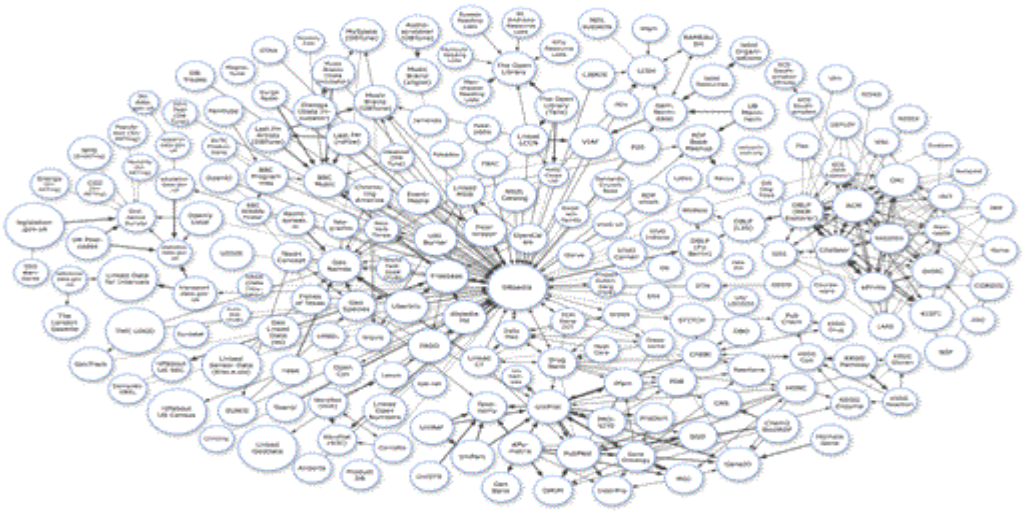
ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Διαχείριση Κατανεμημένων Δεδομένων στο Διαδίκτυο



Του φοιτητή

Τσουκαλά Χρυσόστομου

Αρ. Μητρώου: 05/2758

Επιβλέπων καθηγητής

Δημήτρης Αχιλ. Δέρβος

Θεσσαλονίκη 2011

ΠΡΟΛΟΓΟΣ

Από τότε που ο άνθρωπος άρχισε να εξελίσσεται και να κτίζει πολιτισμούς η αποθήκευση και διαχείριση των διάφορων δεδομένων και πληροφοριών που το περιέβαλαν αποτέλεσε ένα από τα κύρια προβλήματά του. Το συγκεκριμένο πρόβλημα έγινε ακόμα πιο έντονο με την ανακάλυψη των υπολογιστών και στη συνέχεια του Internet. Τη σημερινή εποχή είναι πολύ δύσκολο να διαχειριστούμε τα διάφορα δεδομένα τα οποία κατακλύζουν τον κόσμο γύρω μας. Έτσι η χρήση της τεχνολογίας για το σκοπό αυτό γίνεται αναγκαία σε όσο μεγαλύτερη κλίμακα μπορούμε να την χρησιμοποιήσουμε.

Στην συγκεκριμένη πτυχιακή εργασία παρουσιάζονται τεχνολογίες που μας βοηθούν στην διαχείριση των πληροφοριών οι οποίες βρίσκονται κατανεμημένες στο διαδίκτυο καθώς επίσης και κάποιες προτάσεις που σαν στόχο έχουν την εξόρυξη πληροφοριών από τα δεδομένα αυτά.

Η παρούσα πτυχιακή εργασία εκπονήθηκε στα πλαίσια του προγράμματος σπουδών του Τμήματος Πληροφορικής, της Σχολής Τεχνολογικών Εφαρμογών, του Αλεξάνδρειου Τεχνολογικού Εκπαιδευτικού Ιδρύματος Θεσσαλονίκης, υπό την επίβλεψη του κ. Δημήτρη Δέρβου, καθηγητή του τμήματος.

Ιδιαίτερες ευχαριστίες απευθύνονται στον επιβλέποντα καθηγητή για την καθοδήγησή του, για την παροχή βιβλιογραφικών πηγών καθώς και για την βοήθεια που μου παρείχε όπου αυτή του ζητήθηκε.

Τσουκαλάς Χρυσόστομος

ΠΕΡΙΛΗΨΗ

Οι πληροφορίες που βρίσκονται στο διαδίκτυο διαθέσιμες μπορούν να χρησιμοποιηθούν με πολλούς και διάφορους τρόπους αρκεί να μπορέσουμε να τις συλλέξουμε και να τις κατανοήσουμε. Κυρίως στόχος είναι η αποτελεσματική διαχείριση τους. Τις περισσότερες φορές όμως η διαδικασία αυτή δεν είναι καθόλου εύκολη καθώς οι πληροφορίες αυτές βρίσκονται αποθηκευμένες σε διαφορετικές πηγές, έχοντας διαφορετικές μορφές μεταξύ τους. Έτσι η συλλογή και η διαχείρισή τους δεν είναι καθόλου εύκολη. Επίσης αν αναλογιστούμε πόσες πληροφορίες προστίθενται καθημερινά στις ήδη υπάρχουσες και πως πρέπει συνεχώς να ανανεώνουμε τις πληροφορίες που ήδη έχουμε ή να τις εμπλουτίζουμε με καινούργιες, τα πράγματα δυσκολεύουν ακόμα περισσότερο.

Την λύση σε αυτά τα προβλήματα ήρθε να δώσει η ανάπτυξη τεχνολογιών και προτύπων για την διαχείριση των πληροφοριών αυτών, οι οποίες βρίσκονται κατανεμημένες στο διαδίκτυο. Μια τέτοια τεχνολογία είναι το RDF το οποίο είναι μια γλώσσα η οποία χρησιμοποιείται για την αναπαράσταση πληροφοριών στο διαδίκτυο, την αναπαράσταση των ιδιοτήτων τους και των σχέσεων που υπάρχουν ανάμεσά τους. Επίσης υπάρχουν γλώσσες όπως το RDF Schema, η OWL και η SKOS οι οποίες χρησιμοποιούνται για την αναπαράσταση τύπων και οντολογιών. Ακόμη υπάρχει μια γλώσσα για την διαχείριση των πληροφοριών αυτών, η SPARQL, με την οποία μπορούμε να εκτελέσουμε πολύ ισχυρά σημασιολογικά ερωτήματα στα δεδομένα αυτά, και να πάρουμε τις πληροφορίες που χρειαζόμαστε. Όλα αυτά αποτελούν τα Linked Data και απαρτίζουν ένα τεράστιο γράφο κατανεμημένων πληροφοριών στο διαδίκτυο. Έπειτα αν προσθέσουμε στα δεδομένα αυτά σημασιολογικό περιεχόμενο, έτσι ώστε οι υπολογιστές να έχουν την δυνατότητα να κατανοούν την σημασιολογία των πληροφοριών, έχουμε το Σημασιολογικό Διαδίκτυο (Semantic Web). Μέσω του σημασιολογικού διαδικτύου μπορούμε να «αναθέσουμε» στους υπολογιστές εργασίες όπως η κατανόηση των δεδομένων. Η διεργασία αυτή θα έπρεπε να εκτελεστεί από ανθρώπους κάτι που την καθιστά ιδιαίτερα χρονοβόρα, η εκτέλεσή της όμως από υπολογιστές ολοκληρώνεται πολύ γρηγορότερα. Επίσης μέσω του Σημασιολογικού διαδικτύου μπορούμε να προτείνουμε στον χρήστη πληροφορίες οι οποίες ίσως θα τον ενδιέφεραν.

Σε αυτή την πτυχιακή εργασία παρουσιάζονται και αναλύονται οι τεχνολογίες αυτές, καθώς και πώς εφαρμόζονται στο διαδίκτυο με την μορφή που έχει σήμερα. Τέλος παρουσιάζονται οι βάσεις γράφου, οι οποίες έχουν την δυνατότητα να αποθηκεύουν αυτές τις πληροφορίες, καθώς επίσης και κάποιες προτάσεις για την εξόρυξη πληροφοριών από δεδομένα τα οποία βρίσκονται σε μορφή γράφου.

ABSTRACT

The available information that already exists on the Web can be used in many ways. The most difficult part is to collect and understand them. Unfortunately most of the times this procedure is not so easy because this information are distributed in different sources, and they also have differences between them. In this way collecting and managing information is not so easy. Things are getting worse if we consider that we have to update them and add new information as frequently as we can.

The solution to this problem was given by the development of new technologies and prototypes for managing this information. Such technology is the RDF which is a language that is used for representing information on the Web, the properties that they have and the relations between them. Moreover there are languages such as RDF Schema, OWL and SKOS which are used for representing types and ontologies. Additionally there is one language to manage this information, SPARQL, with which we can perform very strong semantic queries in this data, and collect the information that we want. All above can be called Linked Data and they are forming a huge graph of distributed data on the Web. Furthermore if we add semantic meaning to these data, in order to the machines understand what these data means, we have the Semantic Web. With the Semantic Web we can give to the machines processes like understanding the data. This process should be done by the humans. By giving this process to the machine we can have the results in a very short time. As well with the Semantic Web we can propose to the user information that maybe he is interested in.

In this thesis I present and analyze these technologies and how they are used on the Web with the current form. Finally I present the graph databases which can store this information and also I present some ways in which we can do graph mining on the data which are in RDF/XML form.

ΕΥΧΑΡΙΣΤΙΕΣ

Θα ήθελα να ευχαριστήσω το *Khaos Group* του πανεπιστημίου της Μάλαγα για την ευκαιρία που μου δόθηκε να γνωρίσω και να ασχοληθώ με τις τεχνολογίες αυτές, καθώς επίσης τους καθηγητές μου στο τμήμα που μου πρόσφεραν τις βασικές γνώσεις γύρω από την επιστήμη των υπολογιστών.

Τέλος θέλω να ευχαριστήσω την οικογένειά μου γιατί πάντα με στήριζε και με στηρίζει στις επιλογές μου σωστές ή λάθος.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Την σημερινή εποχή ο όγκος των διαθέσιμων πληροφοριών και των πληροφοριών που διακινούνται στο διαδίκτυο είναι τεράστιος. Η εξέλιξη του Internet και η εισχώρησή του στην καθημερινή μας ζωή έχει συμβάλει σημαντικά σε αυτό. Έτσι η διαχείρισή και η κατανόηση των πληροφοριών είναι εξαιρετικά δύσκολη, πρακτικά αδύνατη θα μπορούσαμε να πούμε, χωρίς την βοήθεια της τεχνολογίας. Όταν δημιουργήθηκε το Διαδίκτυο ήταν ένα δίκτυο το οποίο βασιζόταν στο ARPANET, ένα δίκτυο που είχε αναπτυχθεί από το υπουργείο αμύνης των Ηνωμένων Πολιτειών Αμερικής, και συνέδεε συγκεκριμένα πανεπιστήμια με σκοπό την ανταλλαγή πληροφοριών μεταξύ τους. Το δίκτυο αυτό άρχισε να εξαπλώνεται όλο και περισσότερο, και εκτός από πανεπιστήμια άρχισαν να συνδέονται σε αυτό και κόμβοι οι οποίοι δεν ήταν πανεπιστημιακοί. Έτσι άρχισε να διευρύνεται παρέχοντας σε όποιον ήταν συνδεδεμένος σε αυτό, πρόσβαση σε πληροφορίες που δεν θα μπορούσε να έχει διαφορετικά.

Η μορφή που είχε τότε το διαδίκτυο είναι πολύ διαφορετική με την μορφή που έχει σήμερα. Στην αρχική του μορφή οι ιστοσελίδες παρουσιάζονταν σαν υπερκείμενα τα οποία είχαν συνδέσμους, οι οποίοι με την σειρά τους οδηγούσαν σε άλλα υπερκείμενα. Έτσι μπορούσες να πλοηγηθείς ανάμεσα σε στατικές ιστοσελίδες και να ανακτήσεις πληροφορίες. Αυτό αποκαλείται Web 1.0. Η ανάγκη όμως για περισσότερη λειτουργικότητα και το άνοιγμα του διαδικτύου στο ευρύτερο κοινό οδήγησαν στην εξέλιξή του και στην αντικατάσταση των στατικών ιστοσελίδων οι οποίες περιείχαν μόνο υπερσυνδέσμους, με ιστοσελίδες που επιτρέπουν στον χρήστη να αλληλεπιδρά και να συνεργάζεται ώστε να του παρουσιάζεται ένα user-generated περιεχόμενο σε μια ιστοσελίδα, και όχι κάτι το οποίο κατασκευάστηκε εκ των προτέρων για να του παρουσιαστεί. Με αυτό τον τρόπο άρχισαν να δημιουργούνται οι web εφαρμογές και το διαδίκτυο να γίνεται όλο και πιο προσιτό και πιο ελκυστικό για τους απλούς χρήστες. Αυτό αποτελεί το Web 2.0 και μερικά παραδείγματα αυτής της εξέλιξης του είναι: οι ιστοσελίδες κοινωνικής δικτύωσης (facebook.com, twiter.com), τα blocks, τα wikis

(Wikipedia.com), οι ιστοσελίδες για την προβολή βίντεο (YouTube.com) και οι web εφαρμογές.

Τα τελευταία χρόνια το διαδίκτυο τείνει να πάρει μια άλλη μορφή. Γίνεται προσπάθεια να δοθεί μια σημασιολογική έννοια στα δεδομένα που δημοσιεύονται στο διαδίκτυο, ώστε να γίνονται κατανοητά όχι μόνο από τους ανθρώπους αλλά και από τις μηχανές. Επίσης προωθείται η διασύνδεση μεταξύ των δημοσιευμένων δεδομένων, και σε συνδυασμό με την σημασιολογία που τους δίνεται μπορούν να δώσουν πραγματικό νόημα στα δεδομένα ενισχύοντάς τα όχι μόνο σημασιολογικά, αλλά και προσθέτοντας πληροφορίες σε αυτά οι οποίες βρίσκονται σε άλλα μέρη, διάσπαρτες στο διαδίκτυο. Οι πληροφορίες αυτές βρίσκονται κατανεμημένες στο διαδίκτυο, αποτελώντας ένα γιγάντιο κατευθυνόμενα συνδεδεμένο γράφο, ο οποίος προσφέρει πληροφορίες από διαφορετικές πηγές, με σκοπό την ποιο ολοκληρωμένη αναπαράσταση των δεδομένων. Αυτή η καινούργια μορφή του διαδικτύου τείνει να γίνει το Web 3.0.

Η αναπαράσταση καθώς και η διαχείριση αυτού του τεράστιου όγκου πληροφοριών που είναι διαθέσιμες, αποτελεί το ερώτημα που εξετάζεται παρακάτω στη πτυχιακή εργασία. Στα κεφάλαια που ακολουθούν εξετάζονται οι τεχνολογίες και τα στάνταρ που χρησιμοποιούνται, καθώς επίσης παρουσιάζονται εφαρμογές που τις εφαρμόζουν και είναι ανοιχτές για χρήση από τον καθένα μας, με στόχο την αποτελεσματικότερη και όσο το δυνατόν πιο ολοκληρωμένη συλλογή δεδομένων.

ΚΕΦΑΛΑΙΟ 1

URI

ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο αυτό παρουσιάζεται ο τρόπος για την αναπαράσταση των δεδομένων στο διαδίκτυο με μια μορφή η οποία είναι εύχρηστη από τους υπολογιστές, δίνοντας ένα μοναδικό προσδιοριστικό στις πληροφορίες που αναπαρίστανται και στις ιδιότητές τους.

1.1 Τι είναι το URI και πως συντάσσεται

Το URI (Uniform Resource Identifier) είναι ένα μοναδικό όνομα που χρησιμοποιείται για τον μοναδικό προσδιορισμό των πόρων στο διαδίκτυο. Επιτρέπει την ομοιόμορφη αναγνώριση των πόρων μέσω ενός ξεχωριστού καθορισμένου επεκτάσιμου συνόλου σχημάτων ονοματοδοσίας. Δεν τίθεται κανένα όριο σχετικά με τη φύση ενός πόρου ή τα είδη των συστημάτων που θα μπορούσαν να χρησιμοποιήσουν URIs για να βρουν πόρους αυτούς. Επίσης είναι ένας τρόπος αναπαράστασης δομών αρχείων, έτσι ώστε ο browser να γνωρίζει ποιά κομμάτια του αρχείου είναι παράγραφοι, ποιά είναι κεφαλίδες, ποιά είναι σύνδεσμοι και ούτω καθεξής. Τέλος είναι ένας συμπαγής τρόπος για ένα πρόγραμμα πελάτη να προσδιορίσει ποιο πόρο χρειάζεται συγκεκριμένα.

Το URI μπορεί να είναι ένα URL (Uniform Resource Locator), ένα URN (Uniform Resource Name) ή και τα δύο μαζί. Το URN αποτελεί το αναγνωριστικό του πόρου και το URL είναι η διαδρομή που πρέπει να ακολουθήσουμε για να το βρούμε. Κάθε URI ξεκινάει με ένα όνομα σχήματος (ή πιο απλά πρωτοκόλλου) που αναφέρεται σε μια προδιαγραφή για την ταξινόμηση των αναγνωριστικών μέσω αυτού του σχήματος. Το σχήμα που χρησιμοποιείται είναι το http, αλλά μπορεί να είναι επίσης και ftp, mailto, file κ.α. Μετά, το σχήμα ακολουθείται από το ιεραρχικό κομμάτι. Αυτό το κομμάτι του URI διατηρεί πληροφορίες για την ιεραρχία της διαδρομής που πρέπει να ακολουθήσουμε για να φτάσουμε σε ένα συγκεκριμένο πόρο. Το πεδίο αυτό χωρίζεται σε δύο μέρη. Το πρώτο μέρος είναι το μέρος δικαιοδοσίας στο οποίο περιέχεται η «αρχή της πληροφορίας», δηλαδή πληροφορίες πρόσβασης χρήστη, domain name και ο αριθμός της θύρας. Το

δεύτερο μέρος είναι το μονοπάτι (path) που πρέπει να ακολουθήσουμε για να φτάσουμε στην πληροφορία. Έπειτα από το ιεραρχικό κομμάτι ακολουθεί ένα προαιρετικό πεδίο, το πεδίο ερώτημα (query) το οποίο παρέχει επιπλέον πληροφορίες οι οποίες δεν είναι ιεραρχικής φύσης. Τέλος είναι το πεδίο fragment, το οποίο είναι και αυτό ένα προαιρετικό μέρος του URI και χωρίζεται από το υπόλοιπο μέρος του με το σύμβολο της δίεσης “#”. Το τελευταίο μέρος του URI περιέχει πληροφορίες αναγνώρισης οδηγώντας σε ένα δευτερεύοντα πόρο. Όταν ο κυρίως πόρος είναι μια ιστοσελίδα το fragment είναι συνήθως το id ενός χαρακτηριστικού ενός συγκεκριμένου στοιχείου και ο web browser θα κάνει ορατό αυτό το στοιχείο.

1.2 Namespaces

Ένα Namespace είναι ένα URI που έχει επιλεγεί για το χώρο ονομάτων ενός συγκεκριμένου XML λεξιλόγιο που περιγράφει έναν πόρο υπό τον έλεγχο του δημιουργού ή του οργανισμού ο οποίος καθορίζει το λεξιλόγιο αυτό, όπως μια διεύθυνση URL για το Web server. Ωστόσο τα namespaces δεν υποχρεώνουν, ούτε προτείνουν τα URI τους να χρησιμοποιηθούν για την ανάκτηση πληροφοριών. Απλά αντιμετωπίζονται από ένα XML parser σαν string και παρουσιάζουν το namespace στους ανθρώπους. Έτσι με την χρήση των namespaces λύθηκε και το πρόβλημα σύγχυσης των εννοιών που έχουν το ίδιο όνομα. Πολύ απλά ένας XML προγραμματιστής αναφέρεται στο συγκεκριμένο URI namespace και έτσι γίνεται ο διαχωρισμός μεταξύ των όρων που έχουν το ίδιο όνομα. Τέλος όπου μπορεί να χρησιμοποιηθεί ένα URI, μπορεί να χρησιμοποιηθεί και ένα prefix όνομα για το namespace ενός πόρου, και έτσι όταν είναι να τον χρησιμοποιήσουμε δεν χρειάζεται να γράφεται ολόκληρο το URI του παρά το συγκεκριμένο prefix, αρκεί να έχει οριστεί εξ αρχής.

ΕΠΙΛΟΓΟΣ

Τα URI μας βοηθούν στην μοναδική αναπαράσταση πληροφοριών στο διαδίκτυο και αυτό μας βοήθησε στην ανάπτυξη τεχνολογιών για την αναπαράσταση κατανεμημένων δεδομένων στο διαδίκτυο. Με τις τεχνολογίες αυτές μπορούμε να συνδέσουμε δεδομένα μεταξύ τους, δίνοντάς τους ιδιότητες.

ΚΕΦΑΛΑΙΟ 2

RDF

ΕΙΣΑΓΩΓΗ

Για την αναπαράσταση των πληροφοριών οι οποίες βρίσκονται κατανεμημένες στο διαδίκτυο χρειαζόμαστε ένα γενικό τρόπο ο οποίος θα είναι εύχρηστος από τους υπολογιστές. Επίσης πρέπει να έχουμε και την δυνατότητα να αναπαραστήσουμε τις ιδιότητες των πληροφοριών αυτών. Την λύση σε αυτό το πρόβλημα την έδωσε το RDF. Το RDF είναι ένα μοντέλο για την αναπαράσταση των πληροφοριών οι οποίες βρίσκονται κατανεμημένες στο διαδίκτυο, δίνοντάς τους ιδιότητες και τιμές στις ιδιότητες αυτές.

2.1 RDF μοντέλο

Το RDF (Resource Description Framework) δημιουργήθηκε το 1999 και είναι μια γλώσσα η οποία βασίζεται στην XML (αναφορά στο Παράρτημα Ι) και σχεδιάστηκε σαν μοντέλο μεταδεδομένων (metadata) για την περιγραφή, την αναπαράσταση και την μοντελοποίηση των πληροφοριών και των διαδικτυακών πόρων. Κατασκευάστηκε με σκοπό να γίνεται η διαχείριση των πληροφοριών από εφαρμογές και όχι μόνο για την αναπαράστασή τους. Παρέχει ένα κοινό πλαίσιο για την αναπαράσταση των πληροφοριών έτσι ώστε όταν γίνεται ανταλλαγή των πληροφοριών μεταξύ των εφαρμογών, να μην χάνεται η σημασιολογία τους. Η δυνατότητα να ανταλλάσσονται πληροφορίες μεταξύ των διάφορων εφαρμογών σημαίνει πως οι πληροφορίες μπορεί να είναι διαθέσιμες και σε διαφορετικές εφαρμογές από αυτές για τις οποίες δημιουργήθηκαν αρχικά.

Για να γίνει ο προσδιορισμός των εννοιών χρησιμοποιούνται Web identifiers τα λεγόμενα URIs (Uniform Resource Identifier). Έτσι με την χρήση των URIs αναπαρίστανται απλές δηλώσεις (statements) πόρων δημιουργώντας ένα γράφο από κόμβους, οι οποίοι ενώνονται μεταξύ τους με τόξα που αναπαριστούν τις ιδιότητες των πόρων και τις τιμές τους.

Οι πληροφορίες αναπαριστώνται σαν τριπλέτες οι οποίες περιγράφουν ένα πόρο. Η ιδέα της αναπαράστασης ενός πόρο με αυτή την μορφή είναι η εξής: αυτό που περιγράφεται έχει κάποιες ιδιότητες και αυτές οι ιδιότητες έχουν τιμές. Έτσι όλα αυτά μαζί (αυτό που περιγράφεται, οι ιδιότητες και οι τιμές αυτών των ιδιοτήτων) αποτελούν ένα πόρο. Για να αναφερθούμε στα κομμάτια του πόρου, το

RDF χρησιμοποιεί συγκεκριμένο λεξιλόγιο. Έτσι σαν υποκείμενο (subject) αποκαλούμε αυτό που περιγράφεται, σαν κατηγόρημα (predicate) την ιδιότητα ή το χαρακτηριστικό του υποκειμένου που προσδιορίζεται και η τιμή της ιδιότητας ή του χαρακτηριστικού του κατηγορήματος ονομάζεται αντικείμενο (object).

Για τον προσδιορισμό των υποκειμένων, των κατηγορημάτων και τον αντικειμένων χρησιμοποιούνται URI references (*URIref*). Το URIref είναι ένα URI που στο τέλος έχει το προεραιτικό αναγνωριστικό κομμάτι fragment, το οποίο χωρίζεται με δίεση (“#”) από το υπόλοιπο URI. Το RDF περιγράφει τους πόρους μέσω URI references, και έτσι τα URIrefs επιτρέπουν μέσω του RDF να περιγράφεται οτιδήποτε, και έτσι να δημιουργηθούν σχέσεις μεταξύ αυτών των πραγμάτων. Το υποκείμενο μπορεί να είναι URI ή ανώνυμος κόμβος, το κατηγόρημα είναι υποχρεωτικά URI που αντιπροσωπεύει ένα πόρο ο οποίος δίνει ένα γνώρισμα στο υποκείμενο, ενώ το αντικείμενο μπορεί να είναι URI, μια σταθερή τιμή (Literal) ή ένας ανώνυμος κόμβος.

Για την αναπαράσταση των δηλώσεων (statements) με τρόπο κατανοητό από τους υπολογιστές χρησιμοποιείται η XML. Χάρης της ελευθερίας που παρέχει η XML όσο αναφορά στο να δημιουργήσει ο καθένας τον δικό του τύπο αρχείων και να τον χρησιμοποιεί για να φτιάχνει τέτοια αρχεία, ορίστηκε η RDF/XML γλώσσα που χρησιμοποιείται για την αναπαράσταση των RDF πληροφοριών. Σε μια αναπαράσταση ενός RDF γράφου, οι κόμβοι που είναι URI αναπαριστώνται σαν ελλείψεις, ενώ οι σταθερές τιμές (literals) σαν ορθογώνια παραλληλόγραμμα. Τα κατηγορήματα αναπαρίστανται σαν τόξα που ενώνουν τους κόμβους μεταξύ τους. Έτσι δημιουργείται ένας κατευθυνόμενος γράφος ο οποίος αναπαριστά τις πληροφορίες.

Παρ’ όλα αυτά δεν είναι πάντα βολικό να σχεδιάζουμε γράφους για να αναπαριστούμε την πληροφορία, καθώς η αναπαράσταση σε γράφο πολλών πόρων, οι οποίοι μάλιστα συνδέονται και μεταξύ τους, δεν μας δίνει μια «καθαρή» εικόνα για τις πληροφορίες που έχουμε. Έτσι οι πόροι μπορούν να αναπαρασταθούν και με άλλους τρόπους. Οι τρόποι αυτοί για την αναπαράσταση των statements εκτός από την RDF/XML είναι Notation 3 (N3), Turtle, N-Triples. Η συχνότερη μορφή είναι η N-Triples που αποκαλείται και κοινός triples (τριπλέτες). Σε αυτή την μορφή κάθε statement γράφεται σαν μια απλή τριπλέτα που

αποτελείται από το υποκείμενο (subject), το κατηγόρημα (predicate) και το αντικείμενο (object).

Επίσης χρησιμοποιείται ένας τρόπος για να γράφονται συντομότερα τα URI έτσι ώστε να οργανώνονται καλύτερα τα URIRef σε ένα σχετικό σύνολο όρων. Στις συντομογραφίες χρησιμοποιείται ένα πρόθεμα (prefix) το οποίο έχει οριστεί από πριν σε ένα URI namespace στην κορυφή του RDF εγγράφου. Αυτό το πρόθεμα ακολουθείται από το σύμβολο ":" και έπειτα ακολουθεί το τοπικό όνομα. Έτσι όσες φορές χρησιμοποιείται το ίδιο namespace, μπορούμε να το αντικαθιστούμε ένα URI με το πρόθεμα που έχουμε ορίσει και μετά να ακολουθείται από τοπικό όνομα που προσδιορίζει το συγκεκριμένο URI. Π.χ. : έχουμε το URI http://example.com/someone/ και του αναθέτουμε το πρόθεμα exp. Όταν θέλουμε να αναπαραστήσουμε το URI http://example.com/someone/else μπορούμε να το γράψουμε με την συντομογραφία exp:else. Αυτό ο τρόπος συντομογραφίας των URI είναι απλά μια σύμβαση καθώς το RDF model δεν αναγνωρίζει τις συντομογραφίες, αλλά μόνο ολόκληρα τα URIrefs και δεν αναλύει την δομή τους. Έτσι δεν γίνεται καμία υπόθεση πως τα URI με ίδιο πρόθεμα έχουν κάποια σχέση μεταξύ τους. Ακόμα δεν μας λέει κάτι πως αυτά που έχουν διαφορετικό πρόθεμα μπορούν να θεωρηθούν πως ανήκουν σε διαφορετικό λεξιλόγιο από namespaces.

2.2 Ανώνυμοι κόμβοι

Αν χρειάζεται να αναπαραστήσουμε μια σύνθετη πληροφορία χωρίς να χάσει την σημασία της, π.χ. αν θέλουμε να αναπαραστήσουμε μια πληροφορία για μια ημερομηνία και θέλουμε να αναπαραστήσουμε ξεχωριστά τον μήνα, την μέρα, το έτος, θα μπορούσαμε να δημιουργήσουμε ξεχωριστούς κόμβους για κάθε κομμάτι πληροφορίας και να τα συνδέσουμε με ένα νέο κόμβο ο οποίος συνδέει το αρχικό υποκείμενο και έχει σαν αντικείμενα, τους νέος αυτούς κόμβους. Επειδή όμως αυτός ο νέος κόμβος ενδέχεται να μην χρειαστεί να γίνει αναφορά σε αυτών έξω από τον γράφο δεν είναι απαραίτητο να του δώσουμε κάποιο URI έτσι ώστε να είναι μοναδικός. Μπορεί να είναι ένας κόμβος χωρίς όνομα που συνδέεται με το αρχικό κόμβο και αποτελεί ένα σύνθετο πόρο ο οποίος περιέχει το σύνολο της

πληροφορίας. Αυτός ο κόμβος ονομάζεται κενός κόμβος (blank node) ή και ανώνυμος πόρος. Σε μια γραφική αναπαράσταση του γράφου είναι εύκολο να παρουσιαστεί ένας ανώνυμος κόμβος. Στην περίπτωση που θέλουμε να αναπαραστήσουμε τον γράφο σε τριπλέτες πρέπει να αναπαραστήσουμε με κάποιο τρόπο και τους ανώνυμους κόμβους, και επειδή μπορεί να υπάρχουν περισσότεροι από ένας ανώνυμοι κόμβοι τους αναπαριστούμε δίνοντάς τους ένα μοναδικό όνομα. Η μορφή του ονόματος είναι _:“name” για να ξεχωρίζουν μεταξύ τους. Τα ονόματα αυτά χρησιμοποιούνται αποκλειστικά μόνο για την αναπαράσταση των ανώνυμων κόμβων όταν αναπαριστούμε των γράφο σε μορφή τριπλετών καθώς όταν γίνεται γραφική αναπαράσταση του γράφου οι ανώνυμοι κόμβοι δεν έχουν όνομα. Κάθε όνομα που χρησιμοποιείται για τον προσδιορισμό των ανώνυμων κόμβων όταν αναπαρίστανται σε τριπλέτες είναι διακριτό. Αν υπάρχει περίπτωση να γίνει αναφορά σε ένα ανώνυμο κόμβο έξω από τον γράφο, τότε πρέπει να προσδιοριστεί μέσω ενός URIref το οποίο θα δοθεί στον ανώνυμο κόμβο. Τέλος ένας ανώνυμος κόμβος λόγω του ότι αναπαριστά ένα κόμβο, μπορεί να είναι είτε υποκείμενο είτε αντικείμενο, και όχι κατηγόρημα.

2.3 Τύποι δεδομένων (Datatypes)

Οι τύποι δεδομένων χρησιμοποιούνται από το RDF για να αναπαραστήσουν τιμές όπως ακέραιοι, αριθμοί κινητής υποδιαστολής και ημερομηνίες. Η αφαίρεση (abstraction) των τύπων δεδομένων που χρησιμοποιείται στο RDF είναι συμβατή με αυτή που χρησιμοποιείται στην XML. Ένας τύπος δεδομένων (datatype) αποτελείται από ένα λεξιλογικό χώρο (lexical space), ένα εύρος τιμών (value space) και μια λεξιλογική σε τιμή χαρτογράφηση (lexical-to-value mapping).

Ο λεξιλογικός χώρος είναι το σύνολο όλων των συμβολοσειρών, το εύρος τιμών είναι ένα σύνολο οντοτήτων που ονομάζονται και τιμές XML, και η λεξιλογική σε τιμή χαρτογράφηση είναι μια «ένα προς ένα» χαρτογράφηση από τον λεξιλογικό χώρο στο εύρος τιμών.

2.4 Literals

Τα κυριολεκτικά (Literals) έχουν λεκτικό χαρακτήρα και αποτελούνται από μια Unicode συμβολοσειρά. Τα Literal χωρίζονται σε δύο κατηγορίες: τα απλά (plain literals) και τα typed literal.

Τα απλά Literal αποτελούνται από ένα λεκτικό τύπο και προαιρετικά ένα αναγνωριστικό γλώσσας (language tag) με πεζά γράμματα. Π.χ.

<http://example.com/city/Thessaloniki> <http://example.com/city#name> “Thessaloniki”@en.

Ένα typed literal αποτελείται από ένα λεκτικό τύπο και ένα URI που προσδιορίζει ένα τύπο δεδομένων. Έτσι ένα typed literal παρουσιάζεται σαν ένας ενιαίος κυριολεκτικός κόμβος (literal node) στο RDF γράφημα αποτελούμενος από το λεκτικό κομμάτι και το URI που προσδιορίζει των τύπο δεδομένων και παρουσιάζεται όλο μαζί σαν κυριολεκτικό (literal). Π.χ.

<http://example.com/person#34> <http://example.com/age> "30"^^<http://www.w3.org/2001/XMLSchema#integer>.

Δυστυχώς το RDF σε αντίθεση με τις γλώσσες προγραμματισμού και τις βάσεις δεδομένων, δεν έχει κάποιο ενσωματωμένο σύνολο τύπων δεδομένων, όπως integer ή string. Αντί αυτού τα typed literals παρέχουν ένα τρόπο σύμφωνα με τον οποίο, ένα δεδομένο literal, είναι ο τύπος δεδομένων που πρέπει να χρησιμοποιηθεί προκειμένου να ερμηνευτεί. Αυτοί οι τύποι που χρησιμοποιούνται στα typed literals, ορίζονται έξω από το RDF και προσδιορίζονται από το datatype URI τους. Το πλεονέκτημα αυτής της μεθόδου είναι ότι το RDF δίνει την δυνατότητα να αναπαρίστανται πληροφορίες, που προέρχονται σε διαφορετικές πηγές δεδομένων, χωρίς να χρειάζεται να μετατραπούν οι διάφοροι τύποι δεδομένων από τις πηγές αυτές και τους τύπους δεδομένων ενός τοπικού RDF αρχείου. Δεν είναι όλοι οι τύποι δεδομένων κατάλληλη για να χρησιμοποιηθούν από το RDF. Για να είναι ένας τύπος δεδομένων κατάλληλος ώστε να χρησιμοποιηθεί από το RDF, θα πρέπει να επιβεβαιώνει το εννοιολογικό πλαίσιο

που περιγράφτηκε παραπάνω. Αυτό τυπικά σημαίνει ότι, ο τύπος δεδομένων θα πρέπει απερίφραστα να ορίζεται από το URI το οποίο ακολουθείται.

2.5 RDF containers

Το RDF παρέχει ένα αριθμό από επιπλέον δυνατότητες, όπως ενσωματωμένοι τύποι και ιδιότητες για την αναπαράσταση ομάδων από πόρους και RDF δηλώσεις (statements), και δυνατότητες για την αναπαράσταση των XML fragments σαν τιμές ιδιοτήτων.

Συχνά παρουσιάζεται η ανάγκη να περιγραφούν ομάδες πραγμάτων π.χ. ένα βιβλίο γράφτηκε από μια ομάδα συγγραφέων. Το RDF παρέχει ορισμένους ενσωματωμένους τύπους και ιδιότητες που μπορούν να χρησιμοποιηθούν για να περιγράψουν μια τέτοια ομάδα. Το RDF παρέχει ένα container λεξιλόγιο που αποτελείται από τρείς προκαθορισμένους τύπους. Ένα container είναι ένας πόρος ο οποίος περιέχει πράγματα. Τα περιεχόμενα πράγματα ονομάζονται μέλη (members). Τα members ενός container μπορεί να είναι πόροι (συμπεριλαμβανομένων και των κενών κόμβων) ή literal. Το RDF ορίζει τρεις τύπους από containers:

- rdf:Bag
- rdf:Seq
- rdf:Alt

❖ Ένας σάκος (Bag) αναπαριστά μια ομάδα από πόρους ή literal, που πιθανών περιλαμβάνουν διπλά members και δεν έχει σημασία η σειρά των μελών. Π.χ. ένας Bag μπορεί να περιγράφει ένα σύνολο αριθμών των οποίων η σειρά εισαγωγής ή επεξεργασίας τους δεν έχει σημασία.

❖ Μια ακολουθία (Sequence ή Seq) αναπαριστά μια ομάδα από πόρους ή literal, που πιθανότατα περιέχει διπλά μέλη (members), όπου η σειρά των μελών είναι σημαντική. Π.χ. μια ακολουθία μπορεί να χρησιμοποιηθεί για να περιγράψει μια ομάδα η οποία πρέπει να διατηρείται με αλφαβητική σειρά.

❖ Μια εναλλαγή (Alternative ή Alt) αναπαριστά μια ομάδα από πόρους ή literal τα οποία είναι εναλλακτικές τιμές μιας μοναδικής τιμής κάποιας ιδιότητας. Π.χ. ένα alt μπορεί να χρησιμοποιηθεί για να περιγράψει εναλλακτικές γλώσσες μετάφρασης για τους τίτλους ενός βιβλίου, ή για να περιγράψουν μια λίστα από εναλλακτικές ιστοσελίδες στις οποίες ένας πόρος μπορεί να βρεθεί. Μια εφαρμογή η οποία χρησιμοποιεί μια ιδιότητα της οποίας η τιμή είναι ένα alt container, θα πρέπει να γνωρίζει ότι μπορεί να επιλέξει οποιοδήποτε από τα μέλη της ομάδας ανάλογα με την περίπτωση.

Για να περιγραφεί ένας πόρος σαν τύπος δοχείο (container), δίνεται στο συγκεκριμένο πόρο μια ιδιότητα rdf:type της οποίας η τιμή είναι ένας από τους προκαθορισμένους τύπους rdf:Bag, rdf:Seq, rdf:Alt. Ο πόρος δοχείο (container) ο οποίος είναι πόρος ή κενός κόμβος, δηλώνει την ομάδα σαν σύνολο. Τα μέλη (members) αυτού του δοχείου (container) μπορεί να περιγραφούν ορίζοντας μία ιδιότητα περιεχόμενου μέλους (*container membership property*) για κάθε μέλος με ένα περιεχόμενο πόρο σαν το υποκείμενό του, και το μέλος σαν το αντικείμενό του. Αυτές οι ιδιότητες περιεχόμενου μέλους έχουν ονόματα της μορφής rdf:_n, όπου n είναι ένας ακέραιος μεγαλύτερος του μηδενός, και χρησιμοποιείτε ειδικά για να περιγράφει τα μέλη του δοχείου (container). Οι περιεχόμενοι πόροι (container resources) μπορεί επίσης να έχουν και άλλες ιδιότητες οι οποίες περιγράφουν το δοχείο (container), επιπρόσθετα από τις ιδιότητες περιεχόμενου μέλους και την ιδιότητα rdf:type.

Μερικές φορές υπάρχουν ξεκάθαρες διαφορές στην χρήση των RDF τύπων δοχείων(container types). Για παράδειγμα μια σχέση μεταξύ ενός συγκεκριμένου πόρου και μιας ομάδας από άλλους πόρους μπορεί να αναφέρεται θέτοντας τον πρώτο πόρο σαν υποκείμενο πολλαπλών statements χρησιμοποιώντας την ίδια ιδιότητα. Αυτό είναι δομικά διαφορετικό από το ένας πόρος να είναι υποκείμενο μιας statement της οποίας αντικείμενο είναι ένα δοχείο (container) το οποίο περιέχει πολλαπλά μέλη (members). Σε μερικές περιπτώσεις, αυτές οι δύο δομές μπορεί να έχουν ίδια σημασία, αλλά σε άλλες όχι. Η επιλογή στο τι θα χρησιμοποιηθεί σε μια συγκεκριμένη περίπτωση πρέπει να γίνει έχοντας κατά νου τις διαφορές και τα πλεονεκτήματα της κάθε μεθόδου.

Όταν χρησιμοποιούνται τα RDF δοχεία (containers), είναι σημαντικό να καταλάβουμε ότι οι statements δεν είναι κατασκευαστικά δοχεία (constructing containers), όπως οι δομές δεδομένων σε μια γλώσσα προγραμματισμού. Οι statements είναι περιγραφικά δοχεία (describing containers) τα οποία δεν χρειάζεται να δημιουργήσουν μια δομή δεδομένων για να «κρατήσουν» τα μέλη μιας ομάδας. Αντί αυτού, περιγράφονται σαν χαρακτηρίστηκα τα οποία έχει στην κατοχή του ένας πόρος, ονομάζοντας τα μέλη τα οποία κατέχει. Αυτό δεν σημαίνει απαραίτητα ότι τα χαρακτηριστικά που περιγράφονται σαν μέλη (members) είναι τα μόνα μέλη που κατέχει ένας πόρος. Παρόλα αυτά είναι σημαντικό να καταλάβουμε ότι το RDF δεν μας αναγκάζει να χρησιμοποιήσουμε το συγκεκριμένο λεξιλόγιο δοχείων (RDF container vocabulary), και έτσι είναι δυνατό να χρησιμοποιήσουμε αυτό το λεξιλόγιο και με άλλους τρόπους. Για παράδειγμα σε μερικές περιπτώσεις μπορεί να είναι καταλληλότερο να χρησιμοποιήσουμε ένα πόρο δοχείο ο οποίος έχει ένα URIref από το να χρησιμοποιήσουμε ένα κενό κόμβο (blank node). Ακόμα είναι δυνατό να χρησιμοποιήσουμε το λεξιλόγιο δοχείων με τρόπους οι οποίοι μπορεί να μην περιγράφουν καλά δομημένους γράφους.

ΕΠΙΛΟΓΟΣ

Με το μοντέλο RDF το οποίο βασίζεται στην XML μπορούμε να αναπαραστήσουμε πληροφορίες οι οποίες βρίσκονται κατανεμημένες στο διαδίκτυο και οι πληροφορίες αυτές γίνονται κατανοητές από τους υπολογιστές. Έτσι μπορούμε να τις επεξεργαστούμε και να τις διαχειριστούμε όπως ακριβώς θέλουμε. Για την καλύτερη περιγραφή των τύπων δεδομένων που χρησιμοποιεί το μοντέλο RDF, έχουν αναπτυχθεί ειδικές γλώσσες για την περιγραφή τους.

ΚΕΦΑΛΑΙΟ 3

ΓΛΩΣΣΕΣ ΠΕΡΙΓΡΑΦΗΣ ΤΥΠΩΝ

ΕΙΣΑΓΩΓΗ

Οι γλώσσες περιγραφείς τύπων είναι γλώσσες οι οποίες περιγράφουν τύπους δεδομένων και οντότητες, για την καλύτερη και πληρέστερη περιγραφή των πληροφοριών που αναπαριστώνται μέσω του μοντέλου RDF. Κάθε μια έχει τον δικό της ρόλο στην αναπαράσταση και την περιγραφή των τύπων αυτών.

3.1 RDF Schema

3.1.1 Τι είναι το RDF Schema, αρχιτεκτονική και ιδιότητες

Το RDF δεν παρέχει κάποιο μηχανισμό για την περιγραφή ιδιοτήτων, και επίσης δεν παρέχει κάποιο μηχανισμό για να περιγράφει τις σχέσεις ανάμεσα σε ιδιότητες και άλλους πόρους. Αυτός είναι ο ρόλος του RDF Vocabulary Description Language, ή πιο απλά RDF Schema. Το RDF Schema ορίζει κλάσεις και ιδιότητες οι οποίες μπορεί να χρησιμοποιηθούν για να προσδιορίσουν κλάσεις, ιδιότητες και άλλους πόρους. Το RDF Schema δεν προσδιορίζει κάποιο λεξιλόγιο από περιγραφικές ιδιότητες όπως «φοιτητής». Αντί γι' αυτό προσδιορίζει ένα μηχανισμό ο οποίος χρησιμοποιείτε για να ονομάζει και να περιγράφει ιδιότητες και κλάσεις πόρων που περιγράφουν. Το RDF Schema είναι μια επέκταση του RDF. Παρέχει ένα μηχανισμό για την περιγραφή ομάδων από συσχετιζόμενους πόρους και τις σχέσεις ανάμεσά τους. Οι περιγραφές λεξιλογίου του RDF Schema γράφονται σε μορφή RDF. Οι πόροι που περιγράφονται με το RDF Schema χρησιμοποιούνται για να προσδιορίζουν χαρακτηριστικά άλλων πόρων, όπως domains και ranges των ιδιοτήτων. Το σύστημα περιγραφής των κλάσεων και των ιδιοτήτων που χρησιμοποιεί το RDF Schema είναι παρόμοιο με τις γλώσσες αντικειμενοστραφής προγραμματισμού όπως η Java. Το RDF διαφέρει από πολλά τέτοια συστήματα στα οποία, αντί να καθορίσουμε μια κλάση όσον αφορά τις ιδιότητες που μπορεί να έχει, η γλώσσα RDF λεξιλογίου περιγραφής (RDF Schema) περιγράφει τις ιδιότητες με αναφορά στις κλάσεις των πόρων για τις οποίες ισχύουν, χρησιμοποιώντας την RDF προσέγγιση, και είναι εύκολο να προσδιορίσουν. Πλεονέκτημα αυτής της αρχιτεκτονικής ιδιοτήτων είναι ότι επιτρέπει στον οποιονδήποτε να επεκτείνει την περιγραφεί των ήδη υπαρχόντων πόρων, κάτι το οποίο είναι μια αρχιτεκτονική αρχή του διαδικτύου. Οι σχεδιαστές του RDF λεξιλογίου μπορούν να δημιουργήσουν και να αναπτύξουν εφαρμογές

σημασιολογικού διαδικτύου (Semantic Web applications) χρησιμοποιώντας το RDF Schema. Το κυρίως λεξιλόγιο ορίζεται σε ένα namespace με το την συντομογραφία rdfs. Αυτό το namespace προσδιορίζεται από το URIref Reference http://www.w3.org/2000/01/rdf-schema# και σχετίζεται με το πρόθεμα rdfs. Επίσης αυτό σχετίζεται και χρησιμοποιεί το πρόθεμα rdf το οποίο αναφέρεται στο RDF namespace http://www.w3.org/1999/02/22-rdf-syntax-ns#.

3.1.2 Κλάσεις

Οι πόροι μπορεί να χωριστούν σε ομάδες που ονομάζονται κλάσεις. Τα μέλη των κλάσεων είναι γνωστά σαν instances (στιγμιότυπα) των κλάσεων. Οι κλάσεις είναι από μόνες τους πόροι. Συχνά προσδιορίζονται από URIref και ίσως περιγράφονται χρησιμοποιώντας RDF ιδιότητες. Η rdf:type ιδιότητα μπορεί να χρησιμοποιηθεί για να δηλώσει ότι ένας πόρος είναι ένα στιγμιότυπο μιας κλάσεις.

Μεταξύ του RDF διακρίνεται μια κλάση και μια ομάδα των στιγμιότυπων της. Αυτή η ομάδα συνδέεται με κάθε κλάση σαν ένα σύνολο, το οποίο ονομάζεται κλάση επέκτασης (class extension) της κλάσης, η οποία είναι ένα σύνολο από στιγμιότυπα κλάσεων. Δύο κλάσεις μπορεί να έχουν το ίδιο σύνολο στιγμιότυπων αλλά να είναι δυο διαφορετικές κλάσεις. Μια κλάση μπορεί να είναι μέλος της ίδιας της επέκτασης κλάσης και επίσης μπορεί να είναι στιγμιότυπο του εαυτού της. Η ομάδα των πόρων οι οποίες είναι κλάσεις του RDF Schema είναι από μόνες τους μια κλάση και ονομάζονται rdfs:Class.

- rdfs:Class

Αυτή είναι η κλάση των πόρων οι οποίοι είναι RDF κλάσεις. Το rdfs:Class είναι ένα στιγμιότυπο του rdfs:Class.

- rdfs:Resource

Όλα όσα περιγράφονται από το RDF ονομάζονται πόροι (resources), και είναι στιγμιότυπα της κλάσης rdfs:Resource. Αυτή είναι η κλάση όλων πόρων . Όλες οι άλλες κλάσεις είναι υποκλάσεις αυτής της κλάσης. Το Rdfs:Resource είναι στιγμιότυπο του rdfs:Class.

- rdfs:Literal

Η κλάση rdfs:Literal είναι η κλάση τον κυριολεκτικών (literal) τιμών όπως τα string και οι ακέραιοι. Τα κυριολεκτικά (literal) μπορεί να είναι plain ή typed. Ένα typed literal είναι ένα στιγμιότυπο μιας datatype κλάσης. Η προδιαγραφή αυτή δεν ορίζει την κλάση των plain literals. Το rdfs:Literal είναι ένα στιγμιότυπο του rdfs:Class και υποκλάση του, rdfs:Resource.

- rdfs:Datatype

Το rdfs:Datatype είναι μια κλάση που περιγράφει ένα τύπο δεδομένων. Το rdfs:Datatype είναι μια υποκλάση του rdfs:Class και κάθε στιγμιότυπο του rdfs:Datatype είναι μια υποκλάση του rdfs:Literal.

- rdf:XMLLiteral

Η κλάση rdf:XMLLiteral είναι η κλάση των XML literal values. Το rdf:XMLLiteral είναι ένα στιγμιότυπο του rdf:Datatype και είναι υποκλάση του rdfs:Literal

- rdf:Property

Το rdf:Property είναι η κλάση των ιδιοτήτων του RDF. Το rdf:Property είναι ένα στιγμιότυπο του rdfs:Class.

3.1.3 Ιδιότητες

Μια ιδιότητα περιγράφεται σαν μια σχέση ανάμεσα στον πόρο υποκείμενο και στο πόρο αντικείμενο. Αυτός ο προσδιορισμός ορίζει την γενική ιδέα των υπό-ιδιοτήτων (subproperty). Η ιδιότητα rdfs:subPropertyOf μπορεί να χρησιμοποιηθεί για να δηλώσει ότι μια ιδιότητα είναι υπό-ιδιότητα (subproperty) μιας άλλης. Αν μια ιδιότητα P1 είναι υπό-ιδιότητα μιας ιδιότητας P2, τότε όλα τα ζευγάρια των πόρων τα οποία συσχετίζονται με την ιδιότητα P1 συσχετίζονται επίσης και με την ιδιότητα P2. Ο όρος υπέρ-ιδιότητα (super-property) συνήθως χρησιμοποιείται σαν αντίστροφο της υπό-ιδιότητας. Αν μια ιδιότητα P3 είναι υπερ-ιδιότητα της ιδιότητας P1, τότε όλα τα ζευγάρια των πόρων τα οποία συσχετίζονται με τον P1

συσχετίζονται επίσης και με το P3. Αυτός ο προσδιορισμός δεν προσδιορίζει μια κορυφαία ιδιότητα η οποία είναι υπέρ-ιδιότητα όλων των υπόλοιπων ιδιοτήτων.

- rdfs:range

Το rdfs:range είναι μια αναφορά του rdf:Property το οποίο χρησιμοποιείται για να δηλώσει ότι οι τιμές μιας ιδιότητας είναι αναφορές μίας ή παραπάνω κλάσεων.

- rdfs:domain

Το rdfs:domain είναι μια αναφορά του rdf:Property και χρησιμοποιείται για να δηλώσει ότι ένας πόρος ο οποίος έχει μια δοσμένη ιδιότητα είναι αναφορά μιας η παραπάνω κλάσεων.

- rdf:type

Το rdf:type είναι μια αναφορά του rdf:Property η οποία χρησιμοποιείται για να δηλώσει ότι ένας πόρος είναι αναφορά μιας κλάσης.

- rdfs:subClassOf

Η ιδιότητα rdfs:subClassOf είναι μια αναφορά του rdf:Property η οποία χρησιμοποιείται για να δηλώσει ότι όλες οι αναφορές μιας κλάσης είναι αναφορές μιας άλλης.

Το rdfs:domain του rdfs:subClassOf είναι rdfs:Class. Το rdfs:range του rdfs:subClassOf είναι rdfs:Class.

- rdfs:subPropertyOf

Η ιδιότητα rdfs:subPropertyOf είναι μια αναφορά της κλάσης rdf:Property η οποία χρησιμοποιείται για να δηλώσει ότι όλοι οι πόροι που συσχετίζονται με μια ιδιότητα συσχετίζονται επίσης και με μια άλλη διαφορετική.

Το rdfs:domain του rdfs:subPropertyOf είναι rdf:Property και το rdfs:Range του είναι rdf:Property.

- rdfs:label

Το rdfs:label είναι μια αναφορά του rdf:Property το οποίο μπορεί να παρέχει μια εκδοχή του ονόματος του πόρου η οποία μπορεί να διαβαστεί από τον άνθρωπο.

Το rdfs:domain του rdfs:label είναι rdfs:Resource και το rdfs:range του είναι rdfs:Literal. Επίσης υποστηρίζονται οι πολύγλωσσες ετικέτες με την χρήση των ετικετών γλώσσας που χρησιμοποιούν τα κυριολεκτικά (Literals) στο RDF.

- rdfs:comment

Το rdfs:comment είναι μια αναφορά του rdf:Property η οποία μπορεί να χρησιμοποιηθεί για να παρέχει μια περιγραφή ενός πόρου η οποία είναι κατανοητή από τον άνθρωπο. Το rdfs:domain του rdfs:comment είναι rdfs:Resource και το rdfs:range είναι rdfs:Literal.

3.1.4 Utility Properties

- rdfs:seeAlso

Το rdfs:seeAlso είναι μια αναφορά της κλάσης rdf:Property η οποία χρησιμοποιείται για να υποδηλώσει ότι ένας πόρος μπορεί να παρέχει και πρόσθετες πληροφορίες για τον υποκείμενο πόρο.

- Rdfs:isDefinedBy

Το rdfs:isDefinedBy είναι μια αναφορά της κλάσης rdf:Property η οποία χρησιμοποιείται για να υποδηλώσει ένα πόρο που προσδιορίζει τον υποκείμενο πόρο. Αυτή η ιδιότητα μπορεί να υποδηλώσει ένα RDF λεξιλόγιο στο οποίο περιγράφεται ένας πόρος.

3.2 OWL

Η Web Ontology Language (OWL) είναι μια γλώσσα για τον προσδιορισμό και την δημιουργία στιγμιοτύπων οντολογιών στο διαδίκτυο. Οντολογία είναι ένας όρος δανεισμένος από την φιλοσοφία ο οποίος αναφέρεται στην επιστήμη για να

περιγράψει τα ήδη των οντοτήτων στον κόσμο και πως συσχετίζονται μεταξύ τους. Μία οντολογία OWL μπορεί να περιέχει περιγραφές κλάσεων, ιδιοτήτων και των αναφορών τους. Η OWL αποτελεί μια επέκταση του RDF και του RDFS. Πρωταρχικός της στόχος είναι να φέρει την εκφραστική δύναμη και το σκεπτικό της λογικής περιγραφής στο Σημασιολογικό Ιστό (Semantic Web). Δυστυχώς δεν μπορούν να περιγραφούν όλα στο RDF με την OWL. Γι' αυτό το λόγω μπορεί να είναι μόνο μια συντακτική επέκταση του RDF και του RDFS. Για να ξεπεραστεί εν μέρει το πρόβλημα αυτό, και επίσης να επιτρέψει την διαστρωμάτωση εντός της OWL, ορίζονται τρία είδη της OWL.

- OWL Lite
- OWL DL
- OWL Full

Η OWL χρησιμοποιεί για την σύνταξή της την XML. Έτσι οι πληροφορίες της OWL μπορούν εύκολα να ανταλλάσσονται μεταξύ των διαφόρων τύπων των υπολογιστών που χρησιμοποιούν διαφορετικούς τύπους λειτουργικών συστημάτων και των γλωσσών εφαρμογής.

Η OWL είναι μέρος της λειτουργίας του σημασιολογικού διαδικτύου (Semantic Web). Η προσπάθεια αυτή έχει στόχο να καταστήσει τους πόρους του διαδικτύου πιο εύκολα προσιτούς σε αυτοματοποιημένες διαδικασίες, με την προσθήκη πληροφοριών σχετικά με τους πόρους που περιγράφουν ή παρέχοντας περιεχόμενο στον Παγκόσμιο Ιστό. Όσο το σημασιολογικό διαδίκτυο είναι εγγενώς κατανεμημένο, η OWL πρέπει να επιτρέπει στις πληροφορίες να μπορούμε να τις συλλέξουμε από κατανεμημένους πόρους. Αυτό εν μέρει επιτυγχάνεται με το ότι οι οντολογίες συσχετίζονται μεταξύ τους, συμπεριλαμβάνοντας την εισαγωγή πληροφοριών από άλλες οντολογίες.

Προκειμένου να γράψουμε μια οντολογία η οποία μπορεί να ερμηνευτεί και να χρησιμοποιηθεί από τους software agents, απαιτείται μια σύνταξη και μια επίσημη σημασιολογία της OWL.

Η OWL επεκτείνει την περιγραφικότητα του RDFS με επιπλέον ερχέτυπα μοντελοποίησης. Παραδείγματος χάριν, η OWL προσδιορίζει τα αρχέτυπα

owl:equivalentClass και owl:equivalentProperty. Όταν συνδυαστούν με τα rdfs:subClassOf και rdfs:subPropertyOf, μας παρέχουν ένα πολύ ισχυρό μηχανισμό για τον προσδιορισμό της χαρτογράφησης ανάμεσα σε όρους από διάφορα λεξικά, τα οποία μπορούν να αυξήσουν τη διαλειτουργικότητα των συνόλων δεδομένων που διαμόρφωσε τη χρήση διαφορετικών λεξιλογίων. Ένα άλλο αρχέτυπο μοντέλο της OWL το οποίο είναι πολύ χρήσιμο στο πλαίσιο των δεδομένων του Web είναι owl:InverseFunctionalProperty. Οι τιμές των ιδιοτήτων οι οποίες έχουν προσδιοριστεί ώστε να έχουν αντίστροφη λειτουργικότητα προσδιορίζονται μοναδικά έχοντας το συγκεκριμένο αρχέτυπο ως κατηγόρημα.

Το σημασιολογικό διαδίκτυο είναι μια προοπτική για το μέλλον του διαδικτύου κατά το οποίο οι πληροφορίες λαμβάνουν σαφή έννοια, και έτσι κάνει ευκολότερο για τις μηχανές να αυτοματοποιήσουν την διαδικασία και να ενσωματώσουν τις διαθέσιμες πληροφορίες στο διαδίκτυο. Το σημασιολογικό διαδίκτυο (Semantic Web) «χτίζεται» χρησιμοποιώντας την ικανότητα της XML να προσδιορίζει προσαρμοσμένα σχήματα σήμανσης και την ευέλικτη προσέγγιση του RDF για την αναπαράσταση των δεδομένων. Το πρώτο επίπεδο πάνω από το απαιτούμενο RDF για το σημασιολογικό διαδίκτυο είναι μια γλώσσα οντολογιών οι οποία μπορεί να περιγράψει επίσημα την ορολογία που χρησιμοποιείται σε έγγραφα του Διαδικτύου. Αν αναμένεται από τις μηχανές να εκτελέσουν χρήσιμες εργασίες ανάλυσης σε αυτά τα έγγραφα, η γλώσσα αυτή πρέπει να είναι πέρα από τη βασική σημασιολογία του RDF Schema. Η OWL μας παρέχει αυτή την δυνατότητα καθώς επίσης και τα εργαλεία για να προσδιορίζουμε τις διάφορες οντότητες έτσι ώστε να μπορούν να χρησιμοποιηθούν αποτελεσματικά από τις μηχανές.

3.3 FOAF

3.3.1 Τι είναι το FOAF

Το Friend of a friend (FOAF) project δημιουργεί ένα ιστό από σελίδες οι οποίες γίνονται κατανοητές από τις μηχανές και περιγράφουν ανθρώπους, τις σχέσεις

μεταξύ τους και τα πράγματα τα οποία δημιουργούν και κάνουν. Το FOAF προσδιορίζει μια ανοιχτή, αποκεντρωμένη τεχνολογία για να συνδέει κοινωνικά τις ιστοσελίδες, και τους ανθρώπους τους οποίους περιγράφει. Η περιγραφή αυτή αφορά το μέρος αυτού που περιγράφει στον παγκόσμιο ιστό, και το μέρος του παγκόσμιου ιστού στον κόσμο μας. Είναι μια απλή τεχνολογία που κάνει εύκολο το να μοιράζεται και να χρησιμοποιεί πληροφορίες για ανθρώπους και τις δραστηριότητές τους, να μεταφέρει αυτές τις πληροφορίες ανάμεσα στις ιστοσελίδες, και αυτόματα να τις επεκτείνει, να τις συγχωνεύει και να τις επαναχρησιμοποιεί. Το FOAF project ξεκίνησε το 2000 σαν ένα πείραμα συνδεδεμένης πληροφορίας, από τους Dan Brickley και Libby Miller. Το FOAF είναι ένα μικρό αλλά καλοφτιαγμένο κομμάτι του ευρύτερου Semantic Web project.

3.3.2 Στόχοι

Ο κύριος στόχος αυτού του project είναι να αποκτήσουμε ένα καλύτερο τρόπο για την παρακολούθηση των διάσπαρτων κομματιών από δεδομένα που αντιπροσωπεύουν το διαδίκτυο. Πέρα όμως από τον κύριο στόχο του Project αυτού υπάρχουν και άλλοι επιμέρους στόχοι, οι οποίοι δεν περνάνε σε δεύτερη μοίρα. Οι στόχοι αυτοί είναι οι εξής: να είμαστε σε θέση να βρίσκουμε αρχεία στο διαδίκτυο για τα οποία θα βασιζόμαστε στις ιδιότητες και στις διασυνδέσεις τους. Να μπορούμε να βρίσκουμε πληροφορίες για ανθρώπους με βάση τις δημοσιεύσεις τους, τα εργασιακά τους στοιχεία, τα γκρουπ στα οποία είναι μέλη και στα ενδιαφέροντά τους. Να έχουμε την δυνατότητα να μοιραζόμαστε πράγματα χρησιμοποιώντας κοινές υποδομές και η αναζήτηση στο διαδίκτυο να γίνει περισσότερο σαν μια αναζήτηση σε μια βάση συγκεκριμένα δεδομένων παρά μια τυχαία αναζήτηση με βάση κάποιο κείμενο ή μια ακολουθία χαρακτήρων. Το RDF φαίνεται να υπόσχεται πολλά σε αυτό τον τομέα. Καθώς όμως το RDF προσδιορίζει όρους με ένα πιο αφαιρετικό μοντέλο πληροφοριών, έχουμε ανάγκη στο να ρωτάμε το ίδιο το διαδίκτυο λογικές ερωτήσεις για κοινά πράγματα (όπως αρχεία, οργανισμούς, ανθρώπους) και να παίρνουμε λογικά αποτελέσματα. Με το FOAF μπορούμε να το πετύχουμε αυτό σε μεγάλο βαθμό παίρνοντας ικανοποιητικά αποτελέσματα.

3.3.3 Περιγράφοντας πράγματα μέσω RDF

Κατά βάση το RDF είναι ένα μοντέλο πληροφοριών βασισμένο γύρω από μια ιδέα μιας απλής πρότασης με τρία μέρη. Κατέληξε αυτός ο κάπως δύσκαμπτος τρόπος για να περιγράφεις πράγματα, να είναι πολύ εκφραστικός, καθώς επιτρέπει στον καθένα να προσδιορίζει καινούργιους όρους. Το FOAF λεξιλόγιο παρέχει μερικούς βασικούς όρους σαν και αυτούς που παρέχει μια κοινή γλώσσα για την δημιουργία σελίδων οι οποίες είναι αναγνώσιμες από τις μηχανές. Το RDF επιτρέπει πολλές τέτοιες γλώσσες να αναμιχθούν μεταξύ τους. Για το σύστημα FAOF αυτό καθ' εαυτό δεν έχει σημασία για πιο πράγμα θα μιλάς, αλλά ο τρόπος με τον οποίο το λες. Χρησιμοποιώντας προσεκτικά σχεδιασμένους τύπους αρχείων και ένα απλό κοινό μοντέλο πληροφοριών, μπορούμε να φτιάξουμε μια κατανεμημένη βάση δεδομένων στην οποία ο καθένας μπορεί να λέει οτιδήποτε για οτιδήποτε.

Με το να συμπιέσουμε τις πληροφορίες μας στο απλό πλαίσιο τριπλετών, μπορούμε να κατασκευάσουμε συστήματα που συνδυάζουν αυτόματα τα δεδομένα από πολλαπλές πηγές. Το FOAF δημιουργήθηκε για να εξερευνήσει αυτή την πλευρά του διαδικτύου, και να δείξει πως πρακτικά εργαλεία μπορούν να κατασκευαστούν γύρω από αυτές τις αφηρημένες ιδέες.

Μερικές υπηρεσίες που χρησιμοποιούν το FOAF είναι: SWSE, Falcons, Sindice, Swoogle, PTSW, Yahoo's SearchMonkey.

3.4 SKOS

Το SKOS είναι ένα λεξιλόγιο που χρησιμοποιείται για την έκφραση των εννοιολογικών ιεραρχιών, που συχνά αναφέρονται σαν ταξινομήσεις, καθώς το RDFS και η OWL μας παρέχουν λεξιλόγια που περιγράφουν θεωρητικά μοντέλα από την πλευρά των κλάσεων και των ιδιοτήτων τους. Συλλογικά, SKOS, RDFS και OWL παρέχουν μια συνέχεια εκφραστικότητας. Το SKOS είναι η συντομογραφία του Simple Knowledge Organization System (SKOS) και είναι μια

οικογένεια από επίσημες γλώσσες, το οποίο σχεδιάστηκε και χρησιμοποιείται ευρέως για να αναπαριστά γλωσσάρια, ταξινομήσεις και τοπικές ιεραρχίες. Το SKOS είναι φτιαγμένο με βάση το RDF και το RDFS, και ο κύριος στόχος του είναι να επιτρέπει την εύκολη δημοσίευση των ελεγχόμενων δομημένων λεξιλογίων για το Σημασιολογικό Ιστό (Semantic Web). Το RDFS και η OWL χρησιμοποιούνται στις περιπτώσεις όπου πρέπει να αναπαρασταθούν σχέσεις υπαγωγής μεταξύ όρων. Τα δεδομένα του SKOS παρουσιάζονται σαν RDF τριπλέτες και μπορούν να κωδικοποιηθούν χρησιμοποιώντας οποιαδήποτε συμπαγή RDF σύνταξη.

ΕΠΙΛΟΓΟΣ

Οι γλώσσες αυτές αναπαριστούν τύπους και οντότητες για την καλύτερη περιγραφή των πόρων που αναπαρίστανται με το μοντέλο RDF. Οι πληροφορίες όμως που μας παρέχονται μέσω του RDF και των γλωσσών αναπαράστασης δεν είναι ευανάγνωστες από τους ανθρώπους, καθώς για να γίνουν κατανοητές από τους υπολογιστές αναπαριστώνται σε μορφή XML. Έτσι για να ανακτήσουμε πληροφορίες από τα δεδομένα αυτά πρέπει με κάποιο τρόπο να τις ανακτήσουμε σε μια μορφή η οποία είναι κατανοητή από τον άνθρωπο ή μια μορφή την οποία μπορούμε εύκολα να την διαχειριστούμε.

ΚΕΦΑΛΑΙΟ 4

SPARQL

ΕΙΣΑΓΩΓΗ

Την λύση στο πρόβλημα της ανάκτησης πληροφοριών από δεδομένα τα οποία βρίσκονται σε μορφή RDF και των γλωσσών αναπαράστασης τύπων, την έδωσε η SPARQL, η οποία είναι μια γλώσσα ερωτημάτων για δεδομένα τα οποία βρίσκονται σε μορφή RDF/XML.

4.1 Τι είναι η SPARQL

Η SPARQL είναι ένα πρότυπο γλώσσας ερωτημάτων και ένα πρωτόκολλο για την πρόσβαση στα δεδομένα του μοντέλου RDF. Είναι το ακρωνύμιο του Simple Protocol and RDF Query Language (SPARQL) και λειτουργεί για κάθε πηγή δεδομένων η οποία μπορεί να χαρτογραφηθεί σε μορφή RDF. Προφέρεται σπαρκλ (sparkl) και είναι μια προδιαγραφή του W3C (World Wide Web Consortium). Με την SPARQL μπορούμε να εκτελέσουμε ερωτήματα (queries) σε δεδομένα RDF, όπως επίσης μπορούμε να εκτελούμε ερωτήματα σε δεδομένα τα οποία βρίσκονται σε οποιοδήποτε από τους τύπους δεδομένων RDF (π.χ. lexical space, value space και lexical-to-value mapping). Η λέξη πρωτόκολλο (Protocol) που είναι μέρος του ακρωνυμίου SPARQL, αναφέρεται στους κανόνες και στο πως ένα πρόγραμμα πελάτης (client program) και ένας SPARQL server επεξεργασίας δεδομένων ανταλλάσουν SPARQL ερωτήματα και αποτελέσματα. Σε γενικές γραμμές ένα SPARQL ερώτημα είναι ένα ερώτημα το οποίο προσδιορίζει το μέρος της πληροφορίας που χρειαζόμαστε από το υποσύνολο των δεδομένων, το οποίο πληροί συγκεκριμένες προϋποθέσεις. Στο ερώτημα περιγράφονται οι συγκεκριμένες προϋποθέσεις με την μορφή προτύπου τριπλετών, οι οποίες είναι όμοιες με RDF τριπλέτες περιέχοντας επίσης και μεταβλητές οι οποίες προσθέτουν ευελιξία στο ταίριασμα των δεδομένων. Με την χρήση των μεταβλητών αυτών μπορούμε να παίρνουμε όλα τα διαφορετικά δεδομένα τα οποία πληρούν τα κριτήρια των προτύπων που έχουμε θέσει, χωρίς να γνωρίζουμε τη δομή των δεδομένων και το πώς είναι οργανωμένα μέσα στο RDF μοντέλο, στο οποίο εκτελούμε το συγκεκριμένο ερώτημα. Έτσι έχουμε τη δυνατότητα να έχουμε διαθέσιμη όλη την πληροφορία στα χέρια μας χωρίς απαραίτητα να γνωρίζουμε κάτι συγκεκριμένο για τα δεδομένα στα οποία εκτελέσαμε το ερώτημα.

4.2 Δομή και σύγκριση με την SQL

Η δομή ενός ερωτήματος SPARQL είναι παρόμοια με αυτή ενός ερωτήματος SQL. Παρόλα αυτά η SPARQL είναι διαφορετική από την SQL, και τα διάφορα σκέλη του ερωτήματος που αποτελείται έχουν διαφορετική σημασία. Ένα ερώτημα SPARQL έχει την εξής δομή: στο πρώτο σκέλος έχει προαιρετικά την ενότητα PREFIX στην οποία ορίζονται οι χώροι ονομάτων (namespaces) και οι συντομογραφίες τους που μπορούμε να χρησιμοποιήσουμε στο ερώτημα. Μετά ακολουθεί η ενότητα που προσδιορίζει την λειτουργία που θα εκτελέσει το SPARQL ερώτημα. Σε αυτή την ενότητα όπως και σε ένα ερώτημα SQL αναφέρονται οι πληροφορίες που θέλουμε να μας επιστραφούν από τα δεδομένα στα οποία εκτελούμε το ερώτημα. Έπειτα ακολουθεί προαιρετικά η ενότητα FROM. Η ενότητα αυτή διαφέρει από την αντίστοιχη της SQL στο ότι στην SQL στην συγκεκριμένη περιοχή αναγράφονται οι πίνακες στους οποίους αναφερόμαστε για να εκτελέσουμε το ερώτημα και να μας επιστραφούν τα δεδομένα. Στην SPARQL σε αυτήν την περιοχή αναφέρεται το ή τα end point, τα οποία είναι οι διευθύνσεις στις οποίες ακούει ένας SPARQL server που θα επεξεργαστεί το ερώτημα και θα μας επιστρέψει τα αποτελέσματα. Σε περίπτωση που εκτελούμε ένα ερώτημα σε κάποιο end point μέσω ενός interface το οποίο παρέχεται από το ίδιο το end point, δεν χρειάζεται να αναφέρουμε το πεδίο FROM στο ερώτημα, καθώς το ερώτημα μετατρέπεται πριν αναλυθεί από τον SPARQL server συμπληρώνοντας το συγκεκριμένο πεδίο και εκτελείται αυτόματα στο end point που μας προσφέρει το interface. Τέλος υπάρχει το πεδίο WHERE στο οποίο όπως και στην SQL προσδιορίζει τις προϋποθέσεις που πρέπει να πληρούν τα δεδομένα που θα επιστραφούν. Η διαφορά με την SQL είναι ότι στην SPARQL οι προϋποθέσεις αυτές παρουσιάζονται με μορφή προτύπων από τριπλέτες που πρέπει να συμφωνούν με τα δεδομένα που επιστρέφονται. Επίσης στο πεδίο αυτό εκτός από πρότυπα για το ταίριασμα των δεδομένων μπορούμε να εφαρμόσουμε επιπλέον διάφορα φίλτρα τα οποία θα μας βοηθήσουν στο να καθαρίσουμε τα επιστρεφόμενα δεδομένα κρατώντας αυτά που μας ενδιαφέρουν.

Οι λειτουργίες που μπορεί να εκτελέσει ένα SPARQL ερώτημα είναι οι εξής:

- Η λειτουργία SELECT είναι η πιο συνηθισμένη λειτουργία σε ένα SPARQL ερώτημα. Επιστρέφει τα δεδομένα που πληρούν τα κριτήρια που θέσαμε στο πεδίο WHERE.

- Η λειτουργία CONSTRUCT επιστρέφει τριπλέτες τις οποίες μπορούμε να τις εισάγουμε σε ένα γράφο. Μπορούμε να «τραβήξουμε» τριπλέτες απ' ευθείας από μια πηγή δεδομένων χωρίς να τις αλλάξουμε, ή μπορούμε να «τραβήξουμε» τις τιμές και να τις χρησιμοποιήσουμε για να δημιουργήσουμε καινούργιες τριπλέτες. Αυτό μας επιτρέπει να αντιγράψουμε, να δημιουργήσουμε, και να μετασχηματίσουμε RDF δεδομένα, και κάνει ευκολότερη την αναγνώριση των δεδομένων τα οποία δεν είναι σύμφωνα με τους ειδικούς επιχειρηματικούς κανόνες.

- Η λειτουργία ASK ρωτάει ένα επεξεργαστή ερωτημάτων αν ένας δοσμένο πρότυπο γράφου περιγράφει ένα σύνολο από τριπλέτες σε ένα συγκεκριμένο data set ή όχι, και ο επεξεργαστής επιστρέφει μια Boolean τιμή true ή false. Αυτό είναι ιδανικό για να εκφράζει επαγγελματικούς κανόνες για όρους που μπορούν ή δεν μπορούν να είναι αληθείς στα δεδομένα μας. Τέτοιοι κανόνες μπορούν να χρησιμοποιηθούν ώστε να αυτοματοποιηθεί ο ποιοτικός έλεγχος στον επεξεργαστή δεδομένων.

- Η λειτουργία DESCRIBE ρωτάει για τριπλέτες οι οποίες περιγράφουν ένα συγκεκριμένο πόρο. Η προδιαγραφή της SPARQL αφήνει στον επεξεργαστή ερωτημάτων να αποφασίσει ποιες από τις τριπλέτες θα αποσταλούν πίσω σαν μια περιγραφή από ονοματισμένους πόρους.

Τα αποτελέσματα ενός SPARQL ερωτήματος μπορεί να επιστραφούν σε διάφορες μορφές. Αυτές οι μορφές είναι RDF/XML, XML ή JSON.

Στη μορφή RDF μπορούν να επιστραφούν τα αποτελέσματα των ερωτημάτων CONSTRUCT και DESCRIBE. Σε μορφή XML συνήθως επιστρέφονται τα αποτελέσματα των ερωτημάτων SELECT και ASK. Επίσης τα αποτελέσματα των ερωτήματα CONSTUCT και DESCRIBE μπορούν να επιστραφούν χρησιμοποιώντας RDF/XML σύνταξη. Τέλος τα αποτελέσματα των ερωτημάτων

SELECT και ASK μπορούν να επιστραφούν σε μια πολύ εύχρηστη μορφή για τους Web clients, την μορφή JSON, και έτσι να χρησιμοποιηθούν άμεσα για την επεξεργασία των πληροφοριών που επεστράφησαν χωρίς να χρειάζεται να γίνει κάποια μετατροπή στα δεδομένα.

Τέλος η SPARQL μας παρέχει κάποιες επιπλέον λειτουργίες με τις οποίες μπορούμε να κάνουμε αποτελεσματικότερα ερωτήματα. Αυτές οι λειτουργίες είναι οι εξής:

- ORDER BY
- DISTINCT
- REDUSE
- OFFSET
- LIMIT

Με τη λειτουργία ORDER BY όπως και στης SQL έχουμε την δυνατότητα να ταξινομήσουμε τα αποτελέσματα ενός ερωτήματος σύμφωνα με κάποιο κριτήριο. Με τη λειτουργία DISTINCT αφαιρούμε από τα αποτελέσματα τις διπλές εγγραφές και παρουσιάζονται οι μοναδικές εγγραφές. Με τη λειτουργία REDUSE συμβαίνει το αντίθετο από τη λειτουργία DISTINCT. Με αυτή τη λειτουργία φροντίζουμε στα αποτελέσματα που θα μας επιστραφούν να περιέχονται και οι διπλές εγγραφές. Η λειτουργία OFFSET καθορίζει μετά από ποιο σημείο των επιστρεφόμενων αποτελεσμάτων θα ξεκινήσει η εμφάνισή τους. Τέλος η λειτουργία LIMIT θέτει το ανώτατο όριο των αποτελεσμάτων που θα επιστραφούν από ένα ερώτημα.

4.3 SPARQL Update (SPARUL)

Άλλη μια διαφορά της SPARQL και της SQL είναι ότι με την υπάρχουσα τυποποιημένη έκδοση της SPARQL δεν μπορούμε να προσθέσουμε ή να τροποποιήσουμε τα δεδομένα τα οποία είναι αποθηκευμένα, όπως έχουμε αυτή την δυνατότητα με την SQL. Αυτό είναι μια προδιαγραφή η οποία βρίσκεται υπό υλοποίηση και θα είναι ενσωματωμένη πιθανότατα στην επόμενη τυποποιημένη έκδοση της SPARQL 1.1. Σε αυτή την νέα τυποποιημένη έκδοση εκτός από της στάνταρ λειτουργίες (select, construct, describe και ask) που αναφέρθηκαν πιο

πάνω, θα δίνεται η δυνατότητα στον χρήστη να εκτελεί και άλλες λειτουργίες για την μετατροπή, την ενημέρωση και την διαγραφή των δεδομένων μέσα στον RDF γράφο ενός Graph Store ή καλύτερα ενός triplestore. Οι νέες λειτουργίες που μελετούνται είναι οι εξής:

- Insert (εισαγωγή τριπλετών σε ένα RDF γράφο ενός triplestore).
- Delete (διαγραφή μεμονωμένων τριπλετών από ένα RDF γράφο σε ένα triplestore).
- Load (φορτώνει ένα RDF γράφο σε ένα triplestore).
- Clear (διαγράφη ένα ολόκληρο RDF γράφο από ένα triplestore).
- Create (δημιουργεί ένα νέο RDF γράφο σε ένα triplestore).
- Drop (διαγράφη ένα συγκεκριμένο ή συγκεκριμένους RDF γράφους από ένα triplestore).
- Copy, move, ή add (Αντιγράφει, μετακινεί ή προσθέτει περιεχόμενο σε ένα RDF γράφο ενός triplestore).
- Εκτέλεση πολλών διαφορετικών ενεργειών σαν μία ενέργεια.

ΕΠΙΛΟΓΟΣ

Η SPARQL έδωσε την λύση για την συλλογή πληροφοριών από δεδομένα τα οποία βρίσκονται σε μορφή RDF/XML. Με την SPARQL μπορούμε να εκτελέσουμε ερωτήματα σε δεδομένα τα οποία δεν γνωρίζουμε την δομή τους, να πάρουμε τα αποτελέσματα και να τα διαχειριστούμε κατάλληλα.

ΚΕΦΑΛΑΙΟ 5

Linked Data

ΕΙΣΑΓΩΓΗ

Η αναπαράσταση κατανεμημένων δεδομένων τα οποία βρίσκονται διάσπαρτα στο διαδίκτυο σε διαφορετικές πηγές, καθώς και η σύνδεση μεταξύ τους, προσδίδοντάς τους ιδιότητες και γνωρίσματα, οδήγησαν στην ανάπτυξη ενός νέου όρου για τα δεδομένα στο διαδίκτυο τα οποία βρίσκονται σε διαφορετικές πηγές και συνδέονται μεταξύ τους. Ο όρος που αναφέρεται στα δεδομένα αυτά είναι Linked Data. Ο όρος αυτός αντιπροσωπεύει πληροφορίες που είναι δημοσιευμένες στο διαδίκτυο και είναι διαθέσιμες σε όλους .

5.1 Βασικές αρχές

Τα Linked Data περιγράφουν ένα σύνολο βέλτιστων πρακτικών για την δημοσίευση δομημένων δεδομένων στο διαδίκτυο έτσι ώστε να είναι συνδεμένα μεταξύ τους και έτσι να γίνονται χρησιμότερα και για τους ανθρώπους αλλά και για τους υπολογιστές. Τα Linked Data «χτίστηκα» πάνω σε στάνταρ τεχνολογίες που χρησιμοποιεί το διαδίκτυο όπως το HTTP και τα URIs, αλλά αντί να χρησιμοποιούνται για να προβάλουν ιστοσελίδες σε ανθρώπους απλά για ανάγνωση, επεκτείνουν αυτή την δυνατότητα και προσφέρουν τις πληροφορίες με τέτοιο τρόπο ώστε να μπορούν αυτόματα να διαβαστούν και από τους υπολογιστές. Αυτό μας δίνει την δυνατότητα δεδομένα από διαφορετικές πηγές να συνδέονται μεταξύ τους και να μπορούν να εκτελέσουμε ερωτήματα πάνω σε αυτά. Οι πρακτικές αυτές προτάθηκαν από τον Tim Berners-Lee (τον δημιουργό του διαδικτύου) και είναι γνωστές σαν αρχές των Linked Data (Linked Data principle). Οι αρχές αυτές είναι οι ακόλουθες:

1. Χρησιμοποιούμε URIs για να δώσουμε όνομα στα πράγματα.
2. Χρησιμοποιούμε τα HTTP URIs έτσι ώστε οι άνθρωποι να μπορούν να αναζητούν αυτά τα ονόματα.
3. Όταν κάποιος αναζητά ένα URI, του παρέχονται χρήσιμες πληροφορίες, χρησιμοποιώντας τα standards RDF, και SPARQL.
4. Στα δεδομένα εμπεριέχονται σύνδεσμοι σε άλλα URIs, έτσι ώστε να μπορούν να εξερευνηθούν και άλλα πράγματα.

Η βασική ιδέα των Linked Data είναι να εφαρμόσουν μια γενική αρχιτεκτονική του διαδικτύου, σε μια διεργασία για να μοιράζονται δομημένα δεδομένα σε παγκόσμια κλίμακα. Για να γίνει κατανοητή η αρχιτεκτονική των αρχών των Linked Data, πρέπει να έχουν γίνει κατανοητά πιο πριν το στάνταρ Uniform Resource Identifiers (URIs) σαν ένα παγκόσμιο μοναδικό μηχανισμό προσδιορισμού, το στάνταρ Hypertext Transfer Protocol (HTTP) σαν ένα καθολικό μηχανισμό πρόσβασης, και το Hypertext Markup Language (HTML) σαν μια ευρέως διαδεδομένη μορφή του περιεχομένου. Επιπλέον το διαδίκτυο βασίστηκε στην ιδέα ύπαρξης υπερσυνδέσμων ανάμεσα στις ιστοσελίδες που μπορεί να φυλάσσονται σε διαφορετικούς server. Οι υπερσύνδεσμοι επιτρέπουν τους χρήστες να πλοηγούνται ανάμεσα στους διάφορους server. Οι υπερσύνδεσμοι υπάρχουν για να συνδέουν το περιεχόμενο από διάφορους server σε ένα παγκόσμιο χώρο πληροφορίας. Τα Linked Data με την σειρά τους βασίζονται απευθείας στην αρχιτεκτονική του διαδικτύου και εφαρμόζουν αυτές τις αρχιτεκτονικές για να μοιραστούν τα δεδομένα σε παγκόσμια κλίμακα.

Παραβιάζοντας κάποιο από αυτούς τους κανόνες δεν καταστρέφει κάτι, αλλά χάνεται η ευκαιρία να κάνουμε τα δεδομένα διασυνδεμένα. Αυτό με την σειρά του περιορίζει τους τρόπους με τους οποίους θα μπορούσαν να επαναχρησιμοποιηθούν χωρίς να προβλέψουμε πως. Αυτή η απρόσμενη επαναχρησιμοποίηση των πληροφοριών είναι η επιπλέον αξία που προστίθεται στο διαδίκτυο.

5.2 Χρήση των URIs στα Linked Data

Για να δημοσιευτούν δεδομένα στο διαδίκτυο πρέπει με κάποιο τρόπο να τα προσδιορίσουμε. Επίσης πρέπει να προσδιοριστούν οι ιδιότητές τους και οι σχέσεις που περιγράφουν αυτά τα δεδομένα που επίσης μπορεί σε όλα αυτά να περιλαμβάνονται και αρχεία διαδικτύου ή ένας πραγματικός κόσμος οντοτήτων και αφηρημένων εννοιών. Έτσι για την αναπαράσταση των δεδομένων χρησιμοποιείται η αρχιτεκτονική του διαδικτύου για τον προσδιορισμό πόρων, τα

HTTP URIs. Τα HTTP URIs είναι χρήσιμα για δύο λόγους στην αναπαράσταση των δεδομένων:

i) Μας παρέχουν ένα απλό τρόπο για να δημιουργούνται παγκόσμια μοναδικά ονόματα.
ii) Ένα URI δεν είναι απλά μόνο ένα όνομα, αλλά και ένα μέσω πρόσβασης σε πληροφορίες που περιγράφουν την προσδιοριζόμενη έννοια.

5.3 Web of Data

Αρκετοί άτομα αλλά και οργανισμοί έχουν υιοθετήσει τα Linked Data για να δημοσιεύσουν τα δεδομένα τους στο διαδίκτυο, και όχι απλά να τα τοποθετήσουν σε αυτό. Το αποτέλεσμα είναι ένας παγκόσμιος χώρος δεδομένων που ονομάζεται το διαδίκτυο των δεδομένων (Web of Data). Το Web of Data αποτελεί ένα γιγάντιο παγκόσμιο γράφο που απαρτίζεται από δισεκατομμύρια RDF statements που βρίσκονται σε αναρίθμητες πηγές, καλύπτοντας όλους τους τύπους των θεμάτων, όπως π.χ. βιβλία, επιστημονικές δημοσιεύσεις, ταινίες, στατιστικά δεδομένα και άλλα πολλά. Το Web of Data μπορεί να θεωρηθεί ως ένα επιπλέον επίπεδο που είναι στενά συνυφασμένο με το κλασικό αρχείο διαδικτύου και έχει πολλά από τα ίδια χαρακτηριστικά:

- Το Web of Data είναι γενικό και μπορεί να περιέχει οποιοδήποτε τύπο δεδομένων.
- Οποιοσδήποτε μπορεί να δημοσιεύσει δεδομένα .
- Το Web of Data μπορεί να αναπαραστήσει διαφορές και αντιφατικές πληροφορίες για μια οντότητα.
- Οι οντότητες μπορούν να συνδέονται με RDF συνδέσμους, δημιουργώντας ένα παγκόσμιο γράφο δεδομένων που εκτείνεται σε πηγές δεδομένων και επιτρέπει την ανακάλυψη νέων. Αυτό σημαίνει πως οι εφαρμογές δεν χρειάζεται να έχουν υλοποιηθεί για την χρήση συγκεκριμένων πηγών δεδομένων, αλλά μπορούν να ανακαλύψουν καινούργιες πηγές κατά την διάρκεια εκτέλεσης ακολουθώντας τους RDF συνδέσμους.

- Αυτοί που εκδίδουν τα δεδομένα δεν περιορίζονται κατά την επιλογή των λεξιλογίων με τα οποία θα αναπαραστήσουν τα δεδομένα.
- Τα δεδομένα είναι αυτό-περιγραφόμενα. Αν μια εφαρμογή που καταναλώνει Linked Data συναντήσει στοιχεία που περιγράφονται με άγνωστο λεξιλόγιο, η εφαρμογή από μόνη της μπορεί να αναφερθεί σε URIs τα οποία προσδιορίζουν τους όρους του λεξιλογίου με σκοπό να βρουν τον ορισμό τους.
- Η χρήση του HTTP σαν τυποποιημένος μηχανισμός πρόσβασης στα δεδομένα και το RDF σαν τυποποιημένο μοντέλο δεδομένων απλοποιεί την πρόσβαση στα δεδομένα σε σχέση με Web APIs, τα οποία στηρίζονται σε ετερογενείς μοντέλα δεδομένων και διεπαφές πρόσβασης.

5.4 Basic web look-up

Ο απλούστερος τρόπος για να κατασκευάσουμε linked data είναι να χρησιμοποιήσουμε σε ένα αρχείο, ένα URI το οποίο δείχνει σε ένα άλλο. Όταν γράφουμε σε ένα RDF αρχείο, ας πούμε <http://example.org/smith>, μπορούμε να χρησιμοποιήσουμε τοπικά προσδιοριστικά ενδιάμεσα του αρχείου, π.χ. #albert, #brian και #carol. Σε μορφή RDF/XML:

```
<rdf:Description about="#albert">
 <fam:child rdf:Resource="#brian">
  <fam:child rdf:Resource="#carol">
</rdf:Description>
```

Τώρα η αρχιτεκτονική του παγκόσμιου ιστού μας δίνει ένα παγκόσμιο αναγνωριστικό "http://example.org/smith#albert" για τον Albert. Αυτό που κάνει είναι ένα πολύτιμο πράγμα, καθώς οποιοσδήποτε στον πλανήτη μπορεί τώρα να χρησιμοποιήσει αυτό το παγκόσμιο προσδιοριστικό και να αναφέρεται στον Albert, έχοντας την δυνατότητα να του προσθέσει περισσότερες πληροφορίες. Αυτό το URI ονομάζεται εύρεση τιμών. Από αυτό εξαρτάται το Semantic Web.

5.5 Πλοηγήσιμος γράφος

Ένα πολύ σημαντικό πρότυπο είναι ένα σύνολο δεδομένων τα οποία μπορείς να εξερευνήσεις καθώς πλοηγείσαι από σύνδεσμο σε σύνδεσμο ανακτώντας δεδομένα. Όποτε κάποιος αναζητά το URI για ένα κόμβο σε ένα RDF γράφο, ο server του επιστρέφει πληροφορίες σχετικά με τα statements στα οποία ο συγκεκριμένος όρος εμφανίζεται σαν υποκείμενο και σαν αντικείμενο. Επίσημα, ένας γράφος G αποκαλείται πλοηγήσιμος αν, για κάθε URI που αναπαριστά ένα κόμβο μέσα στο γράφο G, αναζητήσουμε αυτό το URI θα μας επιστραφούν πληροφορίες οι οποίες περιγράφουν τον κόμβο αυτό, το οποίο σημαίνει:

- Θα μας επιστραφούν όλες οι δηλώσεις (statements) στις οποίες ο κόμβος είναι υποκείμενο ή αντικείμενο.
- Θα περιγράφονται όλοι οι κενοί κόμβοι που πρόσκεινται στον κόμβο μέσω ενός τόξου.

Πρακτικά όταν τα δεδομένα είναι αποθηκευμένα σε δυο αρχεία, οποιαδήποτε δήλωση (statement) η οποία συνδέει πράγματα μεταξύ αυτών των δύο αρχείων, πρέπει να επαναληφθεί σε κάθε ένα από αυτά. Π.χ. αν γίνει μια αναφορά ενός αντικειμένου ότι ανήκει σε μια ομάδα αντικειμένων, τότε πρέπει και στην συγκεκριμένη ομάδα αντικειμένων να γίνει αναφορά ότι της ανήκει το αντικείμενο. Έτσι αν κάποιος ξεκινήσει από την ομάδα μπορεί να βρει πως το συγκεκριμένο αντικείμενο ανήκει και αυτό στην ομάδα, και ακόμα αν κάποιος ξεκινήσει από το συγκεκριμένο αντικείμενο, μέσω του URI του μπορεί να φτάσει στην ομάδα που ανήκει και να βρει όλα τα υπόλοιπα αντικείμενα που ανήκουν στην συγκεκριμένη ομάδα.

5.6 Περιορισμοί

Έτσι δηλώσεις (statements) οι οποίες συνδέουν πράγματα ανάμεσα σε δυο αρχεία πρέπει να επαναλαμβάνονται στο κάθε ένα από τα αρχεία. Παρόλα αυτά δεν είναι πάντα χρήσιμο να έχουμε αποθηκευμένα τα ίδια δεδομένα σε διαφορετικές πηγές. Π.χ. αν σε ένα αρχείο στο οποίο έχουμε αποθηκευμένο το

γεωγραφικό μήκος και το γεωγραφικό πλάτος σε μια χώρα της κατοικίας κάποιου υπαλλήλου μιας εταιρίας, δεν είναι χρήσιμη αυτή η πληροφορία να είναι αποθηκευμένη στο αρχείο το οποίο περιγράφει τη συγκεκριμένη χώρα. Είναι λογικό να υπάρχει σύνδεση από τις πληροφορίες του υπαλλήλου προς την χώρα στην οποία διαμένει αλλά όχι το αντίθετο. Κατά αυτό τον τρόπο τα δεδομένα πρέπει να αποθηκεύονται αμφίδρομα όταν οι πληροφορίες έχουν σημασία και για τις δύο πλευρές. Έτσι όταν έχουμε δεδομένα από πολλαπλές πηγές τότε πρέπει να υπάρχουν και συμβιβασμοί μεταξύ αυτών των πηγών. Οι συμβιβασμοί αυτοί μπορούν να διευθετηθούν, αν απαντήσουμε στην ερώτηση: «Αν κάποιος έχει το URI ενός αντικειμένου, ποιες σχέσεις με άλλα αντικείμενα θα ήταν χρήσιμες για αυτών να γνωρίζει;». Με αυτό τον τρόπο, παρόλο που χάνεται μέρος της πληροφορίας καθώς τα δεδομένα δεν είναι προσπελάσιμα και από τις δύο πλευρές, διατηρείται το μέρος της πληροφορίας που είναι χρήσιμη και δεν κρατάμε ίδιες πληροφορίες αποθηκευμένες σε πολλαπλές πηγές.

5.7 Υπηρεσίες Ερωτημάτων

Ο τεράστιος όγκος των δεδομένων μας αναγκάζει να διατηρούμαι και να προσφέρουμε (serve) τα δεδομένα σε όσο το δυνατόν πιο πολλά αρχεία γίνεται, αλλά αυτό καθιστά δύσκολη την διαδικασία για την αποτελεσματική εκτέλεση ερωτημάτων από απόσταση στην βάση δεδομένων. Έτσι σε αυτή την περίπτωση, το ιδανικότερο θα ήταν να παρέχουμε μια υπηρεσία ερωτημάτων SPARQL για τα δεδομένα αυτά. Για να κάνουμε τα δεδομένα αποτελεσματικά συνδεδεμένα, κάποιος ο οποίος έχει μόνο το URI από ένα αντικείμενο θα πρέπει να μπορεί να βρει τον δρόμο για το SPARQL endpoint από το οποίο θα μπορέσει να πάρει πληροφορίες για αυτό. Αυτό θα μπορούσε να γίνει αν αυτός που αιτείται πληροφορίες για το συγκεκριμένο URI καθοδηγούνταν σε ένα αρχείο το οποίο θα περιείχε πληροφορίες μεταδεδομένων όπου εκεί θα αναφέρονταν πληροφορίες σχετικά με το ποια endpoint μπορεί να χρησιμοποιήσει και ποιες πληροφορίες για τις κλάσεις των URIs μπορεί να πάρει. Δυστυχώς ακόμη δεν έχουν τυποποιηθεί λεξιλόγια για να γίνει κάτι τέτοιο.

5.8 Παραδείγματα Link Data

Ένα τυπικό και πολύ μεγάλο dataset από Linked Data είναι η DBPedia (www.dbpedia.org) η οποία, ουσιαστικά κάνει το περιεχόμενο του Wikipedia διαθέσιμο σε μορφή RDF. Το σημαντικό με την DBPedia είναι ότι δεν περιέχει μόνο τα δεδομένα του Wikipedia αλλά ενσωματώνει και συνδέσεις σε άλλα dataset στο διαδίκτυο, π.χ. το Geonames (www.geonames.org) . Παρέχοντας αυτούς τους έξτρα συνδέσμους (σε τριπλέτες RDF), οι εφαρμογές μπορούν να εκμεταλλευτούν αυτή την παραπάνω γνώση από άλλα dataset, η οποία πιθανότατα μπορεί και να είναι πιο ακριβείς, καθώς εξελίσσεται μια αίτηση από μια εφαρμογή. Λόγω της ενσωμάτωσης των γεγονότων αυτών από διάφορα datasets, η εφαρμογή μπορεί να παρέχει στον χρήστη πολύ καλύτερα αποτελέσματα.

Ένα άλλο παράδειγμα Linked Data βρίσκεται στην διεύθυνση *data.gov.uk*. Σε αυτή την διεύθυνση η αγγλική κυβέρνηση έχει ανεβάσει δημόσια δεδομένα για να βοηθήσει τον κόσμο να καταλάβει πως δουλεύει η κυβέρνηση και πως δημιουργούνται οι πολιτικές της. Κάνοντας αυτά τα δεδομένα διαθέσιμα οι πολίτες μπορούν ευκολότερα να παίρνουν αποφάσεις και να κάνουν προτάσεις πάνω στο κυβερνητικό έργο βασιζόμενοι σε λεπτομερείς πληροφορίες. Πιο συγκεκριμένα υπάρχουν ήδη πάνω από 5.400 dataset διαθέσιμα, από όλα τα κεντρικά κυβερνητικά παραρτήματα και μια σειρά άλλων φορέων του δημόσιου τομέα και των τοπικών αρχών.

5.9 Παραδείγματα εφαρμογών Linked Data

Όλα αυτά μας έχουν βοηθήσει στο να αναπτυχθούν διάφορες εφαρμογές που χρησιμοποιούν τα Linked Data, προσφέροντας εύκολα χρήσιμες πληροφορίες σε αυτούς που τις χρησιμοποιούν. Μερικές εφαρμογές οι οποίες χρησιμοποιούν τα Linked Data είναι οι εξής:

- *Tabulator*

Ο Tabulator είναι ένα γενικό πρόγραμμα περιήγησης και σύνταξης δεδομένων. Χρησιμοποιεί το περίγραμμα και τις λειτουργίες του πίνακα, και παρέχει έναν τρόπο για περιήγηση σε RDF / Linked Data στο διαδίκτυο.

- *BBC's Music Beta*

Το BBC εξέδωσε μια έκδοση του νέου τους Music beta site το οποίο φτιάχτηκε με βάση τα μεταδεδομένα του Musicbrainz και τα προσδιοριστικά (identifiers) του DBPedia στις αρχές του 2009. Μεταδεδομένα που αφορούν την μουσική όπως συσχετιζόμενοι καλλιτέχνες, καθώς και τους συνδέσμους τους που δείχνουν στο Wikipedia «τραβήχτηκαν» από το Musicbrainz, το εισαγωγικό κείμενο της βιογραφίας κάθε καλλιτέχνη «τραβήχτηκε» από το Wikipedia μέσω του μηχανισμού διασύνδεσης του DBPedia.

- *Semantic CrunchBase Twitter Bot*

Το Semantic CrunchBase είναι ένας linked data wrapper, βασισμένος στο δημόσιο CrunchBase API ένα ελεύθερο λεξικό για εταιρίες τεχνολογίας, ανθρώπους και επενδυτές. Το CrunchBase Twitter Bot μας επιτρέπει να κάνουμε ερωτήματα σχετικά με εταιρείες του Silicon Valley όπως πίνακες, επιχειρήσεις κ.α.

- *DBpedia mobile*

Το DBPedia mobile είναι μια ενδιαφέρουσα εφαρμογή για mobile περιβάλλοντα. Κατά βάση είναι μια τοποθεσιοκεντρική DBPedia client εφαρμογή για κινητές συσκευές, οι οποίες βασίζονται στο σήμα GPS μιας κινητής συσκευής, η οποία μπορεί να καταστήσει ένα χάρτη υποδεικνύοντας τις κοντινές τοποθεσίες βασιζόμενη στις πληροφορίες από τη DBPedia.

- *OpenLink Data Calendar*

Το ODS-Calendar είναι μία εφαρμογή που προσφέρει δυνατότητες για την οργάνωση ενός ημερολογίου συμβάντων, εργασιών και σημειώσεων μέσω χώρων δεδομένων που παρέχονται από Weblogs, Wikis, shared bookmarks κ.α. υποστηρίζοντας ένα αριθμό υπηρεσιών ερωτημάτων όπως GData και SPARQL, διάφορα πρωτόκολλα δημοσίευσης, και ένα εύρος εφαρμογών.

ΕΠΙΛΟΓΟΣ

Τα Linked Data είναι συνδεδεμένα δεδομένα τα οποία είναι εμπλουτισμένα με πληροφορίες από διάφορες πηγές στο διαδίκτυο. Είναι δημόσια και μπορούν να χρησιμοποιηθούν από τον οποιονδήποτε. Έτσι έχουν αναπτυχθεί εφαρμογές που χρησιμοποιούν αυτά τα δεδομένα που είναι δημόσια και μέσω αυτών των εφαρμογών μπορούμε συνδυάσουμε και να αποκτήσουμε διάφορες πληροφορίες της οποίες δεν θα μπορούσαμε διαφορετικά.

ΚΕΦΑΛΑΙΟ 6

Semantic Web

ΕΙΣΑΓΩΓΗ

Με το RDF μπορούμε να αναπαραστήσουμε δεδομένα, να τους δώσουμε ιδιότητες και γνωρίσματα. Έτσι προσθέτουμε στα δεδομένα που συνδέονται μεταξύ τους πληροφορίες οι οποίες βρίσκονται σε άλλες πηγές και έχουμε τα Linked Data. Τα δεδομένα αυτά όμως από μόνα τους δεν σημαίνουν κάτι και ούτε ξεχωρίζουν με κάποιο τρόπο για τους υπολογιστές. Την λύση σε αυτό το πρόβλημα ήρθε να δώσει το Σημασιολογικό Διαδίκτυο (Semantic Web) το οποίο έχει σαν στόχο να προσδιορίσει σημασιολογικά τις έννοιες που αναπαρίστανται στο διαδίκτυο έτσι ώστε οι υπολογιστές να μπορούν να ξεχωρίσουν τις διαφορές ανάμεσά τους και να τις προσδιορίσουν σημασιολογικά.

6.1 Τι είναι το Σημασιολογικό διαδίκτυο (Semantic Web)

Το σημασιολογικό διαδίκτυο είναι ένας ιστός ο οποίος είναι ικανός να περιγράψει πράγματα με τέτοιο τρόπο έτσι ώστε να είναι κατανοητά από τους υπολογιστές. Δεν αφορά συνδέσμους μεταξύ των ιστοσελίδων, αλλά περιγράφει τις σχέσεις που παρουσιάζονται μεταξύ των πραγμάτων και τις ιδιότητες των πραγμάτων αυτών. Δίνει την δυνατότητα στους ανθρώπους να μοιράζονται το περιεχόμενο των πληροφοριών πέρα από τα όρια των εφαρμογών και τον ιστοσελίδων. Σε αντίθεση με το απλό διαδίκτυο, το οποίο είναι ένας «ιστός αρχείων» τα οποία συνδέονται μεταξύ τους μέσω υπερσυνδέσμων (hyperlinks), το W3C βοηθάει στην ανάπτυξη μιας τεχνολογίας η οποία στηρίζει ένα «ιστό δεδομένων», όπως τα δεδομένα σε μια βάση δεδομένων. Ο μεγαλύτερος στόχος του «ιστού δεδομένων» είναι να κάνει τους υπολογιστές να εκτελούν πιο παραγωγικές εργασίες και στο να αναπτυχθούν συστήματα τα οποία μπορούν να υποστηρίξουν αξιόπιστες αλληλεπιδράσεις μέσω του δικτύου. Αυτή τη στιγμή μπορούμε να χρησιμοποιήσουμε τους υπολογιστές για να αναζητήσουν αρχεία που βρίσκονται στο διαδίκτυο, αλλά και πάλι τα αρχεία αυτά πρέπει να διαβαστούν και να ερμηνευτούν από ανθρώπους πριν προβληθεί κάποια χρήσιμη πληροφορία. Οι υπολογιστές μπορούν να αναπαραστήσουν πληροφορίες αλλά δεν μπορούν να κατανοήσουν τι είναι αυτές οι πληροφορίες και να προβάλουν τα κατάλληλα δεδομένα τα οποία είναι σχετικά σε κάθε περίσταση. Το Semantic Web διαχειρίζεται τα δεδομένα όπως το διαδίκτυο χειρίζεται τα αρχεία, έτσι οι μηχανές

μπορούν να επεξεργάζονται, να αλλάζουν, να ερμηνεύουν ακόμα και να επιδρούν στα δεδομένα αυτά. Οι υπολογιστές δεν καταλαβαίνουν πραγματικά τις πληροφορίες με τον τρόπο που μπορεί να τις καταλάβει ο άνθρωπος, αλλά έχουν αρκετές πληροφορίες για να κάνουν μια λογική σύνδεση μεταξύ των πληροφοριών αυτών και να πάρουν αποφάσεις. Έτσι το Semantic Web επιτρέπει στους ανθρώπους να δημιουργούν περιοχές δεδομένων, λεξιλόγια, και να θέτουν κανόνες στην διαχείριση των δεδομένων αυτών. Για να γίνουν όλα αυτά, όπως και στα Linked data, το Semantic Web χρησιμοποιεί τεχνολογίες όπως URI, RDF,RDF Schema, SPARQL, OWL, και SKOS για να περιγράψει τα πράγματα.

6.2 Πως υλοποιείται

Για να υλοποιηθεί το Semantic Web απαιτείται να προστεθούν σημασιολογικά μεταδεδομένα ή δεδομένα τα οποία περιγράφουν δεδομένα στους πόρους πληροφοριών. Αυτό θα επιτρέψει στις μηχανές να επεξεργαστούν αποτελεσματικά τα δεδομένα βασιζόμενες στις σημασιολογικές πληροφορίες οι οποίες περιγράφουν τα δεδομένα. Όταν υπάρχει αρκετή σημασιολογική πληροφορία η οποία συσχετίζεται με τα δεδομένα, οι υπολογιστές μπορούν να βγάλουν συμπεράσματα για τα δεδομένα, π.χ. να καταλάβουν τι είδους είναι μια πηγή δεδομένων και πως αυτή σχετίζεται με τα υπόλοιπα δεδομένα.

Για να καταλάβουν λοιπόν οι μηχανές τα δεδομένα και τι είναι αυτά, πρέπει να μετατραπούν σε μια ενιαία μορφή, όπου κάθε πεδίο που θα έχει τον ίδιο τύπο θα περιέχει ίδιου τύπου πληροφορίες. Αυτή η λειτουργικότητα βρίσκεται στο διαδίκτυο σήμερα σε ιστοσελίδες που χρησιμοποιούν τύπους, οι οποίοι επιτρέπουν την εισαγωγή πληροφοριών και την εκτέλεση ερωτημάτων. Το Semantic Web απαιτεί τα δεδομένα, από διαφορετικές πηγές να χαρακτηρίζονται με βάση τις ιδιότητες τους και τη σχέση τους με άλλα δεδομένα. Εδώ είναι που πρέπει να χρησιμοποιηθούν τεχνολογίες όπως RDF, RDFS, και OWL ώστε να είναι κάτι τέτοιο εφικτό.

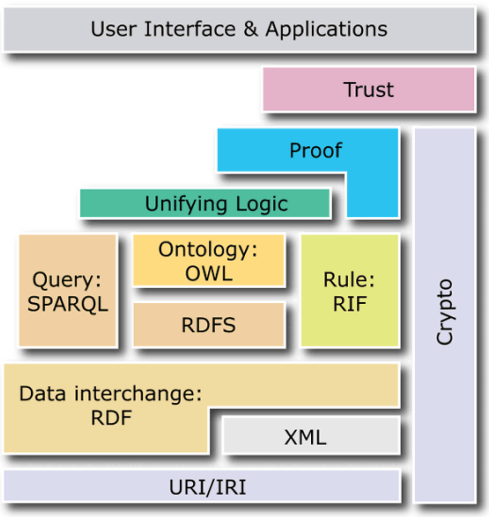
Το RDF επιτρέπει στις μηχανές να κάνουν λογικούς ισχυρισμούς βασισμένες στις συσχετίσεις ανάμεσα στο υποκείμενο και το αντικείμενο. Επίσης το RDF χρησιμοποιεί URIs για να προσδιορίσει τους πόρους, έτσι κάθε πόρος είναι

συνδεδεμένος με ένα μοναδικό ορισμό στο διαδίκτυο. Παρόλα αυτά καθώς το RDF μας παρέχει ένα μοντέλο και κανόνες σύνταξης για να περιγράψουμε ένα πόρο, δεν προσδιορίζει την σημασιολογία των πόρων αυτών. Για να προσδιορίσουμε πραγματικά την σημασιολογία των δεδομένων, χρειαζόμαστε το RDFS και την OWL.

Επίσης χρησιμοποιούμε το RDF για την αναπαράσταση των δεδομένων, γιατί μέσω αυτού, οι πληροφορίες χαρτογραφούνται άμεσα και απερίφραστα σε ένα μοντέλο, το οποίο είναι αποκεντρωμένο, και για το οποίο υπάρχουν ήδη πολλοί γενική parsers διαθέσιμοι. Αυτό σημαίνει πως όταν έχεις μια RDF εφαρμογή, ξέρει ποια bit δεδομένων είναι τα σημασιολογικά, και ποια είναι συμπλήρωμα.

Το RDFS είναι μια απλή γλώσσα λεξιλογίου που μας βοηθάει στο να εκφράζουμε τις σχέσεις μεταξύ των πόρων. Τέλος εκτός από το RDFS, με τη χρήση της OWL, η οποία ένα πιο πλούσιο και πιο εκφραστικό λεξιλόγιο, μας βοηθάει για τον ορισμό των οντολογιών του Semantic Web.

6.3 Semantic Web agents



Οι Semantic Web agents είναι προγράμματα υπολογιστών ικανά να ερμηνεύσουν σημασιολογικές πληροφορίες σε RDF και OWL, και είναι επίσης απαραίτητοι για την εκμετάλλευση της δύναμης του Σημασιολογικού Ιστού. Οι agents αυτοί μπορούν να βρουν τις πιθανές ανάγκες των χρηστών και να τους προσφέρουν περισσότερες επιλογές. Όπως ένας ταξιδιωτικός πράκτορας σου προσφέρει διάφορες επιλογές και συνδυασμούς για ένα ταξίδι, έτσι και ένας Semantic web agent σου προσφέρει διάφορες επιλογές ανάλογα, με τις προτιμήσεις σου, ώστε να πάρεις τις πληροφορίες που χρειάζεσαι από το διαδίκτυο. Πρέπει να σημειωθεί ότι οι agents δεν μπορούν να αντικαταστήσουν τους ανθρώπινους χρήστες, ούτε απαραίτητα παίρνουν αποφάσεις. Στις περισσότερες, αν όχι σε όλες τις

περιπτώσεις, ο ρόλος τους είναι να συλλέξουν και να οργανώσουν τις πληροφορίες, και να παρουσιάσουν τις επιλογές αυτές στους χρήστες για να επιλέξουν. Οι agents κάνουν χρήση όλων των τεχνολογιών που αφορούν το Semantic Web.

6.4 Λογική

Για να γίνει ο Σημασιολογικός Ιστός εκφραστικός αρκετά έτσι ώστε να μας βοηθήσει σε ένα ευρύ φάσμα καταστάσεων, θα καταστεί αναγκαία να κατασκευαστεί μια ισχυρή λογική γλώσσα για την εξαγωγή συμπερασμάτων. Αυτό σημαίνει πως μια αυτόματη διαδικασία εξαγωγής συμπερασμάτων μπορεί να δημιουργήσει νέες σχέσεις μεταξύ των οντοτήτων βασισμένες στα δεδομένα και σε κάποιες επιπλέον πληροφορίες με την μορφή λεξιλογίου. Στο Σημασιολογικό διαδίκτυο, οι πόροι από τέτοιες επιπλέον πληροφορίες μπορούν να προσδιοριστούν μέσω λεξιλογίων ή σαν ένα σύνολο κανόνων. Γενικά, οι οντολογίες επικεντρώνονται σε μεθόδους κατηγοριοποίησης, δίνοντας έμφαση στον προσδιορισμό των κλάσεων, των υποκλάσεων και πώς οι διάφοροι πόροι μπορούν να συσχετιστούν με τέτοιες κλάσεις, και χαρακτηρίζουν τις σχέσεις μεταξύ των κλάσεων και των αναφορών τους. Οι κανόνες από την άλλη επικεντρώνονται στον προσδιορισμό ενός γενικού μηχανισμού στην εύρεση και την παραγωγή νέων σχέσεων, βασιζόμενοι στις ήδη υπάρχουσες, περισσότερο όπως κάνουν προγράμματα λογικής. Με την εφαρμογή της λογικής στο Σημασιολογικό ιστό έχουμε ένα εργαλείο επιλογής να βελτιώσουμε την ποιότητα των ενσωματωμένων δεδομένων στο διαδίκτυο, ανακαλύπτοντας νέων σχέσεων μεταξύ των δεδομένων, αυτόματη ανάλυση του περιεχομένου των δεδομένων, ή γενικά τη διαχείριση της γνώσης του διαδικτύου.

6.5 Εφαρμογές που χρησιμοποιούν Semantic Web

Μερικές εφαρμογές που χρησιμοποιούν το Σημασιολογικό διαδίκτυο είναι οι εξής:

- Freebase

Το Freebase στοχεύει στο «να ανοίξει τις αποθήκες των πληροφοριών και να τις ενώσει μεταξύ τους». Είναι μια βάση δεδομένων η οποία περιέχει όλων των ειδών τα δεδομένα και ένα API. Επειδή είναι μια ανοιχτή βάση δεδομένων, ο οποιοσδήποτε μπορεί να εισάγει νέα δεδομένα. Θα μπορούσαμε να πούμε πως το Freebase είναι παρόμοιο με την ιστοσελίδα www.wikipedia.com. Μόλις εισάγουμε νέα δεδομένα η εφαρμογή μας κάνει προτάσεις σύμφωνα με το περιεχόμενο. Τα θέματα είναι οργανωμένα ανάλογα με τον τύπο, και μπορείς να συνδέσεις ιστοσελίδες με συνδέσμους που είναι η σημασιολογική σήμανση της σελίδας.

- Tripit

Το Tripit είναι μια εφαρμογή η οποία διαχειρίζεται τον σχεδιασμό του ταξιδίου. Χρησιμοποιώντας το Tripit, απλά προωθείς στην διεύθυνση plans@tripit.com τα εισερχόμενα email από τις διάφορες κρατήσεις που έχεις κάνει για ένα ταξίδι και η εφαρμογή αναλαμβάνει τα υπόλοιπα. Η εφαρμογή εξάγει χρήσιμες πληροφορίες από αυτά τα email και δημιουργεί μια καλά δομημένη και οργανωμένη παρουσίαση του ταξιδιού σας. Συλλέγει πληροφορίες από το Wikipedia για τα μέρη που θα επισκεφτείτε και τις παρουσιάζει συγκεντρωμένες. Τα μελλοντικά σχέδια της εφαρμογής είναι να εσωκλείουν στο πρόγραμμα πληροφορίες για το που μπορείς να πας και ποιον μπορείς να συναντήσεις.

- Apture

Το Apture παρέχει άμεση πρόσβαση σε πληροφορίες και επιτρέπει στους αναγνώστες μιας ιστοσελίδας να πλοηγηθούν στις πληροφορίες αυτές στο διαδίκτυο χωρίς να χρειάζεται να ανοίξουν νέες καρτέλες ή παράθυρα στον browser. Το Apture μετατρέπει τις επίπεδες σελίδες του διαδικτύου σε μια συνδεδεμένη διαδικτυακή εμπειρία, και έτσι οι αναγνώστες μπορούν εύκολα να πλοηγηθούν σε σχετικές πληροφορίες τις οποίες μπορούν να δουν, να ακούσουν και να διαβάσουν και έτσι να κατανοήσουν πλήρως το περιεχόμενο μιας σελίδας. Το Apture αλλάζει τον τρόπο με τον οποίο οι πληροφορίες είναι συνδεδεμένες μεταξύ τους στο διαδίκτυο. Μετατρέπει οτιδήποτε μέσα σε μια ιστοσελίδα σε ένα σύνδεσμο για περισσότερες πληροφορίες.

- Evri

Το Evri είναι μια μοναδική πρωτοποριακή προσέγγιση στην εύρεση περιεχομένου. Το Evri αυτόματα και συνεχώς δημιουργεί ευρετήρια από εκατομμύρια ειδικές θεματικές κατηγορίες, από χιλιάδες διαφορετικές πηγές οι οποίες φιλτράρονται από τον θόρυβο του διαδικτύου και προσφέρουν προσαρμοσμένες ειδήσεις στους χρήστες.

ΕΠΙΛΟΓΟΣ

Με τα Linked Data συνδέουμε δημοσιευμένα δεδομένα μεταξύ τους και αποκτάμε πληροφορίες από διάφορες πηγές στο διαδίκτυο. Με το Semantic Web όμως δίνουμε ένα σημασιολογικό προσδιορισμό στα δεδομένα αυτά έτσι ώστε οι υπολογιστές να μπορούν να κατανοούν τα δεδομένα, όχι μόνο να τα αναπαριστούν, αλλά και να μπορούν να βρίσκουν τα δεδομένα εκείνα τα οποία είναι παρόμοια ή που συνδέονται λογικά μεταξύ τους και να τα παρουσιάζουν στον χρήστη. Έτσι οι εφαρμογές σημασιολογικού διαδικτύου είναι εφαρμογές που στηρίζονται στην σημασιολογία των δεδομένων και παρουσιάζουν μια «έξυπνη» πλευρά του διαδικτύου που μπορεί να συνδέσει δεδομένα και να πάρει αποφάσεις για τα δεδομένα αυτά.

ΚΕΦΑΛΑΙΟ 7

ΒΑΣΕΙΣ ΓΡΑΦΟΥ

ΕΙΣΑΓΩΓΗ

Όλες αυτές οι διασυνδεδεμένες πληροφορίες που είναι κατανεμημένες στο διαδίκτυο βρίσκονται αποθηκευμένες σε βάσεις οι οποίες είναι βάσεις γράφου και όχι σχεσιακές. Έτσι τα δεδομένα που αποθηκεύονται στις βάσεις τέτοιου τύπου συνδέονται μεταξύ τους και είναι πιο ευέλικτα καθώς το σχήμα αναπαράστασής τους δεν είναι τόσο «σφιχτό» και μπορεί εύκολα να τροποποιηθεί.

7.1 Τι είναι οι βάσεις γράφου

Μια βάση γράφου χρησιμοποιεί δομές γράφου με κόμβους, ακμές και ιδιότητες για να αναπαραστήσει και να αποθηκεύσει τις πληροφορίες. Εξ ορισμού, βάση γράφου είναι κάθε σύστημα αποθήκευσης που παρέχει δείκτες χωρίς γειτνίαση. Γενικά οι βάσεις γράφου οι οποίες μπορούν να αποθηκεύσουν κάθε γράφο είναι διαφορετικές από εξειδικευμένες βάσεις γράφου όπως είναι τα triplestore.

Οι βάσεις γράφου βοηθούν τους προγραμματιστές να κατασκευάσουν λογισμικό, ιστοσελίδες και εφαρμογές για φορητές συσκευές τα οποία απαντούν σε πολύπλοκες ερωτήσεις οι οποίες δεν μπορούν να απαντηθούν με ένα απλό SQL ερώτημα σε μια σχεσιακή βάση δεδομένων. Έτσι μας βοηθούν στην βελτίωση της νοημοσύνης, της πρόγνωσης, της ανάλυση της κοινωνική δικτύωση, και στην απόφαση και επεξεργασία της διαχείρισης, τα οποία όλα εμπεριέχουν υψηλή περιπλοκότητα στις σχέσεις μεταξύ τους.

Σε σύγκριση με τις σχεσιακές βάσεις δεδομένων, οι βάσεις γράφου είναι συχνά γρηγορότερες για σχετικά data set, και χαρτογραφούνται πιο άμεσα στις δομές των αντικειμενοστραφών εφαρμογών. Επειδή εξαρτώνται λιγότερο από ένα «άκαμπτο» σχήμα, είναι καταλληλότερες στην διαχείριση ad-hoc και εναλλασσόμενων δεδομένων με σχήματα τα οποία συνεχώς εξελίσσονται.

7.2 Δομή

Οι βάσεις δεομένων γράφου βασίζονται στην θεωρεία των γράφων. Χρησιμοποιούν κόμβους, ιδιότητες και ακμές. Κόμβοι είναι παρόμοιοι με τα αντικείμενα στο αντικειμενοστραφή προγραμματισμό. Αναπαριστούν έννοιες όπως

άνθρωποι, πράγματα, επιχειρήσεις και κάθε είδους αντικείμενο. Οι ιδιότητες είναι συναφείς πληροφορίες οι οποίες συνδέουν τους κόμβους. Π.χ. σε ένα άνθρωπο θα μπορούσαμε να του δώσουμε ιδιότητες όπως «όνομα», «επώνυμο», «διεύθυνση» κ.α. Ακμές είναι οι «γραμμές» οι οποίες ενώνουν τους κόμβους μεταξύ τους ή τους κόμβους με τις ιδιότητες και αναπαριστούν τη σχέση ανάμεσά τους. Οι πιο σημαντικές πληροφορίες βρίσκονται στις ακμές που συνδέουν τους κόμβους ή τον κόμβο με μια ιδιότητα. Αυτές προκύπτουν αν κάποιος εξετάσει τις συνδέσεις των κόμβων, των ιδιοτήτων και των ακμών σαν πρότυπα (patterns).

7.3 Triplestores

Μια ειδική κατηγορία βάσεων γράφου είναι τα triplestore τα οποία είναι κατασκευασμένα για την αποθήκευση και την ανάκτηση RDF μεταδεδομένων. Υπάρχουν εμπορικές και open source υλοποιήσεις triplestore.

Όπως μια σχεσιακή βάση δεδομένων, έτσι και στα triplestore οι πληροφορίες που αποθηκεύονται, ανακτώνται μέσω μιας γλώσσας ερωτημάτων. Αντίθετα όμως με μια σχεσιακή βάση δεδομένων, η αποθήκευση και η ανάκτηση των δεδομένων από ένα triplestore γίνεται μέσω τριπλετών που αποτελούνται από το υποκείμενο, το κατηγόρημα και το αντικείμενο. Κάποια triplestore μπορούν να αποθηκεύουν δισεκατομμύρια τριπλετών.

Μερικά triplestore έχουν κατασκευαστεί σαν βάσεις δεδομένων από το μηδέν, ενώ άλλα φτιάχτηκαν πάνω σε ήδη υπάρχουσες εμπορικές σχεσιακές βάσεις δεδομένων. Όπως η πρώιμη ανάπτυξη των βάσεων δεδομένων OLAP, αυτή η ενδιάμεση προσέγγιση μας επέτρεψε να κατασκευάσουμε μεγάλες και δυνατές βάσεις δεδομένων με μικρό προγραμματιστικό κόπο στις αρχικές φάσεις ανάπτυξης των triplestore. Μακροπρόθεσμα όμως φαίνεται πως τα triplestore θα έχουν το πλεονέκτημα στην απόδοση. Μια δυσκολία στην υλοποίηση των triplestore πάνω σε σχεσιακές βάσεις δεδομένων που υποστηρίζουν SQL είναι ότι παρόλο που οι τριπλέτες, μπορεί μεν να αποθηκεύονται στην βάση, η υλοποίηση όμως αποτελεσματικών ερωτημάτων σε ένα RDF μοντέλο γράφου με SQL ερωτήματα είναι κάτι δύσκολο και απαιτεί αρκετό κόπο για την υλοποίησή του.

Ένα triplestore παρέχει λογικά συμπεράσματα για τα δεδομένα χρησιμοποιώντας την OWL, και για την ανάκτηση των αποτελεσμάτων την SPARQL. Τα δεδομένα αποθηκεύονται με την μορφή τριπλετών σε ειδικές αποθήκες οι οποίες αποκαλούνται repositories. Η χωρητικότητα ενός repository δεν έχει κάποιο περιορισμό και εξαρτάται από το triplestore.

Για να ανακτήσουμε τα δεδομένα από ένα triplestore, η ίδια η εφαρμογή μας παρέχει ένα endpoint, δηλαδή ένα «σημείο» το οποίο στην ουσία είναι ένα συγκεκριμένο URL, όπου μέσω του πρωτοκόλλου http εκτελούμε τα SPARQL ερωτήματα στο triplestore και λάβουμε τις απαντήσεις. Επίσης μπορούμε να χρησιμοποιήσουμε ένα SQL-client αρκεί να μπορεί να εκτελέσει SPARQL ερωτήματα. Τα δεδομένα μπορούν να επιστραφούν σε διάφορες μορφές αναλόγως με την επιλογή που θα κάνουμε. Αυτές οι μορφές μπορεί να είναι RDF/XML, JSON, N3, HTML, XML, CSV κ.α.

7.4 Μερικές υλοποιήσεις triplestore

- AllegroGraph. Το AllegroGraph είναι ένα μια βάση γράφου κλειστού κώδικα. Σχεδιάστηκε για να καταχωρεί RDF tuples, μια standard μορφή δεδομένων Linked Data. Παρέχει επίσης ένα προσαρμοσμένο browser για την αναπαράσταση των δεδομένων σαν γράφο. Το AllegroGraph βασίζεται στην γλώσσα Common Lisp.
- Oracle Database 11g Semantic Technologies. Είναι μια ανοιχτή, βασισμένη σε στάνταρ τεχνολογίες, ασφαλή και αξιόπιστη πλατφόρμα διαχείρισης RDF δεδομένων. Βασισμένη σε ένα μοντέλο γράφου, τα δεδομένα RDF μπορούν να αποθηκευτούν, να δημιουργηθούν ευρετήρια και να ερωτηθούν όπως σε άλλους αντικειμενο-σχεσιακούς τύπους δεδομένων. Η Oracle Database 11g βασίζεται στην γλώσσα Java.
- Virtuoso. Είναι ένας καινοτόμος επιχειρησιακής ποιότητας server δεδομένων για επιχειρήσεις και ιδιώτες. Παρέχει δυνατότητες διαχείρισης δεδομένων σε μορφή RDF, διαχείριση σχεσιακών δεδομένων, κ.α. Το Virtuoso βασίζεται στην γλώσσα C.

7.5 Δημοσίευση σχεσιακών δεδομένων

Πολλά triplestore παρέχουν επίσης και δυνατότητες αποθήκευσης σχεσιακών δεδομένων ή μπορούν να συνδεθούν με μια σχεσιακή βάση δεδομένων, και έτσι να μετατρέψουν αυτά τα δεδομένα σε μορφή RDF, να τα δημοσιεύσουν και να τα συνδέσουν με άλλα ήδη δημοσιευμένα.

Δημοσιεύοντας τα δεδομένα που έχουμε αποθηκευμένα, προσθέτουμε πληροφορίες σε αυτά από εξωτερικές πηγές οι οποίες έχουν και αυτές τα δεδομένα τους δημοσιευμένα, και δίνουμε επιπλέον αξία στα δεδομένα μας. Μπορούμε να χρησιμοποιήσουμε αυτές τις επιπλέον πληροφορίες από τις άλλες πηγές, για να εκτελέσουμε ερωτήματα με εξαιρετικά ισχυρό σημασιολογικό χαρακτήρα και έτσι να ανακτήσουμε πληροφορίες που δεν θα μπορούσαμε διαφορετικά. Π.χ. έχουμε μια σχεσιακή βάση δεδομένων, με δεδομένα που αφορούν τις πωλήσεις μιας εταιρίας και στα δεδομένα αυτά περιλαμβάνονται πόλεις στις οποίες η εταιρεία αποστέλλει τα προϊόντα της. Δημοσιεύοντας η εταιρεία κάποια από τα σχεσιακά δεδομένα της βάσης της και συνδέοντας τα με άλλα τα οποία βρίσκονται δημοσιευμένα στο διαδίκτυο, μπορούμε να εκτελέσουμε ερωτήματα όπως: να μας επιστραφεί ο μέσος όρος των πωλήσεων της εταιρίας στις πόλεις που έχουν πληθυσμό πάνω από κάποιο συγκεκριμένο αριθμό κατοίκων. Ο αριθμός των κατοίκων των πόλεων μπορεί να ανακτηθεί από τα δεδομένα με τα οποία έχουμε συνδέσει τα δεδομένα της εταιρείας. Αυτό θα ήταν αδύνατο να το κάνουμε έχοντας στην διάθεσή μας μόνο τα σχεσιακά δεδομένα από την βάση δεδομένων, εκτός αν είχαμε καταχωρημένες τις πληροφορίες αυτές για κάθε πόλη ξεχωριστά, κάτι όμως που θέλει αρκετό κόπο και φορτώνει την βάση της εταιρίας με «άχρηστες» πληροφορίες, τις οποίες θα έπρεπε κάποιος να τις ανανεώνει, και το πιο πιθανό είναι αυτές οι πληροφορίες να μην έχουν σχέση με το αντικείμενό της.

ΕΠΙΛΟΓΟΣ

Έτσι οι βάσεις γράφου βοηθούν στην αποθήκευση δεδομένων τα οποία μπορούμε να τα δημοσιεύσουμε και να τα ενώσουμε με άλλα έτσι ώστε να τους προσθέσουμε επιπλέον πληροφορίες. Κάνοντάς τα δημόσια μπορούν να χρησιμοποιηθούν από άλλους χρήστες εμπλουτίζοντας τα δεδομένα τους ή ενισχύοντάς τα δικά μας με επιπλέον πληροφορίες.

ΚΕΦΑΛΑΙΟ 8

ΓΡΑΦΟΣ ΚΑΙ ΠΛΗΡΟΦΟΡΙΑ

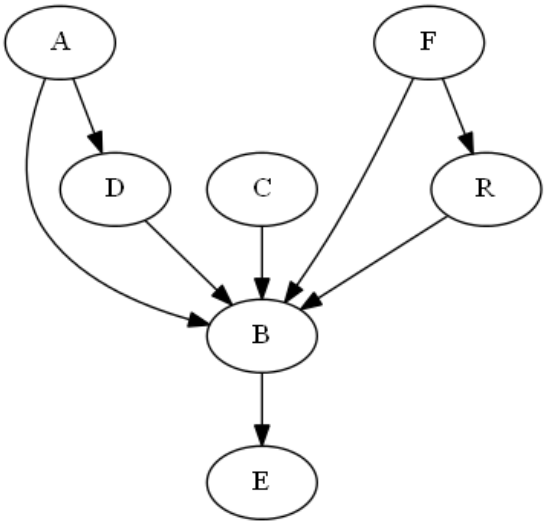
ΕΙΣΑΓΩΓΗ

Λαμβάνοντας υπόψη όλα όσα αναπτύχθηκαν μέχρι τώρα, σε αυτό το κεφάλαιο παρουσιάζονται κάποιες προτάσεις που σχετίζονται με την εξαγωγή συμπερασμάτων και χρήσιμης πληροφορίας από ένα γράφο. Οι εν λόγω προτάσεις συνιστούν αντικείμενο ερευνητικής δραστηριότητας και η οποία πρόκειται να δημοσιευτεί στο άμεσο μέλλον [19].

8.1 Υπολογισμός του κύρους ενός κόμβου σε ένα Dataset

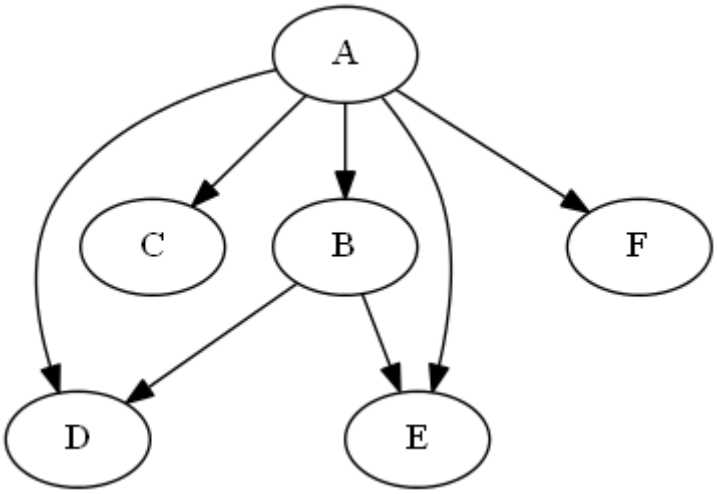
Σε ένα συγκεκριμένο dataset το οποίο αναπαρίσταται σε μορφή RDF/XML τα δεδομένα βρίσκονται πάνω σε ένα γράφο. Κάθε κόμβος στο γράφο αναπαριστά και ένα συγκεκριμένο πόρο στο dataset. Τα τόξα που ενώνουν τον γράφο αποτελούνται από τα κατηγορήματα τα οποία δίνουν μια ιδιότητα σε ένα συγκεκριμένο πόρο. Τα κατηγορήματα αυτά συνδέουν τους πόρους μεταξύ τους ή συνδέουν ένα πόρο με μία τιμή, η οποία είναι η τιμή της ιδιότητα το πόρου αυτού που του δίνει το συγκεκριμένο κατηγόρημα.

Το κύρος ενός κόμβου (Node Prestige) θα μπορούσαμε να το χωρίσουμε σε δυο μέρη, στο εισερχόμενο κύρος (input prestige) και στο εξερχόμενο κύρος (output prestige). Σαν εισερχόμενο κύρος ονομάζεται το πλήθος των κατηγορημάτων τα οποία δείχνουν σε ένα συγκεκριμένο κόμβο εμφανίζοντάς τον σαν αντικείμενο στο dataset.



Σχήμα 8.1 Εισερχόμενο κύρος κόμβου

Όπως φαίνεται στο σχήμα 8.1 το εισερχόμενο κύρος του κόμβου Β στον συγκεκριμένο dataset είναι 5 καθώς ο κόμβος αυτός εμφανίζεται σαν αντικείμενο, δηλαδή δείχνουν σε αυτόν 5 κατηγορήματα. Αντίστοιχα σαν εξερχόμενο κύρος ενός κόμβου ονομάζουμε το πλήθος των κατηγορημάτων τα οποία εξέρχονται από ένα συγκεκριμένο κόμβο ο οποίος είναι το υποκείμενό τους.



Σχήμα 8.2 Εξερχόμενο κύρος κόμβου

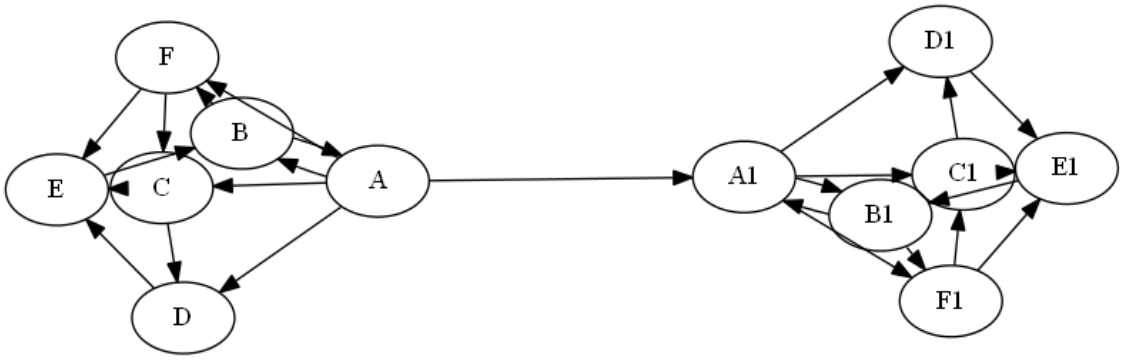
Όπως φαίνεται και στο σχήμα 8.2 το εξερχόμενο κύρος του κόμβου Α στο dataset είναι 5 καθώς ο κόμβος αυτός παρουσιάζεται σαν υποκείμενο 5 φορές, δηλαδή ξεκινάνε από αυτών 5 κατηγορήματα.

Με τον υπολογισμό του εξερχόμενου κύρους ενός κόμβου σε ένα dataset μπορούμε να πούμε πως για ένα κόμβο με υψηλό κύρος έχουμε περισσότερες πληροφορίες από τους υπόλοιπους μέσα στο dataset. Έτσι αν επεξεργαστούμε τα δεδομένα με κάποιο τρόπο, τα αποτελέσματα που θα έχουμε για τον κόμβο αυτό θα είναι πιο αξιόπιστα λόγω του ότι έχουμε περισσότερες πληροφορίες σε σχέση με τους υπόλοιπους.

Ο υπολογισμός του εισερχόμενου κύρους σε ένα dataset και συγκεκριμένα σε ένα κόμβο ο οποίος εμφανίζει να έχει υψηλό εισερχόμενο κύρος, μπορούμε να πούμε ότι είναι ένας «σημαντικός» κόμβος για το dataset καθώς παρέχει πληροφορίες για αρκετούς κόμβους μέσα σε αυτό.

8.2 Υπολογισμός των κατηγορημάτων «Γέφυρες»

Σαν «γέφυρα» σε ένα dataset ονομάζουμε ένα κατηγόρημα το οποίο ενώνει δύο κόμβους μεταξύ τους και αν διαγράψουμε αυτό το κατηγόρημα δεν υπάρχει άλλος δρόμος ώστε να φτάσουμε από τον ένα κόμβο στον άλλο. Έτσι διαγράφοντας το συγκεκριμένο κατηγόρημα κόβουμε στην ουσία το dataset σε δυο μέρη και χάνουμε μέρος της πληροφορίας.



Σχήμα 8.3 Γέφυρα σε ένα Dataset

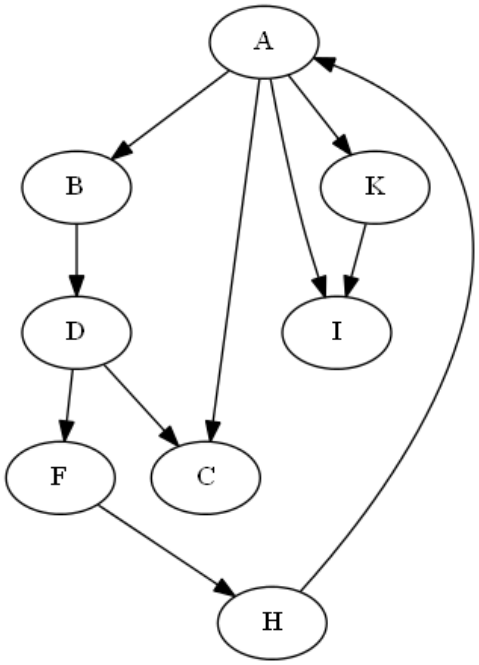
Όπως φαίνεται και στο σχήμα 8.3 το συγκεκριμένο dataset αποτελείται από δυο κομμάτια τα οποία ενώνονται μεταξύ τους με το statement που έχει σαν υποκείμενο τον κόμβο A και σαν αντικείμενο τον κόμβο A1. Έτσι αν διαγράψουμε το κατηγόρημα που ενώνει τους δυο αυτούς κόμβους, το dataset χωρίζεται σε δύο μέρη και δεν μπορούμε να έχουμε πρόσβαση σε πληροφορίες στις οποίες είχαμε πριν την διαγραφεί.

Έτσι τα statements τα οποία περιέχουν ένα κατηγόρημα «γέφυρα» μπορούμε να πούμε ότι είναι τα αδύναμα σημεία σε ένα dataset καθώς αν τα διαγράψουμε μπορούμε να το χωρίσουμε στα δύο. Ένα dataset το οποίο περιέχει πολλές «γέφυρες» μπορούμε να πούμε πως δεν είναι συμπαγές καθώς πολύ εύκολα μπορούμε αποκόψουμε πληροφορίες από αυτό.

Αν συγκεντρώσουμε όλα τα statement των οποίων τα κατηγορήματά τους είναι «γέφυρες» μπορούμε να δημιουργήσουμε ένα κρίσιμο μονοπάτι μέσα στο dataset και έτσι να αξιολογήσουμε πόσο σημαντικά ή όχι είναι τα αποτελέσματα τα οποία έχουμε πάρει κάποια άλλη διεργασία, ή ακόμα μπορούμε να διαπιστώσουμε πόσο σημαντικά ή όχι είναι τα statement για τα οποία ενδιαφερόμαστε.

8.3 Κύκλοι μέσα σε ένα Dataset

Ένα κλασικό πρόβλημα κατευθυνόμενων γράφων είναι η ανεύρεση των κύκλων οι οποίοι μπορεί να παρατηρούνται μέσα σε αυτόν. Έτσι στην περίπτωση των δεδομένων που βρίσκονται σε μορφή RDF/XML παίρνουμε σαν αρχή ένα συγκεκριμένο κόμβο και ελέγχουμε αν μπορούμε ακολουθώντας ένα μονοπάτι μέσα στον γράφο να φτάσουμε πάλι στο σημείο (κόμβο) από το οποίο ξεκινήσαμε.



Σχήμα 8.4 Κύκλοι μέσα σε ένα Dataset

Στο σχήμα 8.4 παρουσιάζεται ένα dataset στο οποίο μπορούμε να διακρίνουμε πως αν ξεκινήσουμε από τον κόμβο A, ακολουθώντας τα κατηγορήματα μπορούμε καθώς πλοηγούμαστε στο γράφο, αν ακολουθήσουμε τους κόμβους B, D, F, H να φτάσουμε πάλι στον κόμβο A.

8.4 Στατιστικά στοιχεία για ένα δεδομένο Dataset

Τέλος σε ένα συγκεκριμένο dataset μπορούμε να υπολογίσουμε ορισμένα στατιστικά στοιχεία, τα οποία μπορούν να μας βοηθήσουν στο να βγάλουμε διάφορα συμπεράσματα για τη συνοχή των δεδομένων σε ένα dataset.

8.4.1 Bridge/Statements percentage

Το να εφαρμόσουμε μια κλασική τεχνική γράφου για την ανεύρεση της συνοχής των δεδομένων σε ένα κατευθυνόμενο γράφο δεν θα ταίριαζε σε δεδομένα Linked Data γιατί με αυτό τον τρόπο θα έπρεπε να διαγράψουμε statements και κατά

συνέπεια πληροφορίες για το dataset, διότι οι πληροφορίες αναπαρίστανται μέσω των συνδέσεων αυτών. Άρα δεν έχει νόημα η διαγραφεί πληροφοριών ώστε να βρούμε την συνοχή τους. Έτσι πρέπει να βρούμε ένα άλλο τρόπο για να προσδιορίσουμε την συνοχή των δεδομένων μέσα στο γράφο των πληροφοριών.

Με την στατιστική μέθοδο Bridge/Statements percentage μετράμε το ποσοστό των statements τα οποία είναι «γέφυρες» προς το σύνολο των statement μέσα σε ένα dataset. Με αυτό τον τρόπο μπορούμε να πούμε πως ένα dataset του οποίου το ποσοστό αυτό είναι πολύ υψηλό, η συνοχή των δεδομένων του δεν είναι υψηλή καθώς μπορούμε εύκολα να το αποκόψουμε. Αν πάλι είναι χαμηλό σημαίνει πως οι περισσότεροι από τους κόμβους μας είναι συνδεμένοι μεταξύ τους περισσότερες από μια φορές, και κατ’ επέκταση και το dataset αυτό είναι ισχυρά συνδεδεμένο και δύσκολα μπορούμε να το χωρίσουμε και να χάσουμε πληροφορίες.

8.4.2 Mean reachability

Σαν προσβασιμότητα (reachability) ορίζουμε τον αριθμό τον κόμβων στους οποίους μπορούμε να φτάσουμε σε ένα dataset ξεκινώντας από ένα συγκεκριμένο κόμβο. Έτσι mean reachability είναι η μέση τιμή των κόμβων τους οποίους μπορούμε να φτάσουμε σε ένα dataset από όλους τους κόμβους μέσα σε αυτό. Με αυτό τον τρόπο μπορούμε να πάρουμε χρήσιμες πληροφορίες για τις συνδέσεις μέσα σε dataset και πως μπορούμε να πλοηγηθούμε μέσα σε αυτό.

Αν χρησιμοποιήσουμε τα αποτελέσματα αυτής της μεθόδου μαζί με αυτά της μεθόδου υπολογισμού του ποσοστού «bridges/statements» μπορούμε να έχουμε μια πολύ ξεκάθαρη εικόνα για το dataset και την συνοχή των δεδομένων μέσα σε αυτό.

ΕΠΙΛΟΓΟΣ

Η εξόρυξη πληροφορίας από δεδομένα γράφου βρίσκεται ακόμα σε πολύ αρχικό στάδιο καθώς οι τεχνολογίες που αναπαριστούν διασυνδεμένα δεδομένα είναι πολύ καινούργιες. Αυτή ήταν μια προσέγγιση του προβλήματος, κατά την οποία έγινε μια προσπάθεια να εφαρμόσουμε κλασικούς αλγόριθμους γράφων στα Linked Data, να εξάγουμε κάποια στατιστικά στοιχεία και να υπολογίσουμε κάποια σημαντικά χαρακτηριστικά σε ένα dataset. Η έρευνα πάνω στο συγκεκριμένο τομέα μπορεί να συνεχιστεί και να επεκταθεί προς διάφορες κατευθύνσεις δίνοντας περισσότερα αποτελέσματα για την εξόρυξη πληροφοριών από δεδομένα γράφου.

ΚΕΦΑΛΑΙΟ 9

ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ

ΕΙΣΑΓΩΓΗ

Οι παραπάνω προτάσεις αναπτύχθηκαν σε συναρτήσεις χρησιμοποιώντας την γλώσσα προγραμματισμού JAVA. Οι συναρτήσεις αυτές υπολογίζουν τις προτάσεις που προαναφέρθηκαν για ένα ολόκληρο dataset το οποίο βρίσκεται σε κάποια από τις μορφές RDF/XML, N3, N-Triples. Για την υλοποίηση των συναρτήσεων αυτών χρησιμοποιήθηκε μια βιβλιοθήκη της JAVA που ονομάζεται Jena (αναφορά στο Παράρτημα Ι). Επίσης υλοποιήθηκαν παραλλαγές των κύριων συναρτήσεων ώστε να μπορούν να εφαρμοστούν όχι μόνο σε ολόκληρο το dataset αλλά και σε μέρος αυτού, αναλόγως με τις παραμέτρους που δίνουμε στην συνάρτηση, καθώς επίσης και ορισμένες συναρτήσεις από αυτές να μας επιστραφούν αποτελέσματα σε διαφορετική μορφή από την κύρια συνάρτηση. Τα παρακάτω αποτελέσματα όπως και οι προτάσεις που αναφέρθηκαν στο προηγούμενο κεφάλαιο συνιστούν αντικείμενο έρευνας η οποία βρίσκεται σε εξέλιξη [19].

9.1 Συναρτήσεις σε γλώσσα προγραμματισμού JAVA

9.1.1 Συναρτήσεις υπολογισμού κύρους ενός κόμβου

Οι συναρτήσεις που αναπτύχθηκαν για τον υπολογισμό του κύρους ενός κόμβου χωρίζονται σε δύο κατηγορίες. Στην κατηγορία των συναρτήσεων που υπολογίζουν το εισερχόμενο κύρος ενός κόμβου, και στις συναρτήσεις που υπολογίζουν το εξερχόμενο κύρος του.

Για τον υπολογισμό του εισερχόμενου και του εξερχόμενου κύρους αναπτύχθηκαν συναρτήσεις οι οποίες επιστέφουν το κύρος όλων των κόμβων ενός dataset στην στάνταρ δομή που μας παρέχει η JAVA Map <String, Integer> (ο κώδικας επισυνάπτεται στο Παράρτημα ΙΙ). Στο πρώτο μέρος κάθε αποτελέσματος είναι ένα String το οποίο είναι το όνομα του κόμβου μέσα στο dataset. Στο δεύτερο μέρος βρίσκεται ένας Integer (ακέραιος) ο οποίος μας δείχνει το εισερχόμενο ή εξερχόμενο κύρος του συγκεκριμένου κόμβου.

Οι συναρτήσεις αυτές στην κύριά τους μορφή αναπτύχθηκαν για να υπολογίζουν το κύρος ενός κόμβου σε ολόκληρο το dataset. Οι κύριες αυτές συναρτήσεις υπερφορτώθηκαν ώστε να μπορούμε να τις εφαρμόσουμε και να έχουμε αποτελέσματα από ένα υποσύνολο του αρχικού dataset το οποίο το

επιλέγουμε εμείς. Αυτό γίνεται περνώντας σαν όρισμα στην συνάρτηση μια λίστα με τα κατηγορήματα τα οποία μας ενδιαφέρουν από το συγκεκριμένο dataset. Έτσι δημιουργούμε ένα νέο dataset (μοντέλο) το οποίο περιέχει μόνο τα statement τα οποία περιλαμβάνουν τα συγκεκριμένα κατηγορήματα και εφαρμόζουμε τον αλγόριθμο υπολογισμού του κύρους σε αυτό το νέο μοντέλο.

Επίσης μπορούμε να εξειδικεύσουμε κι άλλο τον υπολογισμό του κύρους, όχι μόνο σε ένα υποσύνολο του αρχικού dataset, αλλά επίσης μπορούμε να υπολογίσουμε το κύρος ενός κόμβου μόνο για ορισμένα κατηγορήματα. Αυτό γίνεται περνώντας σαν όρισμα στην συνάρτηση τα κατηγορήματα που μας ενδιαφέρουν και υπολογίζουμε το κύρος των κόμβων στο dataset, μόνο για τα statements τα οποία περιλαμβάνουν τα συγκεκριμένα κατηγορήματα. Η επιλογή αυτή μπορεί να γίνει είτε σε ολόκληρο το dataset, είτε δημιουργώντας ένα υποσύνολο αυτού όπως αναφέρθηκε πιο πάνω.

Ακόμα στις συναρτήσεις με τις οποίες υπολογίζουμε το εισερχόμενο κύρος έχουμε την δυνατότητα να επιλέξουμε αν θέλουμε ή όχι να εμφανίζονται στα αποτελέσματα εκτός από τους κόμβους και τα Literal τα οποία εμφανίζονται μέσα στο dataset.

Τέλος η μορφή των επιστρεφόμενων αποτελεσμάτων από τις συναρτήσεις υπολογισμού του κύρους ενός κόμβου μπορεί να είναι διαφορετική. Εκτός από τη στάνταρ μορφή της δομής δεδομένων Map που μας παρέχει η JAVA, τα αποτελέσματα μπορούν να μας επιστραφούν και σαν ένα αντικείμενο τύπου PrestigeArray το οποίο ορίστηκε από εμάς (ο κώδικας περιέχεται στο Παράρτημα ΙΙ) το οποίο μας παρέχει μεθόδους για την διαχείριση των αποτελεσμάτων αυτών σαν αντικείμενο.

9.1.2 Συναρτήσεις υπολογισμού κατηγορημάτων «Γέφυρες»

Οι συναρτήσεις αυτές υπολογίζουν τα κατηγορήματα «Γέφυρες» σε ένα συγκεκριμένο dataset. Η κύρια συνάρτηση υπολογίζει τα κατηγορήματα αυτά από όλα τα statement μέσα στο dataset και τα επιστρέφει σε μορφή ενός αντικειμένου τύπου Model το οποίο ορίζεται από την βιβλιοθήκη Jena που χρησιμοποιήθηκε για την υλοποίηση των συναρτήσεων (ο κώδικας περιέχεται στο Παράρτημα ΙΙ). Η βιβλιοθήκη αυτή μας παρέχει πολλές μεθόδους, τις οποίες μπορούμε να τις

χρησιμοποιήσουμε για να διαχειριστούμε τα δεδομένα που περιέχονται στο αντικείμενο αυτό, καθώς επίσης και να τα αποθηκεύσουμε ή να τα αναπαραστήσουμε.

Η κύρια μέθοδος έχει υπερφορτωθεί όπως και οι προηγούμενες ώστε να δουλεύει όχι μόνο με ολόκληρο το dataset, αλλά και με ένα υποσύνολό του το οποίο δημιουργείται από εμάς, περνώντας σαν όρισμα στην μέθοδο μια λίστα με τα κατηγορήματα τα οποία θέλουμε να περιλαμβάνονται στο καινούργιο αυτό dataset. Έτσι υπολογίζονται οι «Γέφυρες» μόνο για τα κατηγορήματα που μας ενδιαφέρουν.

Ακόμα μπορούμε να διαλέξουμε από ολόκληρο το dataset ή από το υποσύνολο του dataset που δημιουργήσαμε εμείς με την επιλογή συγκεκριμένων κατηγορημάτων, ποια κατηγορήματα μας ενδιαφέρουν περνώντας σαν όρισμα στην συνάρτηση μια λίστα που τα περιέχει. Έτσι μας επιστρέφονται μόνο τα statements «Γέφυρες» τα οποία περιέχουν τα συγκεκριμένα κατηγορήματα.

Τέλος μας δίνεται η δυνατότητα να επιλέξουμε αν στα αποτελέσματα που μας επιστρέφονται θα περιλαμβάνονται τα Literal ή όχι, καθώς σε ένα dataset όπου παρουσιάζεται ένα Literal αυτόματα το κατηγόρημα το οποίο το συνδέει με τον κόμβο, είναι κατηγόρημα «Γέφυρα» γιατί τα Literal είναι σταθερές τιμές ιδιοτήτων και δεν μπορούν να έχουν δικές τους ιδιότητες. Έτσι αποκλείοντας τα Literal από τα αποτελέσματα, καθαρίζουμε τα αποτελέσματα αυτά από τις «Γέφυρες» οι οποίες δεν ενώνουν πόρους μεταξύ τους αλλά δίνουν τιμές σε ιδιότητες.

9.1.3 Συναρτήσεις υπολογισμού κύκλων σε ένα Dataset

Σε ένα dataset το οποίο αναπαρίσταται σαν ένας κατευθυνόμενος γράφος, πολύ πιθανό είναι να παρατηρούνται κύκλοι, δηλαδή αν ξεκινήσουμε από ένα κόμβο στο dataset ακολουθώντας τις συνδέσεις που ενώνουν τους κόμβους μεταξύ τους, μπορούμε να φτάσουμε στον κόμβο από τον οποίο ξεκινήσαμε. Με αυτήν την συνάρτηση υπολογίζουμε τους κύκλους αυτούς μέσα στο dataset και τους επιστρέφουμε σε μια λίστα με την μορφή αντικειμένων τύπου Path, το οποίο ορίζεται από την βιβλιοθήκη Jena (ο κώδικας περιέχεται στο Παράρτημα ΙΙ).

Επίσης μπορούν να μας επιστραφούν οι κύκλοι από το dataset μόνο για τα κατηγορήματα τα οποία μας ενδιαφέρουν. Σε περίπτωση που θεωρούμαι ένα

αριθμό κατηγορημάτων πιο σημαντικά από τα άλλα και ότι μας παρέχουν περισσότερες πληροφορίες, μπορούμε να περάσουμε μια λίστα με τα κατηγορήματα αυτά σαν όρισμα στην μέθοδο και να μας επιστραφούν οι κύκλοι οι οποίοι περιέχουν μόνο τα κατηγορήματα που επιλέξαμε.

Ακόμα μπορούμε να υπολογίσουμε τους κύκλους μέσα στο dataset, για ένα συγκεκριμένο κόμβο. Περνώντας σαν όρισμα το όνομα του κόμβου αυτού, αν αυτός εμφανίζεται σε κάποιο κύκλο, τότε μας επιστρέφεται ένα αντικείμενο τύπου Path που περιέχει την διαδρομή αυτή. Αν δεν παρουσιάζεται κάποιος κύκλος με αρχή τον συγκεκριμένο κόμβο, τότε επιστρέφεται η τιμή null. Τέλος για την συγκεκριμένη μέθοδο εκτός από τον κόμβο που μας ενδιαφέρει, μπορούμε να επιλέξουμε και τα κατηγορήματα από τα οποία θέλουμε να αποτελείται ο κύκλος αυτός, και έτσι μας επιστρέφεται ο κύκλος μόνο αν περιέχει τα συγκεκριμένα κατηγορήματα.

9.1.4 Συναρτήσεις υπολογισμού Closure ενός κόμβου

Το closure ενός κόμβου είναι μια συνάρτηση η οποία παίρνει σαν αρχή ένα συγκεκριμένο κόμβο και υπολογίζει τους κόμβους τους οποίους μπορούμε να φτάσουμε μέσα στο συγκεκριμένο dataset. Σαν όρισμα δίνουμε το όνομα του κόμβου και παίρνουμε σαν αποτέλεσμα ένα αντικείμενο τύπου Model το οποίο ορίζεται από την βιβλιοθήκη Jena το οποίο περιέχει όλα τα statements στα οποία μπορούμε να φτάσουμε αν ξεκινήσουμε από τον κόμβο που εισάγαμε σαν όρισμα (ο κώδικας περιέχεται στο Παράρτημα ΙΙ).

Επίσης και αυτή η μέθοδος είναι υπερφορτωμένη και μπορούμε να δώσουμε σαν όρισμα μια λίστα με τα κατηγορήματα που μας ενδιαφέρουν και να μας επιστραφούν μόνο τα statements τα οποία περιλαμβάνουν τα συγκεκριμένα κατηγορήματα.

9.1.5 Συναρτήσεις υπολογισμού στατιστικών στοιχείων σε ένα Dataset

Μέσω τον συναρτήσεων αυτών μπορούμε να υπολογίσουμε στατιστικά στοιχεία τα οποία μας βοηθούν στην εξαγωγή συμπερασμάτων σχετικά με την συνοχή των δεδομένων σε ένα dataset.

Ένα στατιστικό στοιχείο που μπορούμε να υπολογίσουμε με αυτές τις συναρτήσεις είναι το ποσοστό «Bridges/Statements» που παρουσιάζονται μέσα

στο dataset. Με την συνάρτηση αυτή μας επιστρέφεται μια float τιμή που αντιπροσωπεύει το ποσοστό των statements που είναι γέφυρες προς τον συνολικό αριθμό των statements μέσα στο dataset (ο κώδικας περιέχεται στο Παράρτημα ΙΙ).

Η συνάρτηση αυτή είναι υπερφορτωμένη και λειτουργεί όχι μόνο για ολόκληρο το dataset αλλά και για ένα μέρος του, αν επιλέξουμε τα κατηγορήματα που μας ενδιαφέρουν και τα εισάγουμε σαν παράμετρο στην συνάρτηση αυτή. Επίσης μπορούμε να επιλέξουμε ποια κατηγορήματα μας ενδιαφέρουν για τον υπολογισμό των «Γεφυρών» μέσα σε ολόκληρο το dataset ή σε μέρος αυτού. Τέλος μπορούμε να επιλέξουμε αν θέλουμε στο ποσοστό που θα μας επιστραφεί να περιλαμβάνονται τα statements «Γέφυρες» που έχουν σαν αντικείμενο Literal.

Άλλη μια συνάρτηση για τον υπολογισμό στατιστικών στοιχείων είναι η συνάρτηση για τον υπολογισμό της μέσης τιμής των κόμβων στους οποίους μπορούμε να φτάσουμε από όλους τους κόμβους σε ένα dataset. Σαν αποτέλεσμα μας επιστρέφεται ένας float ο οποίος αντιπροσωπεύει την τιμή αυτή (ο κώδικας περιέχεται στο Παράρτημα ΙΙ).

Για την συνάρτηση αυτή παρέχεται επίσης και μια υπερφορτωμένη μέθοδο στην οποία δίνουμε σαν όρισμα μια λίστα, η οποία περιέχει τα κατηγορήματα που μας ενδιαφέρουν, και μας επιστρέφεται η μέση τιμή των κόμβων στους οποίους μπορούμε να φτάσουμε μέσα στο dataset ακολουθώντας μόνο τα κατηγορήματα που έχουμε θέσει σαν όρισμα.

9.2 Dataset

Για την δοκιμή των συναρτήσεων αυτών δημιουργήθηκε ειδικά ένα dataset το οποίο περιλαμβάνει τις χώρες της λατινικής Αμερικής. Οι πληροφορίες αυτές αντλήθηκαν από το DBpedia.org το οποίο περιέχει τα δημοσιευμένα δεδομένα σε μορφή Linked Data τα οποία προέρχονται από την ιστοσελίδα www.wikipedia.com και δημιουργήθηκε το ανάλογο dataset. Στα δεδομένα αυτά περιέχονται πληροφορίες για όλες τις χώρες της λατινικής Αμερικής, όπως ακριβώς παρουσιάζονται από την πηγή από όπου αντλήθηκαν. Οι πληροφορίες αυτές περιλαμβάνουν τον πληθυσμό κάθε χώρας, με ποιες χώρες συνορεύει, ποια βουνά και ποτάμια βρίσκονται στο έδαφός της, τις πρωτεύουσές τους, τις σελίδες

τους στο www.wikipedia.com και άλλες. Το συγκεκριμένο dataset αποτελείται από 16038 τριπλέτες. Επίσης επιλέχθηκε τυχαία, μια λίστα από κατηγορήματα. Από την λίστα αυτή δημιουργήθηκε μια δεύτερη μικρότερη λίστα επιλέγοντας πάλι τυχαία τα κατηγορήματα που την αποτελούν. Οι δυο αυτές λίστες χρησιμοποιήθηκαν για την δοκιμή των υπερφορτωμένων συναρτήσεων.

9.3 Αποτελέσματα

Εφαρμόζοντας τις συναρτήσεις αυτές στο παραπάνω dataset πήραμε τα ακόλουθα αποτελέσματα:

Μετά την εκτέλεση της συνάρτησης για τον υπολογισμό το εισερχόμενου κύρους σε ολόκληρο το dataset ο κόμβος με το υψηλότερο κύρος είναι ο κόμβος “http://dbpedia.org/resource/Argentina” που αντιπροσωπεύει την Αργεντινή με τιμή 2001. Αυτό σημαίνει πως ο κόμβος που αντιπροσωπεύει την Αργεντινή φαίνεται να παρουσιάζεται σαν αντικείμενο 2001 φορές, τις περισσότερες από κάθε άλλο στο dataset. Έπειτα καλέσαμε την μέθοδο υπολογισμού εισερχόμενου κύρους για μια λίστα κατηγορημάτων τα οποία είχαν προεπιλεγεί τυχαία και τα αποτελέσματα είναι πως ο κόμβος με το μεγαλύτερο εισερχόμενο κύρος είναι “http://dbpedia.org/resource/Colombia” που αντιπροσωπεύει την Κολομβία με τιμή 1377. Τέλος εκτελέστηκε η συνάρτηση με ορίσματα και τις δυο λίστες που είχαμε δημιουργήσει, χρησιμοποιώντας την πρώτη για την δημιουργία ενός καινούργιου dataset από το αρχικό έχοντας μόνο τα κατηγορήματα της λίστας, και έπειτα υπολογίσαμε το εισερχόμενο κύρος μόνο για τα κατηγορήματα της δεύτερης λίστας. Ο κόμβος με το υψηλότερο κύρος είναι “http://dbpedia.org/resource/Colombia” με τιμή 1289. Τα αποτελέσματα παρουσιάζονται συγκεντρωμένα στον παρακάτω πίνακα:

Εισερχόμενο κύρος κόμβου		
	Όνομα κόμβου:	Τιμή:
Κόμβος με το υψηλότερο εισερχόμενο κύρος	“http://dbpedia.org/resource/Argentina”	2001
Κόμβος με το υψηλότερο εισερχόμενο κύρος μετά την κατασκευή νέου dataset από	“http://dbpedia.org/resource/Colombia”	1377

λίστα κατηγορημάτων		
Κόμβος με το υψηλότερο εισερχόμενο κύρος μετά την κατασκευή νέου dataset από λίστα κατηγορημάτων και επιλογή κατηγορημάτων	“http://dbpedia.org/resource/Colombia”	1289

Πίνακας 9.1 Αποτελ. Εισερχόμενου κύρους

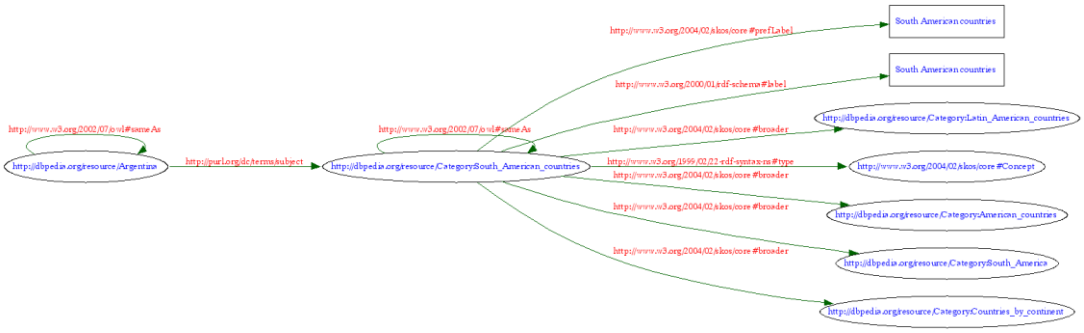
Η εκτέλεση της συνάρτησης για τον υπολογισμό του εξερχόμενου κύρους των κόμβων στο dataset έδωσε σαν αποτέλεσμα πως ο κόμβος με το μεγαλύτερο εξερχόμενο κύρος είναι “http://dbpedia.org/resource/Suriname” με τιμή 182. Έπειτα εκτελέστηκε η συνάρτηση και δόθηκε σαν όρισμα η πρώτη λίστα με τα κατηγορήματα. Το αποτέλεσμα που προέκυψε είναι πως ο κόμβος με το υψηλότερο κύρος είναι ο “http://dbpedia.org/resource/ System_of_Cooperation_Among_the_American_Air_Forces” με τιμή 8. Τέλος εκτελέστηκε η συνάρτηση με ορίσματα και τις δυο λίστες. Το αποτέλεσμα της συνάρτησης αυτής είναι πως ο κόμβος με υψηλότερο εξερχόμενο κύρος είναι “http://dbpedia.org/resource/System_of_Cooperation_Among_the_American_Air_ Forces” με τιμή 8. Τα αποτελέσματα παρουσιάζονται συγκεντρωμένα στον παρακάτω πίνακα:

Εξερχόμενο κύρος κόμβου		
	Όνομα κόμβου:	Τιμή:
Κόμβος με το υψηλότερο εξερχόμενο κύρος	“http://dbpedia.org/resource/Suriname”	182
Κόμβος με το υψηλότερο εξερχόμενο κύρος μετά την κατασκευή νέου dataset από λίστα κατηγορημάτων	“http://dbpedia.org/resource/ System_of_Cooperation_Among_the_American_Air_Forces”	8
Κόμβος με το υψηλότερο εξερχόμενο κύρος μετά την κατασκευή νέου dataset από λίστα κατηγορημάτων και επιλογή κατηγορημάτων	“http://dbpedia.org/resource/ System_of_Cooperation_Among_the_American_Air_Forces”	8

Πίνακας 9.2 Αποτελ. Εξερχόμενου κύρους

Μετά την εκτέλεση των συναρτήσεων για τον υπολογισμό των κατηγορημάτων που είναι «Γέφυρες» μέσα στο dataset το αποτέλεσμα ήταν πως στο dataset παρουσιάζονται 12418 statement των οποίων το κατηγόρημα είναι «Γέφυρα».

Κατόπιν εκτελέστηκε η συνάρτηση για τον υπολογισμό του Closure για τον κόμβο που αντιπροσωπεύει την Αργεντινή. Τα αποτελέσματα αναπαρίστανται γραφικά με την βοήθεια του εργαλείου που παρέχεται από την ιστοσελίδα http://www.w3.org/RDF/Validator/. Τα αποτελέσματα της συνάρτησης είναι τα ακόλουθα:



Σχήμα 9.1 Αποτελ. Συνάρτησης Closure

Η εκτέλεση των στατιστικών μεθόδων έδωσε τα ακόλουθα αποτελέσματα:

Ο υπολογισμός του ποσοστού «Bridges/Statements» έδειξε πως στο συγκεκριμένο dataset έχουμε 16038 statements από τα οποία τα 12418 είναι «Γέφυρες». Έτσι το ποσοστό των «Γεφυρών» μέσα στο dataset είναι 77%. Αυτό μας δείχνει πως η συνοχή του dataset που χρησιμοποιούμε δεν είναι αρκετά υψηλή και πως μπορούμε εύκολα να το αποκόψουμε. Έπειτα εκτελέστηκε η συνάρτηση δίνοντας σαν όρισμα την πρώτη λίστα ώστε να υπολογιστούν οι γέφυρες μόνο για τα κατηγορήματα της λίστας. Το ποσοστό που μας επιστράφηκε είναι 73%, αρκετά υψηλό και αυτό. Τέλος εκτελέστηκε η συνάρτηση έχοντας σαν ορίσματα και τις δυο λίστες ώστε να δημιουργηθεί ένα νέο dataset από την πρώτη λίστα και εξετάσουμε μόνο τις «γέφυρες» που περιέχουν κάποιο κατηγόρημα από την δεύτερη λίστα. Η συνάρτηση αυτή επέστρεψε το ποσοστό 93%. Τα αποτελέσματα παρουσιάζονται συγκεντρωμένα στον παρακάτω πίνακα:

Ποσοστό «Bridges/Statements»	
Ποσοστό σε:	Ποσοστό:
Ολόκληρο το dataset	77
Dataset που δημιουργήθηκε από πρώτη λίστα κατηγορημάτων	73
Dataset που δημιουργήθηκε από την πρώτη λίστα κατηγορημάτων και χρήση της δεύτερης λίστας για προσδιορισμό των Γεφυρών	93

Πίνακας 9.3 Αποτελέσματα ποσοστών

Τέλος εκτελέστηκε η συνάρτηση υπολογισμού του mean reachability για ολόκληρο το dataset. Το αποτέλεσμα που μας επιστράφηκε είναι πως η μέση τιμή των κόμβων που μπορούμε να έχουμε πρόσβαση από όλους τους κόμβους του dataset είναι 257,15933.

Έπειτα εκτελέστηκε η συνάρτηση δίνοντας σαν όρισμα την πρώτη λίστα των κατηγορημάτων ώστε να δημιουργηθεί ένα νέο dataset και το αποτέλεσμα που μας επιστράφηκε για την μέση τιμή των κόμβων που έχουμε πρόσβαση στο νέο αυτό dataset είναι 4.7225914. Τα αποτελέσματα παρουσιάζονται συγκεντρωμένα στον παρακάτω πίνακα:

Μέσος όρος πρόσβασης από ένα κόμβο	
Dataset:	Κόμβοι:
Ολόκληρο το dataset	257,15933
Dataset που δημιουργήθηκε από πρώτη λίστα κατηγορημάτων	4.7225914

Πίνακας 9.4 Μέσος όρος πρόσβασης κόμβων

ΕΠΙΛΟΓΟΣ

Εφαρμόζοντας αυτές τις συναρτήσεις μπορούμε να εξάγουμε χρήσιμα συμπεράσματα που αφορούν δεδομένα τα οποία βρίσκονται σε μορφή γράφου. Τα αποτελέσματα αυτά μπορούν να μας δώσουν πληροφορίες για την συνοχή των δεδομένων καθώς και άλλες πληροφορίες για το πώς συνδέονται τα δεδομένα μεταξύ τους. Αυτή ήταν μια πρώτη προσέγγιση του προβλήματος και στο μέλλον θα μπορούσαν να αναπτυχθούν τεχνικές οι οποίες θα μας βοηθούν να κάνουμε data mining στα δεδομένα αυτά, μπορεί όχι με την μορφή που εφαρμόζεται στα σχεσιακά δεδομένα αλλά όμως με τα ίδια αποτελέσματα. Κάποιες ερευνητικές ομάδες έχουν προχωρήσει προς αυτή την κατεύθυνση και σαν μελλοντική εργασία της δουλειάς αυτής που έχει γίνει μέχρι σήμερα, είναι η σύνδεση των αποτελεσμάτων ώστε να μπορέσουμε να έχουμε μια πιο ολοκληρωμένη εικόνα για τα δεδομένα που έχουμε διαθέσιμα.

ΣΥΜΠΕΡΑΣΜΑΤΑ

Η ανάγκη στις μέρες μας για αναζήτηση και εκμετάλλευση πληροφοριών από εφαρμογές ή και από απλούς χρήστες είναι μεγάλη. Όταν μάλιστα η ανάγκη αυτή απευθύνεται σε μια τεράστια και ανεξέλεγκτα αναπτυσσόμενη βάση δεδομένων όπως το διαδίκτυο, γίνεται μια απίστευτα δύσκολη διαδικασία. Ο διαχωρισμός των πληροφοριών σε χρήσιμες και μη, και η λογική σύνδεση μεταξύ τους για μια πιο ολοκληρωμένη πληροφόρηση, απαιτεί μια χρονοβόρα διαδικασία η οποία μάλιστα δεν μπορεί να γίνει αποκλειστικά από τους υπολογιστές, και πετυχαίνεται με την παρέμβαση του ανθρώπου, για τον πολύ απλό λόγο ότι οι υπολογιστές δεν κατανοούν την σημασία των πληροφοριών που επεξεργάζονται, αλλά παρά μόνο μπορούν να τις αναπαραστήσουν.

Την λύση σε αυτό το πρόβλημα έδωσε η ανάπτυξη τεχνολογιών όπως τα Linked Data και το Semantic Web. Οι τεχνολογίες αυτές συνδέουν πληροφορίες οι οποίες βρίσκονται αποθηκευμένες διάσπαρτα στο διαδίκτυο σε διαφορετικά μέρη, δίνοντάς τους πρόσθετες πληροφορίες με την μορφή ιδιοτήτων, και εμπλουτίζοντάς τες με σημασιολογικό περιεχόμενο. Με αυτό τον τρόπο οι υπολογιστές μπορούν να συνδέουν πληροφορίες ανάλογα με την σημασιολογία που τους έχει δοθεί, και να μας επιστρέψουν πληροφορίες από διάφορες πηγές οι οποίες σχετίζονται μεταξύ τους χωρίς την παρέμβαση του ανθρώπου.

Αυτό μας δίνει ένα τρομερό πλεονέκτημα στη διαχείριση των πληροφοριών καθώς απλοποιούνται και αυτοματοποιούνται όλες οι διαδικασίες αναζήτησης και διασύνδεσης των πληροφοριών αυτών. Έτσι η δημοσίευση των δεδομένων μας στο διαδίκτυο και η διασύνδεσή τους με ήδη υπάρχοντα δημοσιευμένα δεδομένα, αποτελεί μια ενέργεια που βοηθάει κατά την αναζήτηση να έχουμε αποτελεσματικότερη πληροφόρηση για το αντικείμενο που αναζητάμε. Οι πληροφορίες που μας επιστρέφονται είναι εμπλουτισμένες με ένα σημασιολογικό περιεχόμενο το οποίο μπορεί να χρησιμοποιηθεί για την κατανόηση των πληροφοριών αυτών από τους υπολογιστές, και να τις χρησιμοποιήσουν ώστε να δώσουν επιπλέον πληροφορίες σε άλλες οντότητες, οι οποίες είναι σημασιολογικά όμοιες.

Το διαδίκτυο τείνει να μετατραπεί σε ένα τεράστιο συνδεδεμένο γράφο πληροφοριών, όπου η πληροφορία θα έχει αξία και σημασία όχι μόνο για τους ανθρώπους αλλά και για τους υπολογιστές. Μπορεί αυτό να μην είναι απόλυτα εφαρμόσιμο στις μέρες μας και να μην επηρεάζει άμεσα αυτή τη στιγμή τους χρήστες του, αλλά η συνεχείς ανάπτυξη προς αυτή την κατεύθυνση στα επόμενα χρόνια θα μας δώσει την δυνατότητα να χρησιμοποιούμε ποιο αποτελεσματικά τις πληροφορίες που υπάρχουν διαθέσιμες, και να τις αξιοποιούμε όχι μόνο στατικά, αλλά και σημασιολογικά, παρέχοντας στους χρήστες τη δυνατότητα να προσαρμόσουν όλο αυτό τον τεράστιο όγκο πληροφοριών που είναι διαθέσιμος στα μέτρα τους.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. About SPARQL: http://www.thefigtrees.net/lee/sw/sparql-faq (Ανάκτηση 30 Αυγούστου 2011)
2. Antoniou Grigoris, Frank van Harmelen (2008), *A semantic Web primer*, The MIT Press Cambridge, Massachusetts London, England.
3. DuCharme Bob (2011), *Learning SPARQL: Querying and Updating with SPARQL 1.1*, O'Reilly Media, United States of America.
4. Heath Tom, Bizer Christian (2011), *Linked Data: Evolving the Web into a Global Data Space*, Morgan & Claypool, London, England.
5. Linked Data Book: http://linkeddatabook.com/editions/1.0/ (Ανάκτηση 25 Αυγούστου 2011)
6. Linked Data Tools: http://www.linkeddatatools.com/ (Ανάκτηση 25 Αυγούστου 2011)
7. RDF: http://www.w3.org/TR/rdf-primer/ (Ανάκτηση 25 Αυγούστου 2011)
8. Resource Description Framework: http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/ (Ανάκτηση 25 Αυγούστου 2011)
9. Segaran Toby, Evans Colin, Taylor Jamie (2009), *Programming the Semantic Web*, O'Reilly Media, United States of America.
10. Semantic Web Activity: http://www.w3.org/2001/sw/ (Ανάκτηση 25 Αυγούστου 2011)
11. Semantic Web Applications: http://www.readwriteweb.com/archives/10_semantic_apps_to_watch.php (Ανάκτηση 30 Αυγούστου 2011)
12. Semantic Web Standards: http://www.w3.org/standards/semanticweb/ (Ανάκτηση 30 Αυγούστου 2011)
13. Semantic Web Wiki: http://semanticweb.org/wiki/Main_Page (Ανάκτηση 30 Αυγούστου 2011)
14. SPARQL query language: http://www.w3.org/TR/rdf-sparql-query/ (Ανάκτηση 25 Αυγούστου 2011)
15. SPARQL Update(1.1): http://www.w3.org/Submission/SPARQL-Update/ (Ανάκτηση 30 Αυγούστου 2011)

16. SPARQL: http://www.w3.org/2009/sparql/wiki/Main_Page (Ανάκτηση 25 Αυγούστου 2011)
17. The Semantic Web, An Introduction: http://infomesh.net/2001/swintro/ (Ανάκτηση 30 Αυγούστου 2011)
18. Triplestores: http://www.ontoba.com/services/triple-store-integration (Ανάκτηση 15 Σεπτεμβρίου 2011)
19. Tsoukalas Chrysostomos, Dervos Dimitrios, Martinez-Gil Jorge, Aldana-Montes Jose F. , *TheMa: An API for Mining Linked Datasets*, working paper.
20. URI: http://labs.apache.org/webarch/uri/rfc/rfc3986.html (Ανάκτηση 15 Σεπτεμβρίου 2011)
21. W3 Design Linked Data: http://www.w3.org/DesignIssues/LinkedData.html (Ανάκτηση 5 Σεπτεμβρίου 2011)
22. W3 OWL Guide: http://www.w3.org/TR/owl-guide/ (Ανάκτηση 28 Αυγούστου 2011)
23. W3 schools OWL: http://www.w3schools.com/rdf/rdf_owl.asp (Ανάκτηση 28 Αυγούστου 2011)
24. W3 Semantic Web Standards: http://www.w3.org/standards/semanticweb/data (Ανάκτηση 5 Σεπτεμβρίου 2011)
25. W3 SPARQL Query Language for RDF: http://www.w3.org/TR/rdf-sparql-query/ (Ανάκτηση 28 Αυγούστου 2011)
26. W3C Semantic Web Inference: http://www.w3.org/standards/semanticweb/inference (Ανάκτηση 15 Σεπτεμβρίου 2011)
27. Watson Mark (2010), *Practical Semantic Web and Linked Data Applications*, reative Commons Attribution, United States of America.
28. Wikipedia SPARQL: http://en.wikipedia.org/wiki/SPARQL (Ανάκτηση 30 Αυγούστου 2011)
29. Wikipedia, Graph Databases: http://en.wikipedia.org/wiki/Graph_database (Ανάκτηση 15 Σεπτεμβρίου 2011)
30. Wikipedia, Linked Data: http://en.wikipedia.org/wiki/Linked_Data (Ανάκτηση 30 Αυγούστου 2011)

31. Wikipedia, RDF Schema: http://en.wikipedia.org/wiki/RDF_Schema (Ανάκτηση 28 Αυγούστου 2011)
32. Wikipedia, RDF: http://en.wikipedia.org/wiki/.rdf (Ανάκτηση 25 Αυγούστου 2011)
33. Wikipedia, Triplestores: http://en.wikipedia.org/wiki/Triplestore (Ανάκτηση 15 Σεπτεμβρίου 2011)
34. Wikipedia, URI: http://en.wikipedia.org/wiki/Uniform_Resource_Identifier (Ανάκτηση 28 Αυγούστου 2011)
35. XML.com What Is RDF: http://www.xml.com/pub/a/2001/01/24/rdf.html (Ανάκτηση 25 Αυγούστου 2011)

ΠΑΡΑΡΤΗΜΑ Ι

Jena

Το Jena είναι ένα open source framework το οποίο χρησιμοποιείται για την κατασκευή Semantic Web εφαρμογών. Περιλαμβάνει ένα API για RDF, ένα API για OWL, μπορεί να διαβάζει και να γράφει RDF σε RDF/XML, N3 και N-Triples, και επίσης μια μηχανή ερωτημάτων SPARQL.

XML

Συντομογραφία για το **Ex***tensible* **M***arkup* **L***anguage.* Μια προδιαγραφή που αναπτύχθηκε από το W3C. Η XML είναι μια συγκρίσιμη προς τα κάτω έκδοση του SGML, που σχεδιάστηκε αποκλειστικά για τα έγγραφα του διαδικτύου. Δίνει την δυνατότητα στους σχεδιαστές να δημιουργήσουν τις δικές τους ετικέτες, επιτρέποντας τον ορισμό, την μετάδοση, την επικύρωση, και την ερμηνεία των δεδομένων ανάμεσα στις εφαρμογές και ανάμεσα στους οργανώσεις.

ΠΑΡΑΡΤΗΜΑ ΙΙ

Κώδικας Αντικειμένου για επιστροφή αποτελεσμάτων από τις συναρτήσεις υπολογισμού κύρους ενός κόμβου:

```
public class PrestigeArray {

	private String[] objNames;
	private int[] objCounts;

	PrestigeArray(){};

	public PrestigeArray(int length){
		objNames = new String[length];
		objCounts = new int[length];
	}


	public PrestigeArray(int length,String[] arr){

		objNames = new String[length];
		objCounts = new int[length];

		objNames = arr;

		for(int i=0;i<objCounts.length;i++)
			objCounts[i]=0;
	}


	public int size(){

		return objNames.length;
	}

	public String getName(int place){

		return objNames[place];
	}

	public int getCount(int place){

		return objCounts[place];
	}

	protected void incValue(int place){

		++objCounts[place];
	}

	public String toString(){

		String s = "\nObject Name ------------------ Value\n";
		for(int i=0;i<objNames.length;i++){
			s+= getName(i) + " ----- " + getCount(i) + "\n";
```

```
			}

			return s;
		}


}
```

Υπολογισμός εισερχόμενου κύρους:

```
//Calculates the in prestige of the nodes in a rdf file
//In prestige is the links that point to this node
//Start function inputPrestigeMap() the same as calcInPrestige()
//but returns a standard form(Map) that you can handle it easier
//Start function inputPrestigeMap()
public Map<String,Integer> inputAllPrestigeMap(){

	//Create the Map that is going to be returned
	Map<String,Integer> myMap = new HashMap<String,Integer>();

	//Call the function calcInPrestige()
	PrestigeArray myObj = this.calcAllInPrestige();

	//Puts the results of the PrestigeArray object to the map
	for(int i=0;i<myObj.size();i++){
		myMap.put(myObj.getName(i), myObj.getCount(i));
	}

	//Sorting Map
	myMap = Utilities.sortMapByValues(myMap);

	//Returns the Map
	return myMap;
}
//End of function inputPrestigeMap()



//Calculates the in prestige of the nodes in a rdf file
//In prestige is the links that point to this node
//Start function calcInPrestige()
public PrestigeArray calcAllInPrestige(){

	//Construct the object that is going to be returned
	PrestigeArray myObj = new PrestigeArray(getNumObj(),getListObj());

	//Getting all the statements from the model
	 StmtIterator reI = myModel.listStatements();

	 //Start the loop to check each statement
	 while (reI.hasNext()) {

		//Getting the object of the statement
		RDFNode n = reI.next().getObject();

			//Test Code
			//System.out.println("Object Name:");
			//System.out.println(n);
```

```
	//Finds this objects inside the object that we have created and
	//increases its value
	for(int i=0;i<myObj.size();i++){
		if(n.toString().equals(myObj.getName(i))){
				myObj.incValue(i);
			}
		}
	}
	//End of the loop

	//Return the object with the values of the incoming links
	return myObj;
	}
	//End of function calcInPrestige()
```

Υπολογισμός εξερχόμενου κύρους:

```
//Calculates the out prestige of the nodes in a rdf file
//Out prestige is the links that come out of the node
//Start function outputPrestigeMap() the same as calcOutPrestige()
//but returns a standard form(Map) that you can handle it easier
//Start function outputPrestigeMap()
public Map<String,Integer> outputAllPrestigeMap(){

	//Create the Map that is going to be returned
	Map<String,Integer> myMap = new HashMap<String,Integer>();

	//Call the function calcOutPrestige()
	PrestigeArray myObj = this.calcAllOutPrestige();

	//Puts the results of the PrestigeArray object to the map
	for(int i=0;i<myObj.size();i++){
		myMap.put(myObj.getName(i), myObj.getCount(i));
	}

	//Sorting Map
	myMap = Utilities.sortMapByValues(myMap);

	//Returns the Map
	return myMap;
}
//End of function outputPrestigeMap()


//Calculates the out prestige of the nodes in a rdf file
//Out prestige is the links that come out of the node
//Start function calcOutPrestige()
public PrestigeArray calcAllOutPrestige(){

		//Construct the object that is going to be returned
	PrestigeArray mySbj = new PrestigeArray(getNumSbj(),getListSbj());

	//Getting all the statements from the model
	StmtIterator reI = myModel.listStatements();
```

```
    //Start the loop to check each statement
    while (reI.hasNext()) {

      //Getting the object of the statement
      RDFNode n = reI.next().getSubject();

      //Test Code
      //System.out.println("Object Name:");
      //System.out.println(n);

      //Finds this objects inside the object that we have created and
      //increases its value
      for(int i=0;i<mySbj.size();i++){
            if(n.toString().equals(mySbj.getName(i))){
                  mySbj.incValue(i);
            }
      }
    }
     //End of the loop

     //Return the object with the values of the incoming links
return mySbj;
}
//End of function calcOutPrestige()
```

Υπολογισμός κατηγορημάτων «Γέφυρες» σε ένα Dataset:

```
//Calculate the bridges in a rdf file and return a model with this
statements
//Bridge is a predicate that if you delay it from a statement then you
can't
//reach from this subject the object of the statement in any way
//Start function getBridges()
public Model getAllBridges(){

//Variables
//Create 3 models to process the final model that it would be returned
      Model returnModel = ModelFactory.createDefaultModel();
      Model testModel = ModelFactory.createDefaultModel();
      Model emptyModel = ModelFactory.createDefaultModel();
      testModel = emptyModel.union(myModel);

      String[] objAr = this.getListObj();
      Vector<String> v = new Vector<String>();
      StmtIterator myRes = myModel.listStatements();

      //Adds all the objects in the vector for processing
      for(int i=0;i<objAr.length;i++){
            v.add(objAr[i]);
      }


      //Start the loop for processing the statements
      while(myRes.hasNext()){
```

```
            //Gets the statement and the Subject and the object
            Statement myState = myRes.nextStatement();
            Resource res = myState.getSubject();
            RDFNode obj = myState.getObject();

            //Starts loop and check every object in the vector
            for(int j=0;j<v.size();j++){

                  //Local var
                  boolean flag;

//Checks if the object in the vector is the same with the object in the
statement
                  if (obj.toString().equals(v.get(j))) {
//Removes the statement from the testModel and "ask" if you can reach
this object from this subject
                        flag =
            this.askConnection(testModel.remove(myState), res, obj);

                  //returns the test model to the previous situation
                        testModel = emptyModel.union(myModel);

                        //If you can't adds the statement to the return
model
                        if (!flag) {
                              returnModel = returnModel.add(returnModel

      .createStatement(myState.getSubject(),

      myState.getPredicate(),

      myState.getObject()));
                              }
                        }
                  }
                  //End of the check object loop
            }
            //End statement loop

      testModel.close();
      emptyModel.close();

      //Return model
      return returnModel;
}
//End of function getBridges()
```

Υπολογισμός όλων των κύκλων σε ένα Dataset:

```
//Checks if a node represent a circle in the file.
//If you start from this node and follow the graph if you can
//reach again this node and returns a list with this paths one
//for each node as a list of Path objects from jena.
//Start of function getGraphCircles()
public List<OntTools.Path> getGraphCircles(){

      //Variables
      List<OntTools.Path> list = new ArrayList<OntTools.Path>();
      List<String> nodeList = this.getListSO();
```

```
        //Control int
        int j=0;

    //Start a loop and checks all the objects in the model
    for(int i=0;i<nodeList.size();i++){

      //We get all the subjects
      ResIterator myS = myModel.listSubjects();

      //Start a loop for all the subjects in the model
            while (myS.hasNext()) {

                  //Takes the next subject
                  Resource sub = myS.nextResource();

                  //Checks if the position of the list of the object
                  //is the same with the control int
                  if (i==j) {
                        //System.out.println(i);
                        //System.out.println(nodeList.get(i));

                        //Checks if the subject is the same with the
object in the list position
                        if (sub.toString().equals(nodeList.get(i))) {

                              //System.out.println(i);
                              //Takes all the objects from the model
                              NodeIterator myO = myModel.listObjects();

                              //Starts a loop for all the objects in the
model
                              while (myO.hasNext()) {

                                    //Takes the next object
                                    RDFNode obj = myO.nextNode();

                                    //Checks if is the same with the
object in the list
                              if (obj.toString().equals(nodeList.get(i)))
{

      //We call a function from Jena to check if there is a to reach
                                          //this node as object
                        OntTools.Path p = OntTools.findShortestPath(
                                          myModel, sub, obj, Filter.any);

                        //If there is we add this path to the list
                                    if (p != null) {

      //System.out.println(p);
                                          list.add(p);
                                    }
                              }
                        }
                        //End loop of the objects
                  }
            }
      }
      //End loop of the subjects
      j++;
```

```
     }

 //System.out.println(j);

      //Return the list
      return list;
}
//End of function getGraphCircles()
```

Υπολογισμός ενός κύκλου μέσα σε ένα Dataset για ένα συγκεκριμένο κόμβο:

```
//Checks if a string(node) represent a circle in the file.
//If you start from this node and follow the graph if you can
//reach again this node and returns this path as a Path object from jena.
//Start of function returnPathCircle(String node)
public OntTools.Path returnPathCircle(String node){

            //Variables
            OntTools.Path myPath = null;
            ResIterator myS = myModel.listSubjects();
        NodeIterator myO = myModel.listObjects();

        //Start the loop and examine all the subjects in the model
        while(myS.hasNext()){

            //Takes the subject
            Resource sub = myS.nextResource();

            //Checks if this subject is the same with the node that we
have enter
            if(sub.toString().equals(node)){

                  //Start the loop and examine all the objects in the
model
                  while(myO.hasNext()){

                        //Takes the object
                        RDFNode obj = myO.nextNode();

                  //Checks if this object is the same with the node that
we have enter
                        if(obj.toString().equals(node)){

                              //Call a methode from Jena API to find if
there is a circle with this node
                              OntTools.Path p =
OntTools.findShortestPath(myModel,sub,obj,Filter.any);

                              //If there is a path we keep it
                              if(p!=null){
                                    //System.out.println(p);
                                    myPath = p;
                              }
                        }
                  }
                  //End of object loop
            }
        }
```

```
        //End of subject loop

            //return the result
            return myPath;
        }
        //End of function returnPathCircle(String node)
```

Υπολογισμός του Closure για ένα κόμβο:

```
public Model closure(String node){

        System.out.println(node);
        Model returnModel = ModelFactory.createDefaultModel();
        //Resource resource = ResourceFactory.createResource(node);
        List <String> myList = new ArrayList<String>();

        returnModel.add(closure1(node,myList));


        return returnModel;
}
```

Υπολογισμός του ποσοστού «Bridges/Statements» σε ένα Dataset:

```
//Computes the persentage of the of the statements that they are bridges.
//With this metric you can estimate if the rdf file is strong connected
//because if you have a high persantage in statements that they are
bridges
//your file is not strong connected but you can split it easy.
//This method is very close to the classic graph's method cohession.
//Start of function bridgesStatementsPers()
public float bridgesAllStatementsPerc(){

        //Variables
        Model bridgeModel = this.getAllBridges();
        StmtIterator myRes = bridgeModel.listStatements();
        StmtIterator myRes2 = myModel.listStatements();
        int bridges = 0;
        int statements = 0;

        //Geting the size the model wich contains the bridges of the model
          bridges = myRes.toList().size();

          System.out.println("Bridges: " + bridges);

          //Geting the size the model
          statements = myRes2.toList().size();

          System.out.println("Statements: " + statements);

          //Returns the result
          if(statements!=0)
              return (float)bridges/statements;
          else
```

```
		return 0;
	}
	//End of function bridgeStatementsPers
```

Υπολογισμός του mean reachability για ένα Dataset:

```
//Compute the mean rechability for all the subjects in the model
//Start function meanReachability()
public float meanReachability(){

	//Variables
	int subs=0,sum = 0;
	ResIterator myRes = myModel.listSubjects();

	//Computes the reachability for each node and add them together
		while(myRes.hasNext()){

			Resource sub = myRes.nextResource();
			sum += this.reachability(sub.toString());
			subs++;
		}

		//Computes and returns the result
		return (float)sum/subs;
	}
	//End of function meanReachability
```