ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αναπαράσταση Γνώσης και Συμπερασμός σε Περιβάλλοντα Κινητού Υπολογισμού

Ιωάννα Ι. Μποζινέκη

Επιβλέποντες: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής ΕΚΠΑ
Βασίλειος Παπαταξιάρχης, Υποψήφιος Διδάκτωρ ΕΚΠΑ

ΑΘΗΝΑ

ΜΑΙΟΣ 2011

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αναπαράσταση Γνώσης και Συμπερασμός σε Περιβάλλοντα Κινητού Υπολογισμού

Ιωάννα Ι. Μποζινέκη
Α.Μ.: 1115200400134

ΕΠΙΒΛΕΠΟΝΤΕΣ: **Ευστάθιος Χατζηευθυμιάδης,** Επίκουρος Καθηγητής ΕΚΠΑ
Βασίλειος Παπαταξιάρχης, Υποψήφιος Διδάκτωρ ΕΚΠΑ

ΠΕΡΙΛΗΨΗ

Η αναπαράσταση γνώσης μελετά τη μορφοποίηση της γνώσης και την επεξεργασία της από μηχανές. Τεχνικές αυτοματοποιημένης συλλογιστικής επιτρέπουν σε ένα υπολογιστικό σύστημα, να εξάγει συμπεράσματα από μια βάση γνώσης πάνω σε κάποιο πεδίο ενδιαφέροντος. Οι οντολογίες παρέχουν ένα εννοιολογικό αλλά και υπολογιστικό μοντέλο ενός συγκεκριμένου πεδίου ενδιαφέροντος. Με τον τρόπο αυτό, τα υπολογιστικά συστήματα μπορούν να παίρνουν αποφάσεις, μέσω κάποιας συλλογιστικής διαδικασίας στη γνώση του πεδίου, όπως και οι άνθρωποι. Η εκφραστικότητα των γλωσσών αναπαράστασης οντολογιών και οι αντίστοιχες διαδικασίες εξαγωγής συμπερασμάτων, χαρακτηρίζονται συνήθως από υψηλή πολυπλοκότητα και μεγάλη κατανάλωση πόρων, κάτι που δεν είναι αποδεκτό για κινητά περιβάλλοντα.

Στην παρούσα εργασία παρουσιάζουμε τις μεθόδους αναπαράστασης γνώσης και τα αποτελέσματα των αντίστοιχων διαδικασιών συμπερασμού, στα πλαίσια ενός περιβάλλοντος διάχυτου υπολογισμού. Αρχικά, μελετάμε τα χαρακτηριστικά που διέπουν τα περιβάλλοντα κινητού υπολογισμού και, κατ' επέκταση, τους περιορισμούς που πρέπει να αντιμετωπίζονται από τις κινητές εφαρμογές. Στη συνέχεια, αναλύουμε τις διαφορετικές μορφές αναπαράστασης γνώσης και τις συλλογιστικές διαδικασίες που επιτρέπει η καθεμία. Παρουσιάζουμε την ανάγκη μετατροπής της γλώσσας οντολογιών σε διαφορετική μορφή, ώστε να αντιμετωπίζονται οι περιορισμοί που τίθενται από το περιβάλλον μιας κινητής συσκευής. Έτσι, επιλέγουμε μορφές γνώσης, αποδεκτές για περιβάλλοντα κινητού υπολογισμού, και μελετάμε τη χρήση μηχανών συμπερασμού πάνω σε αυτές. Τέλος, υλοποιούμε ένα περιβάλλον δοκιμών για την πειραματική εξέταση των δυνατοτήτων συμπερασμού σε μια κινητή συσκευή και παρουσιάζουμε τα αποτελέσματα.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Τεχνολογίες γνώσης

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: διάχυτος υπολογισμός, κινητός υπολογισμός, αναπαράσταση γνώσης, μέθοδοι συλλογιστικής, mobile reasoning

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΑΤΑΛΟΓΟΣ ΣΧΗΜΑΤΩΝ

ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΠΡΟΛΟΓΟΣ

Η παρούσα πτυχιακή εργασία εκπονήθηκε στα πλαίσια του Προπτυχιακού Προγράμματος Σπουδών του Τμήματος Πληροφορικής και Τηλεπικοινωνιών του Εθνικού και Καποδιστριακού Πανεπιστημίου Αθηνών. Το αντικείμενο μελέτης της είναι οι μορφές αναπαράστασης γνώσης και οι αντίστοιχοι μηχανισμοί συλλογιστικής. Ειδικότερα, η εργασία πραγματεύεται την εφαρμογή των παραπάνω μορφών και συλλογιστικών, μέσω μηχανών συμπερασμού, σε περιβάλλον κινητού υπολογισμού.

Θα ήθελα να ευχαριστήσω θερμά τους υποψήφιους διδάκτορες του Τμήματος Πληροφορικής και Τηλεπικοινωνιών Βασίλη Παπαταξιάρχη και Βασίλη Τσέτσο, και τον επίκουρο καθηγητή κ. Ευστάθιο Χατζηευθυμιάδη για την εμπιστοσύνη τους και την συνεισφορά τους κατά την εκπόνηση της παρούσας πτυχιακής εργασίας.

Αθήνα, Μάιος 2011

Ιωάννα Ι. Μποζινέκη

1. ΕΙΣΑΓΩΓΗ

1.1 Διάχυτος Υπολογισμός

Ο διάχυτος υπολογισμός (pervasive computing) αποτελεί ένα σημαντικό βήμα εξέλιξης ενός έργου που χρονολογείται από τα μέσα της δεκαετίας του 1970 και είναι η τάση προς όλο και περισσότερο «πανταχού παρόντων» (ένα άλλο όνομα της περιοχής είναι «πανταχού παρών» υπολογισμός (ubiquitous)), συνδεδεμένων υπολογιστικών συσκευών, μια τάση η οποία υλοποιείται από τη σύγκλιση προηγμένων ηλεκτρονικών - και ιδιαίτερα, ασύρματων - τεχνολογιών και του Internet. Διάχυτες υπολογιστικές συσκευές δεν είναι οι προσωπικοί υπολογιστές, αλλά μικρές- ακόμη και αόρατες - συσκευές, κινητές ή ενσωματωμένες σε, σχεδόν κάθε είδος αντικειμένου, συμπεριλαμβανομένων των αυτοκινήτων, εργαλείων, συσκευών, ρουχισμού και διάφορων καταναλωτικών αγαθών. Όλα επικοινωνούν μέσω των όλο και περισσότερο διασυνδεδεμένων δικτύων.

Δύο πρότερα βήματα αυτή της εξέλιξης είναι τα κατανεμημένα συστήματα και ο κινητός υπολογισμός. Παρακάτω ακολουθεί μια σύντομη περιγραφή των κατανεμημένων συστημάτων, καθώς και τα χαρακτηριστικά και οι προκλήσεις που διέπουν τον διάχυτο υπολογισμό. Ο κινητός υπολογισμός περιγράφεται αναλυτικά στην ενότητα 1.2.

1.1.1 Κατανεμημένα συστήματα

Ο τομέας των κατανεμημένων συστημάτων (distributed systems) προέκυψε από τον συνδυασμό των προσωπικών υπολογιστών και των τοπικών δικτύων. Η έρευνα που ακολούθησε από τα μέσα της δεκαετίας του '70 έως τις αρχές της δεκαετίας του '90 δημιούργησε ένα εννοιολογικό πλαίσιο και μια αλγοριθμική βάση που έχει διαχρονική αξία σε οποιαδήποτε εργασία εμπλέκονται δύο ή περισσότεροι υπολογιστές (σταθεροί ή κινητοί, ενσύρματοι ή ασύρματοι). Η γνώση αυτή εκτείνεται σε πολλές περιοχές οι οποίες είναι θεμελιώδεις για τον διάχυτο υπολογισμό. Οι σημαντικότερες εξ΄ αυτών είναι οι παρακάτω:

• *Η απομακρυσμένη επικοινωνία (remote communication)* που περιλαμβάνει τη στρωματοποιημένη αρχιτεκτονική των πρωτοκόλλων και την απομακρυσμένη κλήση διαδικασίας (remote procedure call)

• *Η ανοχή σφαλμάτων (fault tolerance).* Η περιοχή αυτή περιλαμβάνει τις ατομικές συναλλαγές (atomic transactions) και τις κατανεμημένες συναλλαγές (distributed transactions)

• *Η υψηλή διαθεσιμότητα (high availability)*, περιλαμβάνει τον αισιόδοξο και απαισιόδοξο έλεγχο αντιγράφων και την αισιόδοξη αποκατάσταση (optimistic recovery)

• *Η απομακρυσμένη πρόσβαση πληροφοριών (remote information access)*. Σε αυτήν την κατηγορία περιλαμβάνονται τα κατανεμημένα συστήματα αρχείων και οι κατανεμημένες βάσεις δεδομένων

• *Η ασφάλεια (security)*, που περιλαμβάνει την κρυπτογραφημένη πιστοποίηση και ιδιωτικότητα (encrypted authentication and privacy)

1.1.2 Χαρακτηριστικά του διάχυτου υπολογισμού

Ένα περιβάλλον διάχυτου υπολογισμού είναι «γεμάτο» με υπολογιστικές και επικοινωνιακές δυνατότητες και χαρακτηρίζεται δίκαια πολλές φορές σαν «τεχνολογία που εξαφανίζεται». Δεδομένου ότι η κίνηση είναι ένα αναπόσπαστο τμήμα της καθημερινής ζωής, ένα τέτοιο περιβάλλον πρέπει να υποστηρίζει την κινητικότητα, διαφορετικά ένας χρήστης θα συνειδητοποιεί έντονα την ύπαρξη της τεχνολογίας (λόγω της απουσίας της) όταν αυτός θα κινείται. Ως εκ τούτου, η έρευνα που είναι σχετική με τον διάχυτο υπολογισμό συμπεριλαμβάνει εκτός από τα ερευνητικά πεδία του κινητού υπολογισμού και τέσσερα επιπλέον σημαντικά θέματα που περιγράφονται παρακάτω.

• *Αποτελεσματική χρήση ευφυών χώρων (smart spaces).* Η πρώτη ερευνητική ώθηση του διάχυτου υπολογισμού είναι η αποτελεσματική χρήση των ευφυών (έξυπνων) χώρων. Ένας χώρος μπορεί να είναι μέρος ενός κτιρίου (μια αίθουσα συνεδριάσεων, ένας διάδρομος, κτλ.) ή μπορεί να είναι μια καθορισμένη ανοικτή περιοχή όπως ένα προαύλιο ή ένας υπαίθριος χώρος. Η ενσωμάτωση υπολογιστικής υποδομής στην κτιριακή υποδομή είναι αυτό που χαρακτηρίζεται σαν «ευφυής χώρος» και αποτελεί ουσιαστικά τη συνένωση δύο διαφορετικών περιοχών [1]. Η σύντηξη (fusion) αυτών των περιοχών επιτρέπει την αλληλεπίδρασή τους. Ένα απλό παράδειγμα είναι η αυτόματη ρύθμιση της θέρμανσης, της ψύξης και του φωτισμού σε ένα δωμάτιο ανάλογα με τις προτιμήσεις (προφίλ) ενός χρήστη. Η επιρροή προς την άλλη κατεύθυνση είναι επίσης δυνατή. Π.χ. το λογισμικό στο τερματικό ενός χρήστη μπορεί να συμπεριφερθεί διαφορετικά ανάλογα με τη θέση του κατόχου του.

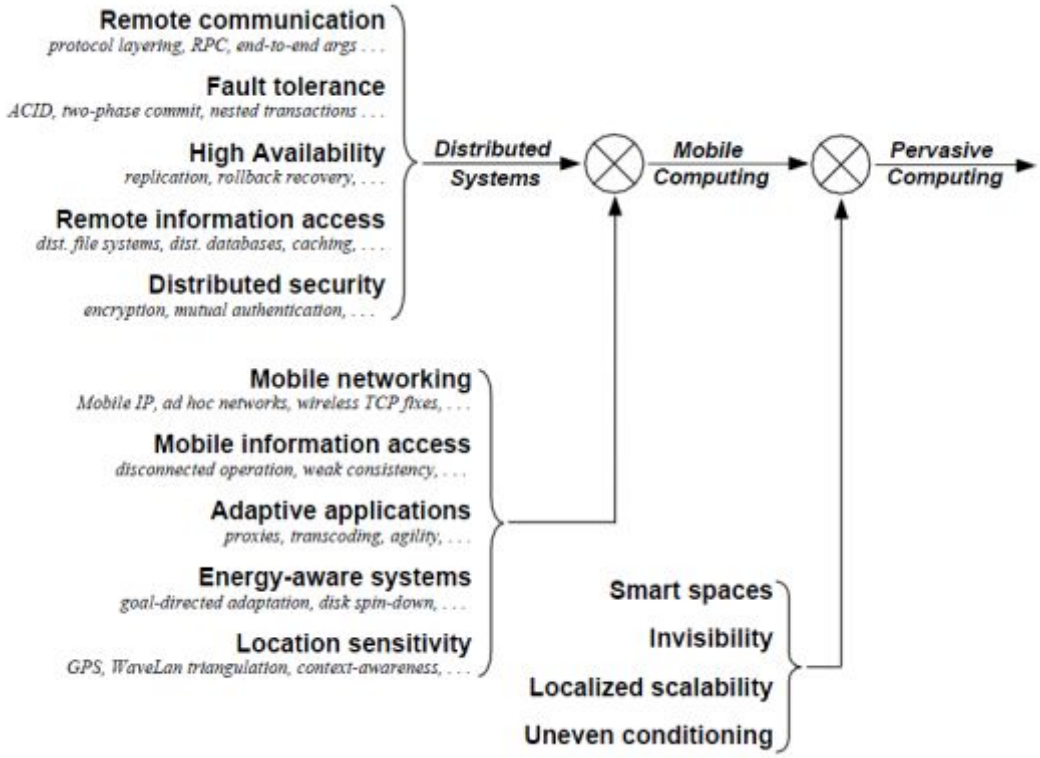
• *Αορατότητα.* Η δεύτερη ώθηση είναι η αορατότητα (invisibility). Το ιδανικό που εκφράστηκε από τον M. Weiser είναι η πλήρης εξαφάνιση της τεχνολογίας του διάχυτου υπολογισμού από την επίγνωση του χρήστη. Στην πράξη, μια λογική προσέγγιση σε αυτό το ιδανικό είναι η ελάχιστη απόσπαση της προσοχής των χρηστών. Εάν ένα περιβάλλον διάχυτου υπολογισμού ικανοποιεί συνεχώς τις προσδοκίες των χρηστών και σπάνια τους παρουσιάζει εκπλήξεις, τους επιτρέπει μία ξεκούραστη αλληλεπίδραση σχεδόν σε υποσυνείδητο επίπεδο [2]. Συγχρόνως, μία αναμονή μπορεί να είναι απαραίτητη για την αποφυγή μία μεγάλης και δυσάρεστης έκπληξης αργότερα.

• *Δυνατότητα κλιμάκωσης*. Η τρίτη ερευνητική ώθηση είναι η κλιμάκωση. Καθώς οι ευφυείς χώροι εξελίσσονται συνεχώς, οι αλληλεπιδράσεις μεταξύ του εξοπλισμού ενός χρήστη (κινητό τερματικό, κτλ.) και του περιβάλλοντός του αυξάνονται. Αυτό έχει σημαντικές επιπτώσεις στο εύρος ζώνης, αλλά και στην ενέργεια των τερματικών συσκευών. Όπως είναι φυσικό η παρουσία παραπάνω χρηστών περιπλέκει το πρόβλημα. Έτσι, η κλιμάκωση, υπό την ευρύτερη έννοια, είναι μία πτυχή που πρέπει να λαμβάνουμε σοβαρά υπ' όψιν στον διάχυτο υπολογισμό. Προηγούμενες προσεγγίσεις πάνω στην κλιμάκωση είχαν αγνοήσει τη φυσική απόσταση. Για παράδειγμα, ένας κεντρικός υπολογιστής δικτύου (web server) ή ένας κεντρικός υπολογιστής αρχείων (file server) πρέπει να είναι δυνατό να υποστηρίξουν όσο το δυνατόν περισσότερους χρήστες, ασχέτως εάν είναι τοποθετημένοι πολύ κοντά ή χιλιόμετρα μακριά από αυτούς. Η κατάσταση είναι πολύ διαφορετική στον διάχυτο υπολογισμό. Εδώ, η συχνότητα των αλληλεπιδράσεων πρέπει να μειωθεί καθώς ο χρήστης απομακρύνεται διαφορετικά και το τερματικό του αλλά και το σύστημα θα υπερφορτωθεί. Παρόλο που ένας κινητός χρήστης ο οποίος βρίσκεται μακριά (στο σπίτι του) μπορεί να παράγει ακόμα αλληλεπιδράσεις με τις περιοχές που είναι σχετικές σ' αυτόν, ο κύριος όγκος των αλληλεπιδράσεών του είναι κυρίως τοπικός.

• *Κάλυψη (απόκρυψη) διαφορετικών συνθηκών περιβάλλοντος*. Η τέταρτη ώθηση είναι η ανάπτυξη των τεχνικών για την κάλυψη διαφορετικών συνθηκών περιβάλλοντος. Ο βαθμός διείσδυσης του διάχυτου υπολογισμού στις υποδομές μπορεί να ποικίλει αρκετά και εξαρτάται από πολλούς μη τεχνικούς παράγοντες όπως η οργανωτική δομή και τα οικονομικά και επιχειρησιακά πρότυπα. Η ομοιόμορφη διείσδυση απέχει πολλά έτη ή δεκαετίες μακριά. Στο μεσοδιάστημα, θα υπάρχουν τεράστιες διαφορές «ευφυΐας» των διαφορετικών περιβαλλόντων. Ο διαθέσιμος εξοπλισμός σε γραφεία, αίθουσες

συσκέψεων ή σε τάξεις μπορεί να είναι περισσότερο περίπλοκος απ' ό,τι σε άλλες θέσεις. Αυτή η μεγάλη δυναμική περιοχή της «ευφυΐας» μπορεί να είναι ενοχλητική σε έναν χρήστη, μειώνοντας έτσι την αορατότητα του διάχυτου υπολογισμού. Η πλήρης αορατότητα μπορεί να είναι αδύνατη, αλλά η μειωμένη μεταβλητότητα είναι κάτι που μπορεί να επιτευχθεί.

Στην Εικόνα 1.1 διακρίνεται πώς τα ερευνητικά προβλήματα στον διάχυτο υπολογισμό είναι σχετικά με εκείνα του κινητού υπολογισμού και των κατανεμημένων συστημάτων. Τα νέα προβλήματα εμφανίζονται καθώς κινούμαστε από τα αριστερά προς τα δεξιά. Επιπλέον, η επίλυση πολλών προβλημάτων που αντιμετωπίσαμε προηγουμένως τώρα γίνεται πιο σύνθετη. Η αύξηση της πολυπλοκότητας είναι πολλαπλασιαστική παρά προσθετική. Είναι πολύ δυσκολότερο να σχεδιαστεί και να εφαρμοστεί ένα σύστημα διάχυτου υπολογισμού από ένα απλό κατανεμημένο σύστημα της ίδιας ευρωστίας και αποδοτικότητας.



Εικόνα 1.1: Η εξέλιξη των τεχνολογιών

Στη συνέχεια παραθέτουμε δύο παραδείγματα που παρουσιάζονται στο [3] και είναι σενάρια που ίσως να μπορούν να πραγματοποιηθούν σε λίγα χρόνια.

Σενάριο 1

Η Jane είναι στην πύλη 23 στον αερολιμένα του Pittsburgh, περιμένοντας την πτήση της. Έχει πληκτρολογήσει πολλά έγγραφα, και θα επιθυμούσε να χρησιμοποιήσει την ασύρματη σύνδεση για να τα στείλει με e-mail. Δυστυχώς όμως, το εύρος ζώνης δεν είναι αρκετό καθώς πολλοί επιβάτες στις πύλες 22 και 23 χρησιμοποιούν το ασύρματο δίκτυο για να περιηγούνται στο Internet. Ένα έξυπνο σύστημα διάχυτου υπολογισμού παρατηρεί ότι στο τρέχον εύρος ζώνης η Jane δε θα είναι σε θέση να ολοκληρώσει την αποστολή των εγγράφων προτού αναχωρήσει η πτήση της. Το έξυπνο σύστημα ανακαλύπτει ότι το ασύρματο εύρος ζώνης στην πύλη 15 είναι πολύ καλό, και ότι δεν υπάρχει καμία πτήση αναχώρησης ή άφιξης στις κοντινές πύλες για την επόμενη μισή ώρα. Ένα πλαίσιο διαλόγου εμφανίζεται στην οθόνη της Jane και της προτείνει να πάει στην πύλη 15, η οποία είναι μόνο τρία λεπτά μακριά. Της ζητά επίσης να δώσει προτεραιότητα στο e-mail της, έτσι ώστε τα πιο σημαντικά μηνύματα να αποσταλούν πρώτα. Η Jane δέχεται τις συμβουλές του συστήματος και κατευθύνεται προς την πύλη 15. Όταν φτάσει αρχίζει να παρακολουθεί τηλεόραση έως ότου το σύστημα την ενημερώνει ότι τα μηνύματά της σχεδόν εστάλησαν, και ότι μπορεί να γυρίσει πίσω. Το τελευταίο μήνυμα αποστέλλεται κατά τη διάρκεια της επιστροφής της στην πύλη 23 όπου ετοιμάζεται να αναχωρήσει με την προγραμματισμένη πτήση.

Σενάριο 2

Ο Fred βρίσκεται στο γραφείο του, και προετοιμάζεται για μια συνεδρίαση στην οποία θα κάνει μια παρουσίαση και μια επίδειξη λογισμικού. Η αίθουσα συνεδριάσεων βρίσκεται σε μικρή απόσταση (περίπου 10 λεπτά με τα πόδια) από την πανεπιστημιούπολη. Είναι η ώρα να φύγει, αλλά ο Fred δεν είναι αρκετά έτοιμος. Παίρνει βιαστικά τον ασύρματο φορητό υπολογιστή του (PalmXXII) και φεύγει. Ένα έξυπνο σύστημα μεταφέρει την εργασία του από τον υπολογιστή του γραφείου του (desktop computer) στο φορητό του, και του επιτρέπει να κάνει τις τελευταίες διορθώσεις κατά τη διάρκεια του περιπάτου με τη χρήση φωνητικών εντολών. Το σύστημα συμπεραίνει πού πηγαίνει ο Fred από το ημερολόγιό του και από την υπηρεσία προσδιορισμού θέσης της πανεπιστημιούπολης. Κατεβάζει την παρουσίαση και το λογισμικό επίδειξης στον υπολογιστή προβολής και θερμαίνει τον προβολέα. Ο Fred τελειώνει τις διορθώσεις λίγο πριν φτάσει στην αίθουσα συνεδριάσεων. Καθώς περπατά μέσα, το έξυπνο σύστημα μεταφέρει την τελική παρουσίαση στον υπολογιστή προβολής. Καθώς η παρουσίαση προχωρά, ο Fred είναι έτοιμος δείξει μια διαφάνεια με τις ιδιαίτερα ευαίσθητες πληροφορίες των

προϋπολογισμών. Το έξυπνο σύστημα αντιλαμβάνεται ότι αυτό μπορεί να είναι λάθος διότι η ανίχνευση και αναγνώριση προσώπων δείχνει ότι υπάρχουν μερικά άγνωστα πρόσωπα παρόντα στην αίθουσα. Επομένως, προειδοποιεί το Fred. Συνειδητοποιώντας ότι το σύστημα είναι σωστό, ο Fred δεν παρουσιάζει τη διαφάνεια. Τελειώνει την παρουσίαση του, αφήνοντας το ακροατήριο εντυπωσιασμένο.

Τα παραπάνω δύο σενάρια ενσωματώνουν πολλές βασικές ιδέες του διάχυτου υπολογισμού. Το *Σενάριο 1* παρουσιάζει τη σημασία της προδραστικότητας (proactivity): η Jane είναι σε θέση να ολοκληρώσει τη μετάδοση των e-mail της επειδή ένα έξυπνο σύστημα είχε την πρόβλεψη να υπολογίσει πόσο χρόνο θα έπαιρνε η όλη διαδικασία. Είναι σε θέση να αρχίσει να περπατάει πίσω προς την πύλη αναχώρησής της προτού ολοκληρωθεί η μετάδοση επειδή το σύστημα κάνει προβλέψεις για το απαιτούμενο εύρος ζώνης και τον εναπομείναντα χρόνο αποστολής των μηνυμάτων.

Αυτό το σενάριο παρουσιάζει επίσης τη σημασία του συνδυασμού της γνώσης από τα διαφορετικά επίπεδα (στρώματα) του συστήματος. Η ασύρματη συμφόρηση είναι ένα χαμηλού επιπέδου φαινόμενο ενώ η γνώση για την ώρα αναχώρησης είναι μια εφαρμογή υψηλότερου επιπέδου. Μόνο με το συνδυασμό αυτών των διαφορετικών επιπέδων της γνώσης μπορεί το σύστημα να βοηθήσει τη Jane. Το σενάριο παρουσιάζει, επίσης, την αξία ενός ευφυούς χώρου. Το σύστημα είναι σε θέση να λάβει γνώση για τις συνθήκες που επικρατούν σε άλλες πύλες, τις ώρες άφιξης/αναχώρησης των πτήσεων, τις πύλες αναχώρησης, την απόσταση μεταξύ των πυλών, κ.ά.

Το *Σενάριο 2* παρουσιάζει τη δυνατότητα να μετακινούμε καταστάσεις-δεδομένα με ευκολία μεταξύ διαφορετικών πλατφόρμων. Π.χ. από έναν υπολογιστή γραφείου προς ένα φορητό και από το φορητό υπολογιστή προς τον υπολογιστή προβολής. Επίσης η αυτόματα ρυθμιζόμενη συμπεριφορά (self-tuning) ανάλογα με τις περιστάσεις, φαίνεται από τη δυνατότητα που έχει ο χρήστης να μπορεί να υπαγορεύει στη φορητή συσκευή αντί να χρησιμοποιεί ποντίκι και πληκτρολόγιο. Το σενάριο ενσωματώνει πολλές περιπτώσεις προδραστικότητας (proactivity): συμπεραίνοντας το σύστημα ότι ο Fred κατευθύνεται στο χώρο που θα γίνει η παρουσίαση, θερμαίνει τον προβολέα. Επιπλέον, μεταφέρει την παρουσίαση και το λογισμικό προς επίδειξη στον τοπικό υπολογιστή. Τέλος, προβλέποντας ότι η διαφάνεια του προϋπολογισμού έπεται και ξέροντας ότι στο δωμάτιο υπάρχουν άγνωστα πρόσωπα, προειδοποιεί τον Fred και αποτρέπει την εμφάνιση της διαφάνειας. Η αξία των ευφυών χώρων φαίνεται σε πολλές καταστάσεις: ο

εντοπισμός της θέσης και το ημερολόγιο επιτρέπουν στο σύστημα να συμπεράνει που κατευθύνεται ο Fred, ο προβολέας επιτρέπει την προθέρμανση πριν την άφιξη, το δωμάτιο που είναι εξοπλισμένο με κάμερες δίνει τη δυνατότητα αναγνώρισης προσώπων, κάτι το οποίο είναι βασικό για την προειδοποίηση στο Fred.

Η πραγματική έρευνα εστιάζει στη συνεχή ένταξη και ενσωμάτωση των επί μέρους συστατικών τεχνολογιών σε ένα ολοκληρωμένο σύστημα. Τα δύσκολα προβλήματα εντοπίζονται στο σχεδιασμό της αρχιτεκτονικής και τη σύνθεση των επιμέρους τεχνολογιών. Στην παρακάτω ενότητα περιγράφουμε τις προκλήσεις του διάχυτου υπολογισμού όπου εντοπίζουμε και αναλύουμε μερικά από αυτά τα προβλήματα.

1.1.3 Προκλήσεις για τον Διάχυτο Υπολογισμό

Η πρακτική εφαρμογή του διάχυτου υπολογισμού απαιτεί τη λύση πολλών και δύσκολων προβλημάτων. Βασιζόμενοι στα προαναφερθέντα, θα εξετάσουμε τώρα μερικά από αυτά τα προβλήματα. Υποθέτουμε ότι κάθε χρήστης είναι «εμβυθισμένος» σε ένα προσωπικό χώρο υπολογισμού που τον συνοδεύει παντού και μεσολαβεί για όλες τις αλληλεπιδράσεις του με το περιβάλλον. Αυτός ο προσωπικός χώρος μπορεί να είναι π.χ. ένας φορητός υπολογιστής. Όπως υποδεικνύεται παρακάτω, ο υπολογιστής αυτός του χρήστη (client) πρέπει να είναι αρκετά περίπλοκος και, ως εκ τούτου, σύνθετος.

1.1.3.1 Πρόθεση του χρήστη

Για να είναι αποτελεσματική η προδραστικότητα (proactivity), θα πρέπει ένα σύστημα διάχυτου υπολογισμού να αντιλαμβάνεται τις προθέσεις του χρήστη. Διαφορετικά, θα είναι σχεδόν αδύνατο να καθοριστούν ποιες ενέργειες θα τον βοηθήσουν. Για παράδειγμα, υποθέτουμε ότι κάποιος παρακολουθεί βίντεο πάνω από ένα δίκτυο που το εύρος ζώνης του πέφτει απότομα. Τι πρέπει λοιπόν να γίνει σε αυτήν την περίπτωση;

(α) το σύστημα να μειώσει την ποιότητα του βίντεο

(β) να γίνει μία μικρή διακοπή στη ροή μήπως βρεθεί κάποια καλύτερη σύνδεση με μεγαλύτερο εύρος ζώνης ή

(γ) το σύστημα να πληροφορήσει το χρήστη ότι δεν είναι πλέον εφικτή η παρακολούθηση του βίντεο.

Η σωστή επιλογή θα εξαρτηθεί από αυτό που ο χρήστης θα ήθελε, δηλαδή από την πρόθεσή του. Τα σημερινά συστήματα δεν είναι ικανά να συλλάβουν και να εκμεταλλευτούν την πρόθεση του χρήστη. Αφ' ενός μεν, είναι γενικές εφαρμογές που δεν έχουν καμία ιδέα τι προσπαθεί να κάνει χρήστης, και μπορούν επομένως να προσφέρουν ελάχιστη υποστήριξη για την προσαρμογή και την προδραστικότητα. Αφ' ετέρου δε, είναι εφαρμογές που προσπαθούν να προβλέψουν την πρόθεση του χρήστη αλλά το κάνουν τόσο άσχημα που καταλήγουν να γίνονται ενοχλητικές. Η ανάγκη για την ενσωμάτωση της πρόθεσης των χρηστών σε ένα σύστημα δημιουργεί εύλογα ερευνητικά ερωτήματα:

• Μπορεί η πρόθεση χρηστών να προβλεφθεί, ή πρέπει να μας παρασχεθεί έτοιμη; Στη δεύτερη περίπτωση από πού θα αντλείται (π.χ. αρχείο;)

• Πώς μοντελοποιείται η πρόθεση των χρηστών; Πόσο λεπτομερής πρέπει να είναι αυτή η πληροφορία ώστε να είναι χρήσιμη; Πότε και πώς ενημερώνεται; Πώς τα διαφορετικά επίπεδα ενός συστήματος έχουν πρόσβαση σε αυτήν την πληροφορία;

• Πώς κάποιος χαρακτηρίζει την ακρίβεια της γνώσης; Είναι χρήσιμη η ελλιπής ή η ανακριβής γνώση της πρόθεσης των χρηστών; Είναι καλύτερα να αγνοήσει κανείς τέτοιου είδους γνώση στη λήψη των αποφάσεων;

• Μπορεί η προσπάθεια να ληφθεί υπόψη η πρόθεση να δημιουργήσει ένα αδικαιολόγητο φόρτο στο χρήστη; Θα επηρεαστεί η απόδοση του συστήματος;

1.1.3.2 Κυβερνοβοσκή (Cyber Foraging)

Η ανάγκη να σμικρύνουμε τις κινητές συσκευές, να τις κάνουμε ελαφρύτερες και να έχουν μπαταρίες μεγάλης διάρκειας οδηγεί στο γεγονός ότι οι υπολογιστικές τους ικανότητες θα πρέπει να μειωθούν. Η *κυβερνοβοσκή* μπορεί να είναι ένας αποτελεσματικός τρόπος ώστε να αντιμετωπιστεί αυτό το πρόβλημα. Η ιδέα είναι να αυξάνονται δυναμικά οι υπολογιστικοί πόροι ενός ασύρματου κινητού υπολογιστή (laptop) με την εκμετάλλευση μίας ενσύρματης υποδομής. Οι υπολογιστές γραφείου πωλούνται σήμερα για μερικές εκατοντάδες ευρώ, και οι τιμές τους συνεχίζουν να μειώνονται. Στο μέλλον, προβλέπεται οι δημόσιοι χώροι όπως τα αεροδρόμια και τα καταστήματα καφέ να εξοπλίζονται με ισχυρούς κεντρικούς υπολογιστές (servers) και να

παρέχουν υπηρεσίες προς όφελος των πελατών τους. Αυτοί οι χώροι έχοντας και ενσύρματη υποδομή μπορεί να προσφέρουν στους χρήστες υπηρεσίες οι οποίες απαιτούν μεγάλο εύρος ζώνης υποκαθιστώντας (αναπληρώνοντας) έτσι την ασύρματη υποδομή.

Παραθέτουμε στη συνέχεια ένα χαρακτηριστικό σενάριο. Όταν ένας κινητός υπολογιστής εισέρχεται σε ένα χώρο, ανιχνεύει αρχικά την παρουσία πιθανών αναπληρωματικών (surrogate) συστημάτων -υποδομών και διαπραγματεύεται τη χρήση τους. Η επικοινωνία με ένα τέτοιο σύστημα γίνεται ασύρματα με τεχνολογίες peer-to-peer, με το αναπληρωματικό σύστημα να παίζει το ρόλο της πύλης προς το διαδίκτυο (internet gateway) για τον κινητό υπολογιστή. Όταν πρέπει να εκτελεσθούν δύσκολοι και χρονοβόροι υπολογισμοί σε ένα μεγάλο όγκο δεδομένων, ο κινητός υπολογιστής αναθέτει την διαδικασία στο αναπληρωματικό σύστημα το οποίο μπορεί να αποθηκεύσει δεδομένα από το διαδίκτυο στον τοπικό του δίσκο κατά την εκτέλεση των υπολογισμών. Εναλλακτικά, το σύστημα μπορεί να είχε αποθηκεύσει τα απαραίτητα δεδομένα πριν την άφιξη του κινητού υπολογιστή του χρήστη στο χώρο. Όταν ο κινητός υπολογιστής τελικά απομακρύνεται, οι συνδέσεις με το αναπληρωματικό σύστημα καταργούνται και οποιαδήποτε δεδομένα που είχαν αποθηκευθεί εξ’ ονόματός του απορρίπτονται. Η κυβερνοβοσκή δημιουργεί πολλά ανοικτά ερευνητικά ερωτήματα. Παρακάτω παρουσιάζουμε μερικά από αυτά:

• Πώς κάποιος ανακαλύπτει την παρουσία αναπληρωματικών συστημάτων και υποδομών; Ποια από τις ήδη υπάρχουσες τεχνολογίες (JINI, UPnP, Bluetooth) είναι καλύτερη; Μπορεί κάποιος να δημιουργήσει έναν μηχανισμό ανακαλύψεων (discovery mechanism) που ενσωματώνει όλες τις τεχνολογίες για να επιτευχθεί μεγαλύτερη ευελιξία;

• Πώς κάποιος καθιερώνει ένα κατάλληλο επίπεδο εμπιστοσύνης (trust) με ένα αναπληρωματικό σύστημα; Πόσο εφαρμόσιμη και χρήσιμη είναι η έννοια του caching trust [4];

• Πώς γίνεται η κατανομή φόρτου (load balancing) στα αναπληρωματικά συστήματα; Πόσο σχετικές είναι οι προηγούμενες εργασίες στην κατανομή φόρτου σε δίκτυα υπολογιστών;

1.1.3.3 Στρατηγική Προσαρμογής (Adaptation Strategy)

Η προσαρμογή (adaptation) είναι απαραίτητη όταν υπάρχει ένας αποτυχημένος συνδυασμός μεταξύ της προσφοράς και της ζήτησης ενός πόρου. Ο πόρος μπορεί να είναι το εύρος ζώνης ενός ασύρματου δικτύου, η ενέργεια της μπαταρίας ενός κινητού υπολογιστή, η μνήμη, κτλ. Υπάρχουν τρεις εναλλακτικές στρατηγικές για την προσαρμογή στον διάχυτο υπολογισμό.

Κατ' αρχάς, ένας χρήστης (client) μπορεί από μόνος του να ρυθμίσει τις εφαρμογές του έτσι ώστε αυτές να χρησιμοποιούν λιγότερο κάποιον ανεπαρκή πόρο. Αυτή η ρύθμιση όμως μειώνει την απόδοση και την ποιότητα μιας εφαρμογής. Το Odyssey [5, 6] είναι ένα παράδειγμα ενός συστήματος που χρησιμοποιεί αυτήν την στρατηγική. Δεύτερον, ένας χρήστης μπορεί να ζητήσει από το περιβάλλον να του εγγυηθεί ένα ορισμένο επίπεδο ποιότητας ενός πόρου. Αυτή είναι η προσέγγιση που χρησιμοποιείται από τα Quality of Service (QoS) συστήματα [7]. Τρίτον, το σύστημα μπορεί να προτείνει μια διορθωτική κίνηση (δράση) στο χρήστη. Εάν ο χρήστης ενεργήσει σύμφωνα με τις προτάσεις που του υποδεικνύει το σύστημα είναι πιθανό ότι θα υπάρχουν αρκετοί πόροι για να ικανοποιηθούν οι ανάγκες του. Ένα παράδειγμα αυτής της προσέγγισης περιγράφηκε νωρίτερα στο *Σενάριο 1*, όπου το σύστημα συμβούλεψε τη Jane να κατευθυνθεί στην πύλη 15 διότι εκεί θα είχε επαρκές ασύρματο εύρος ζώνης. Και οι τρεις στρατηγικές προσαρμογής είναι σημαντικές αλλά πολλά ερωτήματα ανακύπτουν:

• Πώς ένα σύστημα επιλέγει μεταξύ των στρατηγικών προσαρμογής; Ποιους παράγοντες θα έπρεπε να λάβει υπόψη μια διαδικασία απόφασης; Πώς θα έπρεπε οι διαφορετικοί παράγοντες να σταθμιστούν; Ποιο ρόλο θα έπρεπε, ενδεχομένως, ο χρήστης να διαδραματίσει στη λήψη της απόφασης;

• Ποιες είναι οι πιο κατάλληλες πολιτικές ελέγχου αποδοχής (admission control policies) όταν υπάρχουν ανταγωνιστικά αιτήματα από τους πολλούς χρήστες; Ποιοι πόροι εκτός από το εύρος ζώνης των ασύρματων δικτύων είναι σημαντικοί και χρήσιμο σε έναν ευφυή χώρο;

• Είναι εφικτή η προσαρμογή που χρησιμοποιεί τις διορθωτικές ενέργειες; Μήπως οι χρήστες βρίσκουν μια τέτοια στρατηγική παρεισφρητική ή ενοχλητική; Ποια είναι τα απαραίτητα πρότυπα προγραμματισμού που υποστηρίζουν τις διορθωτικές ενέργειες;

1.1.3.4 Διαχείριση Ενέργειας (Energy Management)

Περίπλοκες διαδικασίες όπως η προδραστικότητα και η αυτόματα ρυθμιζόμενη συμπεριφορά (self-tuning) αυξάνουν τις απαιτήσεις για ενέργεια στον κινητό υπολογιστή ενός χρήστη. Συγχρόνως, η πίεση να γίνουν αυτοί οι υπολογιστές ελαφρύτεροι και πιο συμπαγείς θέτει αυστηρούς περιορισμούς στην χωρητικότητα των μπαταριών. Η πρόοδος στην τεχνολογία των μπαταριών και τα χαμηλής ισχύος ολοκληρωμένα κυκλώματα δεν μπορούν από μόνα τους να λύσουν το πρόβλημα. Πρέπει να ληφθούν υπόψη και τα υψηλότερα επίπεδα του συστήματος [8, 9].

Πώς όμως κάποιος περιλαμβάνει τα υψηλότερα επίπεδα ενός συστήματος στη διαχείριση της ενέργειας; Ένα παράδειγμα είναι η διαχείριση μνήμης που λαμβάνει υπόψη την ενέργεια [10]. Εδώ το λειτουργικό σύστημα ελέγχει δυναμικά το ποσό της φυσικής μνήμης που πρέπει να ανανεωθεί. Ένα άλλο παράδειγμα είναι η προσαρμογή [5], όπου μεμονωμένες εφαρμογές (ελεγχόμενες από το λειτουργικό σύστημα) αλλάζουν το τρόπο λειτουργίας τους ώστε να καταναλώνουν λιγότερη ενέργεια. Πολλές ερωτήσεις ερευνητικού ενδιαφέροντος ακολουθούν:

• Με ποιους άλλους τρόπους μπορούν τα υψηλότερα επίπεδα ενός συστήματος να συμβάλλουν στην διαχείριση της ενέργειας; Ποια είναι τα πλεονεκτήματα και μειονεκτήματα αυτών των προσεγγίσεων; Πότε μία μέθοδος είναι προτιμότερη από κάποια άλλη;

• Πώς η υψηλού επιπέδου διαχείριση της ενέργειας προσκρούει στο στόχο της αορατότητας στον διάχυτο υπολογισμό; Πόσο παρεισφρητικές ή ενοχλητικές βρίσκουν οι χρήστες τέτοιες τεχνικές;

• Μπορεί η γνώση της πρόθεσης των χρηστών να χρησιμοποιηθεί στη διαχείριση της ενέργειας; Σε αυτή την περίπτωση, πόσο εύρωστη είναι αυτή η προσέγγιση;

• Μπορούν οι ευφυείς χώροι και τα αναπληρωματικά συστήματα να χρησιμοποιηθούν για να μειώσουν την ενεργειακή απαίτηση σε έναν κινητό υπολογιστή; Ποιες είναι οι πιθανές προσεγγίσεις;

• Ποιος είναι ο ρόλος της απομακρυσμένης εκτέλεσης στην επέκταση της ζωής των μπαταριών; Κάτω από ποιες περιστάσεις η ενέργειά της μπαταρίας υπερβαίνει το ενεργειακό κόστος της ασύρματης επικοινωνίας; Μπορεί ένα σύστημα να τα προβλέψει αυτά;

1.1.3.5 Επίγνωση πλαισίου (Context Awareness)

Ένα σύστημα διάχυτου υπολογισμού που προσπαθεί να είναι ελάχιστα παρεισφρητικό πρέπει να είναι context-aware. Με άλλα λόγια, πρέπει να έχει γνώση της κατάστασης του χρήστη και του περιβάλλοντος χώρου ώστε να τροποποιεί κατάλληλα την συμπεριφορά του. Οι πληροφορίες που αφορούν ένα χρήστη (πληροφορίες πλαισίου) μπορεί να είναι αρκετά πλούσιες και να αποτελούνται από ιδιότητες όπως φυσική θέση, φυσική κατάσταση (θερμοκρασία του σώματος και παλμοί της καρδιάς), συναισθηματική κατάσταση (θυμωμένος ή ήρεμος), καθημερινή συμπεριφορά, κτλ. Εάν δίνονταν αυτές οι πληροφορίες σε έναν άνθρωπο, θα λάμβανε αποφάσεις με προδραστικό τρόπο, προλαμβάνοντας και προβλέποντας τις ανάγκες των χρηστών. Στη λήψη αυτών των αποφάσεων, ο άνθρωπος δεν θα ενοχλούσε το χρήστη εκτός και αν υπήρχε μια επείγουσα περίπτωση. Μπορεί ένα σύστημα διάχυτου υπολογισμού να μιμηθεί έναν τέτοιο ανθρώπινο βοηθό; Η δημιουργία ενός context-aware συστήματος απαιτεί την αντιμετώπιση πολλών ζητημάτων. Π.χ.

• Πώς μοντελοποιούνται όλες αυτές οι πληροφορίες που αφορούν το χρήστη; Πώς αυτές οι πληροφορίες συνδυάζονται με την κατάσταση του συστήματος και των εφαρμογών; Πού αποθηκεύονται οι πληροφορίες;

• Ποιες είναι οι ελάχιστες υπηρεσίες που ένα περιβάλλον πρέπει να παρέχει ώστε να θεωρείται context-aware; Είναι οι ιστορικές πληροφορίες χρήσιμες;

• Ποιες είναι οι σχετικές αξίες των διαφορετικών τεχνολογιών προσδιορισμού; Κάτω από ποιες περιστάσεις θα έπρεπε να χρησιμοποιηθεί η καθεμία; Θα έπρεπε οι πληροφορίες θέσης να αντιμετωπιστούν ακριβώς όπως οποιεσδήποτε άλλες πληροφορίες, ή θα έπρεπε να αντιμετωπίζονται διαφορετικά; Είναι χρήσιμα τα ιστορικά δεδομένα;

1.1.3.6 Προδραστικότητα και Διαφάνεια

Η προδραστικότητα μπορεί να θεωρηθεί σαν δίκοπο μαχαίρι. Ένα προδραστικό σύστημα αν δεν είναι προσεκτικά σχεδιασμένο, μπορεί να ενοχλεί το χρήστη και να μην υπάρχει πλέον η ιδέα της αορατότητας. Πώς μπορεί να βρεθεί μία μέση λύση; Η αυτόματα ρυθμιζόμενη συμπεριφορά (self-tuning) είναι ένα σημαντικό εργαλείο σε αυτήν την προσπάθεια. Η ανάγκη και η ανοχή ενός χρήστη για την προδραστικότητα είναι στενά συνδεδεμένες με την πείρα του και με την οικειότητά του με το περιβάλλον.

Ένα σύστημα που μπορεί να συμπεράνει αυτούς τους παράγοντες με την παρατήρηση της συμπεριφοράς των χρηστών μπορεί να βρει τη μέση λύση. Το ιδανικό σύστημα είναι το διαφανές σύστημα. Για παράδειγμα η αποθήκευση (caching) είναι ελκυστική στα κατανεμημένα συστήματα αρχείων επειδή είναι απολύτως διαφανής. Αρκετά λεπτά προβλήματα προκύπτουν στο σχεδιασμό ενός συστήματος που προσπαθεί να είναι προδραστικό αλλά ταυτόχρονα και διαφανές. Παραδείγματα:

• Πώς λαμβάνονται υπόψη οι μεμονωμένες προτιμήσεις και οι ανοχές των χρηστών; Είναι στατικές ή αλλάζουν δυναμικά;

• Μπορεί κάποιος να παρέχει συστηματικές οδηγίες σχεδιασμού στους σχεδιαστές εφαρμογών; Μπορεί να τοποθετηθούν μηχανισμοί εξισορρόπησης στις υπάρχουσες εφαρμογές;

1.1.3.7 Ιδιωτικότητα και Εμπιστοσύνη (Privacy and Trust)

Το θέμα της ιδιωτικότητας, που αποτελεί ένα πρόβλημα στα κατανεμημένα συστήματα και στον κινητό υπολογισμό, περιπλέκεται πιο πολύ στον διάχυτο υπολογισμό. Μηχανισμοί όπως ο εντοπισμός της θέσης, οι ευφυείς χώροι, και η χρήση των αναπληρωματικών συστημάτων (surrogate systems) παρακολουθούν τους χρήστες σε μόνιμη βάση. Καθώς ο χρήστης εξαρτάται όλο και πιο πολύ από ένα σύστημα διάχυτου υπολογισμού, το σύστημα γίνεται πιο πεπειραμένο σχετικά με τις μετακινήσεις του συγκεκριμένου χρήστη, τη συμπεριφορά του και τις συνήθειές του. Η εκμετάλλευση αυτών των πληροφοριών είναι κρίσιμη για την επίτευξη επιτυχούς προδραστικότητας και αυτόματα ρυθμιζόμενης συμπεριφοράς (self-tuning). Η περίπτωση για σοβαρή απώλεια της ιδιωτικότητας μπορεί να αποτρέψει τους πεπειραμένους χρήστες από τη χρησιμοποίηση ενός συστήματος διάχυτου υπολογισμού. Επίσης το σύστημα πρέπει να είναι βέβαιο για την ταυτότητα του χρήστη πριν ανταποκριθεί στα αιτήματά του. Είναι μια δύσκολη πρόκληση για να καθιερωθεί αυτή η αμοιβαία εμπιστοσύνη με τρόπο που να είναι ελάχιστα παρεισφρητικός ώστε να επιτυγχάνεται η αορατότητα.

Η ιδιωτικότητα και η εμπιστοσύνη είναι πιθανό να αντιμετωπίσουν προβλήματα στον διάχυτο υπολογισμό. Πολλά ερευνητικά ερωτήματα ανακύπτουν:

• Πώς μπορεί να βρεθεί μία μέση λύση μεταξύ της μονοκόμματης συμπεριφοράς του συστήματος και της ανάγκης να προειδοποιηθούν οι χρήστες για πιθανή απώλεια της ιδιωτικότητας; Ποιοι είναι οι μηχανισμοί, οι τεχνικές και οι σχεδιαστικές αρχές που

σχετίζονται με αυτό το πρόβλημα; Πόσο συχνά θα έπρεπε το σύστημα να υπενθυμίζει σε έναν χρήστη ότι οι ενέργειές του καταγράφονται;

• Ποιες είναι οι καταλληλότερες τεχνικές για πιστοποίηση (authentication) σε ένα περιβάλλον διάχυτου υπολογισμού; Τεχνικές όπως ο Kerberos [11] είναι επαρκείς ή απαιτούνται πιο περίπλοκες τεχνικές όπως η βιομετρική πιστοποίηση [12]; Ποιο ρόλο μπορεί να παίξουν οι έξυπνες κάρτες (smart cards);

1.1.3.8 Στρωματοποίηση (Layering)

Ένα θέμα που θίξαμε προηγουμένως είναι η συγχώνευση των πληροφοριών από τα διαφορετικά επίπεδα (στρώματα) ενός συστήματος για να παραχθεί μια αποτελεσματική απάντηση. Π.χ., το *Σενάριο 1* παρουσίασε την αξία του συνδυασμού των πληροφοριών χαμηλού επιπέδου (εύρος ζώνης δικτύου) με τις υψηλού επιπέδου πληροφορίες πλαισίου (ώρα αναχώρησης της πτήσης). Η προδραστικότητα και η προσαρμογή που βασίζεται σε διορθωτικές ενέργειες φαίνεται ότι ευνοούν την ανταλλαγή περισσότερων πληροφοριών μεταξύ των επιπέδων από ότι τα σημερινά τυπικά συστήματα.

Η στρωματοποίηση (layering) διαχωρίζει την αφαίρεση (abstraction) από την εφαρμογή. Συμβάλλει επίσης στην τυποποίηση, δεδομένου ότι ενθαρρύνει τη δημιουργία τμημάτων λογισμικού. Η αποσύνθεση ενός σύνθετου συστήματος σε επίπεδα (στρώματα) ή ενότητες δεν είναι εύκολη διαδικασία, και παραμένει πιο πολύ μια τέχνη παρά μια επιστήμη. Πολλές ερευνητικές ερωτήσεις ακολουθούν:

• Πώς μπορούν τα οφέλη της στρωματοποίησης να διατηρηθούν προσαρμόζοντας τα στις ανάγκες του διάχυτου υπολογισμού; Ποιος είναι ο αντίκτυπος αυτών των προσαρμογών στην αποδοτικότητα;

• Τα υπάρχοντα στρώματα επεκτείνονται καλύτερα για τον διάχυτο υπολογισμό με τη διεύρυνση των αρχικών διεπαφών τους ή με τη δημιουργία δευτεροβάθμιων διεπαφών (SNMP [13]);

• Υπάρχουν συστηματικές οδηγίες που μπορούμε να προσφέρουμε για να εξασφαλίσουμε συμβατότητα με τις ανάγκες του διάχυτου υπολογισμού όταν δημιουργούμε ένα στρώμα;

1.1.3.9 Συμπεράσματα

Ο διάχυτος υπολογισμός αποτελεί μία ανοικτή ερευνητική περιοχή με πολλά ερωτήματα και προβλήματα που ζητούν επίλυση. Θα πρέπει επίσης να εξετάσουμε τις ερευνητικές προκλήσεις σε περιοχές διαφορετικές από την περιοχή των υπολογιστών. Αυτές οι περιοχές, που περιλαμβάνουν την αλληλεπίδραση ανθρώπου-μηχανής, τους πράκτορες λογισμικού (software agents), τα έμπειρα συστήματα και την τεχνητή νοημοσύνη, θα πρέπει να ενσωματωθούν με τα είδη των υπολογιστικών συστημάτων που συζητήσαμε πιο πάνω. Σύμφωνα με τον M. Weiser για να επιτευχθεί αυτό απαιτείται δημιουργικότητα και προσπάθεια από πολλούς ανθρώπους για πολλά έτη.

1.2 Περιβάλλοντα Κινητού Υπολογισμού και Εφαρμογές

1.2.1 Κινητός Υπολογισμός

Ο κινητός υπολογισμός (mobile computing) [14] μπορεί να οριστεί ως ένα υπολογιστικό περιβάλλον πάνω από τη φυσική κινητότητα. Ο χρήστης του κινητού υπολογιστικού περιβάλλοντος μπορεί να έχει πρόσβαση σε δεδομένα, πληροφορίες ή άλλα λογικά αντικείμενα από οποιαδήποτε συσκευή σε οποιοδήποτε δίκτυο, ενώ κινείται. Το κινητό υπολογιστικό σύστημα επιτρέπει στον χρήστη να εκτελέσει μια εργασία από οπουδήποτε, χρησιμοποιώντας μια υπολογιστική συσκευή στο χώρο των δημόσιων (διαδίκτυο), εταιρικών (επιχειρηματικές πληροφορίες) και προσωπικών δεδομένων (ιατρικό ιστορικό, βιβλίο διευθύνσεων). Όταν βρίσκεται σε κίνηση, η προτιμώμενη συσκευή είναι μια φορητή συσκευή, ενώ στο σπίτι ή στο γραφείο η συσκευή θα μπορούσε να είναι ένας επιτραπέζιος υπολογιστής.

Ο κινητός υπολογισμός χρησιμοποιείται σε διαφορετικά περιβάλλοντα με διαφορετικά ονόματα. Τα πιο συνηθισμένα ονόματα είναι:

Κινητός υπολογισμός (mobile computing): Το περιβάλλον υπολογισμού είναι κινητό και μετακινείται μαζί με το χρήστη: Αυτό είναι παρόμοιο με τον αριθμό τηλεφώνου ενός GSM (Παγκόσμιο Σύστημα Κινητών Επικοινωνιών) τηλεφώνου, που μετακινείται με το τηλέφωνο. Το αποσυνδεδεμένο (τοπικό) και σε πραγματικό χρόνο (απομακρυσμένο) υπολογιστικό περιβάλλον θα μετακινηθεί μαζί με το χρήστη. Σε λειτουργία πραγματικού χρόνου, ο χρήστης θα είναι σε θέση να χρησιμοποιήσει όλα τα απομακρυσμένα δεδομένα και τις υπηρεσίες σε απευθείας σύνδεση (online).

Πληροφορία οπουδήποτε, οποτεδήποτε (anywhere, anytime information): Αυτός είναι ο γενικός ορισμός της πανταχού παρουσίας, όπου οι πληροφορίες είναι διαθέσιμες οπουδήποτε, όλη την ώρα.

Εικονικό οικείο περιβάλλον (virtual home environment – VHE) : Το VHE ορίζεται ως ένα περιβάλλον σε ένα ξένο δίκτυο, έτσι ώστε οι κινητοί χρήστες να μπορούν να βιώσουν την ίδια εμπειρία χρήσης του υπολογιστή που έχουν στο υπολογιστικό περιβάλλον του σπιτιού ή της εταιρίας τους. Για παράδειγμα, κάποιος θα ήθελε να ενεργοποιήσει τη θερμάστρα του δωματίου του, όταν βρίσκεται περίπου 15 λεπτά μακριά από το σπίτι.

Νομαδικός υπολογισμός (nomadic computing): Το υπολογιστικό περιβάλλον είναι νομαδικό και κινείται μαζί με τον κινητό χρήστη. Αυτό ισχύει τόσο για τις τοπικές όσο και για τις απομακρυσμένες υπηρεσίες.

Διάχυτος Υπολογισμός (pervasive computing): Ένα υπολογιστικό περιβάλλον, το οποίο είναι διάχυτο στη φύση του και μπορεί να γίνει διαθέσιμο σε οποιοδήποτε περιβάλλον.

Πανταχού παρών υπολογισμός (ubiquitous computing): Ένα «αόρατο» (κανείς δεν αντιλαμβάνεται την παρουσία του) υπολογιστικό περιβάλλον μπορεί να βρίσκεται οπουδήποτε. Ο χρήστης είναι σε θέση να χρησιμοποιεί τόσο τοπικές όσο και απομακρυσμένες υπηρεσίες.

Παγκόσμια Φορητότητα Υπηρεσιών (global service portability): Η πραγμάτωση μιας υπηρεσίας ως φορητή και διαθέσιμη σε κάθε περιβάλλον. Κάθε υπηρεσία, κάθε περιβάλλοντος είναι διαθέσιμη σε παγκόσμιο επίπεδο.

Φορετοί Υπολογιστές (wearable computers): Οι φορετοί υπολογιστές μπορούν να φορεθούν από τον άνθρωπο, όπως ένα καπέλο, τα παπούτσια ή τα ρούχα. Οι φορετοί υπολογιστές πρέπει να έχουν κάποια πρόσθετα χαρακτηριστικά σε σύγκριση με τις καθιερωμένες κινητές συσκευές. Οι φορετοί υπολογιστές είναι πάντα ενεργοί, λειτουργούν εν κινήσει και έχουν αντίληψη του περιβάλλοντός τους (με διάφορους τύπους αισθητήρων).

1.2.1.1 Το ασύρματο είναι και κινητό ή το κινητό είναι και ασύρματο;

Η ασύρματη συνδεσιμότητα, ήταν ένας πολύ καλός τρόπος που ανακαλύφθηκε, ώστε οι συσκευές κινητού υπολογισμού να επικοινωνούν με άλλες συσκευές του δικτύου. Στην πραγματικότητα, αυτή είναι μια μεγάλη πηγή σύγχυσης μεταξύ των ασύρματων επικοινωνιών και του κινητού υπολογισμού. Οι συσκευές κινητού υπολογισμού δε χρειάζεται να είναι ασύρματες. Φορητοί υπολογιστές, αριθμομηχανές, ηλεκτρονικά ρολόγια και πολλές άλλες συσκευές, είναι όλες συσκευές κινητού υπολογισμού. Καμία από αυτές δε χρησιμοποιεί υποχρεωτικά, κάποιο είδος ασύρματης επικοινωνίας για να συνδεθεί σε ένα δίκτυο. Τα ασύρματα συστήματα επικοινωνιών είναι ένα είδος συστήματος επικοινωνίας. Αυτό που διακρίνει ένα ασύρματο σύστημα επικοινωνίας από τα άλλα είναι ότι το κανάλι επικοινωνίας είναι ο ίδιος ο χώρος. Τα ασύρματα συστήματα επικοινωνίας χρησιμοποιούνται συχνά στα συστήματα κινητού υπολογισμού για τη διευκόλυνση της σύνδεσης με το δίκτυο, αλλά δεν είναι συστήματα κινητού υπολογισμού.

Τα περισσότερα συστήματα κινητού υπολογισμού, μπορούν να συνδεθούν με το δίκτυο μέσω ενσύρματων ή ασύρματων συνδέσεων. Λόγω της φύσης τους, η σύνδεση ενός κινητού συστήματος με το δίκτυο γίνεται ολοένα και περισσότερο μέσω ασύρματων συστημάτων επικοινωνίας, κάτι που τείνει να γίνει ένα στοιχείο διάκρισης μεταξύ κινητών και σταθερών συστημάτων. Αν δεν αποτελεί απαίτηση να είναι ασύρματο ένα κινητό σύστημα, τα περισσότερα κινητά συστήματα είναι ασύρματα.

Επίσης, αν και είναι σημαντικό να κατανοήσουμε ότι τα σταθερά και κινητά υπολογιστικά συστήματα είναι εγγενώς διαφορετικά, αυτό δεν σημαίνει ότι δεν έχουν κοινά σημεία. Βασιζόμαστε σε υπάρχουσες τεχνολογίες, λογισμικό και τεχνικές που χρησιμοποιούνται για τα σταθερά συστήματα, όπου υπάρχουν αυτά τα κοινά στοιχεία ή υπάρχει μια λογική επέκταση μιας σταθερής τεχνικής ή τεχνολογίας που θα την καταστήσει κινητή.

1.2.1.2 Λειτουργίες κινητού υπολογισμού

Ένα υπολογιστικό περιβάλλον μπορεί να οριστεί ως κινητό αν υποστηρίζει ένα ή περισσότερα από τα ακόλουθα χαρακτηριστικά:

Κινητότητα χρήστη (user mobility): Ο χρήστης θα πρέπει να μπορεί να μετακινείται από τη μία φυσική τοποθεσία σε μια άλλη και να χρησιμοποιεί την ίδια υπηρεσία. Η υπηρεσία θα μπορούσε να βρίσκεται στο οικιακό δίκτυο ή σε ένα απομακρυσμένο δίκτυο. Για παράδειγμα, ένας χρήστης θα μπορούσε να μετακινηθεί από το Λονδίνο στη Νέα Υόρκη και να χρησιμοποιεί το Διαδίκτυο, για να έχει πρόσβαση σε μια εταιρική εφαρμογή, με τον ίδιο τρόπο που το χρησιμοποιεί στο γραφείο ή στο σπίτι.

Κινητότητα δικτύου (network mobility): Ο χρήστης θα πρέπει να μπορεί να μετακινείται από το ένα δίκτυο σε κάποιο άλλο και να κάνει χρήση της ίδιας υπηρεσίας. Για παράδειγμα, ένας χρήστης θα μπορούσε να μετακινείται από το Χονγκ Κονγκ στο Νέο Δελχί και να χρησιμοποιεί το ίδιο τηλέφωνο GSM για να έχει πρόσβαση σε μια εταιρική εφαρμογή μέσω WAP (Wireless Application Protocol – πρωτόκολλο ασύρματης εφαρμογής). Στο οικιακό του δίκτυο, χρησιμοποιεί την υπηρεσία αυτή μέσω GPRS (General Packet Radio Service - Γενική Ραδιοϋπηρεσία Μεταγωγής Πακέτου), ενώ στο Δελχί αποκτά πρόσβαση μέσω του δικτύου GSM.

Κινητότητα φορέα (bearer mobility): Ο χρήστης θα πρέπει να μπορεί να μετακινείται από τον ένα φορέα στον άλλο και να χρησιμοποιεί την ίδια υπηρεσία. Για παράδειγμα, έστω ότι ένας χρήστης χρησιμοποιούσε μια υπηρεσία μέσω φορέα WAP στο οικιακό του δίκτυο στην Μπενγκαλούρου. Μετακομίζει στην Κοϊμπατόρε, όπου το WAP δεν υποστηρίζεται, και έτσι αλλάζει σε φορέα φωνής ή SMS (Short Message Service – υπηρεσία σύντομων γραπτών μηνυμάτων) για να αποκτήσει πρόσβαση στην ίδια εφαρμογή.

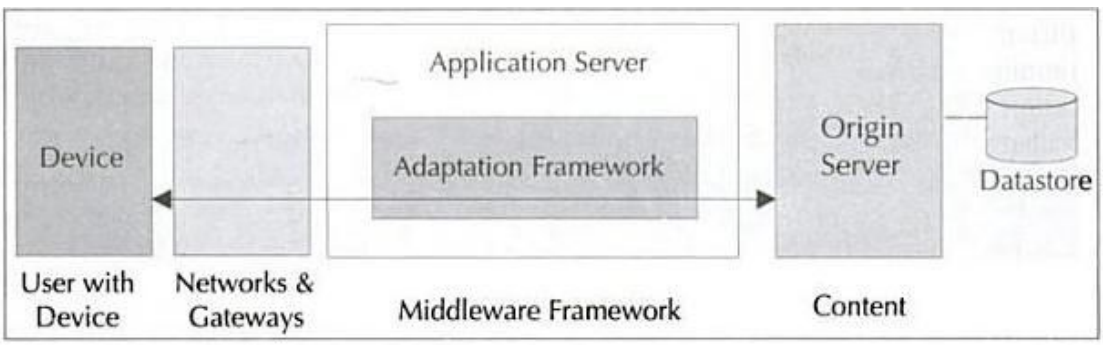
Κινητότητα συσκευής (device mobility): Ο χρήστης θα πρέπει να μπορεί να μετακινείται από τη μία συσκευή στην άλλη και να χρησιμοποιεί την ίδια υπηρεσία. Για παράδειγμα, αντιπρόσωποι πωλήσεων που χρησιμοποιούν επιτραπέζιο υπολογιστή στο οικιακό τους γραφείο, κατά τη διάρκεια της ημέρας, όταν βρίσκονται στο δρόμο, θα ήθελαν να χρησιμοποιήσουν τον υπολογιστή παλάμης τους (palmtop), για να έχουν πρόσβαση σε μια εφαρμογή.

Κινητότητα συνεδρίας (session mobility): Η συνεδρία ενός χρήστη θα πρέπει να μπορεί να μετακινηθεί από ένα περιβάλλον χρήστη-πράκτορα σε ένα άλλο. Για παράδειγμα, έστω ότι ένας χρήστης χρησιμοποιεί υπηρεσίες, μέσω ενός CDMA (Code Division Multiple Access - πολλαπλή προσπέλαση με διαίρεση κωδίκων) δικτύου ανταλλαγής πληροφοριών. Ο χρήστης βρίσκεται στο υπόγειο για να παρκάρει το αυτοκίνητο του και, έτσι, αποσυνδέεται από το δίκτυο CDMA. Πηγαίνει στο σπίτι και αρχίζει να χρησιμοποιεί τον επιτραπέζιο υπολογιστή του. Η ημιτελής συνεδρία της CDMA συσκευής, μεταφέρεται από την κινητή συσκευή στον επιτραπέζιο υπολογιστή.

Κινητότητα υπηρεσίας (service mobility): Ο χρήστης θα πρέπει να μπορεί να μεταβαίνει από μία υπηρεσία σε άλλη. Για παράδειγμα, έστω ότι ένας χρήστης γράφει ένα μήνυμα. Για να το ολοκληρώσει πρέπει να προσφύγει σε κάποιες άλλες πληροφορίες. Σε έναν επιτραπέζιο υπολογιστή, ο χρήστης απλά ανοίγει μια άλλη υπηρεσία και τις εναλλάσσει χρησιμοποιώντας τη γραμμή εργασιών. Ο χρήστης θα πρέπει να είναι σε θέση να εναλλάσσει υπηρεσίες σε ασύρματες συσκευές μικρού ίχνους, όπως και στον επιτραπέζιο υπολογιστή.

Κινητότητα κεντρικού υπολογιστή (host mobility): Η συσκευή του χρήστη μπορεί να είναι είτε πελάτης είτε διακομιστής. Στην περίπτωση κινητότητας κεντρικού υπολογιστή, θα πρέπει να ληφθεί μέριμνα για την κινητότητα του IP.

Οι λειτουργίες του κινητού υπολογισμού μπορούν να χωριστούν λογικά στα ακόλουθα μεγάλα τμήματα (Εικόνα 1.2):



Εικόνα 1.2: Οι λειτουργίες του κινητού υπολογισμού

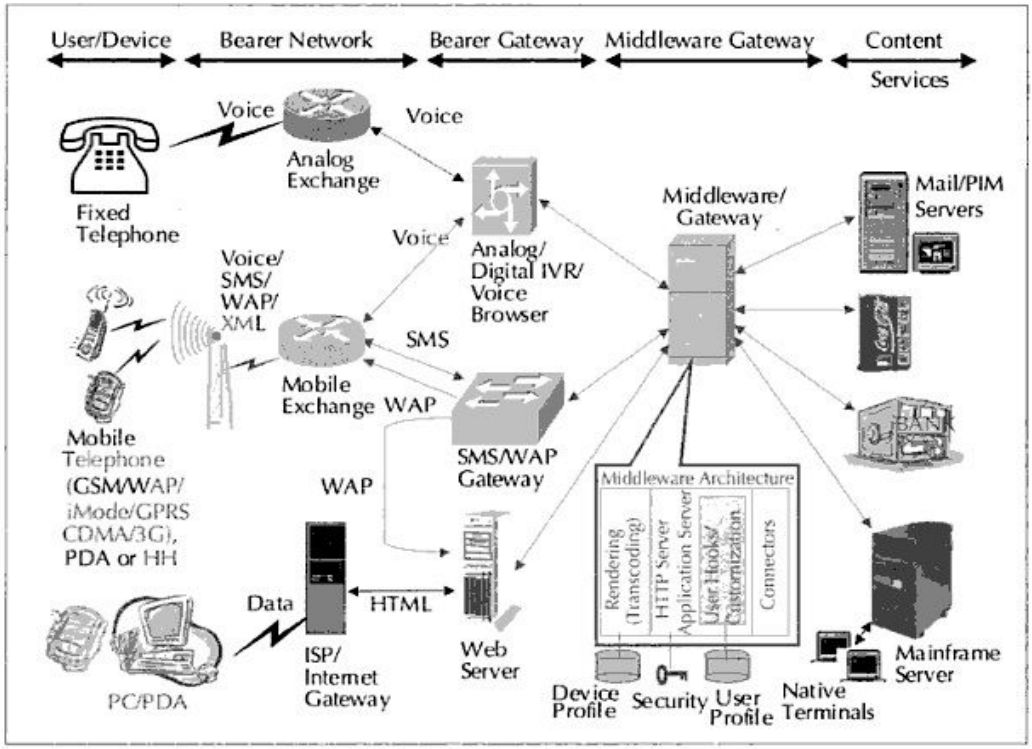
1. Χρήστης με συσκευή: Η συσκευή του χρήστη. Θα μπορούσε να είναι μια σταθερή συσκευή, όπως ένας επιτραπέζιος υπολογιστής στο γραφείο ή μια φορητή συσκευή όπως ένα κινητό τηλέφωνο. Παραδείγματα: φορητοί υπολογιστές (laptops), επιτραπέζιοι υπολογιστές, σταθερό τηλέφωνο, κινητά τηλέφωνα, ψηφιακή τηλεόραση με αποκωδικοποιητή, υπολογιστές παλάμης, υπολογιστές τσέπης, τερματικά χειρός, κ.λπ.

2. Δίκτυο: Κάθε κινητός χρήστης, χρησιμοποιεί διαφορετικά δίκτυα σε διαφορετικές τοποθεσίες και σε διαφορετικό χρόνο. Παραδείγματα: GSM, CDMA, iMode, Ethernet, ασύρματο LAN, Bluetooth κ.λπ.

3. Πύλη δικτύου (gateway): Είναι απαραίτητη για τη διασύνδεση διαφορετικών φορέων μεταφοράς. Αυτές οι πύλες μετατρέπουν ένα συγκεκριμένο φορέα μεταφοράς σε άλλο φορέα μεταφοράς. Παραδείγματα: Μπορούμε να προσπελάσουμε μια υπηρεσία, από σταθερό τηλέφωνο (με διεπαφή φωνής), πατώντας διάφορα πλήκτρα του τηλεφώνου. Αυτά τα πλήκτρα δημιουργούν σήματα DTMF (Dual Tone Multi Frequency - Πολυσυχνότητα Διπλού Τόνου). Αυτά τα αναλογικά σήματα μετατρέπονται σε ψηφιακά δεδομένα από την πύλη IVR (Interactive Voice Response – διαδραστική φωνητική απόκριση) για τη διασύνδεση με μια εφαρμογή υπολογιστή. Άλλα παραδείγματα είναι η πύλη WAP, η πύλη SMS κλπ.

4. Ενδιάμεσο λογισμικό (middleware): Είναι περισσότερο σα μια λειτουργία και όχι ένας ξεχωριστός ορατός κόμβος. Στο παρόν πλαίσιο, το ενδιάμεσο λογισμικό χειρίζεται την παρουσίαση και την απόδοση του περιεχομένου σε μια συγκεκριμένη συσκευή. Μπορεί να χειριστεί, επίσης, την ασφάλεια και την εξατομίκευση (personalization) για διαφορετικούς χρήστες.

5. Περιεχόμενο: Αυτός είναι ο τομέας όπου βρίσκονται ο πηγαίος εξυπηρετητής και το περιεχόμενο. Θα μπορούσε να είναι μια εφαρμογή, ένα σύστημα, ή ακόμα και μια συνάθροιση συστημάτων. Το περιεχόμενο μπορεί να είναι μαζικού, προσωπικού ή εταιρικού περιεχόμενου. Ο πηγαίος εξυπηρετητής έχει πρόσβαση στη βάση δεδομένων και τις συσκευές αποθήκευσης.



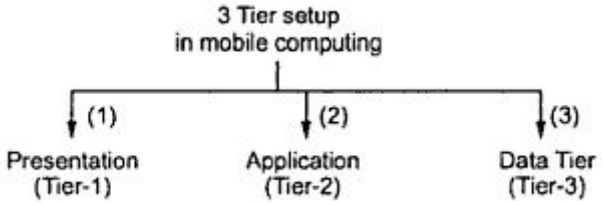
Εικόνα 1.3: Σχηματική αναπαράσταση ενός κινητού περιβάλλοντος

1.2.1.3 Συσκευές κινητού υπολογισμού

Μια συσκευή κινητού υπολογισμού μπορεί να είναι είτε μια υπολογιστική συσκευή είτε μια συσκευή επικοινωνίας. Στην κατηγορία υπολογιστικών συσκευών, μπορεί να είναι ένας επιτραπέζιος υπολογιστής, ένας φορητός υπολογιστής, ή ένας υπολογιστής παλάμης. Ως συσκευή επικοινωνίας, μπορεί να είναι μια σταθερή γραμμή τηλεφώνου, ένα κινητό τηλέφωνο ή μια ψηφιακή τηλεόραση. Η χρήση των εν λόγω συσκευών γίνεται όλο και πιο ενοποιημένη σε μια ροή εργασίας, όπου σταθερά και κινητά, υπολογιστές και συσκευές επικοινωνίας χρησιμοποιούνται από κοινού.

1.2.1.4 Δομή του κινητού υπολογισμού

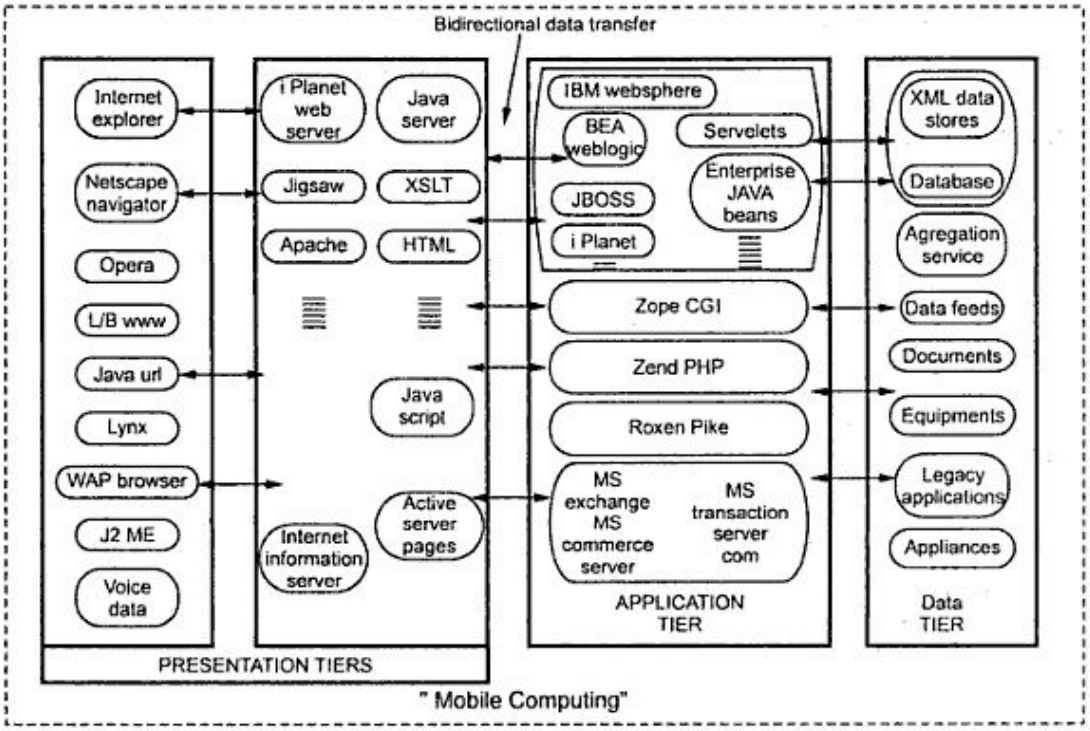
Η δομή τριών βαθμίδων [15], που απεικονίζεται στην Εικόνα 1.4, χρησιμοποιείται κυρίως για κινητά περιβάλλοντα. Πρόκειται για τις βαθμίδες παρουσίασης, εφαρμογής και δεδομένων.



Σχήμα 1.1 Βαθμίδες

Η βαθμίδα παρουσίασης αφορά στην αλληλεπίδραση με το χρήστη. Οι εφαρμογές της λειτουργούν στις συσκευές πελάτες. Αυτό το επίπεδο περιλαμβάνει, επίσης, τα προγράμματα περιήγησης στο Διαδίκτυο, καθώς και τα εξατομικευμένα προγράμματα πελάτη.

Η βαθμίδα εφαρμογής είναι γνωστή ως μέση βαθμίδα. Διαδραματίζει ζωτικό ρόλο για τις εφαρμογές σε ασύρματα τοπικά δίκτυα (LAN). Εκτελεί την επεξεργασία των δεδομένων που εισάγονται από τον χρήστη, τη συλλογή πληροφοριών και, στη συνέχεια, τη λήψη αποφάσεων, και είναι ανεξάρτητη από βάση δεδομένων (database independent).



Εικόνα 1.4: Δομική απεικόνιση κινητού υπολογισμού

1.2.2 Χαρακτηριστικά κινητών συστημάτων και υπηρεσιών

Οι ιδιαιτερότητες των συστημάτων κινητού υπολογισμού και επικοινωνιών αποτελούν πρόκληση τόσο για τους σχεδιαστές των συστημάτων, όσο και για τους σχεδιαστές μιας τυπικής μεθόδου που υποστηρίζει την κινητότητα. Οι ιδιαιτερότητες αυτές οφείλονται στις ιδιότητες του ασύρματου περιβάλλοντος, στα χαρακτηριστικά των κινητών συσκευών και στην κινητότητα των συσκευών αυτών. Παρακάτω αναφέρονται, ανά κατηγορία, αυτά τα ιδιαίτερα χαρακτηριστικά:

1.2.2.1 Περιορισμοί της ασύρματης επικοινωνίας

Μικρό εύρος ζώνης: Το εύρος ζώνης των ασύρματων δικτύων μπορεί να είναι μικρό, ιδιαίτερα στα ασύρματα δίκτυα ευρείας περιοχής και όταν υπάρχουν πολλοί χρήστες.

Μεταβλητό εύρος ζώνης: Ο δυναμικός χαρακτήρας των κινητών δικτύων μπορεί να έχει ως αποτέλεσμα τη δραματική μεταβολή του διαθέσιμου εύρους ζώνης μιας συσκευής. Για παράδειγμα, παρατηρείται μείωση όταν αυξάνεται ο αριθμός των χρηστών σε μια κυψέλη, ενώ έχουμε αύξηση όταν ο χρήστης μετακινείται από ένα δίκτυο χαμηλής ταχύτητας σε δίκτυο υψηλής ταχύτητας.

Αποσύνδεση: Οι ασύρματες συνδέσεις είναι πιθανό να χαθούν ή να μειωθεί το επίπεδο του σήματος λόγω της κινητικότητας του χρήστη. Για παράδειγμα οι χρήστες μπορεί να εξέλθουν από την ακτίνα κάλυψης της κυψέλης ή να εισέλθουν σε περιοχή με υψηλά επίπεδα παρεμβολών ή θορύβου.

Άλλοι περιορισμοί: Υψηλά επίπεδα καθυστέρησης δικτύου, συχνές απώλειες πακέτων δεδομένων, υψηλοί ρυθμοί απωλειών κ.α.

1.2.2.2 Χαρακτηριστικά κινητών συσκευών

Περιορισμοί πόρων: Οι περιορισμοί σε πόρους των κινητών συσκευών οφείλονται στο μικρό τους μέγεθος, στις μικρές δυνατότητες αποθήκευσης, στην υπολογιστική ισχύ και στη διάρκεια ζωής της μπαταρίας [16].

Συχνές αποσυνδέσεις από το δίκτυο: Για εξοικονόμηση πόρων (ειδικά λόγω περιορισμένης διάρκειας ζωής της μπαταρίας), οι χρήστες είναι απαραίτητο να αποσυνδέουν τις συσκευές τους από το δίκτυο. Επίσης, κατά τη διάρκεια μετακίνησης μιας συσκευής από μια κυψέλη σε άλλη (διαπομπή), είναι πιθανό να αποσυνδέεται από το δίκτυο για απρόβλεπτο χρονικό διάστημα.

Ασφάλεια: Λόγω της κινητότητας, μια συσκευή εκτίθεται σε μια σειρά από κινδύνους. Μια συσκευή είναι πιθανό να κλαπεί ή δεδομένα που περιέχονται σε αυτή να καταστραφούν μερικώς ή ολικώς. Οι κινητές συσκευές είναι πιο εύθραυστες και επιρρεπείς σε φυσικές καταστροφές σε σχέση με τις παραδοσιακές σταθερές συσκευές.

1.2.2.3 Περιορισμοί λόγω της κινητότητας

Ετερογένεια: Ένα περιβάλλον κινητών επικοινωνιών και υπολογισμού είναι δυνατό να αποτελείται από ετερογενή ασύρματα δίκτυα. Ένας κινητός χρήστης είναι πιθανό να χρειασθεί να διασχίσει τα όρια διαφορετικών δικτύων.

Δυναμικό περιβάλλον: Οι υπολογιστικές οντότητες (δηλ. οι κινητές συσκευές), λειτουργούν σε ένα άκρως δυναμικό περιβάλλον.

Μεταβλητή διαθεσιμότητα πόρων: Οι διαφορετικές περιοχές σε ένα κινητό περιβάλλον μπορεί να έχουν διαφορετικούς τύπους διαθέσιμων πόρων, που να μπορούν να χρησιμοποιηθούν από τους κινητούς χρήστες.

1.2.2.4 Ιδιότητες κινητών υπηρεσιών

Οι ιδιότητες που πρέπει να ικανοποιεί μια υπηρεσία – εφαρμογή, που λειτουργεί σε κινητό περιβάλλον, έχουν άμεση σχέση με τις παραπάνω ιδιαιτερότητες και ουσιαστικά αποτελούν τις διαφορές ανάμεσα στις εφαρμογές που προορίζονται για περιβάλλοντα κινητού υπολογισμού και για παραδοσιακά περιβάλλοντα κατανεμημένου υπολογισμού. Οι ιδιότητες αυτές είναι οι εξής:

- Αποσυνδεσιμότητα: Οι υπηρεσίες θα πρέπει να μπορούν να εκτελούνται σε κατάσταση αποσύνδεσης ή αδύναμης σύνδεσης [17].

- Επίγνωση θέσης: Μια εφαρμογή θα πρέπει να είναι σε θέση να επωφελείται από την κινητότητα και να παρέχει υπηρεσίες με βάση τη θέση [18].

- Εξάρτηση από το πλαίσιο: Οι κινητοί χρήστες μπορούν να προσπελάσουν και να χρησιμοποιήσουν πόρους και πληροφορίες, ανάλογα με τη θέση τους. Η διαθεσιμότητα πόρων εξαρτάται από τη θέση. Επίσης, μια υπηρεσία μπορεί να απαιτεί διαφορετικά είδη πόρων, ανάλογα με το περιβάλλον που εκτελείται [19].

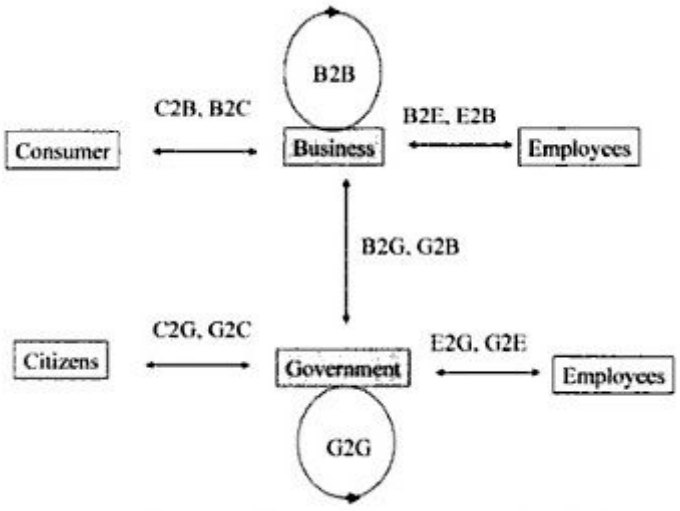
- Προσαρμοστικότητα: Η ετερογένεια στις συσκευές και τα δίκτυα απαιτεί από μια υπηρεσία να μπορεί να προσαρμόζεται στις μεταβολές του περιβάλλοντος [20].

- Συνεργασία: Οι κατανεμημένες υπολογιστικές οντότητες μιας υπηρεσίας θα πρέπει να συντάσσονται και να συνεργάζονται για την επίτευξη των στόχων της υπηρεσίας [21].

- Ασφάλεια: Οι κινητές συσκευές, οι υπηρεσίες που εκτελούνται σε αυτές και τα δεδομένα που είναι αποθηκευμένα σε αυτές τις συσκευές πρέπει να προστατεύονται από κακόβουλους χρήστες και λογισμικό.

1.2.3 Εφαρμογές

Οι εφαρμογές κινητού υπολογισμού [22] υποστηρίζουν το κινητό επιχειρείν (m-business), την κινητή διακυβέρνηση (m-government) και το κινητό ζην (mobile life). Οι εφαρμογές αυτές έχουν σημαντικό αντίκτυπο στον τρόπο που οι εταιρείες ασκούν τις δραστηριότητές τους και οι κυβερνητικές υπηρεσίες αντιμετωπίζουν το κοινό, μέσω της αξιοποίησης της κινητότητας. Ειδικότερα, αυτές οι εφαρμογές επιτρέπουν τις λειτουργίες «καταναλωτής προς επιχείρηση» (consumer to business - C2B), «επιχείρηση προς επιχείρηση» (business to business - B2B), «επιχείρηση προς εργαζόμενο» (business to employee - B2E), «πολίτης προς κυβερνητική υπηρεσία» (citizen to government - C2G), «επιχείρηση προς κυβερνητική υπηρεσία» (business to government - B2G), «κυβερνητική υπηρεσία προς κυβερνητική υπηρεσία» (government to government - G2G) και «κυβερνητική υπηρεσία προς εργαζόμενο» (government to employee - G2E), μεταξύ καταναλωτών, επιχειρηματικών μονάδων, κυβερνητικών υπηρεσιών και εργαζομένων (βλ. Εικόνα 1.5).



Εικόνα 1.5: Αλληλεπιδράσεις μεταξύ καταναλωτών, επιχειρήσεων, κυβερνητικών υπηρεσιών, εργαζόμενων και πολιτών

Οι περισσότερες εφαρμογές κινητού υπολογισμού δεν είναι, θεμελιωδώς, νέες εφαρμογές. Αντιθέτως, η πρόσβαση σε μια κινητή συσκευή μέσω ασύρματων δικτύων, αποτελεί μια άλλη διάσταση των περισσότερων υπαρχουσών εφαρμογών. Στην πραγματικότητα, η κινητή πρόσβαση προστίθεται σε υπάρχουσες εφαρμογές με τρόπο παρόμοιο με αυτόν που συνέβη η προσθήκη πρόσβασης στον Ιστό, στη δεκαετία του 1990. Η κινητή πρόσβαση δεν είναι το μοναδικό νέο χαρακτηριστικό των εφαρμογών κινητού υπολογισμού. Μπορούν, επίσης, να περιλαμβάνουν χαρακτηριστικά θέσης που εκμεταλλεύονται τη θέση των χρηστών, φωνητικές δυνατότητες για την υποστήριξη εφαρμογών φωνής και χαρακτηριστικά τηλεόρασης για τη διασύνδεσή τους με τηλεοράσεις. Αυτά τα χαρακτηριστικά ονομάζονται MPTV (Mobile, Positional, Television, Voice). Τέλος, υπάρχουν κάποιες βασικές κινητές εφαρμογές που χρησιμοποιούνται, με μικρές και αναγκαίες τροποποιήσεις, για το κινητό επιχειρείν και την κινητή διακυβέρνηση. Παραδείγματα αυτών των εφαρμογών είναι τα εξής:

- Ασύρματες υπηρεσίες μηνυμάτων
- Ασύρματες ιστοσελίδες και κινητές πύλες (m-portals)
- Το κινητό εμπόριο και οι παραλλαγές του
- Κινητά συστήματα διαχείρισης πελατειακών σχέσεων (m-CRM)
- Κινητά συστήματα διαχείρισης εφοδιαστικής αλυσίδας (m-SCM)
- Εξειδικευμένες εφαρμογές που αφορούν κινητούς πράκτορες και δίκτυα αισθητήρων

Πίνακας 1.1: Εφαρμογές κινητού υπολογισμού για το κινητό επιχειρείν, την κινητή διακυβέρνηση και το κινητό ζην

Εφαρμογές κινητού υπολογισμού	Κινητό Επιχειρείν (m-Business)	Κινητή Διακυβέρνηση (m-Government)	Κινητό Ζην (Mobile Life)
υπηρεσίες ασύρματων μηνυμάτων (email, SMS, MMS)	SMS για B2E	G2C, G2G, G2B, G2E	κοινωνικά μηνύματα
ασύρματος ιστός και κινητές πύλες (m-portals)	m-Marketing, επιχειρησιακές πύλες με ασύρματη πρόσβαση	κυβερνητικές ιστοσελίδες και πύλες με ασύρματη πρόσβαση	ιστοσελίδες προσβάσιμες από ασύρματες και κινητές πύλες όπως το Mobile Yahoo!

κινητό εμπόριο	κινητό εμπόριο	C2G για πληρωμή φόρων κτλ	Αγορά εισιτηρίου για το θέατρο
M-CRM	M-CRM	αδιευκρίνιστο	αδιευκρίνιστο
M-SCM	προμηθεύσεις υλικών σε B2B	προμηθεύσεις αγαθών σε G2C και G2B	αδιευκρίνιστο
εξειδικευμένες κινητές εφαρμογές	υπηρεσίες βάσει τοποθεσίας (LBS), κινητοί πράκτορες για κινητό εμπόριο και ασύρματα δίκτυα αισθητήρων (WSN)	κινητή ψηφοφορία (m-Voting) αμυντικές εφαρμογές κινητών πρακτόρων και WSNs	LBS για τον εντοπισμό των κοντινότερων εστιατορίων, κινηματογράφων κτλ

Δυνατότητες MPTV

Οι δυνατότητες κίνησης (mobile) επιτρέπουν στους κινητούς χρήστες, που συνήθως είναι συνδεδεμένοι μέσω ασύρματων δικτύων, την εκτέλεση καθημερινών δραστηριοτήτων, όπως η αναζήτηση στον Ιστό, η πρόσβαση σε απομακρυσμένες εφαρμογές και οι επιχειρησιακές συναλλαγές.

Οι δυνατότητες φωνής (voice) υποστηρίζουν τις φωνητικές επικοινωνίες μέσω κινητών συσκευών. Η φωνητική υποστήριξη στα κινητά τηλέφωνα και το φωνητικό εμπόριο αποτελούν τέτοια παραδείγματα.

Οι δυνατότητες θέσης (positional) υποστηρίζουν τη γεωγραφική θέση (τοποθεσία) για ορισμένες εφαρμογές. Για παράδειγμα, το τοποκεντρικό εμπόριο (location commerce) σου παρέχει πληροφορίες σχετικά με ευκαιρίες στην περιοχή της Βοστώνης, όταν βρίσκεσαι στη Βοστώνη.

Οι δυνατότητες τηλεόρασης (television) εκμεταλλεύονται την τηλεόραση για την πραγματοποίηση αγορών, την πλοήγηση στον Ιστό και άλλες λειτουργίες που κατά κανόνα διατίθενται σε κινητές συσκευές.

1.2.3.1 Ασύρματη ανταλλαγή μηνυμάτων

Η ασύρματη ανταλλαγή μηνυμάτων είναι μια ενδιαφέρουσα και σημαντική ανάπτυξη στις ασύρματες εφαρμογές. Η πρόσβαση σε σημαντικές πληροφορίες και επικοινωνίες είναι ζωτικής σημασίας για εκατομμύρια ανθρώπους. Οι κινητοί χρήστες περνούν ένα σημαντικό χρονικό διάστημα μακριά από το γραφείο τους και μπορεί να απογοητευτούν λόγω της αδυναμίας τους να μείνουν συνδεδεμένοι με τις πληροφορίες που οδηγούν την ημέρα τους.

Τέτοιες εφαρμογές για τα κυψελοειδή δίκτυα είναι:

- οι υπηρεσίες σύντομων γραπτών μηνυμάτων (SMS)
- οι υπηρεσίες μηνυμάτων πολυμέσων (MMS)
- το Blackberry από την Research In Motion (RIM)

Οι εφαρμογές αυτές χρησιμοποιούνται ευρέως στο κινητό επιχειρείν, την κινητή διακυβέρνηση και σε περιστάσεις του κινητού ζην. Για παράδειγμα, πολλές B2E, καθώς και C2G, λειτουργίες υποστηρίζονται μέσω SMS.

1.2.3.2 Το κινητό εμπόριο και οι παραλλαγές του

Το κινητό εμπόριο (m-commerce) περιγράφει το φαινόμενο της χρήσης ασύρματων κινητών συσκευών, όπως ψηφιακά τηλέφωνα και PDA, για την αναζήτηση στο Διαδίκτυο, την πρόσβαση σε δεδομένα και πληροφορίες και την πραγματοποίηση αγορών ή επιχειρησιακών συναλλαγών. Το κινητό εμπόριο τροφοδοτείται από την τεράστια δημοτικότητα κινητών συσκευών, όπως οι φορητοί προσωπικοί υπολογιστές, τα κινητά τηλέφωνα, τα PDA και οι υπολογιστές παλάμης. Ωστόσο, η συντριπτική πλειοψηφία των συσκευών και η χρήση τους, εξακολουθήσει να εξαρτάται από φορητούς υπολογιστές και προσωπικούς υπολογιστές, οι οποίοι και θα παραμείνουν οι καθιερωμένες συσκευές, που χρησιμοποιούνται για την πρόσβαση σε επιχειρησιακά δεδομένα και εφαρμογές.

Μία από τις κύριες προτάσεις αξίας του κινητού εμπορίου, είναι η ικανότητα εξατομίκευσης των εφαρμογών για τους μεμονωμένους χρήστες. Οι πάροχοι που επιθυμούν να προσφέρουν καλύτερες υπηρεσίες κινητού εμπορίου, χρειάζονται πληροφορίες όπως το όνομα, τη διεύθυνση, την τοποθεσία και τα στοιχεία χρέωσης

(τον αριθμό της πιστωτικής κάρτας ή του τραπεζικού λογαριασμού, για παράδειγμα) του χρήστη. Επιπλέον, μιας και το μέγεθος της οθόνης επηρεάζει το είδος των πληροφοριών που μπορούν να προβληθούν, προσδιορίζεται και ο τύπος συσκευής που χρησιμοποιείται για τη σύνδεση με την υπηρεσία.

Το **φωνητικό εμπόριο (v-commerce)** υποστηρίζει τους χρήστες που θέλουν να χρησιμοποιούν τηλέφωνα και άλλες φωνητικές συσκευές για τη διεξαγωγή ηλεκτρονικού εμπορίου. Για παράδειγμα, κατά την οδήγηση και το περπάτημα, είναι πιο εύκολη η χρήση ενός τηλεφώνου από έναν υπολογιστή. Τεχνολογίες και πρότυπα, όπως η Τηλεφωνία μέσω Διαδικτύου (Voice over IP - VoIP) και η Γλώσσα Σήμανσης Φωνής (Voice Markup Language - VML), διαδραματίζουν κεντρικό ρόλο στο φωνητικό εμπόριο.

Το **τοποκεντρικό εμπόριο (p-commerce)** γίνεται δημοφιλές για την παροχή υποστήριξης στους πελάτες, βάσει της γεωγραφικής τους θέσης (π.χ. μπορεί να σας δώσει πληροφορίες σχετικά με ευκαιρίες στην περιοχή της Βοστώνης, όταν είστε στη Βοστώνη). Τα συστήματα χρησιμοποιούν ένα Παγκόσμιο Σύστημα Εντοπισμού (Global Positioning System - GPS) για να εντοπίσουν τη θέση των πελατών. Εκτός από το GPS, η ασύρματη πρόσβαση βρίσκεται στο επίκεντρο της υποστήριξης της κινητότητας. Επιπλέον, οι κινητοί πράκτορες εφαρμόζονται ευρέως, για την υποστήριξη του τοποκεντρικού εμπορίου.

1.2.3.3 Κινητές επιχειρηματικές εφαρμογές (m-Portal, m-CRM, m-SCM)

Πολλές εφαρμογές κινητού υπολογισμού, όπως οι κινητές πύλες (m-portal), τα κινητά συστήματα διαχείρισης πελατειακών σχέσεων (m-CRM) και τα κινητά συστήματα διαχείρισης εφοδιαστικής αλυσίδας (m-SCM), αποτελούν μια κινητή ενεργοποίηση των ενδοεταιρικών εφαρμογών. Οι κινητές ενδοεταιρικές εφαρμογές (mobile enterprise business application - MEBA) προσθέτουν δυνατότητες κινητότητας στις κεντρικές εταιρικές εφαρμογές (ERP, SCM, CRM κ.λπ.), ώστε να είναι διαθέσιμες στους εργαζόμενους, συνεργάτες και πελάτες που μπορεί να βρίσκονται οπουδήποτε στον κόσμο. Η χρήση κινητών συσκευών, όπως φορητοί υπολογιστές, προσωπικοί ψηφιακοί βοηθοί (PDA) και ψηφιακά τηλεφώνα, με δυνατότητες Διαδικτύου και ασύρματης πρόσβασης σε δεδομένα, είναι ευρέως διαδεδομένη. Η ικανότητα υποστήριξης αυτών

των ιδιαίτερα κινητών συσκευών, ως μέρος μιας εκτεταμένης στρατηγικής για επιχειρήσεις, είναι κρίσιμη. Οι κινητές εφαρμογές ηλεκτρονικού επιχειρείν επιτρέπουν σε κινητούς πελάτες να διεξάγουν συναλλαγές με χρηματοπιστωτικές υπηρεσίες, παρόχους τηλεπικοινωνιών ή προμηθευτές προϊόντων της επιλογής τους.

Ένα ενδιαφέρον ζήτημα είναι οι υπεύθυνοι συγκέντρωσης περιεχομένου, δηλαδή οι επιχειρήσεις που σχεδιάζουν και διαχειρίζονται πύλες (που παρέχουν πληροφορίες για μια κατηγορία) ή κέντρα αναζήτησης, για να βοηθήσουν τους χρήστες να πλοηγηθούν στο Διαδίκτυο. Η λειτουργία αυτή είναι ιδιαίτερα σημαντική για τους κινητούς χρήστες, επειδή τα κινητά τηλέφωνα έχουν μικρές οθόνες και περιορισμένους μηχανισμούς εισροής δεδομένων. Συγκεκριμένα, δεν έχουν ποντίκι και το πληκτρολόγιο τους δεν είναι QWERTY (δηλαδή δεν είναι όπως ενός τερματικού υπολογιστή). Οι χρήστες χρειάζονται κινητές πύλες που απλοποιούν την αναζήτηση, αποφεύγουν την προβολή πολλών πληροφοριών και απαιτούν ελάχιστη συμβολή.

Οι κινητές ενδοεταιρικές εφαρμογές δημιουργούν πολλές ευκαιρίες, όπως η ανάπτυξη των επιχειρήσεων και των εσόδων, η υποστήριξη για νέους τύπους πελατών και η προσαρμογή σε διαφορετικά κοινωνικά μοντέλα, για το πως και που διεξάγονται επιχειρηματικές υποθέσεις. Οι εφαρμογές αυτές, όμως, εισαγάγουν πολλούς κινδύνους. Η ασφάλεια και η άνευ αδείας πρόσβαση είναι ένα φυσικό ζήτημα. Επιπλέον, οι κινητοί οργανισμοί πρέπει να διαχειρίζονται τα αποτελέσματα των φορητών υπολογιστών, καθώς και τα στοιχεία που τηρούνται σε κινητές συσκευές. Ειδικότερα, οι οργανώσεις αυτές πρέπει να χειριστούν το συγχρονισμό δεδομένων, τη διανομή αρχείων, τη διανομή του λογισμικού και των εργαλείων διαχείρισης συστημάτων, που χρειάζονται για κινητές εφαρμογές

Οι κινητές ενδοεταιρικές εφαρμογές, φυσικά, υποστηρίζουν το κινητό επιχειρείν, αλλά μπορούν να χρησιμοποιηθούν και στα πεδία της κινητής διακυβέρνησης και του κινητού ζην. Για παράδειγμα, οι κινητές πύλες μπορούν να χρησιμοποιηθούν από κρατικούς φορείς για C2G ή B2G εργασίες, όπου πρακτορεία παρέχουν μια πύλη για τους πολίτες. Οι κινητές πύλες για παράδειγμα, μπορούν να χρησιμοποιηθούν για την υγεία, καθώς και ψυχαγωγικούς σκοπούς: το WebMD είναι ένα παράδειγμα.

1.2.3.4 Εξειδικευμένες εφαρμογές με κινητούς πράκτορες και δίκτυα αισθητήρων

Για την πραγμάτωση εξειδικευμένων σκοπών, αναπτύσσονται και διάφορες εξειδικευμένες κινητές εφαρμογές. Τέτοια παραδείγματα αποτελούν οι κινητοί πράκτορες και τα ασύρματα δίκτυα αισθητήρων. Ένας πράκτορας είναι μια οντότητα λογισμικού (π.χ., ένα πρόγραμμα) που έχει κάποιο βαθμό αυτονομίας. Προβαίνει σε ενέργειες για λογαριασμό του χρήστη ή κάποιου άλλου προγράμματος, και αντιπροσωπεύει ή γνωρίζει τους στόχους και τις επιθυμίες του χρήστη. Υπό αυτήν την έννοια, ένας πράκτορας λογισμικού είναι παρόμοιος με έναν πράκτορα της πραγματικής ζωής, όπως ένας πράκτορας ασφάλειας ζωής, ένας πράκτορας ασφάλειας αυτοκινήτου, ένας ταξιδιωτικός πράκτορας, ένας κτηματομεσίτης κλπ. Όλοι οι πράκτορες, λογισμικό ή άνθρωποι, προβαίνουν σε μια σειρά ενεργειών για λογαριασμό ενός χρήστη (πελάτη) και το κάνουν με κάποιο βαθμό αυτονομίας για την ικανοποίηση του στόχου του χρήστη τους. Οι πράκτορες λογισμικού, όπως οι πραγματικοί πράκτορες, μπορεί να είναι:

- έξυπνοι ή χαζοί
- στατικοί ή κινητοί

Οι κινητοί πράκτορες είναι προγράμματα που μπορούν να μεταφερθούν σε απομακρυσμένα συστήματα, προκειμένου να ασκούν διάφορα καθήκοντα για λογαριασμό των χρηστών τους. Οι κινητοί (φορητοί) πράκτορες έχουν τη δυνατότητα να κινούνται μέσω του δικτύου. Ένας κινητός πράκτορας μπορεί να σταματήσει την εκτέλεσή του, να μετακινηθεί σε άλλο υπολογιστή στο δίκτυο διατηρώντας την κατάσταση του, και να συνεχίσει την εκτέλεση στον υπολογιστή προορισμού.

Μια άλλη περιοχή των κινητών εφαρμογών είναι τα ασύρματα δίκτυα αισθητήρων (WSN) και οι νανο-τεχνολογίες. Οι εξαιρετικά μικροί αισθητήρες, ή νανο-υπολογιστές, μπορούν να «διασκορπιστούν» σε μια συγκεκριμένη περιοχή για να συλλέξουν πληροφορίες. Για παράδειγμα, πολλοί αισθητήρες εγκαθίστανται σε μια περιοχή για την ανίχνευση μετακινήσεων οχημάτων, τη συλλογή διακυμάνσεων της θερμοκρασίας ή τη συγκέντρωση διάφορων άλλων χρήσιμων πληροφοριών. Αλλά αυτοί οι αισθητήρες δεν είναι πολύ χρήσιμοι από μόνοι τους, αν δεν αποτελούν δίκτυα, που ονομάζονται ασύρματα δίκτυα αισθητήρων, τα οποία μεταφέρουν πληροφορίες σε σημεία ελέγχου/διάδοσης. Γενικά, οι εν λόγω συσκευές γρήγορα δημιουργούν δίκτυα και στέλνουν πληροφορίες σε απομακρυσμένες τοποθεσίες. Φυσικά, αυτοί οι υπολογιστές δεν είναι ενσύρματοι. Σχηματίζουν κινητά δίκτυα ad-hoc (mobile ad-hoc networks – MANETs), που χρησιμοποιούνται σε πολλές στρατιωτικές και πολιτικές εφαρμογές.

1.3 Αντικείμενο και στόχος της εργασίας

Το βασικό αντικείμενο της εργασίας είναι η αναπαράσταση γνώσης και ο συμπερασμός σε περιβάλλοντα κινητού υπολογισμού. Σκοπός της εργασίας είναι η προσθήκη κάποιου είδους ευφυΐας σε κινητές εφαρμογές, μέσω της συλλογιστικής (reasoning). Η συλλογιστική είναι η διαδικασία με την οποία μπορούμε να παράγουμε νέα γνώση από την υπάρχουσα, ρητά εκφρασμένη γνώση, με αποτέλεσμα οι εφαρμογές/συσκευές που τη χρησιμοποιούν να γίνονται πιο «έξυπνες». Έτσι, αρχικά ερευνούνται οι υπάρχουσες μορφές αναπαράστασης γνώσης και οι αντίστοιχες μεθοδολογίες για την εξαγωγή συμπερασμάτων. Στη συνέχεια, μελετάμε τη διαδικασία επιλογής των κατάλληλων αναπαραστάσεων και μηχανών συμπερασμού, οι οποίες θα μπορούν να ανταπεξέλθουν στα περιορισμένα σε πόρους κινητά περιβάλλοντα. Προς αυτήν την κατεύθυνση, σχεδιάστηκε και υλοποιήθηκε ένα περιβάλλον δοκιμών για την μελέτη των δυνατοτήτων συμπερασμού σε περιβάλλον κινητής συσκευής και την παρουσίαση των αρνητικών και θετικών στοιχείων των γλωσσών-φορμαλισμών αναπαράστασης γνώσης που έχουν επιλεχθεί.

2. ΑΝΑΠΑΡΑΣΤΑΣΗ ΓΝΩΣΗΣ ΚΑΙ ΣΥΜΠΕΡΑΣΜΟΣ

2.1 Αναπαράσταση γνώσης

Ένα σύστημα αναπαράστασης γνώσης περιγράφει τον τρόπο με τον οποίο ένα πρόγραμμα μπορεί να μοντελοποιήσει ό,τι γνωρίζει για τον κόσμο. Ο στόχος της αναπαράστασης της γνώσης είναι η δημιουργία συστημάτων που θα επιτρέπουν την αποτελεσματική αποθήκευση, τροποποίηση και συλλογιστική πληροφοριών. Η έρευνα στον τομέα αυτό, έχει γεννήσει έναν αριθμό γλωσσών αναπαράστασης γνώσης, η καθεμιά με το δικό της σύνολο χαρακτηριστικών, πλεονεκτημάτων και μειονεκτημάτων. Αυτές οι γλώσσες διαφέρουν ως προς τον τρόπο με τον οποίο αποκτάται η γνώση, την εκφραστικότητα που παρέχουν, καθώς και το είδος των συμπερασμάτων που μπορούν να εξάγουν.

2.1.1 Αρχές αναπαράστασης γνώσης

Η αναπαράσταση γνώσης μπορεί επίσης να περιγραφεί από τους πέντε θεμελιώδεις ρόλους, που έχει στην τεχνητή νοημοσύνη. Είναι οι αρχές της αναπαράστασης γνώσης [23]:

- Μια αναπαράσταση γνώσης αποτελεί μια **αντικατάσταση**: Σύμβολα χρησιμοποιούνται για να αντιπροσωπεύσουν εξωτερικά στοιχεία που δεν μπορούν να αποθηκευτούν σε έναν υπολογιστή, δηλαδή υλικά αντικείμενα, γεγονότα και σχέσεις. Τα σύμβολα είναι υποκατάστατα για τα εξωτερικά στοιχεία. Τα σύμβολα και οι σχέσεις μεταξύ τους, δημιουργούν ένα μοντέλο του εξωτερικού συστήματος, που με την κατάλληλη διαχείριση μπορεί να το προσομοιώσει ή να εκφέρει συμπεράσματα για αυτό.
- Μια αναπαράσταση γνώσης είναι ένα **σύνολο από οντολογικές δεσμεύσεις**: Οντολογία είναι η μελέτη της ύπαρξης. Έτσι, η οντολογία καθορίζει τις κατηγορίες των πραγμάτων που υπάρχουν ή μπορεί να υπάρχουν σε ένα πεδίο εφαρμογής. Οι εν λόγω κατηγορίες θέτουν τις οντολογικές δεσμεύσεις του σχεδιαστή της εφαρμογής ή του μηχανικού της γνώσης.
- Μια αναπαράσταση γνώσης είναι μια **αποσπασματική θεωρία ευφυούς συλλογιστικής**: Για την υποστήριξη συλλογιστικής σχετικά με μοντελοποιημένα στοιχεία σε έναν τομέα, μια αναπαράσταση γνώσης πρέπει να περιγράφει τη συμπεριφορά και τις αλληλεπιδράσεις τους. Η περιγραφή αυτή συνιστά μια θεωρία

για το εκάστοτε πεδίο εφαρμογής. Μπορεί να ορίζεται, για παράδειγμα, μέσω ρητών αξιωμάτων ή να συντάσσεται σε υπολογίσιμα προγράμματα.

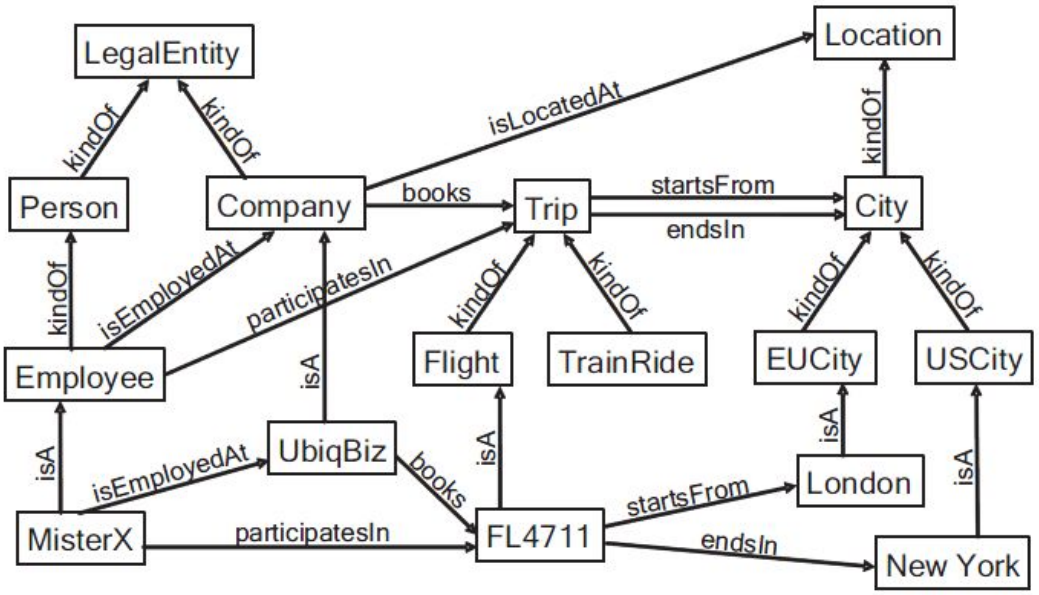
- Μια αναπαράσταση γνώσης είναι ένα **μέσο για αποδοτικό υπολογισμό**: Πέραν της αναπαράστασης γνώσης, ένα σύστημα τεχνητής νοημοσύνης πρέπει να κωδικοποιεί τη γνώση σε μια μορφή που μπορεί να επεξεργαστεί αποδοτικά, από το διαθέσιμο υπολογιστικό εξοπλισμό. Ως εκ τούτου, οι εξελίξεις στο υλικό του υπολογιστή και τη θεωρία προγραμματισμού έχουν μεγάλη επιρροή στην αναπαράσταση γνώσης.
- Μια αναπαράσταση γνώσης είναι **ένα μέσο για την ανθρώπινη έκφραση**: Μια καλή γλώσσα αναπαράστασης γνώσης, θα πρέπει να διευκολύνει την επικοινωνία μεταξύ των μηχανικών της γνώσης, οι οποίοι διαχειρίζονται τα εργαλεία γνώσης, και των ειδικών σε αυτόν τον τομέα, που κατανοούν το πεδίο εφαρμογής. Οι ειδικοί θα πρέπει να είναι σε θέση να διαβάσουν και να επιβεβαιώσουν τους ορισμούς και τους κανόνες του πεδίου, που γράφτηκαν από τους μηχανικούς γνώσης.

2.1.2 Μεθοδολογίες

2.1.2.1 Σημασιολογικά δίκτυα και πλαίσια

Τα σημασιολογικά δίκτυα (semantic networks) απορρέουν από τους "υπαρξιακούς γράφους", που παρουσιάστηκαν από τον Charles Peirce το 1896 για να εκφράσουν λογικές προτάσεις σα γραφικά διαγράμματα κόμβου και δεσμού [24]. Αργότερα, παρουσιάστηκαν παρόμοιοι τρόποι συμβολισμού, όπως οι εννοιολογικοί γράφοι [25], που διέφεραν ελαφρώς στην σύνταξη και τη σημασιολογία. Παρά τις διαφορές αυτές, όλοι οι φορμαλισμοί σημασιολογικών δικτύων επικεντρώνονται στην έκφραση της ταξινομικής δομής κατηγοριών αντικειμένων και των μεταξύ τους σχέσεων.

Ένα σημασιολογικό δίκτυο είναι ένα γράφημα του οποίου οι κόμβοι αντιπροσωπεύουν έννοιες και τα τόξα αντιπροσωπεύουν τις σχέσεις μεταξύ αυτών των εννοιών. Παρέχουν μια διαρθρωτική αναπαράσταση δηλώσεων για ένα πεδίο ενδιαφέροντος. Στο πεδίο των επαγγελματικών ταξιδιών, τυπικές έννοιες θα ήταν οι “Company”, “Employee” ή “Flight”, ενώ τυπικές σχέσεις θα ήταν οι “books”, “isEmployedAt” ή “participatesIn”. Η Εικόνα 2.1 δείχνει ένα παράδειγμα σημασιολογικού δικτύου για τον τομέα «επαγγελματικά ταξίδια».



Εικόνα 2.1: Παράδειγμα σημασιολογικού δικτύου

Τα σημασιολογικά δίκτυα παρέχουν έναν τρόπο απομάκρυνσης από τη φυσική γλώσσα, αναπαριστώντας τη γνώση που καταγράφεται σε ένα κείμενο, σε μια μορφή πιο κατάλληλη για υπολογισμό. Οι γνώσεις που εκφράζονται στο δίκτυο της Εικόνας 2.1, συμπίπτει με το περιεχόμενο του παρακάτω κειμένου φυσικής γλώσσας.

> «Οι υπάλληλοι των εταιρειών είναι άνθρωποι, ενώ και οι άνθρωποι και οι εταιρείες είναι νομικά πρόσωπα. Οι εταιρείες κλείνουν ταξίδια για τους υπαλλήλους τους. Αυτά τα ταξίδια μπορεί να είναι πτήσεις ή διαδρομές με τρένο που αρχίζουν και ολοκληρώνονται σε πόλεις της Ευρώπης ή των ΗΠΑ. Οι εταιρείες έχουν τοποθεσίες που μπορεί να είναι πόλεις. Η εταιρεία UbiqBiz κλείνει την πτήση FL4711 από Λονδίνο για Νέα Υόρκη για τον Mister X.»

Συνήθως, οι έννοιες επιλέγονται έτσι ώστε να εκπροσωπούν το νόημα των ουσιαστικών ενός τέτοιου κειμένου, ενώ οι σχέσεις αντιστοιχίζονται σε ρηματικές φράσεις. Το κομμάτι $\boxed{\text{Company}} \xrightarrow{books} \boxed{\text{Trip}}$ διαβάζεται ως «η εταιρεία κλείνει ταξίδι» και εκφράζεται ως μια δυαδική σχέση μεταξύ δύο εννοιών. Ωστόσο, αυτό δεν είναι υποχρεωτικό. Η σχέση $\xrightarrow{books}$ θα μπορούσε να μετασχηματιστεί στην έννοια $\boxed{\text{Booking}}$ με τις σχέσεις

hasActor, hasParticipant και hasObject να δείχνουν στις Company, Employee και Trip, αντίστοιχα. Με τον τρόπο αυτό, ο τριαδικός χαρακτήρας της θα εκφραζόταν με μεγαλύτερη ακρίβεια από ό,τι στο αρχικό δίκτυο όπου οι πληροφορίες σχετικά με τη συμμετοχή ενός υπαλλήλου σε μια κράτηση υπονοούνται. Θεωρητικά, οι έννοιες και οι σχέσεις σε ένα σημασιολογικό δίκτυο είναι γενικού χαρακτήρα και θα μπορούσαν να αντιπροσωπεύουν οτιδήποτε σχετικό με το πεδίο ενδιαφέροντος. Ωστόσο, έχουν δημιουργηθεί κάποιες ιδιαίτερες σχέσεις για κάποιες πρότυπες περιπτώσεις αναπαράστασης γνώσης και συλλογιστικής.

Το σημασιολογικό δίκτυο στην Εικόνα 2.1 απεικονίζει τη διάκριση μεταξύ γενικών εννοιών, όπως η Employee και ατομικών εννοιών, όπως η MisterX. Η τελευταία αντιπροσωπεύει συγκεκριμένα άτομα ή αντικείμενα στο πεδίο ενδιαφέροντος, ενώ η προηγούμενη αντιπροσωπεύει κλάσεις που ομαδοποιούν αυτά τα άτομα που έχουν ορισμένες κοινές ιδιότητες, όπως π.χ. όλοι οι υπάλληλοι. Η ιδιαίτερη σχέση που συνδέει τα άτομα με τις κλάσεις τους, είναι ο προσδιορισμός (instantiation) και συμβολίζεται με isA. Έτσι, ο MisterX ονομάζεται στιγμιότυπο (instance) της έννοιας υπάλληλος. Το κάτω μέρος του δικτύου αφορά στη γνώση για τα αντικείμενα, αντικατοπτρίζοντας μια συγκεκριμένη κατάσταση του υπαλλήλου MisterX, που συμμετέχει σε μια συγκεκριμένη πτήση, ενώ το πάνω μέρος αφορά σε γνώσεις σχετικά με γενικές έννοιες, δηλαδή στις διάφορες πιθανές καταστάσεις.

Το πιο χαρακτηριστικό είδος σχέσης σε σημασιολογικά δίκτυα, ωστόσο, είναι αυτό της υπαγωγής (subsumption), η οποία εδώ συμβολίζεται ως kindOf. Ένας δεσμός υπαγωγής συνδέει δύο γενικές έννοιες και εκφράζει εξειδίκευση ή γενίκευση, αντίστοιχα. Στην Εικόνα 2.1, μια πτήση είναι ένα είδος ταξιδιού, δηλαδή η Trip συμπεριλαμβάνει την Flight. Αυτό σημαίνει ότι κάθε πτήση είναι και ταξίδι, αλλά ενδέχεται να υπάρχουν άλλα ταξίδια που δεν είναι πτήσεις, όπως η διαδρομή με τρένο. Η υπαγωγή συνδέεται με την έννοια της κληρονομικότητας, μιας και μια εξειδικευμένη έννοια κληρονομεί όλες τις ιδιότητες των πιο γενικών μητρικών εννοιών. Για παράδειγμα, από αυτό το δίκτυο μπορεί κανείς να καταλάβει ότι μια εταιρεία μπορεί να βρίσκεται σε μια ευρωπαϊκή πόλη, δεδομένου ότι η ακμή locatedAt δείχνει από την

Company στην Location, ενώ η EUCity είναι ένα είδος της City η οποία είναι ένα είδος της Location. Η έννοια EUCity κληρονομεί την ιδιότητα να αποτελεί μια πιθανή τοποθεσία για μια εταιρεία, από την έννοια Location.

Τα σημασιολογικά δίκτυα συνδέονται στενά με μια άλλη μορφή αναπαράστασης γνώσης, που ονομάζεται συστήματα πλαισίων (frame systems). Στην πραγματικότητα, τα συστήματα πλαισίων και τα σημασιολογικά δίκτυα μπορεί να είναι ίδια στην εκφραστικότητά τους, αλλά χρησιμοποιούν διαφορετικούς τρόπους αναπαράστασης [24]. Ενώ η αναπαράσταση στο σημασιολογικό δίκτυο είναι ένα γράφημα με κόμβους-έννοιες να συνδέονται με τόξα-σχέσεις, τα πλαίσια αναπαριστούν τις έννοιες ως κουτιάκαι τις σχέσεις ως θυρίδες (slots) εντός των πλαισίων, που μπορούν να γεμίσουν από άλλα πλαίσια. Έτσι, στο σύστημα πλαισίων το γράφημα μετατρέπεται σε εμφωλευμένα κουτιά.

2.1.2.2 Κανόνες

Μια άλλη φυσική μορφή έκφρασης της γνώσης σε κάποιο πεδίο ενδιαφέροντος, είναι οι κανόνες που αντανακλούν την έννοια της συνέπειας. Οι κανόνες έχουν τη μορφή δομών IF-THEN και επιτρέπουν την έκφραση διαφόρων ειδών πολύπλοκων καταστάσεων. Οι κανόνες βρίσκονται στα συστήματα λογικού προγραμματισμού, όπως η γλώσσα Prolog [26], στις επαγωγικές βάσεις δεδομένων (deductive databases) [27] ή σε επαγγελματικά συστήματα κανόνων. Το ακόλουθο είναι ένα παράδειγμα κανόνων που εκφράζουν γνώση πάνω στο πεδίο επαγγελματικών ταξιδιών, βασισμένων στη διαισθητική ερμηνεία τους.

(1) **IF** something is a flight

THEN it is also a trip

(2) **IF** some person participates in a trip booked by some company

THEN this person is an employee of this company

(3) **FACT** the person MisterX participates in a flight booked by the company UbiqBiz

(4) **IF** a trip's source and destination cities are close to each other

THEN the trip is by train

Το τμήμα IF, ονομάζεται σώμα (body) του κανόνα, ενώ το τμήμα THEN ονομάζεται κεφαλή (head) του κανόνα. Τα συστήματα αναπαράστασης γνώσης που βασίζονται σε κανόνες, δρουν πάνω σε γεγονότα (facts), τα οποία συχνά δηλώνονται ως ένα ιδιαίτερο είδος κανόνα με κενό σώμα. Ξεκινούν από ένα δεδομένο σύνολο γεγονότων, όπως το γεγονός (3), και στη συνέχεια εφαρμόζουν τους κανόνες, προκειμένου να προσδιοριστούν νέα γεγονότα και, έτσι, να «εξάγουν συμπεράσματα». Ωστόσο, η διαισθητική ερμηνεία με φράσεις της φυσικής γλώσσας, δεν είναι κατάλληλη για υπολογισμό και, κατά συνέπεια, οι εν λόγω φράσεις μετατρέπονται σε κατηγορήματα (predicates) και μεταβλητές (variables) πάνω σε αντικείμενα του πεδίου ενδιαφέροντος. Μια μετατροπή των ανωτέρω κανόνων στο τυπικό στυλ των γλωσσών κανόνων έχει ως εξής:

(1) Trip(?*t*) :− Flight(?*t*)

(2) Employee(?*p*) Λ isEmployedAt(?*p*, ?*c*) :−

Trip(?*t*) Λ books(?*c*, ?*t*) Λ Company(?*c*) Λ participatesIn(?*p*, ?*t*) Λ Person(?*p*)

(3) Person(MisterX)Λ participatesIn(MisterX, FL4711) Λ

Flight(FL4711) Λ books(UbiqBiz, FL4711) Λ Company(UbiqBiz) :−

(3) TrainRide(?*t*):−Trip(?*t*) Λ startsFrom(?*t*, ?*s*) Λ endsIn(?*t*, ?*d*) Λ close(?*s*,?*d*)

Στα περισσότερα συστήματα λογικού προγραμματισμού ένας κανόνας διαβάζεται ως μια αντίστροφη συνεπαγωγή, η οποία υποδεικνύεται από το σύμβολο :- που μοιάζει με ένα ανάποδο βέλος. Σε αυτή τη μορφή, οι διαισθητικές έννοιες από το κείμενο, που ήταν έννοιες και σχέσεις που αφορούν την περίπτωση του σημασιολογικού δικτύου, έγιναν κατηγορήματα που συνδέονται μέσω μεταβλητών και σταθερών που προσδιορίζουν αντικείμενα του πεδίου ενδιαφέροντος. Οι μεταβλητές ξεκινούν με το σύμβολο ? και έχουν ως τιμές τους, σταθερές που εμφανίζονται σε γεγονότα όπως το (3). Ο κανόνας (1) αντικατοπτρίζει την κληρονομικότητα – ή υπαγωγή - μεταξύ ταξιδιών και πτήσεων δηλώνοντας ότι «κάθε τι που είναι πτήση είναι και ταξίδι». Ο κανόνας (2) εξάγει συμπεράσματα σχετικά με την επαγγελματική κατάσταση των συμμετεχόντων των πτήσεων των επιχειρήσεων. Από τα γεγονότα στον κανόνα (3), οι δύο παραπάνω κανόνες μπορούν να εξάγουν το έμμεσο γεγονός ότι "ο MisterX είναι υπάλληλος της

UbiqBiz". Ενώ οι κανόνες (1) και (2) εκφράζουν γενική γνώση στον τομέα, ο κανόνας (4) μπορεί να ερμηνευθεί ως μέρος της πολιτικής ταξιδιών κάποιας εταιρείας, δηλώνοντας ότι τα ταξίδια μεταξύ κοντινών πόλεων θα διεξάγονται με τρένο.

Τα συστήματα αναπαράστασης γνώσης που βασίζονται σε κανόνες, είναι ιδιαίτερα κατάλληλα για συλλογιστική σχετικά με συγκεκριμένα δεδομένα στιγμιότυπων, δηλαδή απλά γεγονότα της μορφής Employee(MisterX). Πολύπλοκα σύνολα κανόνων μπορούν να αντλήσουν αποτελεσματικά τέτοια έμμεσα γεγονότα από ρητά δοσμένα γεγονότα. Γίνονται, όμως, προβληματικά αν εξαχθούν περισσότερο σύνθετες και γενικές προτάσεις σχετικά με το πεδίο, οι οποίες δεν ταιριάζουν με την κεφαλή ενός κανόνα.

2.1.2.3 Λογική και λογική πρώτης τάξης

Τα σημασιολογικά δίκτυα, καθώς και οι κανόνες, έχουν μορφοποιηθεί χρησιμοποιώντας τη λογική για να τους δώσει μια ακριβή σημασιολογία (semantics). Χωρίς μια τέτοια ακριβή μορφοποίηση είναι αόριστες και διφορούμενες και, συνεπώς, προβληματικές για υπολογιστικούς σκοπούς. Για παράδειγμα, από τη γραφική αναπαράσταση του σημασιολογικού δικτύου στην Εικόνα 2.1, δεν είναι σαφές κατά πόσον οι επιχειρήσεις μπορούν κλείνουν πτήσεις μόνο για τους δικούς τους υπαλλήλους ή και για το προσωπικό των συνεταιρικών τους εταιριών. Αλλά ούτε και από το κομμάτι Company —books→ Trip είναι σαφές αν κάθε εταιρία κλείνει ταξίδια ή μόνο μερικές. Επίσης, για τους κανόνες, παρά την πιο επίσημη εμφάνισή τους, το ακριβές νόημα παραμένει ασαφές όταν, για παράδειγμα, εισάγονται μορφές άρνησης (negation), που επιτρέπουν πιθανές συγκρούσεις μεταξύ των κανόνων. Ανάλογα με την επιλογή της διαδικαστικής αξιολόγησης ή τη χρήση της σημασιολογίας, παράγονται διαφορετικά αποτελέσματα.

Ο πιο σημαντικός και ουσιώδης λογικός φορμαλισμός που χρησιμοποιείται για την αναπαράσταση γνώσης, είναι ο «κατηγορηματικός λογισμός πρώτης τάξης» (first-order predicate calculus), ή λογική πρώτης τάξης (first-order logic - FOL) εν συντομία, και εδώ επιλέγουμε αυτόν τον φορμαλισμό για να παρουσιάσουμε τη λογική ως μια μορφή αναπαράστασης γνώσης. Η λογική πρώτης τάξης επιτρέπει την περιγραφή του πεδίου ενδιαφέροντος σα να αποτελείται από αντικείμενα, δηλαδή πράγματα που έχουν

ατομική ταυτότητα, και την κατασκευή λογικών τύπων γύρω από αυτά τα αντικείμενα, που σχηματίζονται από κατηγορήματα, συναρτήσεις, μεταβλητές και λογικούς συνδέσμους [24].

Μια γλώσσα FOL αποτελείται από λογικά και μη λογικά σύμβολα. Τα λογικά σύμβολα αντιπροσωπεύουν την ποσοτικοποίηση (quantification), τη συνεπαγωγή (implication), τη σύζευξη (conjunction) και τη διάζευξη (disjunction), ενώ τα μη-λογικά σύμβολα είναι σταθερές, κατηγορήματα (predicates), συναρτησιακά σύμβολα (function symbols) και μεταβλητές. Τα σύμβολα των σταθερών, μεταβλητών και συναρτήσεων χρησιμοποιούνται για την κατασκευή των όρων, που μπορούν να συνδυαστούν με κατηγορήματα για την κατασκευή εκφράσεων. Ένα υποσύνολο των πιθανών εκφράσεων που υπακούει σε ορισμένους κανόνες συντακτικής κατασκευής, ονομάζονται καλοσχηματισμένες εκφράσεις (well-formed formulas - wff).

Η σημασιολογία της FOL δίνεται από τη θεωρία μοντέλων του Tarski, η οποία αναφέρεται ως μοντελο-θεωρητική ή δηλωτική σημασιολογία. Σε αυτήν, μια ερμηνεία (interpretation) χρησιμοποιείται για να συσχετίσει τα σύμβολα της γλώσσας με τον κόσμο. Μια ερμηνεία αποτελείται από ένα σύνολο D από πρωτογενείς οντότητες (individuals) που ονομάζεται πεδίο αναφοράς (domain of discourse), μια συνάρτηση που αντιστοιχίζει σύμβολα σταθερών με το D, μια συνάρτηση που αντιστοιχίζει σύμβολα συναρτήσεων με τις συναρτήσεις στο D και μια συνάρτηση που αντιστοιχίζει σύμβολα κατηγορημάτων με τις σχέσεις στο D. Εάν μια έκφραση είναι αληθής (true) υπό κάποια ερμηνεία, τότε η ερμηνεία αυτή είναι ένα μοντέλο της πρότασης. Ομοίως, εάν ένα σύνολο εκφράσεων είναι αληθές υπό κάποια ερμηνεία, τότε η ερμηνεία είναι ένα μοντέλο αυτών των εκφράσεων. Δεδομένου ενός συνόλου εκφράσεων Γ, αν κάποια έκφραση ϕ είναι κατ' ανάγκην αληθής, τότε λέμε ότι το Γ συνεπάγεται τη ϕ, το οποίο γράφεται Γ |= ϕ. Τυπικά, υπάρχει μια ειδική ερμηνεία, που ονομάζεται επιδιωκόμενη ερμηνεία (intended interpretation), η οποία αντικατοπτρίζει με ακρίβεια το επιθυμητό νόημα για ένα σύνολο προτάσεων. Μια θεωρία Τ, είναι ένα σύνολο προτάσεων που είναι κλειστές υπό λογική συνέπεια. Έτσι, για κάθε ϕ τέτοια ώστε Τ |= ϕ, $\phi \in$ Τ.

Μια συμπερασματική διαδικασία είναι ένας αλγόριθμος που μπορεί να υπολογίσει τις προτάσεις που προκύπτουν λογικά από μια βάση γνώσης. Η FOL έχει μια ορθή (sound) και πλήρης (complete) συμπερασματική διαδικασία, που ονομάζεται διάψευση

ανάλυσης (resolution refutation). Μια ορθή διαδικασία παράγει μόνο προτάσεις που συνεπάγονται από ένα σύνολο Γ, ενώ μια πλήρης διαδικασία μπορεί να βρει μια απόδειξη για οποιαδήποτε συνεπαγόμενη πρόταση. Ωστόσο, η διάψευση είναι δυσεπίλυτη, γεγονός που την καθιστά μια κακή επιλογή για τον χειρισμό μεγάλων βάσεων γνώσεων. Η FOL είναι μια εξαιρετικά εκφραστική αναπαράσταση, και μπορεί να χρησιμοποιηθεί για την περιγραφή σημασιολογικών δικτύων και συστημάτων πλαισίου.

Παρόμοια με τα σημασιολογικά δίκτυα, οι περισσότερες προτάσεις σε φυσική γλώσσα μπορούν να εκφραστούν με λογικές προτάσεις για τα αντικείμενα του πεδίου ενδιαφέροντος, με την κατάλληλη επιλογή συμβόλων για τα κατηγορήματα και τις συναρτήσεις. Οι έννοιες αντιστοιχίζονται σε μοναδιαία κατηγορήματα, ενώ οι σχέσεις σε δυαδικά κατηγορήματα. Επιδεικνύουμε τη χρήση της λογικής για την αναπαράσταση γνώσης μετατρέποντας σε αξιώματα τμήματα του σημασιολογικού δικτύου της Εικόνας 2.1, με μεγαλύτερη ακρίβεια. Η υπαγωγή, για παράδειγμα, μπορεί να εκφραστεί άμεσα μέσω μιας λογικής συνεπαγωγής, πράγμα που αντανακλάται στη μετάφραση του παρακάτω κομματιού.

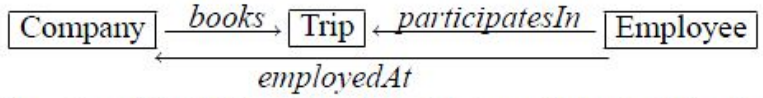
Employee $\xrightarrow{kindOf}$ Person $\qquad \forall x : (Employee(x) \rightarrow Person(x))$.

Λόγω του καθολικού ποσοδείκτη (universal quantifier), η μεταβλητή x του λογικού τύπου, εκτείνεται σε όλα τα αντικείμενα του πεδίου και διαβάζεται ως «κάθε τι που είναι υπάλληλος είναι επίσης άνθρωπος». Σε άλλα τμήματα του δικτύου μπορούν να εφαρμοστούν περαιτέρω περιορισμοί χρησιμοποιώντας λογικούς τύπους, όπως φαίνεται στο ακόλουθο παράδειγμα.

Company $\xrightarrow{books}$ Trip $\qquad \forall x, y : (books(x, y) \rightarrow Company(x) \wedge Trip(y))$
$\forall x : \exists y : (Trip(x) \rightarrow Company(y) \wedge books(y, x))$

Η γραφική αναπαράσταση του τμήματος του δικτύου αφήνει ορισμένες λεπτομέρειες ανοιχτές, ενώ οι λογικοί τύποι συλλαμβάνουν τη σχέση των κρατήσεων μεταξύ των εταιρειών και των ταξιδιών με μεγαλύτερη ακρίβεια. Ο πρώτος τύπος δηλώνει ότι το πεδίο ορισμού (domain) και το πεδίο τιμών (range) της σχέσης της κράτησης είναι οι

εταιρείες και τα ταξίδια, αντίστοιχα, ενώ ο δεύτερος τύπος εξασφαλίζει ότι για κάθε ταξίδι υπάρχει πράγματι μια εταιρεία που το έχει κλείσει. Ειδικότερα, πιο σύνθετοι περιορισμοί που εκτείνονται σε μεγαλύτερα τμήματα ενός γραφήματος δικτύου, μπορούν να κατασκευαστούν με τη λογική, όπου η διαισθητική γραφική σημειολογία στερείται εκφραστικότητας. Ως παράδειγμα εξετάζονται οι σχέσεις μεταξύ ταξιδιών, εταιρειών και εργαζομένων στο παρακάτω τμήμα.



$$\forall x : \exists y : (Trip(x) \rightarrow Employee(y) \wedge participatesIn(y, x) \wedge books(employer(y), x))$$

Ο λογικός τύπος εκφράζει επιπλέον γνώσεις, που δεν αποτυπώνονται στη γραφική απεικόνιση. Δηλώνει ότι, για κάθε ταξίδι, πρέπει να υπάρχει υπάλληλος που συμμετέχει σε αυτό το ταξίδι, ενώ ο εργοδότης του είναι η εταιρεία που έχει κάνει την κράτηση της πτήσης. Οι κανόνες μπορούν επίσης να μορφοποιηθούν με τη λογική. Ένας IF-THEN κανόνας μπορεί να αναπαρασταθεί ως μια λογική συνεπαγωγή με καθολικά ποσοτικοποιημένες μεταβλητές. Για παράδειγμα, μια συνήθης μορφοποίηση του κανόνα

IF a trip's source and destination cities are close to each other

THEN the trip is by train

είναι η μετατροπή του στον λογικό τύπο

$$\forall x, y, z : (Trip(x) \wedge startsFrom(x, y) \wedge endsIn(x, z) \wedge close(y, z) \rightarrow TrainRide(x)).$$

Ωστόσο, τα τυπικά συστήματα κανόνων δεν ερμηνεύουν έναν τέτοιο τύπο με την κλασική έννοια της λογικής πρώτης τάξης, αλλά χρησιμοποιούν διαφορετικά είδη σημασιολογίας.

Στο πλαίσιο του Σημασιολογικού Ιστού [28] (Semantic Web), δύο συγκεκριμένοι λογικοί φορμαλισμοί έχουν ξεχωρίσει, αντικατοπτρίζοντας τις μορφές αναπαράστασης γνώσης του σημασιολογικού δικτύου και των κανόνων. Οι συμβολισμοί των γραφημάτων των σημασιολογικών δικτύων έχουν μορφοποιηθεί μέσω των περιγραφικών λογικών

(description logics)[29], οι οποίες είναι τμήματα της λογικής πρώτης τάξης με την τυπική μοντελο-θεωρητική σημασιολογία του Tarski, αλλά περιορισμένη σε μοναδιαία και δυαδικά κατηγορήματα για την αποτύπωση των εννοιών και των σχέσεων. Από την άλλη, οι κανόνες έχουν μορφοποιηθεί μέσω φορμαλισμών λογικού προγραμματισμού (logic programming) με ελάχιστη σημασιολογία μοντέλου, με επίκεντρο την παραγωγή απλών γεγονότων που αφορούν συγκεκριμένα αντικείμενα.

2.1.2.4 Περιγραφικές λογικές

Οι περιγραφικές λογικές (description logics - DL) είναι πάρα πολύ εκφραστικές, ώστε έγιναν ένα σημαντικό πρότυπο αναπαράστασης γνώσης. Στη συνέχεια, περιγράφεται μια από τις πιο σημαντικές και ισχυρές περιγραφικές λογικές, που ονομάζεται ALC. Άλλες περιγραφικές λογικές γίνονται καλύτερα κατανοητές ως περιορισμοί ή επεκτάσεις της ALC. Παρακάτω, παρουσιάζεται η πρότυπη σημειολογία περιγραφικής λογικής και δίνεται μια επίσημη απεικόνιση σε τυπική σύνταξη λογικής πρώτης τάξης.

Η περιγραφική λογική ALC

Η θεωρία μιας περιγραφικής λογικής αποτελείται από δηλώσεις σχετικά με έννοιες (concepts), άτομα (individuals) και τις σχέσεις/συσχετίσεις (relations) τους. Τα άτομα αντιστοιχούν σε σταθερές στη λογική πρώτης τάξης, και οι έννοιες αντιστοιχούν σε μοναδιαία κατηγορήματα. Από πλευράς των σημασιολογικών δικτύων, οι έννοιες των περιγραφικών λογικών αντιστοιχούν σε γενικές έννοιες των σημασιολογικών δικτύων, ενώ τα άτομα αντιστοιχούν σε ατομικές έννοιες.

Οι έννοιες μπορεί να έχουν κάποια ονομασία ή να είναι ανώνυμες (σύνθετες) έννοιες. Οι ονομασμένες έννοιες αποτελούνται απλώς από ένα όνομα, π.χ. «άνθρωπος», το οποίο αντιστοιχεί σε ένα μοναδιαίο κατηγόρημα στη λογική πρώτης τάξης. Οι σύνθετες έννοιες διαμορφώνονται από ονομασμένες έννοιες με τη χρήση των μεθόδων κατασκευής εννοιών (concept constructors), παρόμοια με τη δημιουργία σύνθετων τύπων από τους ατομικούς τύπους στη λογική πρώτης τάξης. Στην ALC, υπάρχουν οι εξής boolean μέθοδοι κατασκευής:

- σύζευξη (conjunction) ⊓, η οποία είναι δυαδική
- διάζευξη (disjunction) ⊔, η οποία είναι δυαδική
- άρνηση (negation) ¬ , η οποία είναι μοναδιαία.

Έτσι, αν οι C και D είναι έννοιες, τότε $C \sqcap D$ και $\neg C$ είναι, επίσης, έννοιες. Οι μέθοδοι κατασκευής εννοιών μπορούν να εμφωλεύονται. Η μετατροπή των boolean μεθόδων σε κατηγορηματική λογική πρώτης τάξης είναι προφανής. Για παράδειγμα, η δήλωση $C \sqcap \neg D$ μεταφράζεται στον τύπο $C(x) \wedge \neg D(x)$.

Οι δηλώσεις στην ALC συσχετίζουν ονομασμένες ή ανώνυμες έννοιες, μέσω των ακόλουθων αξιωμάτων:

- υπαγωγή (subsumption, inclusion) $\sqsubseteq$
- αντίστροφη υπαγωγή (reverse inclusion) $\sqsupseteq$
- ισοδυναμία (equivalence) $\equiv$

Η σημασία τους στη λογική πρώτης τάξης είναι συνεπαγωγή $\rightarrow$, αντίστροφη συνεπαγωγή $\leftarrow$ και ισοδυναμία $\leftrightarrow$. Οι ελεύθερες μεταβλητές που προκύπτουν είναι καθολικά ποσοτικοποιημένες. Για παράδειγμα, η δήλωση της $C \sqsubseteq D \sqcup \neg E$ μεταφράζεται ως $\forall x : (C(x) \rightarrow (D(x) \vee \neg E(x)))$.

Η ALC παρέχει δύο ειδικές κλάσεις ως συντομεύσεις, τις $\bot$ και $\top$. Ορίζονται από τις ισοδυναμίες $\bot \equiv C \sqcap \neg C$ και $\top \equiv C \sqcup \neg C$, όπου C είναι κάποια αυθαίρετη έννοια. Δηλαδή, η $\bot$ είναι η κενή έννοια και η $\top$ είναι η έννοια η οποία περιέχει τα πάντα.

Η ALC επιτρέπει την περιορισμένη περαιτέρω χρήση ποσοδεικτών μέσω των αποκαλούμενων περιορισμών ρόλων (role restrictions). Ένας ρόλος είναι μια ονομασμένη οντότητα, η οποία μεταφράζεται σε ένα δυαδικό κατηγόρημα στη λογική πρώτης τάξης. Στο μοντέλο του σημασιολογικού δικτύου, οι ρόλοι είναι οι σχέσεις μεταξύ των εννοιών. Με δεδομένο έναν τέτοιο ρόλο r και μια (ονομασμένη ή ανώνυμη) έννοια C, μπορούν να δημιουργηθούν οι σύνθετες έννοιες $\forall r.C$ και $\exists r.C$. Οι περιορισμοί ρόλων και οι boolean κατασκευαστές μπορούν να εμφωλεύονται αυθαίρετα ο ένας με τον άλλο, για να σχηματίσουν ανώνυμες έννοιες. Η σύνθετη έννοια $\forall r.C$ μεταφράζεται σε $\forall y : (r(x, y) \rightarrow C(y))$ στη λογική πρώτης τάξης, ενώ η $\exists r.C$ μεταφράζεται σε $\exists y : (r(x, y) \wedge C(y))$.

Τέλος, ένα TBox στην ALC αποτελείται από ένα σύνολο δηλώσεων της μορφής C ⊑ D, C ⊒ D ή C ≡ D, όπου C και D είναι ονομασμένες ή σύνθετες έννοιες. Προφανώς, κάθε TBox μπορεί να μετατραπεί σε λογική πρώτης τάξης και, έτσι, κληρονομεί μια σχέση λογικής συνέπειας από αυτήν. Ακολουθούν μερικά παραδείγματα δηλώσεων του TBox στο πεδίο «επαγγελματικά ταξίδια». Η δήλωση

Employee ⊑ Person

κωδικοποιεί τη γνώση ότι κάθε εργαζόμενος είναι άνθρωπος, ενώ η

Trip ⊑∃bookedBy.(Company⊔Person)

δηλώνει ότι κάθε ταξίδι έχει κλειστεί από κάποια εταιρεία ή από κάποιον άνθρωπο. Τα άτομα αντιστοιχούν σε σταθερές στη λογική πρώτης τάξης. Η ALC επιτρέπει να δηλώσουμε ότι ορισμένα άτομα ανήκουν σε (ονομασμένες ή σύνθετες) έννοιες, όπως π.χ. η C(a) δηλώνει ότι το άτομο a ανήκει σε μία έννοια C. Ομοίως, η δήλωση r(a,b), όπου r είναι ένας ρόλος, σημαίνει ότι τα άτομα a και b έχουν τη σχέση r.

Ένα ABox στην ALC αποτελείται από ένα σύνολο δηλώσεων της μορφής C(a) ή r(a,b), όπου C είναι μια ονομασμένη ή ανώνυμη έννοια, r είναι ένας ρόλος, και τα a, b είναι άτομα. Μια βάση γνώσεων ALC αποτελείται από ένα ALC ABox και ένα ALC TBox. Παραδείγματα δηλώσεων ABox είναι τα bookedBy(FL4711, UbiqBiz) και Flight(FL4711) με προφανείς σημασίες.

2.1.2.5 Λογικός προγραμματισμός

Αρχικά, ο λογικός προγραμματισμός (logic programming) είχε σχεδιαστεί ως ένας τρόπος χρήσης της λογικής πρώτης τάξης, ως μία γλώσσα προγραμματισμού. Προκειμένου να καταστεί δυνατός ο αποτελεσματικός υπολογισμός, οι τύποι ήταν συντακτικά περιορισμένοι στις λεγόμενες προτάσεις Horn (Horn clauses). Επιπλέον, μόνο ορισμένα είδη λογικής συνέπειας λαμβάνονταν υπ' όψιν. Συντακτικά, οι προτάσεις Horn μπορούν να αντιμετωπιστούν σαν κανόνες. Για παράδειγμα, η έκφραση Trip(t) ∨ ¬Flight(t) είναι μια πρόταση Horn, η οποία είναι σημασιολογικά ισοδύναμη με την πρόταση ∀t : Trip(t) ← Flight(t) της FOL. Αυτή, με τη σειρά της, μπορεί επίσης να ερμηνευθεί ως ο κανόνας Trip(?t) :- Flight(?t).

Ωστόσο, η σημασιολογία της πρότασης Horn δίδεται μέσω της σημασιολογίας της λογικής πρώτης τάξης, ενώ οι κανόνες του λογικού προγραμματισμού έχουν συνήθως διαφορετική έννοια. Μία από τις διαφορές πηγάζει από το γεγονός ότι σε ένα σύστημα λογικού προγραμματισμού χρησιμοποιούνται μόνο ορισμένα είδη λογικής συνέπειας, δηλαδή τα βασικά στιγμιότυπα (ground instances) των κατηγορημάτων. Στο παράδειγμα, η προσθήκη ενός γεγονότος Flight(FL4711) θα επέτρεπε την εξαγωγή του γεγονότος Trip(FL4711), τόσο στην FOL όσο και σε ένα σύστημα λογικού προγραμματισμού. Παρόλα αυτά, ένα συμπέρασμα όπως το Trip(FL4711) ∨ ¬Flight(FL4711) θα ήταν δυνατό μόνο σε FOL και δεν θα μπορούσε να παραχθεί μέσω της σημασιολογίας του λογικού προγραμματισμού.

Η δεύτερη διαφορά μεταξύ των σημασιολογιών αφορά στο χειρισμό των αρνητικών πληροφοριών. Στο παραπάνω παράδειγμα θα μπορούσαμε να ενδιαφερόμαστε για το αν η δήλωση Trip(FL2306) ισχύει. Στη FOL, δε γίνεται να υπολογιστεί αν αυτή η δήλωση είναι αληθής ή ψευδής. Στο λογικό προγραμματισμό, ωστόσο, η δήλωση θα μπορούσε να θεωρηθεί εσφαλμένη. Ο χειρισμός των αρνητικών πληροφοριών για τον λογικό προγραμματισμό από αυτή την άποψη στηρίζεται στην υπόθεση κλειστού κόσμου (closed world assumption – CWA): μιας και δεν υπάρχουν διαθέσιμες πληροφορίες σχετικά με την FL2306, θεωρείται ότι δεν είναι ένα ταξίδι.

Συνεπώς, η σημασιολογία του λογικού προγραμματισμού είναι μη μονοτονική: άμα προσθέσουμε το γεγονός Flight (FL2306) στη βάση γνώσης, τότε το γεγονός Trip(FL2306) γίνεται αληθές. Αυτή η επίγνωση, προκάλεσε σημαντικές ερευνητικές προσπάθειες σχετικά με τη συσχέτιση του λογικού προγραμματισμού και της μη μονοτονικής συλλογιστικής, οι οποίες οδήγησαν στην εισαγωγή μη μονοτονικών ειδών άρνησης στον λογικό προγραμματισμό.

2.1.2.6 Οντολογίες

Προς επίτευξη της ενοποίησης πληροφοριών από διαφορετικές πηγές, πρέπει να υπάρχει μια κοινή κατανόηση του σχετικού πεδίου. Οι φορμαλισμοί αναπαράστασης γνώσης παρέχουν δομές για την οργάνωση αυτής της γνώσης, αλλά δεν παρέχουν μηχανισμούς για το διαμοιρασμό της. Οι οντολογίες (ontologies) παρέχουν ένα κοινό λεξιλόγιο για να υποστηρίξουν τη διαμοίραση και επαναχρησιμοποίηση της γνώσης.

Όπως συζητήθηκε από τους Guarino και Giaretta [30], η έννοια του όρου οντολογία είναι συχνά ασαφής. Χρησιμοποιήθηκε αρχικά για να περιγράψει τη φιλοσοφική μελέτη της φύσης και της οργάνωσης της πραγματικότητας. Στην τεχνητή νοημοσύνη, ο πιο παρατεθειμένος ορισμός οφείλεται στον Tom Gruber [31]:

«***Μια οντολογία είναι μια τυπική (formal), κατηγορηματική (explicit) προδιαγραφή μιας διαμοιρασμένης (shared), εννοιολογικής αναπαράστασης (conceptualization)***.»

- Ο όρος *«εννοιολογική αναπαράσταση»* (conceptualization) αναφέρεται σε ένα αφηρημένο μοντέλο φαινομένων του κόσμου, στο οποίο έχουν προσδιοριστεί οι έννοιες που σχετίζονται με τα φαινόμενα αυτά.
- Ο όρος *«κατηγορηματική»* (explicit) σημαίνει ότι το είδος των εννοιών που χρησιμοποιούνται, και οι περιορισμοί που αφορούν τη χρήση αυτών των εννοιών είναι προσδιορισμένα με σαφήνεια.
- Ο όρος «*αυστηρή*» (formal) σημαίνει ότι η οντολογία πρέπει να είναι μηχανικά αναγνώσιμη.
- Ο όρος «*διαμοιρασμένη*» (shared) σημαίνει ότι η οντολογία πρέπει να αποτυπώνει γνώση κοινής αποδοχής στα πλαίσια μιας κοινότητας.

Ειδικότερα, πρόκειται για μια πλειάδα $\langle D, R \rangle$, όπου D είναι το πεδίο αναφοράς και R είναι ένα σύνολο σχέσεων στο D. Μια οντολογία συνδέει όρους λεξιλογίου με οντότητες που προσδιορίζονται στην εννοιολογική αναπαράσταση και παρέχει τους ορισμούς για να περιοριστούν οι ερμηνείες των όρων αυτών.

Οι Guarino και Giaretta υποστηρίζουν ότι ο ορισμός της εννοιολογικής αναπαράστασης των Genesereth και Nilsson, δεν πρέπει να χρησιμοποιείται για τον προσδιορισμό της οντολογίας, επειδή υπονοεί ότι μια εννοιολογική αναπαράσταση αποτελεί μια και μοναδική κατάσταση των πραγμάτων/σχέσεων (δηλαδή, πρόκειται για μια εκτατική (extensional) δομή).

Ωστόσο, μια οντολογία πρέπει να προβλέπει όρους για την εκπροσώπηση όλων των πιθανών καταστάσεων σε σχέση με ένα συγκεκριμένο πεδίο. Ως εκ τούτου, προτείνουν ότι η εννοιολογική αναπαράσταση θα πρέπει να αποτελεί μια εντατική (intensional) δομή $\langle W, D, R \rangle$, όπου W είναι το σύνολο των δυνατών κόσμων, D είναι το πεδίο

αναφοράς και R είναι ένα σύνολο εντατικών σχέσεων, όπου μια n-αδική εντατική σχέση είναι μια συνάρτηση από το W έως 2^{D^n} (το σύνολο όλων των πιθανών n-αδικών σχέσεων στο D). Σε ένα μεταγενέστερο άρθρο, ο Guarino τελειοποιεί το μοντέλο αυτό και δίνει τον παρακάτω ορισμό για μια οντολογία [32].

Μια οντολογία είναι μια λογική θεωρία, η οποία αντιπροσωπεύει το επιδιωκόμενο νόημα ενός επίσημου λεξιλογίου, δηλαδή την οντολογική δέσμευσή του σε μια συγκεκριμένη εννοιολογική αναπαράσταση του κόσμου. Τα μοντέλα μιας λογικής γλώσσας που χρησιμοποιεί ένα τέτοιο λεξιλόγιο, περιορίζονται από την οντολογική δέσμευσή του. Μια οντολογία αντικατοπτρίζει έμμεσα τη δέσμευση αυτή (καθώς και την υποκείμενη εννοιολογική αναπαράσταση), προσεγγίζοντας αυτά τα επιδιωκόμενα μοντέλα.

Οι οντολογίες έχουν ένα κοινό σύνολο δομικών συστατικών για την αποτελεσματική αναπαράσταση της γνώσης και την εξαγωγή συμπερασμάτων, ανεξαρτήτως της γλώσσας αναπαράστασης που χρησιμοποιείται. Τα συστατικά αυτά που τυποποιούν τη γνώση στις οντολογίες είναι τα εξής:

- Κλάσεις (classes):
  - έννοιες που σχετίζονται με ένα πεδίο ή κάποιες εργασίες, οι οποίες είναι συνήθως οργανωμένες σε κάποιο ταξινομικό σύστημα.
  - σε μια οντολογία που αφορά το πανεπιστήμιο: ο «φοιτητής» και ο «καθηγητής» αποτελούν δύο κλάσεις
- Σχέσεις (relations):
  - ένας τύπος αλληλεπίδρασης μεταξύ εννοιών ενός πεδίου, όπως subclass-of, is-a
- Συναρτήσεις (functions)
  - μια ειδική περίπτωση σχέσης στην οποία το v-οστό στοιχείο της σχέσης προσδιορίζεται μοναδικά από τα v-1 προηγούμενα στοιχεία.
  - παράδειγμα: Η τιμή-μεταχειρισμένου-αυτοκινήτου μπορεί να προσδιορίζεται σαν συνάρτηση της αρχικής τιμής του καινούριου αυτοκινήτου, του μοντέλου του αυτοκινήτου, των χαρακτηριστικών του αυτοκινήτου και των χιλιομέτρων που έχει διανύσει

– Αξιώματα (axioms)
    - αναπαριστούν προτάσεις που είναι πάντα αληθείς
    - παράδειγμα: αν ο Φ είναι δευτεροετής φοιτητής τότε μπορεί να εγγραφεί στο επιλεγόμενο μάθημα Μ
– Στιγμιότυπα (instances)
    - αναπαριστούν συγκεκριμένα στοιχεία
    - παράδειγμα: ο φοιτητής με το όνομα Νίκος είναι ένα στιγμιότυπο της κλάσης «φοιτητής»

Οι περισσότεροι ερευνητές συμφωνούν ότι η οντολογία πρέπει να περιλαμβάνει ένα λεξιλόγιο και τους αντίστοιχους ορισμούς, αλλά δεν υπάρχει συναίνεση για έναν πιο λεπτομερή χαρακτηρισμό. Συνήθως, το λεξιλόγιο περιλαμβάνει όρους για τις κλάσεις και τις σχέσεις, ενώ οι ορισμοί των όρων αυτών μπορεί να είναι άτυπο κείμενο, ή μπορεί να προσδιοριστεί με τη χρήση μιας επίσημης γλώσσας, όπως η κατηγορηματική λογική. Το πλεονέκτημα των επισήμων ορισμών είναι ότι επιτρέπουν σε μια μηχανή να εκτελέσει πολύ βαθύτερο συμπερασμό. Το μειονέκτημα είναι ότι οι ορισμοί αυτοί είναι πολύ πιο δύσκολο να κατασκευαστούν.

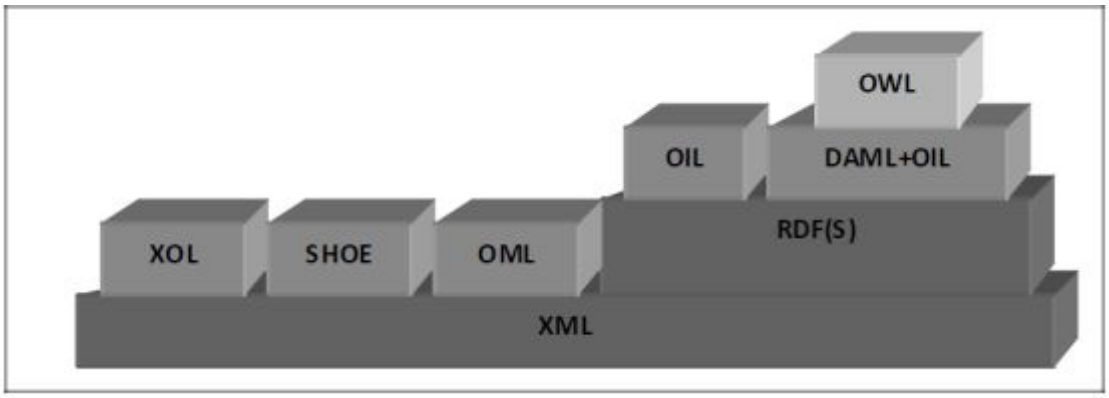
Πολλές οντολογίες έχουν κατασκευαστεί, με διαφορετικά πεδία εφαρμογής, επίπεδα λεπτομέρειας και οπτική. Οι Noy και Hafner [33] παρέχουν μια καλή επισκόπηση και σύγκριση ορισμένων από αυτά τα έργα. Ένα από τα πιο εξέχοντα θέματα στην έρευνα της οντολογίας, είναι η κατασκευή των επαναχρησιμοποιήσιμων στοιχείων. Τα πλεονεκτήματα τέτοιων στοιχείων είναι σαφή: οι μεγάλες οντολογίες μπορούν να κατασκευάζονται γρήγορα συναρμολογώντας και τελειοποιώντας τα υπάρχοντα στοιχεία και η ενοποίηση οντολογιών είναι ευκολότερη, όταν οι οντολογίες μοιράζονται στοιχεία.

Ένας από τους πιο κοινούς τρόπους για την επίτευξη της επαναχρησιμοποίησης (reusability) είναι να επιτραπεί η προδιαγραφή μιας συμπεριληπτικής σχέσης, που αναφέρει ότι μία ή περισσότερες οντολογίες περιλαμβάνονται στη νέα θεωρία. Εάν αυτές οι σχέσεις είναι ακυκλικές και αντιμετωπίζουν όλα τα στοιχεία της περιλαμβανόμενης οντολογίας σαν να είχαν προσδιοριστεί σε τοπικό επίπεδο, τότε μια οντολογία μπορεί να επεκτείνει τις οντολογίες που περιλαμβάνει. Αυτό συμβαίνει για τα

περισσότερα συστήματα, ωστόσο η Ontolingua [34], έχει ακόμα πιο ισχυρά χαρακτηριστικά για τη δυνατότητα της επαναχρησιμοποίησης: συμπεριληπτικές σχέσεις που μπορεί να περιέχουν κύκλους, δυνατότητα περιορισμού αξιωμάτων, και πολυμορφική τελειοποίηση.

Γλώσσες Περιγραφής Οντολογιών

Κατά καιρούς έχουν προταθεί διάφορες μηχανικά αναγνώσιμες γλώσσες, για την αναπαράσταση και ανταλλαγή γνώσης και οντολογιών. Στις πρώτες, ιστορικά, από αυτές τις γλώσσες και τα πρωτόκολλα συγκαταλέγονται η KIF και η OKBC. Ο συνδυασμός των πλεονεκτημάτων που έφερε η XML και η επιθυμία για σημασιολογική σήμανση στο Διαδίκτυο, οδήγησαν σε μια νέα γενιά γλωσσών, κατάλληλων για αναπαράσταση οντολογιών στον Ιστό: η SHOE, η XOL, η OML, η DAML+OIL και η OWL ανήκουν στις γλώσσες αυτές. Οι γλώσσες αυτές είναι βασισμένες στις περιγραφικές λογικές και διαφέρουν μεταξύ τους ως προς την εκφραστικότητα και την μορφή σειριακής διάταξης (serialization) που υποστηρίζουν. Η Εικόνα 2.2 παρουσιάζει την ιεραρχία, από άποψη της χρονολογικής εξέλιξης και εκφραστικής ισχύος, των βασισμένων σε XML γλωσσών αναπαράστασης.



Εικόνα 2.2: Πυραμίδα των γλωσσών αναπαράστασης οντολογιών

2.2 Συλλογιστική - συμπερασμός

Ο τρόπος με τον οποίο εμείς οι άνθρωποι επεξεργαζόμαστε τη γνώση, είναι μέσω της συλλογιστικής (reasoning), δηλαδή της διαδικασίας μέσω της οποίας καταλήγουμε σε συμπεράσματα. Κατ' αναλογία, ένας υπολογιστής επεξεργάζεται τη γνώση που είναι

αποθηκευμένη σε μια βάση γνώσης, εξάγοντας συμπεράσματα από αυτήν, δηλαδή εξάγοντας νέες δηλώσεις που απορρέουν από τις δεδομένες. Οι βασικές λειτουργίες που μπορεί να εκτελέσει ένα σύστημα που βασίζεται στη γνώση, πάνω στη βάση γνώσης του, υποδηλώνονται από τις λειτουργίες tell και ask [24]. Η λειτουργία tell προσθέτει μια νέα δήλωση στη βάση γνώσης, ενώ η λειτουργία ask χρησιμοποιείται για την πραγματοποίηση επερωτήσεων σε αυτά που είναι γνωστά. Οι δηλώσεις που έχουν προστεθεί σε μια βάση γνώσης μέσω της λειτουργίας tell, αποτελούν τη ρητή γνώση που έχει ένα σύστημα για το πεδίο ενδιαφέροντος. Η ικανότητα υπολογιστικής επεξεργασίας της ρητής γνώσης, επιτρέπει σε ένα σύστημα, βασισμένο σε γνώση, να εξάγει συμπεράσματα για μια περιοχή ενδιαφέροντος παράγοντας έμμεση γνώση που προκύπτει από αυτή που έχει δηλωθεί ρητά.

Αυτό οδηγεί στην έννοια της λογικής συνέπειας ή συνεπαγωγής (entailment). Μια βάση γνώσης KB λέγεται ότι συνεπάγεται μια δήλωση *α*, αν η *α* απορρέει από τη γνώση που αποθηκεύεται στην KB και συμβολίζεται ως KB |= *α*. Μια βάση γνώσης συνεπάγεται όλες τις δηλώσεις που έχουν προστεθεί μέσω της λειτουργίας tell, καθώς και εκείνες που είναι λογικές συνέπειες τους. Για παράδειγμα, ας δούμε την ακόλουθη βάση γνώσης με προτάσεις σε λογική πρώτης τάξης:

$$
\begin{aligned}
KB = \{\ & Person(MisterX), participates(MisterX, FL4711), \\
& Flight(FL4711), books(UbiqBiz, FL4711), \\
& \forall x, y, z : (Flight(y) \wedge participates(x, y) \wedge books(z, y) \rightarrow employedAt(x, z)), \\
& \forall x, y : (employedAt(x, y) \rightarrow Company(x) \wedge Employee(y)), \\
& \forall x : (Person(x) \rightarrow \neg Company(x)) \qquad\qquad \}
\end{aligned}
$$

Η βάση γνώσης KB αναφέρει ρητά ότι «ο MisterX είναι ένα άτομο που συμμετέχει στην πτήση FL4711, που έχει κρατηθεί από την UbiqBiz", ότι "οι συμμετέχοντες των πτήσεων απασχολούνται στην εταιρεία που κάνει την κράτηση της πτήσης», ότι «η σχέση εργασίας, ισχύει μεταξύ επιχειρήσεων και εργαζομένων" και ότι "τα άτομα είναι διαφορετικά από τις εταιρείες". Αν κάνουμε την επερώτηση "Είναι ο MisterX εργαζόμενος στην UbiqBiz;", δηλαδή ask(KB, employedAt(MisterX, UbiqBiz)), η απάντηση θα είναι ναι. Η βάση γνώσης KB συνεπάγεται το γεγονός ότι "ο MisterX εργάζεται στην UbiqBiz", δηλαδή KB |= employedAt(MisterX, UbiqBiz), αν και δεν ήταν ρητά δηλωμένο. Αυτό προκύπτει από τις γενικές γνώσεις της, σχετικά με το πεδίο. Μια άλλη συνέπεια είναι ότι «η UbiqBiz είναι μια εταιρεία», δηλαδή KB |= Company(UbiqBiz), η οποία εκφράζεται με μια θετική απάντηση στο ερώτημα ask(KB, Company(UbiqBiz)).

Αυτό προκύπτει από την προηγούμενη συνέπεια σε συνδυασμό με το γεγονός «η σχέση εργασίας, ισχύει μεταξύ επιχειρήσεων και εργαζομένων». Μια άλλη σημαντική έννοια που σχετίζεται με τη συνεπαγωγή, είναι αυτή της συνέπειας (consistency) ή ικανοποιησιμότητας (satisfiability). Διαισθητικά, μια βάση γνώσης είναι συνεπής ή ικανοποιήσιμη αν δεν περιέχει αντιφατικά γεγονότα. Αν προσθέταμε στην παραπάνω βάση γνώσης KB το γεγονός «η UbiqBiz είναι ένα άτομο», λέγοντας tell(KB, Person(UbiqBiz)), θα γινόταν μη ικανοποιήσιμη, γιατί έχει δηλωθεί ότι τα άτομα είναι διαφορετικά από τις εταιρίες. Αναφέραμε ρητά ότι η UbiqBiz είναι ένα άτομο, ενώ την ίδια στιγμή μπορεί να εξαχθεί ότι πρόκειται για μια εταιρεία. Σε γενικές γραμμές, μια μη ικανοποιήσιμη βάση γνώσης δεν είναι πολύ χρήσιμη, δεδομένου ότι σε λογικούς φορμαλισμούς θα συμπεριλάμβανε οποιοδήποτε αυθαίρετο γεγονός. Η λειτουργία ask θα επέστρεφε πάντα ένα θετικό αποτέλεσμα ανεξάρτητα από τις παραμέτρους της, που ασφαλώς δεν είναι επιθυμητό για ένα σύστημα βασισμένο σε γνώση.

Οι διαδικασίες συμπερασμού (inference) που εφαρμόζονται σε υπολογιστικές μηχανές συλλογιστικής (reasoners), αποσκοπούν στην πραγματοποίηση της σχέσης της συνεπαγωγής μεταξύ λογικών δηλώσεων [24]. Εξάγουν έμμεσες δηλώσεις από μια συγκεκριμένη βάση γνώσης ή ελέγχουν αν μια συγκεκριμένη δήλωση έπεται λογικά μιας βάσης γνώσης. Μια διαδικασία συμπερασμού που απορρέει μόνο δηλώσεις που έπονται λογικά, ονομάζεται ορθή (sound). Η ορθότητα είναι ένα επιθυμητό χαρακτηριστικό μιας διαδικασίας συμπερασμού, μιας και μια μη ορθή διαδικασία θα επέφερε δυνητικά λάθος συμπεράσματα. Εάν μια διαδικασία συμπερασμού μπορεί να παραγάγει κάθε πρόταση που απορρέει από μια βάση γνώσης, τότε ονομάζεται πλήρης. Η πληρότητα είναι επίσης μια επιθυμητή ιδιότητα, δεδομένου ότι μια πολύπλοκη αλυσίδα συμπερασμάτων μπορεί να καταρρεύσει, αν λείπει έστω και μία μόνο δήλωση. Ως εκ τούτου, για την εφαρμογή της συλλογιστικής σε συστήματα βασισμένα σε γνώση, επιζητούμε ορθές και πλήρεις διαδικασίες συμπερασμού.

2.2.1 Συλλογιστική ανά μέθοδο αναπαράστασης γνώσης

2.2.1.1 Σημασιολογικά Δίκτυα και πλαίσια

Μια κατηγορία (category) αντικειμένων σε ένα σημασιολογικό δίκτυο κληροδοτεί τα χαρακτηριστικά της, σε κάθε υποκατηγορία της. Αντίστοιχα, ένα πλαίσιο, σε μια ιεραρχία, κληρονομεί χαρακτηριστικά (θυρίδες) και τις αντίστοιχες τιμές από τα υπερπλαίσιά του. Η ιδιότητα αυτή, ονομάζεται κληρονομικότητα (inheritance). Η σημειογραφία των σημασιολογικών δικτύων καθιστά πολύ εύκολη την εκτέλεση συλλογιστικής σχετικά με την κληρονομικότητα. Για παράδειγμα, με βάση την ιδιότητα του να είναι άνθρωπος, η *Μαρία* κληρονομεί την ιδιότητα να έχει δυο πόδια. Συνεπώς, για να διαπιστώσει πόσα πόδια έχει η *Μαρία*, ο αλγόριθμος κληρονομικότητας ακολουθεί το τόξο *ΜέλοςΤου* από τη *Μαρία,* μέχρι την κατηγορία στην οποία ανήκει, και στη συνέχεια ακολουθεί τα τόξα *ΥποσύνολοΤου,* προς τα πάνω στην ιεραρχία, μέχρι να βρει μια κατηγορία, από την οποία να υπάρχει τόξο *Πόδια* – σε αυτήν την περίπτωση, την κατηγορία *Άνθρωποι* (βλ. Εικόνα 2.3).

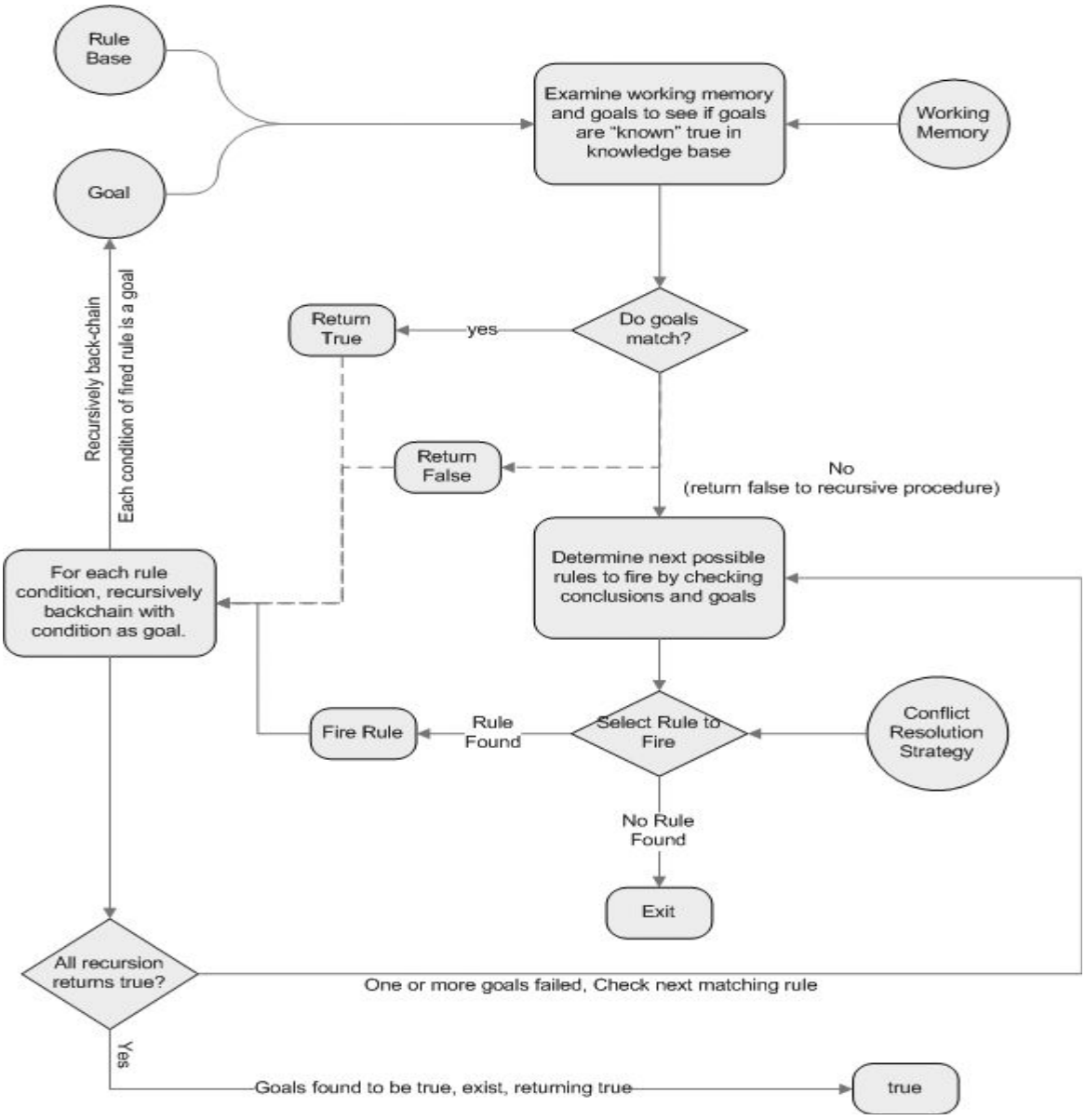
Εικόνα 2.3: Σημασιολογικό δίκτυο

Η απλότητα και αποδοτικότητα αυτού του μηχανισμού συμπερασμού, σε σύγκριση με την απόδειξη λογικών θεωρημάτων, υπήρξε ένα από τους μεγαλύτερους πόλους έλξης των σημασιολογικών δικτύων. Η κληρονομικότητα, γενικά, είναι δυναμική, δηλαδή τα χαρακτηριστικά που κληρονομούνται δεν αντιγράφονται στατικά στο στιγμιότυπο την ώρα της δημιουργίας του, αλλά ανακαλούνται από τα υπερκατηγορίες όταν χρειάζεται, κατά τη διάρκεια μιας διαδικασίας αναζήτησης.

Η κληρονομικότητα είναι ο μοναδικός γενικός μηχανισμός εξαγωγής συμπερασμάτων που διαθέτουν οι γλώσσες οι βασισμένες στα πλαίσια. Συλλογισμός σε μια γλώσσα πλαισίου τυπικά σημαίνει προσπέλαση της τιμής μιας ή περισσότερων θυρίδων ενός πλαισίου-στιγμιοτύπου με μια ενέργεια τύπου read, που πιθανόν να έχει σαν αποτέλεσμα την ενεργοποίηση κάποιας if-needed διαδικασίας.

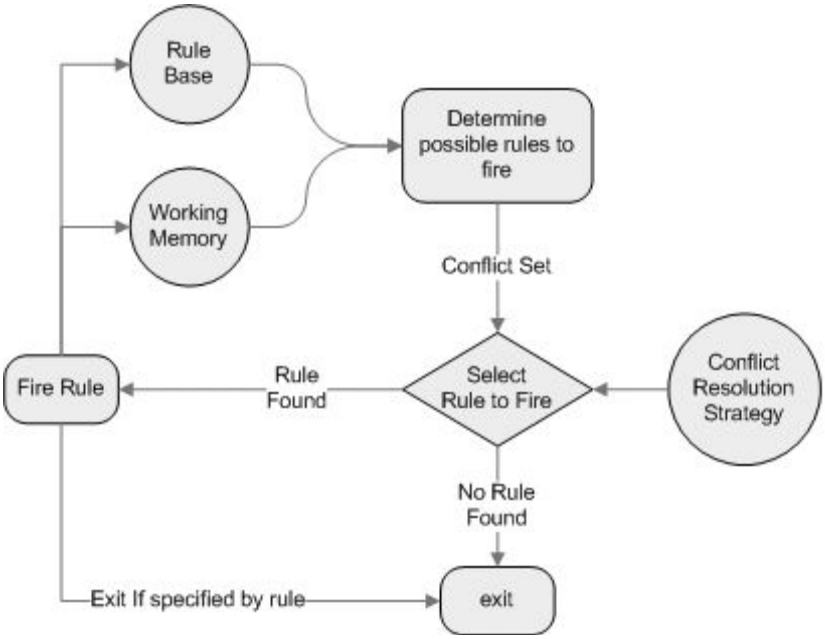
2.2.1.2 Κανόνες

Υπάρχουν δύο μέθοδοι εξαγωγής συμπερασμάτων για γνώση βασισμένη σε κανόνες: η *κατευθυνόμενη από το στόχο* (goal driven) ή *προς τα πίσω αλυσίδα εκτέλεσης* (backward chaining) και η *κατευθυνόμενη από τα δεδομένα* (data driven) ή *προς τα εμπρός αλυσίδα εκτέλεσης* (forward chaining).



Εικόνα 2.4: Προς τα πίσω αλυσίδα εκτέλεσης

Η **προς τα πίσω αλυσίδα εκτέλεσης**, έχει ως σημείο εκκίνησης μία λίστα με στόχους και ενεργεί ανάστροφα για να διαπιστώσει αν υπάρχουν δεδομένα διαθέσιμα που υποστηρίζουν οποιοδήποτε από αυτούς τους στόχους. Μια μηχανή εξαγωγής συμπερασμάτων που χρησιμοποιεί ανάστροφη ακολουθία εκτέλεσης κανόνων, θα αναζητήσει στη βάση γνώσης έναν κανόνα του οποίου το THEN σκέλος συμπίπτει με τον επιθυμητό στόχο. Αν το IF σκέλος του κανόνα δεν ικανοποιείται από τα τρέχοντα γεγονότα της λειτουργικής μνήμης (working memory), τότε προστίθεται στη λίστα με τους στόχους που πρέπει να ικανοποιηθούν.



Εικόνα 2.5: Προς τα εμπρός αλυσίδα εκτέλεσης

Η **προς τα εμπρός αλυσίδα εκτέλεσης**, έχει ως σημείο εκκίνησης τα γεγονότα και χρησιμοποιεί τους κανόνες για να εξάγει περισσότερα γεγονότα, ώσπου να επιτευχθεί η εξαγωγή του αποτελέσματος. Μια μηχανή εξαγωγής συμπερασμάτων που χρησιμοποιεί ορθή ακολουθία εκτέλεσης κανόνων, αναζητά έναν τουλάχιστον κανόνα του οποίου οι προϋποθέσεις ικανοποιούνται από τα γεγονότα της λειτουργικής μνήμης. Μόλις ο κανόνας βρεθεί, το σύστημα μπορεί να εκτελέσει τις ενέργειες που ορίζονται στο δεύτερο σκέλος του κανόνα. Οι γνωστές γλώσσες προγραμματισμού βασισμένου σε κανόνες ακολουθούν κατά βάση αυτή τη μέθοδο.

Σε αυτή την περίπτωση, η διαδικασία εξαγωγής συμπερασμάτων περιλαμβάνει τα ακόλουθα βήματα:

1. Αρχικοποίηση της λειτουργικής μνήμης
2. Εύρεση των κανόνων που ικανοποιούνται (σύνολο σύγκρουσης)
3. Επιλογή ενός κανόνα
4. Πυροδότηση του κανόνα και παραγωγή ενδιάμεσων συμπερασμάτων
5. Ενημέρωση της λειτουργικής μνήμης
6. Αν βρέθηκε λύση ή δεν υπάρχουν κανόνες προς πυροδότηση, τερματισμός.

   Αλλιώς, επαναφορά στο βήμα 2.

Τα βήματα 2-5 αποτελούν έναν κύκλο, που ονομάζεται κύκλος επιλογής-εκτέλεσης (select-execute cycle). Αρχικά, εισάγονται στη λειτουργική μνήμη τα δεδομένα του προβλήματος (δηλαδή τα γνωστά γεγονότα). Στη συνέχεια, βρίσκονται οι κανόνες της βάσης κανόνων (rule base), των οποίων όλες οι συνθήκες/υποθέσεις ικανοποιούνται, δηλαδή ταιριάζουν με γεγονότα στη λειτουργική μνήμη (βήμα 2). Συνήθως, προκύπτουν περισσότεροι του ενός κανόνες, οι οποίοι δημιουργούν ένα σύνολο σύγκρουσης (conflict set), με την έννοια ότι «μάχονται» για το ποιος θα επιλεγεί. Από το σύνολο σύγκρουσης επιλέγεται ένας κανόνας (βήμα 3), ο οποίος και «πυροδοτείται», όπως λέγεται, ή «ενεργοποιείται» (βήμα 4) και παράγει (ενδιάμεσα) συμπεράσματα, τα οποία εισάγονται στη λειτουργική μνήμη (βήμα 5). Αν η προκύπτουσα κατάσταση δίνει λύση στο πρόβλημα ή αν δεν υπάρχουν πια κανόνες που μπορούν να πυροδοτηθούν, τότε σταματά η διαδικασία, αλλιώς ξαναγυρίζει στο βήμα 2.

2.2.1.3 Λογική πρώτης τάξης

Κανόνες εξαγωγής συμπεράσματος

Ένας κανόνας εξαγωγής συμπεράσματος αποτελείται από ένα σύνολο εκφράσεων που ονομάζονται υποθέσεις, και από ένα σύνολο εκφράσεων που λέγονται συμπεράσματα. Μερικοί τέτοιοι κανόνες είναι οι:

- *Τρόπος του Θέτειν (Modus Ponens)*: αν ισχύουν οι υποθέσεις της μορφής ($f_1 \Rightarrow f_2$) και f_1, τότε προκύπτει το συμπέρασμα f_2.

- *Τρόπος του Αναιρείν (Modus Tollens)*: πρόκειται για τον αντίστροφο κανόνα του Modus Ponens. Αν ισχύουν οι υποθέσεις (f_1 => f_2) και ¬f_2, τότε μπορούμε να συμπεράνουμε το ¬f_1.
- *Απαλοιφή Σύζευξης (And Elimination)*: αν ισχύει μια σύζευξη εκφράσεων, τότε ισχύει και κάθε έκφραση που συμμετέχει στη σύζευξη, δηλαδή από την υπόθεση (f_1 /\ f_2) μπορούμε να συμπεράνουμε τις f_1 και f_2.
- *Εισαγωγή Σύζευξης (And Introduction)*: σύμφωνα με αυτόν τον κανόνα, αν ισχύουν ορισμένες υποθέσεις, τότε ισχύει και οι σύζευξή τους, δηλαδή από τις f_1 και f_2 προκύπτει η (f_1 /\ f_2).
- *Καθολικός Προσδιορισμός (Universal Instantiation)*: ο κανόνας αυτός μας επιτρέπει να σκεφτόμαστε από το γενικότερο στο ειδικότερο. Αν ισχύει μια έκφραση που προσδιορίζεται από έναν καθολικό ποσοδείκτη, τότε μπορούμε να συμπεράνουμε ένα στιγμιότυπο αυτής της έκφρασης, στο οποίο η μεταβλητή που προσδιορίζει ο καθολικός ποσοδείκτης έχει αντικατασταθεί από έναν κατάλληλο όρο. Για παράδειγμα, από την έκφραση (∀x)hates(jane, x) προκύπτει ότι hates(jane, mary) ή ακόμα και hates(jane, jane).
- *Υπαρξιακός Προσδιορισμός (Existential Instantiation)*: αν ισχύει μια έκφραση που προσδιορίζεται από έναν υπαρξιακό ποσοδείκτη, μπορούμε να συμπεράνουμε ένα στιγμιότυπο αυτής της έκφρασης, στο οποίο η μεταβλητή που προσδιορίζει ο υπαρξιακός ποσοδείκτης έχει αντικατασταθεί από έναν κατάλληλο όρο. Ο όρος αυτός πρέπει να είναι συνάρτηση των ελεύθερων μεταβλητών της έκφρασης (αν δεν υπάρχουν ελεύθερες μεταβλητές τότε ο όρος είναι μια σταθερά). Για παράδειγμα, από την υπόθεση (∃z) hates(y, z) μπορούμε να συμπεράνουμε την hates(y, f(y)), όπου f μια νέα συνάρτηση του y.

Ένας πολύ σημαντικός κανόνας εξαγωγής συμπεράσματος είναι και η αρχή της επίλυσης (resolution principle). Η αρχή της επίλυσης είναι ένας κανόνας που συνοψίζει σχεδόν όλους τους παραπάνω κανόνες, διατηρώντας ταυτόχρονα μια ξεχωριστή απλότητα. Αυτές οι ιδιότητες κατέστησαν την αρχή της επίλυσης τον ευρύτερα χρησιμοποιούμενο κανόνα εξαγωγής συμπεράσματος σήμερα.

Αν κατά την εξαγωγή της φ από ένα σύνολο υποθέσεων χρησιμοποιούμε μόνο την αρχή της επίλυσης, τότε λέμε ότι έχουμε μια εξαγωγή επίλυσης (resolution deduction) της έκφρασης φ.

Αναγωγή σε προτασιακή λογική

Από την στιγμή που έχουμε κανόνες για το συμπερασμό μη ποσοτικοποιημένων προτάσεων από ποσοτικοποιημένες προτάσεις, γίνεται εφικτή η μετατροπή του συμπερασμού πρώτης τάξης σε προτασιακό συμπερασμό. Η διαδικασία μετατροπής μιας λογικής έκφρασης *G* σε προτασιακή μορφή φαίνεται παρακάτω:

1. Απαλοιφή των συνεπαγωγών από την G:
   - Η $(f_1 \Rightarrow f_2)$ γράφεται σαν $(\neg f_1 \vee f_2)$.
2. Περιορισμός της εμβέλειας των αρνήσεων διανέμοντάς τες στα υπόλοιπα διασυνδετικά:
   - Η $(\neg(\neg f))$ γράφεται σαν f
   - Η $\neg(\forall x)$ f γράφεται σαν $(\exists x)(\neg f)$
   - Η $\neg(\exists x)$ f γράφεται σαν $(\forall x)(\neg f)$
   - Η $\neg(f_1 \wedge f_2 \wedge \ldots \wedge f_n)$ γράφεται σαν $(\neg f_1 \vee \neg f_2 \vee \ldots \vee \neg f_n)$
   - Η $\neg(f_1 \vee f_2 \vee \ldots \vee f_n)$ γράφεται σαν $(\neg f_1 \wedge \neg f_2 \wedge \ldots \wedge \neg f_n)$
3. Μετονομασία μεταβλητών που δεσμεύονται από διαφορετικούς ποσοδείκτες, έτσι ώστε κάθε μεταβλητή να δεσμεύεται από έναν μόνο ποσοδείκτη.
4. Μετατροπή της έκφρασης σε κανονική prenex μορφή. Γίνεται μεταφορά ποσοδεικτών στα αριστερά της έκφρασης, με τη σειρά που εμφανίζονται σ' αυτήν.
5. Απαλοιφή των υπαρξιακών ποσοδεικτών χρησιμοποιώντας συναρτήσεις και σταθερές Skolem (Skolemization).
6. Διαγραφή των καθολικών ποσοδεικτών.
7. Μετατροπή της έκφρασης σε συζευκτική κανονική μορφή (CNF).

- Η $(f \vee (f_1 \wedge f_2 \wedge \ldots \wedge f_n))$ γράφεται σαν $((f \vee f_1) \wedge (f \vee f_2) \wedge \ldots \wedge (f \vee f_n))$

8. Απαλοιφή των διασυνδετικών και γραφή του συνόλου των προτάσεων που προέκυψαν.
9. Μετονομασία των μεταβλητών, έτσι ώστε μια μεταβλητή να μην εμφανίζεται σε περισσότερες από μία προτάσεις.

Κατά τη διαδικασία **Skolemization** αντικαθιστούμε τις μεταβλητές που δεσμεύονται από υπαρξιακό ποσοδείκτη με ειδικές σταθερές Skolem ή ειδικές Skolem συναρτήσεις μεταβλητών, ανάλογα με το αν υπάρχουν ή όχι καθολικοί ποσοδείκτες πριν τον αντίστοιχο υπαρξιακό. Μετά από την αντικατάσταση κάθε τέτοιας μεταβλητής στην έκφραση, απαλείφεται ο αντίστοιχος υπαρξιακός ποσοδείκτης.

Αυτή η τεχνική μετατροπής σε προτασιακή μορφή (propositionalization) μπορεί να γενικευτεί πλήρως · δηλαδή, κάθε βάση γνώσης και ερώτημα πρώτης τάξης μπορούν να μετατραπούν σε προτασιακά έτσι ώστε να διατηρηθεί η λογική κάλυψη. Παραμένει όμως ακόμα ένα πρόβλημα: Όταν η βάση γνώσης περιλαμβάνει ένα συναρτησιακό σύμβολο, το σύνολο των δυνατών αντικαταστάσεων βασικών όρων είναι άπειρο.

Αντικατάσταση και Ενοποίηση

Με τον όρο αντικατάσταση εννοούμε ένα πεπερασμένο σύνολο της μορφής $\{t_1/v_1,...,t_n/v_n\}$, όπου $v_1,...,v_n$ είναι μεταβλητές διάφορες μεταξύ τους και $t_1,...,t_n$ είναι όροι, τέτοιοι ώστε $v_i \neq t_i$ (ήπιος ορισμός) και καμιά μεταβλητή v_i να μην εμφανίζεται σε κάποιον από τους $t_1,\ldots,t_n$ (αυστηρός ορισμός). Οι μεταβλητές $v_1,...,v_n$ λέγονται δεσμευμένες (bound), και οι όροι $t_1,\ldots,t_n$ λέγονται προσδέσεις (bindings) των αντίστοιχων μεταβλητών. Για παράδειγμα, το σύνολο {c/x, f(b)/y, z/w} είναι μια αντικατάσταση στην οποία η μεταβλητή x συνδέεται με τη σταθερά c, η y με τον όρο f(b), και η z με τη μεταβλητή w. Αντίθετα, το σύνολο {g(y)/x, f(x)/y}, ενώ είναι αντικατάσταση με τον ήπιο ορισμό, δεν είναι αντικατάσταση με τον αυστηρό ορισμό, αφού η x εμφανίζεται στον όρο f(x) και η y στον όρο g(y). Συνήθως χρησιμοποιείται ο ήπιος ορισμός.

Μια αντικατάσταση ϑ μπορεί να εφαρμοστεί σε μια έκφραση του κατηγορηματικού λογισμού E για να προκύψει μια νέα έκφραση $E\vartheta$ (που ονομάζεται στιγμιότυπο αντικατάστασης), αντικαθιστώντας κάθε δεσμευμένη μεταβλητή με την πρόσδεσή της στο σύνολο αντικατάστασης. Για παράδειγμα, αν ϑ={a/x, f(b)/y, c/z} και E=p(x,y,z), τότε $E\vartheta$=p(a, f(b), c).

Έστω $\theta=\{t_1/x_1,...,t_n/x_n\}$ και $\lambda=\{u_1/y_1,\ldots,u_m/y_m\}$ δύο αντικαταστάσεις. Τότε η σύνθεση των θ και λ είναι η αντικατάσταση $\theta\circ\lambda$, η οποία προκύπτει από το σύνολο

$$\{t_1\lambda/x_1,\ldots, t_n\lambda/x_n, u_1/y_1,\ldots, u_m/y_m\}$$

με διαγραφή κάθε στοιχείου $t_n\lambda/x_n$ για το οποίο $t_n\lambda=x_n$, και κάθε στοιχείου u_i/y_i τέτοιου ώστε το $y_i \in \{x_1,\ldots,x_n\}$. Για παράδειγμα, αν $\theta=\{f(y)/x, y/z\}$ και $\lambda=\{a/x, b/y, c/z\}$, τότε προκύπτει σαν ενδιάμεσο βήμα, το σύνολο $\{f(b)/x, b/z, a/x, b/y, c/z\}$. Τα στοιχεία a/x και c/z διαγράφονται, οπότε

$$\theta\circ\lambda = \{f(b)/x, b/y, b/z\}$$

Ένα σύνολο εκφράσεων $\{E_1,...,E_\eta\}$ λέγεται ενοποιήσιμο (unifiable), αν και μόνο αν υπάρχει μια αντικατάσταση ϑ, τέτοια ώστε να κάνει όλες τις εκφράσεις ίδιες, δηλαδή $E_1\vartheta=E_2\vartheta=\ldots=E_n\vartheta$. Η ϑ λέγεται ενοποιήτρια (unifier) του συνόλου, και η διαδικασία εύρεσης και εφαρμογής της στο σύνολο των εκφράσεων ονομάζεται ενοποίηση (unification).

Για παράδειγμα, η αντικατάσταση {a/x, b/y, c/z} ενοποιεί τις εκφράσεις p(a, y, z) και p(x,b,z), παράγοντας την p(a, b, c). Παρατηρούμε, όμως, ότι αυτή η αντικατάσταση δεν είναι η μόνη ενοποιήτρια των δύο εκφράσεων, αφού π.χ. δεν χρειάζεται να αντικαταστήσουμε το z με το c, οπότε και το {a/x, b/y} είναι μια ενοποιήτρια. Επομένως, είναι χρήσιμο να ορίσουμε την έννοια της γενικότερης ενοποιήτριας (most general unifier) ενός συνόλου ως εξής :

Μια ενοποιήτρια σ ενός συνόλου $\{E_1,...,E_n\}$ ονομάζεται γενικότερη ενοποιήτρια αν και μόνο αν για κάθε ενοποιήτρια ϑ του συνόλου, υπάρχει μια αντικατάσταση λ, τέτοια ώστε $\vartheta = \sigma \circ \lambda$.

Αρχή της Επίλυσης

Η αρχή της επίλυσης είναι ο πιο ισχυρός κανόνας εξαγωγής συμπερασμάτων και εφαρμόζεται στην προτασιακή μορφή της λογικής πρώτης τάξης. Επειδή, όμως, η αρχή της επίλυσης δεν είναι πλήρης από μόνη της, συνήθως συνδυάζεται με έναν άλλο απλό κανόνα (ή διαδικασία), την παραγοντοποίηση (factoring). Στη συνέχεια, παραθέτουμε τους ορισμούς που συνθέτουν τον συνδυασμό των δύο κανόνων.

Ορισμός *(παραγοντοποίηση):* Αν δύο ή περισσότερα στοιχεία μιας πρότασης C με την ίδια πολικότητα, δηλαδή είτε και τα δύο αρνητικά ή και τα δύο θετικά, έχουν μια γενικότερη ενοποιήτρια γ, τότε το Cγ καλείται παράγοντας (factor) της C. Αν το Cγ είναι μια μοναδιαία πρόταση, τότε καλείται μοναδιαίος παράγοντας (unit factor) της C.

Ορισμός *(αρχή της επίλυσης):* Έστω δύο προτάσεις C_1 και C_2, που ονομάζονται γονικές (parents), με μη κοινές μεταβλητές. Έστω L_1 και L_2 δύο στοιχεία των C_1 και C_2 αντίστοιχα. Αν τα L_1 και $\neg L_2$ έχουν μια γενικότερη ενοποιήτρια σ, τότε η πρόταση $(C_1\sigma - L_1\sigma) \cup (C_2\sigma - L_2\sigma)$ ονομάζεται δυαδική επιλύουσα (binary resolvent) των C_1 και C_2.

Για παράδειγμα, έστω ότι έχουμε τις προτάσεις (p(x), q(x)) και (¬p(a), r(y)). Επιλέγοντας για L_1 και L_2 τα στοιχεία p(x) και ¬(p(a), τότε η γενικότερη ενοποιήτρια είναι σ={a/x}. Έτσι η δυαδική επιλύουσα των δύο προτάσεων είναι η πρόταση (q(a), r(y)).

Ορισμός *(συνδυασμένος κανόνας):* Επιλύουσα (resolvent) των δύο προτάσεων C_1 και C_2 είναι ένα από τα παρακάτω :

1. η δυαδική επιλύουσα των C_1 και C_2.

2. η δυαδική επιλύουσα της C_1 και ενός παράγοντα της C_2.

3. η δυαδική επιλύουσα ενός παράγοντα της C_1 και της C_2.

4. η δυαδική επιλύουσα ενός παράγοντα της C_1 και ενός παράγοντα της C_2.

Για παράδειγμα, έστω C_1 = (p(x), p(f(y)). r(g(y))) και C_2 = (¬p(f(g(a)), q(b)). Ένας παράγοντας της C_1 είναι η C'_1 =(p(f(y)), r(g(y))). Μια δυαδική επιλύουσα των C'_1 και C_2 είναι η C_3=(r(g(g(a))), q(b)). Επομένως, η C_3 είναι μια επιλύουσα των C_1 και C_2. Οι προτάσεις C_1 και C_2 λέγονται γονικές της C_3. Πιο συγκεκριμένα, η C_1 είναι ο αριστερός γονέας και η C_2 ο δεξιός γονέας της C_3.

Για να γίνει φανερό ότι η αρχή της επίλυσης δεν θα ήταν πλήρης αν δε χρησιμοποιούσε την έννοια του παράγοντα, παραθέτουμε το παρακάτω παράδειγμα. Πράγματι, χρησιμοποιώντας μόνο την έννοια της δυαδικής επιλύουσας είναι αδύνατο να εξάγουμε την κενή πρόταση από το σύνολο των προτάσεων (p(u), p(v)) και (¬p(x), ¬p(y)). Με τη χρήση του παράγοντα και τον ορισμό της επιλύουσας επιτυγχάνουμε την απλοποίηση παρόμοιων προτάσεων και εξασφαλίζουμε την πληρότητα της αρχής της επίλυσης.

Αντίφαση της Επίλυσης - Απόδειξη θεωρήματων

Σύμφωνα με τον ορισμό της λογικής συνεπαγωγής, για να βεβαιωθούμε αν μια έκφραση φ συνεπάγεται λογικά από ένα σύνολο εκφράσεων S θα πρέπει να εξετάσουμε αν όλες οι ερμηνείες που ικανοποιούν τις εκφράσεις του S ικανοποιούν και την φ. Αλλά αυτό απαιτεί άπειρο χρόνο, αφού υπάρχουν άπειρες ερμηνείες για ένα σύνολο εκφράσεων.

Ευτυχώς υπάρχει ένα θεώρημα, σύμφωνα με το οποίο αν το σύνολο των εκφράσεων S∪{φ} είναι ασυνεπές, τότε S |= ¬φ. Έτσι, αν θέλουμε να δούμε αν μια έκφραση φ συνεπάγεται λογικά από ένα σύνολο υποθέσεων S αρκεί να δείξουμε ότι το S ∪ {¬φ} είναι μη ικανοποιήσιμο.

Αυτό μπορεί να γίνει με τη χρήση της αρχής της επίλυσης. Πραγματικά, αν ένα σύνολο προτάσεων S' είναι μη ικανοποιήσιμο, τότε μπορούμε πάντα με τη χρήση της αρχής της επίλυσης να εξάγουμε την κενή πρόταση, η οποία δηλώνει την ύπαρξη αντίφασης στο S'. Έτσι, για να αποδείξουμε ότι μια πρόταση φ συνεπάγεται λογικά από ένα σύνολο προτάσεων S, δηλαδή S |= φ, εφαρμόζουμε διαρκώς την αρχή της επίλυσης στο σύνολο S' = S ∪ {¬φ} μέχρι να εξάγουμε την κενή πρόταση, οπότε το σύνολο S' είναι ασυνεπές και άρα S |= φ. Η διαδικασία αυτή ονομάζεται αντίφαση της επίλυσης (resolution refutation).

Η αρχή της επίλυσης είναι ορθή (sound) και πλήρης ως προς την αντίφαση (refutation complete). Ισχύουν δηλαδή τα παρακάτω θεωρήματα:

Ορθότητα (Soundness) Αν υπάρχει μια εξαγωγή επίλυσης μιας πρότασης φ από ένα σύνολο προτάσεων S, τότε η φ συνεπάγεται λογικά από το S.

Πληρότητα (Completeness) Αν ένα σύνολο από προτάσεις S είναι μη ικανοποιήσιμο, τότε υπάρχει μια εξαγωγή επίλυσης της κενής πρότασης από το S.

Όπως στους κανόνες, έτσι και στη λογική πρώτης τάξης χρησιμοποιούνται οι αλγόριθμοι της προς τα εμπρός αλυσίδας εκτέλεσης και της προς τα πίσω αλυσίδας εκτέλεσης, που μαζί με τα συστήματα απόδειξης θεωρημάτων αποτελούν τις τρεις μεγάλες οικογένειες αλγορίθμων συμπερασμού πρώτης τάξης.

2.2.1.4 Περιγραφικές λογικές – Οντολογίες

Η επικρατούσα γλώσσα οντολογιών, OWL, βασίζεται στις γλώσσες Περιγραφικών Λογικών. Στη συνέχεια, παρουσιάζονται οι μέθοδοι εξαγωγής συμπερασμάτων για τις περιγραφικές λογικές, καλύπτοντας και την αναπαράσταση γνώσης με τη χρήση οντολογιών.

Διαδικασίες συλλογιστικής για το Tbox

Με βάση τη λογική, μια έννοια έχει νόημα εάν υπάρχει κάποια ερμηνεία που ικανοποιεί τα αξιώματα που ορίζονται σε μια ορολογία (Tbox) Τα, εάν δηλαδή αποτελεί μοντέλο της Τα, έτσι ώστε η έννοια να δηλώνει ένα μη-κενό σύνολο σε αυτήν την ερμηνεία. Μια έννοια που έχει αυτή την ιδιότητα ονομάζεται ικανοποιήσιμη με βάση την Τ και μη-ικανοποιήσιμη διαφορετικά.

Ο έλεγχος ικανοποιησιμότητας (satisfiability check) των εννοιών αποτελεί μια θεμελιώδη διαδικασία εξαγωγής συμπερασμάτων. Ένας μεγάλος αριθμός από άλλες διαδικασίες εξαγωγής συμπερασμάτων μπορούν να αναχθούν στο πρόβλημα της ικανοποιησιμότητας ή μη-ικανοποιησιμότητας. Για να ελεγχθεί εάν ένα μοντέλο είναι σωστό, ή για να γίνει βελτιστοποίηση επερωτήσεων που έχουν την μορφή εννοιών, είναι απαραίτητο να είναι γνωστό εάν κάποια έννοια είναι πιο γενική από κάποια άλλη. Αυτό αποτελεί το πρόβλημα υπαγωγής (subsumption). Μια έννοια C υπάγεται σε μια έννοια D εάν σε κάθε μοντέλο της Τ το σύνολο που προκύπτει από την C είναι υποσύνολο του συνόλου που προκύπτει από την D. Μέσω της υπαγωγής επιτυγχάνεται η κατηγοριοποίηση (classification) των εννοιών, δηλαδή η τοποθέτηση κάθε έννοιας στην κατάλληλη θέση της ιεραρχίας των εννοιών. Άλλες χρήσιμες σχέσεις μεταξύ εννοιών είναι η σχέση ισοδυναμίας (equivalence) και η σχέση ξένων εννοιών (disjointness). Οι ιδιότητες αυτές ορίζονται τυπικά παρακάτω.

Έστω ότι η T είναι μια ορολογία.

- **Ικανοποιησιμότητα (satisfiability):** Μια έννοια C είναι ικανοποιήσιμη με βάση την T, εάν υπάρχει ένα μοντέλο I της T τέτοιο ώστε το σύνολο C^I να είναι μη-κενό. Στην περίπτωση αυτή η ερμηνεία I είναι μοντέλο της C.
- **Υπαγωγή (subsumption):** Μια έννοια C υπάγεται σε μια έννοια D με βάση την T, εάν $C^I \subseteq D^I$ για κάθε μοντέλο I της T. Στην περίπτωση αυτή γράφεται $C \subseteq_T D$ ή $T \models C \subseteq D$.
- **Ισοδυναμία (equivalence):** Δύο έννοιες C και D είναι ισοδύναμες με βάση την T, εάν $C^I = D^I$ για κάθε μοντέλο I της T. Στην περίπτωση αυτή γράφεται $C \equiv_T D$ ή $T \models C \equiv D$.
- **Ξένες Έννοιες (disjointness):** Δύο έννοιες C και D είναι ξένες μεταξύ τους με βάση την T, εάν $C^I \cap D^I = \varnothing$ για κάθε μοντέλο I της T.

Εάν η ορολογία T είναι ανεξάρτητη από τα συμφραζόμενα, πολλές φορές η συνθήκη «με βάση την T» παραλείπεται. Η συνθήκη αυτή παραλείπεται και στην ειδική περίπτωση όπου η ορολογία (TBox) είναι κενή. Τότε $\models C \subseteq D$, εάν η έννοια C υπάγεται στην έννοια D, και $\models C \equiv D$ εάν οι C και D είναι ισοδύναμες.

Εξαλείφοντας το TBox

Στις διάφορες εφαρμογές, οι έννοιες συνήθως περιέχονται στο περιβάλλον ενός TBox. Παρόλα αυτά, για την ανάπτυξη διαδικασιών είναι εννοιολογικά πιο εύκολο να αφαιρεθούν από το TBox ή να υποτεθεί ότι το TBox είναι άδειο.

Εικόνα 2.6: (α) παράδειγμα TBox, (β) επέκταση του TBox του (α)

Εάν η ορολογία (TBox) T είναι μη-κυκλική είναι εφικτό τα προβλήματα που προκύπτουν με βάση τη T να αναχθούν σε προβλήματα με βάση το κενό TBox. Ισχύει ότι η T είναι

ισοδύναμη με την επέκταση της Τ'. Για κάθε έννοια C ορίζεται η επέκτασή της με βάση την Τ ως η έννοια C' που προκύπτει από τη C αντικαθιστώντας κάθε σύμβολο ονόματος Α στη C με την έννοια D, όπου Α ≡ D είναι ο ορισμός της Α στην Τ'. Για παράδειγμα, η επέκταση της φράσης

Γυναίκα Π Άντρας

με βάση την ορολογία της Εικόνας 2.6 (α), λαμβάνοντας υπ όψιν την επέκταση της στο (β) και αντικαθιστώντας τις έννοιες «Γυναίκα» και «Άντρας» με το δεξί μέλος των ορισμών τους από αυτή την επέκταση είναι η έννοια

Άτομο Π Θηλυκό Π Άτομο Π ¬(Άτομο Π Θηλυκό).

Εφόσον η επέκταση C' προκύπτει από την C αντικαθιστώντας τα ονόματα με περιγραφές, με τέτοιο τρόπο ώστε και οι δύο να μπορούν να ερμηνευθούν με τον ίδιο τρόπο από ένα μοντέλο της Τ, προκύπτει ότι:

- $C \equiv_T C'$.

Για το λόγο αυτό η C είναι ικανοποιήσιμη με βάση την Τ εάν και μόνο εάν η C' είναι ικανοποιήσιμη με βάση την Τ. Παρόλα αυτά, η C' δεν περιέχει ονόματα που έχουν οριστεί και γι' αυτό είναι ικανοποιήσιμη με βάση την Τ εάν και μόνο εάν είναι ικανοποιήσιμη. Άρα:

- η C είναι ικανοποιήσιμη με βάση την Τ εάν και μόνο εάν είναι ικανοποιήσιμη.

Εάν η D είναι μια άλλη έννοια τότε ισχύει και $D \equiv_T D'$. Συνεπώς, ισχύει $C \subseteq_T D$ εάν και μόνο εάν $C' \subseteq_T D'$ και $C \equiv_T D$ εάν και μόνο εάν $C' \equiv_T D'$. Εφόσον οι έννοιες C' και D' περιέχουν μόνο βασικά σύμβολα ισχύει:

- $T \models C \subseteq D$ εάν και μόνο εάν $T \models C' \subseteq D'$,
- $T \models C \equiv D$ εάν και μόνο εάν $T \models C' \equiv D'$.

Με τον ίδιο τρόπο μπορεί να δειχθεί ότι:

- οι C και D είναι ξένες μεταξύ τους με βάση την Τ εάν και μόνο εάν οι C' και D' είναι ξένες μεταξύ τους.

Συνοψίζοντας, γίνεται αντιληπτό ότι η επέκταση των εννοιών με βάση κάποια ορολογία έχει ως αποτέλεσμα το TBox να μην χρειάζεται πλέον. Η επέκταση αυτή μπορεί να είναι υπολογιστικά δαπανηρή, εφόσον στην χειρότερη περίπτωση το μέγεθος της T' είναι εκθετικό σε σχέση με το μέγεθος της T και για το λόγο αυτό η C' μπορεί να έχει πιο μεγάλο μέγεθος από την C κατά ένα παράγοντα εκθετικό ως προς το μέγεθος της T.

Διαδικασίες συλλογιστικής για το Abox

Όπως έχει αναφερθεί και παραπάνω, ένα ABox περιλαμβάνει δύο είδη αναθέσεων, τις αναθέσεις εννοιών που έχουν τη μορφή C(α) και τις αναθέσεις ρόλων που έχουν τη μορφή R(α, β). Η αναπαράσταση αυτής της γνώσης πρέπει να είναι **συνεπής (consistent)** για να μη δημιουργούνται αντιφάσεις και να μην προκύπτουν αυθαίρετα συμπεράσματα. Υπεύθυνο για αυτό τον έλεγχο είναι το σύστημα περιγραφικής λογικής που συνδυάζοντας τις αναθέσεις και τα αξιώματα που δηλώνονται στην ορολογία πρέπει να είναι σε θέση να αποφανθεί εάν οι δηλώσεις αυτές είναι συνεπείς.

Ένα ABox A λέγεται ότι είναι συνεπές με βάση μια ορολογία T, εάν υπάρχει μια ερμηνεία που να είναι μοντέλο και του A και της T. Λέγεται ότι το A είναι συνεπές, εάν είναι συνεπές με βάση την άδεια ορολογία (το άδειο T Box). Για παράδειγμα, το σύνολο των αναθέσεων {Μητέρα(ΜΑΡΙΑ), Πατέρας(ΜΑΡΙΑ)} είναι συνεπές (με βάση το άδειο TBox), επειδή χωρίς κάποιους επιπλέον περιορισμούς στην ερμηνεία των «Μητέρα» και «Πατέρας», οι δύο έννοιες μπορούν να ερμηνευτούν με τέτοιο τρόπο, ώστε να έχουν ένα κοινό στοιχείο. Από την άλλη όμως, οι αναθέσεις αυτές δεν είναι συνεπείς με βάση το TBox της Εικόνας 2.6 μιας και σε κάθε μοντέλο του οι έννοιες «Μητέρα» και «Πατέρας» είναι ξένες μεταξύ τους.

Παρόμοια με τις έννοιες, ο έλεγχος συνέπειας ενός ABox με βάση μια μη-κυκλική ορολογία μπορεί να αναχθεί στον έλεγχο ενός εκτεταμένου ABox. Η επέκταση του A με βάση την T ορίζεται ως το ABox A' που προκύπτει από το A με την αντικατάσταση κάθε ανάθεσης έννοιας C(α) στο A με την ανάθεση C'(α), όπου C' είναι η επέκταση της C με βάση την T. Σε κάθε μοντέλο της T, μια έννοια C και η επέκτασή της C' ερμηνεύονται με τον ίδιο τρόπο. Για το λόγο αυτό, το A' είναι συνεπές με βάση την T, εάν και μόνο εάν είναι συνεπές και το A. Επειδή όμως, το A' δεν περιέχει σύμβολα ονομάτων που ορίζονται στην T, είναι συνεπές με βάση την T, εάν είναι συνεπές. Άρα:

- Το Α είναι συνεπές με βάση την ορολογία Τ, εάν και μόνο εάν η επέκταση Α' είναι συνεπής.
-

Με βάση ένα ABox Α, μπορούν να τεθούν επερωτήσεις όσον αφορά τις σχέσεις μεταξύ εννοιών, ρόλων και ατόμων. Η πρωταρχική διαδικασία εξαγωγής συμπερασμάτων στα ABox που περιέχει τέτοιες επερωτήσεις βασίζεται στον **έλεγχο στιγμιότυπου (instance checking)**, ή στον έλεγχο εάν μια ανάθεση συνεπάγεται από ένα ABox. Το Α **συνεπάγεται (entails)** μια ανάθεση α, Α |= α, εάν κάθε ερμηνεία που ικανοποιεί το Α ικανοποιεί ταυτόχρονα και την α. Εάν η α είναι μια ανάθεση ρόλου, ο έλεγχος στιγμιότυπου είναι απλός, εφόσον η περιγραφική γλώσσα δεν περιέχει κατασκευαστές για τη δημιουργία πιο σύνθετων ρόλων. Εάν η α είναι της μορφής C(α), μπορεί να γίνει αναγωγή του ελέγχου στιγμιότυπου στον έλεγχο συνέπειας για τα ABox εφόσον ισχύει:

- Α |= C(α), εάν και μόνο εάν η Α $\cup$ {$\neg$C(α)} είναι μη-συνεπής.

Επίσης, και η συλλογιστική που αφορά έννοιες μπορεί να αναχθεί στον έλεγχο συνέπειας. Ισχύει:

- η C είναι ικανοποιήσιμη, εάν και μόνο εάν η {C(α)} είναι συνεπής

όπου α είναι ένα αυθαίρετο όνομα ατόμου. Αντίθετα, έχει δειχθεί ότι η συνέπεια ενός ABox μπορεί να αναχθεί στην ικανοποιησιμότητα εννοιών σε γλώσσες που περιέχουν τους κατασκευαστές "ένα από" και "γεμίζει". Εάν οι κατασκευαστές αυτοί δεν υπάρχουν ο έλεγχος στιγμιότυπου μπορεί να αποδειχθεί πιο δύσκολος από τα προβλήματα ικανοποιησιμότητας και υπαγωγής.

Στις διάφορες εφαρμογές, συνήθως απαιτούνται πιο πολύπλοκες διαδικασίες εξαγωγής συμπερασμάτων από τον έλεγχο συνέπειας και στιγμιότυπου. Θεωρώντας μια βάση γνώσης σαν ένα μέσο αποθήκευσης πληροφοριών για κάποια άτομα, μια χρήσιμη λειτουργία μπορεί να είναι η ανεύρεση των ατόμων που είναι στιγμιότυπα μιας έννοιας C. Κάτι τέτοιο αποτελεί το **πρόβλημα ανάκτησης (retrieval problem)**. Το πρόβλημα ανάκτησης, δεδομένου ενός ABox Α και μιας έννοιας C, είναι η ανεύρεση όλων των ατόμων α έτσι ώστε Α |= C(α). Ένας μη-αποδοτικός αλγόριθμος για την λύση του παραπάνω προβλήματος είναι ο έλεγχος εάν κάθε άτομο, που περιέχεται στο ABox, είναι στιγμιότυπο της έννοιας C.

Το δυαδικό ανάλογο της ανάκτησης είναι το **πρόβλημα πραγμάτωσης (realization problem)** το οποίο ορίζεται ως εξής: δεδομένου ενός ατόμου α και ενός συνόλου από έννοιες, ζητούνται οι πιο συγκεκριμένες έννοιες για τις οποίες A |= C(α). Με τον όρο συγκεκριμένες έννοιες εννοούνται οι έννοιες εκείνες που βρίσκονται χαμηλότερα στην ιεραρχία, δηλαδή στο αριστερό μέλος του τελεστή υπαγωγής $\subseteq$.

2.2.1.5 Λογικός προγραμματισμός

Τα συστήματα λογικού προγραμματισμού επιχειρούν να αποδείξουν ένα στόχο μέσω της επίλυσης ή της προς τα πίσω αλυσίδας εκτέλεσης. Η πιο ευρέως χρησιμοποιούμενη γλώσσα λογικού προγραμματισμού είναι η Prolog.

Η Prolog βασίζεται σε μια ειδική μορφή επίλυσης που λέγεται SLD-επίλυση (SLD-resolution - Linear resolution with a Selection function for Definite clauses). Σε κάθε βήμα του διερμηνέα της Prolog, εφαρμόζεται η αρχή της επίλυσης ανάμεσα σε ένα στόχο που θέλουμε να αποδείξουμε και ένα γεγονός ή ένα κανόνα του προγράμματος. Έτσι, αποδεικνύεται ο στόχος (στην περίπτωση του γεγονότος) ή παράγεται ένα νέο σύνολο στόχων (στην περίπτωση του κανόνα) που προστίθενται στους στόχους που έχουμε να αποδείξουμε.

Η εκτέλεση των προγραμμάτων της Prolog γίνεται μέσω προς τα πίσω αλυσίδας εκτέλεσης με προτεραιότητα βάθους, όπου οι προτάσεις δοκιμάζονται με τη σειρά με την οποία είναι γραμμένες στη βάση γνώσης. Μεριές πλευρές της Prolog είναι εκτός του πρότυπου λογικού συμπερασμού [35]:

- Υπάρχει ένα σύνολο από ενσωματωμένες συναρτήσεις για αριθμητικές πράξεις. Τα λεκτικά που χρησιμοποιούν αυτά τα συναρτησιακά σύμβολα "αποδεικνύονται" με την εκτέλεση κώδικα και όχι με τη διεξαγωγή περαιτέρω συμπερασμού. Για παράδειγμα, ο στόχος "X is 4+3" επιτυγχάνει με το X να δεσμεύεται στο 7. Από την άλλη πλευρά, ο στόχος "5 is X+Y" αποτυγχάνει, επειδή οι ενσωματωμένες συναρτήσεις δεν κάνουν επίλυση τυχαίων εξισώσεων.
- Υπάρχουν ενσωματωμένα κατηγορήματα των οποίων η εκτέλεση προκαλεί παρενέργειες. Στα κατηγορήματα αυτά περιλαμβάνονται κατηγορήματα εισόδου εξόδου, καθώς και τα κατηγορήματα assert/retract για την τροποποίηση της βάσης γνώσης. Τέτοια κατηγορήματα δεν έχουν αντίστοιχο στη λογική και μπορούν να δημιουργήσουν κάποια περίεργα αποτελέσματα – όπως, για παράδειγμα, σε περίπτωση που προστεθούν (assert) γεγονότα από έναν κλάδο του δέντρου

απόδειξης ο οποίος τελικά αποτυγχάνει. Η Prolog επιτρέπει μια μορφή άρνησης η οποία ονομάζεται άρνηση ως αποτυχία (negation as failure). Ένας αρνημένος στόχος not P θεωρείται αποδεδειγμένος αν το σύστημα αποτύχει να αποδείξει το P. Έτσι, η πρόταση

ζωντανός(X) :- not νεκρός(X).

μπορεί να διαβαστεί ως «Οποιοσδήποτε είναι ζωντανός αν δεν είναι αποδεδειγμένα νεκρός».

- Η Prolog διαθέτει έναν τελεστή ισότητας. =, αλλά δεν έχει την πλήρη ισχύ της λογικής ισότητας. Ένας στόχος ισότητας επιτυγχάνει όταν οι δυο όροι είναι ενοποιήσιμοι (unifiable), και αποτυγχάνει σε άλλη περίπτωση. Έτσι, η παράσταση X+Y=2+3 επιτυγχάνει με το X να δεσμεύεται στο 2 και το Y να δεσμεύεται στο 3, αλλά η παράσταση αυγερινός = αποσπερίτης, αποτυγχάνει. (Στην κλασική λογική, η τελευταία ισότητα μπορεί να είναι ή να μην είναι αληθής). Γεγονότα ή κανόνες που σχετίζονται με ισότητες δεν μπορούν να προστεθούν.
- Ο αλγόριθμος ενοποίησης της Prolog παραλείπει τον έλεγχο ύπαρξης (occur check). Αυτό σημαίνει ότι υπάρχει περίπτωση να πραγματοποιηθούν κάποιες επισφαλείς εξαγωγές συμπερασμάτων αυτό σπανίως αποτελεί πρόβλημα, εκτός της περίπτωσης χρήσης της Prolog για την απόδειξη μαθηματικών θεωρημάτων.

3. MOBILE REASONING

Οι περισσότεροι κινητοί χρήστες σήμερα είναι εξοπλισμένοι με πρόσβαση στο Διαδίκτυο μέσω ενός ευρέως φάσματος διαφορετικών φορέων. Με την μεταπήδηση από τον επιτραπέζιο υπολογιστή στο πανταχού παρόν μοντέλο, το υπολογιστικό σύστημα θα πρέπει τώρα να προσαρμοστεί στην κατάσταση του χρήστη, αντί της προσαρμογής του χρηστή στον υπολογιστή. Η προσαρμογή κατάστασης, ή εξατομίκευση υπηρεσιών και προϊόντων, αποτελείται από δύο βασικά συστατικά: κάποιον τύπο πληροφοριών πλαισίου, όπως πληροφορίες για το χρήστη, για το περιβάλλον του χρήστη, κλπ., και κάποιο μηχανισμό συλλογιστικής για αυτές τις πληροφορίες.

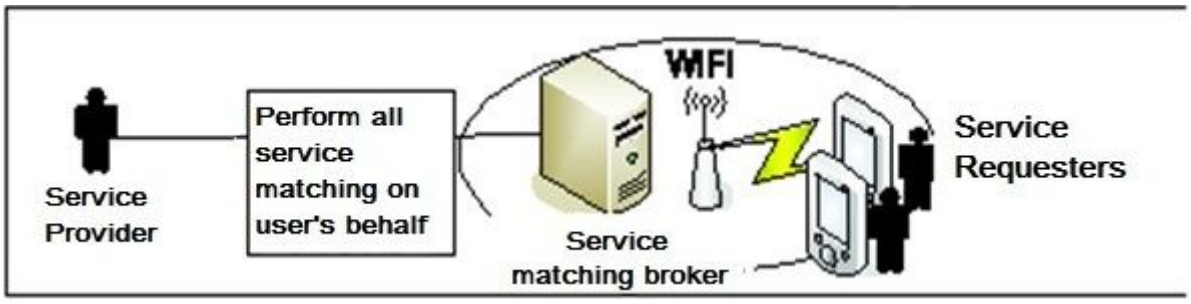
Δεδομένης της συνεχώς αυξανόμενης διαθεσιμότητας των κινητών συσκευών και των διαδικτυακών υπηρεσιών που έχουν στη διάθεσή τους οι χρήστες, απαιτούνται αρχιτεκτονικές διάχυτης ανακάλυψης υπηρεσιών, οι οποίες διαχειρίζονται αποτελεσματικά πρόσθετες προκλήσεις. Οι προκλήσεις αυτές περιλαμβάνουν την εύρεση των σχετικών υπηρεσιών γρήγορα, ενώ αντιμετωπίζουν το πρόβλημα των περιορισμένων υπολογιστικών πόρων αυτών των συσκευών, σε ένα δυναμικό/μεταβαλλόμενο πλαίσιο. Το κλειδί για τη βελτίωση της σχετικότητας των ανακαλυφθέντων υπηρεσιών, είναι η αξιοποίηση των τεχνολογιών του Σημασιολογικού Ιστού. Ωστόσο, οι μηχανές συμπερασμού που χρησιμοποιούνται για την ανακάλυψη σημασιολογικών υπηρεσιών καταναλώνουν πολλούς πόρους και ως εκ τούτου δεν είναι κατάλληλες για κινητά περιβάλλοντα. Για τον λόγο αυτό, απαιτείται η ανάπτυξη στρατηγικών που επιτρέπουν τον συμπερασμό σε κινητές συσκευές με περιορισμένους πόρους (mobile reasoning).

3.1 Ανάγκη για mobile reasoning

Η ανάγκη για mobile reasoning έχει προκύψει από την ανάπτυξη των κινητών υπηρεσιών του Σημασιολογικού Ιστού . Για την υποστήριξη συλλογιστικής σε μια κινητή συσκευή, η πιο εύκολη λύση θα ήταν να συνδεθεί με έναν ειδικό εξυπηρετητή συλλογιστικής στο Διαδίκτυο. Υπάρχουν, όμως, πολλά μέρη όπου μια κυψελοειδής σύνδεση στο Internet δεν είναι διαθέσιμη ή δυνατή. Οι κινητές εφαρμογές μπορεί να βασίζονται στην Bluetooth διεπαφή τους ή την ενσωματωμένη κάμερα για να λαμβάνουν σημασιολογικά σχολιασμένες πληροφορίες. Στις περιπτώσεις αυτές, η ανεξαρτησία από μια σύνδεση στο Internet είναι ένα μεγάλο πλεονέκτημα για τέτοιου είδους εφαρμογές κινητής συλλογιστικής.

Ένα άλλο πρόβλημα των κεντρικών διακομιστών συλλογιστικής του Διαδικτύου είναι ότι δεν έχουν καλή κλιμάκωση (scalability). Όταν πολλές κινητές συσκευές αποκτούν πρόσβαση στον ίδιο διακομιστή συλλογιστικής του Διαδικτύου προκαλείται πρόβλημα. Αν οι εργασίες συλλογιστικής μοιράζονται στις συσκευές που τις χρησιμοποιούν, η κλιμάκωση δεν αποτελεί πλέον ζήτημα. Τέλος, υπάρχει ένα ζήτημα προστασίας της ιδιωτικής ζωής. Τα δεδομένα της συλλογιστικής μπορεί να είναι εμπιστευτικά, όπως, για παράδειγμα, το προφίλ ενός χρήστη. Πολλοί άνθρωποι δεν θα αισθάνονται άνετα εάν τα ιδιωτικά δεδομένα τους αποστέλλονται για επεξεργασία σε ένα διακομιστή του Διαδικτύου και μπορεί να προτιμούν να μη χρησιμοποιούν καθόλου τις υπηρεσίες αυτές.

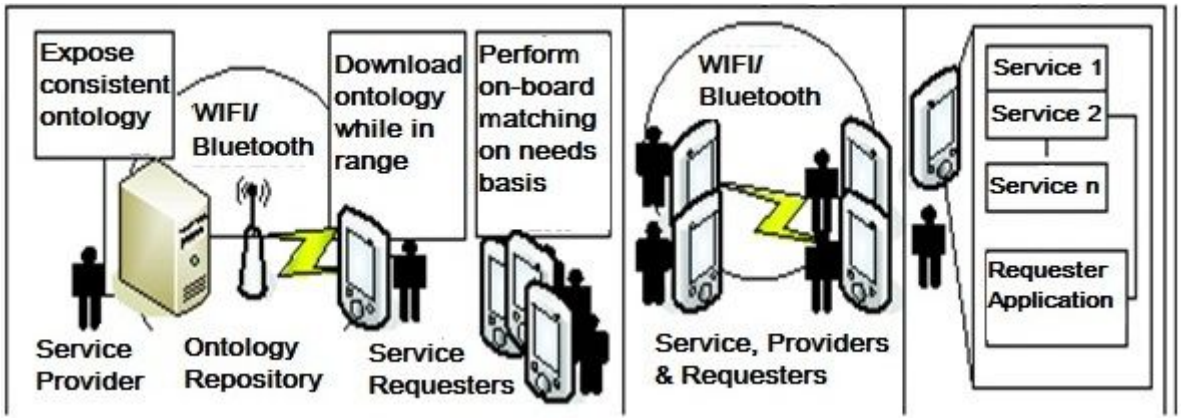
Για παράδειγμα, θα εξετάσουμε το ακόλουθο σενάριο κινητών εφαρμογών. Ο χρήστης της κινητής συσκευής έχει μόλις φτάσει στο αεροδρόμιο του Sydney και θέλει να ψάξει για τα τρόφιμα και άλλα προϊόντα. Το αεροδρόμιο του Sydney παρέχει τερματικά περίπτερα με οθόνη αφής, που επιτρέπουν στο χρήστη να αναζητήσει καταστήματα (και άλλες εγκαταστάσεις του αεροδρομίου) ανά κατηγορία. Εν συνεχεία, η τοποθεσία του καταστήματος και της εγκατάστασης εμφανίζεται σε ένα χάρτη, καθώς και η τοποθεσία του χρήστη (που είναι η σταθερή θέση του περιπτέρου). Αυτά τα περίπτερα δεν είναι πολύ βολικά, καθώς βρίσκονται μόνο σε σταθερές θέσεις, είναι περιορισμένα σε επιλογές αναζήτησης και στην πολυπλοκότητα των αιτήσεων των χρηστών, και δεν λαμβάνουν υπόψη το περιβάλλον του χρήστη. Επιπλέον, δεν έχουν δυνατότητα κλιμάκωσης, αφού τα περίπτερα μπορούν να χρησιμοποιηθούν μόνο από ένα χρήστη κάθε φορά.



Εικόνα 3.1: Κεντροποιημένη προσέγγιση ταιριάσματος υπηρεσιών

Εναλλακτικά, η αυξανόμενη αφθονία των κινητών συσκευών, όπως τα PDA και τα κινητά τηλέφωνα, καθώς και η αύξηση των υπολογιστικών και επικοινωνιακών δυνατοτήτων τους παρέχουν νέες ευκαιρίες για την ενσωματωμένη (on-board) ανακάλυψη υπηρεσιών (service discovery). Ας υποθέσουμε πως το περίπτερο πληροφοριών είναι ένας κατάλογος/αποθετήριο (repository) των υπηρεσιών που διατίθενται στο αεροδρόμιο, με το οποίο μπορούν να συνδεθούν οι χρήστες κινητών συσκευών, από το τηλέφωνο ή το PDA τους. Ο χρήστης μπορεί στη συνέχεια να αποκτήσει πρόσβαση, να αναζητήσει και να χρησιμοποιήσει αυτές τις πληροφορίες χρησιμοποιώντας τα τηλέφωνά τους με ευκολία. Υπάρχουν δύο τρόποι ταιριάσματος υπηρεσίας:

- κεντρικοποιημένο ταίριασμα υπηρεσίας που λαμβάνει χώρα σε ένα διακομιστή για λογαριασμό του χρήστη και
- μερικώς ή πλήρως αποκεντρωμένες προσεγγίσεις, όπου το ταίριασμα συμβαίνει στην ίδια, περιορισμένη σε πόρους, συσκευή.



Εικόνα 3.2: Αποκεντρωμένη προσέγγιση του ταιριάσματος υπηρεσιών

Σύμφωνα με μια κεντροποιημένη προσέγγιση (βλ. Εικόνα 3.1), το περίπτερο (ή κάποιο συνδεδεμένο μηχάνημα) είναι ένας εξυπηρετητής, που χειρίζεται όλες τις αιτήσεις ανακάλυψης υπηρεσιών για λογαριασμό του κινητού χρήστη. Ωστόσο, παρότι η αγορά ενός εξυπηρετητή είναι σχετικά φθηνή, υπάρχουν σημαντικά κόστη για αυτό το είδος της παροχής υπηρεσιών, συμπεριλαμβανομένης της κλιμάκωσης ώστε να διαχειρίζονται δυνητικά χιλιάδες αιτήματα, της παροχής ασύρματων δικτύων, του κόστους συντήρησης και θεμάτων ασφάλειας και ποιότητας υπηρεσιών. Επιπλέον, σε μια τέτοια προσέγγιση θα ίσχυαν και όλα τα προβλήματα που αναφέρθηκαν στην αρχή της ενότητας. Έτσι,

καταλήγουμε στην περίπτωση της αποκεντρωμένης προσέγγισης, όπου ο μηχανισμός συλλογιστικής εκτελείται στην περιορισμένη σε πόρους κινητή συσκευή.

3.2 Ιδιαιτερότητες

Το περιβάλλον των κινητών συσκευών θέτει πολλούς περιορισμούς για την ανάπτυξη μιας μηχανής συμπερασμού. Πόροι όπως η υπολογιστική ισχύ, η μνήμη και η ενέργεια είναι ιδιαίτερα περιορισμένοι. Για παράδειγμα, αν και τα κινητά τηλέφωνα προσφέρουν εκατοντάδες MB ή GB ενσωματωμένης και επεκτάσιμης μνήμης για την αποθήκευση εικόνων και ήχων, η μνήμη που προσφέρεται για εφαρμογές περιορίζεται σε μερικές εκατοντάδες MB. Αυτός είναι, συνήθως, ένας αυστηρός περιορισμός που έχει επιβληθεί από τον κατασκευαστή και δεν επιδέχεται παραμετροποίησης. Επιπροσθέτως, αυτή η μνήμη χρησιμοποιείται για την αποθήκευση και εκτέλεση σε πραγματικό χρόνο του κώδικα της εφαρμογής, των δεδομένων της εφαρμογής, όπως η οντολογία, και το λειτουργικό σύνολο του τρέχοντος υπολογισμού. Κατά συνέπεια, κάθε φορά που υπάρχει η δυνατότητα επιλογής μεταξύ κατανάλωσης μνήμης ή πρόσθετων υπολογισμών, επιλέγονται οι πρόσθετοι υπολογισμοί. Η υπολογιστική ισχύς μειώνεται, έτσι, κατά ένα συντελεστή 50 έως 100, έναντι ενός μέσου επιτραπέζιου υπολογιστή.

Η πραγματικότητα των κινητών περιβαλλόντων είναι ένας κόσμος, που χαρακτηρίζεται από μια διακοπτόμενη συνδεσιμότητα, όπου η εξάρτηση από απομακρυσμένη/κεντρική επεξεργασία (και συνεχή αλληλεπίδραση) δεν είναι πάντα εφικτή ή επιθυμητή, λαμβάνοντας υπόψη την ανάγκη για ταχεία επεξεργασία και δυναμικά μεταβαλλόμενο πλαίσιο. Η διάχυτη ανακάλυψη υπηρεσιών πρέπει αναγκαστικά να υποστηρίζεται από το τρέχον πλαίσιο για να ανταποκριθεί στα κριτήρια σχετικότητας σε συνεχώς μεταβαλλόμενες συνθήκες. Η επιβάρυνση (για να μην αναφέρουμε την αδυναμία/μη εφαρμοστικότητα) της επικοινωνίας από τη συνεχή αναμετάδοση των αλλαγών του πλαισίου και της κατάστασης του χρήστη/συσκευής σε έναν κεντρικό εξυπηρετητή, θα οδηγήσει σε αναπόφευκτες καθυστερήσεις. Έτσι, για να ανταποκριθεί στις προκλήσεις απόκρισης σε πραγματικό χρόνο ενός κινητού περιβάλλοντος, μια σημασιολογικά διάχυτη ανακάλυψη υπηρεσιών πρέπει να εκτελεί το ταίριασμα και τη συλλογιστική στην περιορισμένη σε πόρους συσκευή αυτή.

3.3 Απαιτήσεις σε εκφραστικότητα και αποδοτικότητα χρόνου/μνήμης

Η κινητή ανακάλυψη υπηρεσιών και το ταίριασμα (matchmaking) πρέπει να πληρούν δύο σημαντικές απαιτήσεις των χρηστών. Το ταίριασμα πρέπει να είναι ακριβές/χρήσιμο και γρήγορο. Η ακρίβεια μπορεί να επιτευχθεί με την υιοθέτηση γλωσσών οντολογιών του σημασιολογικού ιστού (OWL) και τη χρήση μηχανών συμπερασμού βασισμένων στη λογική για να συναγάγουν σχέσεις με τις απαιτήσεις του μοντέλου, και τη χρήση λογικών μηχανών συμπερασμού για να συναγάγουν νέες πληροφορίες από αυτές τις οντολογίες. Οι πιο πρόσφατες σημασιολογικές μηχανές συμπερασμού χρησιμοποιούν περιγραφική λογική (DL) η οποία παρέχει την πιο εκφραστική, αποφασίσιμη λογική OWL. Ωστόσο, αυτές οι λογικές μηχανές είναι αργές και καταναλώνουν πολλούς πόρους. Ως εκ τούτου οι τρέχουσες μηχανές συμπερασμού δεν μπορούν να μεταφερθούν σε κινητές συσκευές με περιορισμένους πόρους, όπως έξυπνα τηλέφωνα και PDA, στη σημερινή τους μορφή. Ως εναλλακτική των DL εργαλείων απόδειξης θεωρημάτων βασισμένων στο tableau, μια αντιστοίχιση ενός υποσυνόλου DL με λογικά προγράμματα (LP), κατάλληλη για αξιολόγηση με Prolog έχει προταθεί στο [36]. Αυτή η τομή του DL με το LP που ονομάζεται DLP καλύπτει πλήρως το RDF Schema [37] και ένα μικρό μέρος της OWL [38] (κυρίως το μεγαλύτερο κομμάτι της OWL Lite που έχει επεκταθεί με τη γενική ενσωμάτωση εννοιών).

Δεδομένου ότι οι χρήστες των κινητών συσκευών βρίσκονται συχνά εν κινήσει και σε μια ιδιαίτερα δυναμική κατάσταση, χρειάζονται συνήθως πληροφορίες γρήγορα. Μελέτες, έχουν ήδη αποφανθεί ότι οι χρήστες της κινητής τηλεφωνίας έχουν συνήθως ένα όριο ανοχής 5 έως 15 δευτερολέπτων περίπου, σε ό,τι αφορά στο χρόνο απόκρισης, πριν στρέψουν αλλού την προσοχή τους, ανάλογα με το περιβάλλον τους. Έτσι, οι αρχιτεκτονικές ανακάλυψης υπηρεσιών που λειτουργούν σε κινητά περιβάλλοντα, πρέπει όχι απλώς να ανακαλύπτουν τις σχετικές υπηρεσίες, αλλά και να είναι σε θέση να το πράξουν γρήγορα σε ένα ιδιαίτερα δυναμικό και εναλλασσόμενο πλαίσιο.

Τέλος, η υπολογιστική ισχύ περιορίζεται σε όση διατίθεται από τις συσκευές. Η υπάρχουσα μνήμη σε αυτές τις συσκευές είναι ανεπαρκής για την εκτέλεση κάποιων διαδικασιών συλλογιστικής, που απαιτούν αρκετό χρόνο και μνήμη για να ολοκληρωθούν.

3.4 Εφαρμογές

Με την εμφάνιση των έξυπνων τηλέφωνων/PDA, το κινητό περιβάλλον γίνεται όλο και περισσότερο πλούσιο σε πληροφορίες. Για παράδειγμα, πληροφορίες σε συσκευές μπορεί να περιλαμβάνουν δεδομένα από αισθητήρες, συνθήκες κυκλοφορίας, τις προτιμήσεις του χρήστη ή τις συνήθειες ή περιγραφές διαδικτυακών υπηρεσιών με δυνατότητα απομακρυσμένης κλήσης, που φιλοξενούνται σε αυτές τις συσκευές. Οι πληροφορίες αυτές μπορεί να είναι ιδιαίτερα χρήσιμες σε άλλους χρήστες στο περιβάλλον. Τέτοια σενάρια περιλαμβάνουν: φοιτητές που ανταλλάσουν δεδομένα για μια εκδρομή [39], καταστάσεις έκτακτης ανάγκης, ανταλλαγή πληροφοριών για την τροχαία κυκλοφορία κλπ. Οι εφαρμογές αυτές απαιτούν η ανακάλυψη υπηρεσιών να γίνεται στη συσκευή.

Ένα ακόμα παράδειγμα εφαρμογής του mobile reasoning είναι τα κινητά τηλέφωνα. Τα κινητά τηλέφωνα έχουν γίνει ένας πανταχού παρόν σύντροφος για πολλούς χρήστες. Υπάρχουν δυτικές χώρες με περισσότερα συμβόλαια κινητής τηλεφωνίας από ότι κατοίκους. Η κινητή βιομηχανία πιέζει τους χρήστες να υιοθετήσουν σύγχρονες υπηρεσίες 3G, δηλαδή να χρησιμοποιούν το κινητό τηλέφωνο ως πελάτη για υπηρεσίες μηνυμάτων και ειδήσεων. Η συσκευή θα πρέπει να είναι σε θέση να ξεδιαλύνει τα ανεπιθύμητα μηνύματα και να μειώσει τη διεισδυτικότητα του χρήστη σε μηνύματα ενδιαφέροντος. Αυτό απαιτεί από την κινητή συσκευή να αποφασίζει για το ταίριασμα χωρίς την υποστήριξη εξωτερικών μηχανών συμπερασμού.

Το mobile reasoning είναι χρήσιμο σε κάθε περίπτωση που μια κινητή συσκευή απαιτείται να έχει κάποιο είδος ευφυΐας, ώστε να είναι σε θέση να λαμβάνει αποφάσεις χωρίς ή με ελάχιστη ανθρώπινη παρέμβαση. Για παράδειγμα, μπορεί να χρησιμοποιηθεί για την ανάπτυξη ενός «έξυπνου» περιπτέρου υγείας, όπου η κατάσταση υγείας ενός ατόμου αξιολογείται μέσω της δυναμικής σύνδεσης με τους βιο-αισθητήρες του.

4. ΣΧΕΤΙΚΑ ΣΥΣΤΗΜΑΤΑ

4.1 Pocket KRHyper

Το Pocket KRHyper [42] είναι μια βιβλιοθήκη λογισμικού, υλοποιημένη σε Java 2 Mobile Edition (J2ME1), με σκοπό την παροχή υπηρεσιών αυτοματοποιημένης συλλογιστικής. Μπορεί να ενσωματωθεί σε εφαρμογές σε φορητές συσκευές όπως PDAs, smartphones, και κινητά τηλέφωνα, αλλά μπορεί επίσης να χρησιμοποιηθεί σε εφαρμογές Java 2 Standard Edition (J2SE2). Η μηχανή συλλογισμού βασίζεται στον hyper tableau λογισμό [40] και μπορεί να θεωρηθεί ως μια βελτιστοποιημένη έκδοση του συστήματος KRHyper [41], όσον αφορά στους πόρους. Το πρωτότυπο σύστημα KRHyper δεν έχει σχεδιαστεί για να λειτουργεί σε κινητές συσκευές. Το Pocket KRHyper αναπτύχθηκε για χρήση σε κινητές συσκευές με περιορισμένους πόρους, και είναι μια μηχανή συμπερασμού για κινητές συσκευές σε θέση να αντιμετωπίσει προβλήματα σε λογική πρώτης τάξης.

Το Pocket KRHyper είναι μια υλοποίηση του hyper tableau λογισμού. Σε αντίθεση με το KRHyper, το Pocket KRHyper στερείται ορισμένων χαρακτηριστικών, όπως η εξ' ορισμού άρνηση (default negation) και η ευρετηρίαση όρων (term indexing), αλλά δεν απαιτεί μεγάλη κατανάλωση πόρων, για αυτό και μπορεί να τρέξει σε κινητές συσκευές. Η διαδικασία του συλλογισμού είναι παρόμοια με εκείνη του αρχικού συστήματος KRHyper και περιγράφεται με περισσότερες λεπτομέρειες στο [41].

4.1.1 Hyper tableau λογισμός

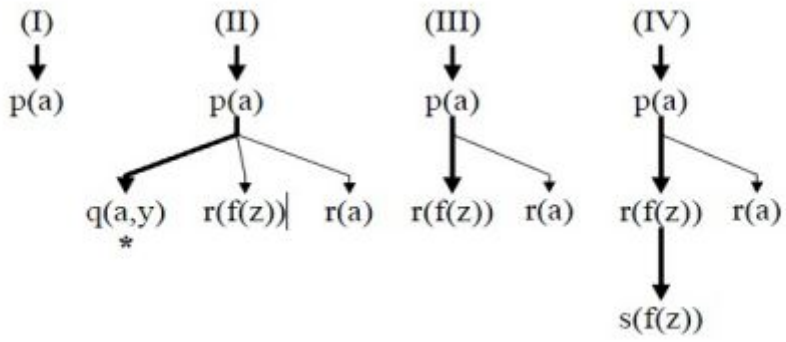
Ο hyper tableau λογισμός είναι μια διαδικασία λογικής πρώτης τάξης, προτασιακού tableau και περιγράφεται αναλυτικά στο [43,44]. Παρακάτω θα δώσουμε μόνο μια σύντομη επισκόπησή του, εστιάζοντας σε ορισμένα από τα χαρακτηριστικά του Pocket KRHyper. Το Pocket KRHyper μπορεί να αντιμετωπίσει προβλήματα της λογικής πρώτης τάξης σε προτασιακή μορφή. Οι προτάσεις θεωρούνται κανόνες, όπου η κεφαλή είναι μια διάζευξη θετικών λεκτικών και το σώμα μια σύζευξη αρνητικών λεκτικών. Κανόνες με κενό σώμα καλούνται γεγονότα. Κανόνες με άδεια κεφαλή αντιπροσωπεύουν περιορισμούς στα μοντέλα. Παρά το γεγονός ότι οι κεφαλές των προτάσεων μπορεί να περιέχουν διαζεύξεις, τα λεκτικά στην κεφαλή δεν μπορούν να μοιράζονται μεταβλητές που δεν αντιστοιχούν σε κάποιο λεκτικό του σώματος της πρότασης. Σε τέτοια περίπτωση, παρουσιάζεται εξαίρεση. Σε κάθε βήμα της hyper

tableau συλλογιστικής, παράγονται νέα στιγμιότυπα κεφαλών κανόνων, από κάτω προς τα επάνω, από την εισαγωγή προτάσεων και τα παραγόμενα γεγονότα. Αν μια hyper tableau παραγωγή τερματίσει χωρίς να καταλήξει σε άρνηση, τα παραγόμενα γεγονότα δημιουργούν μια αναπαράσταση ενός μοντέλου των εισαχθέντων προτάσεων.

Παράδειγμα

Το ακόλουθο παράδειγμα απεικονίζει τον τρόπο που το hyper tableau παράγει μοντέλα. Η Εικόνα 4.1 παρουσιάζει τρία ακόλουθα στάδια μιας παραγωγής από την εισαγωγή των παρακάτω 4 προτάσεων:

$$p(a) \leftarrow \quad (1)$$
$$q(x,y) \vee r(f(z)) \vee r(x) \leftarrow p(x) \quad (2)$$
$$\leftarrow q(x,x) \quad (3)$$
$$s(x) \leftarrow r(x) \quad (4)$$



Εικόνα 4.1: Στάδια παραγωγής hyper tableau λογισμού

Όπως προαναφέρθηκε, η κεφαλή ενός κανόνα μπορεί να είναι μια διάζευξη. Στο hyper tableau, οι διαζεύξεις αντιμετωπίζονται με την διερεύνηση των εναλλακτικών διακλαδώσεων με συστηματικό τρόπο. Αυτό εξηγεί τη δομή του δέντρου στην Εικόνα 4.1. Η δομή των δεδομένων που τηρείται από τη μέθοδο ονομάζεται hyper tableau. Η ενεργός διακλάδωση εμφανίζεται με έντονη γραμμή στο σχήμα. Μετά το βήμα (I), το μοναδικό γεγονός που εισάχθηκε (1), προστίθεται στην τρέχουσα διακλάδωση. Η ενεργός διακλάδωση αντιπροσωπεύει την τρέχουσα προσπάθεια του λογισμού για την κατασκευή ενός μοντέλου. Στο βήμα (I), αυτό το κομμάτι του μοντέλου έρχεται σε αντίθεση, για παράδειγμα, με την πρόταση (2): ένα μοντέλο που περιέχει το p(α) πρέπει επίσης να περιέχει όλα τα στιγμιότυπα του q(a,y) ή του r(f (z)) ή το r(α). Στο βήμα (II),

το κομμάτι "επιδιορθώνεται" με την παραγωγή συνεπαγόμενων λεκτικών και συνδέοντάς τα με το hyper tableau: το αντίστοιχο στιγμιότυπο της πρότασης (2) συνδέεται με το hyper tableau. Δεδομένου ότι έχει μια διαζευκτική κεφαλή, το tableau χωρίζεται σε τρεις διακλαδώσεις. Η πρώτη διακλάδωση επιθεωρείται και αποδεικνύεται αντιφατική με την πρόταση (3) και κατά συνέπεια κλείνει.

Ο υπολογισμός, τώρα, μεταβαίνει σε προηγούμενη κατάσταση (backtrack) και ερευνά τη δεύτερη διακλάδωση, όπως φαίνεται στο βήμα (III). Ο υπολογισμός προχωρά με την παραγωγή του s(f(z)) από την πρόταση (4). Δεδομένου ότι δεν μπορούν να εξαχθούν άλλα γεγονότα, ένα μοντέλο για ολόκληρη την πρόταση έχει βρεθεί, το οποίο αντιπροσωπεύεται από τα γεγονότα στην ενεργό διακλάδωση, {p(a), r(f(z)), s(f (z))}, όπως φαίνεται στο (IV). Το Pocket KRHyper σταματά μετά το πρώτο μοντέλο που θα βρεθεί.

```
public static void main(String[] args){
   int minTermWeight = 5; //initial search depth is a term weight of 5
   int maxTermWeight = 0; //no maximum term weight
   int timeout = 1000;    //stop after 1000 ms
   boolean model = false;
   KnowledgeBase kb = new KnowledgeBase();
   kb.addClause(LogicFactory.newClause("ufo(enterprise)."));
   kb.addClause(LogicFactory.newClause("false:-ufo(X)."));
   Reasoner krhyper = new KrHyper();
   krhyper.setKnowledgeBase(kb);
   try {
      model = krhyper.reason(minTermWeight, maxTermWeight, timeout);
      if (!model){
         // Refutation Found
      } else {
         // Model Found
         Vector model = krhyper.getModel();
      }
   } catch (ProofNotFoundException ex){
         //Timeout reached
   } catch (OutOfMemoryError err){
         //Out of Memory
   }
}
```

Εικόνα 4.2: Παράδειγμα κώδικα του Pocket KRHyper

4.1.2 Χρήση

Το Pocket KRHyper είναι μια βιβλιοθήκη Java σχεδιασμένη για την πλατφόρμα Java 2 Micro Edition (J2ME), υποστηρίζοντας Connected Limited Device Configuration (CLDC3) έκδοση ≥ 1.0 και Mobile Information Device Profile (MIDP4) έκδοση ≥ 1.0. Η βιβλιοθήκη μπορεί, όμως, επίσης να χρησιμοποιηθεί μέσα σε Standard ή Enterprise εκδόσεις Java εφαρμογών.

Χρήση της μηχανής συμπερασμού πρώτης Τάξης

Για να χρησιμοποιηθεί η μηχανή συμπερασμού πρώτης τάξης, θα πρέπει να δημιουργηθεί ένα στιγμιότυπο της βάσης γνώσης (KnowledgeBase), η οποία θα γεμίσει με κάποια προτασιακά (Clause) αντικείμενα και θα περαστεί στο στιγμιότυπο της μηχανής. Η Εικόνα 4.2 δείχνει ένα μικρό παραδειγματικό πρόγραμμα για τη χρήση του Pocket KRHyper. Αρχικά, δημιουργείται ένα στιγμιότυπο KnowledgeBase. Στη συνέχεια, χρησιμοποιώντας την ειδική στατική κλάση LogicFactory, δημιουργούνται μερικά στιγμιότυπα προτάσεων, δίνοντας μια συμβολοσειρά σε σύνταξη PROTEIN [45] ως παράμετρο. Μετά την εισαγωγή όλων των προτάσεων, για το συλλογιστικό πρόβλημα, στη βάση γνώσης, εισάγεται στη μηχανή συμπερασμού χρησιμοποιώντας τη συνάρτηση Reasoner.setKnowledgeBase (KnowledgeBase kb).

Για να ξεκινήσει η μηχανή συμπερασμού, χρησιμοποιείται η Reasoner.reason(int mintermweight, int maxtermweight, int timeout). Ο συλλογιστικός αλγόριθμος εκτελεί μια επαναληπτική αναζήτηση με εμβάθυνση για ένα μοντέλο, διευρύνοντας το δέντρο της απόδειξης σε κάθε επανάληψη, μέχρι ένα ορισμένο βάρος όρου. Οι παράμετροι του βάρους όρου ελέγχουν τα όρια με τα οποία ξεκινά και εγκαταλείπεται η αναζήτηση. Η παράμετρος timeout καθορίζει το μέγιστο επιτρεπόμενο χρονικό διάστημα για την εκτέλεση μιας αναζήτησης. Αν η maxtermweight ή η timeout έχει οριστεί στο 0, αγνοούνται. Η συλλογιστική διαδικασία μπορεί να τερματίσει με πολλούς διαφορετικούς τρόπους. Ιδανικά, έχει καταλήξει σε διάψευση (refutation) ή σε ένα μοντέλο. Ένα μοντέλο μπορεί να ανακτηθεί χρησιμοποιώντας τη μέθοδο Reasoner.getModel(). Υπάρχουν κάποιοι άλλοι λόγοι για τους οποίους η διαδικασία συλλογισμού μπορεί να σταματήσει. Μπορεί να διακοπεί από το χρήστη (Reasoner.interruptReasoner()), να έχει περάσει ένα συγκεκριμένο χρονικό διάστημα (time out) ή να ξεπεραστεί το μέγιστο όριο βάρους όρου. Σε αυτές τις περιπτώσεις, ο Reasoner παρουσιάζει εξαίρεση ProofNotFound. Εάν η εικονική μηχανή δεν έχει επαρκή μνήμη, ο Reasoner παρουσιάζει εξαίρεση OutOfMemoryError. Σε περίπτωση που ο προγραμματιστής επιλέξει να μην ορίσει ένα χρονικό όριο ή μέγιστο βάρος όρου, ο λογισμός δεν μπορεί να τερματίσει (π.χ. εάν το πρόβλημα έχει ένα άπειρο μοντέλο). Ο προγραμματιστής πρέπει να γνωρίζει αυτήν την παγίδα και πάντα να παρέχει ένα μέσο για τη χειροκίνητη διακοπή της συλλογιστικής διαδικασίας.

4.2 JIProlog

Το JIProlog (Java Internet Prolog) είναι ένας διαπεριβαλλοντικός διερμηνέας της Prolog, υλοποιημένος σε Java, ο οποίος ενσωματώνει την Prolog στην Java. Το JIProlog επιτρέπει την κλήση κατηγορημάτων Prolog από την Java χωρίς τη μεταχείριση εγγενή κώδικα (JNI) και επιτρέπει την κλήση μεθόδων Java από την Prolog με τον ίδιο τρόπο που καλούνται τα κατηγορήματα. Το JIProlog έχει σχεδιαστεί ώστε να λειτουργεί με οποιαδήποτε πλατφόρμα Java 1.1 ή νεότερης έκδοσης, συμπεριλαμβανομένων διακομιστών εφαρμογών και φυλλομετρητών που υποστηρίζουν Java, PDA συμβατά με την πλατφόρμα Java 1.1 ή την Personal Java 1.1 ή μεταγενέστερη, και κινητών συσκευών συμβατών με MIDP 1.0/2.0. Το JIProlog είναι εξοπλισμένο με ένα ισχυρό ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) που βοηθά στην επεξεργασία, φόρτωση (consult), εκτέλεση και δοκιμή προγραμμάτων Prolog και Java. Τέλος, το JIProlog παρέχει ένα σύνολο κλάσεων Java, που αποτελεί το JIProlog API, το οποίο δίνει τη δυνατότητα:

- να κληθούν Prolog κατηγορήματα από την Java
- να κληθούν μέθοδοι και κλάσεις Java από την Prolog
- να δημιουργηθούν κατηγορήματα Prolog σε Java, προκειμένου να επεκταθεί το προεπιλεγμένο σύνολο των ενσωματωμένων κατηγορημάτων
- να επεκταθεί το γραφικό περιβάλλον χρήστη ενός προγράμματος Prolog, υλοποιώντας τα παράθυρα και τους διαλόγους σε Java, ως ενσωματωμένα κατηγορήματα
- να ενοποιηθούν βάσεις δεδομένων σε Prolog μέσω μιας JDBC←→Prolog γέφυρας, αντιμετωπίζοντας πινάκες και ερωτήματα, ως Prolog κατηγορήματα.

4.2.1 Συμβατότητα

Το JIProlog είναι συμβατό με τους μεγαλύτερους διερμηνείς Prolog. Υποστηρίζει τις προδιαγραφές Edimburgh, τις περισσότερες από τις προδιαγραφές ISO και πολλά από τα πιο κοινά και ISO ενσωματωμένα κατηγορήματα. Το JIProlog είναι συμβατό με την πλατφόρμα Java 1.1 ή μεταγενέστερη, J2ME (Personal Java 1.1 ή μεταγενέστερη, MIDP 1.0, MIDP 2.0 - CLDC1.0/1.1) και έχει δοκιμαστεί επιτυχώς στις ακόλουθες πλατφόρμες:

- Windows/Linux: JRE 1.1.x, 1.2.x, 1.3.x, 1.4.x, 1.5.x, 1.6.x

- κινητές συσκευές MIDP 2.0-CLDC1.0: J2ME Wireless Toolkit 2.2, Nokia Series 40, 60 ($2^{η}$ και $3^{η}$) και 80, SonyEricsson P900, P910, Z1010, K800, Motorola Razr V3, Siemens CXT65, SK65.

- Windows CE, Pocket PC 2002/2003: Jeode JVM, CrEME JVM σε HP iPAQ, Pocket Loox

- Windows Mobile 2003/2005: Jeode JVM, CrEME JVM σε HP iPAQ, HTC, QTEK, i-mate

- EPOC OS σε psion-m5

- Apache Tomcat

- Internet Explorer 4.x, 5.x, 6.0, 7.0

- Netscape Navigator 4.5, 4.7, 6.0

- Mozilla 1.x

- Firefox 1.0, 2.0

JIProlog2ME

Το JIProlog2ME είναι το πακέτο για τη χρήση του JIProlog σε κινητές συσκευές που υποστηρίζουν Java. Οι ελάχιστες απαιτήσεις που επιτρέπουν τη χρήση του είναι οι παρακάτω:

- MIDP 2.0:

- CLDC 1.0

- 400 KB RAM (heap)

Εξαιτίας των περιορισμών που τίθενται από το MIDP, το JIProlog2ME δεν υποστηρίζει τα ακόλουθα:

- πρόσβαση σε αρχεία χρησιμοποιώντας τα πρωτόκολλα file:// ή jar://

- τα κατηγορήματα: compile/1, load/1, load_library/1

4.2.2 Java σε Prolog

Το JIProlog παρέχει διάφορες κλάσεις Java για τη διαχείριση του διερμηνέα Prolog με τον ίδιο τρόπο που καλούνται γενικές μέθοδοι Java. Η JIPEngine είναι η κύρια κλάση.

Παρέχει τις μεθόδους για την αρχικοποίηση του διερμηνέα Prolog, την υποβολή και τη διαχείριση ερωτημάτων, τη φόρτωση αρχείων και τον ορισμό διάφορων ιδιοτήτων του διερμηνέα. Επιπλέον, υπάρχουν αρκετές κλάσεις που εγκλείουν Prolog όρους όπως λίστες, άτομα, προτάσεις κ.λπ. Μερικά από αυτά είναι τα: JIPTerm, JIPAtom, JIPList, κ.λπ.

Τα ερωτήματα μπορούν να υποβληθούν σύγχρονα ή ασύγχρονα. Στην πρώτη περίπτωση, η μέθοδος που υποβάλλει το ερώτημα τερματίζει, όταν ο στόχος ικανοποιείται ή όταν αποτυγχάνει. Με άλλα λόγια, η μέθοδος περιμένει έως ότου ο διερμηνέας έχει ολοκληρώσει την εργασία του. Στην τελευταία περίπτωση, η μέθοδος που υποβάλλει το ερώτημα τερματίζει αμέσως μετά την υποβολή και το αποτέλεσμα του ερωτήματος κοινοποιείται ασύγχρονα σε listeners που έχουν καταχωρηθεί εκ των προτέρων. Μια σύγχρονη κλήση είναι απλούστερη και ανταποκρίνεται στις ανάγκες των περισσότερων περιπτώσεων. Μια ασύγχρονη κλήση, ακόμα κι αν είναι πιο περίπλοκη, επιτρέπει μια πιο ισχυρή διαχείριση του διερμηνέα.

```
// New instance of Prolog engine
JIPEngine jip = new JIPEngine();
JIPTerm query = null;

try
{// parse query
        JIPTermParser parser = jip.getTermParser();
        query = parser.parseTerm("write('hello world'), nl.");
}
catch(JIPSyntaxErrorException ex)
{// there is a syntax error in the query
        ex.printStackTrace();
        System.exit(0);
}

// open a synchronous query
JIPQuery jipQuery = jip.openSynchronousQuery(query);
JIPTerm solution;

// Loop while there is another solution
while (!jipQuery.hasMoreChoicePoints())
{
        solution = jipQuery.nextSolution();
        System.out.println(solution);
}
```

Εικόνα 4.3: Παράδειγμα σύγχρονης κλήσης του διερμηνέα του JIProlog

4.3 Υπάρχουσες μηχανές συμπερασμού στη Mysaifu JVM

Τα περιβάλλοντα κινητού υπολογισμού, από τη φύση τους, εμπεριέχουν τη χρήση συσκευών οι οποίες να μπορούν να χρησιμοποιηθούν από κινητούς χρήστες. Βάσει της υφιστάμενης κατάστασης του συγκεκριμένου τεχνολογικού πεδίου, οι κινητές συσκευές υπόκεινται σε διάφορους περιορισμούς (μέγεθος μνήμης, ταχύτητα επεξεργαστή, ενεργειακή απόδοση), πράγμα που αποτελεί τροχοπέδη στη χρήση λογισμικού με αυξημένες υπολογιστικές ικανότητες. Στην προσπάθεια ανεύρεσης κατάλληλων μηχανών συμπερασμού (reasoners) που μπορούν να λειτουργήσουν σε περιβάλλοντα κινητού υπολογισμού, στα πλαίσια της παρούσας εργασίας δοκιμάστηκε ένα πλήθος εργαλείων συμπερασμού. Σε αυτή την ενότητα, δίνεται μία σύντομη περιγραφή των δυνατοτήτων του κάθε εργαλείου και παρουσιάζονται τα προβλήματα που αντιμετωπίστηκαν.

Η κινητή συσκευή που χρησιμοποιήθηκε στις δοκιμές των παρακάτω εργαλείων είναι ένα Dell Axim X51V PDA, με επεξεργαστή Intel PXA270 στα 624 MHz και μνήμη RAM 64MB. Το λειτουργικό σύστημα το οποίο χρησιμοποιήθηκε στα πλαίσια της εργασίας είναι τα Windows Mobile 5 και η εκτέλεση των εργαλείων έγινε μέσω της Mysaifu Java J2SE Virtual Machine [46].

4.3.1 RacerPro

Το RacerPro [47] είναι μία βελτιστοποιημένη μηχανή συμπερασμού (reasoner) βασισμένη στις περιγραφικές λογικές, που υποστηρίζει εκφραστικότητα SHIQ(D) και η υλοποίησή της είναι βασισμένη σε tableaux αλγορίθμους. Για συνεχή πεδία (concrete domains), υποστηρίζει ακέραιους και πραγματικούς αριθμούς, καθώς και διάφορες πολυωνυμικές εξισώσεις πάνω σε αυτά, και συμβολοσειρές με ελέγχους ισότητας. Πέρα από τα βασικά στοιχεία συμπερασμού, προσφέρει απάντηση ερωτημάτων στο Abox (Abox querying) βάσει των βελτιστοποιήσεων της nRQL και είναι γραμμένος στην προγραμματιστική γλώσσα Common Lisp.

Η χρήση του RacerPro γίνεται εκτελώντας ένα αρχείο τύπου .exe, εκκινώντας ένα πρόγραμμα εξυπηρετητή (server), το οποίο «ακούει» την θύρα 8080 (προεπιλογή) του υπολογιστή στον οποίο βρίσκεται. Χρησιμοποιώντας τις βιβλιοθήκες της εφαρμογής Protégé [48], για σύνταξη οντολογιών, δημιουργούμε κώδικα που ανοίγει μια σύνδεση με τον εξυπηρετητή του RacerPro . Με αυτόν τον τρόπο, επιτυγχάνεται ο συμπερασμός πάνω σε κάποια οντολογία. Κάτι τέτοιο δεν είναι εφικτό στο σύστημα αυτής της

εργασίας, μιας και τα Windows Mobile 5 δεν υποστηρίζουν την εκτέλεση αρχείων τύπου exe. Έτσι, το RacerPro απορρίπτεται ως επιλογή για mobile reasoning. Σε κάθε περίπτωση, οι αυξημένες υπολογιστικές απαιτήσεις εκτέλεσης της συγκεκριμένης μηχανής συμπερασμού καθιστούν τη χρήση του εργαλείου RacerPro ακατάλληλη σε μηχανές περιορισμένων δυνατοτήτων.

4.3.2 Bossam

Το Bossam [49] είναι μια μηχανή συλλογιστικής για τον σημασιολογικό ιστό. Επί της ουσίας, είναι μια μηχανή κανόνων βασιζόμενη στον αλγόριθμο RETE [50], που υποστηρίζει συμπερασμό σε OWL οντολογίες, SWRL οντολογίες και κανόνες RuleML. Επιπλέον, το Bossam εμπεριέχει διάφορα χαρακτηριστικά εκφραστικότητας όπως: 1) αναφορές URI ως σύμβολα, 2) σύνταξη λογικής δεύτερης τάξης, 3) διάζευξη στην υπόθεση και σύζευξη στο συμπέρασμα, 4) επισύναψη μεθόδων σε Java βάσει URI, 5) υποστήριξη της κλασσικής άρνησης και της άρνησης ως αποτυχία (negation-as-failure). Το μέγεθος εκτέλεσης του Bossam είναι περίπου 750Kb. Μπορεί να τρέξει σε πλατφόρμες J2ME CDC/PP καθώς και σε πλατφόρμες J2SE που χρησιμοποιούν JDK 1.3 ή νεότερη έκδοση.

Προς το παρόν, το Bossam απαιτεί τουλάχιστον 128 MB μνήμης RAM για την εκτέλεση του συμπερασμού σε OWL οντολογίες. Το PDA που χρησιμοποιήθηκε σε αυτή την εργασία διαθέτει μόνο 64 MB μνήμης, καθιστώντας το Bossam μη δυνατή επιλογή.

4.3.3 Pellet

Ο Pellet [51] είναι μία μηχανή συμπερασμού ανοιχτού λογισμικού, γραμμένη σε Java και υποστηρίζει εκφραστικότητα SROIQ με απλούς τύπους δεδομένων (δηλαδή OWL 1.1). Εφαρμόζει μια συλλογιστική διαδικασία βασισμένη σε tableau για TBoxes (ιεραρχία, ικανοποιησιμότητα και ταξινόμηση) και ABoxes (ανάκτηση, απάντηση συζευγμένων ερωτημάτων). Ο Pellet υιοθετεί πολλές από τις βελτιστοποιήσεις για standard DL reasoning , όπως και μερικοί από τους πιο σύγχρονους DL reasoners. Υποστηρίζει άμεσα, μέσω της διεπαφής του, έλεγχο συνεπαγωγής και βελτιστοποιημένη απάντηση ερωτημάτων στο Abox.

Η νεότερη έκδοση του Pellet είναι η 2.0.1, η οποία απαιτεί java 1.5 ή ανώτερη. Επειδή όμως, η Mysaifu JVM υποστηρίζει Java 1.4, δοκιμάστηκε η παλιότερη, συμβατή έκδοση του Pellet 1.5.1. Παρόλα αυτά, ο συμπερασμός αποδείχθηκε και πάλι αδύνατος, μιας

και το απλό πρόγραμμα που δημιουργήθηκε με τις βιβλιοθήκες του Pellet, δεν κατάφερε να ξεπεράσει το σημείο φόρτωσης της οντολογίας με βάση το URI της. Αναλυτικότερα, η συνάρτηση loadOntology() της κλάσης OWLOntologyManager, αποτυγχάνει επιστρέφοντας OWLOntologyCreationException. Η αιτία για αυτό το exception, εμφανίζεται να είναι κάποιο λάθος στη σύνδεση socket που πρέπει να δημιουργηθεί για την φόρτωση της οντολογίας.

4.3.4 HermiT

Ο HermiT [52] είναι μία ελεύθερα διαθέσιμη μηχανή απόδειξης θεωρημάτων (theorem prover) για περιγραφικές λογικές. Επί του παρόντος διαχειρίζεται πλήρως την DL SHIQ και η νέα έκδοσή του είναι συμβατή με OWL 2. Ο κύριος συμπερασμός που υποστηρίζεται είναι ο υπολογισμός της ιεραρχίας της υπαγωγής. Ο HermiT υλοποιεί έναν hyper tableau αλγόριθμο συλλογιστικής. Το βασικό στοιχείο αυτού του αλγορίθμου είναι ότι είναι πολύ λιγότερο μη-ντετερμινιστικός από τους υπάρχοντες tableau αλγορίθμους.

Ο HermiT, όπως και ο Pellet, χρησιμοποιεί το Manchester owlapi για το χειρισμό των οντολογιών. Επομένως, όπως ήταν αναμενόμενο, παρουσίασε το ίδιο πρόβλημα με τον Pellet, που αναφέρθηκε παραπάνω.

4.4 Ειδικό θέμα: Μετατροπή οντολογίας σε Prolog

Σε αυτήν την ενότητα παρουσιάζουμε τα εργαλεία dlpconvert [53] και Thea [54], που μετατρέπουν μια οντολογία σε προτάσεις της γλώσσας λογικού προγραμματισμού Prolog. Τέτοια εργαλεία είναι ιδιαιτέρως χρήσιμα για να καταστεί δυνατός ο συμπερασμός σε κινητές συσκευές, μιας και η γλώσσα οντολογιών (OWL) έχει πολύ υψηλή υπολογιστική πολυπλοκότητα για να χρησιμοποιηθεί σε συσκευές με περιορισμένους πόρους.

4.4.1 Dlpconvert

Το dlpconvert [53] είναι ένα εργαλείο μετατροπής σύνταξης που διευκολύνει τη χρήση του DLP. Το DLP είναι η τομή – από διαισθητική άποψη - της OWL DL και του (Horn) λογικού προγραμματισμού. Ως εκ τούτου, επιβάλλει ορισμένους περιορισμούς στην OWL DL προκειμένου να διασφαλίσει ότι όλα τα αξιώματα που ορίστηκαν, μπορούν να

μετατραπούν αποδοτικά σε προτάσεις Horn, δηλαδή κανόνες υπό την έννοια του παραδοσιακού λογικού προγραμματισμού. Επιτρέπει τη μετατροπή τμημάτων DLP κωδικοποιημένων σε OWL, σε (Edinburgh) σύνταξη λογικού προγραμματισμού, όπως χρησιμοποιείται από τα τυποποιημένα συστήματα Prolog.

Το dlpconvert βασίζεται στους αλγόριθμους που εφαρμόζονται στο KAON23 και περιγράφονται στο [55] και αναλυτικότερα στο [56], για τον περιορισμό και τη μετατροπή των περιγραφικών λογικών στη γλώσσα Datalog (υποσύνολο της Prolog). Διαβάζει μία OWL οντολογία, τη μετατρέπει σε διαζευκτική Datalog και τελικά τη σειριοποιεί σε ένα λογικό πρόγραμμα, το οποίο μπορεί να χρησιμοποιηθεί για την ευκολότερη ανάγνωση και, επομένως, κατανόηση από ανθρώπους με το κατάλληλο υπόβαθρο ή ως είσοδος για διερμηνείς Prolog. Το dlpconvert είναι ένα εργαλείο γραμμής εντολών, υλοποιημένο σε Java, με διάφορες επιλογές σειριοποίησης. Μπορεί να χρησιμοποιηθεί για να μετατρέψει, απευθείας, ένα OWL DL αρχείο σε αρχείο προγράμματος Prolog, που μπορεί να χρησιμοποιηθεί από ένα διερμηνέα Prolog ως έχει.

Πίνακας 4.1: Mapping του dlpconvert

OWL	Prolog code generated
Class C is equivalent to Class D	C(X) :- D(X). D(X) :- C(X).
subclassOf(C,D)	D(X) :- C(X).
Restriction(property,Value)	property(X,V)
Restriction(property,allValuesFrom(D))	D(Y) :- property(X,Y).
Restriction(property, hasValue v)	not supported yet
Restriction(property,someValuesFrom(D))	not in the DLP fragment
C (class URI)	classURI(X)
S is super property of P	S(X,Y) :- P(X,Y)
C in the domain of P	C (X) :- p(X,Y)
C in the range of P	C (Y) :- p(X,Y)
P is functional property	kaon2:equal (X,Y) :- p(Z,X),p(Z,Y).
P is inverse functional property	kaon2:equal (X,Y) :- p(X,Z),p(Y,Z).
P is a transitive property	p(X,Z) :- p(X,Y), p(Y,Z).
P is a symmetric property	p(X,Y) :- p(Y,X)
P is the inverse of Q	p(X,Y) :- q(Y,X). q(X,Y) :- p(Y,X)

Παράδειγμα

Παρακάτω, δίνουμε ένα παράδειγμα για να παρουσιάσουμε τη λειτουργία του dlpconvert:

All humans are mortal.
Socrates is a human.
Therefore Socrates is mortal.

Αυτός ο συλλογισμός, μαζί με ένα ακόμα γεγονός που αφορά στο Σωκράτη - ότι ο συντάκτης της Πολιτείας είναι στην πραγματικότητα ο Πλάτων και όχι ο Σωκράτης - μορφοποιείται στο ακόλουθο OWL αρχείο:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE Ontology [
<!ENTITY ex
"http://logic.uni-karlsruhe.de/dlpconvert/example1#">]>
<owlx:Ontology owlx:name="&ex;"
xmlns:owlx="http://www.w3.org/2003/05/owl-xml#">
<owlx:Class owlx:name="#human" owlx:complete="false">
<owlx:Class owlx:name="#mortal"/>
</owlx:Class>
<owlx:Individual owlx:name="#Socrates">
<owlx:type owlx:name="#human" />
</owlx:Individual>
<owlx:ObjectProperty owlx:name="#isauthor">
<owlx:domain owlx:class="#human"/>
<owlx:range owlx:class="#book"/>
</owlx:ObjectProperty>
<owlx:Individual owlx:name="#Plato">
<owlx:ObjectPropertyValue owlx:property="#isauthor">
<owlx:Individual owlx:name="#Politeia" />
</owlx:ObjectPropertyValue>
</owlx:Individual>
</owlx:Ontology>
```

Μεταποιώντας αυτό το αρχείο μέσω του dlpconvert το αποτέλεσμα είναι το ακόλουθο.

```
mortal(X) :- human(X).
book(Y) :- isauthor(X, Y).
human(X) :- isauthor(X, Y_0).
isauthor(plato, politeia).
human(socrates).
```

Και οι δύο αναπαραστάσεις έχουν το ίδιο νόημα, πράγμα που αποτελεί την ουσία της συντακτικής μετάφρασης. Η δεύτερη είναι αναμφίβολα μικρότερη και, για πολλούς αναγνώστες, η σύνταξη είναι πολύ πιο εύκολα κατανοητή, και για κάθε άτομο με κάποιο υπόβαθρο στον λογικό προγραμματισμό, το νόημα της δεύτερης αναπαράστασης είναι άμεσα εμφανές. Το αποτέλεσμα αυτό μπορεί να δοθεί απευθείας σε μια μηχανή Prolog, όπως η XSB-Prolog. Με αυτό τον τρόπο, το σύστημα XSB καταλαβαίνει τη σημασιολογία του αρχικού OWL αρχείου και μπορούμε να κάνουμε ερωτήσεις για τη δεδομένη οντολογία, όπως στο παρακάτω παράδειγμα.

?- mortal(socrates).

yes

?- isauthor(X, politeia).

X = plato

yes

?- human(plato).

yes

Το σύστημα δίνει τη σωστή απάντηση: Ο Σωκράτης είναι θνητός. Ρωτώντας την XSB για τον συγγραφέα της Πολιτείας, θα απαντήσει σωστά. Γνωρίζει, επίσης, ότι ο Πλάτων είναι άνθρωπος, διότι το πεδίο ορισμού της σχέσης isauthor είναι το human (να σημειωθεί ότι ποτέ δεν αναφέραμε ρητά, ότι ο Πλάτων είναι άνθρωπος).

4.4.2 Thea

Το Thea [54] είναι μια βιβλιοθήκη Prolog για την παραγωγή και το χειρισμό OWL περιεχομένου. Η έκδοση 0.5.5 αποτελείται από:

- συντακτικό αναλυτή OWL,
- γεννήτρια OWL,
- μετατροπέα από SQL σε OWL και
- μηχανή συμπερασμού OWL.

Ο συντακτικός αναλυτής χρησιμοποιεί τη βιβλιοθήκη Σημασιολογικού Ιστού της SWI-Prolog για την ανάλυση RDF/XML σειριοποιήσεων εγγράφων OWL σε RDF τριάδες και στη συνέχεια, κατασκευάζει μια αναπαράσταση της OWL οντολογίας, όπως αυτή ορίζεται στο Αφηρημένο Συντακτικό και Σημασιολογία της Οντολογικής Γλώσσας του Ιστού OWL (OWL Web Ontology Language Abstract Syntax and Semantics) των προδιαγραφών της OWL. Το αφηρημένο συντακτικό της OWL οντολογίας, υλοποιείται ως όροι Prolog.

Η γεννήτρια OWL χρησιμοποιείται για τα σχήματα αφηρημένης σύνταξης OWL από Prolog όρους σε RDF τριάδες και αποθηκεύει το μοντέλο RDF που προκύπτει σε αρχείο RDF/XML. Η γεννήτρια χρησιμοποιεί, επίσης, τη βιβλιοθήκη Σημασιολογικού Ιστού της SWI-Prolog για την αποθήκευση μοντέλων RDF σε αρχεία RDF/XML.

Ο μετατροπέας από SQL σε OWL, χρησιμοποιείται για να παράγει γεγονότα OWL από εγγραφές σε μια σχεσιακή βάση δεδομένων. Χρησιμοποιεί το πακέτο ODBC της SWI-Prolog για πρόσβαση στο RDBMS. Η μετατροπή αυτή καθοδηγείται από μια αντιστοίχιση ανάμεσα σε σχεσιακές οντότητες (πίνακες και στήλες) και έννοιες OWL (κλάσεις και ιδιότητες). Η αντιστοίχηση ορίζεται, με δηλωτική μορφή, μέσω όρων Prolog.

Η μηχανή συμπερασμού OWL περιλαμβάνει δύο υποενότητες:

- ένα μετατροπέα αφηρημένων όρων OWL σε Prolog, που βασίζεται στο DLP και
- ένα wrapper Prolog για τη διεπαφή DIG, έτσι ώστε μια οντολογία του Thea να μπορεί να επικοινωνήσει με μια μηχανή συμπερασμού που υποστηρίζει το DIG.

OWL σε Prolog

Το κατηγόρημα owl_as2prolog (+ OwlAsTerm, + Options), μετατρέπει τον όρο Prolog του αφηρημένου συντακτικού της OWL (όπως έχει αναλυθεί από το συντακτικό αναλυτή της OWL) σε λογικό κώδικα Prolog. Ο κώδικας Prolog γράφεται στο τρέχον ρεύμα εξόδου. Οι επιλογές είναι γενικές επιλογές για την τροποποίηση της συμπεριφοράς της παραγωγής του κώδικα. Προς το παρόν υποστηρίζεται μόνο η επιλογή no_base(χώρος_ονομάτων), της οποίας η λειτουργία αποσαφηνίζεται παρακάτω.

Πίνακας 4.2: Mapping του Thea

OWL AS axiom and fact	Prolog code generated
Class C complete declaration with single description D	C and D equivalent → subclassOf(C,D) AND subclassOf(D,C).
Class C complete declaration with multiple descriptions DL	subclassOf(C, Map(intersectionOf(DL))).
Class C partial declaration with multiple descriptions DL	subclassOf(C,D) for each D in DL.
subclassOf(C,D)	Map(D)(X) :- Map(C)(X).
intersectionOf(DL) (only if intersection in head or body of a rule).	Map(D1),Map(D2)…Map(Dn)
unionOf(DL) (only as body of a rule or as facts)	Map(D1) ; Map (D2) ; . . . ; Map(Dn)
oneOf(IL) (only in body of rules)	member(X,IL)
Restriction(property,Value)	property(X,V)
Restriction(property,allValuesFrom(D)) (head)	Map(D)(Y) :- property(X,Y).
Restriction(property,allValuesFrom(D)) (fact)	Map(D)(X) :- property(ID,X)
Restriction(property,someValuesFrom(D)) (body)	Map(D)(Y) , property(X,Y).
C (class URI)	classURI(X)
S is super property of P	S(X,Y) :- P(X,Y)
C in the domain of P	Map(C)(X) :- p(X,Y)
C in the range of P	Map(C)(Y) :- p(X,Y)
P is functional property	sameIndividuals(X,Y) :- p(Z,X),p(Z,Y).
P is inverse functional property	sameIndividuals(X,Y) :- p(X,Z),p(Y,Z).
P is a transitive property	p(X,Z) :- p(X,Y), p(Y,Z).
P is a symmetric property	p(X,Y) :- p(Y,X)
P is the inverse of Q	p(X,Y) :- q(Y,X). q(X,Y) :- p(Y,X)
individual(IID,_,DescriptionList,ValueList)	Map(D)(IID) for each D in DescriptionList p(IID,V) for each V in ValueList

Παράδειγμα

Ας υποθέσουμε ότι έχουμε αναλύσει συντακτικά την οντολογία «Wine». Οι ορισμοί των κλάσεων «WhiteWine» και «WhiteTableWine» βρίσκονται (μεταξύ άλλων) στους Prolog όρους:

class('http://www.w3.org/2002/03owlt/miscellaneous/consistent001#WhiteWine',
false,
complete,
[ ],
[intersectionOf(['http://www.w3.org/2002/03owlt/miscellaneous/consistent001#Wine',
restriction('http://www.w3.org/2002/03owlt/miscellaneous/consistent001#hasColor',
value('http://www.w3.org/2002/03owlt/miscellaneous/consistent001#White'))])]).
class('http://www.w3.org/2002/03owlt/miscellaneous/consistent001#WhiteTableWine'
,false,
complete,
[ ], [intersectionOf(['http://www.w3.org/2002/03owlt/miscellaneous/consistent001#TableWine',
restriction('http://www.w3.org/2002/03owlt/miscellaneous/consistent001#hasColor',
value('http://www.w3.org/2002/03owlt/miscellaneous/consistent001#White'))])]).

Για τη δημιουργία κώδικα σε Prolog, καλούμε το ακόλουθο κατηγόρημα:

:- C='http://www.w3.org/2002/03owlt/miscellaneous/consistent001#WhiteWine',

class(C,A2,A3,A4,A5), owl_as2prolog(class(C,A2,A3,A4,A5),[no_base('wine')]).

Το αποτέλεσμα της μετατροπής είναι:

'Wine'(X):-

'WhiteWine'(X).

hasColor(X,'wine:White'):-

'WhiteWine'(X).

'WhiteWine'(X):-

'Wine'(X),hasColor(X,'wine:White').

Η επιλογή no_base υπαγορεύει στη γεννήτρια του κώδικα Prolog να μη χρησιμοποιήσει το κομμάτι του χώρου ονομάτων των παραγόμενων κατηγορημάτων. Τα εισαγωγικά παράγονται αυτόματα όταν χρειάζεται (χρησιμοποιώντας το κατηγόρημα writeq της SWI). Καλώντας το ίδιο κατηγόρημα χωρίς την επιλογή no_base το αποτέλεσμα θα ήταν το εξής:

wine_Wine(X):-

wine_WhiteWine(X).

wine_hasColor(X,'wine:White'):-

wine_WhiteWine(X).

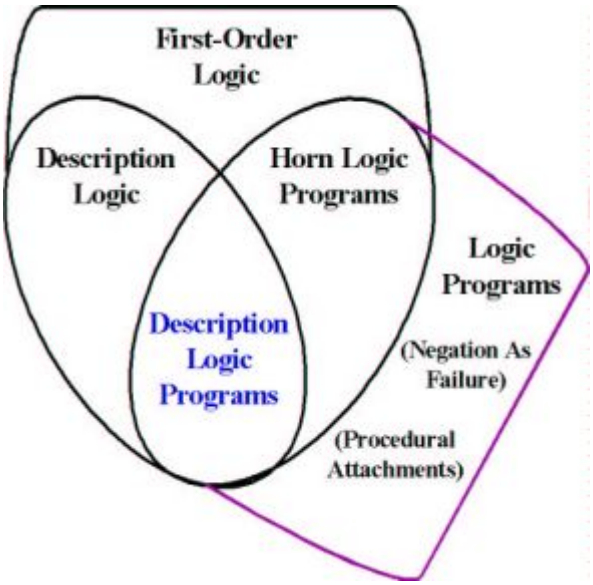
wine_WhiteWine(X):-

wine_Wine(X),wine_hasColor(X,'wine:White').

Δηλαδή, όλα τα παραγόμενα κατηγορήματα έχουν ως πρόθεμα το χώρο ονομάτων συν το « _ ».

4.5 Συμπεράσματα

Δεδομένου ότι η κινητή πρόσβαση σε εξωτερικές πληροφορίες ή υπηρεσίες είναι μια από τις κύριες επωφελείς εφαρμογές στον κινητό υπολογισμό, η μηχανικά αναγνώσιμη περιγραφή περιεχομένου, οι μέθοδοι πρόσβασης και η λειτουργικότητα είναι σημαντικές. Οι οντολογίες, ως μέρος του Σημασιολογικού Ιστού, προτείνονται και χρησιμοποιούνται ευρέως σε στατικά περιβάλλοντα για αυτόν τον σκοπό. Η προσθήκη οντολογιών σε κινητά περιβάλλοντα απαιτεί αποδοτική αποθήκευση και επεξεργασία των οντολογιών. Όπως έχουμε προαναφέρει, όμως, οι κινητές συσκευές διαθέτουν περιορισμένους πόρους (ταχύτητα επεξεργαστή, μνήμη). Για αυτό το λόγο χρειάζεται να περιορίσουμε την εκφραστικότητα της γλώσσας οντολογιών και να τη μετατρέψουμε σε μια αναπαράσταση με χαμηλότερη πολυπλοκότητα και κατανάλωση πόρων. Δυο τέτοιες τεχνικές είναι η μετατροπή μιας οντολογίας σε προτάσεις λογικής πρώτης τάξης (FOL) ή σε πρόγραμμα περιγραφικής λογικής (DLP) (στην περίπτωσή μας σε κώδικα Prolog).



Εικόνα 4.4: Εκφραστική επικάλυψη της περιγραφικής λογικής με τα λογικά προγράμματα

Μια τέτοια μετατροπή, αν και επιτρέπει τη διαδικασία εξαγωγής συμπερασμάτων στις ίδιες τις κινητές συσκευές, μειώνει την εκφραστικότητα της γλώσσας επιφέροντας την απώλεια δεδομένων. Συγκεκριμένα, η λογική πρώτης τάξης δεν υποστηρίζει την ισότητα (π.χ. ισότητα κλάσεων), απαιτώντας τη δημιουργία ενός προκαθορισμένου κατηγορήματος και τη σύλληψη των αξιωμάτων της για την προσομοίωσή της. Ως αποτέλεσμα, κάποιοι περιορισμοί και χαρακτηριστικά ιδιοτήτων, που χρησιμοποιούν

αυτό το κατηγόρημα για τη μετατροπή τους σε FOL (ή Prolog), καταναλώνουν περισσότερους πόρους κατά τη συλλογιστική διαδικασία. Οι περιορισμοί και τα χαρακτηριστικά αυτά είναι τα maximum cardinality, functional property και inverse functional property.

Στην Prolog η ισότητα δεν έχει την ισχύ της λογικής ισότητας. Το σύμβολο ισότητας (=) της υπαγορεύει να εκκινήσει μια προσπάθεια για να ταιριάξει τις δύο πλευρές μιας έκφρασης και να τις καταστήσει ίσες. Έτσι, αν τα X και Y είναι ελεύθερες μεταβλητές, τότε οι παρακάτω «συγκρίσεις» παράγουν, yes:

?- X = 5.

?- X = Y.

?- X + Y = a + b.

Όπως είναι αναμενόμενο, λόγω της μείωσης της εκφραστικότητας κάποιοι περιορισμοί δεν μπορούν να εκφραστούν καθόλου. Τέτοιοι περιορισμοί είναι τα minimum cardinality (απαιτεί skolemization) και cardinality, ενώ ο περιορισμός someValuesFrom απαιτεί τη δημιουργία ενός βοηθητικού κατηγορήματος για τη μετατροπή του. Ο πίνακας με τα αποτελέσματα της μετατροπής των βασικών στοιχείων της OWL σε FOL δίνεται στο κεφάλαιο 5.

Ένα ακόμα μειονέκτημα της μετατροπής μιας οντολογίας σε Prolog είναι ότι δεν υποστηρίζει συλλογιστική στο Tbox. Για παράδειγμα, έστω ότι έχουμε τις παρακάτω προτάσεις:

Vegetarian(X) :- Cow(X).

Animal(X) :- Vegetarian(X).

Η Prolog δε μπορεί να εξάγει το συμπέρασμα Animal(X) :- Cow(X).

Επιπλέον, η Prolog δε μπορεί να αντιμετωπίσει αναδρομικούς τύπους. Η ισότητα κλάσεων και ιδιοτήτων, καθώς και τα χαρακτηριστικά inverseOf, symmetric και transitive των ιδιοτήτων, όταν μετατρέπονται σε προτάσεις Prolog δημιουργούν ατέρμονους βρόχους. Έτσι, τα στοιχεία αυτά αφαιρούνται από την αρχική οντολογία, επιφέροντας τελικά ακόμα μεγαλύτερη απώλεια δεδομένων.

Τέλος, η Prolog, σε αντίθεση με τη λογική πρώτης τάξης και τις περιγραφικές λογικές, ακολουθεί την υπόθεση κλειστού κόσμου (closed world assumption - CWA). Σύμφωνα με αυτή όλη η απαραίτητη πληροφορία για την επίλυση του προβλήματος περιέχεται στο πρόγραμμα και οτιδήποτε δεν περιέχεται σε αυτό θεωρείται λάθος (οποιαδήποτε σχέση δεν μπορεί να αποδειχθεί από το σύστημα θεωρείται ως ορθή η άρνησή της). Ωστόσο, η αντιμετώπιση της σημασιολογικά σχολιασμένης γνώσης υπό την υπόθεση κλειστού κόσμου, δεν είναι πάντοτε εννοιολογικά σωστή. Πιο συγκεκριμένα, αυτό εξαρτάται άμεσα από τη φύση του προβλήματρος που μοντελοποιούμε, καθώς και από τον τύπο επερωτήσεων που θέλουμε να απαντάει το σύστημά μας Αυτό οφείλεται στην ανοιχτή φύση των σημασιολογικών δεδομένων, τα οποία προστίθενται και αλλάζουν συνεχώς. Συνεπώς, αν ένα συγκεκριμένο κομμάτι γνώσης δε μπορεί να ανακτηθεί, τότε σύμφωνα με την υπόθεση του ανοιχτού κόσμου (open world assumption - OWA) δεν μπορεί να υποτεθεί ασφαλώς ότι είναι ψευδές.

5. ΣΥΓΚΡΙΣΗ ΣΥΣΤΗΜΑΤΩΝ

5.1 Περιγραφή Μεθοδολογίας

Για την προσθήκη ευφυΐας σε εφαρμογές, απαιτείται η δυνατότητα εξαγωγής συμπερασμάτων από τη ρητά εκφρασμένη γνώση του πεδίου ενδιαφέροντος. Με τον τρόπο αυτό, οι εφαρμογές θα μπορούν να παίρνουν μόνες τους αποφάσεις, όπως ακριβώς και οι άνθρωποι. Οι οντολογίες είναι η μέθοδος αναπαράστασης γνώσης που προτιμάται γι αυτόν τον σκοπό. Παρόλα αυτά, η γλώσσα οντολογιών (OWL) χαρακτηρίζεται από υψηλή πολυπλοκότητα και η εξαγωγή συμπερασμάτων από αυτή απαιτεί υψηλή κατανάλωση πόρων, στοιχεία που δε μπορούν να αντιμετωπιστούν σε ένα περιβάλλον με περιορισμένους πόρους (ταχύτητα επεξεργαστή, μνήμη), όπως μια κινητή συσκευή. Για τον λόγο αυτό ενδείκνυται η μετατροπή της γλώσσας οντολογιών σε γλώσσα χαμηλότερης εκφραστικότητας, της οποίας η μηχανή συμπερασμού να μπορεί να εκτελεστεί σε κινητές συσκευές.

Έχουμε ήδη αναφέρει ότι δυο τέτοιες αναπαραστάσεις γνώσης είναι η λογική πρώτης τάξης και ο λογικός προγραμματισμός – η Prolog στην περίπτωσή μας. Σε αυτό το κεφάλαιο θα παρουσιάσουμε τη μεθοδολογία και τα αποτελέσματα των συλλογιστικών διαδικασιών αυτών των γλωσσών σε μια κινητή συσκευή, με σκοπό να αναδείξουμε τις δυνατότητες και τις απώλειες της αναπαράστασης γνώσης και του συμπερασμού σε κινητά περιβάλλοντα. Οι μηχανές συμπερασμού που έχουν επιλεχθεί για την υλοποίηση των δοκιμών είναι τα Pocket KRHyper και JIProlog, αντίστοιχα.

Η κινητή συσκευή στην οποία έγιναν οι παρακάτω μετρήσεις, είναι ένα Dell Axim X51V PDA, με επεξεργαστή Intel PXA270 στα 624 MHz και μνήμη RAM 64MB. Το λειτουργικό σύστημα το οποίο χρησιμοποιήθηκε στα πλαίσια της εργασίας είναι τα Windows Mobile 5 και η εκτέλεση των εργαλείων έγινε μέσω της Mysaifu Java J2SE Virtual Machine (JVM). Η Mysaifu JVM είναι ένα δωρεάν λογισμικό ανοιχτού κώδικα και ο λόγος που επιλέχθηκε είναι ότι υποστηρίζει Java 2 Standard Edition (J2SE), επιτρέποντας, έτσι, την ανάπτυξη και εκτέλεση λογισμικού σε standard Java. Η μνήμη που παραχωρήθηκε για τις μετρήσεις ήταν 25 MB, μιας και περίπου τόσα ήταν ελεύθερα από το λειτουργικό σύστημα του PDA σε κατάσταση αναμονής.

Οι οντολογίες που χρησιμοποιήθηκαν για τις δοκιμές είναι απλές συνθετικές οντολογίες OWL χωρίς περιορισμούς, των οποίων η κατασκευή περιγράφεται στην ενότητα 5.2.1. Για τη μετατροπή μιας οντολογίας σε προτάσεις λογικής πρώτης τάξης δημιουργήθηκε κώδικας (σε γλώσσα Java) του οποίου κανόνες μετατροπής παρουσιάζονται στον πίνακα . Η μετατροπή σε Prolog, έγινε αφαιρώντας από τον ίδιο κώδικα τα κομμάτια των στοιχείων που δεν υποστηρίζει η Prolog και οι κανόνες μετατροπής παρουσιάζονται, επίσης, στον Πίνακα 5.1. Οι κενές θέσεις του πίνακα αντιστοιχούν σε primitives που δε δοκιμάστηκαν σε Prolog.

Η ισότητα στη λογική πρώτης τάξης προσομοιώθηκε με τη δημιουργία του προκαθορισμένου κατηγορήματος '=' και τη σύλληψη των αξιωμάτων της. Η ανακλαστικότητα, η συμμετρία και η μεταβατικότητα είναι τα αξιώματα που εξασφαλίζουν ότι το κατηγόρημα της ισότητας κατέχει τις αλγεβρικές ιδιότητες της σχέσης της ισοδυναμίας:

= (x, x) ←

= (x, y) ← = (y, x)

= (x, z) ← = (x, y),= (y, z)

Το αξίωμα της αντικατάστασης εξασφαλίζει τη σωστή σημασιολογία της ισοδυναμίας, π.χ. όταν ισχύει το $= (c , d) \wedge Q(c)$, τότε ισχύει και το $Q(d)$ για όλα τα κατηγορήματα Q.

Q(x1, .., xn) ← Q(y1, .., yn) ∪ {= (xi, yi)|1 ≤i ≤ n}

Πρέπει να σημειωθεί ότι, το αξίωμα της αντικατάστασης πρέπει να υλοποιηθεί για όλα τα κατηγορήματα Q που χρησιμοποιούνται στη βάση γνώσης.

Πίνακας 5.1: Το mapping που χρησιμοποιήθηκε για τη μετατροπή των οντολογιών

OWL primitives	FOL	Prolog
C sameClassAs D	C(X) :- D(X). D(X) :- C(X).	Δεν υποστηρίζεται
C subClassOf D	D(X) :- C(X).	D(X):-C(X).
C unionOf D (μόνο στο σώμα)	C(X), D(X)	C(X), D(X)
C intersectionOf D (μόνο στην κεφαλή)	C(X) ; D(X)	C(X) ; D(X)
complementOf C	not(C(X))	not C(X)
C disjointWith D	false :- C(X),D(X).	:- C(X),D(X).
P subPropertyOf Q	Q(X,Y) :- P(X,Y).	Q(X,Y) :- P(X,Y).
P samePropertyAs Q	P(X,Y) :- Q(X,Y). Q(X,Y) :- P(X,Y).	Δεν υποστηρίζεται
P domain C	C(X) :- P(X, Y).	C(X) :- P(X, Y).
P range C	C(Y) :- P(X, Y).	C(Y) :- P(X, Y).
TransitiveProperty P	P(X,Z) :- P(X,Y), P(Y,Z).	Δεν υποστηρίζεται
SymmetricProperty P	P(X,Y) :- P(Y, X).	Δεν υποστηρίζεται
FunctionalProperty P	= (X,Y) :- P(Z,X), P(Z,Y).	
InverseFunctionalProperty P	= (X,Y) :- P(X,Z), P(Y,Z).	
P inverseOf Q	P(X,Y) :- Q(Y,X). Q(X,Y) :- P(Y,X).	Δεν υποστηρίζεται
P minimumCardinality C	Δεν υποστηρίζεται	Δεν υποστηρίζεται
P maximumCardinality C	(;1≤i<n, i<j≤n =(Xi,Xj)) :− P(A,X1),C(X1),…,(P(A,Xn),C(Xn).	
P allValuesFrom C	D(Y) :- P(X,Y),C(X).	
P someValuesFrom C	false :-D(X),not(myPred(X,Y)). myPred(X,Y) :- P(X,Y), C(Y).	
P hasValue b	C(X) :- P(X,v). P(X,v) :- C(X).	
oneOf(b1, b2, ..., bi, bn)	=(X,b1);=(X,b2);…;=(X,bn) :- C(X).	

Για τη μέτρηση της επίδοσης των Pocket KRHyper και JIProlog, πραγματοποιήθηκαν οι ίδιες δοκιμές. Κάθε διαδικασία συμπερασμού σε μια οντολογία, εκτελέστηκε και μετρήθηκε 3 φορές, εξαιρώντας περιπτώσεις που η μέτρηση ήταν πολύ μικρότερη ή μεγαλύτερη από το αναμενόμενο, δεδομένου ότι σε ένα λειτουργικό σύστημα οι διαθέσιμοι πόροι αλλάζουν συνεχώς και άρα τέτοιες περιπτώσεις μετρήσεων δεν αποτελούν ενδεικτικά αποτελέσματα. Το τελικό αποτέλεσμα κάθε δοκιμής είναι ο μέσος όρος των τριών μετρήσεων.

Συνοψίζοντας, η μεθοδολογία που ακολουθήθηκε είναι η εξής:

- Δημιουργία συνθετικών οντολογιών
- Μετατροπή οντολογίας σε προτάσεις λογικής πρώτης τάξης για το συμπερασμό στο Pocket KRHyper
- Μετατροπή οντολογίας σε κώδικα Prolog για το συμπερασμό στο JIProlog
- Εκτέλεση συλλογιστικής διαδικασίας σε κάθε οντολογία με το Pocket KRHyper και καταγραφή μετρήσεων
- Εκτέλεση συλλογιστικής διαδικασίας σε κάθε οντολογία με το JIProlog και καταγραφή μετρήσεων

Οι οντολογίες που δοκιμάστηκαν ήταν συνολικά 57. Στους πίνακες μετρήσεων παρουσιάζονται μόνο οι οντολογίες για τις οποίες πήραμε αποτέλεσμα μέσα στο χρονικό όριο που είχαμε θέσει.

5.2 Υλοποίηση περιβάλλοντος δοκιμών

5.2.1 Κατασκευή οντολογιών

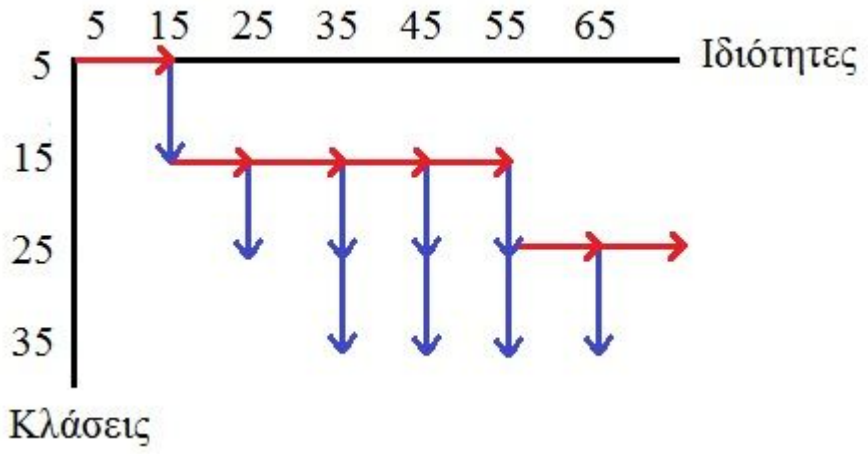
Σε αυτήν την ενότητα θα περιγράψουμε τον τρόπο κατασκευής των συνθετικών οντολογιών που χρησιμοποιήθηκαν στις παρακάτω δοκιμές. Όλες οι οντολογίες παράχθηκαν επεκτείνοντας την αμέσως προηγούμενή τους. Ως σημείο εκκίνησης δημιουργήσαμε μια οντολογία αποτελούμενη από 5 κλάσεις, 5 στιγμιότυπα κλάσεων, 5 ιδιότητες και 5 στιγμιότυπα ιδιοτήτων, την οποία αναφέρουμε συντομογραφικά ως οντολογία 5-5-5-5. Οι σχέσεις των στοιχείων της οντολογίας είναι τυχαίος. Το πρόγραμμα επιλέγει με τυχαίο τρόπο, αν μια κλάση είναι υποκλάση μιας άλλης, αν μια ιδιότητα είναι υπο-ιδιότητα μιας άλλης, ποια χαρακτηριστικά θα έχει μια ιδιότητα και ποιες κλάσεις θα αποτελέσουν το πεδίο ορισμού και το πεδίο τιμών της. Τα στιγμιότυπα κατανέμονται ομοιόμορφα στις κλάσεις.

Κάθε οντολογία που δημιουργείται, επεκτείνεται στη συνέχεια κατά (τουλάχιστον) 10 στοιχεία. Δηλαδή, 10 κλάσεις ή στιγμιότυπα κλάσεων ή ιδιότητες ή στιγμιότυπα ιδιοτήτων προσθέτονται στην προηγούμενη οντολογία, σχηματίζοντας μια καινούρια. Για παράδειγμα, η οντολογία 15-5-15-5 θα δημιουργηθεί προσθέτοντας 10 κλάσεις στην οντολογία 5-5-15-5, ενώ η οντολογία 15-5-25-5 θα δημιουργηθεί προσθέτοντας 10 ιδιότητες στην οντολογία 15-5-15-5.

Η επέκταση αυτή δεν είναι απεριόριστη και δεσμεύεται από τους ακόλουθους κανόνες:

- Ο αριθμός των κλάσεων δεν πρέπει να υπερβαίνει τον αριθμό των ιδιοτήτων.
- Ο αριθμός των ιδιοτήτων δεν πρέπει να υπερβαίνει το τριπλάσιο του αριθμού των κλάσεων.
- Ο αριθμός των στιγμιότυπων των κλάσεων δεν πρέπει να υπερβαίνει τον αριθμό των στιγμιότυπων των ιδιοτήτων.
- Ο αριθμός των στιγμιότυπων των ιδιοτήτων δεν πρέπει να υπερβαίνει το τριπλάσιο του αριθμού των στιγμιότυπων των κλάσεων.

Οι παραπάνω περιορισμοί ελήφθησαν υπόψη κατά την εκτέλεση της πειραματικής αξιολόγησης που περιγράφουμε σε αυτή την ενότητα με σκοπό τη δημιουργία και υιοθέτηση ρεαλιστικών οντολογικών μοντέλων και στιγμιότυπων σε ότι αφορά τα χαρακτηριστικά τους μεγέθη.



Εικόνα 5.1: Γραφική απεικόνιση του τρόπου επέκτασης των οντολογικών μοντέλων

Ο λόγος που έχει επιλέχθηκε αυτός ο επεκτατικός τρόπος κατασκευής οντολογιών, είναι η εξασφάλιση της ομοιότητας των οντολογιών. Τα στοιχεία της νέας οντολογίας διέπονται από τις ίδιες ακριβώς σχέσεις που είχαν τα στοιχεία της προηγούμενης οντολογίας. Αν αυτό δε συνέβαινε, οι μετρήσεις που θα παίρναμε δε θα μας επέτρεπαν να βγάλουμε ορθά συμπεράσματα, αφού οι σχέσεις ανάμεσα στα δομικά στοιχεία της κάθε οντολογίας θα ήταν διαφορετικές. Για παράδειγμα, αν είχαμε δυο οντολογίες με τον ίδιο αριθμό στοιχείων, τα αποτελέσματα που επέστρεφε η διαδικασία συμπερασμού

μπορεί να ήταν ίδια ή και τελείως διαφορετικά. Μη γνωρίζοντας, λοιπόν, την ομοιότητα των οντολογιών οι μετρήσεις δεν επιτρέπουν την εξαγωγή συμπερασμάτων. Οι οντολογίες δημιουργήθηκαν μέσω του Protégé OWL API.

5.2.2 Μετρικές

Η μετρική των δοκιμών είναι ο χρόνος σε δευτερόλεπτα και το χρονικό όριο, που επιτρέπουμε σε μια διαδικασία συμπερασμού να εκτελείται, είναι τα 5 λεπτά. Η εκτέλεση του Pocket KRHyper επιστρέφει τον χρόνο που καταναλώνεται για την εύρεση ή μη, ενός μοντέλου για τη βάση γνώσης που δημιουργήθηκε από μια οντολογία. Ο χρόνος αυτός δεν εμπεριέχει τη δημιουργία της βάσης γνώσης από το .tme αρχείο, που βρίσκονται οι προτάσεις λογικής πρώτης τάξης. Η εκτέλεση του JIProlog επιστρέφει δυο χρόνους. Ο πρώτος είναι ο χρόνος φόρτωσης του .pl αρχείου, που περιέχει τον κώδικα Prolog. Ο δεύτερος είναι ο χρόνος για την εύρεση όλων των λύσεων της επερώτησης «?- property0(X,Y).».

Η κατανάλωση της μνήμης δεν αποτελεί μετρική των δοκιμών της παρούσας εργασίας, μιας και η συσκευή που χρησιμοποιήθηκε δεν παρείχε γραμμή εντολών για την εκτέλεση εργαλείων όπως το jconsole. Η δυνατότητα προβολής της κατανάλωσης της μνήμης από την Mysaifu JVM, δεν ενδείκνυται για μετρήσεις καθώς παρουσιάζεται το ποσοστό επί τοις εκατό της μνήμης που καταναλώνεται, το οποίο αλλάζει συνεχώς με αποτέλεσμα να μην είναι δυνατή η καταγραφή μέτρησης.

5.2.3 Συστήματα

Οι οντολογίες που χρησιμοποιήθηκαν για τις μετρήσεις στο Pocket KRHyper δεν περιέχουν περιορισμούς, αλλά και τα χαρακτηριστικά των ιδιοτήτων functional και inverse functional, λόγω της υψηλής κατανάλωσης πόρων που προκαλούν. Η ισότητα αντιμετωπίζεται με την παροχή ενός προκαθορισμένου κατηγορήματος, που προσομοιώνει τα αξιώματα της ισοδυναμίας. Οι οντολογίες έχουν μετατραπεί σε προτάσεις λογικής πρώτης τάξης και αποθηκεύτηκαν σε αρχείο .tme, που είναι η μορφή αρχείου που χρησιμοποιεί το Pocket KRHyper.

Οι δοκιμές έγιναν αρχικά στο Pocket KRHyper και για αυτό οι οντολογίες που δοκιμάστηκαν στο JIProlog είναι όσες επέστρεψαν αποτέλεσμα, εντός του χρονικού ορίου των 5 λεπτών, στις δοκιμές που έγιναν στο Pocket KRHyper.

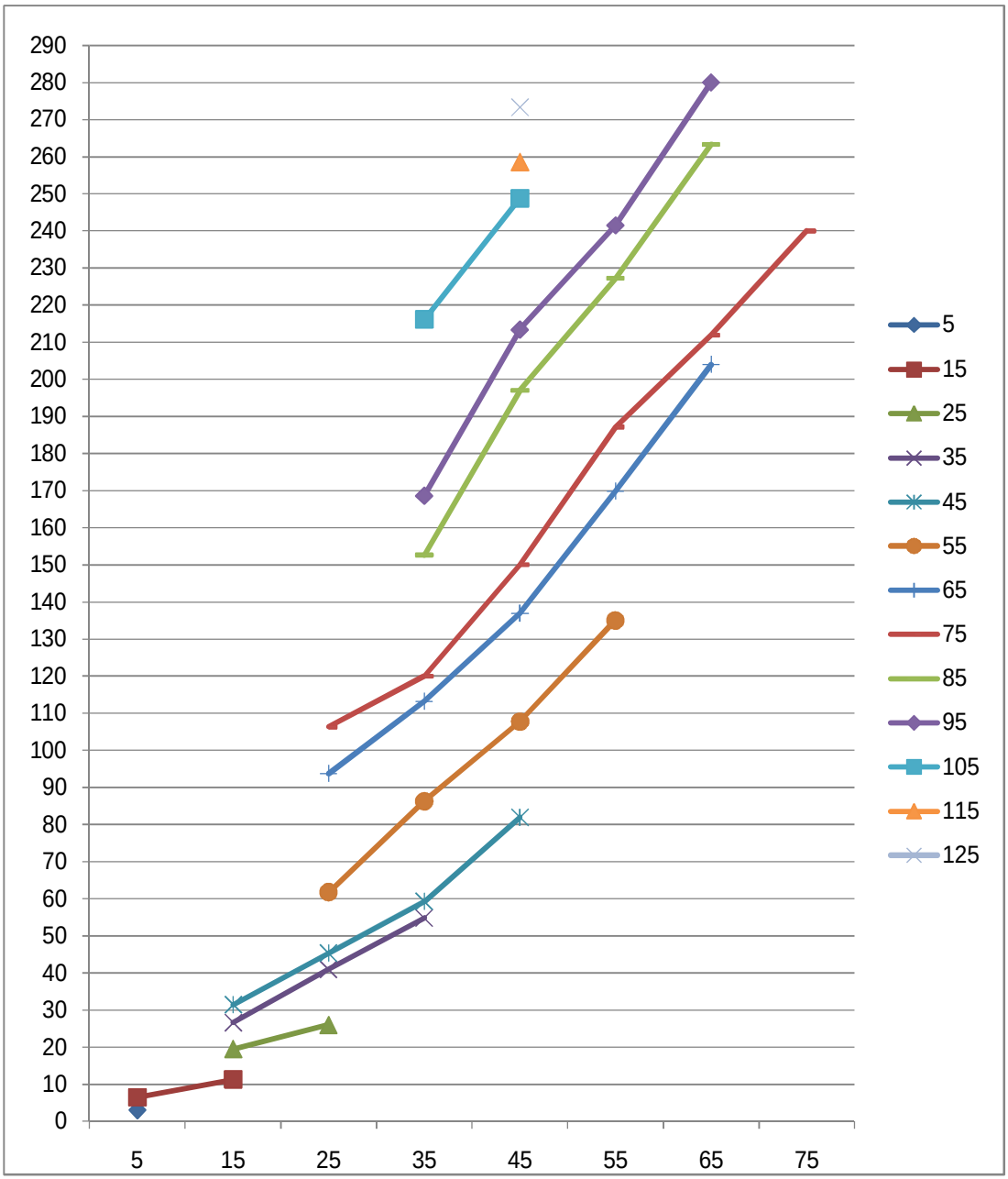
Το JIProlog δεν μπορεί να τερματίσει όταν συναντά μια πρόταση στην οποία το ίδιο κατηγόρημα εμφανίζεται και στο σώμα και στην κεφαλή της πρότασης (π.χ. η πρόταση transitive ιδιότητας) ή όταν ένα σύνολο προτάσεων δημιουργεί κάποιο είδος κύκλου (π.χ. οι προτάσεις της ισότητας κλάσεων). Ως εκ τούτου, οι προτάσεις που αντιπροσωπεύουν την ισότητα και τα χαρακτηριστικά των ιδιοτήτων inverse of, symmetric και transitive αφαιρέθηκαν από τα αρχεία .pl, που δέχεται το JIProlog. Τέλος, στη βάση γνώσης που υλοποιείται από το αρχείο, προστίθεται η πρόταση «property0(X,Y) :- property2(X,Z), property4(Z,Y).», δημιουργώντας μια νέα σχέση ανάμεσα σε αυτές τις ιδιότητες και αυξάνοντας τελικά την πολυπλοκότητα της επερώτησης «?-property0(X,Y).», για την πραγματοποίηση των δοκιμών.

5.3 Δοκιμές – Αποτελέσματα

Στην ενότητα αυτή παρουσιάζονται τα αποτελέσματα των δοκιμών που εκτελέστηκαν στις μηχανές συμπερασμού Pocket KRHyper και JIProlog. Οι μετρήσεις που λήφθηκαν προβάλλονται σε διαγράμματα, που ακολουθούν το ίδιο πρότυπο. Κάθε διάγραμμα αντιστοιχεί σε έναν καθορισμένο αριθμό κλάσεων και στιγμιότυπων. Ο κάθετος άξονας αντιπροσωπεύει τον χρόνο σε δευτερόλεπτα που χρειάστηκε για την εκτέλεση της συλλογιστικής διαδικασίας πάνω σε μια οντολογία, από την εκάστοτε μηχανή συμπερασμού, ή τον χρόνο φόρτωσης μιας οντολογίας στην περίπτωση του JIProlog. Ο οριζόντιος άξονας αντιπροσωπεύει τον αριθμό των στιγμιότυπων των κλάσεων και οι χρωματιστές γραμμές αντιπροσωπεύουν τον αριθμό των στιγμιότυπων των ιδιοτήτων.

5.3.1 Αποτελέσματα πειραμάτων στο pocket KRHyper

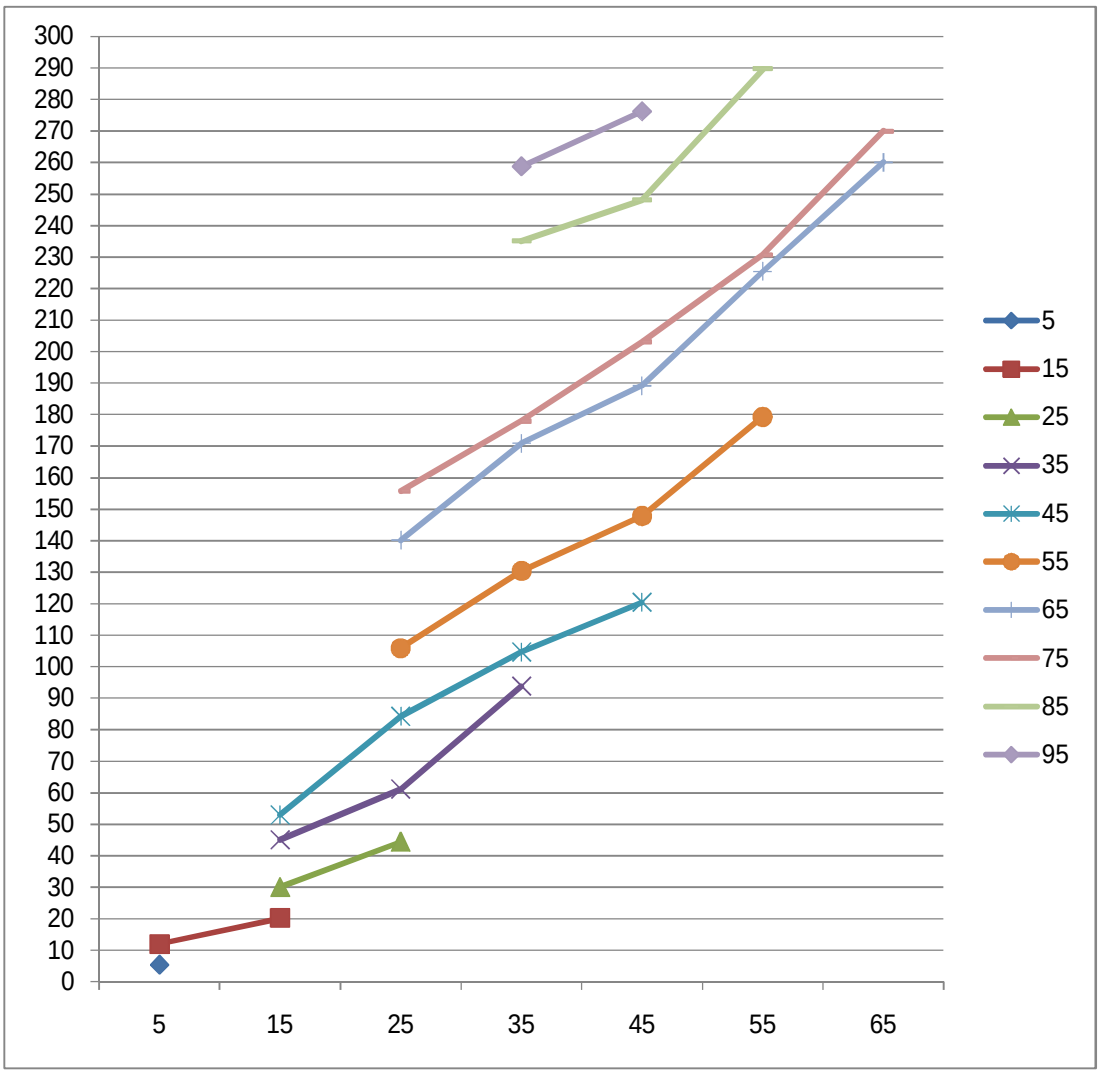
Στα επόμενα διαγράμματα παρουσιάζονται οι μετρήσεις των οντολογιών που δοκιμάστηκαν στο Pocket KRHyper. Η αύξηση κάποιου στοιχείου της οντολογίας, όπως αναφέρθηκε στην ενότητα 5.2.1, σταματούσε όποτε παραβιαζόταν κάποιος από τους κανόνες επέκτασης ή το χρονικό όριο εκτέλεσης, ή παρουσιαζόταν η εξαίρεση «out of memory».



Σχήμα 5.1: Οντολογικό μοντέλο 5 κλάσεων και 5 ιδιοτήτων

Στο πρώτο σχήμα εμφανίζονται οι μετρήσεις σε ένα οντολογικό μοντέλο με 5 κλάσεις και 5 ιδιότητες (ή αλλιώς μοντέλο 5-5). Το μοντέλο αυτό είναι πολύ μικρό και έτσι βλέπουμε ότι για 5 στιγμιότυπα κλάσεων ο χρόνος συμπερασμού κυμαίνεται ανάμεσα σε 3-6 δευτερόλεπτα. Λόγω του μικρού του μεγέθους έχουμε καταφέρει να αυξήσουμε τα στιγμιότυπα κλάσεων μέχρι και 15 φορές και των ιδιοτήτων μέχρι και 25 φορές από το αρχικό, δηλαδή τα 5 στιγμιότυπα. Παρατηρούμε ότι ο χρόνος αυξάνεται καθώς αυξάνονται τα στιγμιότυπα των κλάσεων ή/και τα στιγμιότυπα των ιδιοτήτων, όπως θα

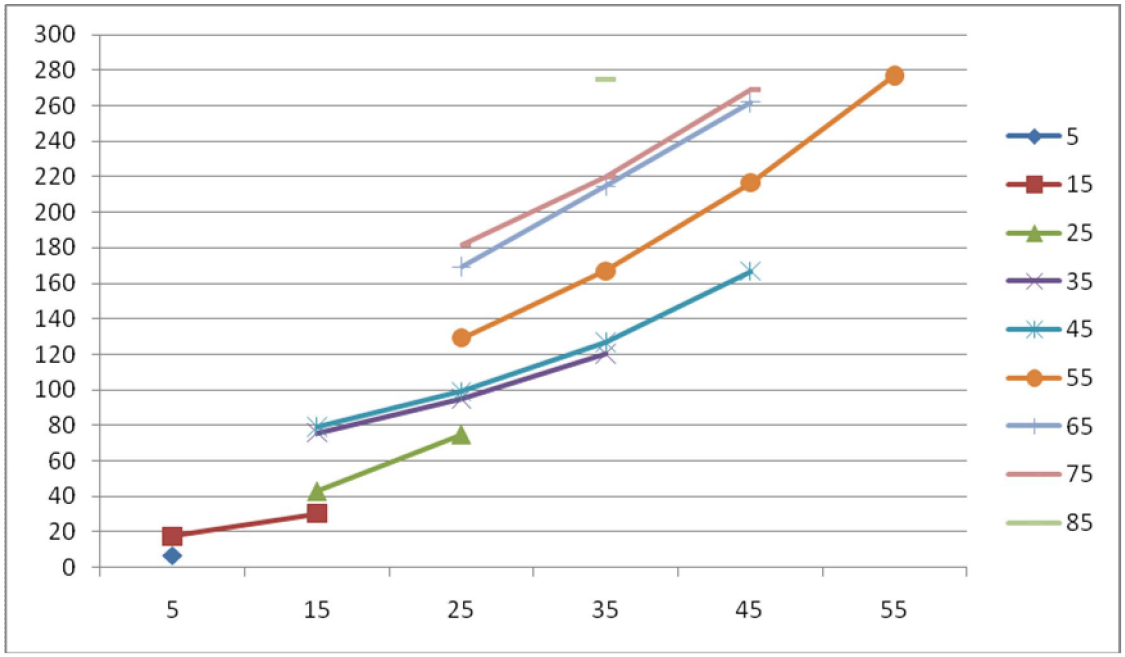
αναμέναμε. Η οντολογία 5-45-5-55 κατανάλωσε 108 δευτερόλεπτα. Αυξάνοντας τα στιγμιότυπα κλάσεων έχουμε την οντολογία 5-55-5-55 η οποία κατανάλωσε 135 δευτερόλεπτα, ενώ αυξάνοντας τα στιγμιότυπα ιδιοτήτων έχουμε την οντολογία 5-45-5-65 που έχει μέτρηση 137 δευτερόλεπτα. Συμπεραίνουμε λοιπόν, ότι οι αριθμοί και των δυο ειδών στιγμιότυπων συνεισφέρουν εξίσου στην αύξηση της πολυπλοκότητας μιας οντολογίας. Το σχήμα αυτό έχει τις περισσότερες οντολογίες και καμία επέκταση αυτών των οντολογιών δεν παρουσίασε εξαίρεση ανεπαρκούς μνήμης.



Σχήμα 5.2: Οντολογικό μοντέλο 5 κλάσεων και 15 ιδιοτήτων

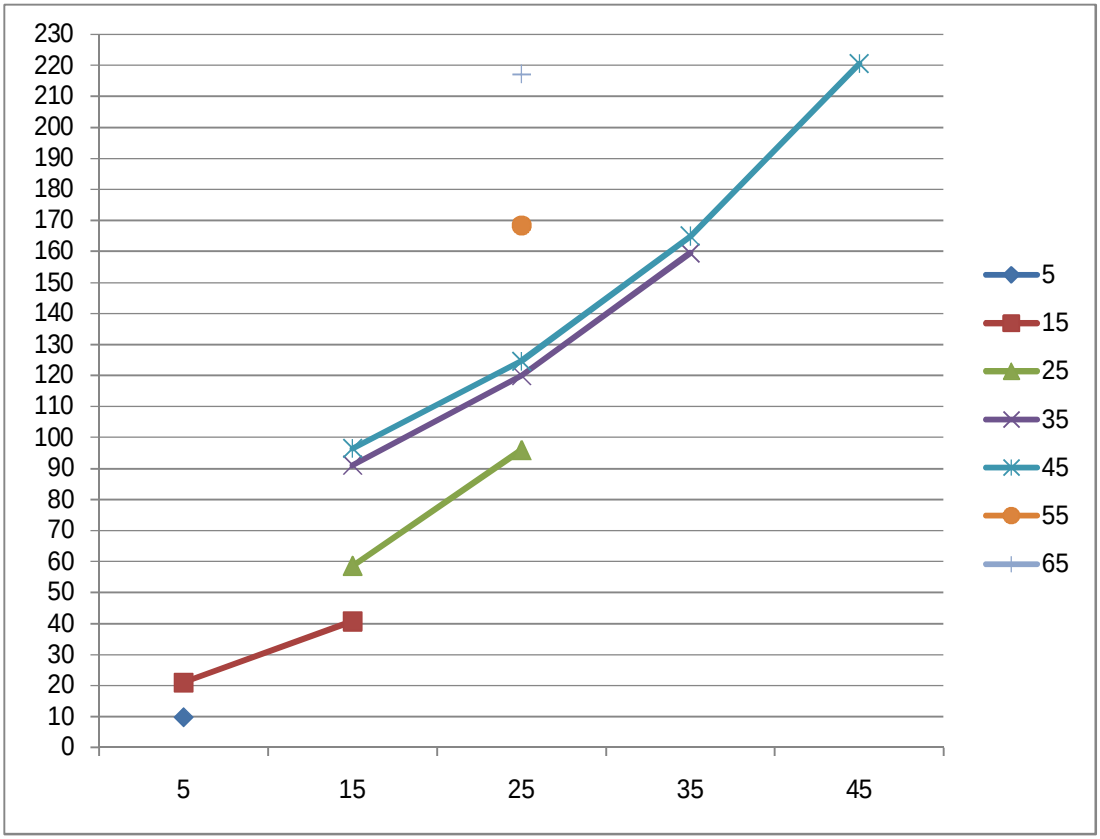
Στο Σχήμα 5.2 το μοντέλο της οντολογίας αποτελείται από 5 κλάσεις και 15 ιδιότητες (μοντέλο 5-15). Ο αριθμός των οντολογιών που δημιουργήθηκαν πάνω σε αυτό το μοντέλο έχει μειωθεί, σε σχέση με το προηγούμενο οντολογικό μοντέλο. Οι χρόνοι έχουν σχεδόν διπλασιαστεί. Για παράδειγμα, οι οντολογίες με 5 στιγμιότυπα κλάσεων,

εδώ κυμαίνονται στα 5-12 δευτερόλεπτα. Επομένως, ο τριπλασιασμός των ιδιοτήτων επέφερε διπλασιασμό στους χρόνους όλων των σχετικών οντολογιών. Έτσι, αν και ακόμα δεν υπήρξε πρόβλημα έλλειψης μνήμης, το προκαθορισμένο χρονικό όριο υπερβαίνεται συντομότερα.



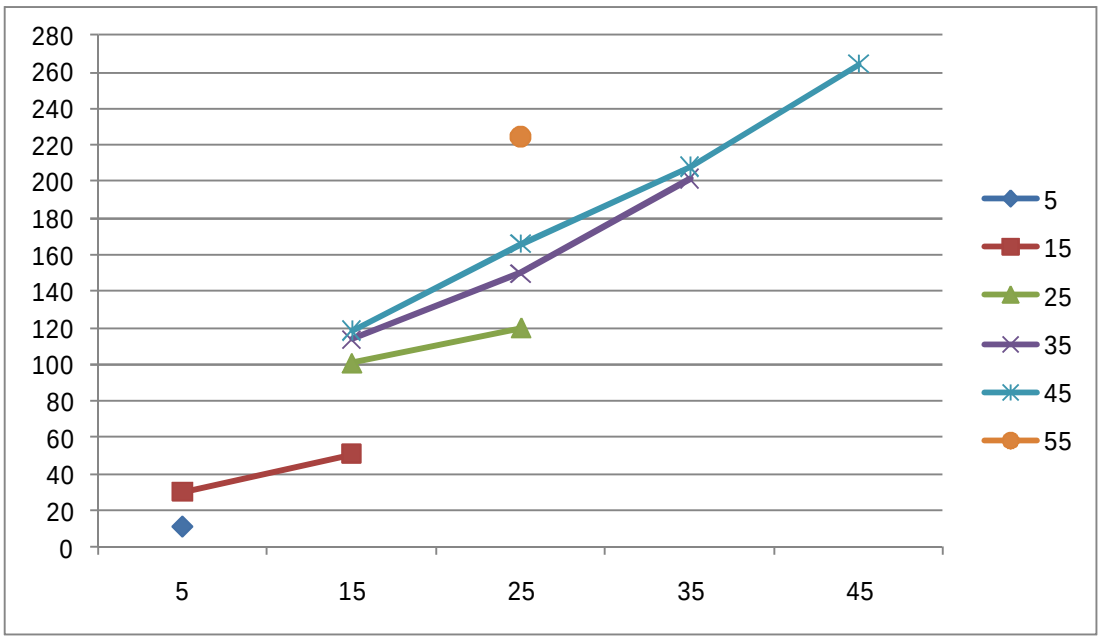
Σχήμα 5.3: Οντολογικό μοντέλο 15 κλάσεων και 15 ιδιοτήτων

Ομοίως, στο Σχήμα 5.3 παρατηρούμε αντίστοιχες χρονικές αυξήσεις, ενώ ο αριθμός των κλάσεων έχει τριπλασιαστεί. Η επέκταση των οντολογιών έχει περιοριστεί. Στο προηγούμενο σχήμα ο μεγαλύτερος αριθμός στιγμιότυπων κλάσεων και ιδιοτήτων, ήταν 65 και 95 αντίστοιχα. Σε αυτό το οντολογικό μοντέλο οι αριθμοί αυτοί έχουν μειωθεί κατά 10 στιγμιότυπα, δηλαδή κατά ένα βήμα επέκτασης.



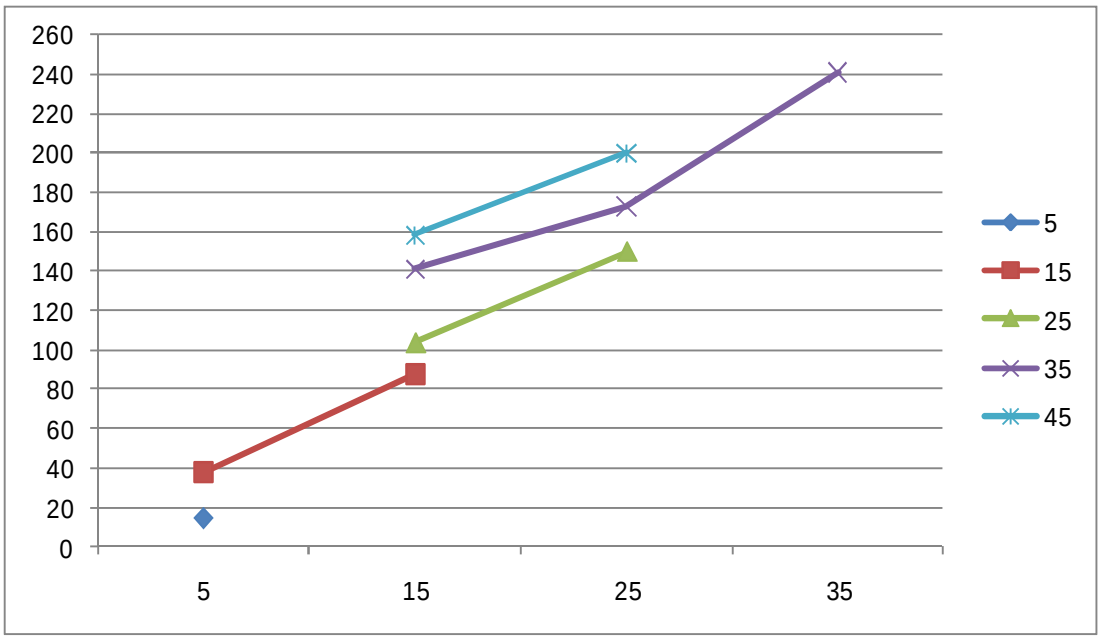
Σχήμα 5.4: Οντολογικό μοντέλο 15 κλάσεων και 25 ιδιοτήτων

Είναι εμφανές ότι όσο μεγαλώνει ο αριθμός των κλάσεων και των ιδιοτήτων, τόσο περισσότερο χρόνο απαιτεί η συλλογιστική διαδικασία σε μια οντολογία, ακόμα και αν έχει λίγα στιγμιότυπα, και τόσο περιορίζονται τα βήματα επέκτασής της. Στο Σχήμα 5.4, το οντολογικό μοντέλο είναι το 15-25. Η μικρότερη οντολογία (δηλαδή με 5 στιγμιότυπα κλάσεων και ιδιοτήτων) καταναλώνει 10 δευτερόλεπτα. Οι μετρήσεις έχουν πλέον υπερτριπλασιαστεί σε σχέση με την πρωταρχική οντολογία 5-5-5-5. Θα πρέπει να σημειωθεί ότι σε αυτό το μοντέλο, παρουσιάζεται πρώτη φόρα η εξαίρεση «out of memory». Συγκεκριμένα, η εξαίρεση αυτή εμφανίστηκε στην προσπάθεια εκτέλεσης της συλλογιστικής στις οντολογίες 15-25-25-75, 15-35-25-55, 15-45-25-55. Αυτό σημαίνει ότι η κινητή συσκευή δε μπορεί να ανταπεξέλθει σε οντολογίες τέτοιου μεγέθους και πολυπλοκότητας.



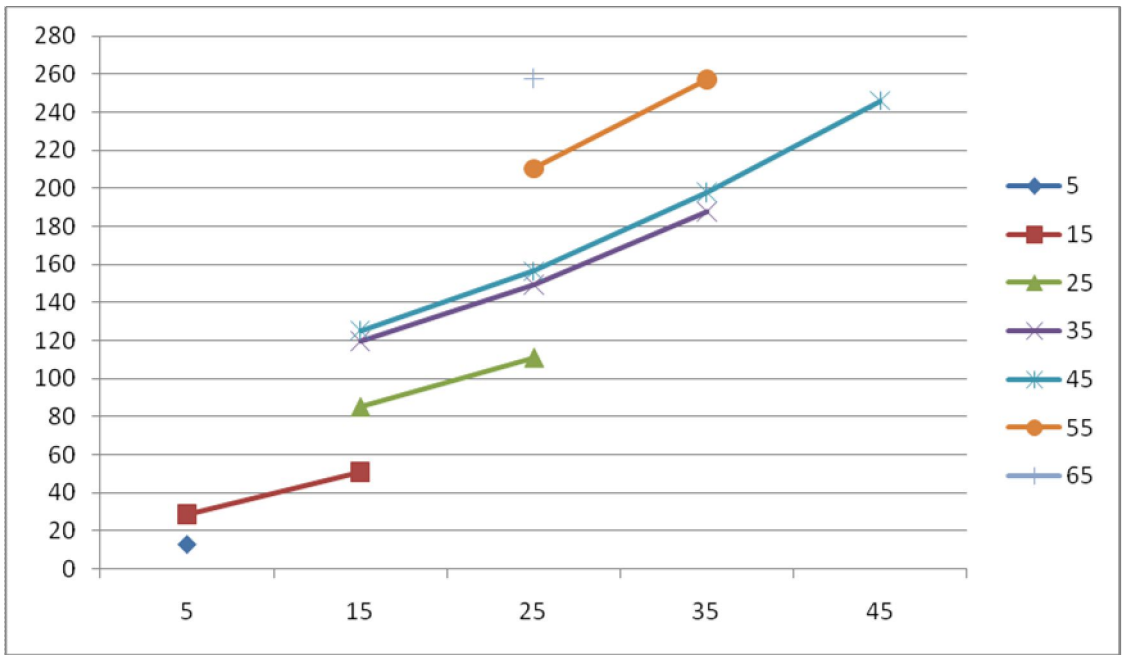
Σχήμα 5.5: Οντολογικό μοντέλο 15 κλάσεων και 35 ιδιοτήτων

Το οντολογικό μοντέλο 15 -35 παρουσιάζεται στο Σχήμα 5.5. Η επέκταση των οντολογιών έχει μειωθεί σχεδόν στο μισό, σε σχέση με την αρχική οντολογία. Στην οντολογία 5-5-5-5, η επέκταση των στιγμιότυπων των κλάσεων έφτανε τα 75 στιγμιότυπα ενώ σε αυτό το μοντέλο φτάνει τα 45 στιγμιότυπα. Ομοίως, τα στιγμιότυπα ιδιοτήτων έφταναν τα 125, ενώ εδώ φτάνουν μόνο τα 55.



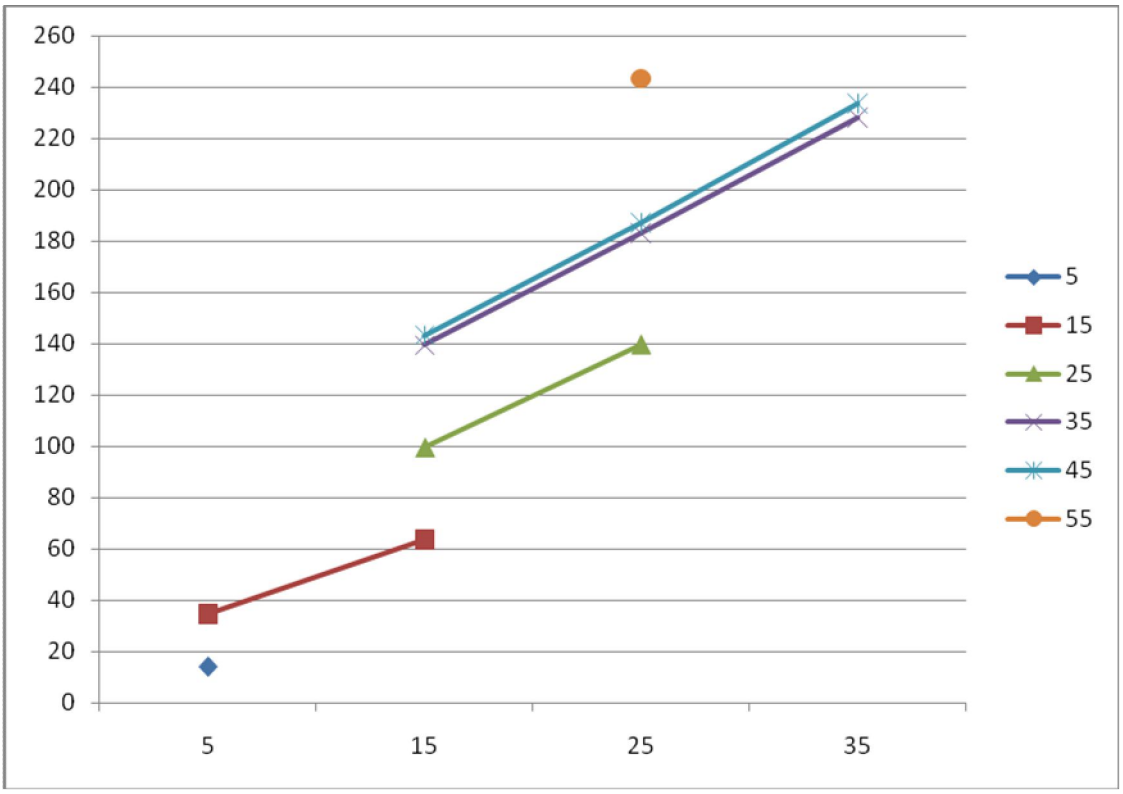
Σχήμα 5.6: Οντολογικό μοντέλο 15 κλάσεων και 45 ιδιοτήτων

Στο Σχήμα 5.6 έχουμε το τελευταίο οντολογικό μοντέλο με 15 κλάσεις. Ο αριθμός των ιδιοτήτων είναι 45. Παρατηρούμε ότι ο χρόνος που καταναλώνεται για την οντολογία 15-5-45-15 είναι σχεδόν 40 δευτερόλεπτα. Οι προηγούμενες οντολογίες 15-5-35-15 και 15-5-25-15, είχαν μέτρηση 30 και 20 δευτερόλεπτα, αντιστοίχως. Επομένως, καθώς αυξάνεται ανά 10 ο αριθμός ιδιοτήτων των οντολογιών, αυξάνεται και ο χρόνος συμπερασμού κατά 10 δευτερόλεπτα.



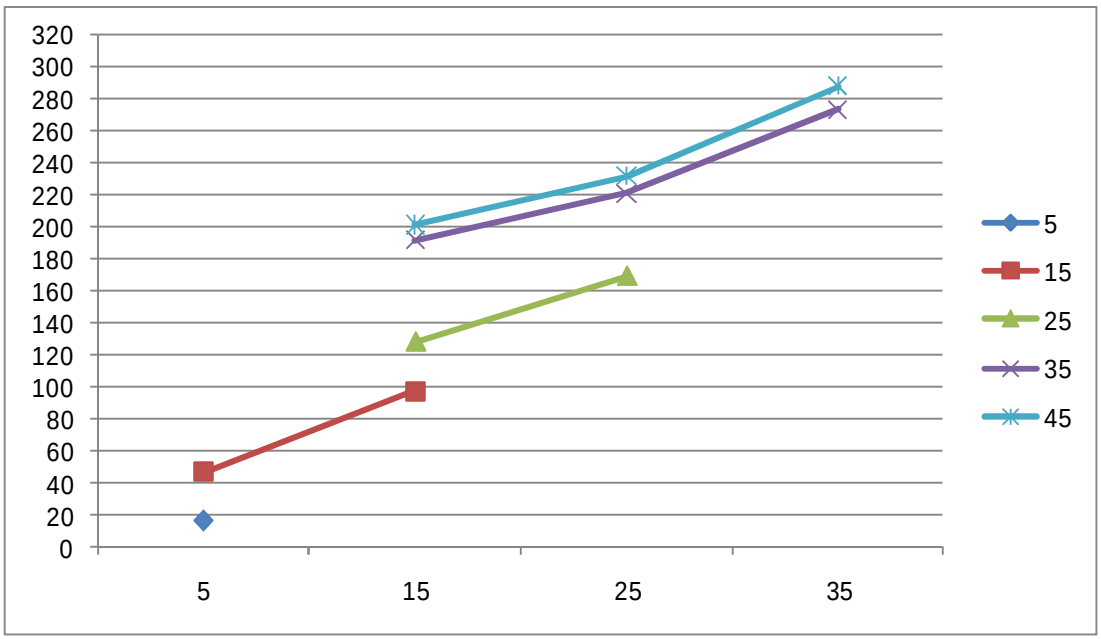
Σχήμα 5.7: Οντολογικό μοντέλο 25 κλάσεων και 25 ιδιοτήτων

Στο Σχήμα 5.7 είναι εμφανής η αύξηση των επεκτατικών βημάτων των οντολογιών του μοντέλου, αλλά και η μείωση των χρόνων. Αυτό συμβαίνει επειδή σε αυτό το διάγραμμα παρουσιάζεται το μοντέλο 25-25. Έτσι, βλέπουμε ότι η οντολογία 25-5-25-15 έχει μέτρηση 29 δευτερόλεπτα, ενώ η 15-5-45-15 είχε μέτρηση 38 δευτερόλεπτα. Η μείωση αυτή οφείλεται στο ότι, αν και έχουμε αύξηση στον αριθμό των κλάσεων, έχουμε τη διπλάσια μείωση των ιδιοτήτων, που χαμηλώνει την πολυπλοκότητα της οντολογίας αυτής.



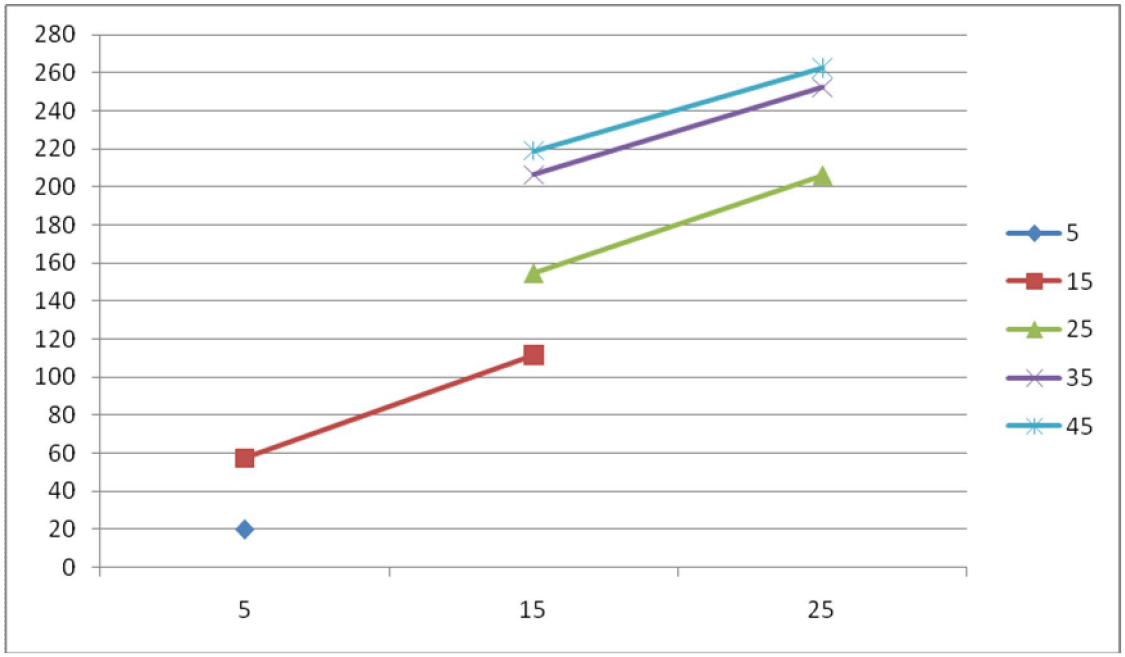
Σχήμα 5.8: Οντολογικό μοντέλο 25 κλάσεων και 35 ιδιοτήτων

Οι χρόνοι συμπερασμού για το μοντέλο 25-35, που εμφανίζονται στο Σχήμα 5.8, φαίνεται ότι αρχίζουν να πλησιάζουν τους χρόνους του μοντέλου 15-45. Οι χρόνοι έχουν αυξηθεί και τα βήματα επέκτασης έχουν μειωθεί. Καθώς μεγαλώνει ο αριθμός των στιγμιότυπων ιδιοτήτων, αυξάνεται κ ο χρόνος συμπερασμού της οντολογίας. Η αύξηση αυτή κυμαίνεται ανάμεσα στα 20-55 δευτερόλεπτα, ανάλογα με τον αριθμό των στιγμιότυπων. Εξαίρεση φαίνεται να αποτελεί η αύξηση των στιγμιότυπων ιδιοτήτων από τα 35 στα 45 στιγμιότυπα. Στην περίπτωση αυτή, η χρονική αύξηση είναι μόνο 4-6 δευτερόλεπτα.



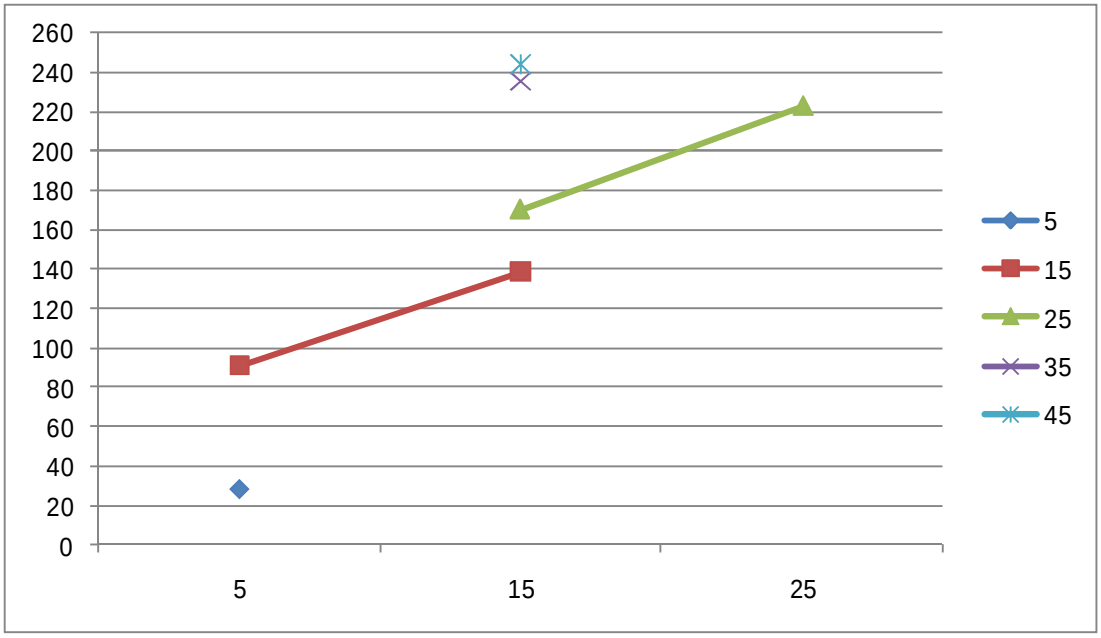
Σχήμα 5.9: Οντολογικό μοντέλο 25 κλάσεων και 45 ιδιοτήτων

Στο Σχήμα 5.9, εμφανίζονται οι μετρήσεις για το μοντέλο 25-45 ιδιοτήτων. Οι χρόνοι πλέον έχουν ξεπεράσει τους αντίστοιχους χρόνους του μοντέλου 15-45. Παρατηρούμε και εδώ ότι οι οντολογίες με 35 στιγμιότυπα ιδιοτήτων απέχουν χρονικά από την επέκταση τους σε 45 στιγμιότυπα ιδιοτήτων, πολύ λιγότερο από ότι από τις άλλες επεκτάσεις τους.



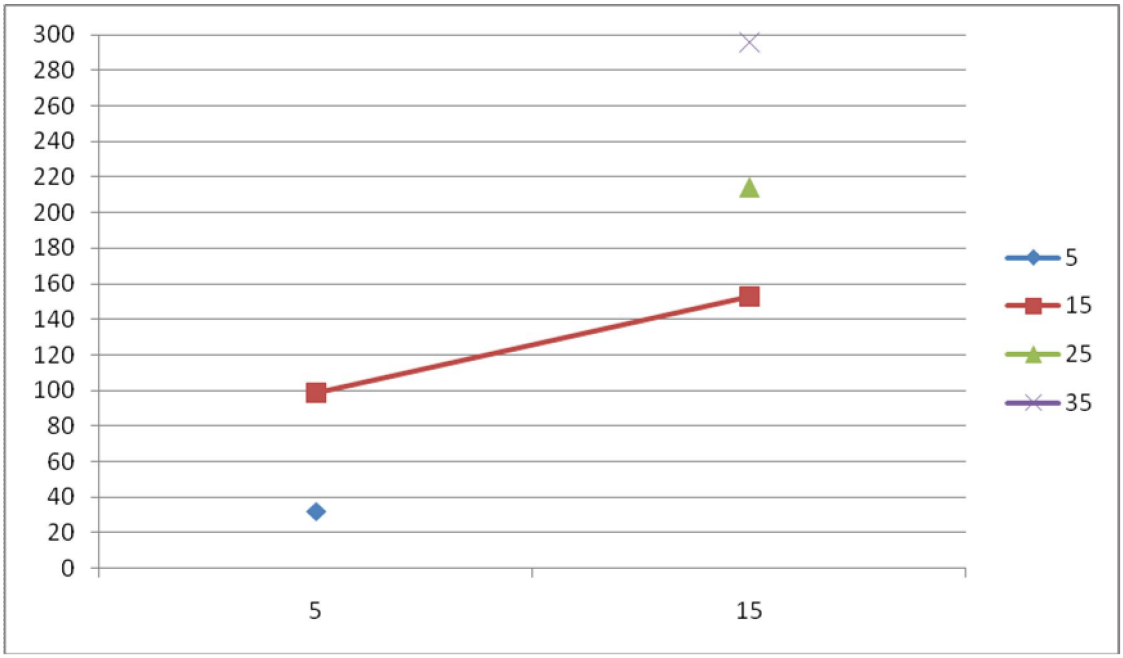
Σχήμα 5.10: Οντολογικό μοντέλο 25 κλάσεων και 55 ιδιοτήτων

Η επέκταση των οντολογιών συνεχίζει να περιορίζεται όπως φαίνεται στο Σχήμα 5.10 του μοντέλου 25-55. Το Σχήμα 5.11 παρουσιάζει το μοντέλο 25-65. Η πολυπλοκότητα των οντολογιών είναι αρκετά αυξημένη πια. Ο μικρότερος καταγεγραμμένος χρόνος είναι τα 28 δευτερόλεπτα, που αποτελεί περίπου το 10% του χρονικού ορίου των 5 λεπτών (300 δευτερόλεπτα) που έχουμε θέσει. Συνεπώς, η επέκταση των οντολογιών αυτού του μοντέλου δε μπορεί να είναι μεγάλη καθώς οι οντολογίες ξεπερνάν γρήγορα το χρονικό όριο.



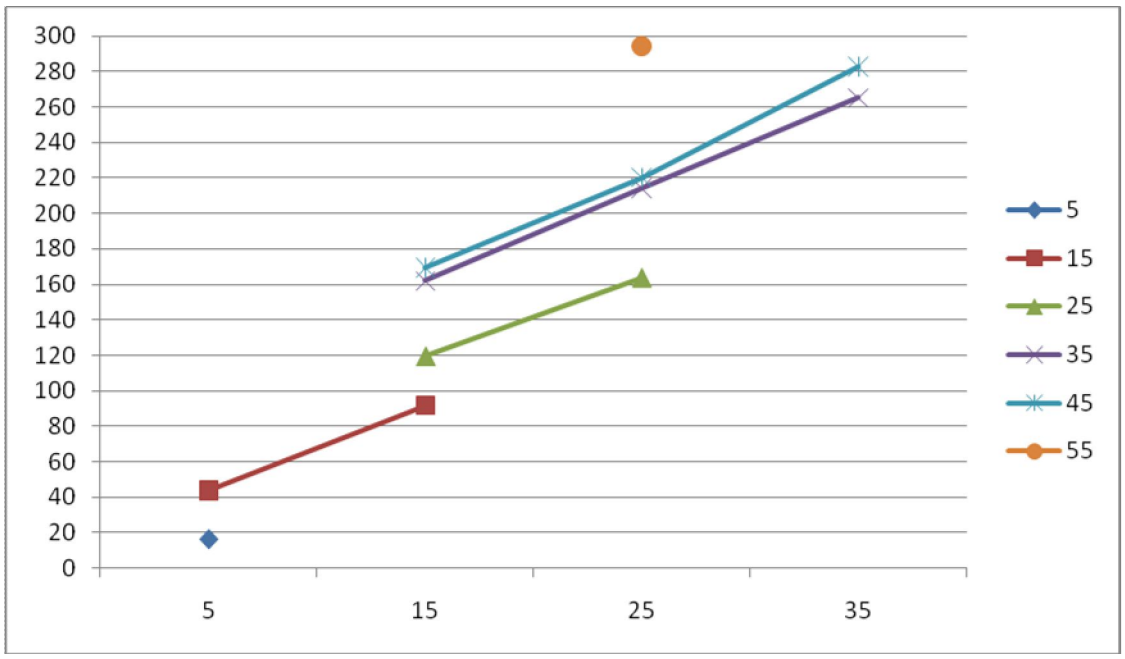
Σχήμα 5.11: Οντολογικό μοντέλο 25 κλάσεων και 65 ιδιοτήτων

.



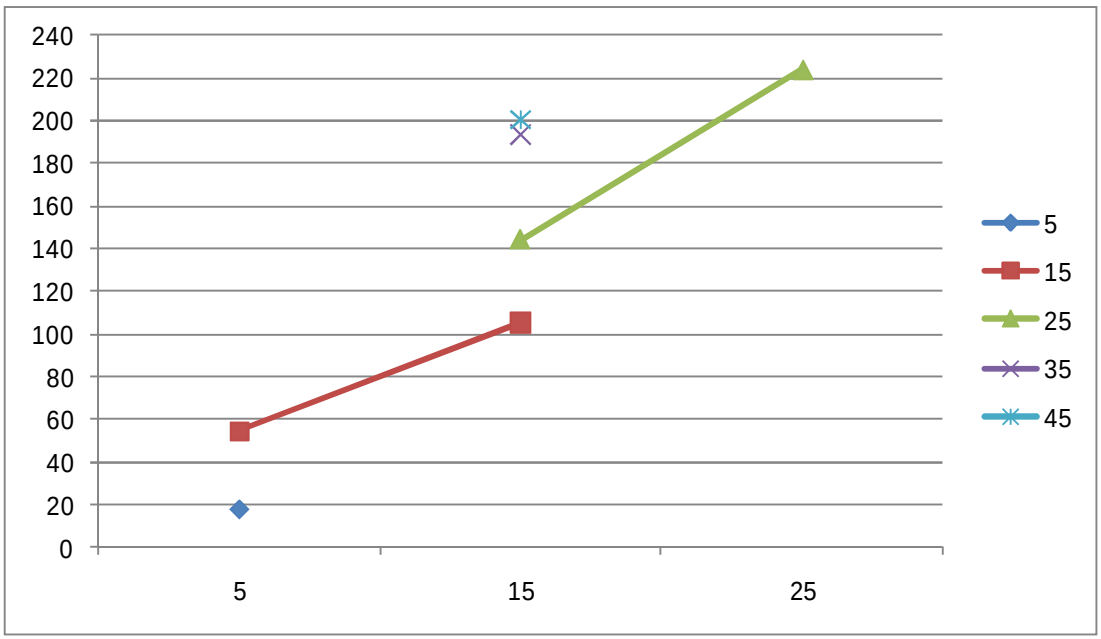
Σχήμα 5.12: Οντολογικό μοντέλο 25 κλάσεων και 75 ιδιοτήτων

Η πολυπλοκότητα του μοντέλου 25-75 είναι αρκετά μεγάλη, ώστε μπορούν να παραχθούν από αυτό μόνο 5 οντολογίες, για τις οποίες η συλλογιστική διαδικασία επιστρέφει τιμές εντός του επιτρεπτού χρονικού εύρους. Παρατηρούμε ότι καθώς αυξάνεται ο αριθμός των στιγμιότυπων ιδιοτήτων έχουμε και μεγάλη αύξηση του χρόνου συμπερασμού τους. Επιπλέον, το μοντέλο δε μπορεί να επεκταθεί στα 25 στιγμιότυπα κλάσεων, μιας και η πολυπλοκότητα του είναι τέτοια που η μνήμη του PDA δεν επαρκεί. Μέχρι στιγμής, ο κύριος λόγος τερματισμού της επέκτασης ενός μοντέλου είναι η υπέρβαση του χρονικού ορίου, αν και δε γνωρίζουμε αν όντως θα τερμάτιζαν οι διαδικασίες για αυτές τις οντολογίες ή αν τελικά θα παρουσίαζαν εξαίρεση ανεπαρκούς μνήμης, λόγω των υψηλών αναγκών τους σε πόρους μνήμης.



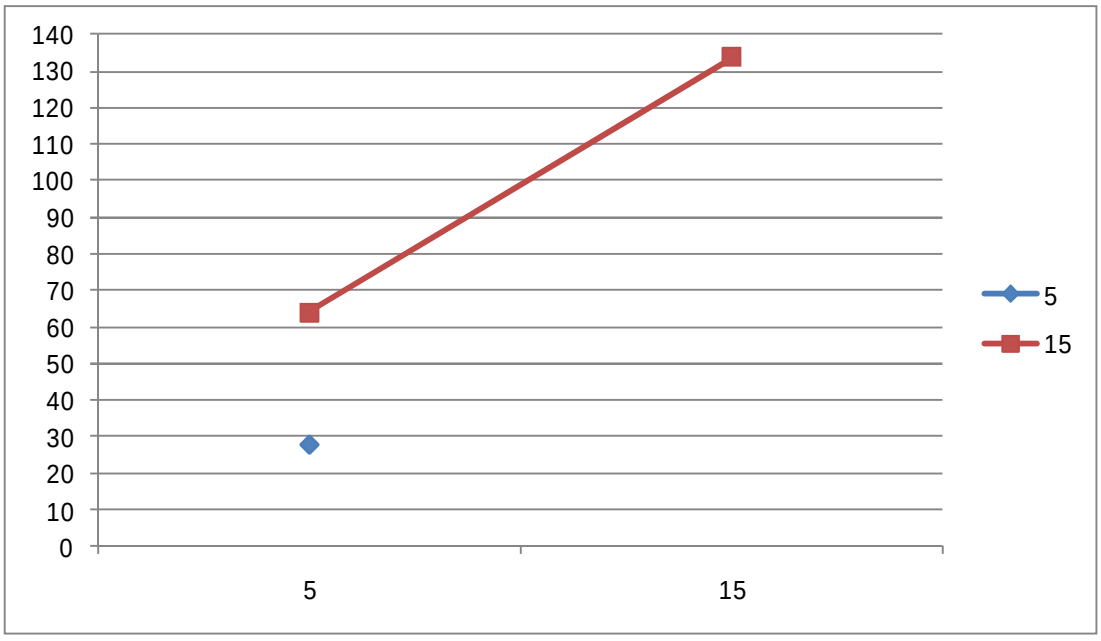
Σχήμα 5.13: Οντολογικό μοντέλο 35 κλάσεων και 35 ιδιοτήτων

Στο Σχήμα 5.13 παρουσιάζεται το μοντέλο 35-35. Όπως και στην περίπτωση του μοντέλου 25-25, βλέπουμε ότι οι χρόνοι μειώθηκαν και οι οντολογίες έχουν επεκταθεί περισσότερο σε σχέση με το προηγούμενο μοντέλο 25-75. Συμπεραίνουμε ότι η πολυπλοκότητα του 25-75 είναι αρκετά μεγαλύτερη ακόμα και αν το 35-35 έχει περισσότερες κλάσεις.



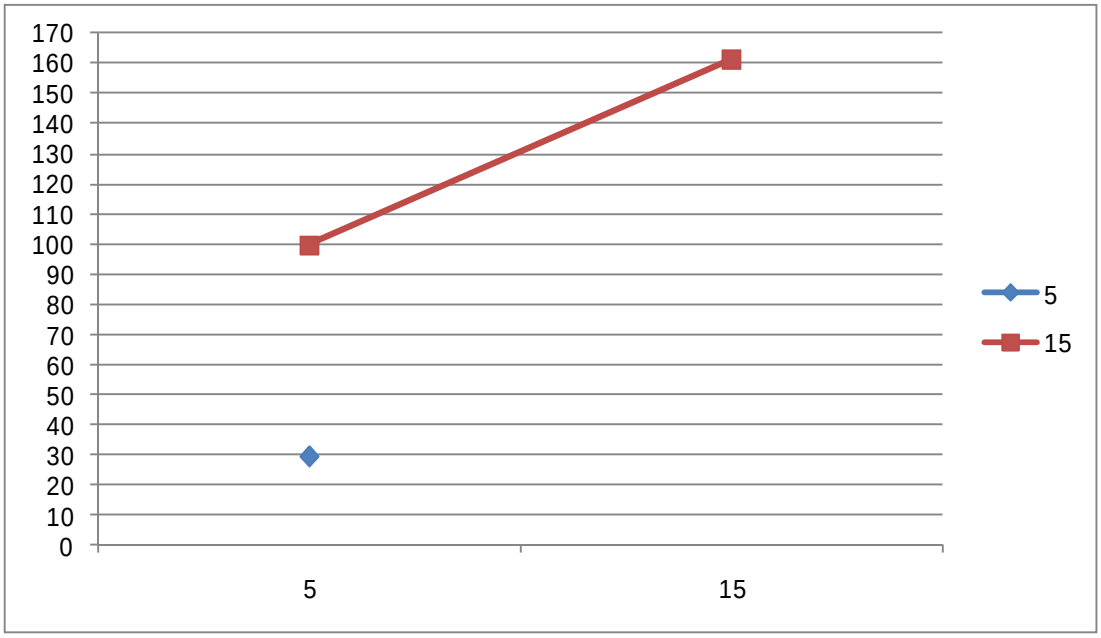
Σχήμα 5.14: Οντολογικό μοντέλο 35 κλάσεων και 45 ιδιοτήτων

Με την επέκταση του μοντέλου 35-35 κατά 10 ιδιότητες, είναι εμφανής μια μεγάλη μείωση στις οντολογίες που δημιουργήθηκαν από αυτό. Οι οντολογίες του μοντέλου 35-45 είναι σχεδόν υποδιπλάσιες από εκείνες του μοντέλου 35-35. Στο Σχήμα 5.14, βλέπουμε ότι το μοντέλο 35-45 έχει επεκταθεί σε 7 μόνο οντολογίες. Αυτό συμβαίνει γιατί οι περαιτέρω επεκτάσεις του μοντέλου παρουσίασαν πρόβλημα ανεπαρκούς μνήμης.



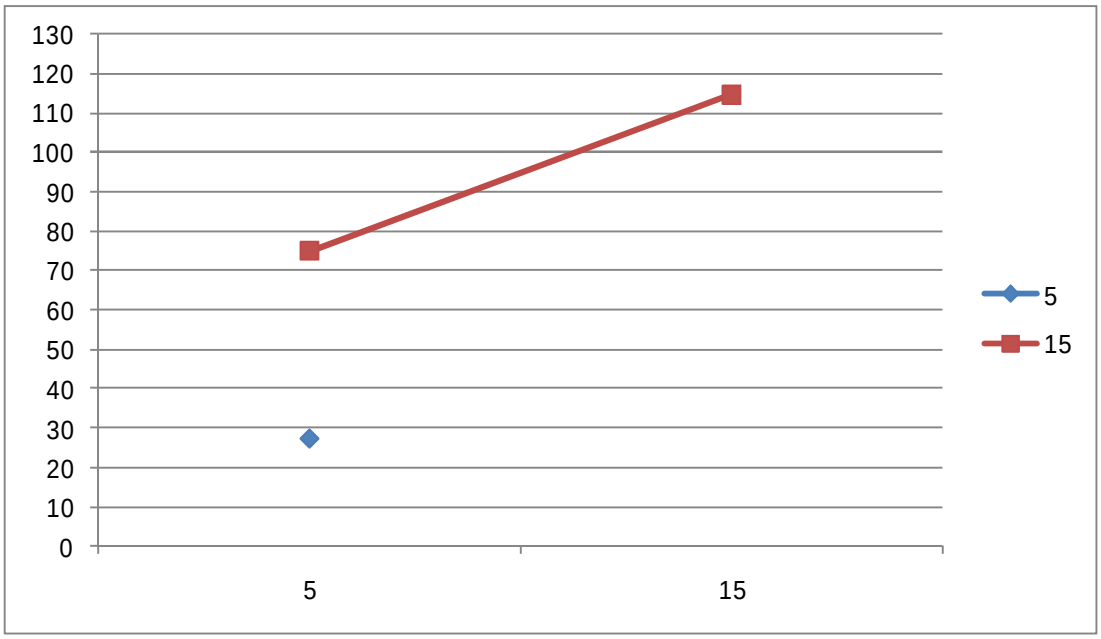
Σχήμα 5.15: Οντολογικό μοντέλο 35 κλάσεων και 55 ιδιοτήτων

Το Σχήμα 5.15 παρουσιάζει το μοντέλο 35-55. Η μνήμη που απαιτείται για τις οντολογίες αυτού του μοντέλου δεν επαρκεί και για αυτό παράγονται μόνο 3 οντολογίες. Ο χρόνος συμπερασμού της μεγαλύτερης οντολογίας του μοντέλου, δηλαδή της 35-15-55-15 είναι 134 δευτερόλεπτα. Η τιμή αυτή είναι μικρότερη από το μισό του χρονικού ορίου και έτσι θα περιμέναμε να εκτελεστεί επιτυχώς ο συμπερασμός στην αμέσως επόμενη οντολογία, δηλαδή την 35-15-55-25. Όμως, ο συμπερασμός σε αυτήν την οντολογία επιφέρει σφάλμα ανεπαρκούς μνήμης, καταδεικνύοντας το μέγεθος της πολυπλοκότητάς της.



Σχήμα 5.16: Οντολογικό μοντέλο 35 κλάσεων και 65 ιδιοτήτων

Το μοντέλο 35-65 παρουσιάζει την ίδια συμπεριφορά με το προηγούμενό του. Έτσι, στο Σχήμα 5.16 εμφανίζονται μόνο οι αντίστοιχες 3 οντολογίες. Η μόνη διαφορά είναι η αύξηση του χρόνου συμπερασμού κάθε οντολογίας. Συγκεκριμένα, οι οντολογίες του 35-65 έχουν αυξηθεί κατά 30 περίπου δευτερόλεπτα, σε σχέση με τις αντίστοιχες οντολογίες του μοντέλου 35-55.



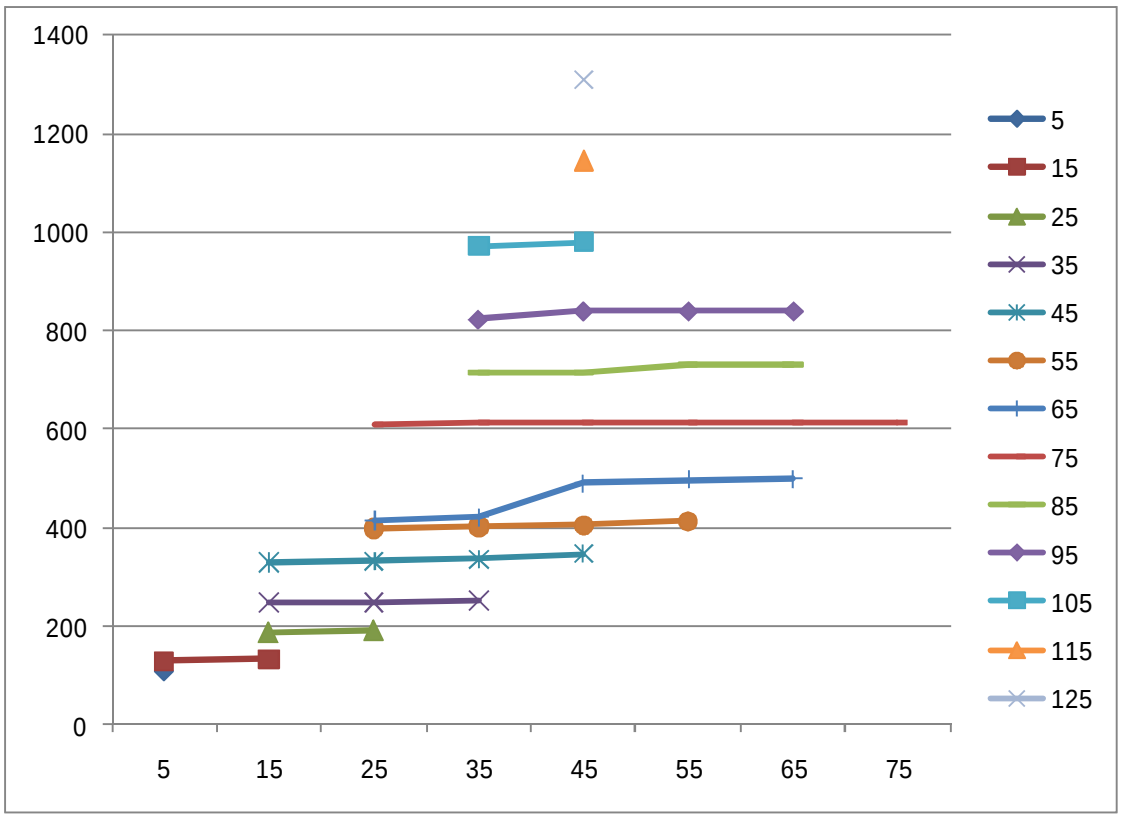
Σχήμα 5.17: Οντολογικό μοντέλο 45 κλάσεων και 45 ιδιοτήτων

Το Σχήμα 5.17 είναι το τελευταίο διάγραμμα αποτελεσμάτων για το Pocket KRHyper και παρουσιάζει το μοντέλο 45-45. Αν και οι χρόνοι έχουν μειωθεί σε σχέση με το προηγούμενο μοντέλο 35-65, προσεγγίζοντας τους χρόνους του μοντέλου 35-55 (το 45-45 έχει ελάχιστα μικρότερους χρόνους από το 35-55), παρατηρούμε ότι ακολουθείται το ίδιο μοτίβο με τα δυο προηγούμενα μοντέλα.

Τέλος, πρέπει να σημειωθεί ότι υπήρξαν και οντολογίες με περισσότερες κλάσεις και ιδιότητες, για τις οποίες το Pocket KRHyper επέστρεψε αποδεκτά αποτελέσματα. Ο λόγος που δεν παρουσιάζονται εδώ είναι ότι δε μπορούσαν να έχουν περισσότερα από 5 στιγμιότυπα κλάσεων και 5 στιγμιότυπα ιδιοτήτων, μιας και σε αυτήν την περίπτωση η υπάρχουσα μνήμη δεν ήταν επαρκής.

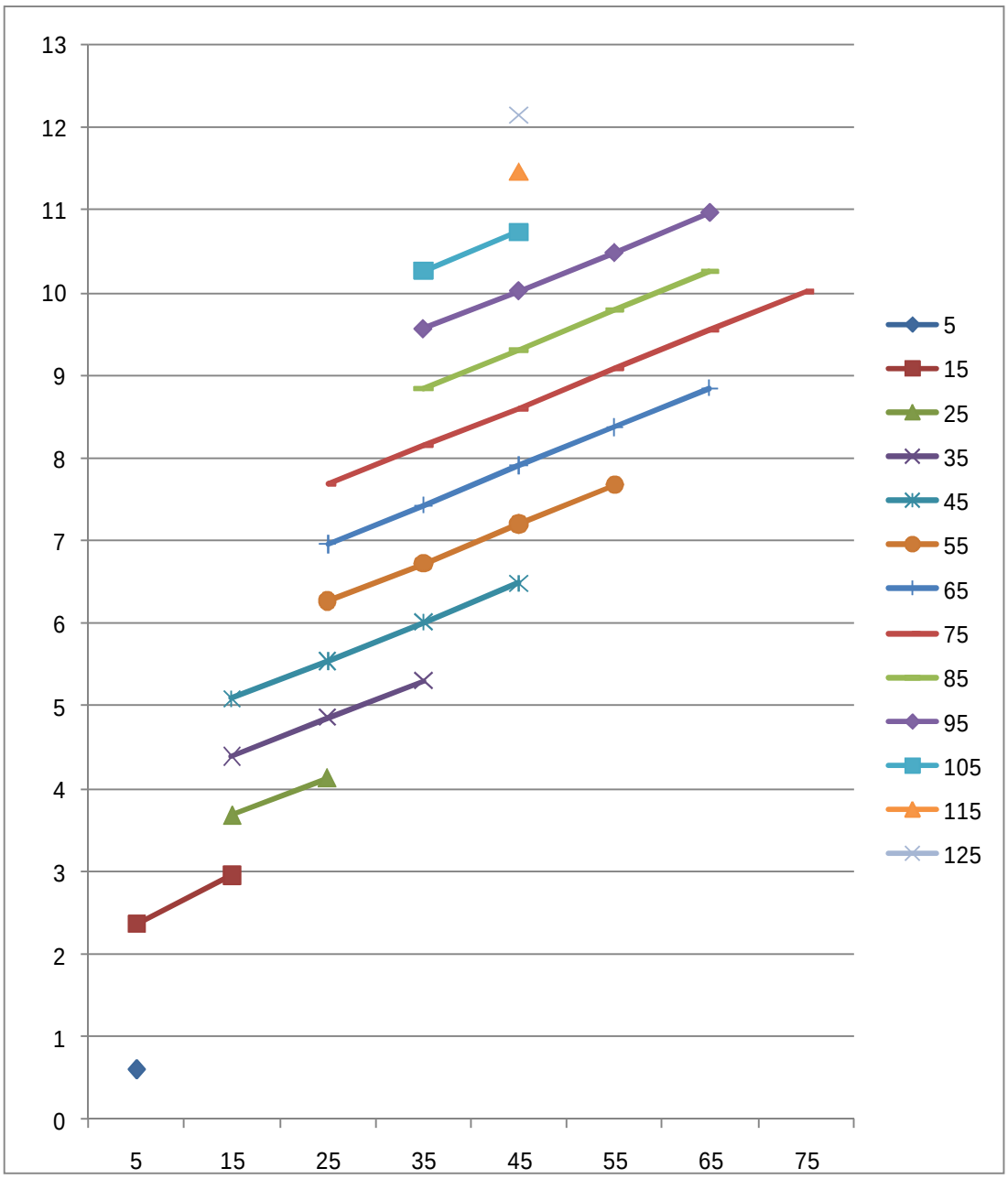
5.3.2 Αποτελέσματα πειραμάτων στο JIProlog

Στην περίπτωση του JIProlog εκτελέστηκαν δυο μετρήσεις σε κάθε συλλογιστική διαδικασία. Η πρώτη μέτρηση αφορά στο χρόνο φόρτωσης του αρχείου Prolog, ενώ η δεύτερη στο χρόνο που απαιτείται για την εύρεση όλων των λύσεων της επερώτησης «?- property0(X,Y).», που έχει χρησιμοποιηθεί σε κάθε συλλογιστική διαδικασία.



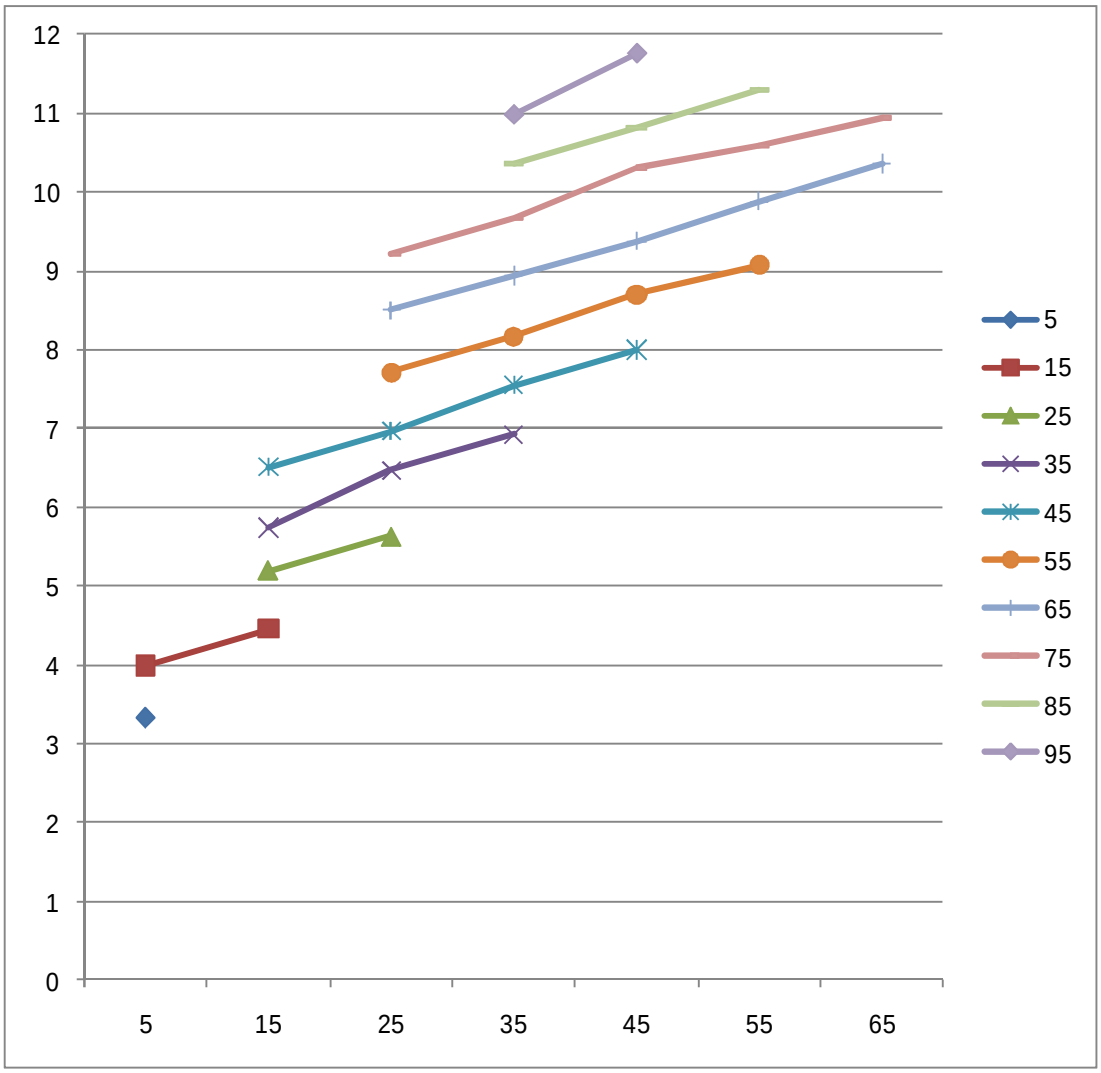
Σχήμα 5.18: Χρόνοι επερώτησης για κάθε οντολογικό μοντέλο

Στο Σχήμα 5.18 παρουσιάζονται οι χρόνοι επερώτησης σε ms. Η αύξηση του αριθμού των κλάσεων ή των ιδιοτήτων δεν έχει καμία επίδραση στον χρόνο επερώτησης. Έτσι, όλες οι οντολογίες που διαθέτουν ίδιο αριθμό στιγμιότυπων κλάσεων και ιδιοτήτων, δίνουν τον ίδιο χρόνο επερώτησης. Έτσι, ένα και μόνο διάγραμμα επαρκεί για την παρουσίαση των χρόνων επερώτησης κάθε οντολογίας. Οι χρόνοι επερώτησης είναι πολύ μικροί, αφού η εύρεση λύσεων σε μια επερώτηση απαιτεί πολύ λιγότερο χρόνο από την εύρεση μοντέλου για μια βάση γνώσης. Εφόσον η επερώτηση είναι πάντα η ίδια, ο χρόνος της επηρεάζεται μόνο από την αύξηση των στιγμιότυπων μιας οντολογίας. Έτσι, ο χρόνος αυξάνεται καθώς αυξάνονται τα στιγμιότυπα κλάσεων ή τα στιγμιότυπα ιδιοτήτων. Λόγω των μικρών διαφορών που έχουν τα αποτελέσματα μεταξύ τους, οι γραμμές στο διάγραμμα φαίνεται σχεδόν ευθείες. Παρατηρούμε, επίσης, ότι ο αριθμός των στιγμιότυπων ιδιοτήτων επηρεάζει περισσότερο τον χρόνο της επερώτησης. Αυτό συμβαίνει γιατί η επερώτηση που χρησιμοποιήθηκε ήταν για την ιδιότητα property0.



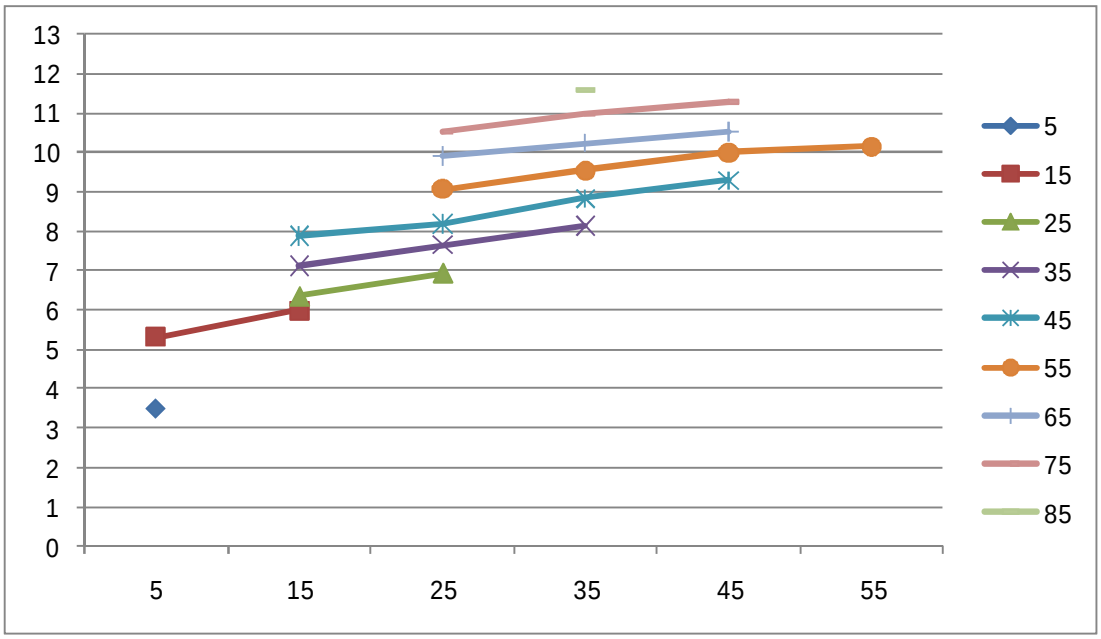
Σχήμα 5.19: Οντολογικό μοντέλο 5 κλάσεων και 5 ιδιοτήτων

Ο χρόνος φόρτωσης μια οντολογίας αυξάνεται καθώς η οντολογία μεγαλώνει, ακολουθώντας παρόμοια πορεία με τους χρόνους εύρεσης μοντέλου της μηχανής Pocket KRHyper. Η αύξηση αυτή, όπως φαίνεται στο Σχήμα 5.19, φαίνεται να είναι σχεδόν η ίδια για κάθε αύξηση των στιγμιότυπων κλάσεων και στιγμιότυπων ιδιοτήτων. Στο JIProlog δοκιμάστηκαν μόνο οι οντολογίες για τις οποίες το Pocket KRHyper επέστρεψε έγκυρο αποτέλεσμα. Η επέκταση των οντολογιών στην περίπτωση του JIProlog θα μπορούσε να είναι πολύ μεγαλύτερη, αφού οι χρόνοι φόρτωσης είναι τόσοι μικροί, αλλά κάτι τέτοιο δεν είναι αναγκαίο για τους σκοπούς της εργασίας.



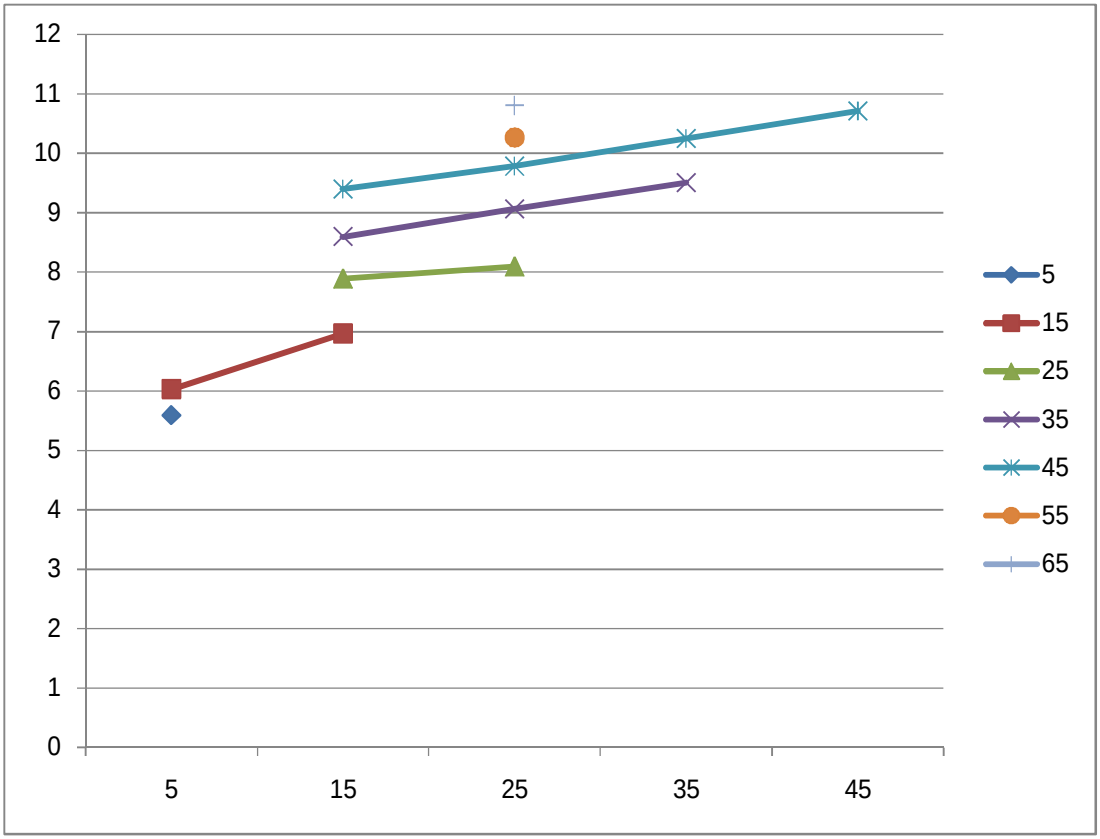
Σχήμα 5.20: Οντολογικό μοντέλο 5 κλάσεων και 15 ιδιοτήτων

Στο Σχήμα 5.20 του μοντέλου 5-15, βλέπουμε ότι υπάρχει μια αύξηση 1,5-2 δευτερολέπτων περίπου σε όλους τους χρόνους φόρτωσης των οντολογιών. Φαινομενικά, η αύξηση αυτή μοιάζει μικρή, αλλά λαμβάνοντας υπόψη το μέγεθος των χρόνων, είναι μια αξιοσημείωτη αύξηση. Ο λόγος είναι ο τριπλασιασμός των ιδιοτήτων του μοντέλου, των οποίων ο αριθμός επηρεάζει το πλήθος των προτάσεων Prolog στο αρχείο που θα χρησιμοποιηθεί και κατ' επέκταση και τον χρόνο φόρτωσης του αρχείου.



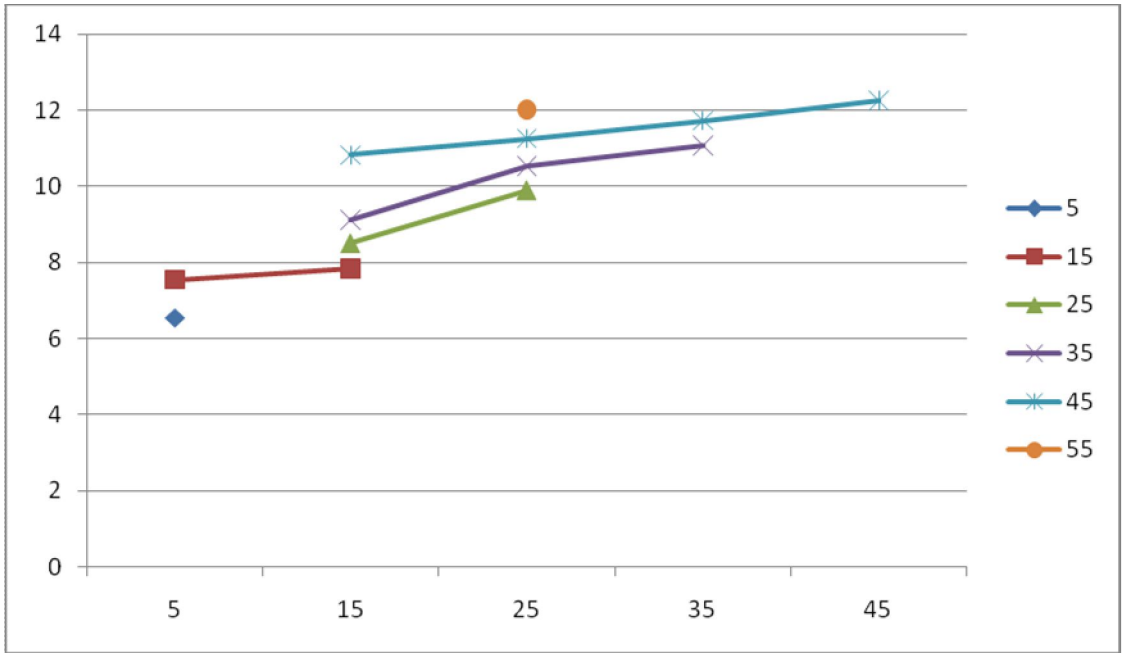
Σχήμα 5.21: Οντολογικό μοντέλο 15 κλάσεων και 15 ιδιοτήτων

Αυξάνοντας τις κλάσεις του προηγούμενου μοντέλου κατά 10, προκύπτει το μοντέλο 15-15. Όπως φαίνεται στο Σχήμα 5.21, ο χρόνος φόρτωσης κάθε οντολογίας έχει αυξηθεί κατά 1-1,5 δευτερόλεπτα. Φυσικά, η αύξηση αυτή είναι αναμενόμενη, αφού περισσότερες κλάσεις συνεπάγονται περισσότερες προτάσεις Prolog, μεγαλύτερο μέγεθος αρχείου και, συνεπώς, υψηλότερο χρόνο φόρτωσης. Εξαίρεση αποτελεί η οντολογία 15-5-15-5 για την οποία η χρονική αύξηση ήταν ελάχιστη.



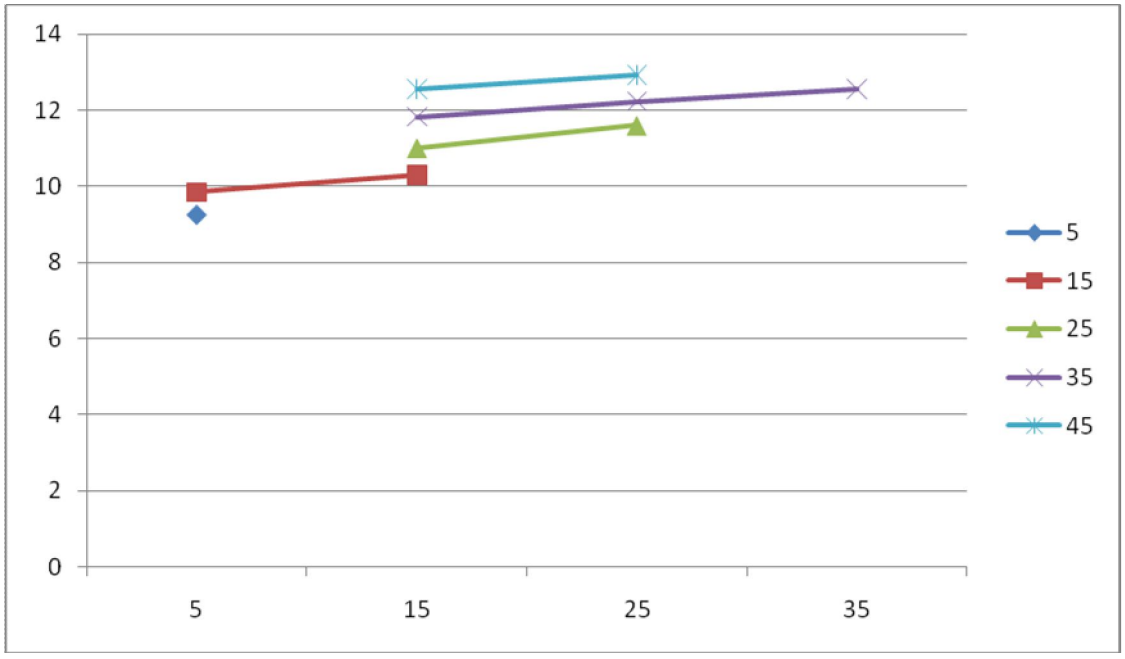
Σχήμα 5.22: Οντολογικό μοντέλο 15 κλάσεων και 25 ιδιοτήτων

Στο Σχήμα 5.22, παρουσιάζεται το μοντέλο 15-25. Στην περίπτωση αυτή παρατηρούμε ότι ο χρόνος της οντολογίες 15-5-25-5 αυξήθηκε κατά 2 δευτερόλεπτα, ενώ η αύξηση της αντίστοιχης οντολογίας του προηγούμενου μοντέλου ήταν μόλις 0,2 δευτερόλεπτα. Αντιθέτως, ο χρόνος της οντολογίας 15-5-25-15 είναι 0,5 δευτερόλεπτα, ενώ η αντίστοιχή της στο μοντέλο 15-15 εμφάνισε αύξηση 1,5 δευτερολέπτων. Οι χρόνοι φόρτωσης των υπολοίπων οντολογιών έχουν αυξηθεί κατά 1-1,5 δευτερόλεπτα.



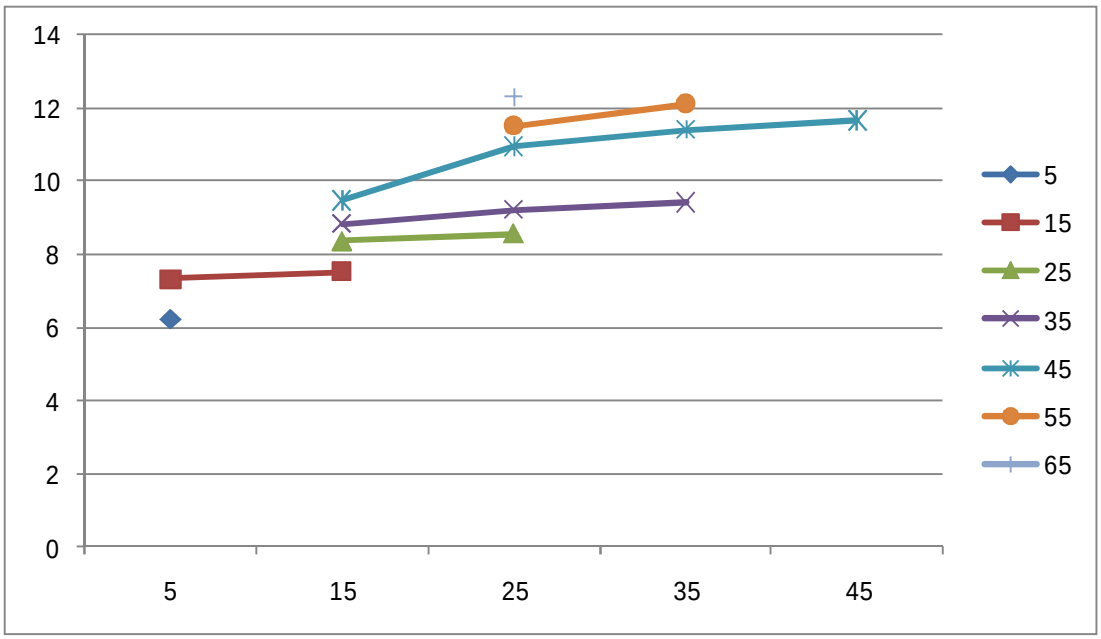
Σχήμα 5.23: Οντολογικό μοντέλο 15 κλάσεων και 35 ιδιοτήτων

Με την επέκταση του μοντέλου 15-35, δημιουργείται το μοντέλο 15-35, τα αποτελέσματα των οντολογιών του οποίου, εμφανίζονται στο Σχήμα 5.23. Η αύξηση εδώ κυμαίνεται ανάμεσα στα 1-2 δευτερόλεπτα περίπου. Οι χρόνοι φόρτωσης εξακολουθούν να είναι πολύ μικροί και η αύξησή τους δεν έχει υπερβεί, προς το παρόν, τα 2 δευτερόλεπτα.



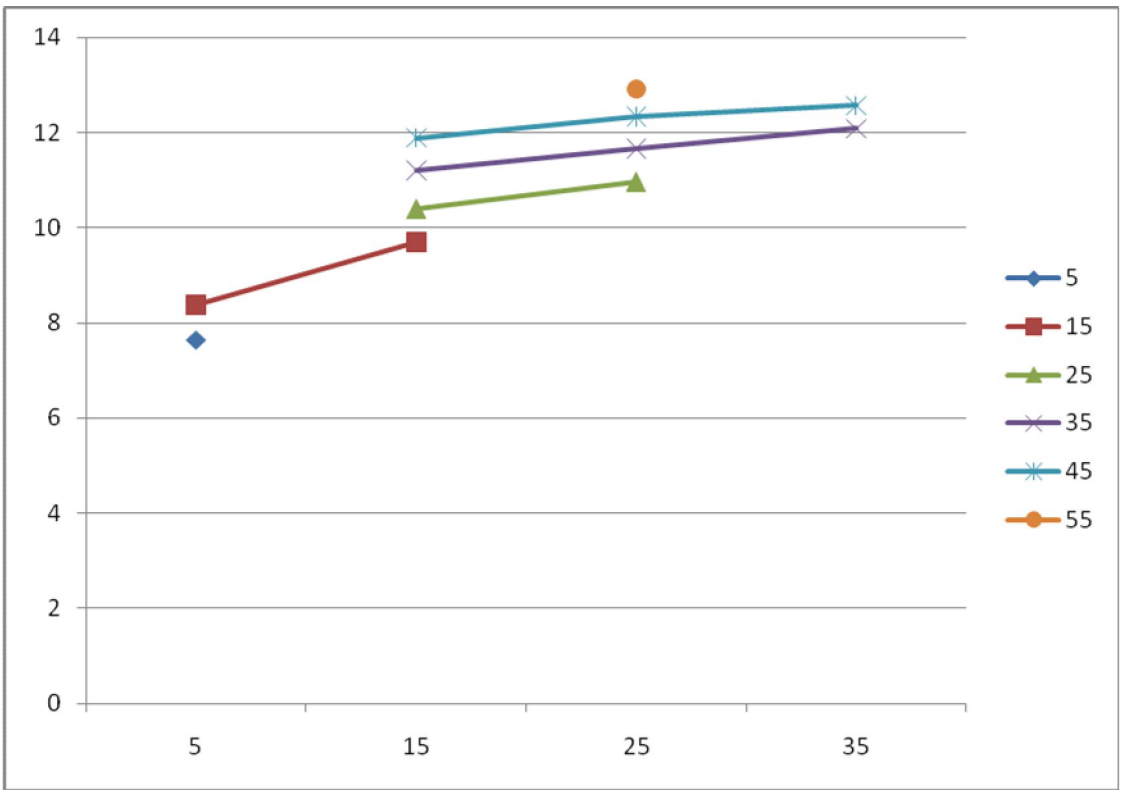
Σχήμα 5.24: Οντολογικό μοντέλο 15 κλάσεων και 45 ιδιοτήτων

Στο Σχήμα 5.24, παρατηρούμε μια μεγαλύτερη αύξηση των χρονικών αποτελεσμάτων. Το διάγραμμα αντιστοιχεί στο μοντέλο 15-45, του οποίου οι χρόνοι φόρτωσης των οντολογιών έχουν αυξηθεί 1,5-3 δευτερόλεπτα. Οι χρόνοι που είχαν αύξηση 3 δευτερολέπτων ανήκουν στις οντολογίες με 5 και 15 στιγμιότυπα κλάσεων, με εξαίρεση την οντολογία 15-15-45-45.

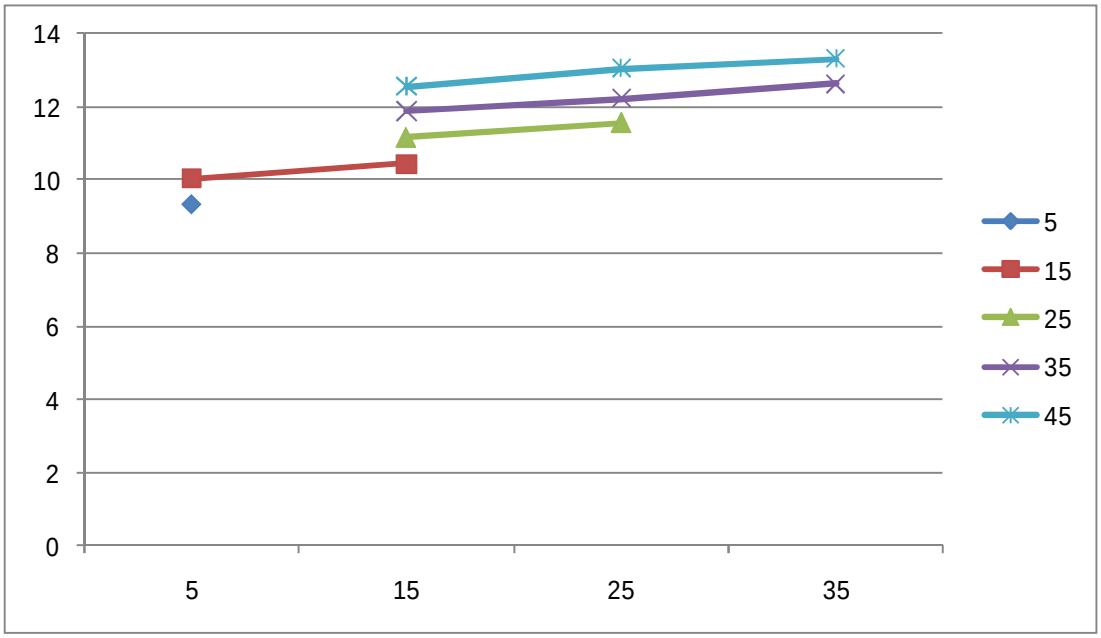


Σχήμα 5.25: Οντολογικό μοντέλο 25 κλάσεων και 25 ιδιοτήτων

Το μοντέλο 25-25 υλοποιείται προσθέτοντας 10 κλάσεις στο μοντέλο 15-25. Στο Σχήμα 5.25, παρατηρούμε ότι οι αυξήσεις των χρόνων του μοντέλου 25-25 σε σχέση με το 15-25, είναι 1-2 δευτερόλεπτα, με εξαίρεση την οντολογία 25-5-25-5 που έχει αυξηθεί κατά 3 περίπου δευτερόλεπτα. Παρόμοια αύξηση, 1-2,5 δευτερολέπτων, παρατηρείτε και στο Σχήμα 5.26 του μοντέλου 25-35. Οι χρόνοι εξακολουθούν να είναι πολύ μικρότεροι από τους αντίστοιχους στο Pocket KRHyper.



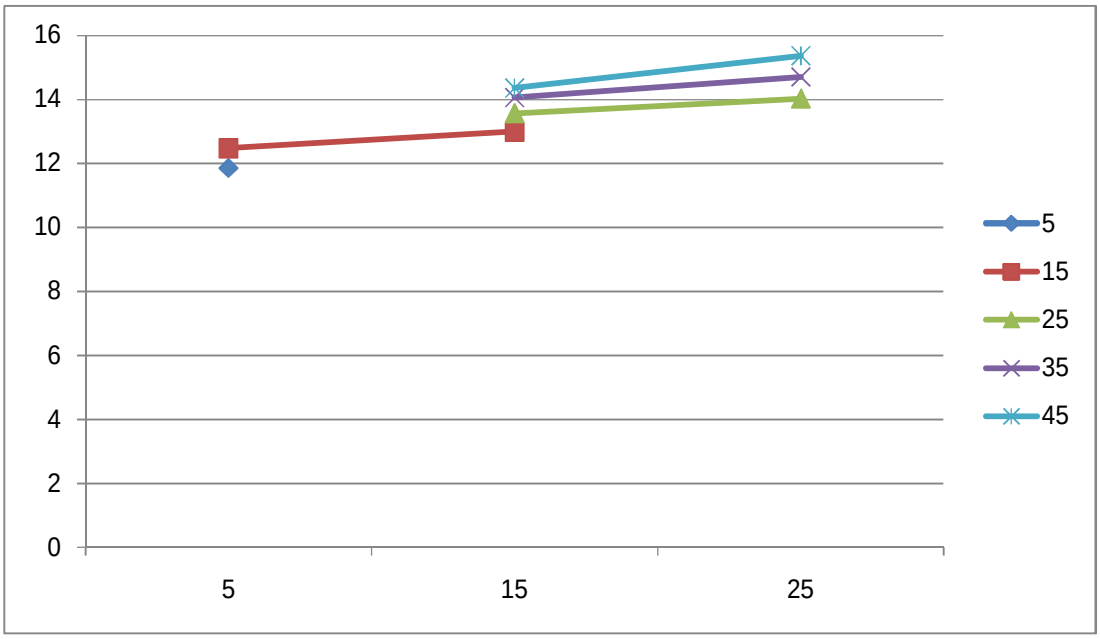
Σχήμα 5.26: Οντολογικό μοντέλο 25 κλάσεων και 35 ιδιοτήτων



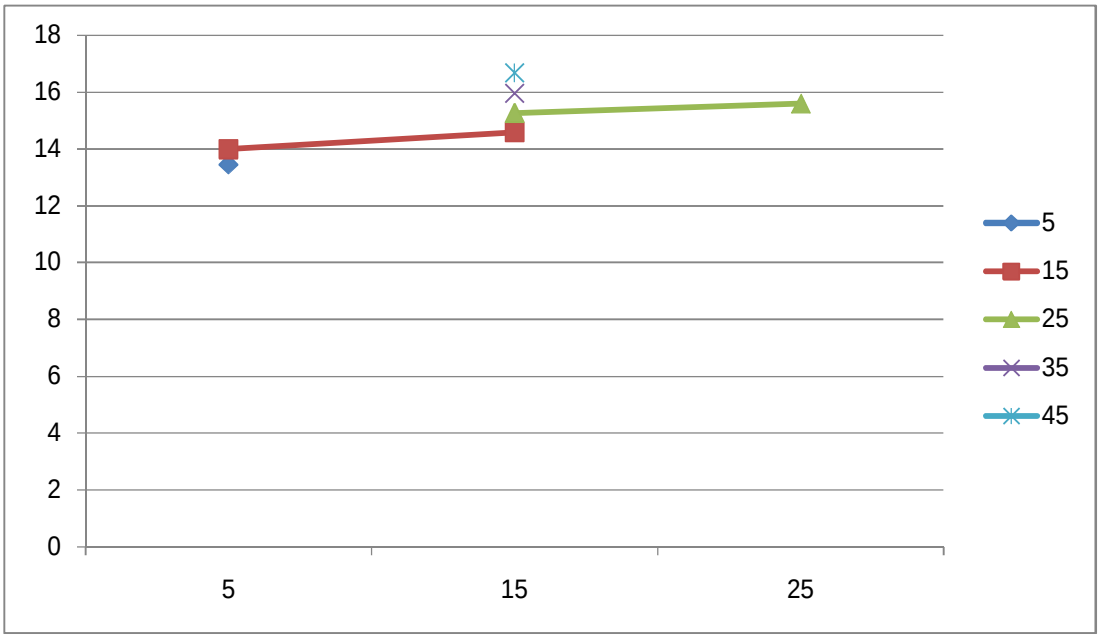
Σχήμα 5.27: Οντολογικό μοντέλο 25 κλάσεων και 45 ιδιοτήτων

Στο Σχήμα 5.27, οι χρόνοι φόρτωσης των οντολογιών του μοντέλου 25-45 έχουν υποστεί μια μικρή αύξηση 0,5-1 δευτερολέπτου, η οποία είναι σχεδόν υποδιπλάσια από του προηγούμενου μοντέλου. Ωστόσο, οι μετρήσεις στις οντολογίες 25-5-45-5 και 25-5-

45-15 είναι αυξημένες κατά 1,7 δευτερόλεπτα, ξεπερνώντας την αύξηση που εμφάνισαν οι αντίστοιχες οντολογίες στο μοντέλο 25-35. Αντιθέτως, το μοντέλο 25-55 παρουσιάζει αύξηση 2-2,5 δευτερολέπτων για όλες τις οντολογίες (Σχήμα 5.28).

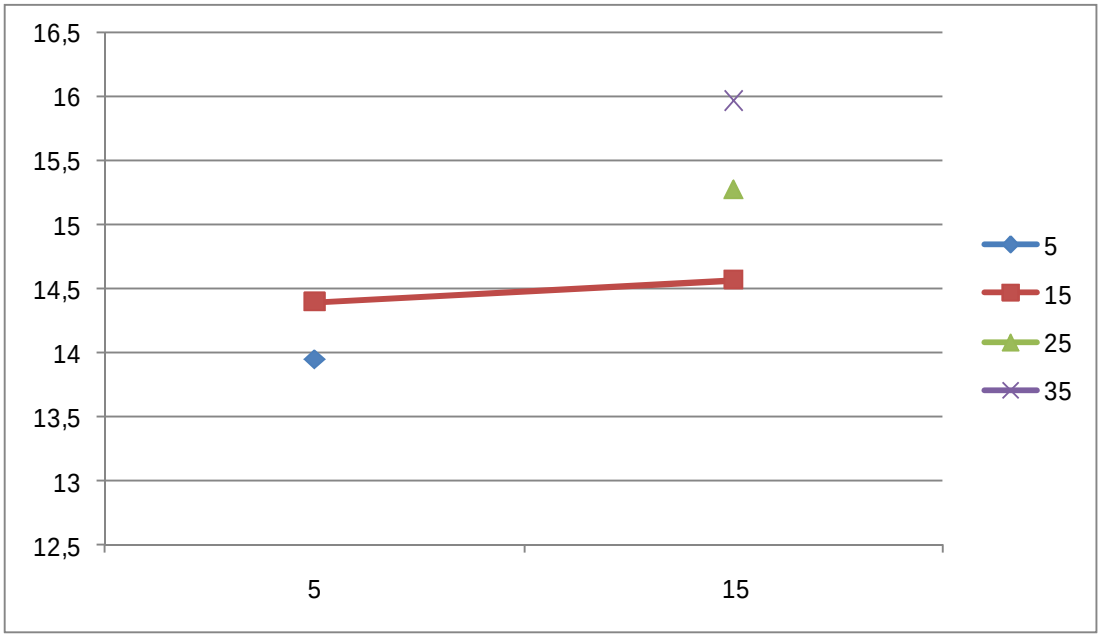


Σχήμα 5.28: Οντολογικό μοντέλο 25 κλάσεων και 55 ιδιοτήτων

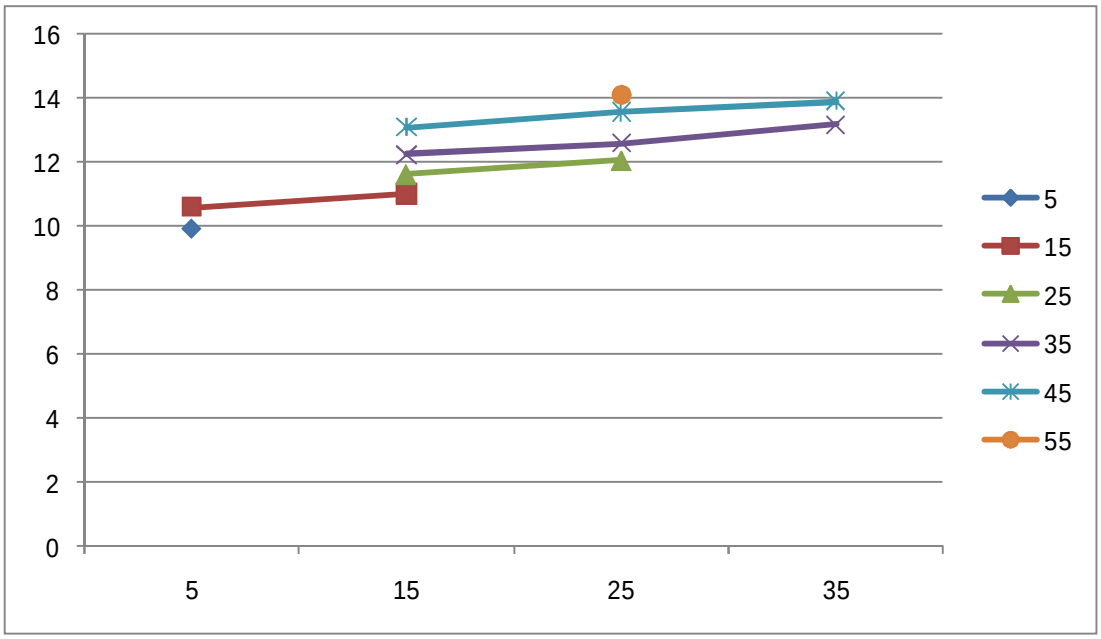


Σχήμα 5.29: Οντολογικό μοντέλο 25 κλάσεων και 65 ιδιοτήτων

Στο Σχήμα 5.29 εμφανίζονται τα αποτελέσματα των μετρήσεων για τις οντολογίες του μοντέλου 25-65. Εδώ η χρονική αύξηση, έναντι του προηγούμενου μοντέλου, κυμαίνεται στα 1,5-2 δευτερόλεπτα. Πρέπει, επίσης, να σημειωθεί ότι η αύξηση του χρόνου καθώς αυξάνονται τα στιγμιότυπα ιδιοτήτων με σταθερό τον αριθμό των στιγμιότυπων κλάσεων, φαίνεται να παραμένει σταθερή για όλα τα μοντέλα και κυμαίνεται στα 0,5-0,7 δευτερόλεπτα. Το τελευταίο μοντέλο με 25 κλάσεις είναι το μοντέλο 25-75 και οι μετρήσεις του εμφανίζονται στο Σχήμα 5.30. όπως και στην περίπτωση του μοντέλου 25-45, η αύξηση των χρόνων φόρτωσης των οντολογιών είναι ιδιαίτερα μικρή (0,1-0,5 δευτερόλεπτα).

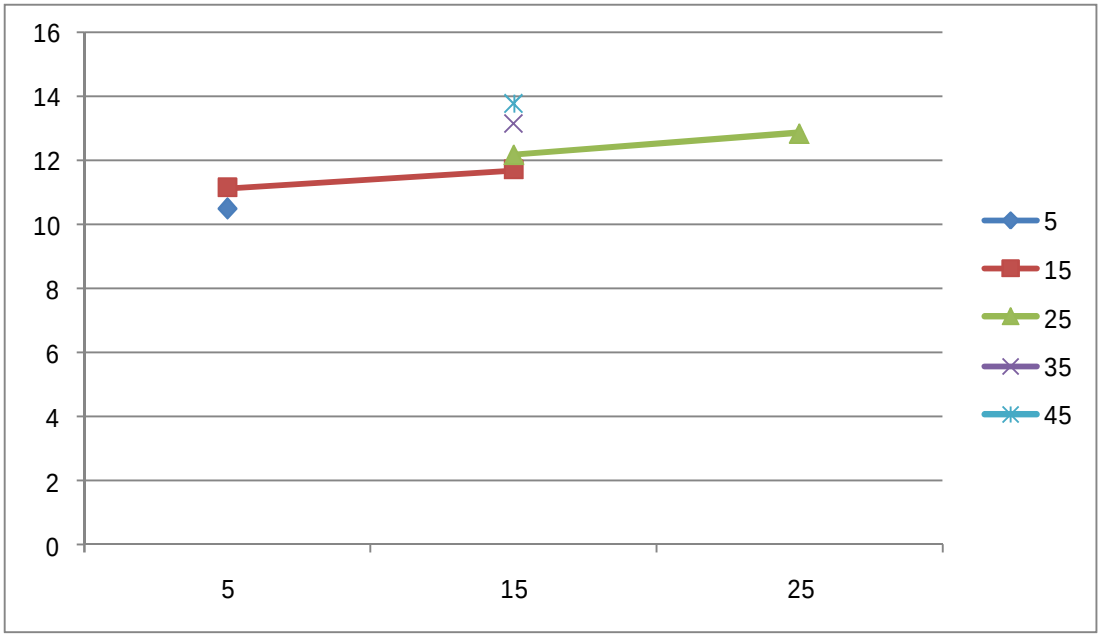


Σχήμα 5.30: Οντολογικό μοντέλο 25 κλάσεων και 75 ιδιοτήτων

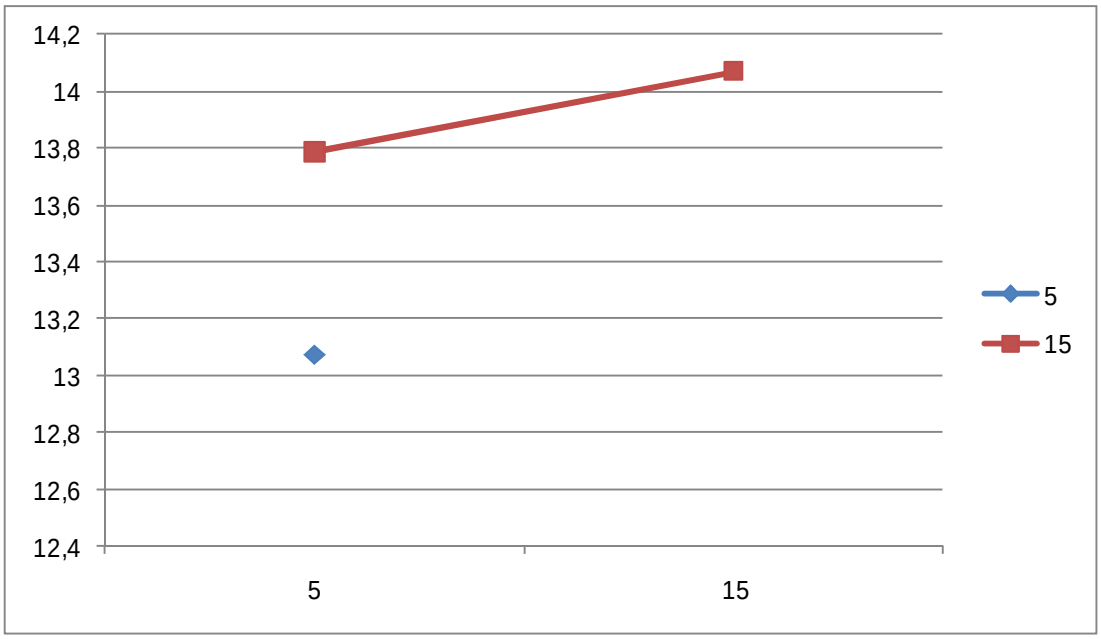


Σχήμα 5.31: Οντολογικό μοντέλο 35 κλάσεων και 35 ιδιοτήτων

Με την επέκταση του αριθμού των κλάσεων στο μοντέλο 25-35 προκύπτει το μοντέλο 35-35 που παρουσιάζεται στο παραπάνω σχήμα. Οι χρόνοι φόρτωσης των οντολογιών του 35-35 έχουν αυξηθεί κατά 1-1,5 δευτερόλεπτα. Εξαίρεση αποτελούν οι οντολογίες 35-5-35-5 και 35-5-35-15 των οποίων οι χρόνοι φόρτωσης αυξήθηκαν κατά 2,3 δευτερόλεπτα. Εν συνεχεία, επεκτείνοντας τον αριθμό των ιδιοτήτων του μοντέλου προκύπτει το μοντέλο 35-45. Η χρονική αύξηση σε αυτήν την περίπτωση είναι μόνο 0,5-1 δευτερόλεπτο.

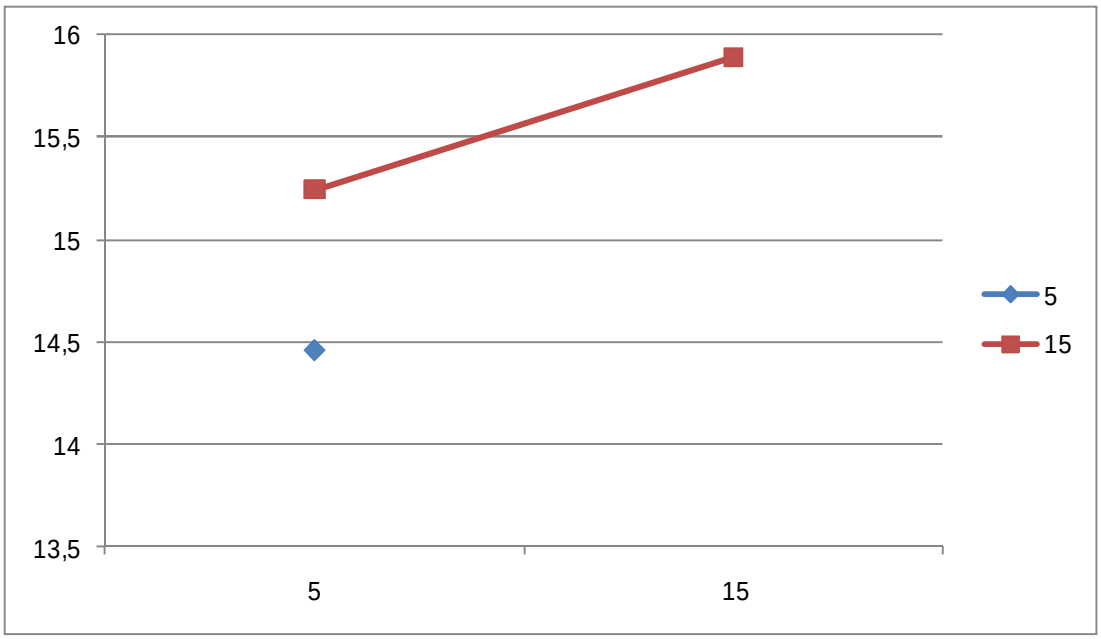


Σχήμα 5.32: Οντολογικό μοντέλο 35 κλάσεων και 45 ιδιοτήτων

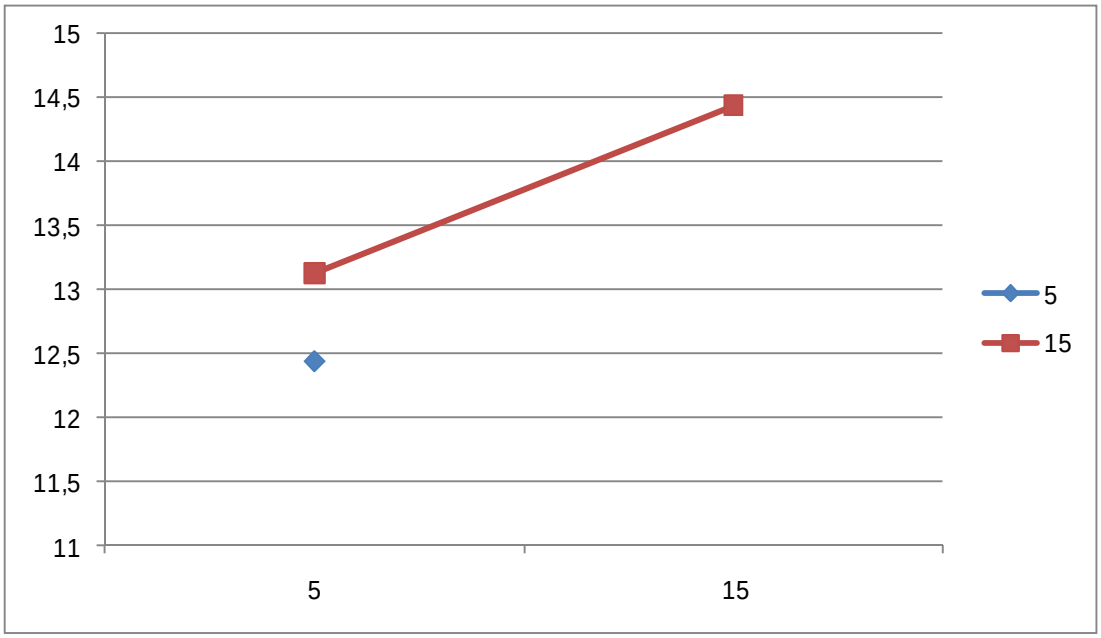


Σχήμα 5.33: Οντολογικό μοντέλο 35 κλάσεων και 55 ιδιοτήτων

Το Σχήμα 5.33 παρουσιάζει το μοντέλο 35-55. Εδώ η χρονική αύξηση είναι μεγαλύτερη. Οι χρόνοι φόρτωσης οντολογιών του 35-55 είναι μεγαλύτεροι κατά 2,5 δευτερόλεπτα, σε σχέση με τους αντίστοιχους χρόνους του μοντέλου 35-45. Αντιθέτως, όταν επεκτείνουμε το μοντέλο κατά 10 ιδιότητες η χρονική αύξηση είναι μικρότερη και κυμαίνεται στα 1,5-2 δευτερόλεπτα (βλ. Σχήμα 5.34).



Σχήμα 5.34: Οντολογικό μοντέλο 35 κλάσεων και 65 ιδιοτήτων



Σχήμα 5.35: Οντολογικό μοντέλο 45 κλάσεων και 45 ιδιοτήτων

Το τελευταίο σχήμα παρουσιάζει τους χρόνους φόρτωσης των οντολογιών του μοντέλου 45-45. Υπενθυμίζουμε ότι οι οντολογίες που δοκιμάστηκαν στο JIProlog είναι μόνο όσες έδωσαν (έγκυρα) αποτελέσματα στις μετρήσεις με το Pocket KRHyper. Οι οντολογίες θα μπορούσαν να είχαν επεκταθεί πολύ περισσότερο για το JIProlog, αφού οι χρόνοι είναι τόσο μικροί. Το μοντέλο 45-45 εμφανίζει αύξηση στους χρόνους φόρτωσης 2-3 δευτερολέπτων, έναντι του προγενέστερου μοντέλου 35-45, από το οποίο και προέκυψε. Οι χρονικές αυξήσεις μεταξύ οντολογιών του ίδιου οντολογικού μοντέλου, με τον ίδιο αριθμό στιγμιότυπων ιδιοτήτων, κυμάνθηκε στα 0,5-0,7 δευτερόλεπτα για όλα τα μοντέλα.

Είναι εμφανές ότι οι μετρήσεις του JIProlog είναι σημαντικά μικρότερες από τις μετρήσεις του Pocket KRHyper για κάθε οντολογία. Ωστόσο, το γεγονός αυτό δε σημαίνει ότι η Prolog αποτελεί αποδοτικότερη αναπαράσταση γνώσης για κινητά περιβάλλοντα. Η μετατροπή των οντολογιών σε Prolog είχε ως συνέπεια την αφαίρεση πολλών στοιχείων της οντολογίας, ώστε το αποτέλεσμα αυτής της αφαίρεσης να μπορεί να υποστηριχθεί από τις εκφραστικές δυνατότητες της Prolog. Με αλλά λόγια, ένα κομμάτι της αρχικής γνώσης «χάθηκε», οι οντολογίες έγιναν απλούστερες και, συνεπώς, η πολυπλοκότητά τους μειώθηκε. Επιπλέον, η μέθοδος συμπερασμού της Prolog δεν ελέγχει κάθε πρόταση στη βάση γνώσης, αλλά μόνο εκείνες που σχετίζονται με την επερώτηση. Τα παραπάνω αποτελούν τους λόγους της μεγαλύτερης αποδοτικότητας χρόνου και μνήμης του JIProlog, έναντι του Pocket KRHyper.

Η λογική πρώτης τάξης προσφέρει μικρότερη απώλεια γνώσης, μιας και είναι πιο εκφραστική από την Prolog. Ωστόσο, παρατηρούμε ότι σε μια κινητή συσκευή με περιορισμένους πόρους, όπως αυτή που χρησιμοποιήθηκε στην παρούσα εργασία, η συλλογιστική διαδικασία της λογικής πρώτης τάξης δε μπορεί να αντιμετωπίσει μεγάλες και πολύπλοκες οντολογίες.

Μια σημαντική παρατήρηση για αυτούς τους φορμαλισμούς είναι πως δεν μπορούμε να πούμε ότι ο ένας είναι καλύτερος από τον άλλο. Για παράδειγμα, η Prolog αποτελεί καλύτερη επιλογή για εφαρμογές πραγματικού χρόνου, όπου ο χρόνος απόκρισης απαιτείται να περιορίζεται σε μερικά δευτερόλεπτα. Αντίθετα, για διαδικτυακές

εφαρμογές (π.χ. σε επίπεδο εξυπηρετητή – server side), όπου η διαχείριση γίνεται πιο κεντρικοποιημένα μπορούμε να υιοθετήσουμε κάποια ισχυρότερη αλλά πιο αργή γλώσσα. Έτσι, η επιλογή γλώσσας αναπαράστασης γνώσης χαρακτηρίζεται από ένα θεμελιώδες πρόβλημα: η γλώσσα πρέπει να είναι αρκετά εκφραστική ώστε να εκπροσωπήσει τα σημαντικά αντικείμενα και τις σχέσεις του πεδίου ενδιαφέροντος και να επιτρέπει ένα αποτελεσματικό τρόπο συλλογιστικής και απάντησης σε ερωτήσεις σχετικά με έμμεσα δεδομένα της βάσης γνώσης σε εύλογο χρονικό διάστημα. Η αύξηση της εκφραστικότητας μίας γλώσσας είναι δυνατό να επιφέρει δυσάρεστες συνέπειες σε ότι αφορά τη δυνατότητα ανάπτυξης αποδοτικών αλγορίθμων συμπερασμού για τη γλώσσα (expressiveness-tractability tradeoff).

Συμπεραίνουμε, λοιπόν, πως η επιλογή της κατάλληλης γλώσσας αναπαράστασης γνώσης εξαρτάται από τις δυνατότητες της συσκευής στην οποία θα χρησιμοποιηθεί (μνήμη, ταχύτητα επεξεργαστή), τις απαιτήσεις των εφαρμογών σε χρόνο απόκρισης και σε εκφραστικότητα για τη μοντελοποίηση του πεδίου ενδιαφέροντος, καθώς και από τις υπάρχουσες μηχανές συμπερασμού που είναι συμβατές με την εκάστοτε πλατφόρμα.

6. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η ραγδαία εξέλιξη της τεχνολογίας πληροφοριών έχει καταστήσει διαθέσιμο μεγάλο όγκο πληροφοριών και υπηρεσιών. Έτσι, δημιουργήθηκε η ανάγκη για «έξυπνες» εφαρμογές που θα μπορούν να παίρνουν αποφάσεις, λαμβάνοντας υπόψη τις ανάγκες και τις απαιτήσεις του εκάστοτε χρήστη. Για να γίνει δυνατή αυτή η προσθήκη ευφυΐας, χρειάζεται ένας τρόπος αναπαράστασης των προαναφερθέντων στοιχείων, ώστε να υλοποιηθεί μια βάση γνώσης που θα είναι κατανοητή σε μηχανές, και μια μηχανή συμπερασμού για την εξαγωγή νέας γνώσης από την ήδη υπάρχουσα. Η πλέον διαδομένη μορφή αναπαράσταση γνώσης για αυτόν τον σκοπό, είναι οι οντολογίες. Ωστόσο, ένα κινητό περιβάλλον χαρακτηρίζεται από περιορισμένους πόρους και έτσι δε μπορεί να υποστηρίξει την υψηλή πολυπλοκότητα των οντολογιών. Συνεπώς, η μετατροπή των οντολογιών σε άλλη αναπαράσταση γνώσης καθίσταται αναγκαία.

Στα πλαίσια της παρούσας εργασίας, έχοντας μελετήσει τις διάφορες μεθόδους αναπαράστασης της γνώσης και τις αντίστοιχες συλλογιστικές διαδικασίες, αναπτύχθηκε ένα περιβάλλον δοκιμών, με σκοπό την παρουσίαση και σύγκριση των αναπαραστάσεων γνώσης, οι οποίες καθιστούν δυνατό τον συμπερασμό σε κινητές συσκευές με περιορισμένους πόρους. Οι οντολογίες που υλοποιήθηκαν για αυτές τις δοκιμές, μετατράπηκαν σε προτάσεις λογικής πρώτης τάξης και στη γλώσσα λογικού προγραμματισμού Prolog. Οι μηχανές συμπερασμού που χρησιμοποιήθηκαν είναι τα Pocket KRHyper και JIProlog αντίστοιχα.

Η εν γένει επιστημονική συνεισφορά της εργασίας μπορεί να συνοψιστεί στα ακόλουθα:

- *Αξιολόγηση τεχνολογιών γνώσης.* Αρχικά, παρουσιάστηκαν οι διαφορετικές μορφές αναπαράστασης γνώσης και τα χαρακτηριστικά τους. Στη συνέχεια, για καθεμία από τις μορφές αυτές, αναφέρθηκαν οι μέθοδοι εξαγωγής συμπερασμάτων. Έτσι, διαφαίνονται οι κατάλληλες αναπαραστάσεις γνώσης για την επίτευξη τις συλλογιστικής σε κινητά περιβάλλοντα.
- *Μετατροπή OWL οντολογιών σε λογική πρώτης τάξης και Prolog.* Η ανάγκη για συλλογιστική σε περιβάλλοντα κινητού υπολογισμού, συνεπάγεται την ανάγκη μετατροπής της μορφής αναπαράστασης γνώσης που χρησιμοποιείται ευρέως στο

Σημασιολογικό Ιστό και τα στατικά περιβάλλοντα, δηλαδή των οντολογιών, σε μορφή που μπορεί να υποστηριχθεί από κινητές συσκευές. Στο κεφάλαιο 4 παρουσιάστηκαν κάποια εργαλεία μετατροπής OWL οντολογιών σε Prolog. Επιπλέον, δημιουργήθηκε κώδικας μετατροπής οντολογιών σε λογική πρώτης τάξης και Prolog, για τις ανάγκες του περιβάλλοντος δοκιμών της παρούσας εργασίας.

- *Σύγκριση συστημάτων συμπερασμού σε περιβάλλον με περιορισμένους πόρους.* Παρουσιάστηκαν οι μετρήσεις για τις συλλογιστικές διαδικασίες δυο μηχανών συμπερασμού, που υποστηρίζουν διαφορετικές μορφές αναπαράστασης γνώσης, σε κινητή συσκευή. Με τον τρόπο αυτό, αναδεικνύονται οι δυνατότητες συμπερασμού σε περιβάλλον με περιορισμένους πόρους και τα αποτελέσματα της μετατροπής της γνώσης από μια μορφή σε μια άλλη.

ΠΙΝΑΚΑΣ ΟΡΟΛΟΓΙΑΣ

Ξενόγλωσσος όρος	**Ελληνικός Όρος**
Pervasive	Διάχυτος
Ubiquitous	Πανταχού παρών
Knowledge representation	Αναπαράσταση γνώσης
Reasoning	Συλλογιστική
Inference	Συμπερασμός
Description Logics	Περιγραφικές Λογικές
First order logic	Λογική πρώτης τάξης
Semantic Web	Σημασιολογικός Ιστός
Predicate	Κατηγόρημα
Head	Κεφαλή (κανόνα)
Body	Σώμα (κανόνα)
Forward chaining	Προς τα εμπρός αλυσίδα εκτέλεσης
Backward chaining	Προς τα πίσω αλυσίδα εκτέλεσης
Semantic network	Σημασιολογικό δίκτυο
Query	Επερώτηση
Reasoner	Μηχανή συμπερασμού
Ontology	Οντολογία
Property	Ιδιότητα
Instance	Στιγμιότυπο
Class	Κλάση
Scalability	Κλιμάκωση

ΣΥΝΤΜΗΣΕΙΣ – ΑΡΚΤΙΚΟΛΕΞΑ – ΑΚΡΩΝΥΜΙΑ

OWL	Web Ontology Language
FOL	First Order Logic
RDF	Resource Description Framework
DL	Description Logics
LP	Logic Programs
DLP	Description Logic Programs
VHE	Virtual Home Environment
KB	Knowledge Base

ΑΝΑΦΟΡΕΣ

[1] R.H. Katz, D. Long, M. Satyanarayanan and S. Tripathi, Workspaces in the Information Age, Leesburg, October, 1996.

[2] M. Weiser and J.S. Brown, “The Coming Age of Calm Technology”, Beyond Calculation: The Next Fifty Years of Computing, Copernicus, 1998.

[3] M. Satyanarayanan, “Pervasive Computing: Vision and Challenges”, IEEE Personal Communications, August 2001.

[4] M. Satyanarayanan, Caching Trust Rather Than Content. Operating System Review, October, 2000.

[5] J. Flinn and M. Satyanarayanan, Energy-aware Adaptation for Mobile Applications, Proceedings of the 17th ACM Symposium on Operating Systems and Principles. Kiawah Island, SC, December, 1999.

[6] B.D. Noble, M. Satyanarayanan, D. Narayanan, J.E. Tilton, J. Flinn and K.R. Walker, “Agile Application-Aware Adaptation for Mobility”, Proceedings of the 16th ACM Symposium on Operating Systems Principles, Saint-Malo, France, October, 1997.

[7] K. Nahrsted, H. Chu and S. Narayan, “QoS-aware Resource Management for Distributed Multimedia Applications”, Journal on High-Speed Networking, 1998.

[8] C.S. Ellis, “The Case for Higher-Level Power Management”, 7th IEEE Workshop on Hot Topics in Operating Systems, March, 1999.

[9] National Research Council, Energy-Efficient Technologies for the Dismounted Soldier Board on Army Science and Technology, Washington DC, 1997.

[10] A.R. Lebeck, X. Fan, H. Zheng and C.S. Ellis, “Power Aware Page Allocation”, Proceedings of the Ninth International Conference on Architerctural Support for Programming Languages and Operating Systems, November, 2000.

[11] J.G. Steiner, G. Neuman and J.I. Schiller, “Kerberos: An Authentication Service for Open Network Systems”, Proceedings of the Winter 1988 USENIX Technical Conference. Dallas, TX, February, 1988.

[12] A. Jain, L. Hong andS. Pankanti, Biometric Identification. Communications of the ACM, February, 2000.

[13] J. Case, M. Fedor, M. Schoffstall and J. Davin, A Simple Network Management Protocol. Internet Engineering Task Force (RFC 1157), 1990. Using Encryption for Authentication in Large Networks of Computers.

[14] A. Talukder and R.Yavagal, Mobile Computing, McGraw-Hill,2005, pp. 7-10.

[15] V. Jeyasri Arokiamary, Mobile Computing, Technical Publications Pune, 2nd revised edition, 2008, pp.13-14

[16] G. H. Formen and J. Zahorjan, “The challenges of mobile computing”, IEEE Computer, vol. 27, 1994, pp. 20-27.

[17] M. Satyanarayanan, “Mobile information access”, IEEE Personal Communications, vol. 3, 1996, pp. 26-33.

[18] T. D. Roza and G. Bilchev, “An overview of location based services”, BT Technology Journal, vol. 21, 2003, pp. 20-27.

[19] P. Korpipaa, “Managing context information in mobile devices”, IEEE Pervasive Computing, vol. 2, 2003, pp. 42-51.

[20] T. Edmonds, S. Hodges, A. Hopper, “Pervasive adaptation for mobile computing”, Proceedings of the 15th International Conference on Information Networking (ICOIN ’01), 2001, pp. 1-8.

[21] G.-C. Roman, A. Murphy and G. Picco, “Coordination and mobility”, A. Ominici, F. Zambonelli, M. Klusch and R. Tolksdorf, eds., Coordination of Internet Agents: Models, Technologies, and Applications, Springer, 2000, pp. 254-273.

[22] A. Umar, Mobile Computing and Wireless Communications, NGE Solutions, 2004.

[23] R. Davis, H.S. and P. Szolovits, "What is knowledge representation?", AI Magazine, vol. 14, 1993, pp. 17-33.

[24] S. Russel and P. Norvig, Artificial Intelligence – A Modern Approach, Prentice-Hall, 1995.

[25] J.F. Sowa, Knowledge Representation, Brooks Cole Publishing, 2000.

[26] J.W. Lloyd, Foundations of Logic Programming, Springer-Verlag, 1988.

[27] J. Minker, “Logic and Databases: Past, Present, and Future”, AI Magazine, vol. 18, 1997.

[28] K. Breitman, M.A. Casanova and W. Truszkowski, Semantic Web: Concept, Technologies and Applications, Springer-Verlag, 2007.

[29] F. Baader, D. Calvanese, D. McGuinness, D. Nardi and P. Patel-Schnider, eds., The description logic handbook, Cambridge Press, 2003.

[30] N. Guarino and P. Giaretta, “Ontologies and knowledge bases: Towards a terminological clarification”, N. Mars, ed., Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing, IOS Press, 1995, pp. 25–32.

[31] T. Gruber, A translation approach to portable ontology specifications, Knowledge Acquisition, vol. 5, 1993.

[32] N. Guarino, “Formal ontology and information systems”, Proceedings of Formal Ontology and Information Systems, June 1998.

[33] N. Noy and C. Hafner, “The state of the art in ontology design”, AI Magazine, vol 18., 1997, pp. 53–74.

[34] A. Farquhar, R. Fikes and J. Rice, “Tools for assembling modular ontologies in Ontolingua”, Proceedings of 14th American Association for Artificial Intelligence Conference, AAAI/MIT Press, 1997, pp. 436–441.

[35] P. Norvig and S. Russell, Artificial Intelligence, A Modern Approach, Prentice Hall Series in Artificial Intelligence, 2003.

[36] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker, “Description Logic Programms: Combining Logic Programms with Description Logic”, Proceedings of the 12th International World Wide Web Conference, May 2003.

[37] D. Brickley and R. V. Guha, RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation, February 2004.

[38] P. F. Patel-Schneider, P. Hayes, and I. Horrocks, OWL Web Ontology Language Semantics and Abstract Syntax, W3C Recommendation, February 2004.

[39] M. A. Chatti, S. Srirama, D. Kensche and Y. Cao, “Mobile Web Services for Collaborative Learning”, 4th International Workshop on Wireless, Mobile and Ubiquitous Technology in Education, 2006, pp. 129-133.

[40] P. Baumgartner, U. Furbach, and I. Niemela, “Hyper Tableaux”, Technical Report, Universitat Koblenz-Landau, vol. 8, 1996.

[41] C. Wernhard, System Description: KRHyper, Fachberichte Informatik, Universit¨at Koblenz-Landau, vol. 14, 2003.

[42] T. Kleemann and A. Sinner, “KRHyper - In Your Pocket, System Description”, Proceedings of the 20th International Conference on Automated Deduction, vol. 3632, 2005.

[43] P. Baumgartner, U. Furbach, and A. H. Yahya, Automated reasoning, knowledge representation and management, KI, vol. 1, 2005, pp. 5-11.

[44] P. Baumgartner, U. Furbach, M. Gross-Hardt, and T. Kleemann, “Model Based Deduction for Database Schema Reasoning”, S. Biundo, T. Fr¨uhwirth, and G. Palm, eds., Advances in Artificial Intelligence, Springer Verlag, 2004, vol. 3238, pp. 168–182.

[45] P. Baumgartner and U. Furbach, “PROTEIN: A PROver with a theory extension INterface”, Conference on Automated Deduction, 1994, pp. 769–773,

[46] Mysaifu, J.V.M. (2009); http://www2s.biglobe.ne.jp/~dat/java/project/jvm/index_en.html. [Προσπελάστηκε 16/5/11]

[47] V. Haarslev, R. Moller, and M. Wessel, “Querying the semantic web with racer+ nrql”, Proceedings of the KI-2004 International Workshop on Applications of Description Logics, September 24, 2004.

[48] The Protégé Ontology Editor and Knowledge Acquisition System; http://protege.stanford.edu/. [Προσπελάστηκε 16/5/11]

[49] Bossam Rule/OWL Reasoner; http://bossam.wordpress.com/. [Προσπελάστηκε 16/5/11]

[50] C. Forgy, "RETE, a fast algorithm for the many patterns many objects Match problem", Artificial Intelligence 19, 1982

[51] E. Sirin, B. Parsia, B. Cuenca Grau, A. Kalyanpur, and Y. Katz, “Pellet: A Practical OWL-DL Reasoner”, University of Maryland Institute for Advanced Computer Studies, 2005; http://mindswap.org/papers/PelletDemo.pdf. [Προσπελάστηκε 16/5/11]

[52] HermiT reasoner; http://www.hermit-reasoner.com/. [Προσπελάστηκε 16/5/11]

[53] B. Motik, D. Vrandecic, P. Hitzler, Y. Sure and R. Studer, “dlpconvert – converting OWL DLP statements to logic programs” , European Semantic Web Conference 2005 Demos and Posters,May 2005; http://owltools.ontoware.org/. [Προσπελάστηκε 16/5/11]

[54] Thea: A Prolog library for OWL2; http://www.semanticweb.gr/thea/. [Προσπελάστηκε 16/5/11]

[55] U. Hustadt, B. Motik and U. Sattler, “Reducing SHIQ Description Logic to Disjunctive Datalog Programs”, Proceedings of the 9th International Conference on Knowledge Representation and Reasoning, AAAI Press, 2004, pp.152–162.

[56] U. Hustadt, B. Motik and U. Sattler, “Reasoning for Description Logics around SHIQ in a Resolution Framework”, Technical Report, FZI, Karlsruhe, Germany,2004; http://www.fzi.de/wim/publikationen.php?id=1172. [Προσπελάστηκε 16/5/11]