Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Τίτλος: survey για web intelligence

Παναγιώτης Ιωαννίδης ΑΜ(2219)

Επιβλέπων καθηγητής : Παπαδάκης Νίκος

Ημερομηνία Παρουσίασης : 3/4/2015

Πίνακας περιεχομένων

ABSTRACT

The World Wide Web is nowadays the most famous and widespread information system. Its success is witnessed by its enormous size and rate of growth. There is the need for further layers of semantics, properly enriching the data that now overflow the classic Web: ontologies, rules, logic, proofs, trust are all ingredients of this ambitious picture. Given these premises, it should not come as a surprise the fact that this evolution is bringing the Web closer and closer to another field, that since quite some time has been facing similar problems of logical organization of knowledge: logic programming. The burst of research in Semantic Web developments has eventually started to touch, connect and reinterprete many topics that were and are mainstream of the logic programming area. We feel this is a necessary progression, as the Semantic Web, and more generally the Web of the future, has a lot to learn from research in the logic programming area. And, conversely, in these new scenarios there are lot of new applied problems that can be challenging and rewarding from a logic programming perspective. This calls for a tighter interaction between the Web and logic programming, which was the reason to motivate this special issue as well: gathering together a selection of the best contributions that could showcase the potential of the cross-breeding.It begins with a paper called "N3Logic: A Logic for the Web". In this paper a new system is proposed to deal with logical rules in the Web. The special issue then proceeds with the paper "Translating OWL and Semantic Web Rules into Prolog: Moving Toward Description Logic Programs": here we bring to higher levels the translational approach of the Metalog project that also uses, and analyzes the problem of reasoning in the semantic web as a domain translation, using Prolog as the back-engine, therefore interestingly exploring how much of the Semantic Web can be reconciled within a pure logic programming framework. A translational approach is also taken by "Querying XML Documents in Logic Programming", although the spirit here is different. This, shows us that logic programming as a back-engine can be useful not only for the Semantic Web, but even for the world wide Web. The fundamental importance of logic programming in the translational picture depicted by the above papers would be only theoretical, without a Prolog system to actually rely upon. In the following paper "SWI-Prolog and the Web", describing one of the most successful systems ever designed in this context, namely SWI-Prolog, and show us how Prolog itself can gracefully evolve to a web programming language, by appropriate extensions that link its inference power to the external Web environment. It ends with two contributions that have been selected as the best ones of the ALPSWS (Applications of Logic Programming in the Semantic Web and Semantic Web Services) International Workshop. In "Query Evaluation and Optimization in the Semantic Web", we study how efficiently Web ontologies can be queried, and introduce the new concept of a Deductive Ontology Base, showing how part of the Semantic Web can be effectively implemented this way. Later, in "Guarded Hybrid Knowledge Bases", we introduce a new reasoning paradigm that nicely cross-breeds between description logics and logic programm ing, therefore trying to reconcile these two so far different approaches for web reasoning.

ΠΕΡΙΛΗΨΗ

Εισαγωγή στα ειδικά θέματα του Λογικού Προγραμματισμού και του διαδικτύου.

Το World wide web (ή αλλιώς Παγκόσμιος Ιστός , που είναι το δίκτυο των συνδεδεμένων υπολογιστών και δικτύων σε παγκόσμια κλίμακα), είναι το πιο διαδεδομένο πληροφοριακό σύστημα. Παρά την αποδεδειγμένα ισχύουσα επιτυχία του, υπάρχει μία τόσο μεγάλη και δύσχρηστη μάζα πληροφοριών, την οποία καλείται να διαχειριστεί το Semantic Web, (ή αλλιώς Σημασιολογικός Ιστός). Η λογική πίσω από το Σημασιολογικό Ιστό, είναι ότι η δημοσιευμένη πληροφορία θα περιέχει μετα-δεδομένα, τα οποία θα είναι κοινά για όλους, θα μπορούν να «κατανοούνται» και από μηχανές, οι οποίες θα βοηθήσουν στην καλύτερη συλλογή και επεξεργασία τους.

Μέσω αυτής της εξέλιξης ένα άλλο πεδίο, ο Λογικός Προγραμματισμός, θα έρθει πιο κοντά με τον Παγκόσμιο Ιστό, πράγμα πολύ χρήσιμο, καθώς ο πρώτος έχει πολλά ωφέλη από μία τέτοια εξέλιξη. Επίσης θα υπάρξη πάρα πολύ μεγάλη αλληλεπίδραση μεταξύ των δύο αυτών εννοιών, στην πράξη και θα δώσει και στα δύο αυτά πεδία πολλές νέες δυνατότητες. Αυτό το ειδικό θέμα παρουσιάστηκε σε paper, με την ονομασία “N3Logic: A Logic For The Web” (N3Logic: Μία λογική για τον Ιστό) ,του Tim Berners-Lee. Σύμφωνα με αυτό προεκτείνεται το layer του Σημασιολογικού Ιστού με κάποιες βασικές λειτουργίες.

Επίσης παρουσιάζεται ένα εναλλακτικό «μονοπάτι», του οποίου η παρουσίαση ονομάζεται “Building Rules on top of Ontologies for the Semantic Web with Inductive Programming” (Χτίζοντας κανόνες στην κορυφή των Οντολογιών για τον Σημασιολογικό Ιστό με επαγωγικό Προγραμματισμό), της Francesca Lisi. Εκεί τοποθετούνται κανόνες μάθησης, στην κορυφή των οντολογιών και ο επαγωγικός προγραμματισμός, γίνεται ιδανικός και εμπλουτίζει τον Παγκόσμιο Ιστό.

Συνεχίζοντας, το ειδικό θέμα μεταβαίνει στο paper “Translating OWL and Semantic Web Rules into Prolog: Moving Toward Description Logic Programs” (Μεταφράζοντας την OWL και τους κανόνες του Σημασιολογικού Ιστού σε Prolog: Κινούμενοι προς λογικά προγράμματα περιγραφής). Ο Ken Samuel, αναλύει το πρόβλημα του reasoning στον Σημασιολογικό Ιστό και χρησιμοποιεί την Prolog σαν βάση για την εξυγείανση του.

Στο επόμενο paper, “Querying XML, Documents in Logic Programming” (Κάνοντας ουρές στην XML, Ντοκουμέντα σε Λογικό Προγραμματισμό) παρουσιάζεται κάτι διαφορετικό. Ο Jesus Almendros Jimenez, διατυπώνει πώς ο Σημασιολογικός Ιστός αλλά και η XML, μπορούν να έχουν σαν υπόβαθρο το Λογικό Προγραμματισμό.

Για να αποφευχθεί το γεγονός, όλα αυτά να είναι απλά θεωρητικά, παρουσιάζεται το επόμενο paper, “SWI-Prolog and the Web” (SWI-Prolog και Ιστός) , ο Jan Wielemaker, παρουσιάζει την πιθανή εξέλιξη της Prolog, σε γλώσσα Web Programming.

Τέλος, το ειδικό θέμα, κλείνει με τις δύο καλύτερες συνεισφορές της ALPSWS.
Η πρώτη παρουσίαση είναι ένα paper με θέμα “Query Evaluation and Optimization in the Semantic Web” (Αξιολόγηση Ουρών και βελτιστοποίηση στον Σημασιολογικό Ιστό). Η Edna Ruckhaus παρουσιάζει το σενάριο της Deductive Ontology Base και αναλύει το πώς ο σημασιολογικός Ιστός μπορεί να υποστηριχθεί με αυτόν τον τρόπο.
Τελειώνοντας, παρουσιάζεται το paper “Guarded Hybrid Knowledge Bases” (Φυλαγμένες βάσεις υβριδικών γνώσεων), ο Stijn Heymans, παρουσιάζει ένα παράδειγμα reasoning, που ενώνει την παραδειγματική λογική με το λογικό προγραμματισμό.

Κεφάλαιο 1ο

N3Logic : Ένα λογικό πλαίσιο για τον Παγκόσμιο Ιστό

Ο Σημασιολογικός Ιστός μας οδηγεί στη χρήση του Web (ή αλληλεπιδρά με λογικά αλληλοσυνδεόμενα δεδομένα. Μέσα από μοντέλα γνώσης, όπως το Πλαίσιο Περιγραφής Πόρων, ή αλλιώς RDF, ο Σημασιολογικός Ιστός παρέχει ενοποιητική αντιπροσώπευση και πλούσια δομημένα δεδομένα .

Στο Web, μέσω του λογικού πλαισίου N3Logic, προστίθεται λογική, η οποία πρέπει να είναι αρκετά ισχυρή για τις λειτουργίες που έχει να εκτελέσει και ανίσχυρη ταυτόχρονα. Ο Παγκόσμιος Ιστός από μόνος του μπορεί να οδηγήσει σε προβλήματα, λόγω του ανοικτού χαρακτήρα του. Γι'αυτό έχουμε το πλαίσιο N3Logic, όπου επιτρέπει τους κανόνες σε ένα Web περιβάλλον, μέσω επέκτασης του RDF, για να χρησιμοποιείται η ίδια γλώσσα για τη λογική και τα δεδομένα.

Το RDF βασίζεται στο θεμελιώδη μηχανισμό δείκτών του Web , το Uniform Resource Identifier (URI). Στην αρχική μορφή του Web , διευθύνσεις URL γενικά αναφέρονταν σε έγγραφα. Τα μέρη των εγγράφων «δείχνονταν» μέσω αγκυρών υπερκειμένου .

Στην αφηρημένη αναπαράσταση του R του Carroll και Klyne, ένα υποσύνολο των κόμβων εναι URL's και αυτά ονομάζονται κόμβοι. Οι ιδιότητες και σχέσεις είναι οι ακμές του γράφου. Η κάθε ακμή αντιπροσωπεύει το δικό της σήμα και ίδιου τύπου σήματα χρησιμοποιούνται όταν θέλουμε να αναφερθούμε σε άλλους κόμβους στο ίδιο γράφημα, ετσι ίδιοι τύποι ιδιοτήτων εκπροσωπούνται σε ένα R του γραφήματος.
Το πλαίσιο N3Logic, χρησιμοποιεί σύνταξη N3 και περιέχει RDF λεξιλόγιο όρων.

Ο στόχος του είναι να παρέχει ένα κοινό μοντέλο δεδομένων και μια κοινή σύνταξη, έτσι ώστε οι επεκτάσεις της γλώσσας να γίνονται απλά με τον καθορισμό νέων όρων σε μια οντολογία. (Αυτό γίνεται ήδη από το 1975 - Steele και Sussman). Μειονέκτημα του λεξιλογίου του N3Logic, είναι ότι στερούνται τη χρήση της καθολικής αναγνώρισης ως σύμβολα. Παρόλα αυτά μέχω της λογικής και των κανόνων, ενσωματώνονται ομαλά με τα RDF. Έχει επίσης σαν στόχο να κάνει μια ελάχιστη επέκταση του RDF μοντέλου δεδομένων, ώστε με την ίδια γλώσσα να χρησιμοποιείται η λογική και τα δεδομένα, σύμφωνα πάντα με την αρχιτεκτονική του Web.

Παρακάτω θα συζητηθούν τα χαρακτηριστικά της N3Logic μέσω παραδειγμάτων που περιγράφουν την άτυπη σημασιολογία του.

1. ΕΠΙΣΚΟΠΗΣΗ ΤΟΥ ΛΟΓΙΚΟΥ ΠΛΑΙΣΙΟΥ N3Logic.

Βασικό κίνητρο της είναι να χρησιμεύει σαν εργαλείο στο ανοιχτό περιβάλλον του Web. Λόγω της δυσκολίας της επέκτασής της στο ανοιχτό περιβάλλον αυτό, μπορούμε να κατασκευάσουμε ένα κλειστό σύστημα με βάση τη λογική αυτή. Μια γλώσσα που θα χρησιμοποιηθεί δυνητικά σε αυτό το περιβάλλον, απαιτεί τυπικά την ικανότητα να εκφράζει ποιο αρχείο ή μήνυμα ανήκει που. Εν τέλει το quoting παρέχει μια ρεαλιστική λύση γιααυτό. Για την αποφυγή προβλημάτων, το N3Logic, δεν περιέχει γενική άρνηση πρώτης τάξης, έχοντας περιορισμένη εκφραστική δύναμη.

Ένας ακόμη στόχος της είναι ότι οι πληροφορίες δεν περιορίζονται σε κανόνες και πρέπει να έχουν τη δυνατότητα διαμοιρασμού με τον ίδιο τρόπο που διαμοιράζεται το RDF. Έτσι η ίδια γνώση να μπορεί να επαναχρησιμοποιηθεί από διαφορετικά άτομα για διαφορετικούς σκοπούς. Τέλος υπάρχει η δυνατότητα έκφρασης νέων γνώσεων, χωρίς την ύπαρξη ασυνεπειών με τις παλιότερες πληροφορίες. Τα υπάρχοντα συστήματα χαρακτηρίζονται από μη-μονοτονία, που προκύπτει από μια μορφή της άρνησης ως αποτυχία (NAF), αν μια πρόταση θεωρηθεί ψευδής όταν δεν πραγματοποιείται από την τρέχουσα βάση γνώσεων. Η άρνηση, ονομάζεται Scoped άρνηση ως αποτυχία (SNAF) και είναι η ικανότητα για ένα συγκεκριμένο δεδομένο έγγραφο (ή , κατ 'ουσίαν , κάποιο αφηρημένο τύπο) να καθορίσει αντικειμενικά αν κατέχει ή όχι , ή επιτρέπει σε κάποιον να αντλήσει , ένα συγκεκριμένο γεγονός.)

2. ΠΑΡΑΔΕΙΓΜΑ ΠΑΡΟΧΗΣ ΚΙΝΗΤΡΩΝ:

Παρακάτω περιγράφεται ένα σενάριο που βασίζεται στο Web , που θα χρησιμοποιηθεί για να απεικονίσει τα διάφορα χαρακτηριστικά του N3logic . Θεωρούμε ένα σύστημα διαχείρισης συνεδρίων που χειρίζεται διάφορες πτυχές της καταχώρισης για συνέδρια .
Θα επιτρέπει στους ανθρώπους να κάνουν εγγραφή, ορίζοντας ονόματα , τις διευθύνσεις , τους συνεταιρισμούς , και μια φίλο-προς-φίλο (FOAF) σελίδα. Μια σελίδα FOAF περιλαμβάνει πληροφορίες όπως η οργάνωση , ο καταχωρίζων ο οποίος εργάζεται εκεί, το / τα τρέχοντα και προηγούμενα έργα του , και τα συμφέροντά του. (Brickley και Miller 1999. Dumbill 2002). Χρησιμοποιώντας αυτές τις πληροφορίες , το σύστημα διαχείρισης συνεδρίων πηγαίνει στο Web και ανακτά τις σχετικές πληροφορίες .
Με τη συλλογιστική πάνω σε αυτές τις πληροφορίες είναι σε θέση να βγάλει συμπεράσματα για τον καταχωρίζοντα , όπως εάν ο καταχωρίζων είναι χορτοφάγος ή όχι ,για ποιο εργαστήρια αυτός / αυτή θα πρέπει να ενδιαφέρονται περισσότερο , αν αυτός / αυτή είναι ένα μέλος μιας συγκεκριμένης επαγγελματικής οργάνωσης , και εάν ο καταχωρίζων είναι φοιτητής . Αυτό επιτρέπει στο σύστημα διαχείρισης συνεδρίων να παράσχει μεγαλύτερη στήριξη στην διαδικασία εγγραφής, υπολογίζοντας αν τα τέλη εγγραφής είναι εφαρμοστέα , είτε για παράδειγμα να διατάξει χορτοφαγικά γεύματα , κτλ.

3. ΕΝΝΟΙΕΣ ΚΑΙ ΟΡΟΛΟΓΙΑ:

Το N3 βασίζεται στην αφηρημένη σύνταξη του RDF . Η συγκεκριμένη σύνταξη του N3, περιλαμβάνει έναν αριθμό συντομογραφιών.

- Βασικές έννοιες του RDF:

Οι ατομικές προτάσεις στο RDF, στην αφηρημένη σύνταξη, ονομάζονται «τριάδες». Ακολουθεί παράδειγμα:

<http://dig.csail.mit.edu/2006/Papers/TPLP/example/exconf#ExConf>

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

<http://example.org/conf#Conference>.

- Το RDF έχει επίσης συνδέσμους . Οι URL όροι μπορούν να συντομογραφηθούν χρησιμοποιώντας nam espaces, και τη λέξη κλειδί "a" , για παράδειγμα:

©prefix conf: <http://example.org/conf#>.

©prefix : <http://dig.csail.mit.edu/2006/Papers/TPLP/example/

exconf#>.

ExConf a conf:Conference.

ExConf conf:homepage <http://www.13s.de/-olmedilla/events/ MTW06_Workshop.html>.

ExConf conf:registrant Judy. Judy a foaf:Person.

<http://dig.csail.mit.edu/2005/09/rein/examples/judy-foaf.rdf>

foaf:maker Judy.

- Το RDF, έχει επίσης κυριολεκτικούς όρους, όπως:

ExConf conf: eventName "\.J\.J\.12006 Workshop on Models of Trust for the Web".

ExConf conf:numOfRegistrants 65.

- Τέλος , η αφηρημένη σύνταξη RDF επιτρέπει την υπαρξιακή ποσοτικοποίηση . Ορισμένες συντάξεις (πχ., " [" .. «J ») επιτρέπουν μια υπαρξιακή μεταβλητή να εισαχθεί χωρίς να χρειάζεται να το αναφέρουμε : αυτό είναι γνωστό ως ένας κενός κόμβος .

©forSome X. j: Joe foaf: knows X. X foaf :name "Fred". j :Joe foaf :knows [foaf :name

"Fred"].

4. <u>Ν3 ΕΠΕΚΤΑΣΗ ΣΕ RDF.</u>

Το Ν3 επεκτείνει την αφηρημένη σύνταξη του R του με δύο τρόπους :

• Διαθέτει όλους τους όρους του R συν εισηγμένες τύπους . Για παράδειγμα,

b : mary says { j : Joe foaf : schoolHomepage < http://example.edu > } .

• Έχει το σύνολο των τύπων των ROF συν καθολικά ποσοτικοποιημένους τύπους . Σε απλές περιπτώσεις , το ποσοτικό @forAll μπορεί να παραμείνει σιωπηρό . Τα παρακάτω είναι ισοδύναμα:

©forAll X. {X a Man} log:implies {X a Mortal}.

{?X a Man} log:implies {?X a Mortal}.

5. ΛΕΞΙΛΟΓΙΟ N3Logic.

Η N3Logic χρησιμοποιεί τη σύνταξη N3 και περιλαμβάνει ένα σύνολο κατηγορημάτων . Το λεξιλόγιό του αποτελεί ένωση αυτής με τη σύνταξη N3 και το σύνολο των αναφορών UR I που ορίζεται στο ημερολόγιο : (http : // www w3 .org / 2000 /10 / ανταλλαγής / log #).

Παρακάτω παρουσιάζεται ένα σύνολο κατηγορημάτων της N3logic:

Table 1. *Some N3Logic predicates.*

log:conclusion, log:content, log:includes, log:semantics, log:notIncludes, log:supports ... crypto:md5, crypto:sign, crypto:verify ... list:in, list:last ... math:lessThan, math:greaterThan ... os:argv, os:environ ... string:contains, string:endsWith, string:scrape ... time:day, time:hour, time:minute ...

Εικόνα 1-predicates

6. ΑΤΥΠΗ ΣΗΜΑΣΙΟΛΟΓΙΑ ΤΟΥ ΛΟΓΙΚΟΥ ΠΛΑΙΣΙΟΥ N3Logic.

Διάφορα λεξιλόγια , κυρίως RDFS και OWL , έχουν οριστεί κατηγορήματα RDF με λογική σημασιολογία . όπως rdfs .range , OWL:sameAs , κ.λπ. Από αυτά ,η N3 Λογική χρησιμοποιεί το RDF : range και OWL.sameAs , και καθορίζει περαιτέρω κατηγορήματα για να επιτρέψει την ύπαρξη κανόνων , την πρόσβαση στο Web , και ενσωματωμένο υπολογισμό λειτουργιών .

Η N3Logic εκτείνεται σε RDF με δύο τρόπους : (i) σύνταξη N3 και (2) ένα λεξιλόγιο των νέων κατηγορημάτων , τα οποία μπορούν να χρησιμοποιηθούν για να μιλήσουμε σχετικά με την προέλευση των πληροφοριών και του περιεχομένου των εγγράφων στο Διαδίκτυο , και να προσφερθεί μια ποικιλία από χρήσιμες λειτουργίες , όπως η χορδή κρυπτο/φησης, και μαθηματικές λειτουργίες .

7. ΣΧΕΣΗ ΤΗΣ RDF, RDFS ΚΑΙ OWL.

Table 2. *N3 formula examples.*

N3 formula : c1 is a Car as defined in the "ex" namespace and its color is green.

```
@keywords a.
@prefix ex:  <http://example.org/car.n3#>.
c1 a ex:Car;
      ex:color  "green".
```

N3 formula : The semantics of <http://www.example.org/myfoaf.rdf> as expressed in N3 is stored into a variable, F.

```
@forSome F.
<http://www.example.org/myfoaf.rdf> log:semantics F.
```

Quoted N3 formula : Joe believes that Peter is a graduate student.

```
j:Joe believes { mit:Peter a school:GraduateStudent }.
```

Quoted N3 Formula : Mary believes that Joe believes that Peter is a graduate student.

```
Mary believes { j:Joe believes { mit:Peter a school:GraduateStudent } }.
```

— Existence of lists

```
(?X).
{?L rdf:rest [ ] } => { [ ] rdf:first ?X; rdf:rest ?L}.
```

— Uniqueness of lists

```
{?L1 rdf:first ?X; rdf:rest ?R.
  ?L1 rdf:first ?X; rdf:rest ?R.} => {?L1 = ?L2}.
```

— Uniqueness of the rdf:first and rdf:rest of a list

```
{?S rdf:first ?O1, ?O2} => {?O1 = ?O2}.
{?S rdf:rest ?O1, ?O2} => {?O1 = ?O2}.
```

- The semantics of RDFS can be easily expressed in N3Logic. A set of N3Logic rules for defining *rdf:domain* and *rdf:range* are as follows

```
{?S [s:domain ?C] ?O} => {?S a ?C}.
{?S [s:range ?C] ?O} => {?O a ?C}.
{?S a [s:subClassOf ?C]} => {?S a ?C}.
```

Εικόνα 2-RDF/RDFS/OWL

8. ΕΥΡΕΣΗ ΤΙΜΩΝ URL's.

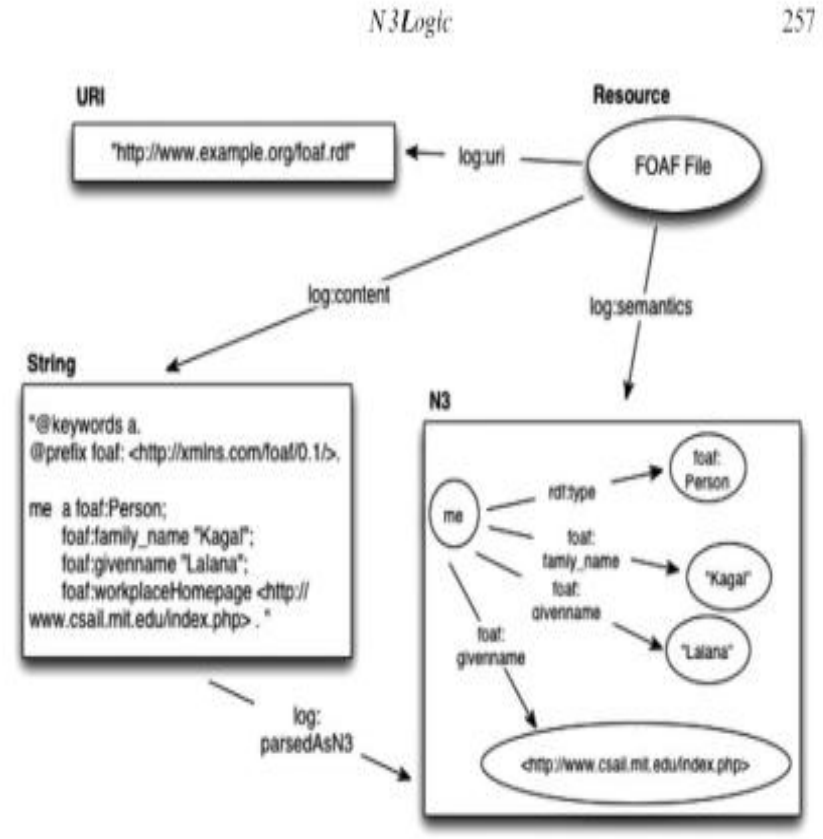


Fig. 1. Relationship between resource, URI, string, N3 representation, and logical properties.

Εικόνα 3-εύρεση τιμών

9. ΚΑΝΟΝΕΣ

Η N3 επεκτείνει την RDF με τις μεταβλητές και με ένθετα γραφήματα για να ενεργοποιήσουν τις περιγραφές των κανόνων.
Ωστόσο , είναι δυνατόν να περιορίσουμε την N3Logic σε καθοριζόμενη γλώσσα υποσυνόλου στην οποία κενοί κόμβοι δεν επιτρέπονται στο συμπέρασμα .

10. ΓΡΑΦΙΚΕΣ ΣΥΝΑΡΤΗΣΕΙΣ

Η λογική λειτουργία log.includes, ελέγει αν ενας τύπος μπορεί να είναι προερχόμενος από N3 ή από κάποια άλλη μέθοδο και ελέγχει αντικειμενικά το περιεχόμενο των αρχείων χωρίς να τα φορτώνει και να «δέχεται» όσα λένε.

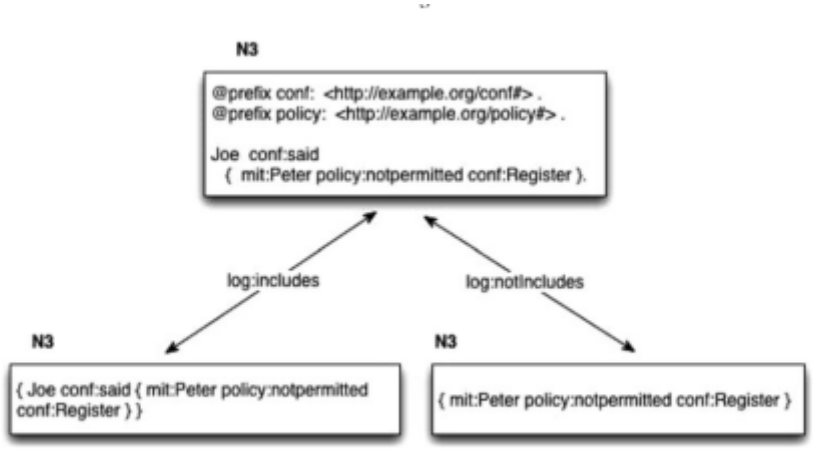


Fig. 2. Quoting and its relation to log:includes and log:notIncludes.

Εικόνα 4-graph functions

11. BUILT-Ins

Η N3Logic παρέχει επίσης και άλλα plug-ins για πρόσθετη λειτουργικότητα . Μερικά παραδείγματα περιλαμβάνουν :

• κρυπτο λειτουργίες -MD5 , δείκτες και την επαλήθευση.
• μαθηματικές λειτουργίες - cos , greaterThan, διαφορά κτλ.
• Λειτουργίες os για την ανάκτηση πληροφοριών περιβάλλοντος - vargv , baseAbsolute κα.
• συναρτήσεις συμβολοσειράς
• Ο συναρτήσεις χρόνου πχ - dayOfWeek . time GM, και local'Time.

12. ΠΑΡΑΓΩΓΗ N3Logic.

Εκτείνεται στο να εντάσει στο κείμενο απλά συμπεράσματα που είναι στάνταρ για τις περισσότερες λογικές.
Αυτό γίνεται με τους παρακάτω τρόπους:

- Conjunction Elimination - Εξάλειψη σύζευξης
- Conjunction Introduction - Σύζευξη εισαγωγής
- Universal elimination - Ολική εξάλειψη
- Existential Introduction - Υπαρξιακή Εισαγωγή
- Variable renaming - Μετονομασία μεταβλητών

13. ΣΥΜΜΕΤΡΙΑ ΤΡΙΑΔΩΝ

Κατά το σχεδιασμό μιας οντολογίας σε RDF η κατεύθυνση που επιλέγεται για ένα συγκεκριμένο ακίνητο είναι αυθαίρετη , μπορεί να καθορίζεται είτε ως "μητρική" ή " παιδί " , " υπάλληλος .. ή ..employer " . Η φιλοσοφία είναι ότι κανείς δεν πρέπει να ευνοεί έναν τρόπο πάνω από ένα άλλο . Η επιλογή του σχεδιασμού σε N3 είναι να επιτρέψει τις εμπρός και προς τα πίσω συνδέσεις και αυτό πρέπει να εκφράζεται με την ίδια ευκολία . Αυτό επιτυγχάνεται με την παροχή λέξεων-κλειδιά όπως: " είναι " και " του " που επιτρέπουν σε κάποιον να αντιστρέψει την κατεύθυνση της περιγραφής μιας τριάδας. Αυτό επιτρέπει επίσης τα γραφήματα με κενούς κόμβους χωρίς να χρειάζεται να έχουν δημιουργήθεί αναγνωριστικά κόμβων .

14. ΕΠΕΚΤΑΣΙΜΟΤΗΤΑ

Η επεκτασιμότητα του RDF είναι σκόπιμη , έτσι ώστε ένα έγγραφο να μπορεί να αντλήσει κατηγορήματα από πολλές πηγές ταυτόχρονα.

15. ΕΚΤΕΛΕΣΕΙΣ

Έχει αναπτύχθει (CWM Berners - Lee (2CDO)) , μια προς τα εμπρός αλυσίδα reasoner σε Python για N3 και N3Logic . Είναι ένας reasoner γενικής χρήσης για το Σημασιολογικό Ιστό που μπορεί να χρησιμοποιηθεί για την αναζήτηση , τον έλεγχο , τη μετατροπή , και το φιλτράρισμα των πληροφοριών .
Με αυτό, υπάρχει πρόσβαση σε πόρους του Ιστού και γίνεται φιλτράρισμα στα γραφήματα RDF μετά τη συγχώνευσή τους. Η N3 λογική είναι αρκετά εκφραστική ώστε να μπορεί να εκφραστεί σε αυτό. Το CWM είναι σε θέση να αιτιολογήσει, χρησιμοποιώντας μια λογική πρώτης τάξης , αλλά χωρίς την κλασική άρνηση .

ΣΥΜΠΕΡΑΣΜΑ

Ο κύριος στόχος της N3 λογικής είναι να επεκταθεί το μοντέλο δεδομένων RD F , έτσι ώστε η ίδια γλώσσα μπορεί να χρησιμοποιηθεί για τη λογική και τα δεδομένα .
Η N3Logic χρησιμοποιεί τη σύνταξη N3 , η οποία παρέχει τα quotations, τις μεταβλητές , και το χειριστή επιπτώσεων. Η N3 Λογική περιλαμβάνει επίσης ενσωματωμένες λειτουργίες που επιτρέπουν τους κανόνες για την πρόσβαση σε πόρους Web, καθώς και άλλες χρήσιμες λειτουργίες , όπως η μαθηματική , κρυπτογραφική κτλ.
Η χρήση των log.notIncludes, είναι για να επιτρέπεται η συλλογιστική προεπιλογής χωρίς αναιρέσιμες συμπεριφορές και επιτυγχάνει ένα στόχο σχεδιασμού κατανεμημένων συστημάτων κανόνα . Η λογική γλώσσα N3 έχει βρεθεί ότι έχει κάποιες χρήσιμες πρακτικές ιδιότητες. Ο διαχωρισμός μεταξύ των επεκτάσεων N3 σε RDF και τις λογικές ιδιότητες επέτρεψε στην N3Logic να παραταθεί με άλλες ιδιότητες για να παρέχει λειτουργικότητα , όπως η έκφραση των διαφορών γραφημάτων και ενημερώσεων (Berners - Lee και ο Connolly 2004) .

Κεφάλαιο 2ο

Κανόνες Δημιουργίας στην κορυφή των οντολογιών του Σημασιολογικού Ιστού, με Επαγωγικό Λογικό Προγραμματισμό.

Κατά την τελευταία δεκαετία, έχει δοθεί μεγαλύτερη προσοχή σε οντολογίες και στο ρόλο τους στη Μηχανική Γνώση(Staab και Studer 2004). Στη φιλοσοφική έννοια, εμείς μπορεί να αναφερόμαστε σε μια οντολογία ως ένα συγκεκριμένο σύστημα κατηγοριών που αντιπροσωπεύουν ένα συγκεκριμένο όραμα για τον κόσμο. Ως εκ τούτου, το σύστημα αυτό δεν εξαρτάται από μια συγκεκριμένη γλώσσα: η οντολογία του Αριστοτέλη είναι πάντα το ίδιο, ανεξάρτητα από τη γλώσσα που χρησιμοποιείται για να το περιγράψουμε.
Από την άλλη πλευρά, στην πιο διαδεδομένη χρήση τους στην Τεχνητή Νοημοσύνη, μια οντολογία αναφέρεται σε ένα τεχνούργημα μηχανικής που αποτελείται από ένα ειδικό λεξιλόγιο για να περιγράψει μια πραγματικότητα, καθώς και μια σειρά από συγκεκριμένες υποθέσεις σχετικά με το επιδιωκόμενο νόημα των λέξεων του λεξιλογίου.

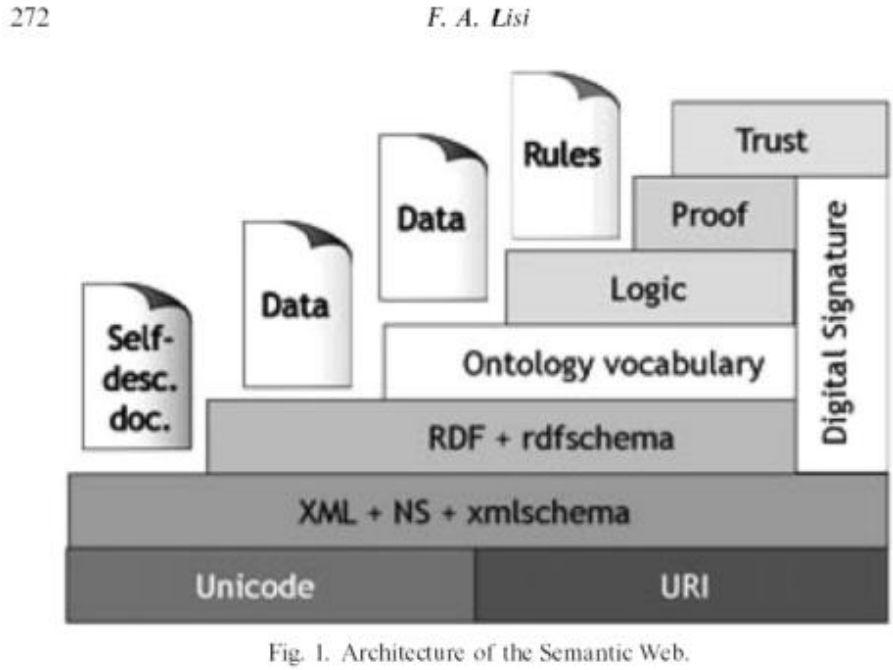


Fig. 1. Architecture of the Semantic Web.

Εικόνα 5-αρχιτεκτονική στο σημασιολογικό Ιστό

Στην απλούστερη περίπτωση, μια οντολογία περιγράφει μια ιεραρχία των εννοιών που σχετίζονται με σχέσεις υπαλληλίας. Σε πιο εξελιγμένες περιπτώσεις, τα κατάλληλα αξιώματα προστίθενται προκειμένου να εκφράσουν και άλλες σχέσεις μεταξύ των εννοιών και να περιορίσουν προορίζονται ερμηνεία τους.
Μια οντολογία είναι μια τυπική ρητή προδιαγραφή μίας κοινής σύλληψης, για έναν τομέα ενδιαφέροντος.

Η Μηχανική Οντολογιών, παίζει ρόλο στον ορισμό του σημασιολογικού Ιστού. Ο Σημασιολογικός Ιστός είναι ένα όραμα όπου το World Wide Web, εμπλουτίζεται με επεξεργασμένες μηχανικές πληροφορίες που υποστηρίζουν την εκτέλεση των καθηκόντων του χρήστη. Η αρχιτεκτονική του Σημασιολογικού Ιστού αποτελείται από πολλά στρώματα, καθένα από τα οποία είναι εξοπλισμένο με μία γλώσσα ad-hoc σήμανσης.

Υβριδικά συστήματα:

Τα υβριδικά συστήματα είναι KR&R συστήματα, που αποτελούνται από δύο ή περισσότερα υποσυστήματα που ασχολούνται με διαφορετικά τμήματα μιας ενιαίας βάσης γνώσεων εκτελώντας συγκεκριμένες διαδικασίες συλλογισμού (Frisch και Cohn 1991).
Ο λόγος που δημιουργήθηκαν τα υβριδικά συστήματα, είναι να βελτιωθούν σε δύο βασικά χαρακτηριστικά με KR&R φορμαλισμούς, δηλαδή παραστατική επάρκεια και επαγωγική δύναμη, διατηρώντας το άλλο κρίσιμο χαρακτηριστικό, δηλαδή αποφασιστικότητα.

Καθορισμός κανόνων:

Ο καθορισμός κανόνων θεωρείται ως ένα απαιτητικό έργο λόγω της Γνώσης Μηχανικών. Συχνά υποστηρίζεται από αλγόριθμους μηχανικής μάθησης που μπορεί να ποικίλλουν στις προσεγγίσεις. Η προσέγγιση που είναι γνωστή με το όνομα του Επαγωγικός Λογικός Προγραμματισμός (ILP) φαίνεται να είναι πολλά υποσχόμενη για την υπόθεση, λόγω των κοινών ριζών με τον Λογικό Προγραμματισμό (Flach και l.avrac 2002).
Εδώ θα δούμε πώς στρεφόμαστε στη μεθοδολογία της ILP για τον καθορισμό ενός γενικού πλαισίου για την εκμάθηση των κανόνων για τις οντολογίες του Σημασιολογικού Ιστού στο εσωτερικό KR&R πλαίσιο του Al-log. Το πλαίσιο που προτείνεται είναι γενικό.

1. Τα βασικά της Sf-Log.

Το σύστημα d Sf-log, ενσωματώνει δύο KR&R συστήματα, αυτά των δομών και των σχέσεων.

Table 1. *Syntax and semantics of* $\mathcal{ALC}$

bottom (resp. top) concept	$\bot$ (resp. $\top$)	$\emptyset$ (resp. $\Delta^{\mathcal{I}}$)
atomic concept	A	$A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$
role	R	$R^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$
individual	a	$a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$
concept negation	$\neg C$	$\Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
concept conjunction	$C \sqcap D$	$C^{\mathcal{I}} \cap D^{\mathcal{I}}$
concept disjunction	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
value restriction	$\forall R \cdot C$	$\{x \in \Delta^{\mathcal{I}} \mid \forall y\ (x;y) \in R^{\mathcal{I}} \rightarrow y \in C^{\mathcal{I}}\}$
existential restriction	$\exists R \cdot C$	$\{x \in \Delta^{\mathcal{I}} \mid \exists y\ (x;y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$
equivalence axiom	$C \equiv D$	$C^{\mathcal{I}} = D^{\mathcal{I}}$
subsumption axiom	$C \sqsubseteq D$	$C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
concept assertion	$a : C$	$a^{\mathcal{I}} \in C^{\mathcal{I}}$
role assertion	$\langle a;b \rangle : R$	$(a^{\mathcal{I}};b^{\mathcal{I}}) \in R^{\mathcal{I}}$

2.1 The structural subsystem

Εικόνα 6-Δομικό υποσύστημα

Το δομικό μέρος **Σ** βασίζεται στην ALB(Schmidr-Schauss και Smolka 199 I) και επιτρέπει την συγκεκριμενοποίηση της γνώσης από την άποψη των κατηγοριών (έννοιες), διμελείς σχέσεις μεταξύ των τάξεων (ρόλους), και περιπτώσεις (άτομα). Περίπλοκες έννοιες μπορούν να οριστούν από ατομικές έννοιες και τους ρόλους μέσω των κατασκευαστών. Επίσης, το **Σ** μπορεί να δηλώσει τόσο τις is-a σχέσεις μεταξύ των εννοιών (αξιώματα) και παραδείγματα-σχέσεις μεταξύ ιδιωτών και τις έννοιες .

Το κύριο καθήκον των Σ reasoners, είναι ο έλεγχος της συνέπειας. Η δοκιμή αυτή πραγματοποιείται με λογισμό-ταμπλώ που ξεκινά με το tableau branch S = Σ και προσθέτει ισχυρισμούς S μέσω των κανόνων διάδοσης, όπως:

- $S \rightarrow_{\sqcup} S \cup \{s : D\}$ if
  (1) $s : C_1 \sqcup C_2$ is in S,
  (2) $D = C_1$ and $D = C_2$,
  (3) neither $s : C_1$ nor $s : C_2$ is in S.
- $S \rightarrow_{\forall} S \cup \{t : C\}$ if
  (1) $s : \forall R \cdot C$ is in S,
  (2) sRt is in S,
  (3) $t : C$ is not in S.
- $S \rightarrow_{\sqsubseteq} S \cup \{s : C' \sqcup D\}$ if
  (1) $C \sqsubseteq D$ is in S,
  (2) s appears in S,
  (3) C' is the NNF concept equivalent to $\neg C$
  (4) $s : \neg C \sqcup D$ is not in S.

Εικόνα 7-κανόνες διάδοσης

- S --> {s: ⊥} if
  (1) s: A and s : ¬A are in S, or
  (2) s : ¬⊤ is in S,
  (3) s: ⊥ is not in S,

Εικόνα 8-κανόνες διάδοσης

Αυτό συμβαίνει μέχρι που δημιουργείται είτε μια αντίφαση ή μια ερμηνεία που ικανοποιεί το S, να μπορεί εύκολα να ληφθεί από αυτό.

2. <u>Το σχεσιακό υποσύστημα</u>

Το σχεσιακό μέρος της AL-log επιτρέπει να οριστούν τα προγράμματα DATALOG^3 εμπλουτισμένα με τους περιορισμούς της μορφής s: C, είτε ως σταθερά είτε ως μεταβλητή, Η χρήση των εννοιών και η πληκτρολόγηση των περιορισμών ισχύει μόνο για μεταβλητές και σταθερές που εμφανίζονται ήδη στη ρήτρα. Το σύμβολο & διαχωρίζει περιορισμούς από DATALO atoms σε μια ρήτρα.

Definition 1
A *constrained* DATALOG *clause* is an implication of the form

$$\square_0 \leftarrow \square_1, \ldots, \square_m \& \square_1, \ldots, \square_n,$$

where $m > 0$, $n > 0$, $\square$ are DATALOG atoms and $\square$ are constraints. A *constrained* DATALOG *program* Π is a set of constrained DATALOG clauses.

Εικόνα 9-Datalog

Αυτές οι ιδιότητες δηλώνουν μια ασφαλή αλληλεπίδραση μεταξύ του διαρθρωτικού και σχεσιακού μέρους μιας Al-log βάσης γνώσεων. Επιλύοντας έτσι τη σημασιολογική αναντιστοιχία μεταξύ του OWA του Alb και το CWA του DATALOG (Rosati 2005).

3. Το γενικό πλαίσιο για την εκμάθηση κανόνων του AL-log.

Στο πλαίσιο για την εκμάθηση του AL-log, αντιπροσωπεύουμε επαγωγικές υποθέσεις ως περιορισμένες ρήτρες DATALOG και τα δεδομένα ως AL βάση γνώσεων. Ειδικότερα, FM αποτελείται από ένα Y γνωστικό υπόβαθρο {και ένα σύνολο 0 παρατηρήσεων.
Υποθέτουμε Y {n 0 = 0.
Για να καθορίσει το πλαίσιο, καταφεύγουμε στο μεθοδολογικό συσκευή της I LP, η οποία απαιτεί τα ακόλουθα συστατικά για να επιλέξει:

a)Τη γλώσσα L των υποθέσεων
b)Μια γενικότητα σειράς >= για την L, ώστε να δομήσει το χώρο των υποθέσεων
c)Μια σχέση για να δοκιμάσει την κάλυψη των υποθέσεων σε L, έναντι se παρατηρήσεις 0 w.r.t. H .

Το πλαίσιο είναι γενικό, ανεξάρτητα από το πεδίο εφαρμογής της. Ως εκ τούτου , η κυριολεκτική q (X)^ 4 στην κεφαλή των υποθέσεων αποτελεί μια έννοια για να ξεχωρίζουν από τις άλλες, είτε για να χαρακτηρίζονται.

3.1 Η Γλώσσα των υποθέσεων.

Για να είναι κατάλληλη για γλώσσα υποθέσεων, οι Datalog ρήτρες, πρέπει να πληρούν κάποιους περιορισμούς.

- ➔ Πρώτον, να επιβάλλονται περιορισμοί DATALOG, για να συνδέονται ή να συνδεθούν ως συνήθως, σε ILP (Nienhuys - Cheng και De Wolf 1997) .

Ορισμός 3:
Έστω H μία περιορισμένη DATALOG ρήτρα. Ένας όρος t, σε κάποια κυριολεκτική li που ανήκει στο H, συνδέεται με τη σύνδεση - αλυσίδα μήκους 0. αν t συμβαίνει στο κεφάλι (head)- (H) , και συνδέεται με τη σύνδεση - αλυσίδα μήκους d + I , αν υπάρχει κάποιος άλλος όρος στο Ii, συνδέεται με τη σύνδεση - αλυσίδα μήκους d.
Η ίδια ρήτρα H συνδέεται , εάν κάθε li που ανήκει στο H, συνδέεται. Η ρήτρα H συνδέεται. Αν κάθε μεταβλητή συμβαίνει στο κεφάλι (H) εμφανίζεται επίσης στο σώμα (H).

- ➔ Δεύτερον, έχουμε επιβάλει εξαναγκασμένες ρήτρες DATALOG, να είναι συμβατές με το Object Identity. (Semeraro et al., 1998).

3.2 Η σχέση γενικότητας.

Στο iLP , ο βασικός μηχανισμός είναι γενίκευση που δημιουργειται ως μια διαδικασία αναζήτησης μέσω του μερικώς διατεταγμένου χώρου των υποθέσεων (Mitchell I 982) .
Ο ορισμός μιας σχέσης γενικότητας για τις ρήτρες περιορισμένου DATALOG, δεν μπορεί να αγνοήσει ούτε τις ιδιαιτερότητες του AL log, ούτε τη μεθοδολογική συσκευή της iLP.
Ως εκ τούτου , βασιζόμαστε στους μηχανισμούς συλλογιστικής που διατίθενται.

3.3 Σχέσεις κάλυψης

Για τον καθορισμό των σχέσεων κάλυψης, κάνουμε υποθέσεις όσον αφορά την αντιπροσώπευση των παρατηρήσεων.Αυτό γιατί έχει επιπτώσεις στον ορισμό της κάλυψης.
Έχουμε δύο επιλογές.
Μπορούμε να αναπαραστήσουμε μια παρατήρηση είτε ως βασική, σαφή ρήτρα ή σαν ένα σύνολο ρητρών, σαν μονάδα.

4. Εμφανίσεις του πλαισίου για τη Βελτίωση της Οντολογίας.

Στοχεύει στην προσαρμογή της υφιστάμενης οντολογίας σε ένα συγκεκριμένο τομέα ή για τις ανάγκες ενός συγκεκριμένου χρήστη.

Building Rules on Top of Ontologies for the Semantic Web with ILP 283

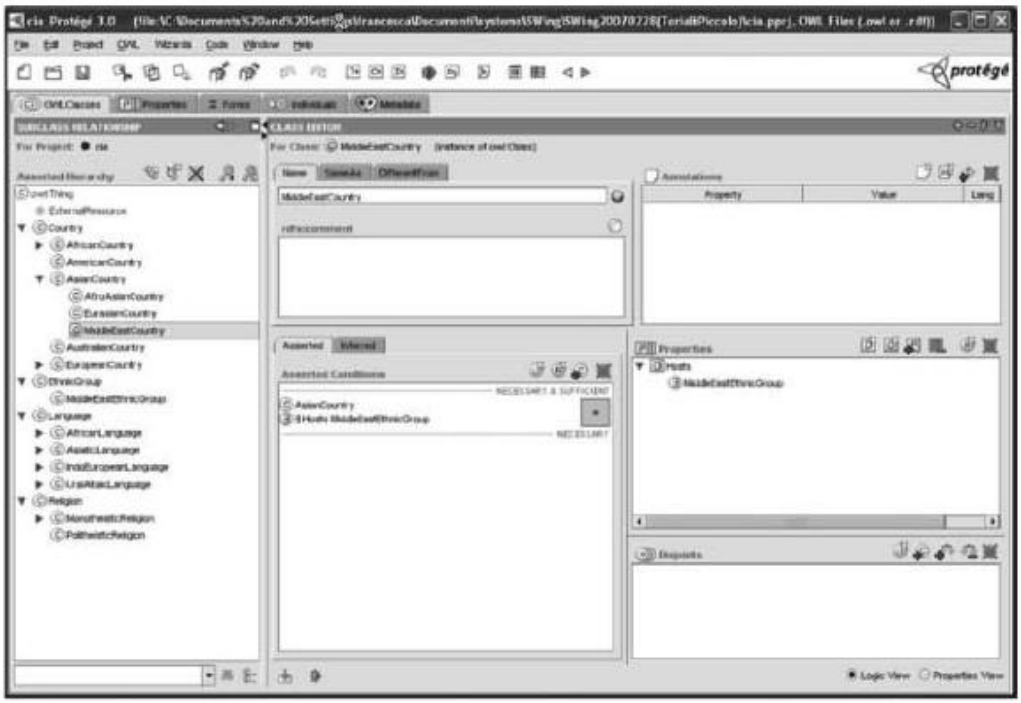


Fig. 2. The ontology Σ_{CIA} used as an example throughout Section 4.1.

Εικόνα 10-οντολογία Σcia

5. Πειραματικά αποτελέσματα

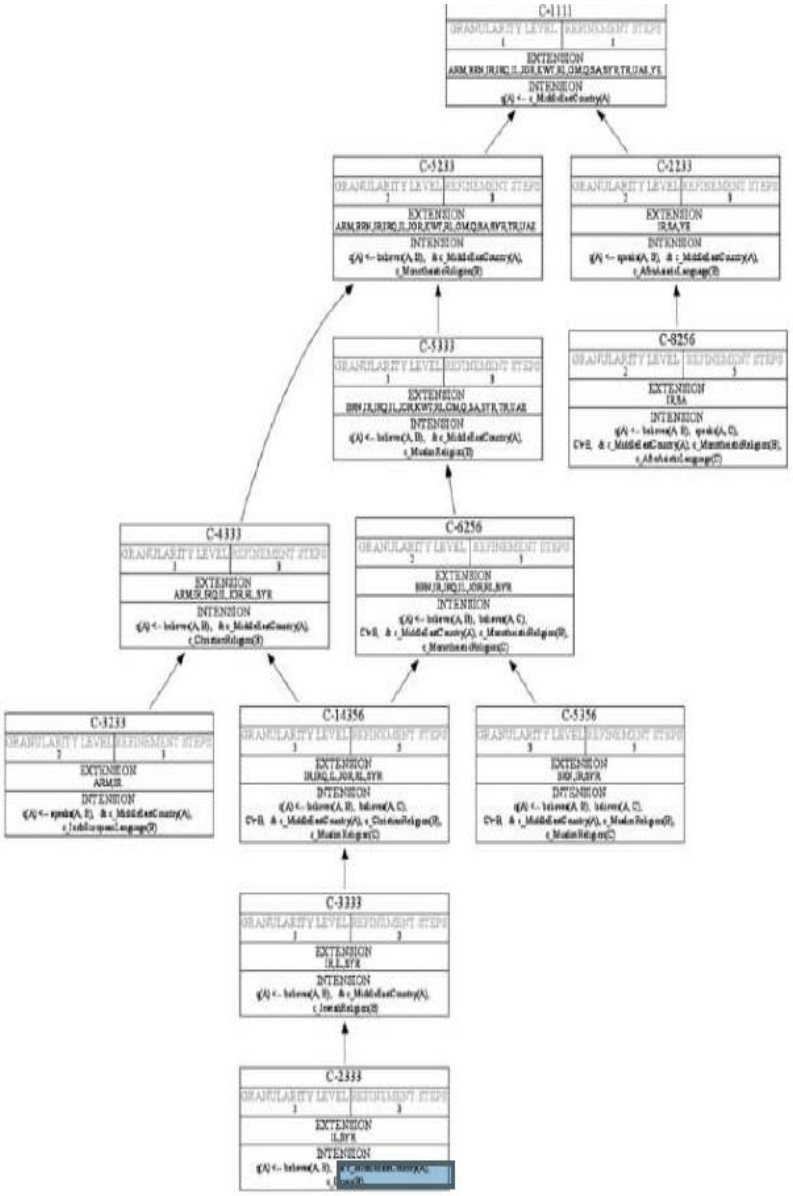


Fig. 3. Taxonomy $\mathcal{G}_{CIA}$ obtained with the m.g.d. criterion for $minG = 2$.

Εικόνα 11-ταξονομία

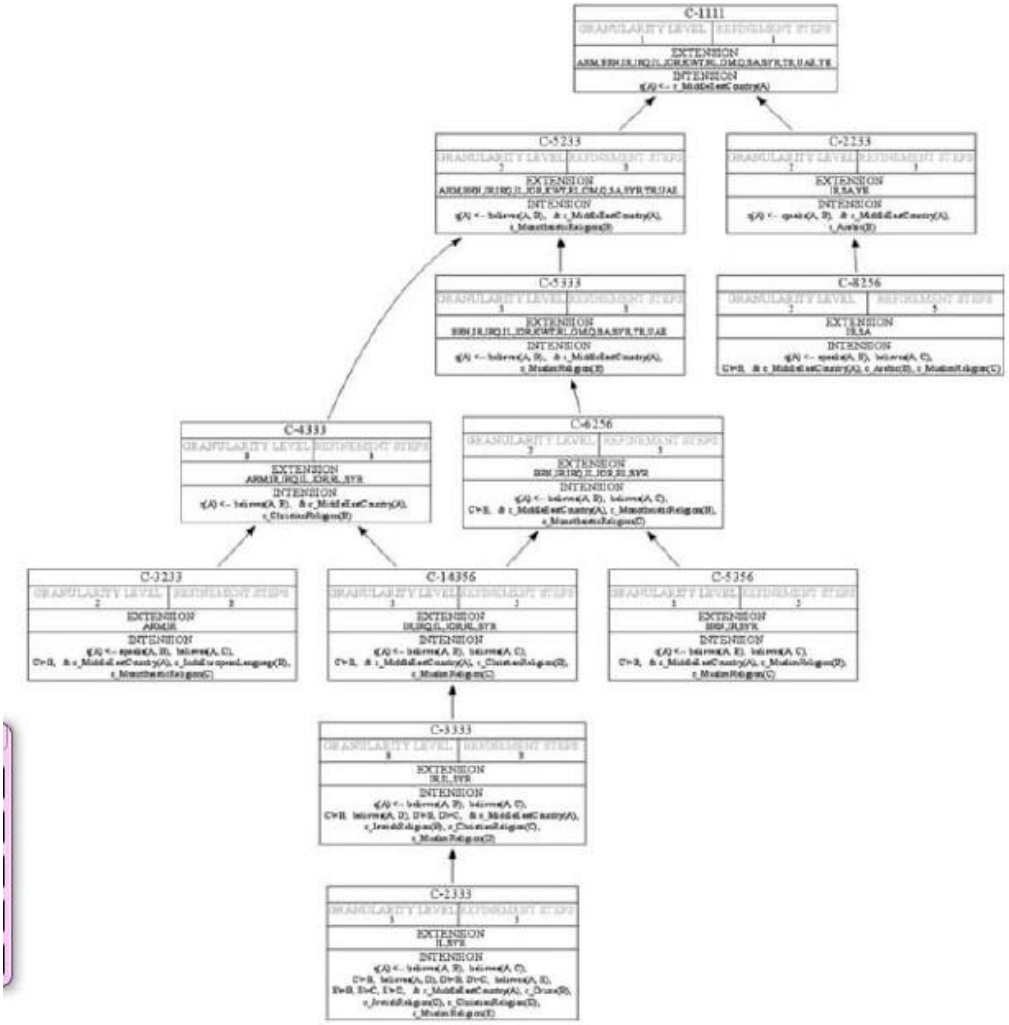


Fig. 4. Taxonomy $\mathscr{G}'_{\mathrm{CTA}}$ obtained with the m.s.d. criterion for $minG = 2$.

Εικόνα 12-ταξονομία

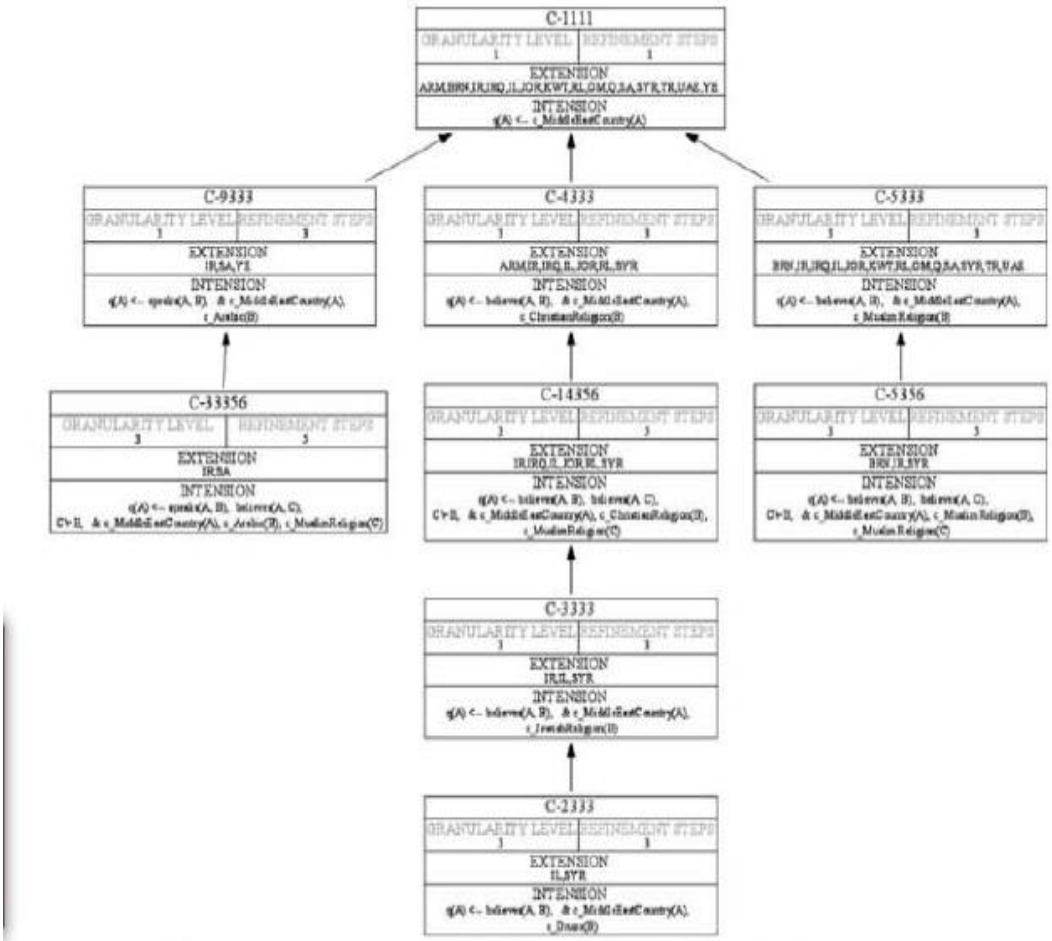
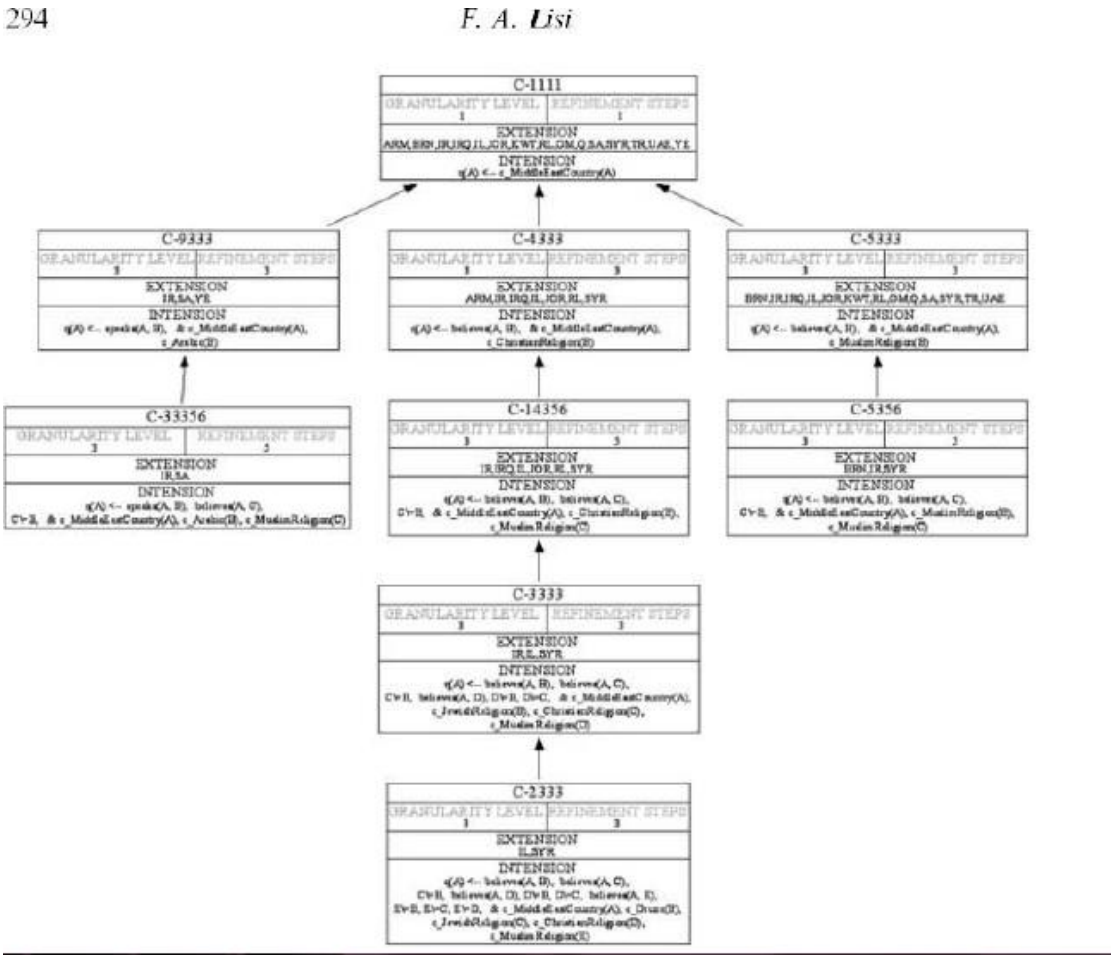


Fig. 5. Taxonomy $\mathscr{G}^{\pi}_{\mathrm{CIA}}$ obtained with the m.g.d. criterion for $minG = 3$.

Εικόνα 13-ταξονομία

6. Συμπεράσματα

Οι κανόνες δημιουργίας στην κορυφή των οντολογιών του Σημασιολογικού Ιστού, με επαγωγικό λογικό προγραμματισμό, μπορούν να αυτοματοποιηθούν με την εφαρμογή αλγορίθμων μηχανικής μάθησης σε δεδομένα που εκφράζονται με τους υβριδικούς φορμαλισμούς που συνδυάζουν DLs και ρήτρες Horn.
Έχει προταθει, ένα γενικό πλαίσιο για τη μάθηση σε AL - log
Έχουμε δύο επιθυμητές ιδιότητες οι οποίες εκτιμώνται ιδιαίτερα τόσο στην ILP και στην περιοχή της Σημασιολογικής Web εφαρμογής .



Εικόνα 14-Τελική Ταξονομία

Παρουσιάστηκε ένα πρόβλημα . Αυτό λοιπόν εμφανίζεται όταν μια οντολογία δίνεται ως είσοδος και επιστρέφει επιμέρους έννοιες μίας από των εννοιών στην οντολογία . Ένα ιδιαίτερο χαρακτηριστικό της ρύθμισης μας για αυτό το πρόβλημα είναι ότι οι προθέσεις αυτών των επιμέρους εννοιών είναι με τη μορφή των κανόνων που έχουν κατασκευαστεί αυτόματα με την ανακάλυψη ισχυρών συσχετίσεις μεταξύ των εννοιών της οντολογίας εισόδου . Η ιδέα του να καταφεύγουν σε συχνές αναζητήσεις στην μάθηση της οντολογίας, έχει ήδη ερευνηθεί (Maedche και Staab 2000).
Για το μέλλον πρόκειται να αξιολογηθεί εκτενώς αυτή η προσέγγιση για σημαντικά μεγάλες οντολογίες . Υπάρχει έλλειψη των προτύπων αξιολόγησης στην μάθηση της οντολογίας . Συγκριτικές εργασίες στον τομέα αυτό θα βοηθούσαν να επιλεγεί η κατάλληλη μέθοδος .

Ένα βήμα προς αυτή την κατεύθυνση είναι το πλαίσιο που παρουσιάζεται στην (Bisson et al., 2000) , αλλά έχει σχεδιαστεί για Οντολογίες εκχύλισης.
Ανεξάρτητα από την απόδοση , κάθε προσέγγιση έχει τα δικά της πλεονεκτήματα . Η προσέγγισή αυτή έχει τα πλεονεκτήματα του που ασχολούνται με την έκφραση των οντολογιών και είναι εννοιολογική. Μια άλλη κατεύθυνση των μελλοντικών εργασιών μπορεί να είναι η επέκταση του προηούμενου κειμένου, προς υβριδικούς φορμαλισμούς , που είναι πιο εκφραστικοί.

Κεφάλαιο 3ο

Μεταγραφική OWL και κανόνες σημασιολογικού ιστού σε Prolog : Προχωρώντας προς τα προγράμματα περιγραφικής λογικής.

Στο παρακάτω κείμενο, ερευνάται η αλληλεπίδραση μεταξύ του κανόνα και τα στρώματα της οντολογίας ή το Σημασιολογικό Ιστό , μέσω της σύγκρισης δύο επιλογών : α) Χρησιμοποιώντας OWL και την επέκταση του κανόνα SWRL της, για να να αναπτύξει μια ολοκληρωμένη γλώσσα οντολογίας / κανόνα , και β) Οι κανόνες layering μιας οντολογίας ή πάνω σε μία οντολογία, με ruleML και OWL .

Η ένωση του σημασιολογικού Ιστού και του Λογικού Προγραμματισμού, δημιούργησε ένα νέο τεχνολογικό πρότυπο, που ονομάζεται περιγραφή λογικού προγραμματισμού.

Επίσης έχουμε τη δημιουργία των SWORIER, που είναι οι οντολογίες του Σημασιολογικού Ιστού και κανόνες που αφορούν τη διαλειτουργικότητά του με Αποτελεσματική Συλλογιστική. Αυτό το σύστημα χρησιμοποιεί λογική του προγράμματος MING και απαντάει σε ερωτήματα σχετικά με οντολογίες και κανόνες. Έχει επιπλέον γραφτεί ενα σύνολο γενικότερων κανόνων σε Prolog, οι οποίοι επιβάλλουν την σημασιολογία των αρχικών OWL. Για να γίνει αυτό , ήταν απαραίτητο να αντιμετωπιστούν ορισμένα ζητήματα που σχετίζονται με την άρνηση , τον κόσμο της ανοιχτής παραδοχής, διαζευκτικά συμπεράσματα , που απαριθμούν κατηγορίες και που ισοδυναμούν με άτομα , μηνύματα λάθους , υπαρξιακή ποσοτικοποίηση , περιορισμούς πληθικότητας , εις διπλούν γεγονότα , κυκλικές ιεραρχίες , και ανώνυμες κατηγορίες κα.

3.2 Υπόβαθρο

Ερευνάται η αλληλεπίδραση μεταξύ των κανόνων και των οντολογιών στο Σημασιολογικό Ιστό, για να προσδιοριστεί, πώς μια τυπική γλώσσα πρέπει να τα εκφράζει καλύτερα. Ειδικότερα , για να διαπιστωθεί αν πρέπει να ενσωματωθεί μια οντολογία και η ταυτότητα στους αντίστοιχους κανόνες ή στρώσεις , θα καθορίζεται μετάφραση και θα εκτελούνται ενημερώσεις σε α) SWRL , η οποία ενσωματώνει την OWL με τους κανόνες Και β) ruleML σε επίπεδα πάνω από την OWL.

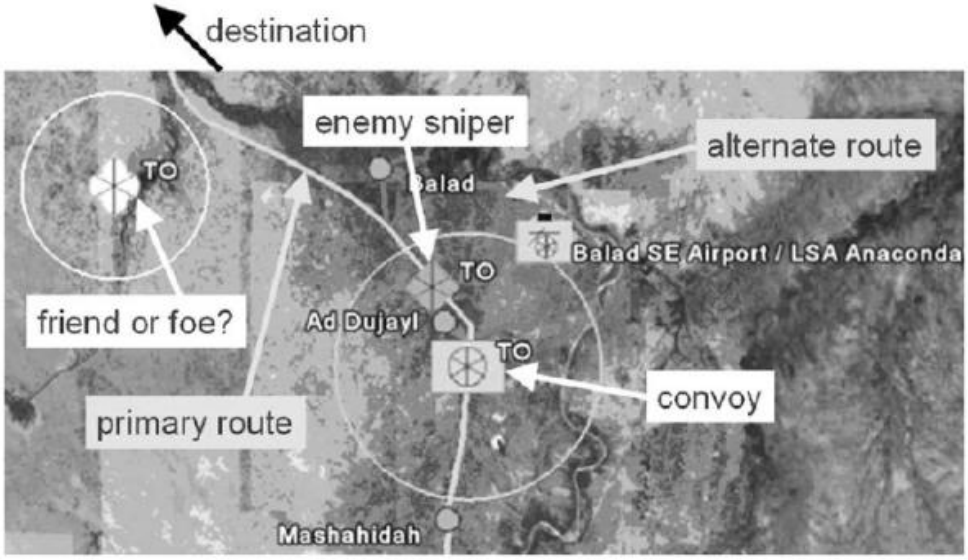


Fig. 1. A military task.

Εικόνα 15-Αρχικό πείραμα

Τα αρχικά πειράματα έγιναν σε μια στρατιωτική βάση για έλεγχο του τομέα με μια φάλαγγα προμηθειών που διακινούνται μέσα από ένα ακάλυπτο χώρο. (Εικόνα 15), παρουσιάζεται λοιπόν μία κατάσταση όπου μια συνοδεία κινείται βόρεια κατά μήκος της πρωτεύουσας διαδρομής και πλησιάζει την τοποθεσία όπου με τεχνητη νοημοσύνη έχει αναφερθεί ένας ελεύθερος σκοπευτής, της εχθρικής πλευράς, σταθμευμένος. Νέα στοιχεία μπορούν να είναι διαθέσιμα ανά πάσα στιγμή , όπως η ανακάλυψη ενός αντικειμένου ή η αρχή μιας αμμοθύελλας . Το σύστημα έχει κανόνες που πυροδοτούν ειδοποιήσεις και συστάσεις για την έκθεση στο διοικητή φάλαγγας.
Έτσι, σε περίπτωση ανάγκης, το σύστημα θα μπορούσε να πει στο διοικητή συνοδείας , " ALERT : έκθεση των μυστικών υπηρεσιών του εχθρού, ελεύθερος σκοπευτής στην περιοχή " και "RECOMMENDAT ION : Πάρτε εναλλακτική διαδρομή . "

Οι περισσότερες γλώσσες αναπαράστασης γνώσης και τα συστήματα που βασίζονται στη γνώση, χρησιμοποιούν μια περιορισμένη έκδοση της πρώτης τάξης λογική (FOL) . Η FOL , ωστόσο, είναι ημι- καθοριζόμενη . Είναι δυνατό να αποφασισθεί ότι αν ένα θεώρημα είναι λογικό αμέσως συνεπάγεται η θεωρία FOL ,ότι δηλαδή μια απόδειξη τελικά θα βρεθεί , αλλά είναι μη αποφασισμένη η περίπτωση ένα θεώρημα να μην είναι λογικό και να συνεπάγεται μια απόδειξη του ότι δεν μπορεί ποτέ να βρεθεί .

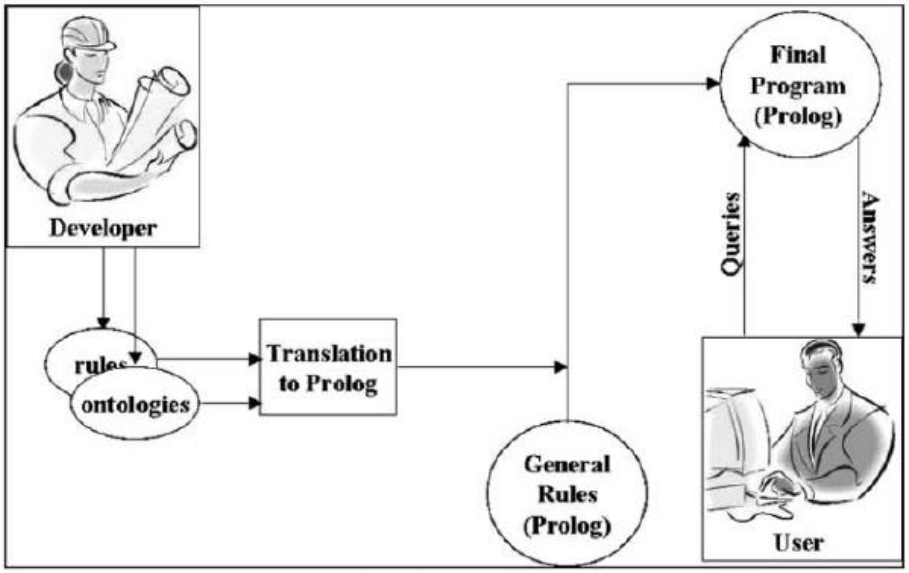


Fig. 2. System design.

Εικόνα 16-σχεδιασμός του συστήματος

3.3 Σχεδιασμός του συστήματος

Η παραπάνω εικόνα περιγράφει το σχεδιασμό του συστήματος SWIPER.
Ο δημιουργός δημιουργεί οντολογίες, βάσεις γνώσης ή / και κανόνες στον φορμαλισμό της OWL . Rulem L. και / ή ΣΑΧ Λ πίτες Εξετάσεων της OWL .
Με αποτέλεσμα οι κωδικοί που εμφανίζονται στους Πίνακες της γλώσσας, θα περιλαμβάνουν λέξεις όπως "είναι " και " του " σε κατηγορήματα και τα ονόματα για να αποφευχθεί η ασάφεια . Διαφορετικά , υπάρχει ο κίνδυνος να παρερμηνεύσουν τους ρόλους των επιχειρημάτων . Για παράδειγμα , η πρόταση « τα κράτη (X , Y)» θα μπορούσε να ερμηνευθεί ως «το Χ είναι ένα μέλος του Υ « ή «το Χ έχει το μέλος Υ» και να υπάρχει ασάφεια.
Παρακάτω παρουσιάζονται οι «μεταφράσεις» των πινάκων της γλώσσας.

a.	<sniper rdf:ID="smith"> <hasCombatIntent rdf:resource="#friendlyIntent"/> </sniper>
b.	<Implies> <head> <Atom> <opr> <Rel>redForceTheaterObject</Rel> </opr> <Var>X</Var> </Atom> </head> <body> <Atom> <opr> <Rel>redForceTheaterObject</Rel> </opr> <Var>X</Var> </Atom> </body> </Implies>
c.	<swrlx:classAtom> <owlx:Class owlx:name = "sniper"/> <owlx:Individual owlx:name="smith"/> </swrlx:classAtom> <ruleml:imp> <ruleml:_body> <swrlx:individualPropertyAtom swrlx:property="isDescribedBy"> <ruleml:var>T</ruleml:var> <ruleml:var>G</ruleml:var> </swrlx:individualPropertyAtom> <swrlx:individualPropertyAtom swrlx:property="hasSpeedObservation"> <ruleml:var>G</ruleml:var> <ruleml:var>S</ruleml:var> </swrlx:individualPropertyAtom> </ruleml:_body> <ruleml:_head> <swrlx:individualPropertyAtom swrlx:property="hasSpeed"> <ruleml:var>T</ruleml:var> <ruleml:var>S</ruleml:var> </swrlx:individualPropertyAtom> </ruleml:_head> </ruleml:imp>

Table 2. *Translations.*

a.	ismemberof(smith, sniper). hasproperty with(smith, hasCombatIntent, friendlyIntent).
b.	ismemberof(X, redForceTheaterObject) :- isMemberOf(X, redForceTheaterObject).
c.	ismemberof(smith, sniper). hasproperty with(T, hasSpeed, S) :- hasPropertyWith(T, isDescribedBy, G), hasPropertyWith(G, hasSpeedObservation, S).
d.	sniper(smith). hasCombatIntent(smith, friendlyIntent).

Εικόνα 17-"μεταφράσεις"

3.4 Γεγονότα μετάφρασης

Η SWORIER χρησιμοποιεί μια σύνταξη διαφορετική από όσες έχουν ήδη αναφερθεί στα προηγούμενα κεφάλαια. Οι περισσότερες υλοποιήσεις της Prolog απαγορεύουν τις κατηγορηματικές μεταβλητές.Αντίθετα, κάνοντας τα ονόματα τάξεων και ονόματα ιδιοτήτων να είναι τα επιχειρήματα αντί των κατηγορημάτων, η SWORIER έχει την ευελιξία να γενικεύει τα προηγούμενα.

Table 3. *The transitive closure of subclass.*

a.	isSubClassOf(C, D) :- issubclassof(C, D). isSubClassOf(C, E) :- issubclassof(C, D), isSubClassOf(D, E).
b.	isSubClassOf(C, E) :- isSubClassOf(C, D), isSubClassOf(D, E).

Εικόνα 18-μετάφραση

3.5 Γενικοί κανόνες

Οι γενικοί κανόνες είναι φτιαγμένοι, να συλλάβουν τη σημασιολογία των πρωταρχικών της OWL . Για παράδειγμα. οι κανόνες του πίνακα, που προβλήθηκε προηγουμένως, επιβάλλουν την μεταβατικότητα της υποκατηγορίας .Υπάρχουν δύο διαφορετικά κατηγορήματα : issubclassof και isSubClassOf. Ένα κατηγόρημα θα είναι ανεπαρκές , διότι έχει αριστερή αναδρομή, με αποτέλεσμα να οδηγεί σε ένα άπειρο βρόχο.
Για λόγους συνέπειας , δημιουργήθηκαν δύο περιπτώσεις σε κάθε κατηγόρημα . Τα all-lowercase και camelcase. Κάθε κατηγόρημα έχει έναν κανόνα μετατροπής . Χρησιμοποιείται πάντα το all- lowercase κατηγόρημα, ενώ τα ερωτήματα των χρηστών είναι πάντα σε CamelCas. Γενικά ακολουθείται η σύμβαση :

Όλα τα κατηγορήματα στο «σώμα» του κανόνα είναι camelcase και το κατηγόρημα της κεφαλής, είναι all-lowercase.

Μερικές από τις εισόδους που παρέχονται σε SWORIER είναι rule ML κανόνες ή SWRL . Δεν είναι δύσκολο να μεταφραστούν οι εν λόγω κανόνες σε Prolog, επειδή είναι γραμμένο στο Horn μορφή ρήτρας. Ωστόσο , δεν μπορούμε να ελέγξουμε ποιοι κανόνες παρέχονται ούτε πώς γράφονται .

3.6 Προκλήσεις

1) Αρνήσεις.

Η Άρνηση σε Prolog δεν είναι το ίδιο με την άρνηση στην OWL . Το Rule ML και το SWRL της Prolog έχει πεπερασμένη αποτυχία άρνησης, πράγμα που σημαίνει ότι είναι not(T) είναι αληθές εάν αυτό δεν είναι δυνατό να αποδειχθεί ότι το T είναι αλήθεια.

2) Η Open World Υπόθεση

Στην Prolog , η close world υπόθεση «κρατάει». Πράγμα που σημαίνει ότι κάτι που δεν μπορεί να αποδειχθεί αληθές , πρέπει να είναι ψευδές . Εναλλακτικά , η OWL έχει μια Open World Υπόθεση, πράγμα που σημαίνει ότι ένας όρος είναι ψευδής μόνο εάν μπορεί να αποδειχθεί ψευδής.

Table 5. *A true/false query.*

?- Q.	?- logicNot(Q).	Is Q true?
yes	no	yes
no	yes	no
no	no	unknown
yes	yes	error

Table 6. *Complementary and disjoint classes.*

```
disjointClasses(C, D) :- complementaryClasses(C, D).
logicNot(isMemberOf(I, C)) :-
 disjointClasses(C, D), is_member_of(I, D).
isMemberOf(I, C) :-
 complementaryClasses(C, D), logicNot(is_member_of(I, D)).
```

Table 7. *Conjunction and disjunction in the head.*

a.	IF I is in C, THEN I is an individual, AND C is a class.
b.	`isindividual(I) :- isMemberOf(I, C).` `isclass(C) :- isMemberOf(I, C).`
c.	IF C and D are complementary classes, AND I is an individual, THEN I is in C, OR I is in D.
d.	`or(ismemberof(I, C), ismemberof(I, D)):-` `complementaryClasses(C, D), isIndividual(I).`

e.	`or(P, Q) :- P.`
	`or(P, Q) :- or(Q, P).`
	`or(or(P, Q), R) :- or(P, or(Q, R)).`
	`Q :- or(P, Q), logicNot(P).`

f.	`isMemberOf(I, C) :- or(P, isMemberOf(I, C)), logicNot(P).`
	`isSubClassOf(I, C) :- or(P, isSubClassOf(I, C)), logicNot(P).`
	`logicNot(Q) :- or(P, logicNot(T)), logicNot(Q).`
	`or(Q, R) :- or(P, or(Q, R)), logicNot(P).`

Εικόνα 19υπόθεση open world-

Κάποιες άλλες προκλήσεις αποτελούν τα παρακάτω:

3)Μηνύματα λάθους

Είναι επιθυμητό για τη SWORIER να ελέγχουμε τα δεδομένα για τη συνοχή . Για το σκοπό αυτό ακολουθούμε την εφαρμογή των κανόνων για να «πιάσει» τις ασυνέπειες . Η ασυνέπεια αντιμετωπίζεται με την αποστολή μηνύματος λάθους στον δημιουργό. Όμως υπάρχουν και άλλοι τρόποι για το χειρισμό των αντιφάσεων.

4)Αριθμημένες Κλάσεις

5)Πολλαπλοί όροι κεφαλής

6)Συμπληρωματικές και Ασύνδετες κλάσεις

7) Υπαρξιακή ποσοτικοποίηση

Η Prolog υποθέτει σιωπηρά ότι όλες οι μεταβλητές σε κάθε κανόνα είναι καθολικά ποσοτικές. Όμως η OWL πρέπει να καθορίσει υπαρξιακά τις ποσοτικοποιημένες μεταβλητές . Για παράδειγμα, ο κωδικός της OWL από τους πίνακές της, δηλώνει ότι κάθε αντικείμενο περιγράφεται από τουλάχιστον μία τεχνητή παρατήρηση. Υπάρχουν τρεις τρόποι για να επιβληθεί αυτός ο περιορισμός.

8)Πληθικότητα

Θεωρητικά, υπάρχει ένας απεριόριστος αριθμός πληθικότητας περιορισμών που ο κατασκυαστής θα μπορούσε να επιβάλει . Ωστόσο , μπορούμε να επεκτείνουμε το σχεδιασμό του συστήματος SWORIER, εισάγοντας μια νέα λειτουργική μονάδα και τυχόν περιορισμοί πληθικότητας που βρέθηκαν στις οντολογίες να σταλεί σε αυτή.

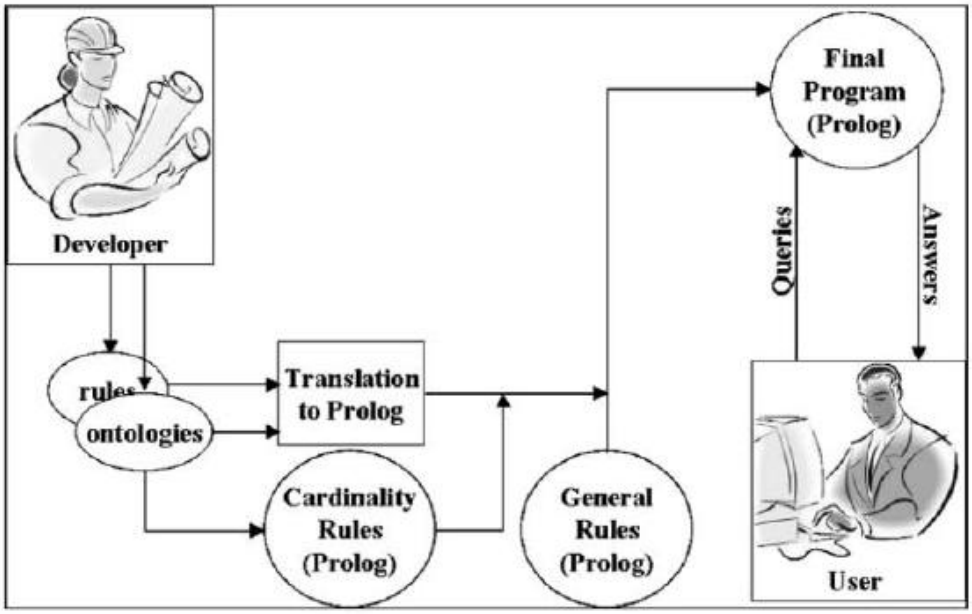


Fig. 3. The cardinality rules module.

Εικόνα 20-πληθικότητα

9)Διπλότυπα γεγονότα

Είναι επιθυμητό να αποφευχθεί στη SWORIER η παραγωγή του ιδίου γεγονότος περισσότερες από μία φορές. Το ερώτημα query(X,c, 1), θα προκαλέσει το σύστημα να επιστρέψει δύο αντίγραφα του γεγονότος (a, c, 1): ένα , λόγω του κανόνα I και το άλλο μέσα από την αλληλεπίδραση μεταξύ των κανόνων 2 και 3.

Αν και δεν είναι δύσκολο να αφαιρέσουμε επαναλαμβανόμενα γεγονότα σε μια διαδικασία μετά την επεξεργασία, μπορεί να οδηγήσει σε ένα σημαντικό κόστος στην απόδοση. Γι 'αυτό πρέπει να είναι σαφές ότι όταν διπλά γεγονότα δημιουργούνται μπορούν δυνητικά να επιβραδύνουν το πρόγραμμα σημαντικά.
Για να αποκλείσουμε τα διπλά γεγονότα, μπορούμε να προσθέσουμε τον όρο not(Y = C).

10) Κυκλικές Ιεραρχίες

Οι Κυκλικές ιεραρχίες κλάσεων και οι property κυκλικές ιεραρχίες μπορεί να είναι προβληματικές . Ας υποθέσουμε ότι η δεδομένη OWL οντολογία περιλαμβάνει κάποια γεγονότα. Στη συνέχεια, ο υπολογισμός του μεταβατικού κλεισίματός της Subclass με τη χρησιμοποίηση των κανόνων, παράγει έναν άπειρο αριθμό των απαντήσεων στο ερώτημα , isSubClassOf (X , Y) , όπως το σύστημα βρόγχων γύρω από τον κύκλο . Ακόμη και αν μπορούμε να πούμε ότι μια κυκλική ιεραρχία είναι εσφαλμένη, δεν μπορούμε να αποτρέψουμε τη δημιουργία τους. Έτσι SW0RIER θα πρέπει να είναι σε θέση να το χειριστεί .
Προτείνουμε την αλλαγή των μεταβατικών κανόνων κλεισίματος της subclass.

11) Ανώνυμες Κλάσεις

12) Δυναμικές Αλλαγές

Μια άλλη χρήσιμη δυνατότητα είναι να αλλάξει η βάση γνώσεων κατά το χρόνο εκτέλεσης . Για παράδειγμα. στο έργο κομβόι μας , εκθέσεις νοημοσύνης μπορεί να έρθει μέσα σε έναν χρόνο ny κατά τη διάρκεια μιας Río Scena . και θέλουμε SWORIER να είναι σε θέση να ενσωματώσει τις νέες πληροφορίες στο ο
βάση γνώσεων . Αυτό πρέπει να γίνει μέσα σε λίγα δευτερόλεπτα . Έτσι , έχουμε ενεργοποιημένη SWOR IE R για να φιλοξενήσει δυναμικές αλλαγές των γεγονότων , προσθέτοντας ή αφαιρώντας τα γεγονότα κατά το χρόνο εκτέλεσης.

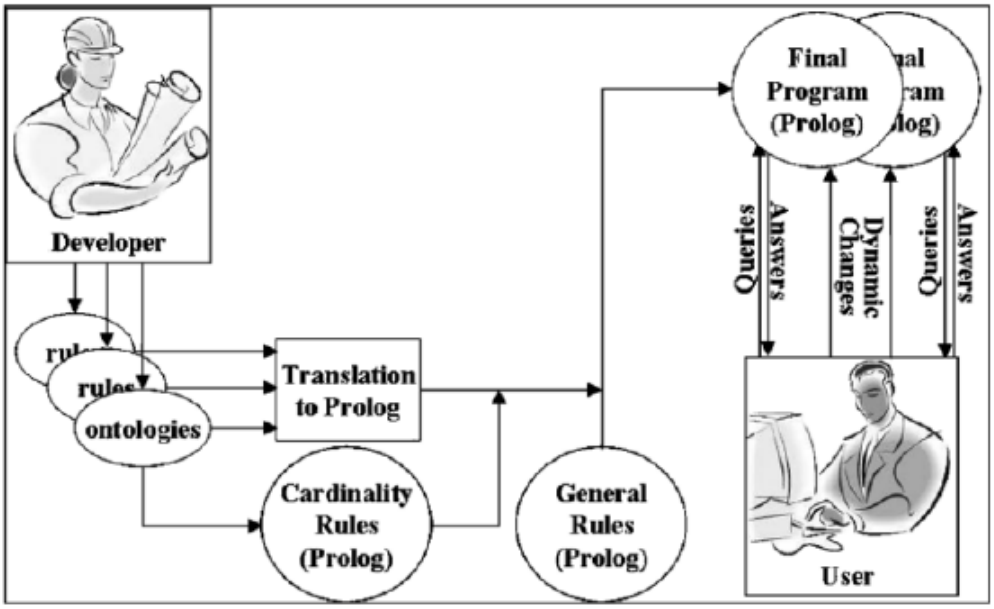


Fig. 4. Dynamic changes of facts and rules.

Εικόνα 21-Δυναμικές αλλαγές

Το SWOR IE R είναι επίσης ικανό να επιτύχει δυναμική αλλαγή των κανόνων , αλλά μόνο με περιορισμένο τρόπο.

13) Αποδοτικότητα

Στην αρχή το σύστημα SWORIER ήταν πολύ αργό. Αφού έγιναν αλλαγές, 1,5 ώρα απαιτείται για να ανταποκριθεί στα εξής ερωτήματα : Ποιες είναι οι θέσεις και τις ταχύτητες όλων των γνωστών μονάδων , και ποιες είναι οι τρέχουσες ειδοποιήσεις και συστάσεις ;
Η αποτελεσματικότητα και ο χρόνος που απαιτείται, εξαρτάται από την εφαρμογή .

14) Επεκτασιμότητα

Προκειμένου να καταστεί το σύστημα ορθό και τιθασευμένο κατά το χρόνο εκτέλεσης , μπήκε σε εφαρμογή μία offline τεχνική για να επιταχύνει το πρόγραμμα. Όλα τα πραγματικά περιστατικά που μπορούν να προκύψουν μετά την μετατροπή του προγράμματος, από μια αρχική intensional μορφή περνάνε σε μια εκτατική μορφή.

15) Ελαχιστοποίηση κώδικα

Μια άλλη βελτίωση της αποδοτικότητας μπορεί να εφαρμοστεί εάν ορισμένες γνώσεις είναι διαθέσιμες πριν από το χρόνο εκτέλεσης . Η ιδέα είναι να καταλάβουμε ποιοί από τους κανόνες είναι πραγματικά αναγκαίοι σε αυτές τις περιπτώσεις.
οι περιττοί κανόνες μπορεί να ρίξουν τη βελτίωση της αποτελεσματικότητας . Αν ένας κανόνας δεν μπορεί ποτέ να ενεργοποιείται με επιτυχία , τότε είναι περιττός . Επίσης , αν δεν τεθεί κάποιο ερώτημα ποτέ, θα έχει ως αποτέλεσμα τη δοκιμή ενός κανόνα , τότε ο κανόνας θα είναι πάλι περιττός. Αυτό όλο επιτυγχάνεται με έναν αλγόριθμο, ο οποίος περιγράφεται στον παρακάτω πίνακα.

a.	**Base Case 1:**	IF and THEN	P is a built-in predicate (reserved keyword) in Prolog, P is not `findall`, P is a satisfiable predicate.
	Base Case 2:	IF THEN	P is a predicate in a fact, P is a satisfiable predicate.
	Base Case 3:	IF and THEN and	P is the predicate in the head of a rule, R, each predicate in R's body is P, P is a satisfiable predicate, R is a satisfiable rule.
	Inductive Case 1:	IF and THEN and	P is the predicate in the head of a rule, R, every predicate in R's body is satisfiable, P is a satisfiable predicate, R is a satisfiable rule.
	Inductive Case 2:	IF and and THEN	a rule, R, has a term, T, in its body, where T is `assert(F)`, `asserta(F)`, or `assertz(F)`, P is the predicate of F, all of the predicates preceding T in the body of R are satisfiable predicates, P is a satisfiable predicate.
	Inductive Case 3:	IF and THEN	A is the second argument of a `findall`, all of the predicates in A are satisfiable predicates, the `findall` is treated as if it was a satisfiable predicate
b.	**Base Case:**	IF THEN	P is a predicate that can be called from outside the rules, [4] P is a testable predicate.
	Inductive Case 1:	IF THEN	`findall` is determined to be a testable predicate, each predicate in P's second argument is also testable.
	Inductive Case 2:	IF and THEN and	P is a predicate such that P or `not(P)` is found in the body of a rule, R, R's head can be determined to have a testable predicate, P is a testable predicate, R is a testable rule.

Εικόνα 22-αλγόριθμος ελαχιστοποίησης κώδικα

Κεφάλαιο 4ο

Querying XML αρχείων, σε Λογικό Προγραμματισμό.

Η XML, ή αλλιώς Extensible Markup Language, είναι απλή, πολύ ευέλικτη μορφή κειμένου που προέρχεται από το SGML. Αρχικά είχε σχεδιαστεί για να ανταποκριθεί στις μεγάλης κλίμακας ηλεκτρονικές εκδόσεις.
Η XML παίζει όλο και πιο σημαντικό ρόλο στην ανταλλαγή δεδομένων στον Παγκόσμιο Ιστό και αλλού. Η Γλώσσα X Path, είναι το αποτέλεσμα μιας προσπάθειας να παρέχονται τα μέρη της διεύθυνσης ενός εγγράφου XML. Αυτό γίνεται με μια γλώσσα επερωτήσεων έναντι ενός εγγράφου XML.
Ακολουθεί μια πρόταση για την εφαρμογή της γλώσσας X Path, στο λογικό προγραμματισμό, δηλαδή μέσω ενός προγράμματος λογικής.
Παρουσιάζεται το πώς θα καταχωρηθούν αρχεία XML, μέσω του λογικού προγραμματισμού. Οι κανόνες (rules), πρέπει να αποθηκεύονται στην κύρια μνήμη, ωστόσο τα γεγονότα(facts), αποθηκεύονται στη δευτερεύουσα μνήμη, χρησιμοποιώντας δύο είδη ευρετηρίων: ένα για κάθε ετικέτα XML, και άλλο για κάθε ομάδα ή τερματικό αντικείμενο. Επιπλέον, αναφέρεται ο τρόπος υποβολής ερωτημάτων μέσω της γλώσσας X Path, έναντι λογικών προγραμμάτων σε ένα έγγραφο XML.

4.1 XML και X Path

Ένα έγγραφο XML είναι ένα επισημασμένο δέντρο με εσωτερικούς κόμβους που αντιπροσωπεύουν τα composed ή μη τερματικά items και τα φύλλα, που αντιπροσωπεύουν τις τιμές ή τα τερματικά items.

```
<books>
<book year="2003">
<author> Abiteboul</author>
<author> Buneman</author>
<author> Suciu</author>
<title>Data on the Web</title>
<review> A <em> fine</em> book.</review>
</book>
<book year="2002">
<author> Buneman</author>
<title> XML in Scotland</title>
<review> <em> The <em> best</em> ever!</em> </review>
</book>
</books>
```

Εικόνα 23-παράδειγμα XML

Στην XML, οι ετικέτες που χρησιμοποιούνται για τον προσδιορισμό ενός συνόλου" books" που περιγράφονται με τη βοήθεια των ονομάτων του συγγραφέα, τον τίτλο και το review. Κάθε βιβλίο έχει ένα attribute, που ονομάζεται "year" .

Για κάθε element “book”, έχουμε τρία υποστοιχεία, αυτά του συγγραφέα, τον τίτλο και το review. Επιπλέον, το review των στοιχείων περιέχει υποστοιχεία που χρησιμοποιούνται για τη μορφοποίηση του κειμένου που περιγράφεται από αυτή.

4.2 Μετάφραση XML εγγράφων σε Λογικό Προγραμματισμό.

Σε γενικές γραμμές, ένα έγγραφο XML περιλαμβάνει:

(α) tagged στοιχεία τα οποία έχουν τη μορφή:

<att1 tag = V1? :::? att11 = 011> subelem1? :::? subelemk <= tag>

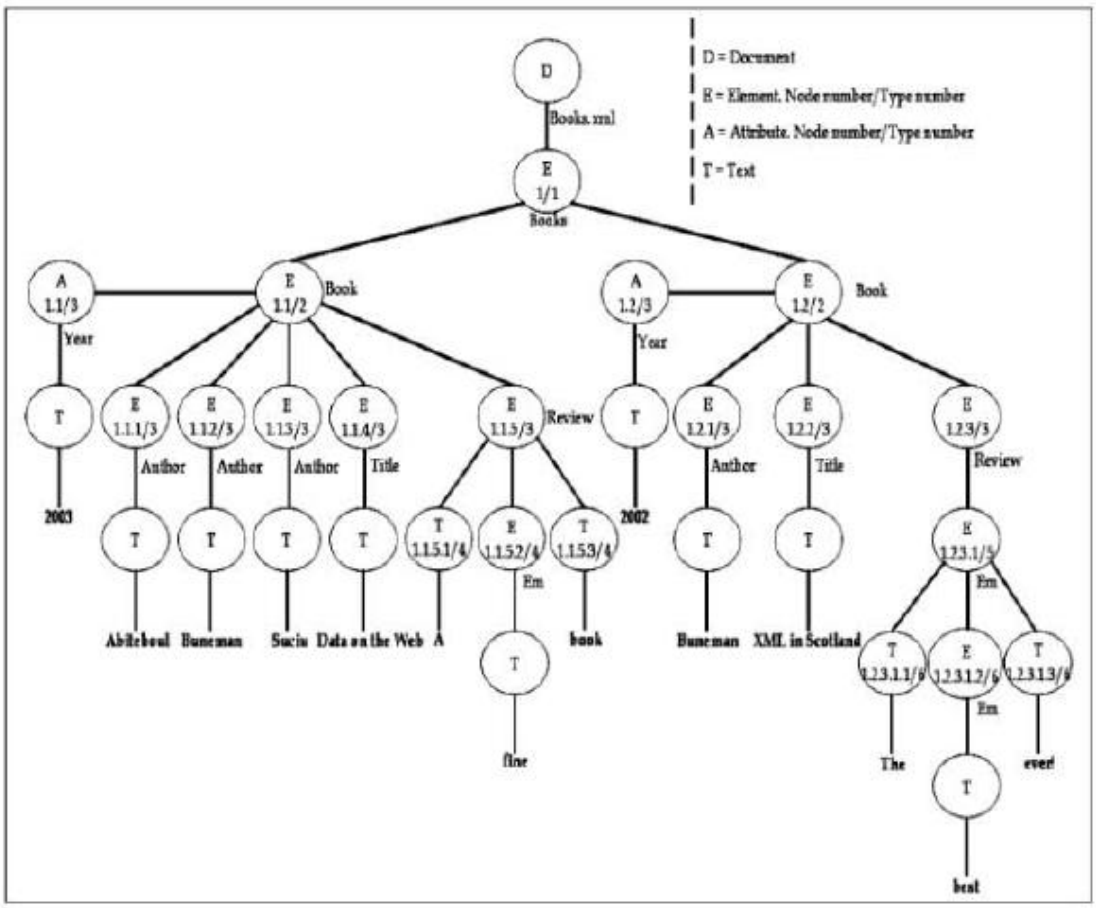
όπου att1? :::? att11 είναι τα ονόματα των χαρακτηριστικών (attributes), v1? :::? V11 είναι οι τιμές των χαρακτηριστικών υποτίθεται ότι έχουν ένα βασικό τύπο, πχ ακέραιοι, πραγματικοί αριθμοί κτλ και τους καταλόγους των ακεραίων ή πραγματικών αριθμών. Επίσης περιέχει τα subelem, που είναι τα υποστοιχεία.

(Β) χωρίς tag στοιχεία τα οποία έχουν ένα βασικό τύπο.

Αυτά, έχουν ένα βασικό τύπο και δεν έχουν τα χαρακτηριστικά. Διαφορετικά καλούνται εσωτερικοί κόμβοι. Δύο tagged στοιχεία είναι παρόμοια αν έχουν την ίδια δομή δηλαδή, έχουν την ίδια ετικέτα και τα χαρακτηριστικά τα ονόματα και τα υποστοιχεία είναι παρόμοια. Στοιχεία χωρίς ετικέτα είναι πάντοτε όμοια.

4.3 Αριθμώντας XML αρχεία.

Για να οριστεί η μετάφραση, πρέπει να αριθμηθούν οι κόμβοι του εγγράφου XML. Ο στόχος είναι να αναγνωριστεί κάθε εσωτερικός κόμβος και τα φύλλα του δέντρου που αντιπροσωπεύεται από το XML έγγραφο.
Ας σημειωθεί ότι στην πράξη, το είδος και η αρίθμηση του κόμβου στα έγγραφα XML μπορούν να παραχθούν ταυτόχρονα με την ίδια φορά όπως η μετάφραση στο πρόγραμμα λογικής. Στην πραγματικότητα, ο τύπος και ο αριθμημένος κόμβος του πρωτοτυπου εγγράφου XML δεν δημιουργείται ως αρχειο XML.



Εικόνα 24-Αρίθμηση XML

4.4 Μετάφραση XML εγγράφων.

Για κάθε μη-τερματικό στοιχείο, στον αριθμημένο τύπο και κόμβο εγγράφου XML, ισχύει:

$$< tag\ att_1 = v_1;:::;att_n = v_n;nodenumber = i;typenumber = k >$$
$$elem_1;:::;elem_s < =tag >$$

we consider the following rule, called *schema rule*:

$$\begin{aligned} tag(tagtype(Tag_{i_1};:::;Tag_{i_r};\ [Att_1;:::;Att_n]);NodeTag;k):- \\ tag_{i_1}(Tag_{i_1};[NodeTag_{i_1}|NodeTag];r), \\ :::, \\ tag_{i_r}(Tag_{i_r};[NodeTag_{i_r}|NodeTag];r), \\ att_1(Att_1;NodeTag;r), \\ :::, \\ att_n(Att_n;NodeTag;r): \end{aligned}$$

Εικόνα 25-Μετάφραση

Όπου:

→Tag type είναι ένα νέο σύμβολο που χρησιμοποιείται για το χτίσιμο ενός όρου Prolog που περιέχει το έγγραφο XML

→$\{tag_{i_j} \mid i_j \in \{1;:::;s\},\ 1 \leq j \leq t\}$ αυτή η ετικέτα είναι το σύνολο των ετικετών των σηματοδοτημένων στοιχείων elem1……

→ Tagi; ::: Tagκ είναι μεταβλητές

→att1; ::: ;; Attn είναι τα ονόματα των γνωρισμάτων

→ATT1; :::; Atn είναι μεταβλητές, μία για κάθε χαρακτηριστικό όνομα

→node tags.. είναι μεταβλητές (χρήση για την αναπαράσταση του πρώτου ψηφίου του αριθμού κόμβου.)

→Node - Tag είναι μια μεταβλητή (που χρησιμοποιείται για να αντιπροσωπεύει τον αριθμό κόμβου της ετικέτας).

→k είναι ο αριθμός ετικέτας.

→r είναι ο αριθμός των σηματοδοτημένων στοιχείων.

4.5 Ειδίκευση Προγράμματος για X Path εκφράσεις.

Παρακάτω θα υποδειχθεί η τεχνική για την αναζήτηση X Path εκφράσεων σε ένα έγγραφο XML που εκπροσωπείται από ένα πρόγραμμα λογικής.

→X Path Semantics

Μία X Path έκφραση xpathexpr έχει τη μορφή $=expr_1=:::=expr_n$ όπου κάθε απλή X Path έκφραση είναι:

1. $expr \equiv tag$
2. $expr \equiv tag[cond]$
3. $expr \equiv @att$
4. $expr \equiv text()$

and cond is a boolean condition which has the form:

(a) $cond \equiv tag = value$
(b) $cond \equiv @att = value$
(c) $cond \equiv cond_1$ and $cond_2$
(d) $cond \equiv cond_1$ or $cond_2$
(e) $cond \equiv xpathexpr$

Εικόνα 26-X Path εκφράσεις

Για ένα μη-τερματικό tagged στοιχείο ισχύει:

$$< \text{tag } att_1 = v_1;:::;att_n = v_n > elem_1;:::;elem_s < /\text{tag} >$$

then

(a.1):

$$subtree(\mathscr{X}/expr_r/:::/expr_n) =_{def}$$
$$< \text{tag } att_1 = v_1;:::;att_n = v_n >$$
$$subtree(elem_1/expr_{r+1}/:::/expr_n);$$
$$::::;$$
$$subtree(elem_s/expr_{r+1}/:::/expr_n);$$
$$elem_{i_1};$$
$$::::;$$
$$elem_{i_k}$$
$$< /\text{tag} >$$

whenever $r < n$ and $\mathscr{X}$ satisfies $expr_r$; where $elem_{i_1};:::;elem_{i_k}$ is the subsequence of $elem_1;:::;elem_s$ satisfying cond whenever $expr_r \equiv tag[cond]$;

(a.2):

$$subtree(\mathscr{X}/expr_n) =_{def} \mathscr{X}$$

whenever $r = n$ and $\mathscr{X}$ satisfies $expr_n$; and

(a.3):

$$subtree(\mathscr{X}/expr_r/:::/expr_n) =_{def} \square$$

otherwise.

(b) If $\mathscr{X}$ is a terminal tagged element then

(b.1):

$$subtree(\mathscr{X}/expr_r/:::/expr_n) =_{def} \mathscr{X}$$

whenever $r = n$ and $\mathscr{X}$ satisfies $expr_r$; and

(b.2):

$$subtree(\mathscr{X}/expr_r/:::/expr_n) =_{def} \square$$

Εικόνα 27-α case

Αλλιώς ισχύει:

(b.1):

$$subtree(\mathscr{X}/expr_r/:::/expr_n) =_{def} \mathscr{X}$$

whenever $r = n$ and $\mathscr{X}$ satisfies $expr_r$; and

(b.2):

$$subtree(\mathscr{X}/expr_r/:::/expr_n) =_{def} \square$$

otherwise.

Εικόνα 28-β case

(c) If $\mathscr{X}$ has a basic type then

(c.1):

$$subtree(\mathscr{X}/text()) =_{def} \mathscr{X}$$

and

(c.2):

$$subtree(\mathscr{X}/xpathexpr) =_{def} \square$$

whenever $xpathexpr \neq text()$

where $\square$ denotes the empty sequence.

Εικόνα 29-b case 2

Τέλος, το υποδένδρο που ορίζεται από την X Path, μπορεί να θεωρηθεί ως το υποδένδρο της εισόδου του εγγράφου XML, το οποίο διασχίζεται για να δοθεί απάντηση στο ερώτημα. Στην πράξη, η απάντηση σε ένα ερώτημα X Path, αποτελείται από την ακολουθία των υποδένδρων (δηλαδή το δάσος) του δέντρου ορίζεται από την έκφραση X Path, του οποίου η ετικέτα είναι ίση με τη δεξιότερη ετικέτα του ερωτήματος της X Path.
Δηλαδή:

<title> XML in Scotland< /title>

4.6 Schema Rule Specialization

Το πρώτο βήμα του προγράμματος εξειδίκευσης αποτελείται από ένα κατηγόρημα που απομακρύνεται από τους κανόνες του σχήματος.

Κάθε X Path έκφραση $\text{xpathexpr} = =\text{expr}_1 :::=\text{expr}_n$ μπορεί να αντιστοιχιστεί σε μία έκφραση ισότητας X Path.

Αυτό αντιστοιχίζεται ως εξής:

1. $\text{expr} \equiv \text{tag}: \text{FE}(\text{expr}) =_{\text{def}} \text{expr}:$
2. $\text{expr} \equiv \text{tag}[\text{cond}]: \text{FE}(\text{expr}) =_{\text{def}} \text{tag}[\text{FE}(\text{cond})]$
3. $\text{expr} \equiv @\text{att}: \text{FE}(\text{expr}) =_{\text{def}} @\text{att}$
4. $\text{expr} \equiv \text{text}(): \text{FE}(\text{expr}) =_{\text{def}} \text{text}()$
5. $\text{cond} \equiv \text{tag} = \text{value}: \text{FE}(\text{expr}) =_{\text{def}} \text{tag}$
6. $\text{cond} \equiv @\text{att} = \text{value}: \text{FE}(\text{expr}) =_{\text{def}} @\text{att}$
7. $\text{cond} \equiv \text{cond}_1 \text{ and } \text{cond}_2: \text{FE}(\text{expr}) =_{\text{def}} \text{FE}(\text{cond}_1) \text{ and } \text{FE}(\text{cond}_2)$
8. $\text{cond} \equiv \text{cond}_1 \text{ or } \text{cond}_2: \text{FE}(\text{expr}) =_{\text{def}} \text{FE}(\text{cond}_1) \text{ or } \text{FE}(\text{cond}_2)$
9. $\text{cond} \equiv \text{xpathexpr}: \text{FE}(\text{expr}) =_{\text{def}} \text{FE}(\text{xpathexpr})$

Εικόνα 30-schema rule

Επιπλέον ισχύουν οι εξής αντιστοιχίσεις:

1. $\text{fexpr} \equiv \text{tag}$
2. $\text{fexpr} \equiv \text{tag}[\text{cond}]$
3. $\text{fexpr} \equiv @\text{att}$
4. $\text{fexpr} \equiv \text{text}()$

and **cond** is a free of equalities boolean condition which has the form:

(a) $\text{cond} \equiv \text{cond}_1 \text{ and } \text{cond}_2$
(b) $\text{cond} \equiv \text{cond}_1 \text{ or } \text{cond}_2$
(c) $\text{cond} \equiv \text{xpathfree}$

Το δεύτερο στάδιο του προγράμματος εξειδίκευσης αποτελείται από την αφαίρεση της ισότητας από την αρχική έκφραση X Path, όταν αυτή είναι ελεύθερη από ισότητες καθώς επίσης από το να δημιουργήσει ένα σύνολο στόχων από αυτές τις ισότητες.

4.7 Ανασυγκρότηση της απάντησης.

Για να ξαναχτιστεί η απάντηση, θα πρέπει να κάνουμε reasoning ως ακολούθως.
Ένα λογικό πρόγραμμα που λαμβάνεται από ένα έγγραφο XML περιέχει κανόνες σχήματος και τα γεγονότα της $att(value;i;r)$ και ετικέτας $tag(value;i;r)$, και αντιστρόφως, από αυτό το σύνολο των γεγονότων και των κανόνων του σχήματος λοιπόν, μπορεί να ξαναχτιστεί το έγγραφο.
Ένα άλλο παράδειγμα χτισίματος του εγγράφου από την αρχή είναι το εξής μέσω Prolog:

book(booktype('Abiteboul', Title, reviewtype('A ', fine, [ ]),['2003']),[1,1],2).

Ως εκ τούτου, το έγγραφο XML αντιπροσωπεύει την απάντηση μιας έκφρασης X Path, kai ορίζεται ως έγγραφο που λαμβάνεται από τους εξειδικευμένους κανόνες σχήματος.

4.8 Τιμαριθμική αναπροσαρμογή

Το μοντέλο αποθήκευσης και ταξινόμησης στην προσέγγισή αυτή είναι ως εξής:

• Χρήση κύρια μνήμη για την αποθήκευση των κανόνων σχήματος.
• Χρήση δευτεροβάθμιας μνήμης (δηλαδή αρχεία) για την αποθήκευση των γεγονότων.
• ταξινόμηση γεγονότων στην δευτερεύουσα μνήμη.
• Υπάρχουν δύο είδη δεικτών: ένας για τα ονόματα τιμαριθμικής αναπροσαρμογής κατηγόρηματος, και άλλος για index των group of facts.

Η χρήση της κύριας μνήμης για την αποθήκευση των κανόνων σχήματος είναι δικαιολογημένη λόγω του ότι στις περισσότερες περιπτώσεις ο αριθμός των κανόνων σχήματος είναι μικρός. Η δευτερεύουσα μνήμη για την αποθήκευση γεγονότων υποστηρίζεται έτσι.

□rst index	second index	group identi□er	facts
author	*pos(1, 0).* *pos(2, 0).* *pos(3, 0).* *pos(9, 8).*	[1;1]	*(0) year('2003', [1, 1], 3).* *(1) author('Abiteboul', [1, 1, 1], 3).* *(2) author('Buneman', [2, 1, 1], 3).* *(3) author('Suciu', [3, 1, 1], 3).* *(4) title('Data on the Web', [4, 1, 1], 3).*
em	*pos(6,5).* *pos(12,11).*	[5;1;1]	*(5) unlabeled('A ', [1, 5, 1, 1], 4).* *(6) em(fine, [2, 5, 1, 1], 4).* *(7) unlabeled(' book.', [3, 5, 1, 1], 4).*
title	*pos(4, 0).* *pos(10, 8).*	[2;1]	*(8) year('2002', [2, 1], 3).* *(9) author('Buneman', [1, 2, 1], 3).* *(10) title('XML in Scotland', [2, 2, 1], 3).*
unlabeled	*pos(5, 5).* *pos(7, 5).* *pos(11, 11).* *pos(13, 11).*	[1;3;2;1]	*(11) unlabeled('The ', [1, 1, 3, 2, 1], 6).* *(12) em(best, [2, 1, 3, 2, 1], 6).* *(13) unlabeled(' ever!', [3, 1, 3, 2, 1], 6).*
year	*pos(0, 0).* *pos(8, 8).*		

Εικόνα 31-indexing example

In the case of the main goal *tag*(_; *Var*; _), the first index will be ever used.

Now, we show the *trace of the execution* of the *X Path* query =*books*=*book*[@*year* = *2002 and author* = "*Buneman*"]=*review* with respect to the above indexing structure.

---

1. call of *book*(*booktype*(*Buneman*; _*G12073*; _*G12074*; [*2002*]); _*G12078*; *2*) (Rule a)
2. call of *year*(*2002*; _*G12128*; *3*) (Rule a)
3. first index accessing to position *0* due to *year*(*2002*; _*G12128*; *3*); recovering *year*(*2003*; [*1*; *1*]; *3*); **fail**.
4. first index accessing to position *8* due to *year*(*2002*; _*G12128*; *3*); recovering *year*(*2002*; [*2*; *1*]; *3*); storing that the position of group [*2*; *1*] is *8*; **success**.
5. call of *author*(Buneman; [_G12100; 2; 1]; 3) (Rule a)
6. second index accessing to position *8* due to the position of group [*2*; *1*] is *8*; recovering *author*(Buneman; [*1*; *2*; *1*]; *3*); **success**
7. call of *review*(_*G12151*; [_*G12148*; *2*; *1*]; *3*) (Rule a)
8. call of *unlabeled*(_*G12190*; [_*G12187*; _*G12212*; *2*; *1*]; *4*) (Rule b)
9. first index accessing to position *11* due to *unlabeled*(_*G12243*; [_*G12240*; _*G12265*; _*G12268*; *2*; *1*]; 6); recovering *unlabeled* (*The*; [*1*; *1*; *3*; *2*; *1*]; *6*); storing that the position of group [*1*; *3*; *2*; *1*] is *11*; **success**.
10. first index accessing to position *13* due to *unlabeled*(_*G12243*; [_*G12240*; _*G12265*; _*G12268*; *2*; *1*]; 6); recovering *unlabeled* (*ever*!; [*3*; *1*; *3*; *2*; *1*]; *6*); storing that position of group [*1*; *3*; *2*; *1*] is *11*; **success**
11. call of *em*(_*G12261*; [_*G12258*; _*G12283*; _*G12286*; *2*; *1*]; *6*) (Rule c)
12. second index accessing to position *11* due to *em*(_*G12261*; [_*G12258*; _*G12283*; _*G12286*; *2*; *1*]; *6*) and that position of group [*1*; *3*; *2*; *1*] is *11*; recovering *em*(*best*; [*2*; *1*; *3*; *2*; *1*]; *6*); **success**
13. *em*(*emtype*(*The*; *best*; [ ]); [*1*; *3*; *2*; *1*]; *5*) **success**
14. *em*(*emtype*(*ever*!; *best*; [ ]); [*1*; *3*; *2*; *1*]; *5*) **success**
15. *review*(*reviewtype*(*emtype*(*The*; *best*; [ ]); [ ]); [*3*; *2*; *1*]; *3*) **success**
16. *review*(*reviewtype*(*emtype*(*ever*!; *best*; [ ]); [ ]); [*3*; *2*; *1*]; *3*) **success**
17. *book*(*booktype*(*Buneman*; _*G12316*; *reviewtype*(*emtype*(*The*; *best*; [ ]); [ ]); [*2002*]); [*2*; *1*]; *2*) **success**
18. *book*(*booktype*(*Buneman*; _*G12316*; *reviewtype*(*emtype*(*ever*!; *best*; [ ]); [ ]); [*2002*]); [*2*; *1*]; *2*) **success**

Εικόνα 32-τελική ταξινόμηση

4.9 Παραδείγματα Querying

XPath Query	Meaning
⊙ /books/book[@year and @pages]/*	To obtain the books which have publishing year and number of pages
⊙ /books/book/author/@*	To obtain all the attributes of the authors
⊙ //book	To obtain all the books included in the *XML* document

XPath Query	Meaning
⊙ //book[review="Very good"]/author	To obtain all the authors of books with a very good review
⊙ //@year	To obtain all the years occurring in the XML document
⊙ /books/*/author	To obtain all the authors inside book records

XPath Query	Meaning
⊙ /books/book[review="Good"]/ author[name="John Durant"]	To obtain all the author information whose name is John Durant and the review is good
⊙ /books[book="The first book"]/book [@year=2003 and review="Good"] /author[name="Benz"]/../..	To obtain the books of the year 2003 and good review whose author is Benz
⊙ /books/book/text()	To obtain the books with textual information
⊙ /books/book[author/name]/title	To obtain the book titles whenever the books have author name
⊙ /books/(book | book2)/(review2 | review)	To obtain the reviews of the two kinds of books
⊙ /books/book/(author | title)	To obtain the book authors and titles
⊙ /books/(book | book2)//text()	To obtain the textual information from the two kinds of books
⊙ //@*	To obtain all the attributes of the document
⊙ /*/*/title	To obtain the titles that are at 3rd level
⊙ /*/*//*	To obtain all the elements and their nested from the 3rd level
⊙ /*/book2/*	To obtain all information from book2 at 2nd level
⊙ //*//author/..	To obtain the records containing author information from the 1st level

Εικόνα 33-examples

Table 1. *A small XML document*

```
<books year="2006">
    A book collection
    <book> empty</book>
    <book year="2003" pages="984">
        The first book
        <author english="yes" spanish="yes">
            Benz
            <name> Brian</name>
        </author>
        <author> John Durant</author>
        <author> John Durant</author>
        <title> XML Programming Bible</title>
        <review> Good</review>
    </book>
    <book year="2002">
        The second book
        <author> Dino Esposito</author>
        <title> Applied XML Programming for Microsoft .NET</title>
        <review> Good</review>
    </book>
    <book>
    The third book
        <author> Apt, Krzystof R.</author>
        <title> The Logic Programming Paradigm and Prolog</title>
        <review> Very good</review>
    </book>
    <book year="1994" pages="560">
        The fourth book
        <author english="yes" spanish="no">
            Leon Sterling
        </author>
        <author> Ehud Shapiro</author>
        <title> The Art of Prolog</title>
        <review> Very good</review>
    </book>
    <book2 year="2001">
        The fifth book
        <author english="yes">
            Elliotte Rusty Harold
        </author>
        <title> XML Bible</title>
        <review2> Good</review2>
    </book2>
    <book year="2003" pages="984">
        The first book
        <author english="yes" spanish="yes">
            Benz
            <name2> Brian</name2>
            <firstname>
            Brian
            <lastname> Benz</lastname>
            <others> no more</others>
            </firstname>
        </author>
        <author> John Durant</author>
        <author> John Durant</author>
        <title> XML Programming Bible</title>
        <review> Very good 2</review>
    </book>
</books>
```

Εικόνα 34-ένα μικρό έγγραφο XML

4.10 Μελλοντικές προεκτάσεις

Η μία προέκταση που θα μπορούσε να δωθεί είναι η επέκταση του X Path σε μια πιο ισχυρή γλώσσα ερωτημάτων όπως η Xquery. Η δεύτερη προέκταση θα ήταν η χρήση του λογικού προγραμματισμού σαν μηχανή εξαγωγής συμπερασμάτων του Σημασιολογικού Ιστού, με την εισαγωγή των εγγράφων σε RDF ή OWL προδιαγραφές.

Κεφάλαιο 5ο

SWI-Prolog και Διαδίκτυο

Το Web είναι ένα μέρος το οποίο προσφέρει νέες ευκαιρίες από άποψη τεχνητής νοημοσύνης, επεξεργασία φυσικής γλώσσας και λογικού προγραμματισμού. Η Εξαγωγή πληροφοριών από το Web, με reasoning στις Web εφαρμογές και στο Σημασιολογικό Ιστο, είναι μερικά παραδείγματα. Έχει αναπτυχθεί η Prolog στις εργασίες του Παγκοσμίου Ιστού για ήδη μεγάλο χρονικό διάστημα. Το μεγαλύτερο μέρος της ανάπτυξης στην SWI-Prolog πραγματοποιείται στο πλαίσιο των έργων που απαιτούν νέα χαρακτηριστικά. Επίσης, το σύστημα και οι βιβλιοθήκες του παρέχουν εκτεταμένη υποστήριξη για Web programming.

Στις δύο απόψεις που υπάρχουν για την Prolog, όσων αφορά τον Πακόσμιο Ιστό, η μία άποψη είναι ότι η Prolog ενεργεί ως ένα ενσωματωμένο συστατικό σε ένα περιβάλλον Web. Σε αυτό το ρόλο, παρέχει γενικά καθήκοντα Reasoning, όπως η έρευνα ή διαμόρφωση εντός περιορισμούς.

Εναλλακτικά, η ίδια η Prolog μπορεί να λειτουργήσει ως αυτόνομος διακομιστής HTTP. Σε αυτή την περίπτωση, είναι ένα στοιχείο που μπορεί να είναι μέρος οποιουδήποτε από τα στρώματα της αρχιτεκτονικής τριών επιπέδων για εφαρμογές Web. Η τελευταία αυτή άποψη είναι πιο ελκυστική. Χρησιμοποιώντας HTTP και XML μέσω HTTP, η υπηρεσία απομονώνεται με τη χρήση τυποποιημένων πρωτοκόλλων παρά αποκλειστική επικοινωνία. Τρέχει ως αυτόνομη εφαρμογή, και λόγω της φύσης της Prolog μπορεί να διατηρηθεί πολύ πιο εύκολα από ό,τι η άλλη περίπτωση.

Κατά την χρήση της Prolog σε μια εφαρμογή Web, χρησιμοποιώντας το πρωτόκολλο HTTP, πρέπει όχι μόνο να εφαρμόστει το πρωτόκολλο μεταφοράς HTTP αλλά και η υποστήριξη ανάλυσης, που παράγει τους σημαντικούς τύπους εγγράφων που χρησιμοποιούνται στο διαδίκτυο. Χρησιμοποιείται ευρέως ανοικτά πρότυπα, που υποστηρίζουν αυτούς τους τύπους εγγράφων. Είναι επίσης πολύτιμη εκτός του πλαισίου των εφαρμογών Web.

5.1 Αναλύοντας και εκπροσωπόντας XML και HTML αρχεία.

⟨*document*⟩	::=	list-of ⟨*content*⟩
⟨*content*⟩	::=	⟨*element*⟩ | ⟨*pi*⟩ | ⟨*cdata*⟩ | ⟨*sdata*⟩ | ⟨*ndata*⟩
⟨*element*⟩	::=	element(⟨*tag*⟩, list-of ⟨*attribute*⟩, list-of ⟨*content*⟩)
⟨*attribute*⟩	::=	⟨*name*⟩ = ⟨*value*⟩
⟨*pi*⟩	::=	pi(⟨*atom*⟩)
⟨*sdata*⟩	::=	sdata(⟨*atom*⟩)
⟨*ndata*⟩	::=	ndata(⟨*atom*⟩)
⟨*cdata*⟩, ⟨*name*⟩	::=	⟨*atom*⟩
⟨*value*⟩	::=	⟨*svalue*⟩ | list-of ⟨*svalue*⟩
⟨*svalue*⟩	::=	⟨*atom*⟩ | ⟨*number*⟩

Fig. 1. SGML/XML tree representation in Prolog. The notation list-of ⟨*x*⟩ describes a Prolog list of terms of type ⟨*x*⟩.

Εικόνα 35-λίστα όρων Prolog

Η XML είναι ένας εξορθολογισμός της SGML χρησιμοποιώντας το ίδιο δέντρο ως μοντέλο, αλλά αφαιρώντας πολλές σπάνια χρησιμοποιούμενες λειτουργίες, καθώς και συντμήσεις που εισήχθησαν στην SGML, για να κάνουν το κέρδοw ευκολότερο για την πληκτρολόγηση και το διάβασμα για τους ανθρώπους. Τα έγγραφα XML χρησιμοποιούνται για να αντιπροσωπεύσουν το κείμενο χρησιμοποιώντας προσαρμοσμένες ετικέτες που είναι προσανατολισμένες στην εφαρμογή καθώς και μια μορφή serialization για αυθαίρετη ανταλλαγή δεδομένων μεταξύ των υπολογιστών.
Το πρώτο πρόγραμμα ανάλυσης της SGML για SWI-Prolog δημιουργήθηκε βάσει του SP Parser.
Η εκπροσώπηση του κειμένου από ένα άτομο Prolog γίνεται με τη χρήση του SWI-Prolog, η οποία δεν έχει όριο μήκους για τα άτομα που μπορεί να αντιπροσωπεύσει σε ένα κείμενο Unicode. Επίσης, οι SWL-Prolog στοίβες περιορίζονται σε 128 MB η καθεμία. Μόνο με τα άτομα, η δομή του δέντρου αντιπροσωπεύεται στη στοίβα, ενώ το μεγαλύτερο μέρος των δεδομένων, είναι αποθηκευμένα στην απεριόριστη σωρό.Η Χρησιμοποίηση των καταλόγων κωδικών χαρακτήρων, είναι μια άλλη δυνατότητα που εγκρίθηκε.
Οι τιμές των Χαρακτηριστικών των ιδιοτήτων με πολλαπλές τιμές επέστρεψε ως μια λίστα Prolog. Με την SGML αυτό είναι πάντα αλήθεια, αλλά όχι με την XML. Προαιρετικά αποδίδουνται τιμές του τύπου NUMBER ή αντιστοιχίζονται με αριθμούς Prolog, αυτή η μετατροπή πάσχει από την πιθανή απώλεια πληροφοριών. Οι τιμές attributes, εκπροσωπήθηκαν ως Name=Value. Χρησιμοποιώντας τη «φράση» Name (Value) είναι μία εναλλακτική λύση. Η παράσταση αυτή, επιλέχθηκε για την ομοιότητά της, με της SGML.

→Υλοποίηση

Ο αναλυτής SWL-Prolog SGML / XML υλοποιείται ως μία C-βιβλιοθήκη που έχει χτιστεί από το μηδέν για να δημιουργήσει ένα ελαφρύ πρόγραμμα ανάλυσης. Το πρόγραμμα ανάλυσης παρέχει δύο διεπαφές.

5.1 Παραγωγή εγγράφων του Ιστού.

Υπάρχουν πολλές προσεγγίσεις για τη δημιουργία ιστοσελίδων από τα προγράμματα γενικά και από την Prolog ειδικότερα. Η επιλογή εξαρτάται από πολλούς παράγοντες.
• Πόσο από το έγγραφο αυτό δημιουργείται από δυναμικά δεδομένα και πόσο από στατικά.
• Για τις σελίδες του προγράμματος που δημιουργούνται, μπορούμε να επιλέξουμε μεταξύ άμεσης εκτύπωσης και γενικευμένης, χρησιμοποιώντας συντακτικό μίας native γλώσσας.
• Τα έγγραφα που περιέχουν σημαντικό στατικό μέρος είναι μία καλύτερη περίπτωση στη γλώσσα σήμανσης.
• Ο μετασχηματισμός σελίδας πραγματοποιείται με την ανάλυση του πρωτότυπου εγγράφου σε δέντρο, με την εκπροσώπησή της, τη διαχείριση του δέντρου και την εγγραφή ενός νέου εγγράφου από το δέντρο.
Η Διαχείριση του κειμένου πηγής δεν είναι αξιόπιστη, καθώς λόγω κωδικοποίησης χαρακτήρων, χρήσης οντοτήτων και συντμήσεων HTML, υπάρχουν πολλά διαφορετικά source κείμενα που αντιπροσωπεύουν το ίδιο δέντρο.

5.2 Δημιουργία εγγράφων με τη χρήση DCG.

Η παραδοσιακή μέθοδος για τη δημιουργία εγγράφων του Ιστού, χρησιμοποιεί ρουτίνες εκτύπωσης, Αν και απλή, η προσέγγιση αυτή έχει σοβαρά μειονεκτήματα από την άποψη της μηχανικής λογισμικού. Εναλλακτικά, μπορούμε να παράγουμε έναν όρο DOM, για να δημιουργήσουμε το HTML έγγραφο.
Η Prolog τους επιτρέπει να χτιστούν από σκελετούς που περιέχουν μεταβλητές. Η προσέγγιση αυτή λαμβάνεται από τοPiLLoW, για τον έλεγχο της πολυπλοκότητας.Το αποτέλεσμα δεν είναι το βέλτιστο, λόγω της αφύσικης σειράς των δηλώσεων. Το PiLLoW υπερνικά εν μέρει αυτό το κενό με τον καθορισμό ενός μεγάλου αριθμού «όρων χρησιμότητας».
Ο κανόνας αυτός μεταφράζεται με δέντρα σε μια λίστα υψηλού επιπέδου HTML/XML εντολών που διανέμονται σε html_print.1.

```
        ...,
        mkthumbnail(URL, Caption, ThumbNail),
        output_html([ h1("Photo gallery"),
                      ThumbNail
                    ]).

mkthumbnail(URL, Caption, Term) :-
        Term = table([ tr(td([halign=center], img([src=URL],[]))),
                       tr(td([halign=center], Caption))
                     ])
```

Fig. 2. Building PiLLoW terms.

```
affiliation_table :-
        findall(Name-Aff, affiliation(Name, Aff), Pairs0),
        keysort(Pairs0, Pairs),
        reply_page(table([border(2),align(center)],
                         [ tr([th('Name'), th('Affiliation')])
                         | \affiliations(Pairs)
                         ])).

affiliations([]) --> [].
affiliations([H|T]) --> affiliation(H), affiliations(T).

affiliation(Name-Aff) --> html(tr(td(Name), td(Aff))).

% database
affiliation(wielemaker, uva).
affiliation(huang, vu).
affiliation('van der meij', vu).

% Page template
reply_page(Term) :-
        format('Content-type: text/html~n~n'),
        phrase(html(Term), Tokens),
        print_html(Tokens).
```

Fig. 3. Library html_write.pl in action.

Εικόνα 36-PiLLoW όροι και εντολές

Είναι επίσης δυνατό να επεκταθεί αναδρομικά το παραγόμενο δέντρο και να επικυρώσει το HTML DTD σε compile χρόνο, ακόμη και να εισάγει ετικέτες που έχουν παραλειφθεί κατά τη μεταγλώττιση.

5.3 Σύγκριση με PiLLoW.

Η βιβλιοθήκη PiLLoW, είναι ένα καθιερωμένο πλαίσιο για τον προγραμματισμό στο Web με βάση την Prolog.
Η PiLLoW, ορίζει την html2terms /2 μετατροπή μεταξύ μιας συμβολοσειράς HTML κι ενός εγγράφου που αντιπροσωπεύεται ως Herbrand όρος.
• Η PiLLoW, δημιουργεί ένα έγγραφο HTML από έναν όρο Herbrand που έχει περάσει σαν html2terms/2. Οι σύνθετοι όροι που αποτελούνται από μερικούς όρους πέρασαν μεταβλητές Prolog. Χρησιμοποιούμε DCGs, το οποίο καθιστά επιφανείς τους όροους που αναφέρονται απευθείας σε HTML στοιχεία και τα οποία λειτουργούν ως ένα «macro».
• Ο αναλυτής PiLLoW δεν δημιουργεί το SGML έγγραφο δέντρο. Ως αποτέλεσμα, τα έγγραφα HTML, διαφέρουν μόνο σε παρελειπόμενες ετικέτες και προεπιλεγμένα ή όχι χαρακτηριστικά περιλαμβάνονται στην πηγή και παράγουν διαφορετικούς όρους.

5.4 RDF έγγραφα.

Ενώ το μοντέλο δεδομένων της HTML και XML είναι μια δομή δέντρου με τα χαρακτηριστικά της, το μοντέλο δεδομένων της γλώσσας SW RDF8 αποτελείται από τριπλέτες {υποκείμενο, κατηγόρημα, αντικείμενο}

```
[env(table, [], [tr$[], td$[], "Hello"])]

[element(table, [],
        [ element(tbody, [],
                [ element(tr, [],
                        [ element(td, [rowspan='1', colspan='1'],
                                ['Hello'])])])])]
```

Fig. 5. Term representations for <table><tr><td>Hello</table> in PiLLoW (top) and our parser (bottom). Our parser completes the tr and td environments, inserts the omitted tbody element and inserts the defaults for the rowspan and colspan attributes.

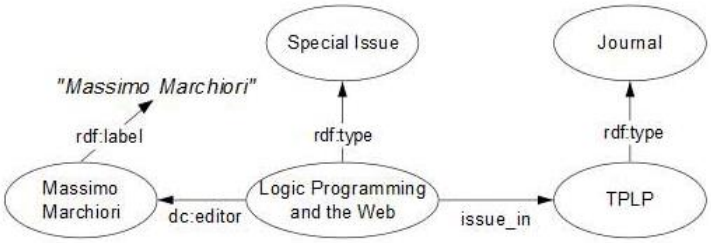


Fig. 6. Sample RDF graph. Ellipses are vertices representing URIs. Quoted text is a literal. Edges are labelled with URIs.

Εικόνα 37-RDF graph

Δεδομένου ότι υπάρχουν πολλές XML αναπαραστάσεις δέντρων για το ίδιο τριπλό σετ RDF εγγράφων, δεν μπορούν να υποβληθούν σε επεξεργασία στο επίπεδο του XML-DOM.

5.5 Είσοδος και έξοδος των αρχείων RDF.

Ο RDF / XML parser λαμβάνει χώρα σαν μία Prolog βιβλιοθήκη στην κορυφή του XML parser Έχει δύο διεπαφές.
Έχουμε δύο βιβλιοθήκες για τη σύνταξη RDF/XML. Ένα, rdf_write_xml (+ Stream. +triples) και ένα άλλο, που ονομάζεται rdf_save/ 2 και είναι μέρος της μονάδας αποθήκευσης RDF που γράφει μια βάση δεδομένων απευθείας σε ένα αρχείο. Η πρώτη χρησιμοποιείται για την ανταλλαγή υπολογιστικών γραφημάτων για εξωτερικά προγράμματα για την επικοινωνία του δικτύου, ενώ η δεύτερη, χρησιμοποιείται για να αποθηκεύσουμε τα τροποποιημένα γραφήματα πίσω στο αρχείο.
Η δημιουργία ενός προσωρινού γραφήματος σε μια βάση δεδομένων απαιτεί δυνητικά πολύ μνήμη, ενώ οι γραφικές παραστάσεις της βάσης δεδομένων σε μια λίστα μπορεί να μην ταιριάζουν σε στοίβες Prolog και είναι επίσης σημαντικά πιο αργό από ό, τι μια άμεση εγγραφή.

5.6 Αποθήκευση του RDF.

Αν υποθέσουμε ότι η case «συγκομιδή» χρήσης, θα πρέπει να εφαρμόσει ένα RDF κατηγόρημα, το να έχουμε κάνει indexing τη βάση δεδομένων είναι ζωτικής σημασίας για την καλή απόδοση. Διατυπώθηκαν οι ακόλουθες απαιτήσεις.

• Μέχρι τουλάχιστον 10 εκατομμύρια τριάδες σε 32-bit υλικό.
• Γρήγορη διάσχιση γραφήματος, χρησιμοποιώντας οποιοδήποτε σχέδιο συγκεκριμενοποίησης.
• Διάκριση πεζών-κεφαλαίων
• Αναζήτηση προθέματος για την ολοκλήρωση στο περιβάλλον εργασίας χρήστη.
• Ψάχνοντας για τις λέξεις που εμφανίζονται στα literals.
• Multi-threaded πρόσβαση με βάση κλειδαριών ανάγνωσης / εγγραφής.
• Διαχείριση συναλλαγών .
• Διατήρηση των πληροφοριών πηγής, ώστε να ενημερώνουμε, να αποθηκεύουμε ή να αφαιρέσουμε τα στοιχεία βάσει της πηγής τους.
• Γρήγορη Φόρτωση / Αποθήκευση της τρέχουσας κατάστασης.

Table 1. *Call statistics on a real-world system*

Index pattern			Calls
−	−	−	58
+	−	−	253,554
−	+	−	62
+	+	−	23,292,353
−	−	+	633,733
−	+	+	7,807,846
+	+	+	26,969,003

Εικόνα 38-call statistics

5.7 Συλλογιστική με έγγραφα RDF (Reasoning)

Υπάρχουν δύο προσεγγίσεις για τη συλλογιστική στην κορυφή των κατηγορημάτων RDF για περισσότερες γλώσσες υψηλού επιπέδου. Μια προσέγγιση λαμβάνεται από το σύστημα ερωτημάτων SeRQL και βασίζεται στην παρατήρηση ότι οι γλώσσες αυτές θεσπίζουν κανόνες που συμπεραίνουν νέα triples από το σύνολο των γνωστών triples.

```
% triples                % entailment interface % RDFS interface
mary type woman.         ?-rdf(mary,type, X).   ?-rdfs_individual_of(mary, X).
woman type Class.        X = woman;             X = woman;
woman subClassOf human.  X = human;             X = human;
human type Class.        No                     No
```

Fig. 10. Different interface styles for RDFS.

Εικόνα 39-διαφορετικά RDF interface styles

Το επαγωγικό κλείσιμο μπορεί να πραγματοποιηθεί με τη χρήση πλήρως και προς τα εμπρός συλλογιστικής, συνάγοντας νέα triples, έως ότου αυτό δεν είναι πλέον δυνατό. Επίσης, από ένα συνδυασμό προς τα πίσω και προς τα εμπρός συλλογιστικής.
Μια εναλλακτική προσέγγιση είναι να εξεταστεί σε RDFS ή OWL, σε εννοιολογικό επίπεδο και να εισαχθεί ένα σύνολο κατηγορημάτων απόαυτό το επίπεδο.

5.8 Υποστήριξη HTTP.

Η HTTP, ή ένα πρωτόκολλο μεταφοράς υπερκειμένου, είναι το κλειδί του W3C πρότυπο πρωτόκολλο για την ανταλλαγή εγγράφων του Ιστού.
Όλοι οι browsers και οι διακομιστές Web την εφαρμόζουν. Η αρχική έκδοση του πρωτοκόλλου ήταν πολύ απλή. Το αίτημα του πελάτη αποτελείται από μία μόνο γραμμή της μορφής (δράση} (path}, ο διακομιστής απαντά με το ζητούμενο έγγραφο και κλείνει τη σύνδεση. Η έκδοση του πρωτοκόλλου είναι πιο περίπλοκη, παρέχοντας επιπλέον ζεύγη ονόματος-τιμής στην αίτηση, καθώς και ως απάντηση, χαρακτηριστικά για την κατάσταση, όπως ο χρόνος τροποποίησης, η μεταφορά επιμέρους εγγράφων, κ.λπ.

5.9 Βιβλιοθήκες HTTP Clients.

Υποστηρίζονται δύο πελάτες. Ο πρώτος είναι ένας ελαφρύς πελάτης που υποστηρίζει μόνο τη μέθοδο GET HTTP μέσω του $\mathbf{http\_open}(+URL, -Stream, +Options).$ Με επιλογές επιτρέπεται η ρύθμιση ενός χρονικού ορίου καθώς και η λήψη πληροφοριών από την κεφαλίδα απάντησης, όπως το μέγεθος του εγγράφου.

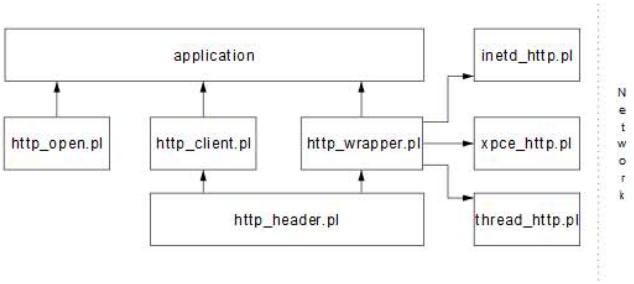


Fig. 11. Module dependencies of the HTTP library.

```
?- use_module(library('http/http_client')).
?- use_module(library('http/http_sgml_plugin')).

?- http_get('http://www.swi-prolog.org/', DOM, []).
DOM = [element(html, [version='-//W3C//DTD HTML 4.0 Transitional//EN'],
                     [element(head, [],
                              [element(title, [],
                                       ['SWI-Prolog\'s Home']), ...
```

Fig. 12. Fetching an HTML document.

Εικόνα 40-HTTP/HTML

Οι μη σωστές απαντήσεις αντιστοιχίζονται με μία εξαίρεση Prolog. Μετά την ανάγνωση του εγγράφου, ο χρήστης πρέπει να κλείσει την επιστροφή, χρησιμοποιώντας το Prolog/1 κατηγόρημα. Αυτό το κατηγόρημα καθιστά την πρόσβαση σε μια πηγή HTTP τόσο απλή όπως την πρόσβαση σε ένα τοπικό αρχείο. Η δεύτερη βιβλιοθήκη. που ονομάζεται http_client.pl, παρέχει υποστήριξη για HTTP POST και ένα plug-in interface που επιτρέπει την εγκατάσταση χειριστών για έγγραφα που προσδιορίζονται από Ml ΜΕ-τύπους.
Τόσο οι PiLLoW και η ECLiPSe προσεγγίσεις επιστρέφουν το περιεχόμενο ως μια σειρά εγγράφων.

5.10 Η Βιβλιοθήκη HTTP Server.

Για να απλοποιήσει την επαναχρησιμοποίηση του κώδικα της εφαρμογής και για να καταστεί δυνατή η χρήση του διακομιστή χωρίς να υποπέσει σε μεγάλες υποδομές, υιοθετήθηκε η στρατηγική απάντησης του πρωτοκόλλου CGI, όπου ο χειριστής γράφει μια σελίδα που αποτελείται από μια κεφαλίδα HTTP που ακολουθείται από το περιεχόμενο του εγγράφου.
Υπάρχει η λογική του χειρισμού μιας ενιαίας αίτησης HTTP να δοθεί ένα κατηγόρημα αξιοποίησης από το χειριστή, ένα εισόδου και ένα εξόδου με χρήση

http_reply(+*Reply*, +*Stream*, +*Header*).

Απαντήσεις εκτός από «200 OK» παράγονται χρησιμοποιώντας μια εξαίρεση Prolog.
Η αποτυχία του χειριστή δημιουργεί μια απάντηση '404 existence error', ενώ άλλες εξαιρέσεις από αυτά που περιγράφονται δημιουργούν '500 Server error' απάντηση.

5.11 Εκτίμηση

Η υποδομή server που παρουσιάστηκε, χρησιμοποιείται σήμερα από πολλά εσωτερικά και εξωτερικά projects. Η κωδικοποίηση ενός server είναι παρόμοια με το γράψιμο CGI handlers και τρέχει στη διαδραστική διαδικασία Prolog. Είναι πολύ πιο εύκολος τρόπος για τον εντοπισμό σφαλμάτων. Η Prolog μπορεί να επαναφόρτωσει πηγαία αρχεία στο σύστημα που λειτουργεί, οι χειριστές (handlers), μπορούν να ενημερωθούν, ενώ ο διακομιστής τρέχει.

Table 2. *HTTP performance executing a trivial query 10,000 times**.

Connection	Elapsed	Server CPU	Client CPU
Close	20.84	11.70	7.48
Keep-Alive	16.23	8.69	6.73

*Times are in seconds. Localhost, dual AMD 1600+ running SuSE Linux 10.0

Εικόνα 41-http performance

Οι χειριστές τρέχουν κατά την ενημέρωση και είναι πιθανό να «πεθάνουν» σε μια εξαίρεση. Σχεδιάζεται η επίλυση αυτού με την εισαγωγή κλειδαριών ανάγνωσης / εγγραφής.

➔ Extensions στην γλώσσα Prolog.

θεωρείται ζωτικής σημασίας για κλιμακωτή και άνετη ανάπτυξη της Prolog στον Web κόσμο.

5.12 Multi-threading

➔ Multi-threading

Η συνδρομικότητα είναι αναγκαία για τις εφαρμογές για τους εξής λόγους:

• Οι καθυστερήσεις του δικτύου μπορεί να κάνουν στην επικοινωνία να πάρει πολύ χρόνο. Δεν είναι αποδεκτό εάν τέτοια περιστατικά δυσκολεύουν την πρόσβαση για άλλους πελάτες. Αυτό μπορεί να επιτευχθεί με τη χρήση πολυπλεξίας 1/0
•Οι CPU-intensive υπηρεσίες, πρέπει να είναι σε θέση να αναπτύξουν πολλαπλές CPU. Αυτό μπορεί να επιτευχθεί χρησιμοποιώντας πολλαπλά instances της υπηρεσίας και το φορτίο εξισορρόπησης ή έναν διακομιστή που εκτελείται σε υλικό πολλαπλών επεξεργαστών ή ένα συνδυασμό αυτών.

Καμία από τις παραπάνω απαιτήσεις δεν απαιτεί υποστήριξη multi-threading σε Prolog. Έχει προστεθεί multi-threading, γιατί επιλύει τα προβλήματα που αναφέρθηκαν για εφαρμογές μεσαίας κλίμακας, ενώ απλοποιούν σε μεγάλο βαθμό την ανάπτυξη και τον εντοπισμό σφαλμάτων.

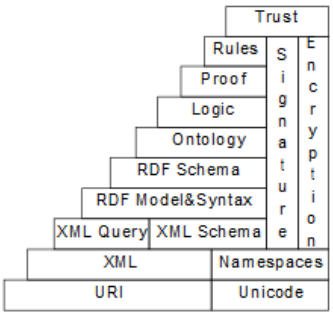


Fig. 16. The Semantic Web layer cake by Tim Burners Lee.

Εικόνα 42-semantic web layer

➔ Τα άτομα (atoms) και υποστήριξη Unicode.

Το Unicode, είναι ένα σύστημα κωδικοποίησης χαρακτήρων, που αναθέτει μοναδικούς ακεραίου (σημεία κώδικα) σε όλους τους χαρακτήρες από όλα τα scripts.

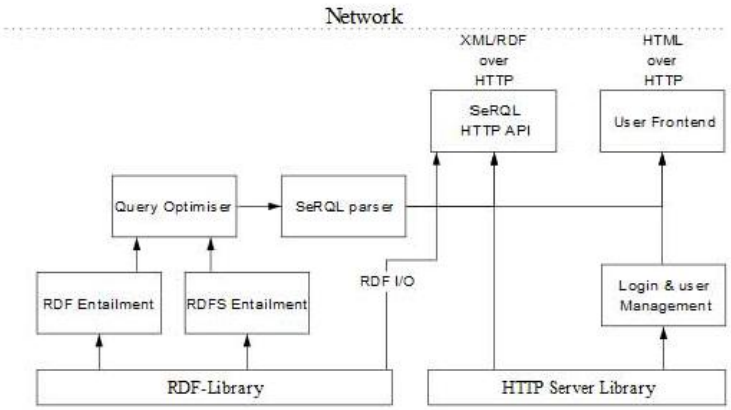


Fig. 17. Module dependencies of the SeRQL system. Arrows denote 'imports from' relations.

Εικόνα 43-ατομικές εξαρτήσεις

5.13 Case study: Μια γλώσσα SW query

Σε αυτή τη μελέτη περίπτωσης, αναφέρεται το SWI-Prolog SeRQL implementation. Η SeRQL είναι μια γλώσσα επερωτήσεων για RDF και χρησιμοποιεί το HTTP ως πρωτόκολλο πρόσβασης. Το Sesame αποτελείται από μια εφαρμογή του διακομιστή ως ένα servlet Java και μια βιβλιοθήκη Java client.
Με την εφαρμογή ενός συμβατού framework, κάναμε τη βασισμένη σε Prolog RDF αποθήκευση και τη συλλογιστική μας (reasoning) να είναι διαθέσιμη για java clients.
Η πρόσβαση στην HTTP αποτελείται από δύο μέρη. Η ανθρωποκεντρική πλευρά, αποτελείται από σελίδες HTML με τα έντυπα για διαχείριση του server , καθώς και στατιστικά στοιχεία που αφορούν τη φόρτωση και εκφόρτωση των εγγράφων και το τρέξιμο SeRQL ερωτημάτων διαδραστικά, παρουσιάζοντας το αποτέλεσμα ως πίνακα HTML. Οι δυναμικές σελίδες δημιουργούνται με τη χρήση της βιβλιοθήκης htmlwrite.pl. Οι στατικές σελίδες, εξυπηρετούνται από αρχεία HTML μέσω διακομιστή Prolog.

Το σύστημα ξέρει για τις διάφορες RDF μορφές εισόδου και εξόδου. Η μορφή εξόδου HTML χρησιμοποιεί τη βιβλιοθήκη htmlwrite. pl. Το RDF/XML Format, χρησιμοποιεί το **rdf_write_xml/2**

5.14 Case study: XDIG

Μετά την μελέτη για το πώς η SWI-Prolog χρησιμοποιείται για ένα ερώτημα σε ένα σύστημα RDF, δηλαδή, μια διαχείριση μεταδεδομένων και το σύστημα συλλογισμού του, εδώ περιγράφεται ένα Prolog-powered σύστημα για τη διαχείριση της οντολογίας και τις Λογικές Περιγραφής (DL). Το DL έχει επηρεάσει σε μεγάλο βαθμό το σχεδιασμό της W3C γλώσσας οντολογιών OWL. Η DL community, ονομάζεται DIG (DL ομάδας υλοποίησης). Πολλοί reasoners DL, όπως ο Racer (Haarslev και Moller 2001) και FACT (Horrocks 1999) υποστηρίζει τη DIG, επιτρέποντας την κατασκευή φορητών και επαναχρησιμοποιήσιμων στοιχείων DL ή επεκτάσεις τους.

```
<equalc>
  <catom name='mad+cow'/>
  <and>
    <catom name='cow'/>
    <some>
      <ratom name='eats'/>
      <and>
        <catom name='brain'/>
        <some>
          <ratom name='part+of'/>
          <catom name='sheep'/>
</some></and></some></and></equalc>
```

Fig. 20. a DIG statement on MadCow.

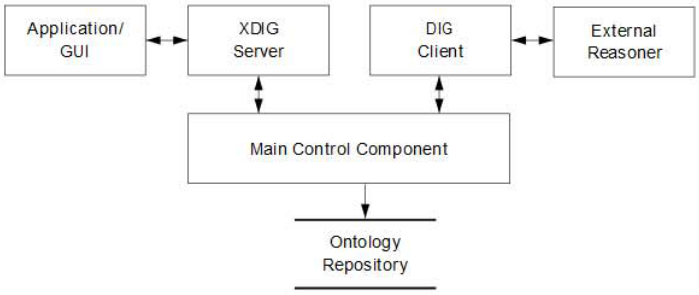


Fig. 21. Architecture of XDIG.

Εικόνα 44-δήλωση και αρχιτεκτονική XDIG

5.15 Case study: Πολύπλευρο πρόγραμμα περιήγησης στη βάση δεδομένων SW με ενσωμάτωση multiple collections.

Σε αυτή τη μελέτη, περιγράφεται ένα πιλοτικό project for the STITCH, του οποίου στόχος είναι η μελέτη και εξεύρεση λύσεων για το πρόβλημα της ένταξης ελεγχόμενων λεξιλογίων, όπως οι θησαυροί και των συστημάτων ταξινόμησης στον τομέα της Πολιτιστικής Κληρονομιάς. Αυτό το project, αποτελείται από την ένταξη δύο συλλογών - την Medieval Illuminations of the Dutch National Library και τη συλλογή "Αριστουργήματα" από το Rijksmuseum, και την ανάπτυξη ενός UI για την περιήγηση σε αυτά. Μια απαίτηση της πιλοτική αυτής προσπάθειας, είναι να χρησιμοποιήσει «πρότυπες τεχνικές SW» σε όλα

τα στάδια, έτσι ώστε να είναι σε θέση να αξιολογήσει την αξία τους. Σαφής στόχος της έρευνας ήταν να αξιολογήσει τα υπάρχοντα εργαλεία.
Το πρόβλημα μπορεί να χωριστεί σε τρία επίπεδα:

• Τη συγκέντρωση στοιχείων, δηλαδή, τα αρχεία των συλλογών και ελεγχόμενα λεξιλόγια που χρησιμοποιούν, και τη μετατροπή τους σε RDF.
• Την καθιέρωση σημασιολογικών σχέσεων μεταξύ των λεξιλόγιων που χρησιμοποιούν off-the-shelf εργαλεία χαρτογράφησης οντολογιών.
• Η οικοδόμηση ενός πρωτοτύπου UI για πρόσβαση (αναζήτηση και περιήγηση) στις ολοκληρωμένες συλλογές και πειραματισμός με διαφορετικούς τρόπους για την αποκτήση πρόσβασης, χρησιμοποιώντας ένα διακομιστή Web.

Το συγκεκριμένο (SW), είναι το μεγαλύτερο υποσύστημα που χρησιμοποιούν αυτές οι βιβλιοθήκες.

5.16 Γενικά

Παραπάνω, παρουσιάστηκαν οι βιβλιοθήκες και οι επεκτάσεις της γλώσσας Prolog. Καθώς οι βιβλιοθήκες που παρουσιάζονται καλύπτουν πολύ διαφορετικές λειτουργίες, έχουμε σύγκριση αυτής της προσέγγισης με τις σχετικές εργασίες σε όλο το έγγραφο. Αποδείχθηκε ότι η Prolog, είναι εξοπλισμένη με μια βιβλιοθήκη σε διακομιστή HTTP, βιβλιοθήκες για ανάγνωση και έγγραφα σήμανσης γραφής, multithreading, υποστήριξη σε Unicode, απεριόριστα άτομα και άτομα συλλογής των απορριμμάτων. Με αυτό τον τρόπο, γίνεται ένα ευέλικτο συστατικό σε μία αρχιτεκτονική multi-tier Server.
Στο μέλλον, η ανάπτυξη αναμένεται να επικεντρωθεί στο SW reasoning, όπως η μετάφραση των κανόνων SWRL στη λογική των προγραμμάτων. Οι μεταφράσεις αυτές θα είναι επωφελείς, για να φτάσουμε σε πιο προβλέψιμους χρόνους απόκρισης και να επιτρέπεται πιο declarative προγράμματα. Σχεδιάζεται να προστεθεί περισσότερη υποστήριξη για τη συλλογιστική OWL, ενδεχομένως, υποστηρίζοντας ζωτικής σημασίας σχέσεις για τη χαρτογράφηση της οντολογίας.

Κεφάλαιο 6ο

Αξιολόγηση και βελτιστοποίηση ερωτημάτων στον Σημασιολογικό Ιστό

Τα συστήματα οντολογιών, συνήθως παρέχουν το reasoning και υπηρεσίες ανάκτησης, που προσδιορίζουν τα στοιχεία, τα οποία ικανοποιούν τις απαίτησεις και να αντλούν γνώση χρησιμοποιώντας αξιώματα συμπερασμά των οντολογιών τους.
Στο πλαίσιο του Σημασιολογικού Ιστού, ο αριθμός των γεγονότων που προκύπτουν, μπορεί να είναι πολύ μεγάλος. Από τη μία, η ποσότητα των βασικών στοιχείων της οντολογίας μπορεί να είναι σημαντική, και από την άλλη, η Open World συλλογιστική οντολογιών Ιστού μπορεί να δώσει τεράστιο εύρος επιλογών..
Στην προσέγγισή αυτή, οι οντολογίες ορίστηκαν σαν μια αφαιρετική βάση δεδομένων που ονομάζεται deductive ontology base(DOB), που περιλαμβάνει δηλώσεις της γλώσσας οντολογίας, που αντιπροσωπεύουν τη γνώση της. Παρέχουμε μια cost-based τεχνική βελτιστοποίησης για οντολογίες Web που δημιουργούνται ως ένα DOB.

Οι παραδοσιακές τεχνικές βελτιστοποίησης επερωτήσεων για συστήματα αφαιρετικής βάσεις δεδομένων περιλαμβάνουν στρατηγικές join-ordering και τεχνικές που συνδυάζουν μια αξιολόγηση από κάτω προς τα πάνω, με την κορυφή στη διάδοση των συνδέσεων μεταβλητών ερωτημάτων.
Ωστόσο, αυτές οι βελτιστοποίησεις δεν είναι σε θέση να εκτιμήσουν το κόστος ή την πληθικότητα των δεδομένων που δεν υπάρχουν εκ των προτέρων. Έτσι προτείνεται ένα μοντέλο- υβρίδιο κόστους, που συνδυάζει δύο τεχνικές για την πληθικότητα και γαι την Εκτίμηση του κόστους: η τεχνική δειγματοληψίας και ένα μοντέλο κόστους.
Υπάρχουν και τρεις στρατηγικές αξιολόγησης .
Βασίζονται στα:
Nested-Loop, Block Nested-Loop, and Hash Join ‹

Για να υπάρξει και α εντοπιστεί ένα καλό σχέδιο αξιολόγησης, παρέχεται ένας αλγόριθμος βελτιστοποίησης δυναμικής προγραμματισμού που διατάζει τους υποστόχους σε ένα ερώτημα, λαμβάνοντας υπόψη τις εκτιμήσεις του κόστους αξιολόγησης του.
Το παρόν κείμενο και ότι περιγράφει, διαφέρει από τα άλλα συστήματα στο Σημασιολογικό Ιστό που συνδυάζουν ένα DL reasoner, με ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (DBMS), προκειμένου να επιλυθούν τα προβλήματα κλιμάκωσης και δεν αναπτύσσουν τη βελτιστοποίηση με βάση το κόστος για την παραγόμενη γνώση, δηλαδή, δεν υπάρχει εκτίμηση του κόστους των δεδομένων που δεν είναι ήδη γνωστό.
Άλλα συστήματα χρησιμοποιούν τη λογική προγραμματισμού (LP) στο λόγο σε οντολογίες μεγάλης κλίμακας.
Όλα αυτά τα συστήματα αναπτύσσουν LP atom συλλογιστική, ενώ στο μοντέλο DOB αναπτύσσουμε Datalog συλλογιστική με τις δύο έννοιες του τομέα.

6.1 Η επαγωγική βάση οντολογίας (DOB)

Σαν πρώτος ορισμός θα μπορούσε να δωθεί το εξής:
Μια ontology knowledge base 0, είναι ένα ζευγάρι $O = \langle \mathcal{F}; \mathcal{A} \rangle$, όπου F είναι ένα σύνολο περιστατικών οντολογίας που αντιπροσωπεύουν τη δομή της (τομέα) και source

annotations (individuals), και το J είναι ένα σύνολο αξιωμάτων τα οποία επιτρέπουν τη συναγωγή των νέων περιστατικών της, όσον αφορά τον τομέα και τους individuals.
Μία DOB αποτελείται από μία επεκταμμένη βάση οντολογίας (EOB) και μία intensional βάση οντολογίας (IOB).
Σαν δεύτερος Ορισμός θα μπορούσε να δωθεί το εξής:
Δεδομένης μίας ontology knowledge base, DOB είναι μια επαγωγική βάση δεδομένων η οποία αποτελείται από ένα σύνολο ενσωματωμένων EOB κατηγορημάτων, που εκπροσωπούν το F και ένα σύνολο IOB ενσωματωμένων κατηγορημάτων, που εκπροσωπούν το J, δηλαδή, που καθορίζουν τη η σημασιολογία των EOB ενσωματωμένων κατηγορημάτων.

6.2 Η αρχιτεκτονική του συστήματος DOB-S

Το DOB-S είναι ένα σύστημα που επιτρέπει σε έναν agent, να θέσει αποτελεσματικά conjunctive ερωτήματα σε ένα σύνολο οντολογιών.

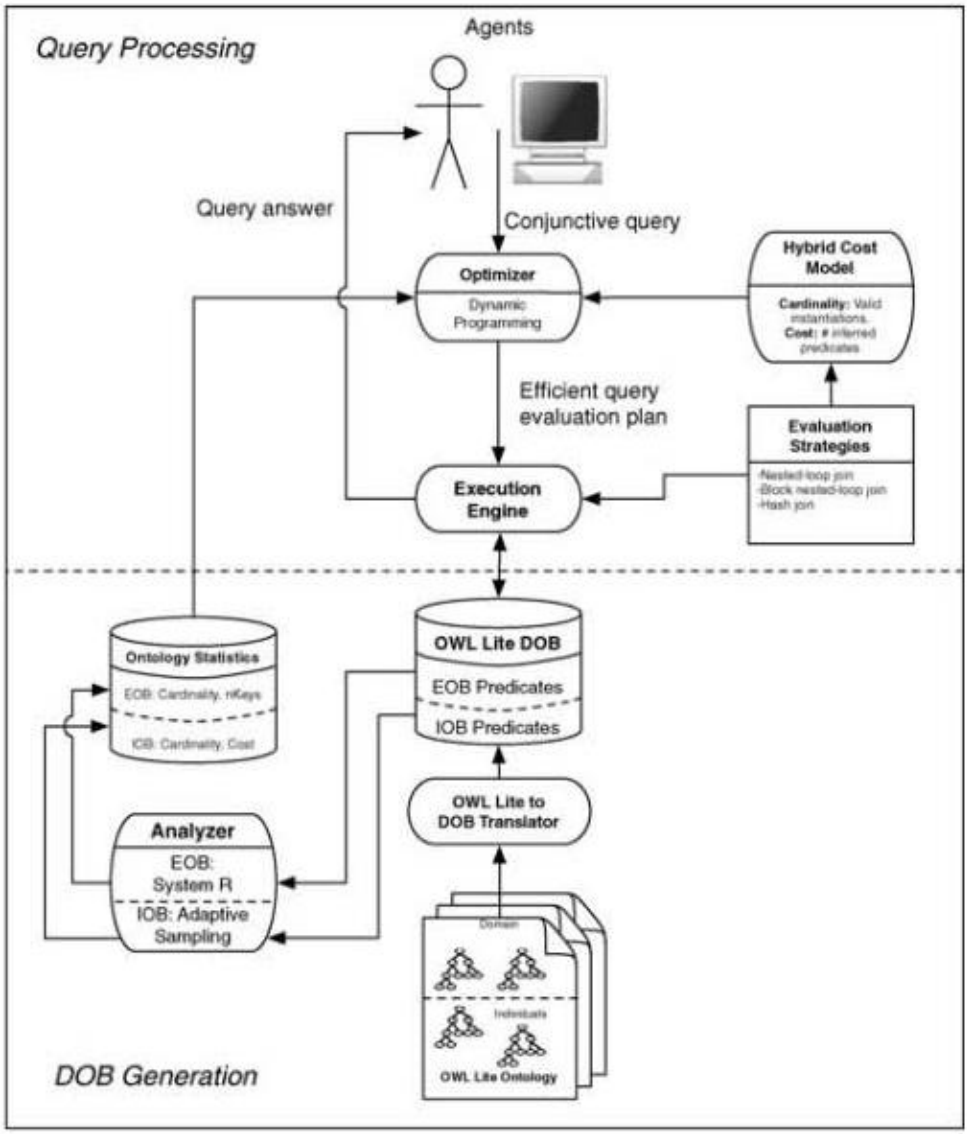


Fig. 1. DOB-S system architecture.

Εικόνα 45-αρχιτεκτονική συστήματος

Ένα υποσύνολο μίας OWL οντολογίας μεταφράζεται σε DOB χρησιμοποιώντας έναν OWL Lite σε DOB μεταφραστή. Τα EOB και IOB κατηγορήματα είναι πια αποθηκευμένα ως μία αφαιρετική βάση δεδομένων. Έπειτα, ο αναλυτής παράγει στατιστικά στοιχεία της οντολογίας για κάθε κατηγόρημα EOB, υπολογίζει τον αριθμό των γεγονότων ή τα έγκυρα στιγμιότυπα στο DOB (πληθικότητας) και τον αριθμό των διαφορετικών τιμών για κάθε ένα από τα επιχειρήματά της. Επίσης, για κάθε κατηγόρημα IOB, ένας προσαρμοστικός αλγόριθμος δειγματοληψίας εφαρμόζεται για τον υπολογισμό της πληθικότητας και τις εκτιμήσεις του κόστους.
Ένα πρόγραμμα βελτιστοποίησης γίνεται και βασίζεται σε ένα υβριδικό μοντέλο κόστους: Το προηγούμενο, χρησιμοποιεί τα EOB και IOB στατιστικά στοιχεία της οντολογίας και υπολογίζει το κόστος ενός ερωτήματος σύμφωνα με τις διαφορετικές στρατηγικές αξιολόγησης που έχουν εφαρμοστεί . Στο τέλος, μια μηχανή εκτέλεσης αξιολογεί το σχέδιο του ερωτήματος και παράγει μια απάντηση σε αυτό το ερώτημα.

6.3 OWL Lite DOB

Μια οντολογία OWL Lite περιλαμβάνει: ένα σύνολο αξιωμάτων που δίνει πληροφορίες σχετικά με τις κλάσεις και τις ιδιότητες αυτών , και μια σειρά από γεγονότα που εκπροσωπούν τα atoms της οντολογίας, τις κλάσεις στις οποίες ανήκουν και τις ιδιότητες τους.
Υπάρχουν περιορισμοί , οι οποίοι επιτρέπουν την κατασκευή των ορισμών των κλάσεων, περιορίζοντας τις τιμές των ιδιοτήτων τους και της πληθικότητας τους. Οι κλάσεις επίσης καθορίζονται μέσα από τη διασταύρωση των άλλων κλάσεων. Οι Object properties, αντιπροσωπεύουν τις binary σχέσεις, μεταξύ individuals.
Το υποσύνολο της OWL Lite ως DOB δεν περιλαμβάνει τον τομέα και τη διασταυρωμένη κατηγορία φάσματος.

6.4 Σύνταξη OWL Lite DOB

Table 1. *Some built-in EOB and IOB predicates for a subset of OWL Lite*

EOB Predicate	**Description**
`isOntology(O)`	An ontology has an Uri `O`
`isImpOntology(O1,O2)`	Ontology `O1` imports ontology `O2`
`isClass(C,O)`	`C` is a class in ontology `O`
`isOProperty(P,D,R)`	`P` is an object property with domain `D` and range `R`
`isDProperty(P,D)`	`P` is a datatype property with domain `D`
`isTransitive(P)`	`P` is a transitive property
`subClassOf(C1,C2)`	`C1` is subclass of `C2`
`AllValuesFrom(C,P,D)`	`C` has property `P` with all values in `D`
`isIndividual(I,C)`	`I` is an individual belonging to class `C`
`isStatement(I,P,J)`	`I` is an individual that has property `P` with value `J`
IOB Predicate	
`areSubClasses(C1,C2)`	`C1` are the direct and indirect subclasses of C2
`areImpOntologies(O1,O2)`	`O1` import the ontologies `O2` directly and indirectly
`areClasses(C,O)`	`C` are all the classes of an ontology and its imported ontologies `O`
`areIndividuals(I,C)`	`I` are the individuals of a class and all of its direct and indirect superclasses `C`; or `I` are the individuals that participate in a property and belong to its domain or range `C`, or are values of a property with all values in `C`

Εικόνα 46-EOB and IOB predicates

6.5 Η OWL Lite DOB σημασιολογία

Ένα μοντέλο θεωρητικής σημασιολογίας για την OWL Lite DOB έχει ως εξής:

An **Interpretation** $I = (\Delta^I; \mathscr{P}^I; \cdot^I)$ consists of

- A nonempty interpretation domain Δ^I corresponding to the union of the sets of valid URIs of ontologies, classes, object and datatype properties, and individuals. These sets are pairwise disjoint.
- A set of interpretations $\mathscr{P}^I$ of the EOB and IOB built-in predicates in Table 1.
- An interpretation function $\cdot^I$ which maps each n-ary built-in predicate $p^I \in \mathscr{P}^I$ to an n-ary relation $\bigtimes_{i=1}^{n} \Delta^I$.

Definition 6 (*Satisfiability*)
Given an OWL Lite DOB $\mathscr{D}$, an interpretation I, and a predicate $p \in \mathscr{D}$, $I \models p$ if

- p is an EOB predicate $p(t_1;:::;t_n)$ and $(t_1;:::;t_n) \in p^I$.
- p is an IOB predicate $R{:}H(\overline{X}) \leftarrow \exists \overline{Y} B(\overline{X};\overline{Y})$, and whenever I satisfies each predicate in the body B, I also satisfies the predicate in the head H.

Definition 7 (*Model*)
Given an OWL Lite DOB $\mathscr{D}$ and an interpretation I, I is a **model** of $\mathscr{D}$ if for every predicate $p \in \mathscr{D}$, $I \models p$.

Εικόνα 47-semantics

6.6 Ένα παρακινητικό παράδειγμα.

Αν υποθέσουμε ότι έχουμε μία οντολογία “Cars & Dealers”, CarsOnt και Web Source Οντολογίες τις source1 και source2. Η source1, δίνει πληροφορίες για όλα τα είδη των οχημάτων και των dealers, επίσης η source2, είναι πιο εκειδικευμένη και δίνει πληροφορίες για τα Sport Utility Vehicles (SUV’s).

Table 2. *Mapping OWL Lite subset to EOB predicates*

OWL abstract syntax	EOB Predicates
Ontology(O)	isOntology(O)
Individual(O1 value(owl : imports O2))	impOntology(O1, O2)
Ontology(O); Class(C partial Thing)	isClass(C,O)
Class(A partial C)	subClassOf(A,C)
Class(C1 partial restriction(P allValuesFrom(C2)))	allValuesFrom(C1,P,C2)
Class(A partial C1:::Cn)	subClassOf(A,C1),..., subClassOf(A,Cn)
ObjectProperty(P domain(D)), ObjectProperty(P range(R))	isOProperty(P,D,R)
DatatypeProperty(P domain(D))	isDProperty(P,D)
Property(P Transitive)	isTransitive(P)
Individual(I type(C))	isIndividual(I,C)
Individual(I value(P J))	isStatement(I,P,J)

Εικόνα 48-EOB predicates

Table 3. *Mapping OWL Lite subset inference rules to IOB predicates*

OWL lite inference rules	IOB rule definitions
If subClassOf(C1,C2) and subClassOf(C2,C3) then subClassOf(C1,C3)	areSubClasses(C1,C2):-subClassOf(C1,C2). areSubClasses(C1,C2):-subClassOf(C1,C3), areSubClasses(C3,C2).
If impOntology(O1,O2) and impOntology(O2,O3) then impOntology(O1,O3)	areImpOntologies(O1,O2):-impOntology(O1,O2). areImpOntologies(O1,O2):-impOntology(O1,O3), areImpOntologies(O3,O2).
If isClass(C1,O2) and impOntology(O1,O2) then isClass(C1,O1)	areClasses(C,O):-isClass(C,O). areClasses(C,O1):-isClass(C,O2), areImpOntologies(O1,O2).
If isSubClassOf(C1,C2) and isIndividual(I,C1) then isIndividual(I,C2)	areIndividuals(I,C):-isIndividual(I,C). areIndividuals(I,C2):-isIndividual(I,C1), areSubClasses(C1,C2).
If isStatement(I,P,J) and isOProperty(P,C,R) then isIndividual(I,C)	areIndividuals(I,C):-isOProperty(P,C,R), areStatements(I,P,J).
If isStatement(I,P,J) and isOProperty(P,D,C) then isIndividual(J,C)	areIndividuals(J,C): isOProperty(P,D,C), areStatements(I,P,J).
If isStatement(I,P,J) and isDProperty(P,C) then isIndividual(I,C)	areIndividuals(I,C):-isDProperty(P,C), areStatements(I,P,J).
If AllValues(C1,P,C) and isStatement(I,P,J) and isIndividual(I,C1) then isIndividual(J,C)	areIndividuals(J,C):-isIndividual(I,C1), allValuesFrom(C1,P,C), areStatements(I,P,J).

Εικόνα 49-IOB predicates

Table 4. *Example OWL Lite ontology*

Ontology carsOnt	Ontology source1	Ontology source2
Class (vehicle partial Thing)	imports carsOnt	imports carsOnt
Class (suv partial vehicle)		individual(s123 type(suv))
Class (car partial vehicle)		
DataProperty(price domain(vehicle))		
Class (dealer partial Thing)		
ObjectProperty(sells domain(dealer))		
ObjectProperty(sells range(vehicle))		
DataProperty(traction domain(suv))		
DataProperty(model domain(vahicle))		

Table 5. *Example DOB ontology*

EOB predicates		
isOntology(carsOnt)	isOntology(source1)	isOntology(source2)
impOntology(source1,carsOnt)	impOntology(source2,carsOnt)	isClass(vehicle,carsOnt)
isClass(vehicle,carsOnt)	isClass(dealer,carsOnt)	subClassOf(car,vehicle)
subClassOf(suv,vehicle)	isOProperty(sells,dealer,vehicle)	isDProperty(model,vehicle)
isDProperty(price,vehicle)	isDProperty(traction,suv)	isIndividual(s123,suv)

Εικόνα 50-OWL LITE & DOB ONTOLOGIES

Η απάντηση σε αυτό το ερώτημα αντιστοιχεί στο σύνολο των οντολογιών. Αν κατορθώσουμε να αντιστρέψουμε τη σειρά, έχουμε ένα ισοδύναμο ερώτημα q.:

```
q'(O):-isDProperty(traction,C),areClasses(C,O).
```

Τα στατιστικά στοιχεία σχετικά με το μέγεθος και τις τιμές του κατηγορήματος EOB μπορούν να υπολογιστούν, ενώ τα στατιστικά στοιχεία για το κατηγόρημα IOB θα πρέπει να να υπολογίζεται κια δεν είναι γνωστό εκ των προτέρων. Μόλις προσδιοριστεί το κόστος του κάθε ερωτήματος, φτιάχνεται η σειρά για τη βελτιστοποίηση.

6.7 DOB υβριδικό μοντέλο κόστους

Η διαδικασία απάντησης σε ένα ερώτημα στηρίζεται στην συναγωγή πραγματικών περιστατικών κατηγορήματα στο DOB. Το Μετρικό του κόστους μας εστιάζεται στον αριθμό των ενδιάμεσων στοιχείων που πρέπει να έχουμε, προκειμένου να δοθεί απάντηση στο ερώτημα. Ο στόχος είναι να βρεθεί μια σειρά για τα κατηγορήματα στο σώμα του ερωτήματος, ώστε ο αριθμός των ενδιάμεσων γεγονότων να μειωθεί. Έτσι φτιάχνεται μία join-ordering στρατηγική βελτιστοποίησης στο System R χρησιμοποιώντας Datalog-relational ισοδυναμίες. Για την εκτίμηση της πληθικότητας και το κόστος αξιολόγησης των intensional κατηγορημάτων, έχει εφαρμοστεί μια προσαρμοστική τεχνική δειγματοληψίας.

Έτσι, προτείνεται ένα υβριδικό μοντέλο κόστους που συνδυάζει την προσαρμοστική τεχνική δειγματοληψίας και τα παραδοσιακά μοντέλα του σχεσιακού κόστους.

6.8 Adaptive τεχνική δειγματοληψίας

Η τεχνική δειγματοληψίας που χρησιμποιείται, βασίζεται στην προσαρμοστική μέθοδο δειγματοληψίας που προτείνει η Lipton, Naughton, και η Schneider.
Αυτή η τεχνική προϋποθέτει ότι υπάρχει ένας πληθυσμός P όλων των διαφορετικών στιγμιότυπων ενός κατηγορήματος P και ότι αυτό διαιρείται σε ν μέρη, πιθανών στιγμιότυπων ενός ή περισσοτέρων επιχειρημάτων P. Κάθε στοιχείο P σχετίζεται με το κόστος αξιολόγησής τους και πληθικότητάς τους, και ο πληθυσμός P χαρακτηρίζεται από τη στατιστική μέση τιμή και τη διασπορά.
Εκτίμηση πληθικότητας.
Για την εκτίμηση της πληθικότητας του P, θα εκτελεστεί ο προσαρμοστικός αλγόριθμος δειγματοληψίας, επιλέγοντας κάποιο επιχείρημα του P, και θα χωρίσει αυτό σύμφωνα με το επιλεγμένο επιχείρημα. Η εκτίμηση της πληθικότητας θα είναι $\mathrm{card}(P) = \overline{Y} \times n$, όπου n είναι ο αριθμός των κατατμήσεων, δηλαδή, ο αριθμός των διαφορετικών instantiations για το επιλεγμένο επιχείρημα.
Όταν εκτιμάται ο αριθμός των στοιχείων του noninstantiated P, μπορούμε να υπολογίσουμε το πλήθος των στοιχείων του συνόλου του αρχικού κατηγορήματος.

6.9 Query Optimization

Table 6. *Query Optimization Algorithm*

Algorithm dynamic programming

INPUT: Predicate: a set of predicates, $P_1, \ldots, P_n$. *OUTPUT*: OrderedPredicate: an ordering of Predicate

1. SubPaths=Predicate;
2. For i=1 to n

(a) For each solution Sub_j in SubPaths
  i For each predicate P_z in Predicate
    - If there are sideways passing variables from Sub_j to P_z, then add $Sub = Sub_j, P_z$ to NewSubPaths

(b) Remove from NewSubPaths any subpath Sub_k if there is another subpath Sub_l in NewSubPaths, such that, Sub_l and Sub_k are **equivalent**, and Sub_l is **better** than Sub_k.
(c) SubPaths=NewSubPaths
(d) Reset NewSubPaths

3. Return the path in SubPaths with lowest cost.

Εικόνα 51-Query optimization algorithm

6.10 Πειραματικά αποτελέσματα

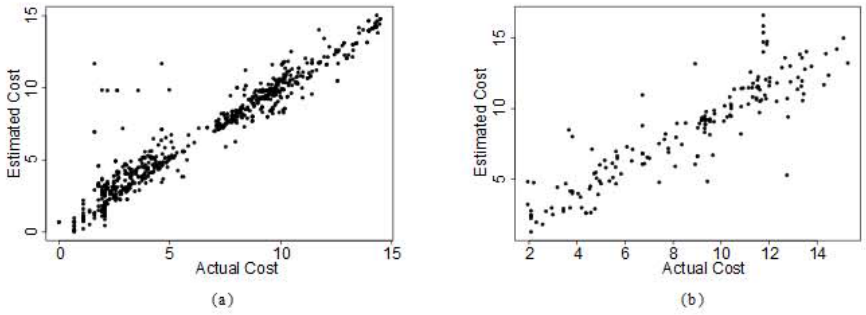
Μια πειραματική μελέτη διεξήχθη για synthetic και πραγματικού κόσμου οντολογίες.
Τα πειράματα σε synthetic οντολογίες εκτελέστηκαν σε Sun Blade 150 (650M Hz) με 1GB RAM. Τα πειράματα σε οντολογίες πραγματικού κόσμο εκτελέστηκαν σε Sun Fire V440 (1281 MHz) με 16GB RAM. Το σύστημά μας υλοποιήθηκε στο SWL-Prolog 5.6.1.
Έχουν μελετηθεί τρείς Real world οντολογίες:
→ Travel (Shell 2002)

- EHR_RM (Protege staff 1999).
- GALEN (Open Clinical Organization 2001)

Το μετρικό κόστος είναι ο αριθμός των ενδιάμεσων γεγονότων για τις synthetics και τις πραγματικού κόσμου οντολογίες και ο χρόνος αξιολόγησης των οντολογιών πραγματικού κόσμου.
Στη μέτρησή μας είχαμε το παρακάτω αποτέλεσμα όσον αφορά το κόστος.

Table 7. *Correlation values for real-world ontologies*

	Nested-loop join	Three evaluation strategies
Travel	0.96	0.94
EHR_RM	0.98	0.92



Εικόνα 52-προβλεπόμενο και πραγματικό κόστος

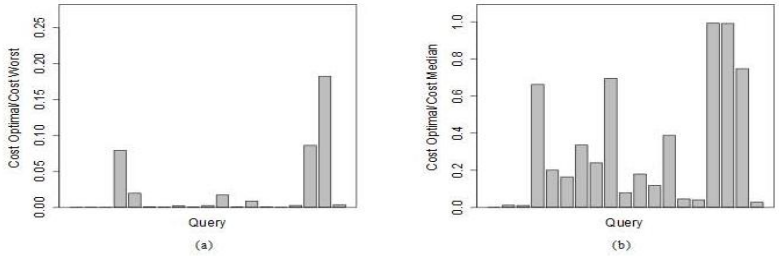


Fig. 3. (a) #Pred. optimal ordering versus #Pred. worst ordering—nested-loop-join—Synt. Ontologies; (b) #Pred. optimal ordering versus #Pred. median ordering—nested-loop-join—Synt. Ontologies.

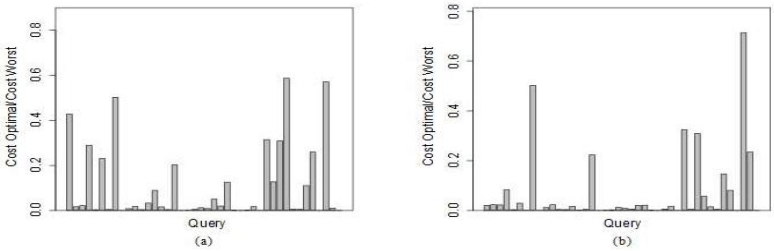
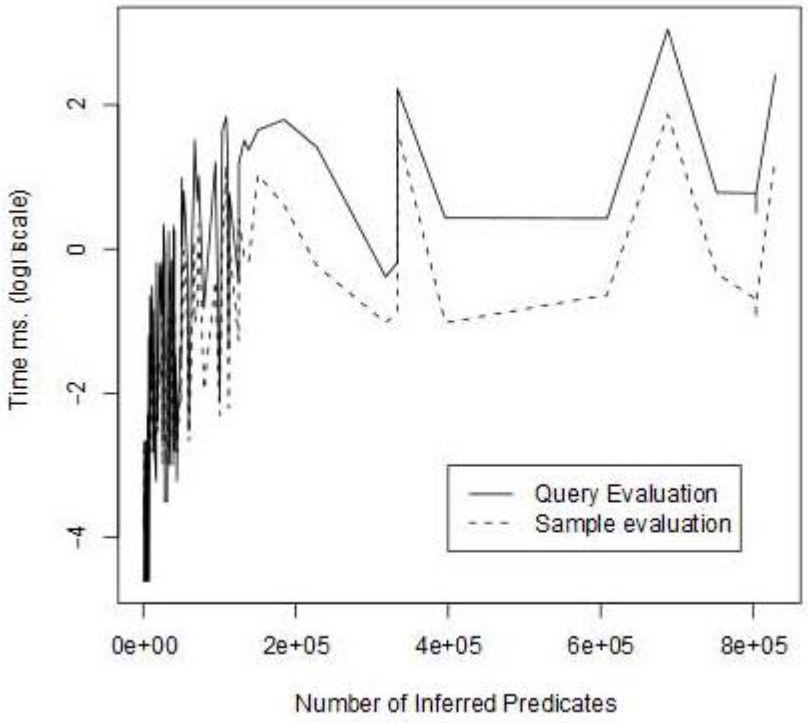


Fig. 4. (a) #Pred. optimal ordering versus #Pred. worst ordering—nested-loop-join—EHR_RM; (b) #Pred. optimal ordering versus #Pred. worst ordering—combination evaluation strategies—EHR_RM.

Εικόνα 53-query evaluation and optimization in the semantic web



Εικόνα 54-number of inferred predicates

6.11 Συμπέρασμα

Έχει αναπτυχθεί ένα μοντέλο που συνδυάζει το κόστος του συστήματος R και κάποιες επιπλέον προσαρμοστικές τεχνικές δειγματοληψίας. Προσαρμοστική τεχνική δειγματοληψίας χρησιμοποιείται για την εκτίμηση των δεδομένων που δεν υπάρχουν εκ των προτέρων. Τα δεδομένα σχετίζονται με την πληθικότητα και το κόστος των intensional κανόνων στο DOB. Τα πειραματικά αποτελέσματα δείχνουν ότι οι προτεινόμενες τεχνικές παράγουν μια σημαντική βελτίωση στο κόστος αξιολόγησης για τα ερωτήματα που θέλουμε να είναι βελτιστοποιημένα. Προς το παρόν, έχουμε την ανάπτυξη ενός υβριδικού μηχανισμού βελτιστοποίησης που έχει σαν βάση το κόστος .
Η ιδέα είναι να υπάρξει πρώτα μια καλή σειρά και στη συνέχεια να μειωθεί ο χρόνος που αξιολογείται το ερώτημα. Πειράματα δείχνουν ότι αυτή η λύση ξεπερνά την κάθε επιμέρους τεχνικής.
Πρόκειται να υπάρξουν και να εφαρμοστούν παρόμοιες τεχνικές βελτιστοποίησης για τα συνδυασμένα ερωτήματα για οντολογίες τετοιου τύπου.

Κεφάλαιο 7ο

Guarded (φυλασσόμενες) υβριδικές βάσεις γνώσης

Οι Guarded(φυλασσόμενες) υβριδικές βάσεις γνώσης, είναι ένας κομψός φορμαλισμός με βάση τα συνδυασμένα μοντέλα περιγραφικών βάσεων γνώσεων Λογικής και μη μονοτονικών λογικών προγραμμάτων.
Προτείνεται η παραλλαγή r-υβριδικών βάσεων γνώσης, που ονομάζεται g-hybrid βάση γνώσης, που δεν απαιτεί τυπικά ονόματα σχετικά με τους κανόνες, αλλά, αντίθετα, απαιτείται το πρόγραμμα να είναι καλυμμένο.
Ο Open answer set programming, συνδυάζει το λογικό προγραμματισμό και τα first-order λογικά παραδείγματα. Από το λογικό προγραμματισμό, κληρονομεί μια παρουσίαση που βασίζεται σε κανόνες και μια μη μονοτονική σημασιολογία μέσω της άρνησης σαν αποτυχία. Σε αντίθεση με τη συνήθη λογική σημασιολογία του προγραμματισμού,ο OASP, επιτρέπει σε τομείς να αποτελούνται από άλλα αντικείμενα από εκείνα που υπάρχουν στο πρόγραμμα λογικής. Ο OASP, είναι ένας βιώσιμος υποψήφιος για την εννοιολογική συλλογιστική, λόγω του ότι η παρουσίασή του βασίζεται σε κανόνες και μπορεί να χρησιμοποιηθεί για reasoning, ως rule-based και με conceptual γνώση στον Σημασιολογικό Ιστό.
Μια σημαντική πρόκληση για τον OASP, είναι ο έλεγχος της αναποφασιστηκότητας της ικανοποιησιμότητας.
Μια g-hybrid βάση γνώσεων, αποτελείται από μια Περιγραφικής Λογικής βάση γνώσης και ένα φυλασσόμενο πρόγραμμα. Η αλληλεπίδραση μεταξύ του προγράμματος και της DL βάσης γνώσης περιορίζεται με την απαίτηση ότι οι μεταβλητές που εμφανίζονται σε μη-DL atoms, εμφανίζονται σε θετικά μη-DL atoms στο σώμα, όπου τα άτομα DL είναι άτομα που περιλαμβάνουν μια έννοια ή ένα σύμβολο, από την βάση γνώσεων DL.
Οι g-hybrid βάσεις γνώσεων δεν απαιτούν ένα τέτοιο περιορισμό.
Παρακάτω ακολουθεί μια εισαγωγή στο Open answer set Programming και στην λογική Περιγραφή.

7.1 Decidable Open Answer Set Programming

Έχουν εισαχθεί τα Decidable Open Answer Set semantics, έτσι ώστε να μην μπορεί να υποτεθεί η μοναδικότητα των ονομάτων από προεπιλογή. Οι σταθερές, οι μεταβλητές, οι όροι και τα άτομα, ορίζονται ως συνήθως. Ένα literal, είναι ένα atom $p(\vec{t})$ ή ένα $naf\text{-}literal \;\; not\, p(\vec{t})$ με t πλειάδα όρων.
Υποθέτουμε την ύπαρξη ισότητας και ανισότητας. Ένα κανονικό atom, είναι atom χωρίς ισότητα. Ένα πρόγραμμα είναι ένα αριθμήσιμο σύνολο κανόνων .
Καλούμε ένα κατηγόρημα p ως free, εάν υπάρχει κανόνας $p(\vec{X}) \vee not\, p(\vec{X}) \leftarrow$. Τα atoms, οι λεκτικοί τους κανόνες, και τα προγράμματα που δεν περιέχουν μεταβλητές είναι ground.

→ΠΕΡΙΓΡΑΦΙΚΗ ΛΟΓΙΚΗ $\mathcal{DLRO}^{-\{\leqslant\}}$

Το $\mathcal{DLR}$, είναι ένα DL που υποστηρίζει τους ρόλους των αυθαίρετων arity, ενώ οι περισσότερες ψηφιακές βιβλιοθήκες υποστηρίζουν μόνο δυαδικό ρόλους. Εισάγεται η επέκτασή αυτών με nominals, που καλείται $\mathcal{DLRO}$. Οι βασικές δομικές μονάδες του τα concept names A και relation names, P. Σ'αυτές, για το P, ισχύει ότι:

$$\text{arbitrary } n\text{-ary relation for } 2 \leqslant n \leqslant n_{\max} \text{ and } n_{\max}$$

Ακολουθούν οι role (R) και concept expressions:

$$\mathbf{R} \rightarrow \top_n \mid \mathbf{P} \mid (\$i/n : C) \mid \neg\mathbf{R} \mid \mathbf{R}_1 \sqcap \mathbf{R}_2 \mid \{(o_1, \ldots, o_n)\}$$

$$C \rightarrow \top_1 \mid A \mid \neg C \mid C_1 \sqcap C_2 \mid \exists[\$i]\mathbf{R} \mid \leqslant k\,[\$i]\mathbf{R} \mid \{o\}$$

Υποθέτουμε, ότι οι παραπάνω κατασκευές είναι καλά τυπωμένες.
Πρέπει, για να ισχύει αυτό να ικανοποιούνται οι εξής όροι:

$$\begin{aligned}
\top_n^{\mathcal{I}} &\subseteq (\Delta^{\mathcal{I}})^n \\
\mathbf{P}^{\mathcal{I}} &\subseteq \top_n^{\mathcal{I}} \\
(\neg\mathbf{R})^{\mathcal{I}} &= \top_n^{\mathcal{I}} \setminus \mathbf{R}^{\mathcal{I}} \\
(\mathbf{R}_1 \sqcap \mathbf{R}_2)^{\mathcal{I}} &= \mathbf{R}_1^{\mathcal{I}} \cap \mathbf{R}_2^{\mathcal{I}} \\
(\$i/n : C)^{\mathcal{I}} &= \{(d_1, \ldots, d_n) \in \top_n^{\mathcal{I}} \mid d_i \in C^{\mathcal{I}}\} \\
\top_1^{\mathcal{I}} &= \Delta^{\mathcal{I}} \\
A^{\mathcal{I}} &\subseteq \Delta^{\mathcal{I}} \\
(\neg C)^{\mathcal{I}} &= \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}} \\
(C_1 \sqcap C_2)^{\mathcal{I}} &= C_1^{\mathcal{I}} \cap C_2^{\mathcal{I}} \\
(\exists[\$i]\mathbf{R})^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid \exists(d_1, \ldots, d_n) \in \mathbf{R}^{\mathcal{I}} \cdot d_i = d\} \\
(\leqslant k\,[\$i]\mathbf{R})^{\mathcal{I}} &= \{d \in \Delta^{\mathcal{I}} \mid |\{(d_1, \ldots, d_n) \in \mathbf{R}^{\mathcal{I}} \mid d_i = d\}| \leqslant k\} \\
\{o\}^{\mathcal{I}} &= \{o^{\mathcal{I}}\} \subseteq \Delta^{\mathcal{I}} \\
\{(o_1, \ldots, o_n)\}^{\mathcal{I}} &= \{(o_1^{\mathcal{I}}, \ldots, o_n^{\mathcal{I}})\}
\end{aligned}$$

Εικόνα 55-όροι

7.2 g-Hybrid βάση γνώσεων.

G-Hybrid βάσεις γνώσεων είναι συνδυασμοί της Βάσης Γνώσεων Περιγραφικής Λογικής (DL), και του Guarded logic Programming (GP). Πρόκειται για παραλλαγή του r-hybrid βάσεων γνώσης.

Ορισμός 1: Ξεκινώντας με μία DL, μία g-Hybrid βάση γνώσεων, είναι ένα ζεύγος (Σ, P)

Όπου το Σ είναι η DL βάση και το P ένα φυλασσόμενο Πρόγραμμα (GP).
Σημειώστε ότι στον παραπάνω ορισμό, δεν υπάρχουν περιορισμοί σχετικά με τη χρήση των κύριων χαρακτήρων. Καλούμε τα atoms και τα literals στο P, που διαθέτουν υποκείμενα σύμβολα κατηγορημάτων που αντιστοιχούν στην έννοια ή το ρόλο των ονομάτων των βάσεων γνώσης DL, των atoms της και στα literals αντίστοιχα. Οι μεταβλητές των κανόνων δεν είναι υποχρεωμένοι να εμφανίζονται σε θετικά μη-DL atoms.

Ορισμός 2: Αν το (Σ, P) είναι μια g-Hybrid βάση γνώσεων. Μια ερμηνεία του θα είναι μία πλειάδα $(U, \mathcal{I}, M)$ όπου:

- $U = (D, \sigma)$ είναι προ-μετάφραση του P.
- $\mathcal{I} = (D, \cdot^{\mathcal{I}})$ είναι μια ερμηνεία του Σ
- M είναι μια ερμηνεία του $\Pi(P_U, \mathcal{I})$ και
- $b^{\mathcal{I}} = \sigma(b)$ για κάθε σύμβολο σταθεράς b, που εμφανίζεται στο Σ και στο P.

Το $(U = (D, \sigma), \mathcal{I}, M)$ είναι ένα μοντέλο g-Hybrid βάσης γνώσεων (Σ,P) αν το J είναι ένα μοντέλο του Σ και το M είναι ένα σύνολο απάντησης του $\Pi(P_U, \mathcal{I})$.

7.3 Μετάφραση σε Guarded Logic Programmes.

Η μετάφραση αυτή πληρεί τα κριτήρια της ικανοποιησιμότητας και χρησιμοποιείται για να επιτύχει την πολυπλοκότητα του συλλογισμού σε g-Hybrid βάσεις γνώσεων. Πριν από την εισαγωγή της μετάφρασης σε φυλασσόμενο πρόγραμμα, είχε εισαχθεί η μετάφραση μέσω παραδείγματος του πρώτου ορισμού του 7.2.

$$socialDrinker \sqsubseteq \exists[\$1](drinks \sqcap (\$3/3 : \{wine\}))$$

Εικόνα 56-παράδειγμα ορισμού 7.2

$$\leftarrow socialDrinker(X), not(\exists[\$1](drinks \sqcap (\$3/3 : \{wine\})))(X)$$

Εικόνα 57-μετάφραση

Μένει να εξασφαλιστεί ότι τα νεοεισαχθέντα κατηγορήματα συμπεριφέρονται σύμφωνα με τη σημασιολογία της DL. Όλα τα ονόματα υπάρχουν σε ψηφιακές βιβλιοθήκες. Στις ψηφιακές βιβλιοθήκες, η πλειάδα είναι είτε αληθείς ή ψευδείς σε μια δεδομένη ερμηνεία. Αυτή η συμπεριφορά μπορεί να συλληφθεί ακριβώς από τα ελεύθερα κατηγορήματα:

$$socialDrinker(X) \vee not\, socialDrinker(X) \leftarrow$$
$$drinks(X, Y, Z) \vee not\, drinks(X, Y, Z) \leftarrow$$

Εικόνα 58-συμπεριφορά που έχει συλληφθεί από κατηγορήματα

Τα εννοιολογικά ονόματα μεταφράζονται σε μοναδιαία free κατηγορήματα, ενώ τα n-αδικά ονόματα των ρόλων, μεταφράζονται σε n-αδικά free κατηγορήματα.

Κατά την ολοκλήρωση της μετάφρασης της DL βάσης γνώσης στην g-Hybrid βάση γνώσεων, το πρόγραμμα μπορεί να μείνει θεωρημένο ως έχει, δεδομένου ότι, εξ ορισμού της g-Hybrid βάσης γνώσης, είναι ήδη ένα φυλασσόμενο πρόγραμμα.

7.3 Η σχέση του DL+Log και άλλη σχετική δουλειά.

Η λεγόμενη DL + log βάση γνώσης συνδυάζει μια DL βάση γνώσεων με μία σχετικά ασφαλή διαζευκτική λογική του προγράμματος. Επισήμως, για μια συγκεκριμένη DL βάση γνώσης, μία DL + log βάση γνώσης είναι ένα ζεύγος (Σ,P) όπου το Σ είναι μια DL βάση γνώσης και αποτελείται από ένα TBox (ένα σύνολο αξιωμάτων ορολογίας) και ένα ABox, (ένα σετ από assertional αξιώματα), και το P περιέχει κανόνες $\alpha \leftarrow \beta$ τέτοους ώστε για κάθε κανόνα $r : \alpha \leftarrow \beta \in P$ ισχύει:

- $\alpha^- = \emptyset$
- β^- δεν περιέχει άτομα DL (DL-θετικότητα)
- κάθε μεταβλητή του r, συμβαίνει σε β^+ $(Datalog\ safeness)$
- κάθε μεταβλητή του r, η οποία λαμβάνει χώρα σε μη-DL atom, λαμβάνει χώρα σε μη-DL atom σε β^+ $(weak\ safeness)$.

Οι σημασιολογία για DL+ log είναι η ίδια όπως εκείνη της g-Hybrid βάσης γνώσης , με τις ακόλουθες εξαιρέσεις:

• Δεν απαιτείται η τυπική υπόθεση ονόματος, που λέει ότι ο domain της κάθε ερμηνείας είναι ουσιαστικά το ίδιο απείρως αριθμήσιμο σύνολο σταθερών. Ούτε έχουμε την παραδοχή μοναδικού ονόματος, κάνοντας τη σημασιολογία για g-Hybrid βάσεις γνώσεων να συμβαδίζουν περισσότερο με τα τωρινά πρότυπα του Σημασιολογικού Ιστού, όπως η OWL , που δεν εφαρμόζεται ούτε η τυπική υπόθεση ονομάτων, ούτε η μοναδική υπόθεση ονομάτων. Ο Rosati παρουσίασε μια έκδοση υβριδικών βάσεων γνώσης που δεν συμμορφώνονται με την μοναδική υπόθεση ονόματος σε προηγούμενη

• Ορίζεται μια ερμηνεία ως τριπλή $(U, \mathcal{I}, M)$, αντί ενός ζεύγους $(U, \mathcal{I}')$ όπου $\mathcal{I}' = \mathcal{I} \cup M$ Αυτό είναι ισοδύναμο με DL+Log.

Οι βασικές διαφορές των δύο προσεγγίσεων είναι:

• Τα προγράμματα DL+Log, μπορεί να έχει πολλά θετικά literals, στην κεφαλή, ενώ επιτρέπουμε το πολύ ένα. Παρόλα αυτά, επιτρέπονται αρνητικά literals, στην κεφαλή, ενώ αυτό δεν επιτρέπεται στο DL+Log. Επιπλέον, δεδομένου ότι τα άτομα DL ερμηνεύονται με κλασικό τρόπο, μπορούν να προσομοιωθούν θετικά άτομα DL στην κεφαλή, μέσω των αρνητικών ατόμων DL στο σώμα.

• Αντί για Datalog safeness, υπάρχει η ανάγκη guardedness.

• Δεν υπάρχει απαίτηση για weak safeness, δηλαδή, οι μεταβλητές της κεφαλής δεν χρειάζεται να εμφανίζονται θετικά σε ένα μη-DL atom. Η guardedness μπορεί να παρέχεται από ένα DL atom.

Παράδειγμα:

Example 1 contains the rule

$$problematic(X) \leftarrow socialDrinker(X), knowsFromAA(X, Y)$$

Αυτό μας επιτρέπει να συμπεράνουμε ότι το Χ μπορεί να είναι μία προβληματική περίπτωση, ακόμη και αν το Χ γνωρίζει κάποιον από το ΑΑ, αλλά δεν είναι «πίνει» με το εν λόγω πρόσωπο. Πιο σωστός θα ήταν ο κανόνας:

$$problematic(X, Z) \leftarrow drinks(X, Y, Z), knowsFromAA(X, Y)$$

όπου επεκτείνουμε το προβληματικό κατηγόρημα με το είδος του ποτού που το Χ έχει ένα πρόβλημα με. Στη συνέχεια, η μεταβλητή Z, φρουρείται από DL atom και ο κανόνας αυτός δεν είναι weakly safe, αλλά παρ 'όλα αυτά φυλάσσεται. Έτσι, η προκύπτουσα βάση γνώσεων δεν είναι DL + Log βάση γνώσης, αλλά είναι μία g-Hybrid βάση γνώσεης.

• Δεν υπάρχει η απαίτηση για DL-θετικότητα Ωστόσο, θα μπορούσε να επιτραπεί κάτι τέτοιο στην DL+ Log βάση γνώσης. Αντίστροφα, πρέπει να γίνει πιο χαλαρός ο περιορισμός σχετικά με την εμφάνιση των θετικών atoms στην κεφαλή (το οποίο επιτρέπει το πολύ ένα θετικό άτομο εκεί. Μπορεί να γραφτεί πάντα ένας τέτοιος κανόνας στο $p(X) \leftarrow \beta, notA(X)$, το οποίο περιέχει το πολύ ένα θετικό atom στην κεφαλή. Βεβαίως, δεν θα πρέπει να επιτρέπεται DL atoms να συμβαίνουν αρνητικά, επειδή τα κατηγορήματα DL ερμηνεύονται κλασικά και έτσι η άρνηση μπροστά από το DL atom, δεν είναι αναιρέσιμη.

• Δεν λαμβάνονται υπόψη τα aABoxes στη βάση γνώσης DL. Επιπλέον, ακόμη και αν η DL δεν περιλαμβάνει Nominals, το ABox μπορεί να γραφτεί ως ground facts σε ένα πρόγραμμα και τα ground facts είναι πάντα επιφυλακτικά.

• Η decidability για τον έλεγχο της satisfiability της DL + Log Βάσης γνώσης καταγραφής είναι εγγυημένη αν η decidability του συνδυαστικού προβλήματος περιορισμού ερώτηματος είναι εγγυημένη για το DL.

7.4 Συμπεράσματα και κατευθύνσεις για περαιτέρω έρευνα.

Ορίστηκαν g-Hybrid βάσεις γνώσεωνς που συνδυάζουν DL βάσεις γνώσεων με φυλασσόμενα λογικά προγραμμάτα (guarded). Συγκεκριμένα,συνδυάστηκε βάση γνώσης του DL $\mathcal{DLR}^{-\{\leq\}}$ το οποίο βρίσκεται κοντά στο OWL DL, με φυλασσόμενο πρόγραμμα, και έδειξε το decidability του παρόντος πλαισίου μέσω της μείωσης στο φυλασσόμενο προγράμματα στο πλαίσιο του Decidable Open Answer Set Programming. Συζητήθηκε η σχέση με DL+ log βάσεις γνώσης: g-Hybrid βάσεις γνώσεων ξεπερνούν ορισμένα από τα limitations του DL+ log, όπως και η μοναδική υπόθεση ονόματος, η καταγραφή δεδομένων safeness, και το αδύναμο DL- safeness. Εισάγεται όμως η απαίτηση του guardedness. Προς το παρόν, ένα σημαντικό μειονέκτημα της προσέγγισης είναι η έλλειψη υποστήριξης για ψηφιακές βιβλιοθήκες με τους περιορισμούς αριθμών που είναι

συνυφασμένοι με τη χρήση guarded προγραμμάτων του decidability. Μια λύση για αυτό θα ήταν να εξετάσει άλλους τύπους προγραμμάτων, όπως τα conceptual logic προγράμματα. Αν και είναι γνωστό ότι υπάρχουν όρια πολυπλοκότητας για πολλά μέρη του Decidable Open Answer Set Programming, συμπεριλαμβανομένου του φυλασσόμενου τμήματος, δεν υπάρχουν γνωστοί αποτελεσματικοί αλγόριθμοι για το Decidable Open Answer Set Programming.

ΒΙΒΛΙΟΓΡΑΦΙΑ

Tim berners – Lee “World Wide Web “

Francesca Lisi “Buidlding Rules on Top of Ontologies For the Semantic Web with Inducive Programming”

Special issue pappers

Ken Samuel “Translating OWL and Semantic Web Rules into Prolog”

Jan Wielemaker “SWI-Prolog and the Web”

Edna Ruckhaus “ALPWS paper ”

ΤΕΛΟΣ