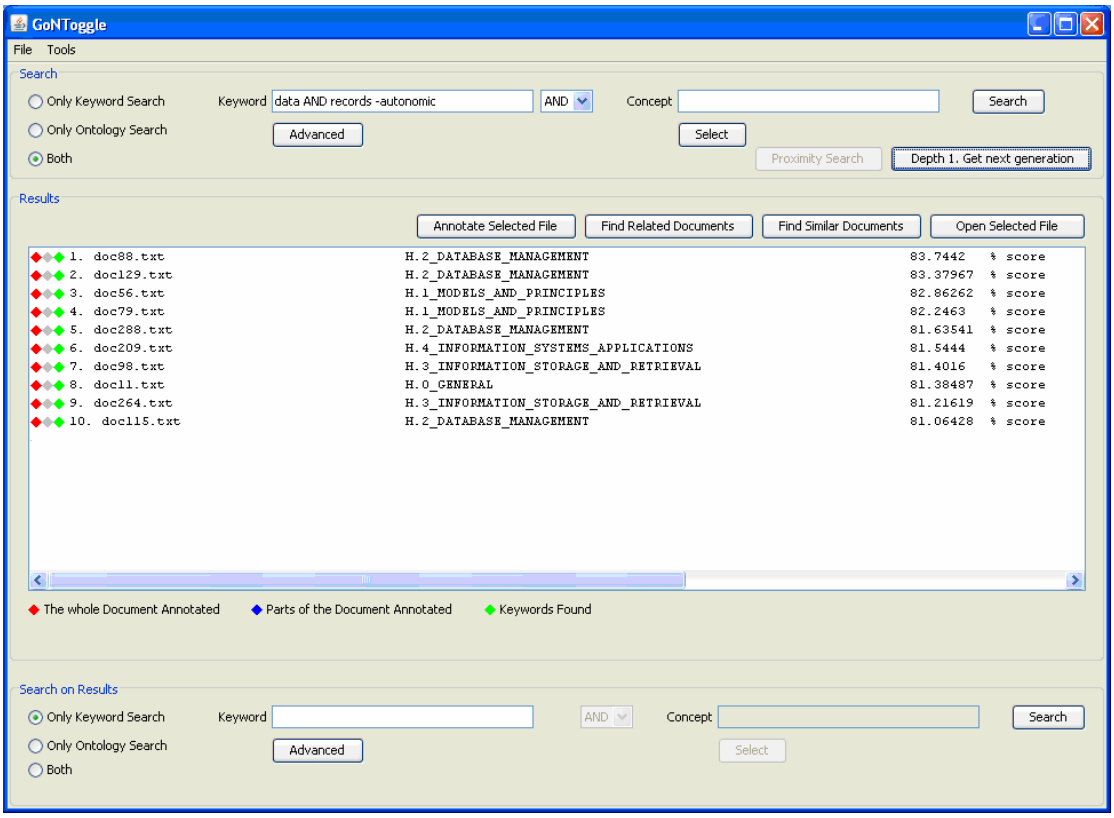
Ακολούθως παρατίθενται παραδείγματα εκτέλεσης αναζήτησης σημασιολογικής γειτονίας και με τις δύο μεθόδους που υλοποιήθηκαν, στο σύνολο δεδομένων που περιγράφηκε. Τα παραδείγματα αυτά αναδεικνύουν την αποδοτικότητα των προτεινόμενων μεθόδων στον εμπλουτισμό των αποτελεσμάτων της συνήθους αναζήτησης (σημασιολογική αναζήτηση και συνδυαστική αναζήτηση). Έχουν ως κοινή αφετηρία την απλή συνδυαστική αναζήτηση εγγράφων της κατηγορίας "H.Information Systems", που να περιέχουν τις λέξεις "data" και "records", αλλά όχι τη λέξη "autonomic", που όπως φαίνεται και στο παρακάτω σχήμα, δεν επιστρέφει απότελέσματα.



Σχήμα 35 Αποτελέσματα απλής συνδυαστικής αναζήτησης

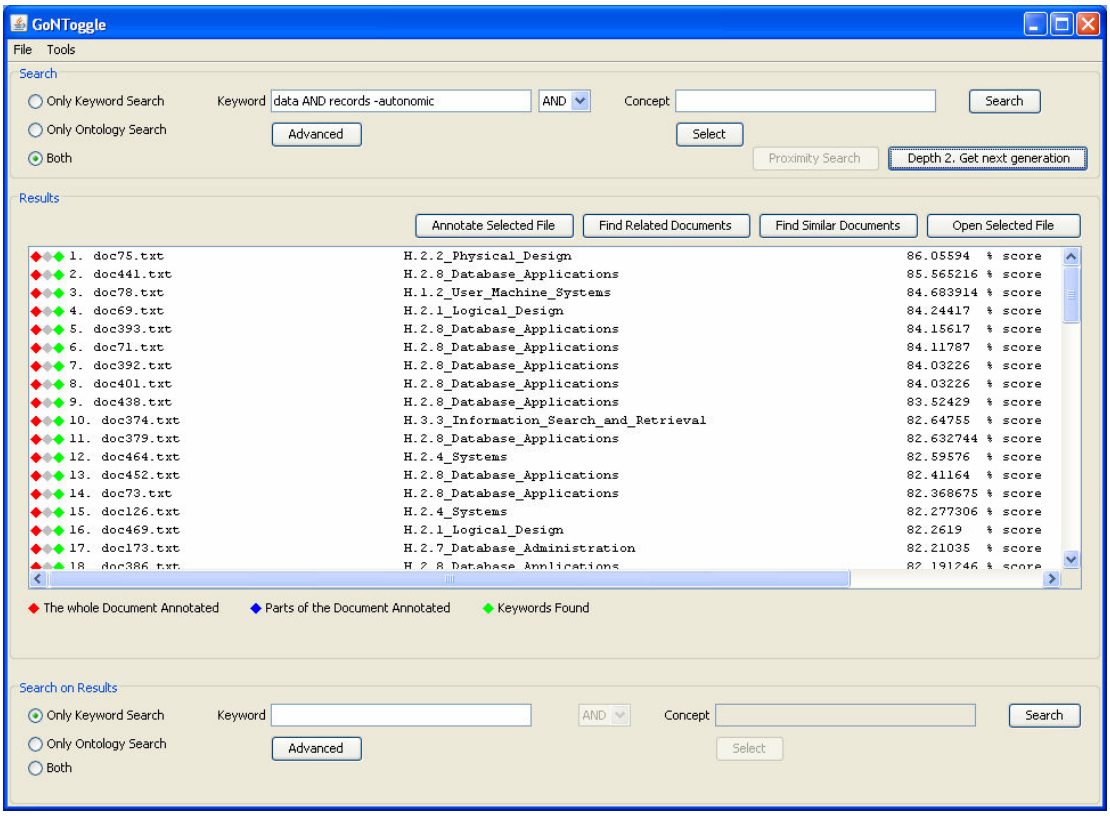
5.2.1 Αναζήτηση Επόμενης Γενιάς

Ο υποθετικός χρήστης Α θέλει να βρει μία παρουσίαση που είχε παρακολουθήσει, και απ' όσο θυμάται ικανοποιεί τα κριτήρια της παραπάνω αναζήτησης. Μη βρίσκοντας αποτελέσματα, χρησιμοποιεί την ΑΕΓ για την διαδοχική ανάκτηση εγγράφων, κατά μειούμενη σημασιολογική σχετικότητα. Με το πάτημα του κουμπιού "Get next generation", ανακτώνται τα έγγραφα που είναι χαρακτηρισμένα με κάποιο κόμβο-παιδί αυτής της κατηγορίας, και περιέχουν τις επιθυμητές λέξεις:



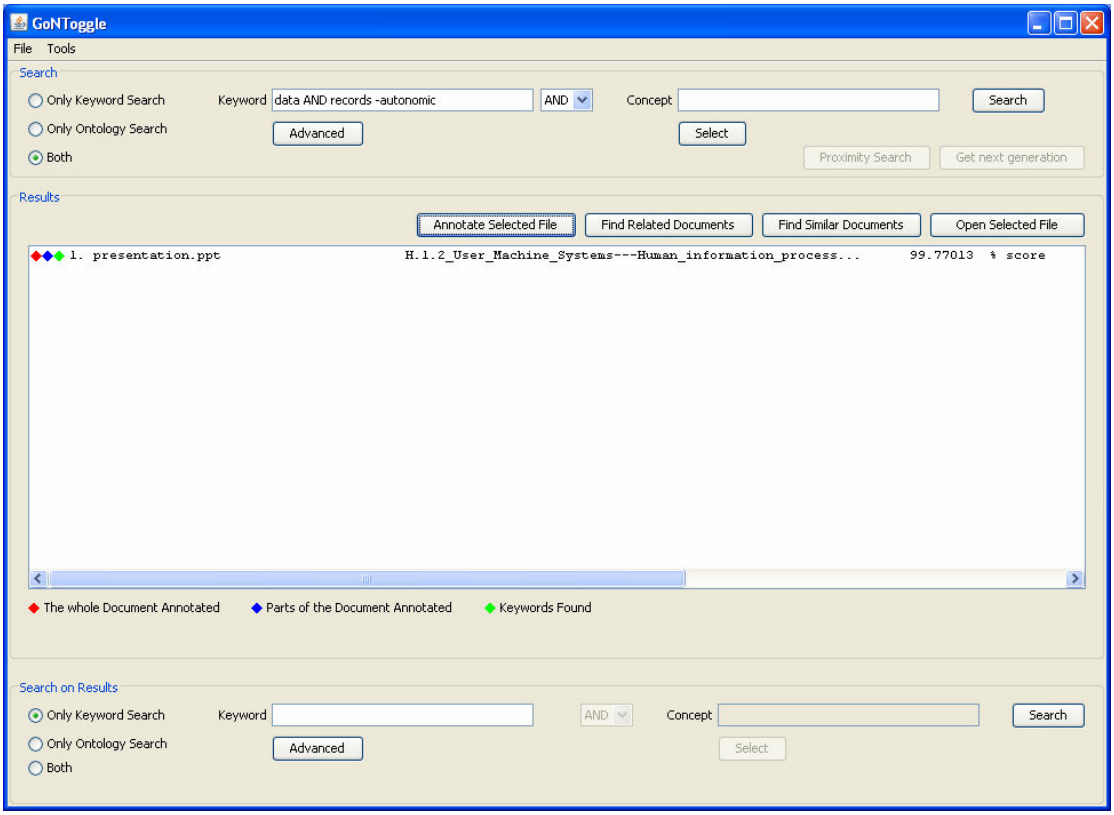
Σχήμα 36 Αποτελέσματα βάθους 1

Εφόσος ο χρήστης δεν έχει εντοπίσει ακόμα το έγγραφο που επιθυμεί, ζητά κι άλλα αποτελέσματα (από κόμβους μεγαλύτερου βάθους), ξαναπατώντας το κουμπί.



Σχήμα 37 Αποτελέσματα βάθους 2

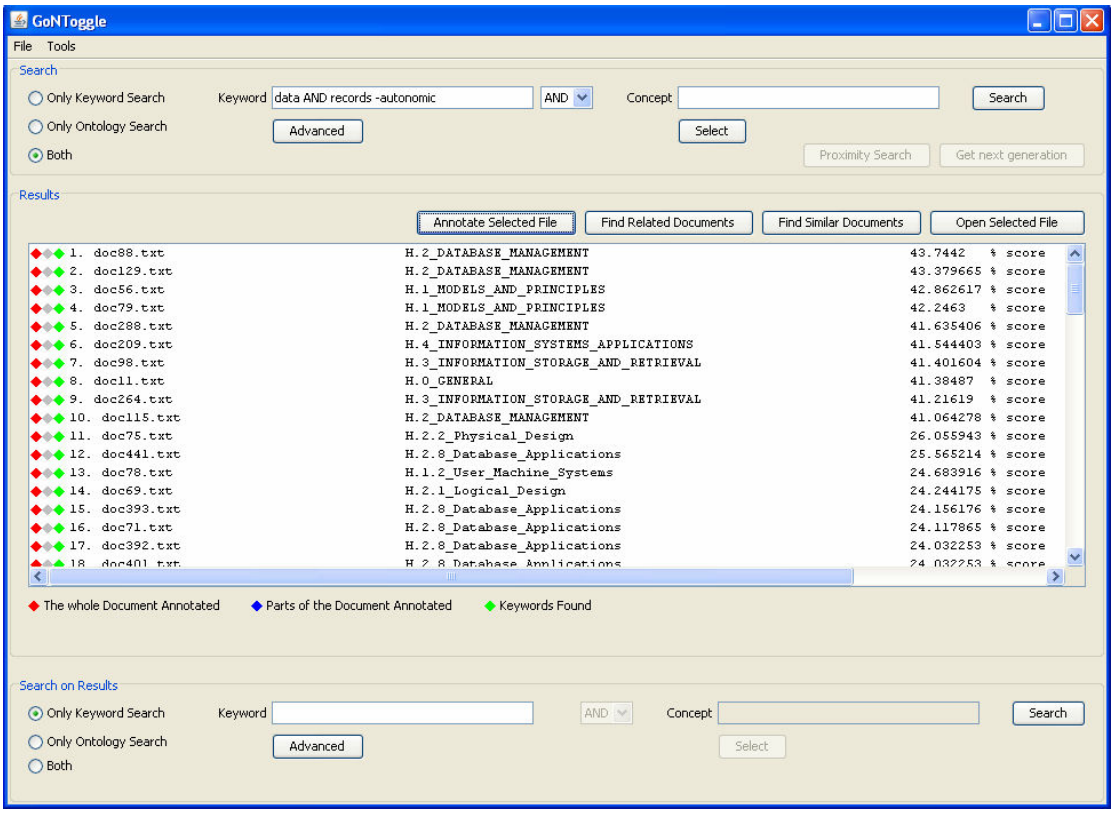
Στην επόμενη επανάληψη ανακτάται το επιθυμητό έγγραφο.



Σχήμα 38 Αποτελέσματα βάθους 3.

5.2.2 Αναζήτηση Στενής Γειτονίας

Ο υποθετικός χρήστης Β αναζητά επιστημονικές δημοσιεύσεις για μια βιβλιογραφική μελέτη, χρησιμοποιώντας την αρχική αναζήτηση. Μη βρίσκοντας αποτελέσματα, χρησιμοποιεί την ΑΣΓ, για να βρει τα έγγραφα που προσεγγίζουν περισσότερο την αρχική κατηγορία. Με την εφαρμογή της μεθόδου, ανακτά πλήθος σχετικών δημοσιεύσεων, όπως φαίνεται στο παρακάτω σχήμα:



Σχήμα 39 Αποτελέσματα ΑΣΓ

6

Λοιπές τροποποιήσεις

6.1 Αλλαγές στο GoNTogle 1.0

Όπως κάθε σύστημα λογισμικού, έτσι και το GoNTogle 1.0 παρουσιάζει ενίοτε μη αναμενόμενες ή και ανεπιθύμητες συμπεριφορές. Στα πλαίσια της διπλωματικής αυτής ανέκυψε η ανάγκη επίλυσης αρκετών από αυτές και ταυτοποιήθηκαν άλλες. Σε αυτό το κεφάλαιο περιγράφονται οι συμπεριφορές αυτές, ο τρόπος με τον οποίο επιλύθηκαν οι πρώτες και ο περιορισμός των ανεπιθύμητων ενεργειών που προκύπτουν απ' τις δεύτερες.

6.1.1 Υποστήριξη νέων τύπων αρχείων

Το GoNTogle είναι επιθυμητό να υποστηρίζει πολλούς τύπους αρχείων για να καλύπτει καλύτερα τις ανάγκες αρχειοθέτησης και αναζήτησης του χρήστη. Γι' αυτό προστέθηκε η δυνατότητα ευρετηριοποίησης και αναζήτησης νέων τύπων αρχείων.

6.1.1.1 Αρχεία Postscript (.ps)

Για την επεξεργασία αρχείων ps το GoNTogle 2.0 καλεί την εξωτερική εφαρμογή ps2pdf από το πακέτο Ghostscript[6], η οποία δημιουργεί ένα νέο αρχείο pdf. Το προσωρινό αυτό αρχείο

[6] Υποθέτουμε ότι το GoNTogle εκτελείται σε λειτουργικό σύστημα Microsoft Windows. Σε περίπτωση άλλων λειτουργικών συστημάτων, πχ Linux, MacOS, X, η αντίστοιχη εφαρμογή είναι μέρος των συνηθέστερων διανομών.

επεξεργάζεται με τον υπάρχοντα handler για pdf του GoNTogle 1.0 και κατόπιν διαγράφεται. Για την υποστήριξη ps αρχείων, το εκτελέσιμο ps2pdf πρέπει να βρίσκεται σε κατάλογο που αναφέρεται στη μεταβλητή PATH του συστήματος.

6.1.1.2 Αρχεία Microsoft Office PowerPoint (.ppt)

Για την ανάγνωση αρχείων ppt γράφτηκε μια μικρή βιβλιοθήκη εξαγωγής κειμένου από τέτοια αρχεία, βασισμένη στη βιβλιοθήκη POI [8]. Κατά τη συγγραφή αυτής της διπλωματικής, λόγω προβλήματος στη βιβλιοθήκη POI, το GoNTogle 2.0 δεν μπορεί να διαβάσει ppt που έχει δημιουργηθεί με PowerPoint 2003 ή μεταγενέστερο. Αυτό το σφάλμα αναμένεται να λυθεί σε μελλοντική έκδοση της βιβλιοθήκης POI.

6.1.2 Πλήρης υποστήριξη λοιπών τύπων αρχείων

Αρκετοί τύποι αρχείων, τύγχαναν ελλιπούς υποστήριξης από το GoNTogle 1.0. Αυτό το πρόβλημα έχει διορθωθεί στο GoNTogle 2.0[7]. Ειδικότερα:

6.1.2.1 Πλήρης υποστήριξη ελληνικών

Προστέθηκε πλήρης υποστήριξη ελληνικών σε τύπους αρχείων που την υποστήριζαν μερικώς (αρχεία doc, html).

6.1.2.2 Αρχεία HTML (.htm, .html)

Ένα συχνό πρόβλημα κατά την ανάγνωση αρχείων html είναι η αναγνώριση της κωδικοσελίδας στην οποία έχουν γραφτεί. Εάν η κωδικοσελίδα δεν αναγνωριστεί σωστά, δεν είναι δυνατή η σωστή ερμηνεία των χαρακτήρων του κειμένου. Το φαινόμενο αυτό εμφανίζεται ενίοτε και σε εμπορικούς φυλλομετρητές (πχ είναι το γνωστό φαινόμενο των ελληνικών ιστοσελίδων που εμφανίζονται με παράξενους χαρακτήρες). Για την επίλυση αυτού του προβλήματος γράφτηκε βιβλιοθήκη αυτόματης αναγνώρισης κωδικοσελίδας html σελίδων βασισμένη στο [Αντ06].

6.1.2.3 Αρχεία Adobe PDF (.pdf)

Το GoNTogle 1.0 κατά την επεξεργασία μεγάλου πλήθους αρχείων pdf εμφάνιζε προβλήματα αστάθειας και καταστάσεις σφαλμάτων. Διαπιστώθηκε ότι οφείλονταν σε ζητήματα διαχείρισης πόρων (κάποιοι περιγραφείς αρχείων εσφαλμένα δεν απελευθερώνονταν), και το πρόβλημα επιδιορθώθηκε.

---

[7] Αρκετές τροποποιήσεις από τις ακόλουθες χρειάστηκαν επέμβαση στη βιβλιοθήκη POI.

6.1.2.4 Αρχεία Microsoft Excel (.xls)

Το GoNTogle 1.0 παρουσίαζε αδυναμία ανάγνωσης τέτοιου τύπου αρχείων με μη συνεχόμενα κελιά. Το πρόβλημα επιλύθηκε επεμβαίνοντας στη βιβλιοθήκη ανάγνωσης των αρχείων Excel.

6.1.3 Λοιπές τροποποιήσεις

6.1.3.1 Χειρισμός ρυθμίσεων προγράμματος

Διορθώθηκε σφάλμα που δεν επέτρεπε τη σωστή ανάγνωση του αρχείου ρυθμίσεων για την ανάγνωση αρχείων (handler.properties).

6.1.3.2 Προστασία από διπλό σημασιολογικό χαρακτηρισμό κειμένων

Το GoNTogle 1.0 παρουσίαζε ασυνεπή συμπεριφορά όταν ο χρήστης επιχειρούσε να προσθέσει εκ των υστέρων σημασιολογικό χαρακτηρισμό σε ήδη χαρακτηρισμένο κείμενο. Για την επίτευξη συνεπούς και αναμενόμενης συμπεριφοράς το σύστημα δεν επιτρέπει το «διπλό χαρακτηρισμό» ενός κειμένου.

6.1.3.3 Βελτίωση συνέπειας της βάσης γνώσης

Για να είναι ορατές οι αλλαγές που γίνονται σε μία οντολογία από το GoNTogle 1.0 (πχ για να γίνει ορατή στο υποσύστημα αναζήτησης μία νέα προσθήκη σημασιολογικού χαρακτηρισμού εγγράφου) είναι απαραίτητη η επαναφόρτωση της οντολογίας από το δίσκο. Για να είναι συνεπής η συμπεριφορά του συστήματος ως προς το χρήστη, πλέον τον ενημερώνει με κατάλληλο πληροφοριακό μήνυμα. Επίσης, σε ότι αφορά τη βάση γνώσης ,επιτρέπεται πλέον η χρήση του χαρακτήρα “ ” (κενό) σε ονόματα κλάσεων.

6.1.3.4 Προσθήκη κουμπιού αλλαγής οντολογίας

Τέλος, GoNTogle 2.0 στο παράθυρο σημασιολογικού χαρακτηρισμού προστέθηκε κουμπί που επιτρέπει την ταχεία αλλαγή της υπό επεξεργασία οντολογίας.

7

Επίλογος

7.1 Σύνοψη και συμπεράσματα

Στη διπλωματική αυτή εργασία προτείναμε επεκτάσεις του συστήματος αναζήτησης GoNTogle 1.0 προσθέτοντάς του νέες λειτουργίες

Το νέο σύστημα GoNTogle 2.0 προσφέρει στο χρήστη τη δυνατότητα αυτόματου σημασιολογικού χαρακτηρισμού εγγράφων, ώστε να μπορεί με λιγότερη προσπάθεια να χαρακτηρίζει σημασιολογικά μεγάλο πλήθος εγγράφων του, και άρα να καθίσταται πρακτική η σημασιολογική αναζήτηση των εγγράφων αυτών. Η υλοποίηση του υποσυστήματος αυτόματου σημασιολογικού χαρακτηρισμού γίνεται με χρήση τεχνικών μηχανικής μάθησης, ώστε ανάλογα με τις επιλογές του χρήστη, το σύστημα να μαθαίνει από τα λάθη του, και τις συνήθειες του χρήστη βελτιώνοντας έτσι την απόδοσή του.

Το GoNTogle 2.0 επίσης, υλοποιεί δύο μεθόδους αναζήτησης για τον εμπλουτισμό σημασιολογικών αποτελεσμάτων, για να ανταποκριθεί σε περιπτώσεις που τα επιστρεφόμενα αποτελέσματα δεν είναι επαρκή σε αριθμό και δεν καλύπτουν τις ανάγκες του χρήστη. Οι μέθοδοι αυτοί πραγματοποιούν αναζήτηση εγγράφων σε κόμβους της οντολογίας που είναι συγγενικοί με τους αρχικούς κόμβους αναζήτησης.

Το GoNTogle 2.0 εισαγάγει πλήθος άλλων μικρών βελτιώσεων και επεκτάσεων, και μία οντολογία για την κατάταξη επιστημονικών δημοσιεύσεων στον τομέα της Πληροφορικής,

βασισμένη στην κατάταξη ACM. Πειράματα με χρήση πραγματικών δεδομένων και της οντολογίας αυτής αναδεικνύουν την αποτελεσματικότητα των προτεινόμενων μεθόδων.

7.2 Μελλοντικές επεκτάσεις

Η εισαγωγή των δυνατοτήτων αυτόματου σημασιολογικού χαρακτηρισμού κειμένων, και προχωρημένων αναζητήσεων σημασιολογικής γειτονίας, αυξάνει τη χρηστικότητα του συστήματος GoNTogle. Παράλληλα, αναδεικνύει νέα ενδιαφέροντα προβλήματα, τα οποία αφήνουμε ως μελλοντικές εργασίες/επεκτάσεις του συστήματος.

Μία επέκταση που θα διευκόλυνε το χρήστη στα πρώτα στάδια εκμάθησης του αλγορίθμου μηχανικής μάθησης, είναι η ύπαρξη ενός συνόλου λέξεων, που θα αποτελεί τα δεδομένα εκπαίδευσης κόμβων που δεν περιέχουν ακόμα σημασιολογικά χαρακτηρισμένα έγγραφα. Μία λύση είναι η χρησιμοποίηση άρθρων της Wikipedia για εξαγωγή λέξεων που θα περιγράφουν τους κόμβους αυτούς και θα αποτελούν τα δεδομένα εκπαίδευσης, μέχρι να υπάρξει επαρκής αριθμός σημασιολογικά χαρακτηρισμένων κειμένων στην οντολογία. Τα άρθρα της Wikipedia ενδείκνυνται επειδή έχουν μεγάλη θεματική πληρότητα (οπότε μπορούν να καλύψουν τις ανάγκες ακόμα και αρκετά εξειδικευμένων οντολογιών), είναι ταξινομημένα σε κατηγορίες, το περιεχόμενο των άρθρων της διαμορφώνεται, εμπλουτίζεται και διορθώνεται από πολλούς χρήστες και έτσι το συνολικό αποτέλεσμα να αποτελεί έγκυρη πηγή πληροφόρησης και δείγματος του μέσου τρόπου συγγραφής κειμένων (οι λέξεις που χρησιμοποιούνται είναι αρκετά ενδεικτικές). Τέλος, λόγω της τακτικής τους ανανέωσης τα άρθρα αυτά ενδείκνυνται για την ακόλουθη χρήση: Κάθε φορά που ο χρήστης δημιουργεί ένα νέο κόμβο της οντολογίας (ή εισάγει νέα οντολογία στο σύστημα), το σύστημα (προαιρετικά) ανακτά το αντίστοιχο άρθρο της Wikipedia, και το χρησιμοποιεί για το χαρακτηρισμό του κόμβου αυτού όπως περιγράφηκε παραπάνω.

Επίσης, σημαντική βελτίωση του συστήματος θα επιτυγχάνονταν με την υποστήριξη της αυτόματης σημασιολογικής επισήμανσης ελληνικών εγγράφων. Στην παρούσα εργασία δεν επικεντρωθήκαμε στην υποστήριξη της ελληνικής γλώσσας λόγω έλλειψης χρησιμοποιούμενων γλωσσικών εργαλείων, π.χ. στελεχωτών (stemmers), η συγγραφή των οποίων για τα ελληνικά θεωρείται δύσκολο πρόβλημα, με αποτέλεσμα οι διαθέσιμοι

στελεχωτές να είναι είτε εμπορικοί (π.χ. του Ινστιτούτου Επεξεργασίας Λόγου) ή σε πρώιμη φάση ανάπτυξης (π.χ. Mitos[8]).

[8] Το Mitos είναι μια μηχανή αναζήτησης που φτιάχτηκε στα πλαίσια του μαθήματος «Συστήματα Ανάκτησης Πληροφορίας», από φοιτητές του Τμήματος Επιστήμης Υπολογιστών του Πανεπιστημίου Κρήτης. http://groogle.csd.uoc.gr/apache2-default/index.php/

8

Βιβλιογραφία

[ACM98] ACM Computing Classification System, http://www.acm.org/class/1998/

[Bur01] Carl Burch, "A survey of machine learning", http://ozark.hendrix.edu/~burch/csbsju/cs/portfolio/learning.pdf, 2001 Ανακτήθηκε την 20 Ιουνίου 2008

[CFV07] Pablo Castells, Miriam Fernández, David Vallet, "An Adaptation of the Vector-Space Model for Ontology-Based Information Retrieval," IEEE Transactions on Knowledge and Data Engineering, vol. 19, no. 2, pp. 261-272, Feb., 2007

[CHS04] Philipp Cimiano, Siegfried Handschuh, Steffen Staab: Towards the self-annotating web. WWW 2004:462-471

[CK87] Paul R. Cohen, Rick Kjeldsen, “Information retrieval by constrained spreading activation in semantic networks”. Inf. Process. Manage. (IPM) 23(4), 1987

[CMB+02] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, “GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications”, Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02). Philadelphia, July 2002

[Dal97] Theodore Dalamagas, “Automatic Construction of News Hypertext”, 1997

[DC00] Susan T. Dumais, Hao Chen, "Hierarchical classification of Web content" SIGIR, 2000

[DOS03] Michael C. Daconta, LeoJ. Obrst, andKevinT. Smith. “The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management”. Wiley Publishing, Inc., 2003

[Dud76] S. A. Dudani, “The distance-weighted k-nearest neighbour rule”, IEEE Transactions on System, Man, and Cybernetics, volume SMC-6, 1976

[GGM+02] Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., Schneider, L, “Sweetening Ontologies with DOLCE”, Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Springer Verlag, pp. 166-181

[GH04] Deborah L. McGuinness, Frank van Harmelen, “OWL Web Ontology Language: Overview”, 2004 http://www.w3.org/TR/owl-features/

[GH05] Otis Gospodnetic, Erik Hatcher, “Lucene in Action”, Manning Publications Co, 2005

[GSW05] Jens Graupmann, Ralf Schenkel, Gerhard Weikum, “The SphereSearch Engine for Unified Ranked Retrieval of Heterogeneous XML and Web Documents”, VLDB 2005

[HM06] Yifen Huang, Tom M. Mitchell, “Text clustering with extended user feedback”, SIGIR 2006

[KN03] Linus W. Kwong, Yiu-Kai Ng, “Performing Binary-Categorization on Multiple-Record Web Documents Using Information Retrieval Models and Application Ontologies”, World Wide Web (WWW), 2003

[Koh95] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection”, In C. S. Mellish, (ed.), Proceedings of IJCAI-95, pages 1137-1143. Morgan Kaufmann, 1995

[KPO+04] Atanas Kiryakov, Borislav Popov, Damyan Ognyanoff, Dimitar Manov, Angel Kirilov, Miroslav Goranov. “Semantic Annotation Indexing and Retrieval”. Journal of Web Semantics 2, Issue 1, Elsevier, 2004

[KSI+08] Gjergji Kasneci, Fabian M. Suchanek, Georgiana Ifrim, Maya Ramanath, Gerhard Weikum, “NAGA: Searching and Ranking Knowledge”, ICDE 2008

[LCV+01] Wen-Syan Li, K. Selçuk Candan, Quoc Vu, Divyakant Agrawal, “Retrieving and organizing web pages by “information unit””, WWW 2001

[LHL01] Tim Berners-Lee, James Hendler, Ora Lassila, “The Semantic Web”, Scientific American, 2001

[NP] I.Niles, A.Pease, “Towards a Standard Upper Ontology”, In Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001), Chris Welty and Barry Smith, eds, Ogunquit, Maine, October 17-19, 2001.

[Por80] M.F. Porter, “An algorithm for suffix stripping”, Program, 14(3) pp 130−137, 1980

[Reu08] Reuters OpenCalais, www.opencalais.com, Ανακτήθηκε στις 25-06-2008

[SDS06] Stefanos Souldatos, Theodore Dalamagas, Timos Sellis, “Captain Nemo: A Metasearch Engine with Personalized Hierarchical Search Space”Informatica, Vol. 30, No 2, pp. 173-181, 2006

[SKW07] Fabian M. Suchanek, Gjergji Kasneci, Gerhard Weikum, “Yago: a core of semantic knowledge”, WWW 2007

[WBL04] Christopher D. Wickens, Sallie E. Gordon-Becker and Yili Liu, “An Introduction to Human Factors Engineering”, Second ed. Upper Saddle River, NJ: Pearson Prentice Hall, 2004

[WMB99] Ian H. Witten, Alistair Moffat, Timothy C. Bell, “Managing Gigabytes: Compressing and Indexing Documents and Images”, Morgan Kaufmann Publishers, 1999

[Zip49] Zipf, G. K “Human behaviour and the principle of least effort”, Addison Wesley, 1949

[Αντ06] Αλβέρτος Άντζελ, «Εξαγωγή γεωγραφικής πληροφορίας από ημιδομημένο κείμενο», Διπλωματική Εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, 2006

[ΒΚΒ+02] Βλαχάβας Ι., Κεφαλάς Π., Βασιλειάδης Ν., Ρεφανίδης Ι., Κόκκορας Φ., Σακελλαρίου Η.. «Τεχνητή Νοημοσύνη». Εκδόσεις Γαρταγάνη, 2002

[Για06] Γιώργος Γιαννόπουλος, «GoNToggle: Έξυπνη μηχανή αναζήτησης με χρήση οντολογιών», Διπλωματική εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, Ιούλιος 2006

[Μαλ05] Πρόδρομος Μαλακασιώτης, “Αναγνώριση μερών του λόγου σε ελληνικά κείμενα με τεχνικές ενεργητικής μάθησης”, Διπλωματική Εργασία Μεταπτυχιακού Διπλώματος, Οικονομικό Πανεπιστήμιο Αθηνών, 2005

[Τζι08] Γιάννης Τζίτζικας, Σημειώσεις του μαθήματος «Συστήματα Ανάκτησης Πληροφοριών», Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης http://www.csd.uoc.gr/~hy463/2008/download/lectures/463_03_RetrievalModels_I.pdf

[1] DAML+OIL http://www.daml.org/ , Ανακτήθηκε στις 25/06/2008

[2] KIF http://www-ksl.stanford.edu/knowledge-sharing/kif/ , Ανακτήθηκε στις 25/06/2008

[3] SWOOP http://code.google.com/p/swoop/ Ανακτήθηκε στις 25/06/2008

[4] Ontolingua http://www.ksl.stanford.edu/software/ontolingua/ Ανακτήθηκε στις 25/06/2008

[5] Ontoedit http://www.ontoknowledge.org/tools/ontoedit.shtml Ανακτήθηκε στις 25/06/2008

[6] JOE http://www.cse.sc.edu/research/cit/demos/java/joe/ Ανακτήθηκε στις 25/06/2008

[7] Protege http://protege.stanford.edu/ Ανακτήθηκε στις 25/06/2008

[8] Apache POI, http://poi.apache.org Ανακτήθηκε στις 29-06-2008.

[9] ACM Digital Library, http://portal.acm.org/dl.cfm Ανακτήθηκε στις 28-6-2008