
Εθνικο Μετσοβιο Πολυτεχνειο

Σχολη Ηλεκτρολογων Μηχανικων
Και Μηχανικων Υπολογιστων

Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων


# Εξατομικευμένη, Σημασιολογική και Διερευνητική Ανάλυση Δεδομένων

Διδακτορική Διατριβή

του


**Νικόλαου Μπικάκη**

Διπλωματούχου Ηλεκτρονικού Μηχανικού &
Μηχανικού Υπολογιστών Πολ. Κρήτης (2009)



Αθήνα, Ιανουάριος 2016

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων Και Μηχανικων Υπολογιστων
Τομεας Τεχνολογιας Πληροφορικης και Υπολογιστων

# Εξατομικευμένη, Σημασιολογική και Διερευνητική Ανάλυση Δεδομένων

Διδακτορική Διατριβή

του

**Νικόλαου Μπικάκη**

Διπλωματούχου Ηλεκτρονικού Μηχανικού &
Μηχανικού Υπολογιστών Πολ. Κρήτης (2009)

**Συμβουλευτική Επιτροπή:** Τ. Σελλής
Ι. Βασιλείου
Γ. Στάμου

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την 22$^{η}$ Ιανουαρίου 2016

| | | |
|---|---|---|
| Τ. Σελλής<br>Καθ. ΕΜΠ | Ι. Βασιλείου<br>Καθ. ΕΜΠ | Γ. Στάμου<br>Επ. Καθ. ΕΜΠ |
| Κ. Κοντογιάννης<br>Αναπλ. Καθ. ΕΜΠ | Θ. Δαλαμάγκας<br>Ερευν. Β΄ ΕΚ Αθηνά | Ε. Κουμπαράκης<br>Καθ. ΕΚΠΑ |

Α. Δεληγιαννάκης
Αναπλ. Καθ. Πολ. Κρήτης

Αθήνα, Ιανουάριος 2016

...

**Nikos Bikakis**
Electronic & Computer Engineer, PhD, NTUA



**Η παρούσα διατριβή εκπονήθηκε με χρηματοδότηση από τον Ειδικό Λογαριασμό Έρευνας Ε.Μ.Π.**

**Η Ελληνική έκδοση της διατριβής έχει προκύψει από *αποσπασματική* μετάφρασης της Αγγλικής έκδοσης [59]. Για την ολόκληρη διατριβή αναφερθείτε στο [59].**

# Περιεχόμενα

# Κατάλογος Σχημάτων

# Κατάλογος Πινάκων

# Περίληψη[1]

Στην εποχή των *Μεγάλων Δεδομένων* (Big Data), τα συστήματα αντιμετωπίζουν σημαντικές προκλήσεις που σχετίζονται με την αποδοτικότητα και την αποτελεσματικότητα τους. Οι προκλήσεις αυτές απορρέουν κυρίως από τον *Όγκο*, την *Ετερογένεια* και την *Ταχύτητα* που χαρακτηρίζει τα δεδομένα σήμερα. Σε αυτό το πλαίσιο, τα σημερινά συστήματα πρέπει σε *πραγματικό χρόνο* να διαχειρίζονται *μεγάλους όγκους* δεδομένων, καθώς και να λειτουργούν σε περιβάλλοντα όπου *διαφορετικοί χρήστες* οι οποίοι εργάζονται σε *διαφορετικά σενάρια*, δημιουργούν, διερευνούν και αναλύουν *διαφορετικές μορφές* δεδομένων. Προς την κατεύθυνση αυτή, η παρούσα διατριβή μελετά την ανάπτυξη εξατομικευμένων, διερευνητικών και σημασιολογικών τεχνικών για την διαχείριση και ανάλυση Μεγάλων Δεδομένων. Πιο συγκεκριμένα, προτείνονται μέθοδοι για: (α) *κλιμακούμενη διαχείριση και ανάλυση δεδομένων βασισμένη σε προτιμήσεις χρηστών*; (β) *αποδοτική διερεύνηση και οπτικοποίηση μεγάλων συνόλων δεδομένων*; και (γ) *σημασιολογική ολοκλήρωση, διερεύνηση και ανάκτηση δεδομένων*.

Όσον αφορά στο πρώτο μέρος εργασιών, αντικείμενο έρευνας αποτέλεσε η *εξατομικευμένη ανάλυση δεδομένων*, όπου μελετήθηκαν τα ακόλουθα προβλήματα. Αρχικά μελετάται το πρόβλημα της εύρεσης και ταξινόμησης αντικείμενων τα οποία θεωρούνται προτιμητέα από μια ομάδα χρηστών, με βάση τις προτιμήσεις τους. Αποτέλεσμα της μελέτης, είναι η διατύπωση μιας αντικειμενική και δίκαιης ερμηνεία αυτού του προβλήματος. Με βάση αυτή την ερμηνεία, αναπτύχθηκαν αποδοτικοί αλγόριθμοι βασισμένοι σε ευρετήρια και προτάθηκε ένα σχήμα αντικειμενικής ταξινόμησης, το οποίο ικανοποιεί αρκετές θεωρητικές ιδιότητες. Σε επόμενο πρόβλημα, πραγματοποιήθηκε εκτεταμένη μελέτη και σύγκριση τεχνικών αποτίμησης ερωτημάτων κορυφογραμμής δευτερεύουσας μνήμη. Πιο συγκεκριμένα, ένα σύνολο αλγορίθμων κορυφογραμμής μοντελοποιήθηκαν και υλοποιήθηκαν σύμφωνα με το μοντέλο εξωτερικής μνήμης. Επιπλέον, για τους υπό εξέταση αλγόριθμους προτείνεται ένα σύνολο παραλλαγών. Η εκτεταμένη πειραματική μελέτη ανέδειξε νέα συμπεράσματα σχετικά με την σχεδίαση και την απόδοση των αλγορίθμων κορυφογραμμής.

Στο δεύτερο μέρος εργασιών, του οποίου αντικείμενο έρευνας αποτέλεσε η *διερευνητική ανάλυση δεδομένων*, μελετήθηκαν δυο προβλήματα. Πιο συγκεκριμένα, μελετήθηκε το πρόβλημα της αποδοτικής και άμεσης οπτικής διερεύνησης σε μεγάλα σύνολα δεδομένων. Αποτέλεσμα της μελέτης, είναι η ανάπτυξη ενός πλαισίου πολλαπλών επιπέδων βασιζόμενο σε μια δεντρική δομή η οποία πραγματοποιεί την ιεραρχική ομαδοποίηση των δεδομένων. Λαμβάνοντας υπόψη διαφορετικά σενάρια διερεύνησης, το πλαίσιο επιτρέπει την αποδοτική διερεύνηση μέσω της σταδιακής κατασκευής της ιεραρχίας, η οποία βασίζεται στην αλληλεπίδραση του χρήστη. Επιπλέον, περιγράφεται μια μέθοδος η οποία παρέχει αποδοτική και άμεση προσαρμογή των ιεραρχιών με βάση

[1] Η Ελληνική έκδοση της διατριβής έχει προκύψει από *αποσπασματική* μετάφρασης της Αγγλικής έκδοσης [59]. Για την ολόκληρη διατριβή αναφερθείτε στο [59].

τις προτιμήσεις του χρήστη. Τέλος, παρουσιάζεται μια εκτεταμένη θεωρητική και πειραματική ανάλυση. Στο δεύτερο πρόβλημα μελετάται η διερεύνηση και οπτικοποίηση πολύ μεγάλων γράφων. Από αυτή τη μελέτη προέκυψε μια καινοτόμα μεθοδολογία η οποία επιτρέπει την αποδοτική οπτική διερεύνηση πολύ μεγάλων γράφων. Η μεθοδολογία που προτείνεται είναι παρόμοια με την μεθοδολογία που έχει υιοθετηθεί για την διερεύνηση γεωγραφικών χαρτών. Επιπλέον, παρουσιάζεται μια νέα τεχνική για την ευρετηρίαση και την αποθήκευση γράφων. Σε αυτό το πλαίσιο, οι αλληλεπιδράσεις του χρήστη μεταφράζονται σε αποδοτικούς χωρικούς τελεστές. Τέλος, προκειμένου να είναι εφικτή η οπτικοποίηση πολύ μεγάλων γράφων, μια προσέγγιση η οποία βασίζεται σε κατάτμηση εισάγεται.

Όσον αφορά στο τρίτο μέρος εργασιών, αντικείμενο έρευνας αποτέλεσε η *σημασιολογική ανάλυση δεδομένων*, όπου μελετήθηκαν τα ακόλουθα προβλήματα. Αρχικά μελετήθηκε το πρόβλημα της ενοποίησης μεταξύ του Σημασιολογικού και του XML περιβάλλοντος. Για το πρόβλημα αυτό, παρουσιάζεται ένα διαλειτουργικό πλαίσιο το οποίο προσφέρει δυνατότητες μετάφρασης ερωτήσεων καθώς και αντιστοίχισης και μετασχηματισμού σχημάτων. Πιο συγκεκριμένα παρουσιάζονται: ένα μοντέλο για την διατύπωση αντιστοιχίσεων μεταξύ OWL-RDF/S και XML Schema, μια μέθοδος για την μετάφραση SPARQL ερωτήσεων σε XQuery, καθώς και ένα μοντέλο για τον μετασχηματισμό XML Schemas σε OWL οντολογίες. Το δεύτερο πρόβλημα αφορά στη χρήση της σημασιολογίας στην επισημείωση και ανάκτηση εγγράφων. Για το πρόβλημα αυτό προτείνεται ένα σημασιολογικό μοντέλο επισημειώσεων, καθώς και μια μέθοδο εκμάθησης για τη σύσταση επισημειώσεων. Τέλος, παρουσιάζεται μια αποτελεσματική μέθοδος ανάκτησης, η οποία εμπλουτίζει τεχνικές ανάκτηση πληροφορίας με σημασιολογία. Στο τελευταίο πρόβλημα, μελετάται η μοντελοποίηση και η εξερεύνηση εξελισσόμενων δεδομένων, υιοθετώντας τεχνικές Διασυνδεμένων Δεδομένων (Linked Data). Αποτέλεσμα αυτής της μελέτης είναι η περιγραφή ενός μοντέλου αλλαγών βασισμένο σε RDF, καθώς και η ανάπτυξη υποδομής Διασυνδεμένων Δεδομένων, η οποία επιτρέπει την διερεύνηση και ανάκτηση εξελισσόμενων δεδομένων.

**Λέξεις Κλειδιά:** Μεγάλα Δεδομένα, Διερεύνηση Δεδομένων, Εξατομικευμένα Συστήματα, Συστήματα Συστάσεων, Ταξινόμηση, Οπτικοποίηση, Οπτική Ανάλυση, Σημασιολογικός Ιστός, Διαλειτουργικότητα, Επεξεργασία Ερωτήσεων, Χωρικά δεδομένα.

# Abstract

In the *Big Data* era, systems in several application areas face significant efficiency and effectiveness challenges, due to the ever increasing *Volume*, *Variety* and *Velocity* of data. In this context, systems have to handle *vast amounts* of data in *real time* and operate in environments where *different users*, working on *different scenarios*, generate, explore and analyse *different forms* of data. To this direction, this thesis studies the development of *personalization*, *exploration* and *semantic* techniques for facilitating Big Data management and analysis. Specifically, we propose methods for: (a) scalable preference-aware data management and analysis; (b) efficient exploration and visualization over large datasets; and (c) semantic data integration, exploration and retrieval.

In the context of *personalized data analysis*, we study the following problems. First, we study the problem of finding and ranking objects that are preferable by a group of users based on their preferences. We propose an objective and fair interpretation of this problem. Based on this interpretation, we develop efficient index-based algorithms and we introduce an objective ranking scheme satisfying several theoretical properties. In the next problem, we thoroughly study the performance of some of the most well-known external memory skyline algorithms. Particularly, the considered algorithms are redesigned following a formal external memory model. Then, we propose numerous different design choices and we study the resulted algorithms' variations.

Regarding *exploratory data analysis* two problems are considered. In the first one we handle efficient on-the-fly visual exploration over large sets of data. For this problem we propose a multilevel framework that exploits a tree-based structure to hierarchically aggregate objects. Considering different exploration scenarios, we enable efficient exploration via incremental hierarchy construction and prefetching based on user interaction. Further, we provide on-the-fly efficient adaptation of the hierarchies based on user preferences. The second problem considers the exploration and visualization of very large graphs. We propose a new paradigm that allows efficient large graph visual exploration, similar to the exploration paradigm used in maps. Also, we present a disk-based scheme in order to index and store the visualized graph. In this setting, user's interactions are translated to efficient spatial operations. Finally, in order to visualize very large graphs, a partition-based visualization approach is introduced.

With respect to *semantic data analysis*, we focus on three problems. The first problem regards the integration between XML and Semantic Web. We present an interoperability framework that bridges the heterogeneity gap by exploiting a model for the expression of OWL–RDF/S to XML Schema mappings, a method for SPARQL to XQuery translation, and model which transforms XML Schemas into

OWL ontologies. The second problem regards the use of semantics in document annotation and retrieval. For this problem we propose a semantic-based annotation model, as well as a learning method for recommending annotations. Finally, we introduce an effective retrieval method that enriches information retrieval techniques with semantics. In the last problem, we study the modelling and the exploration of evolving data, adopting the Linked Data paradigm. As a result, we propose a RDF-based change model and we develop a Linked Data infrastructure that allows exploration and retrieval over evolving data.

**Keywords:** Big Data, Data Exploration, Personalized Systems, Recommender Systems, Ranking, Visualization, Visual Analytics, Semantic Web, Interoperability, Query Processing, Spatial Data.

# Κεφάλαιο 1

# Εισαγωγή

Ο *Όγκος*, η *Ετερογένεια*, και η *Ταχύτητα* είναι τα τρία κύρια χαρακτηριστικά που χρησιμοποιούνται ευρέως για να περιγράψουν την εποχή των *Μεγάλων Δεδομένων* (Big Data). Σε αυτά τα πλαίσια, αρκετές προκλήσεις προκύπτουν για τα σημερινά συστήματα τα οποία θα πρέπει να είναι ικανά να χειριστούν αποδοτικά *μεγάλους όγκους* δεδομένων σε *πραγματικό χρόνο*. Επιπλέον, τα σημερινά συστήματα πρέπει να λειτουργούν σε αρκετά *ετερογενή* περιβάλλοντα τα οποία χαρακτηρίζονται από *διαφορετικούς χρήστες* (π.χ., ενδιαφέροντα, ικανότητες, χαρακτηριστικά) οι οποίοι δουλεύουν σε πληθώρα *διαφορετικών σεναρίων* και δημιουργούν, διερευνούν και αναλύουν δεδομένα σε *διαφορετικές μορφές*.

Στο περιβάλλον που περιγράψαμε, ένας μεγάλος αριθμός προβλημάτων που σχετίζονται με τη διαχείριση και την ανάλυση δεδομένων παρουσιάζει πολλές προκλήσεις, τόσο για τους χρήστες όσο και για τα συστήματα. Στην συνέχεια, περιγράφουμε την υιοθέτηση *εξατομικευμένων* (personalized), *διερευνητικών* (exploration) και *σημασιολογικών* (semantic) τεχνικών, προκειμένου να επιλυθεί μια σειρά προβλημάτων στα πλαίσια της διαχείρισης και ανάλυσης Μεγάλων Δεδομένων. Επιπρόσθετα, παρουσιάζουμε τις κυριότερες προκλήσεις του σημερινού περιβάλλοντος.

Ο μεγάλος όγκος των δεδομένων, σε συνδυασμό με την ετερογένεια των χρηστών και των σεναρίων, έχουν σαν αποτέλεσμα την δυσκολία των χρηστών στην ανεύρεση χρήσιμων πληροφοριών ανάλογα με τα ενδιαφέροντα και τις ανάγκες τους. Σε αυτά τα πλαίσια, η ανάπτυξη και υιοθέτηση τεχνικών εξατομίκευσης είναι θεμελιώδης για τα σημερινά συστήματα παροχής και διαχείρισης πληροφοριών.

Οι τεχνικές εξατομίκευσης επιχειρούν να παρέχουν εξατομικευμένες υπηρεσίες χρησιμοποιώντας πληροφορίες από τα προφίλ των χρηστών, τις προτιμήσεις και τα ενδιαφέροντά τους, κτλ. Τα συστήματα εξατομίκευσης βοηθούν τους χρήστες να ανακτούν, να οργανώνουν και να διαχειρίζονται πληροφορίες, καθώς και να προσαρμόζουν τη συνολική τους εμπειρία βασισμένοι στις εκάστοτε προτιμήσεις και ανάγκες. Παράλληλα, από την πλευρά των παρόχων, η προσφορά εξατομικευμένων υπηρεσιών και εφαρμογών έχει ως αποτέλεσμα την προσέλκυση μεγαλύτερου αριθμού χρηστών. Συνεπώς, η ανάπτυξη συστημάτων εξατομίκευσης είναι μείζονος σημασίας τόσο για τους τελικούς χρήστες, όσο και για τους παρόχους υπηρεσιών και πληροφοριών.

Από τα παραπάνω γίνεται εμφανές ότι η ανάγκη για μεθόδους που παρέχουν εξατομικευμένες υπηρεσίες στους χρήστες αυξάνεται όλο και περισσότερο. Οι τεχνικές εξατομίκευσης θα πρέπει να μπορούν να επεξεργάζονται μεγάλο εύρος σύνθετων τύπων αντικειμένων, τις συσχετίσεις τους, καθώς και τα σχήματα που πιθανά τα περιγράφουν. Επιπλέον, τα συστήματα εξατομίκευσης, θα πρέπει να είναι ικανά να ανταποκριθούν σε

ακόμα πιο σύνθετα προβλήματα, όπως για παράδειγμα, σενάρια στα οποία τα συστήματα πρέπει να προσφέρουν εξατομικευμένες υπηρεσίες σε ομάδες χρηστών. Τέλος, αναφορικά με την απόδοση, οι μέθοδοι πρέπει να είναι ικανές να χειριστούν αποδοτικά πολύ μεγάλους αριθμούς αντικειμένων και χρηστών, όπου οι πληροφορίες των χρηστών (π.χ., προτιμήσεις, ανάγκες) και των αντικειμένων συνεχώς μεταβάλλονται.

Μια άλλη συμπληρωματική κατεύθυνση προς τη βελτίωση της εμπειρίας του χρήστη, είναι η παροχή της δυνατότητας στους χρήστες να διερευνούν και να αναλύουν μεγάλα και σύνθετα σύνολα δεδομένων. Ο σκοπός της *διερεύνησης δεδομένων* είναι να διευκολύνει την αντίληψη και διαχείριση δεδομένων, καθώς και την εξαγωγή γνώσης και συμπερασμάτων. Τα συστήματα διερεύνησης έχουν μεγάλη σημασία στην σημερινή εποχή, στην οποία ο όγκος και η ετερογένεια των διαθέσιμων πληροφοριών δυσχεραίνουν την διερεύνηση και την ανάλυση των δεδομένων.

Οι τεχνικές οπτικοποίησης που υιοθετούνται από την πλειοψηφία των σύγχρονων συστημάτων διερεύνησης, παρέχουν στους χρήστες μέσα, προκειμένου να διερευνούν το περιεχόμενο των δεδομένων, να ταυτοποιούν ενδιαφέροντα μοτίβα, να εξάγουν σχέσεις και αιτιότητες, και να υποστηρίζουν sense-making δραστηριότητες, οι οποίες δεν είναι πάντα δυνατές με τις παραδοσιακές τεχνικές ανάλυσης. Τα σύγχρονα συστήματα διερεύνησης και οπτικοποίησης πρέπει να μπορούν να διαχειριστούν αποτελεσματικά τεράστιους αριθμούς αντικειμένων σε πραγματικό χρόνο. Επιπλέον, τα συστήματα πρέπει να επιλύσουν προβλήματα σχετικά με την οπτική αναπαράσταση, όπως το overploting. Η οπτικοποίηση μεγάλων αριθμών αντικειμένων είναι μια αρκετά δύσκολη διαδικασία, καθώς τα συστήματα πρέπει να *"squeeze a billion records into a million pixels"*. Τέλος, η απαίτηση για κλιμακούμενη διερεύνηση πρέπει να συνδυαστεί με την ποικιλία των προτιμήσεων και των απαιτήσεων που τίθονται από τους διαφορετικούς χρήστες και εργασίες. Για αυτό το λόγο, τα συστήματα πρέπει να παρέχουν στους χρήστες την ικανότητα να προσαρμόζουν την εμπειρία διερεύνησης με βάση τις προτιμήσεις και τις απαιτήσεις που τίθονται από την εκάστοτε εργασία.

Πέρα από τις προκλήσεις που προκύπτουν από την ποικιλία των χρηστών, των εργασιών και των περιεχομένων, επιπρόσθετες προκλήσεις τίθονται από την ετερογένεια που χαρακτηρίζει τα σημερινά δεδομένα, συστήματα και τεχνολογίες. Έτσι, η παροχή ομοιόμορφης πρόσβασης σε ετερογενείς πηγές δεδομένων, και η υποστήριξη διαλειτουργικότητας μεταξύ διαφορετικών συστημάτων και τεχνολογιών, είναι θέμα μείζονος σημασίας. Τέτοιες προκλήσεις έχουν οδηγήσει στην ανάπτυξη του οράματος του *Σημασιολογικού Ιστού* (Semantic Web - SW), που μπορεί να θεωρηθεί ως ένα συνεργατικό περιβάλλον όπου τα συστήματα χρησιμοποιούν και μοιράζονται δεδομένα με διαφάνεια. Η σημασιολογία επιτρέπει τη περιγραφή των πληροφοριών με επίσημο και σαφή τρόπο, επιτρέποντας τον ορισμό σύνθετων εννοιών και σχέσεων. Η χρήση της σημασιολογίας μπορεί να βελτιώσει σημαντικά την αποτελεσματικότητα των συστημάτων σε θέματα αναζήτησης, κοινής χρήσης και συνδυασμού πληροφοριών.

Το SW είναι ένα ανοιχτό περιβάλλον που αποτελείται από εκατοντάδες μεγάλα διασυνδεδεμένα σύνολα δεδομένων. Το SW βασίζεται σε σημασιολογικές τεχνολογίες και πρότυπα για την αναπαράσταση και τη διαχείριση διαδικτυακών πληροφοριών. Οι SW εφαρμογές πρέπει να συνυπάρχουν και να διαλειτουργούν με τις υπάρχουσες εφαρμογές καθώς και να έχουν πρόσβαση σε παραδοσιακά συστήματα. Επομένως, είναι θέμα ζωτικής σημασίας, οι SW υποδομές να μπορούν να έχουν διαφανή πρόσβαση σε πληροφορίες που είναι αποθηκευμένες σε ετερογενείς πηγές δεδομένων. Επιπλέον, οι SW χρήστες δεν πρέπει να χρησιμοποιούν διαφορετικά μοντέλα δεδομένων, διαφορετικές γλώσσες και τεχνολογίες για την ανάπτυξη των εφαρμογών τους και για την

πρόσβασης σε πηγές δεδομένων. Από τα παραπάνω γίνεται εμφανής η αναγκαιότητα για την ανάπτυξη μεθόδων που θα παρέχουν διαλειτουργικότητα μεταξύ διαφορετικών υποδομών, καθώς και διαφανής πρόσβαση σε ετερογενείς πηγές δεδομένων.

## 1.1 Συνεισφορά

Η παρούσα διατριβή παρουσιάζει καινοτόμες μεθόδους για τη διαχείριση και την ανάλυση δεδομένων. Μελετά τρεις κατευθύνσεις με στόχο την επίτευξη διαχείρισης και ανάλυσης Μεγάλων Δεδομένων (Big Data). Συγκεκριμένα, στην πρώτη κατεύθυνση, προτείνονται μέθοδοι για κλιμακούμενη διαχείριση και ανάλυση δεδομένων βασισμένη σε προτιμήσεις χρηστών. Στην δεύτερη, προτείνονται τεχνικές αποδοτικής διερεύνησης και οπτικοποίηση μεγάλων συνόλων δεδομένων. Τέλος, προτείνονται μέθοδοι σημασιολογικής ολοκλήρωσης, διερεύνησης και ανάκτησης δεδομένων. Η συνεισφορά της διατριβής συνοψίζεται στα παρακάτω σημεία.

1. Λαμβάνοντας υπόψιν ένα σύνολο από χρήστες, καθένας από τους οποίους προσδιορίζει τις ατομικές του προτιμήσεις σε χαρακτηριστικά των αντικειμένων, μελετάμε το πρόβλημα της εύρεσης και της ταξινόμησης των αντικειμένων που θεωρούνται προτιμητέα από όλους τους χρήστες. Εισάγουμε και προτείνουμε μια αντικειμενική και δίκαιη ερμηνεία αυτού του προβλήματος, βασισμένη σε Pareto συνάθροιση (aggregation). Λαμβάνοντας υπόψιν αυτή την ερμηνεία, μελετάμε τρία σχετικά προβλήματα. Το πρώτο είναι η εύρεση των αντικειμένων που θεωρούνται ομόφωνα ιδανικά από όλο το σύνολο των χρηστών. Στο δεύτερο πρόβλημα, χαλαρώνεται η απαίτηση της ομοφωνίας και απαιτείται μόνο ένα ποσοστό των χρηστών να συμφωνεί. Τέλος, στο τρίτο πρόβλημα, προτείνεται ένα αποτελεσματικό σχήμα ταξινόμησης (ranking scheme) βασισμένο σε Pareto συνάθροιση. Για να βελτιώσουμε την αποδοτικότητα όταν χειριζόμαστε κατηγορηματικά χαρακτηριστικά (categorical attributes), εισάγουμε έναν μετασχηματισμό κατηγορηματικών τιμών σε αριθμητικές, το οποίο επιδεικνύει μερικές χρήσιμες ιδιότητες επιτρέποντας την χρήση παραδοσιακών δομών δεικτοδότησης (index structures). Βασισμένοι σε αυτό τον μετασχηματισμό, προτείνουμε έναν αλγόριθμο βασισμένο σε ευρετήριο (index-based). Ο αλγόριθμος χρησιμοποιεί ένα ευρετήριο διαμερισμού χώρου (space partitioning index) προκειμένου να ταξινομεί ιεραρχικά τα αντικείμενα. Σχετικά με το πρόβλημα της ταξινόμησης, μελετάμε θεωρητικά την συμπεριφορά του σχήματος ταξινόμησης και παρουσιάζουμε έναν αριθμό θεωρητικών ιδιοτήτων που ικανοποιούνται από την προσέγγισή μας. Επιπλέον, μελετήθηκαν μερικές ενδιαφέρουσες προεκτάσεις των προαναφερθέντων προβλημάτων που περιλαμβάνουν τα παρακάτω ζητήματα: ιδιότητες με πολλαπλές τιμές (multi-values attributes), μη-δεντροειδείς ιεραρχίες (non-tree hierarchies), δεικτοδότηση υποχώρων subspace indexing, και αντικειμενικά χαρακτηριστικά (objective attributes). Μια λεπτομερή πειραματική αξιολόγηση επιβεβαιώνει την αποδοτικότητα και την αποτελεσματικότητα των προτεινόμενων μεθόδων. Συγκεκριμένα, οι index- based τεχνικές είναι μίας τάξης μεγέθους γρηγορότερες από τις απλές προσεγγίσεις, κλιμακώνοντας σε εκατομμύρια αντικείμενα και χιλιάδες χρήστες. Τέλος, το προτεινόμενο σχήματος ταξινόμησης εμφανίζει καλύτερες επιδόσεις από τις παραδοσιακές rank aggregation μεθόδους σε σχέση με την ακρίβεια και την ανάκληση. Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στα [61, 60].

2. Οι ερωτήσεις κορυφογραμμής επιστρέφουν ένα σύνολο από μη-κυριαρχούμενα αντικείμενα, ένα αντικείμενο λέμε ότι κυριαρχείται, όταν υπάρχει άλλο αντικείμενο με καλύτερες τιμές σε όλα τα χαρακτηριστικά του. Λαμβάνοντας υπόψιν το πρόβλημα της κορυφογραμμής, μελετάμε λεπτομερειακά μερικούς από τους πιο γνωστούς αλγορίθμους κορυφογραμμής. Παρόλο που οι αλγόριθμοι που μελετάμε έχουν σχεδιαστεί να λειτουργούν σε δευτερεύουσα μνήμη, έχει δοθεί λίγη προσοχή σε σημαντικές λεπτομέρειες σχετικά με τη διαχείριση μνήμης. Για παράδειγμα, όλοι οι αλγόριθμοι υποθέτουν ότι η μονάδα μεταφοράς I/O (Εισόδου/Εξόδου) είναι το αντικείμενο, ενώ σε ένα αληθινό σύστημα είναι ένα block, δηλαδή ένα σύνολο αντικειμένων. Η εργασία μας μελετάει τέτοια θέματα εισάγοντας ένα πιο ρεαλιστικό μοντέλο για τις I/O λειτουργίες. Επιπλέον, μελετάμε λεπτομερώς τη διαχείριση των αντικειμένων μέσα στη μνήμη (in-memory objects). Συγκεκριμένα, εισάγουμε διάφορες πολιτικές για δύο βασικές λειτουργίες: τη διάσχιση και την απομάκρυνση των in-memory αντικειμένων. Και οι δύο αυτές λειτουργίες έχουν σημαντικές επιπτώσεις τόσο στον αριθμό των απαιτούμενων I/Os όσο και στον απαιτούμενο CPU χρόνο. Η πειραματική αξιολόγηση των αλγορίθμων, πραγματοποιήθηκε χρησιμοποιώντας υλοποιήσεις, αυστηρά βασισμένες σε δευτερεύουσα μνήμη και όχι σε προσομοιώσεις. Από την αξιολόγηση πάνω σε συνθετικά και αληθινά σύνολα δεδομένων, προέκυψαν χρήσιμα συμπεράσματα. Συγκεκριμένα, δείχνουμε ότι, σε πολλές περιπτώσεις και αντίθετα με την κοινή πεποίθηση, οι αλγόριθμοι που πραγματοποιούν προ-επεξεργασία (τυπικά ταξινομούν) την βάση δεδομένων, δεν είναι πιο αποδοτικοί. Τέλος, πραγματοποιήσαμε εκτενή μελέτη των προτεινόμενων πολιτικών. Από την μελέτη προέκυψε, ότι σε δεδομένα με συγκεκριμένα χαρακτηριστικά οι πολιτικές αυτές μπορούν να μειώσουν τον αριθμό των ελέγχων κυριαρχίας (dominance checks) παραπάνω από 50%. Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στο [68].

3. Μελετάμε το πρόβλημα της άμεσης οπτικής διερεύνησης σε μεγάλα σύνολα δεδομένων. Ως αποτέλεσμα, προτείνουμε ένα πλαίσιο (framework) που προσφέρει προσωποποιημένη πολυεπίπεδη διερεύνηση και ανάλυση αριθμητικών και χρονικών δεδομένων. Το πλαίσιό μας βασίζεται σε μια ελαφριά δεντροειδούς δομή δεδομένων (lightweight tree-based structure). Αυτή η δομή ομαδοποιεί (aggregate) τα αντικείμενα εισόδου σε ένα ιεραρχικό πολυεπίπεδο μοντέλο. Ορίζουμε δύο εκδόσεις αυτού του μοντέλου, που υιοθετούν διαφορετικές προσεγγίσεις οργάνωσης δεδομένων. Όταν οι προτιμήσεις των χρηστών δεν είναι διαθέσιμες, μια μέθοδος που λαμβάνει υπόψιν της τα χαρακτηριστικά των δεδομένων εισόδου, καθώς και τις παραμέτρους του περιβάλλοντος (π.χ., ανάλυση οθόνης, παράμετροι συστήματος οπτικοποίησης) εκτιμά τις καταλληλότερες παραμέτρους για την κατασκευή της ιεραρχίας. Ορίζουμε ακόμα διαφορετικά σενάρια διερεύνησης, υποθέτοντας διαφορετικές προτιμήσεις των χρηστών. Προκειμένου να επιτευχθεί η αποδοτική διερεύνηση σε μεγάλα σύνολα δεδομένων, το πλαίσιό μας προσφέρει σταδιακή κατασκευή (ινχρεμενταλ ζονστρυχτιον) καθώς και πρεφετχινγ, βασιζόμενα στην αλληλεπίδραση με τον χρήστη. Επιπρόσθετα, το πλαίσιο παρέχει μια μέθοδο η οποία δυναμικά και αποδοτικά προσαρμόζει την υπάρχουσα ιεραρχία σε μια νέα, υιοθετώντας τις προτιμήσεις του χρήστη. Μια λεπτομερή θεωρητική ανάλυση, μια αξιολόγηση επίδοσης και μια μελέτη με πραγματικούς χρήστες, αναδεικνύουν την αποδοτικότητα και την αποτελεσματικότητα του προτεινόμενου πλασίου. Το πλαίσιο υλοποιήθηκε σαν ένα πρωτότυπο web-based εργαλείο, που

ονομάζεται *synopsViz* και προσφέρει πολυεπίπεδη οπτική διερεύνηση και ανάλυση σε σύνολα Διασυνδεδεμένων Δεδομένων (Linked Data datasets). Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στα [67, 69, 70].

4. Μελετάμε το πρόβλημα της οπτικοποίησης και της διερεύνησης πολύ μεγάλων γράφων. Για αυτό το πρόβλημα εισάγουμε το *graphVizdb*, μια νέα πλατφόρμα που προσφέρει διαδραστική οπτικοποίηση μεγάλων γράφων (large graph interactive visualization). Η προτεινόμενη πλατφόρμα βασίζεται σε ένα νέο τρόπο αλληλεπίδρασης με το οπτικοποιημένο γράφο που είναι παρόμοιος με εκείνον της διερεύνησης γεωγραφικών χαρτών. Η πλατφόρμα οφείλει την αποδοτικότητά της σε μια καινοτόμα τεχνική για την ευρετηρίαση (indexing) και την αποθήκευση του γράφου. Η βασική ιδέα είναι ότι το γράφημα σχεδιάζεται σε ένα offline στάδιο προεπεξεργασίας χρησιμοποιώντας έναν υπάρχον layout αλγόριθμο. Μετά από τη σχεδίαση του γραφήματος, οι συντεταγμένες που αναθέτονται σε κάθε κόμβο (με βάση ένα ευκλείδειο επίπεδο) ευρετηριάζονται με μια χωρική δομή δεδομένων, δηλαδή ένα R-tree, και αποθηκεύονται στην βάση δεδομένων. Κατά τον χρόνο της διερεύνησης, καθώς ο χρήστης πλοηγείται στον γράφο, το σύστημά μας αντιστοιχίζει τις λειτουργίες του χρήστη σε αποδοτικές χωρικές λειτουργίες (δηλαδή window queries). Βασιζόμενοι στις συντεταγμένες, συγκεκριμένα κομμάτια του γράφου ανακτώνται από την βάση δεδομένων και στέλνονται στον χρήστη. Προκειμένου να οπτικοποιήσουμε πολύ μεγάλους γράφους, προτείνουμε μια partition-based προσέγγιση οπτικοποίησης γραφήματος. Σε αυτή την προσέγγιση, ο γράφος εισόδου χωρίζεται σε ένα σύνολο από μικρότερους υπογράφους, έπειτα κάθε υπογράφος οπτικοποιείται, τέλος οι υπογράφοι που προκύπτουν οργανώνονται και συνθέτουν έναν μόνο γράφο. Οι υπογράφοι οργανώνονται και συνθέτονται με βάση έναν άπληστο αλγόριθμο που προσπαθεί να βελτιώσει (δηλαδή, να ελαχιστοποιήσει το μήκος των ακμών, να αποφύγει επικαλύψεις) τη διάταξη του γραφήματος που προκύπτει. Αξιολογούμε την απόδοση των μεθόδων μας χρησιμοποιώντας αρκετά πραγματικά σύνολα γραφικών δεδομένων. Η πλατφόρμα μας μπορεί να προσφέρει αποδοτική οπτική διερεύνηση σε πολύ μεγάλα γραφήματα (δηλαδή, 300M ακμές/κόμβοι) χρησιμοποιώντας commodity hardware. Τέλος, αναπτύσσουμε ένα web-based πρότυπο το οποίο υποστηρίζει τέσσερις κύριες λειτουργίες: διαδραστική πλοήγηση, πολυεπίπεδη διερεύνηση, επιλογή και διαχείριση υπογράφων και αναζήτηση με λέξεις-κλειδιά. Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στα [66, 65].

5. Μελετάμε το πρόβλημα της διαλειτουργικότητας μεταξύ του Σημασιολογικού και του XML περιβάλλοντος. Για αυτό το πρόβλημα, προτείνουμε το πλαίσιο *SPARQL2XQuery* το οποίο γεφυρώνει το κενό ετερογένειας και δημιουργεί ένα διαλειτουργικό περιβάλλον. Το πλαίσιο επιτρέπει στις SPARQL ερωτήσεις που αναφέρονται σε σημασιολογικούς πόρους να μεταφράζονται αυτόματα σε XQuery εκφράσεις, με βάση ένα σύνολο προκαθορισμένων αντιστοιχίσεων. Πιο συγκεκριμένα, ορίζουμε ένα μοντέλο αντιστοιχίσεων για την διατύπωση αντιστοιχίσεων μεταξύ OWL–RDF/S και XML Schema, καθώς και μια μέθοδο για μετάφραση από SPARQL σε XQuery. Επιπλέον, το πλαίσιο υποστηρίζει τόσο τον χειροκίνητο όσο και τον αυτόματο προσδιορισμό των αντιστοιχήσεων μεταξύ οντολογιών και XML σχημάτων. Στην περίπτωση του αυτόματου προσδιορισμού αντιστοιχίσεων, το SPARQL2XQuery εκμεταλλεύεται το στοιχείο XS2OWL το οποίο μετατρέπει XML σχήματα σε OWL οντολογίες. Το στοιχείο XS2OWL υπο-

στηρίζει τις πιο πρόσφατες εκδόσεις των προτύπων (δηλαδή, XML Schema 1.1 και OWL 2). Τέλος, πραγματοποιήθηκε μια λεπτομερής πειραματική αξιολόγηση προκειμένου να μελετηθεί η αποδοτικότητα των προτεινόμενων μεθόδων. Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στα [74, 71, 72, 308, 63, 64].

6. Μελετάμε το πρόβλημα της σημασιολογικής ανάκτησης πληροφοριών. Για το πρόβλημα αυτό, προτείνουμε το πλαίσιο *GoNTogle* το οποίο υποστηρίζει ένα σημασιολογικό μοντέλο επισημειώσεων. Το πλαίσιο παρέχει τόσο αυτόματο όσο και χειροκίνητο μηχανισμό επισημειώσεων. Ο μηχανισμό αυτόματων επισημειώσεων βασίζεται σε μια μέθοδο εκμάθησης η οποία χρησιμοποιεί το ιστορικό επισημειώσεων του χρήστη, καθώς και πληροφορίες κειμένου, προκειμένου να προτείνει αυτόματα επισημειώσεις για νέα κείμενα. Επιπλέον, εισάγουμε μια υβριδική μέθοδο ανάκτησης (hybrid retrieval method) που παρέχει έναν ευέλικτο συνδυασμό textual-based και semantic-based ανάκτησης σε συνδυασμό με ανεπτυγμένες σημασιολογικές λειτουργίες. Οι προτεινόμενες μέθοδοι εφαρμόζονται σε ένα πλήρως λειτουργικό εργαλείο και η αποτελεσματικότητά τους επιβεβαιώνεται πειραματικά. Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στα [62, 164].

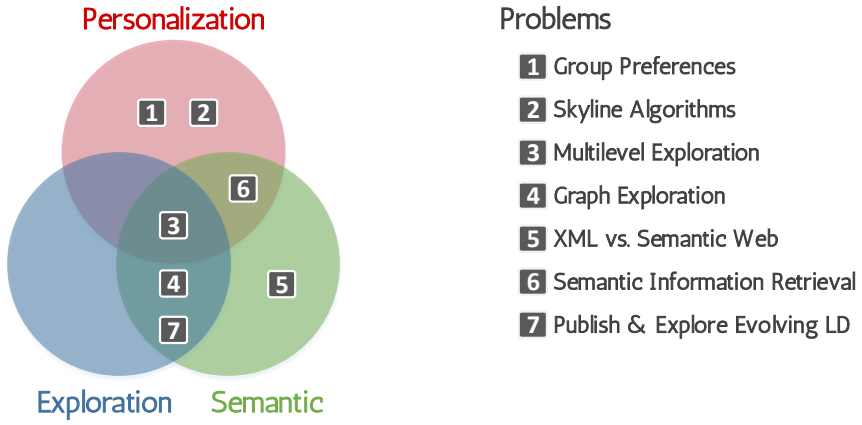
7. Μελετάμε το πρόβλημα της μοντελοποίησης, δημοσίευσης και διερεύνησης εξελισσόμενων επιστημονικών δεδομένων, υιοθετώντας τεχνικές των Διασυνδεδεμένων Δεδομένων. Για το συγκεκριμένο πρόβλημα, προτείνουμε ένα RDF μοντέλο αλλαγών για να την περιγραφή των εξελισσόμενων δεδομένων. Βασισμένοι σε αυτό το μοντέλο, μετατρέπουμε παραδοσιακά βιολογικά δεδομένα σε εξελισσόμενα Διασυνδεδεμένα Δεδομένα. Η υποδομή διασυνδεδεμένων δεδομένων που αναπτύξαμε μπορεί να βοηθήσει τους βιολόγους να διερευνήσουν βιολογικές οντότητες καθώς και να μελετήσουν την εξέλιξή τους. Τα παραπάνω αποτελέσματα δημοσιεύτηκαν στο [125].

**Πίνακας 1.1:** Επισκόπηση Διατριβής

| **Μέρος** | **Πρόβλημα** | **Κεφ./Ενότ..** | **Αποτελ.** |
|---|---|---|---|
| **I: Εξατομικευμένη** | Προτιμητέα Αντικείμενα με βάση Προτιμήσεις Ομάδας Χρηστών | 2 | [61, 60] |
| | Ερωτήματα Κορυφογραμμής Δευτερεύουσας Μνήμης | 3 | [68] |
| **II: Διερευνητική** | Αποδοτική Πολυεπίπεδη Διερεύνηση | 4.1 | [67, 69, 70] |
| | Κλιμακούμενη Εξερεύνηση Γράφων | 4.2 | [66, 65] |
| **III: Σημασιολογική** | XML & Σημασιολογική Διαλειτουργικότητα | 5 | [74, 71, 72, 308, 63, 64] |
| | Σημασιολογική Ανάκτηση Πληροφορίας | 6.1 | [62, 164] |
| | Δημοσιοποίηση & Διερεύνηση Εξελισσόμενων Διασυνδεδεμένων Δεδομένων | 6.2 | [125] |

## 1.2 Δομή

Η διατριβή είναι οργανωμένη σε τρία κύρια μέρη: (I) Εξατομικευμένη, (II) Διερευνητική, και (III) Σημασιολογική ανάλυση των δεδομένων. Η δομή της διατριβή, καθώς και τα αποτελέσματα συνοψίζονται στον Πίνακας 1.1. Επιπρόσθετα, το Σχήμα 1.1 παρουσιάζει τις σχέσεις μεταξύ των εξεταζόμενων προβλημάτων και των τρεις κύριων ερευνητικών περιοχών.




**Σχήμα 1.1:** Προβλήματα Διατριβής vs. Ερευνητικές Περιοχές

Αναλυτικότερα, το υπόλοιπο αυτής της διατριβής είναι δομημένο ως εξής. Το *Κεφάλαιο 2* παρουσιάζει τις μεθόδους για την εύρεση και την ταξινόμηση αντικειμένων λαμβάνοντας υπόψη τις προτιμήσεις από μια ομάδα χρηστών. Το *Κεφάλαιο 3* μελετά αλγόριθμους κορυφογραμμής δευτερεύουσας μνήμης. Στο *Κεφάλαιο 4* παρουσιάζονται προσεγγίσεις σχετικά με την οπτική διερεύνηση και ανάλυση μεγάλων συνόλων δεδομένων. Το *Κεφάλαιο 5* παρουσιάζει τις μεθόδους για την επίτευξη διαλειτουργικότητας μεταξύ του XML και του Σημασιολογικού Ιστού. Στο *Κεφάλαιο 6* εισάγονται οι προσεγγίσεις μας αναφορικά με την σημασιολογική αναζήτηση και ανάκτηση. Τέλος, το *Κεφάλαιο 7* συνοψίζει τις συνεισφορές της διατριβής, και παρουσιάζει πιθανές επεκτάσεις και οι ιδέες για μελλοντικές εργασίες.

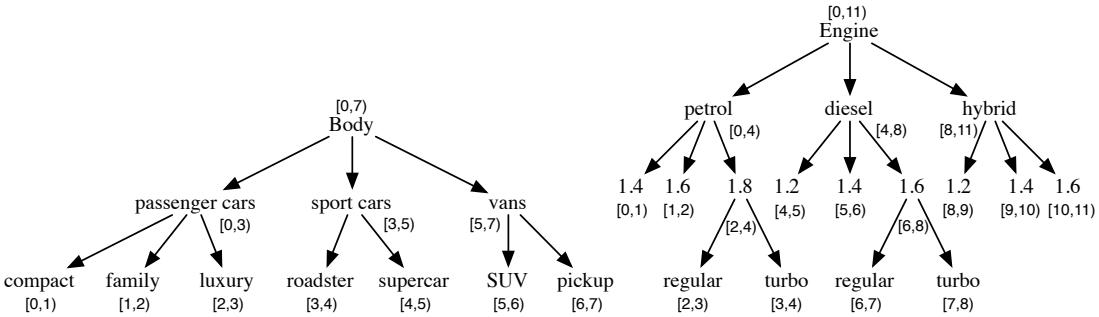
# Μέρος Ι

# Εξατομικευμένη Ανάλυση Δεδομένων

# Κεφάλαιο 2

# Προτιμητέα Αντικείμενα με βάση Προτιμήσεις Ομάδας Χρηστών

Λαμβάνοντας υπόψιν ένα σύνολο από χρήστες, καθένας από τους οποίους προσδιορίζει τις ατομικές του προτιμήσεις σε χαρακτηριστικά των αντικειμένων, μελετάμε το πρόβλημα της εύρεσης και της ταξινόμησης των αντικειμένων που θεωρούνται προτιμητέα από όλους τους χρήστες. Εισάγουμε και προτείνουμε μια αντικειμενική και δίκαιη ερμηνεία αυτού του προβλήματος, βασισμένη σε Pareto συνάθροιση (aggregation). Λαμβάνοντας υπόψιν αυτή την ερμηνεία, μελετάμε τρία σχετικά προβλήματα. Το πρώτο είναι η εύρεση των αντικειμένων που θεωρούνται ομόφωνα ιδανικά από όλο το σύνολο των χρηστών. Στο δεύτερο πρόβλημα, χαλαρώνεται η απαίτηση της ομοφωνίας και απαιτείτε μόνο ένα ποσοστό των χρηστών να συμφωνεί. Τέλος, στο τρίτο πρόβλημα, προτείνεται ένα αποτελεσματικό σχήμα ταξινόμησης (ranking scheme) βασισμένο σε Pareto συνάθροιση. Για να βελτιώσουμε την αποδοτικότητα όταν χειριζόμαστε κατηγορηματικά χαρακτηριστικά (categorical attributes), εισάγουμε έναν μετασχηματισμό κατηγορηματικών τιμών σε αριθμητικές, το οποίο επιδεικνύει μερικές χρήσιμες ιδιότητες επιτρέποντας την χρήση παραδοσιακών δομών δεικτοδότησης (index structures). Βασισμένοι σε αυτό τον μετασχηματισμό, προτείνουμε έναν αλγόριθμο βασισμένο σε ευρετήριο (index-based). Ο αλγόριθμος χρησιμοποιεί ένα ευρετήριο διαμερισμού χώρου (space partitioning index) προκειμένου να ταξινομήσει ιεραρχικά τα αντικείμενα. Σχετικά με το πρόβλημα της ταξινόμησης, μελετάμε θεωρητικά την συμπεριφορά του σχήματος ταξινόμησης και παρουσιάζουμε έναν αριθμό θεωρητικών ιδιοτήτων που ικανοποιούνται από την προσέγγισή μας. Επιπλέον, μελετήθηκαν μερικές ενδιαφέρουσες προεκτάσεις των προαναφερθέντων προβλημάτων που περιλαμβάνουν τα παρακάτω ζητήματα: ιδιότητες με πολλαπλές τιμές (multi-values attributes), μη-δεντροειδείς ιεραρχίες (non-tree hierarchies), δεικτοδότηση υποχώρων subspace indexing, και αντικειμενικά χαρακτηριστικά (objective attributes). Μια λεπτομερή πειραματική αξιολόγηση επιβεβαιώνει την αποδοτικότητα και την αποτελεσματικότητα των προτεινόμενων μεθόδων. Συγκεκριμένα, οι index- based τεχνικές είναι μίας τάξης μεγέθους γρηγορότερες από τις απλές προσεγγίσεις, κλιμακώνοντας σε εκατομμύρια αντικείμενα και χιλιάδες χρήστες. Τέλος, το προτεινόμενο σχήματος ταξινόμησης εμφανίζει καλύτερες επιδόσεις από τις παραδοσιακές rank aggregation μεθόδους σε σχέση με την ακρίβεια και την ανάκληση.




(α′) Χαρακτηριστικό Body (β′) Χαρακτηριστικό Engine

**Σχήμα 2.1:** Ιεραρχίες χαρακτηριστικών

**Πίνακας 2.1:** Αντικείμενα, Χρήστες & Διάνυσμα ταιριάσματος

| **Car** | **Body** | **Engine** |
|---|---|---|
| $o_1$ | family | hybrid 1.4 |
| $o_2$ | roadster | petrol 1.8 turbo |
| $o_3$ | SUV | diesel 1.6 |
| $o_4$ | compact | petrol 1.4 |

(α′) Αντικείμενα

| **User** | **Preference** |
|---|---|
| $u_1$ | {passenger cars, petrol} |
| $u_2$ | {sport cars} |
| $u_3$ | {petrol 1.8} |

(β′) Χρήστες

| | **User** | | |
|---|---|---|---|
| **Car** | $u_1$ | $u_2$ | $u_3$ |
| $o_1$ | $\langle 1/3, 0\rangle$ | $\langle 0, 0\rangle$ | $\langle 0, 0\rangle$ |
| $o_2$ | $\langle 0, 1/4\rangle$ | $\langle 1/2, 0\rangle$ | $\langle 0, 1/2\rangle$ |
| $o_3$ | $\langle 0, 0\rangle$ | $\langle 0, 0\rangle$ | $\langle 0, 0\rangle$ |
| $o_4$ | $\langle 1/3, 1/4\rangle$ | $\langle 0, 0\rangle$ | $\langle 0, 0\rangle$ |

(γ′) Διάνυσμα ταιριάσματος

## 2.1 Εισαγωγή

Θεωρώντας μια συλλογή από αντικείμενα καθώς και τις προτιμήσεις (preferences) ενός χρήστη, το γενικό πρόβλημα συστάσεων (recommendation), είναι να εντοπιστούν εκείνα τα αντικείμενα τα οποία ταιριάζουν περισσότερο με τις προτιμήσεις του χρήστη. Στην παρούσα εργασία μελετάμε ένα πρόβλημα συστάσεων που ονομάζεται πρόβλημα Πολλαπλών Κατηγορικών Προτιμήσεων (Multiple Categorical Preferences - MCP). Το MCP έχει τρία χαρακτηριστικά: (1) Τα αντικείμενα περιγράφονται από ένα σύνολο κατηγορικών χαρακτηριστικών. (2) Οι προτιμήσεις των χρηστών ορίζονται σε ένα υποσύνολο των χαρακτηριστικών. (3) Υπάρχουν πολλαπλοί χρήστες με διαφορετικές, και πιθανόν αντικρουόμενες, προτιμήσεις. Το MCP πρόβλημα μπορεί να εμφανιστεί σε διάφορα πραγματικά σενάρια, για παράδειγμα, η επιλογή εστιατορίου για τον προγραμματισμό δείπνου μεταξύ συναδέρφων, η επιλογή πακέτου διακοπών για μια παρέα φίλων, κτλ.

Για να περιγράψουμε το MCP πρόβλημα, παραθέτουμε το ακόλουθο παράδειγμα. Ας υποθέσουμε ότι μια τριμελής οικογένεια ψάχνει να αγοράσει ένα καινούργιο αυτοκίνητο. Ας υποθέσουμε ότι υπάρχει η λίστα των διαθέσιμων αυτοκινήτων, όπου καθένα τα

αυτοκίνητα χαρακτηρίζεται από δύο κατηγορικά χαρακτηριστικά, Body και Engine. Το Σχήμα 2.1 απεικονίζει τις ιεραρχίες για τα δύο αυτά χαρακτηριστικά, το Body είναι τριών επιπέδων, και το Engine είναι τεσσάρων επιπέδων. Ο Πίνακας 2.1 δείχνει τις τιμές των χαρακτηριστικών των τεσσάρων αυτοκινήτων, και τις προτιμήσεις των μελών της οικογένεια. Για παράδειγμα, το μέλος $u_1$ προτιμά τα επιβατικά αυτοκίνητα με βενζινοκινητήρες, ενώ $u_2$ αρέσκεται στα σπορ αυτοκίνητα, αλλά δεν εκφράζει καμία προτίμηση για το τον τύπο του κινητήρα.

Παρατηρούμε ότι αν κοιτάξουμε σε ένα συγκεκριμένο μέλος της οικογένειας, είναι απλό να καθορίσουμε το ιδανικό αυτοκίνητο με βάση τις υπάρχουσες μεθόδους. Για παράδειγμα, το μέλος $u_1$ προτιμά το $o_4$, το οποίο είναι ένα επιβατικό αυτοκίνητο με κινητήρα βενζίνης, ενώ το $u_2$ προτιμάει το $o_2$, το οποίο είναι ένα σπορτ αυτοκίνητο. Τα συμπεράσματα αυτά μπορούν να επιτευχθούν, με τον ακόλουθο συλλογισμό. Κάθε τιμή προτίμησης $u_j.A_k$ ταιριάζει με το αντίστοιχο χαρακτηριστικού $o_i.A_k$ χρησιμοποιώντας για παράδειγμα, την μετρική Jaccard $\frac{|o_i.A_k \cap u_j.A_k|}{|o_i.A_k \cup u_j.A_k|}$ για να υπολογίσουμε έναν βαθμό ταιριάσματος (matching degree). Λαμβάνοντας αυτούς του βαθμούς, το επόμενο βήμα είναι να συνδεθούν (compose) σε ένα συνολικό βαθμό ταιριάσματος μεταξύ ενός χρήστη $u_j$ ενός ένα αντικείμενο $o_i$. Για την σύνθεση των βαθμών ταιριάσματος έχουν προταθεί πολλές τεχνικές π.χ., [253, 252, 211, 310, 107]. Η απλούστερη λύση είναι να υπολογιστεί ένας γραμμικός συνδυασμός, πχ., το άθροισμα των επιμέρους βαθμών.

Όταν λαμβάνονται υπόψη όλοι οι χρήστες, όπως απαιτείται από το πρόβλημα MCP, διάφορα ερωτήματα προκύπτουν. Όπως, ποιο είναι το καλύτερο αυτοκίνητο που ικανοποιεί το σύνολο της οικογένεια. Και το πιο σημαντικό, τι σημαίνει το καλύτερο αυτοκίνητο. Μια απλή απάντηση στο τελευταίο θα ήταν, το αυτοκίνητο που έχει την υψηλότερη συλλογική βαθμολογία (collective matching) για όλους τους χρήστες. Χρησιμοποιώντας μια παρόμοια μέθοδο όπως και πριν, μπορεί κανείς να ορίσει μια συλλογική βαθμολογία που συνθέτει τον βαθμό ταιριάσματος του κάθε χρήστη. Ωστόσο η προσέγγιση αυτή επιβάλλει ένα πρόσθετο επίπεδο σύνθεσης, το πρώτο επίπεδο εφαρμόζεται στις προτιμήσεις κάθε χρήστη, ενώ η δεύτερη εφαρμόζεται ώστε να συνδυάσει όλους τους χρήστες. Μια τέτοια προσέγγιση, όμως, επισκιάζει τις ατομικές προτιμήσεις, και μπορεί να είναι άδικη, ευνοώντας τους χρήστες με συγκεκριμένες προτιμήσεις και τα αντικείμενα με λεπτομερείς περιγραφές.

Σε αυτή την εργασία, προτείνουμε μια αντικειμενική και δίκαιη προσέγγιση του MCP προβλήματος, η οποία βασίζεται σε δύο Pareto συναθροίσεις (aggregations). Σε αυτή την προσέγγιση, η αντιστοίχιση μεταξύ ενός χρήστη και ενός αντικειμένου σχηματίζει ένα διάνυσμα ταιριάσματος. Κάθε συντεταγμένη (coordinate) αυτού του διανύσματος αντιστοιχεί σε ένα χαρακτηριστικό και λαμβάνει την τιμή του αντίστοιχου βαθμό ταιριάσματος. Η πρώτη Pareto συνάθροιση ορίζεται πάνω στα χαρακτηριστικά και ορίζει μια μερική διάταξη (partial order) για τα διανύσματα αυτά. Διαισθητικά, για ένα συγκεκριμένο χρήστη, η πρώτη μερική διάταξη αντικειμενικά ορίζει ότι ένα αντικείμενο είναι καλύτερο από ένα άλλο (δηλαδή προτιμότερο από ένα άλλο).

Στη συνέχεια, η δεύτερη Pareto συνάθροιση, εφαρμόζεται για όλους του χρήστες, ορίζοντας μια δεύτερη μερική διάταξη πάνω στα αντικείμενα. Σύμφωνα με αυτή τη διάταξη, ένα αντικείμενο είναι καλύτερο από ένα άλλο, αν είναι προτιμότερο σύμφωνα με όλους τους χρήστες. Η λύση στο MCP πρόβλημα, είναι το σύνολο των ανώτατων αντικειμένων (maximal objects) κατά την δεύτερη μερική διάταξη.

Οι κύριες συνεισφορές αυτής της εργασίας συνοψίζονται ως εξής:

1. Προτείνουμε μια αντικειμενική και δίκαιη προσέγγιση του προβλήματος πολλα-

**Πίνακας 2.2:** Συμβολισμοί

| **Σύμβολο** | **Ορισμός** |
|---|---|
| $\mathcal{A}$, $d$ | Σύνολο χαρακτηριστικών, αριθμός χαρακτηριστικών ($\|\mathcal{A}\|$) |
| $A_k$, $\|A_k\|$ | Χαρακτηριστικό, αριθμός των τιμών στο $A_k$ |
| $\mathcal{H}(A_k)$, $\|\mathcal{H}(A_k)\|$ | Η ιεραρχία του $A_k$, αριθμός των κόμβων της ιεραρχίας |
| $\mathcal{O}$, $o_i$ | Σύνολο αντικειμένων, ένα αντικείμενο |
| $\mathcal{U}$, $u_j$ | Σύνολο χρηστών, ένας χρήστης |
| $o_i.A_k$, $u_j.A_k$ | Τιμή του χαρακτηριστικού $A_k$ στο αντικείμενο $o_i$, και στον χρήστη $u_j$ |
| $o_i.I_k$, $u_j.I_k$ | Διάστημα των τιμών του $A_k$ για $o_i$, $u_j$ |
| $m_i^j$ | Διάνυσμα ταιριάσματος του αντικειμένου $o_i$ για τον χρήστη $u_j$ |
| $m_i^j.A_k$ | Βαθμός ταιριάσματος του $o_i$ για τον χρήστη $u_j$ στο χαρακτηριστικό $A_k$ |
| $o_a \succ o_b$ | Το $o_a$ προτιμάται συλλογικά έναντι του $o_b$ |
| $M_i^j$ | Το μέγιστο διάνυσμα ταιριάσματος του $e_i$ για τον χρήστη $u_j$ |
| $M_i^j.A_k$ | Ο μέγιστος βαθμός ταιριάσματος του $e_i$ για τον χρήστη $u_j$ στο $A_k$ |

πλών κατηγορικών προτιμήσεων, βασιζόμενη σε Pareto συναθροίσεις.

2. Παρουσιάζουμε μια μέθοδο για τη μετατροπή του ιεραρχικού πεδίου ορισμού (hierarchical domain) ενός κατηγορικού χαρακτηριστικού, σε αριθμητικό πεδίο ορισμού (numerical domain). Με βάση αυτή τη μετατροπή, προτείνουμε έναν αλγόριθμο βασισμένο σε ευρετήριο (index-based). Συγκεκριμένα, ο προτεινόμενος αλγόριθμος χρησιμοποιεί ένα ευρετήριο διαμέρισης χώρου (space partitioning index) και ομαδοποιεί ιεραρχικά (hierarchically group) ομάδες αντικειμένων.

3. Προτείνουμε μια παραλλαγή του MCP προβλήματος, που ονομάζεται $p$- MCP, το οποίο χαλαρώνει την έννοια του Pareto, προκειμένου να ελέγχεται ο αριθμός των αντικειμένων που επιστρέφονται.

4. Προτείνουμε ένα σχήμα ταξινόμησης (ranking scheme) το οποίο βασίζεται στην έννοια του $p$-MCP. Επιπλέον, παρουσιάζουμε ένα σύνολο από θεωρητικές ιδιοκτήτες οι οποίες ικανοποιούνται από το προτεινόμενο σχήμα ταξινόμησης.

5. Παρουσιάζουμε διάφορες επεκτάσεις σχετιζόμενες με τα ακόλουθα θέματα: χαρακτηριστικά πολλαπλών τιμών (multi-values attributes), μη-δενδρικές ιεραρχίες (non-tree hierarchies), δεικτοδότηση υποχώρων (subspace indexing), και αντικειμενικά χαρακτηριστικά (objective attributes).

6. Διεξάγαμε εκτενή πειραματική αξιολόγηση χρησιμοποιώντας τόσο πραγματικά όσο και συνθετικά δεδομένα.

## 2.2 Το MCP Πρόβλημα

### 2.2.1 Ορισμοί

Ο Πίνακας 6.1 παρουσιάζει τα πιο σημαντικά σύμβολα και τους ορισμούς τους. Θεωρούμε ένα σύνολο από $d$ κατηγορηματικά χαρακτηριστικά $\mathcal{A} = \{A_1, \ldots, A_d\}$. Το πεδίο

ορισμού κάθε χαρακτηριστικού $A_k$ είναι μια ιεραρχία $\mathcal{H}(A_k)$. Μια ιεραρχία $\mathcal{H}(A_k)$ ορίζει ένα δέντρο, όπου ένα φύλλο αντιστοιχεί στη τιμή του χαμηλότερου επιπέδου, ενώ ένας εσωτερικός κόμβος αντιστοιχεί σε μια κατηγορία, δηλαδή, σε ένα σύνολο το οποίο περιλαμβάνει όλες τις τιμές μέσα στο υπο-δένδρο με ρίζα τον κόμβο αυτό. Η ρίζα μιας ιεραρχίας αντιπροσωπεύει την κατηγορία και περιλαμβάνει όλες τις τιμές των χαμηλότερων επιπέδων. Χρησιμοποιούμε το σύμβολο $|A_k|$ (αντίστοιχα $|\mathcal{H}(A_k)|$) για να δηλώσουμε τον αριθμό των φύλλων (αντίστοιχα όλους τους κόμβους της ιεραρχία).

Ας υποθέσουμε ένα σύνολο από αντικείμενα $\mathcal{O}$. Ένα αντικείμενο $o_i \in O$, ορίζεται πάνω σε όλα τα χαρακτηριστικά, και η τιμή του χαρακτηριστικού $o_i.A_k$ είναι ένας από τους κόμβους της ιεραρχίας $\mathcal{H}(A_k)$. Επιπλέον, ας υποθέσουμε ένα σύνολο από χρήστες $\mathcal{U}$. Ο χρήστης $u_i \in \mathcal{U}$ ορίζεται πάνω σε ένα υποσύνολο χαρακτηριστικών, για χαρακτηριστικό $u_i.A_j$ που ορίζεται, η τιμή του προέρχεται από έναν κόμβο της ιεραρχίας $\mathcal{H}(A_j)$.

Λαμβάνοντας υπόψη ένα αντικείμενο $o_i$, έναν χρήστη $u_j$, και ένα συγκεκριμένο χαρακτηριστικό $A_k$, ο *βαθμός ταιριάσματος* του $o_i$ και του $u_j$ σε σχέση με το $A_k$, συμβολίζεται με $m_i^j.A_k$, και καθορίζεται από μια συνάρτηση ταιριάσματος $\mathbf{M}$: $dom(A_k) \times dom(A_k) \rightarrow [0,1]$. Η συνάρτηση ταιριάσματος ορίζει την σχέση μεταξύ των προτιμήσεων του χρήστη και τις τιμές των χαρακτηριστικών των αντικειμένων.

Λαμβάνοντας υπόψη ένα αντικείμενο $o_i$ και έναν χρήστη $u_j$, το διάνυσμα ταιριάσματος των $o_i$ και $u_j$, συμβολίζεται ως $m_i^j$ και είναι ένα $d$-διάστατο σημείο στο $[0,1]^d$. Επιπλέον, ορίζουμε την νόρμα του διανύσματος ταιριάσματος να είναι $\|m_i^j\| = \sum_{A_k \in \mathcal{A}} m_i^j.A_k$. Στο παράδειγμά μας, το διάνυσμα ταιριάσματος του αυτοκινήτου $o_1$ σε χρήστη με τον $u_1$ είναι $\langle 1/3, 0 \rangle$. Όλα τα διανύσματα ταιριάσματος του παραδείγματος φαίνονται στον Πίνακα 2.1 γ.

Στη συνέχεια, επιλέγουμε ένα συγκεκριμένο χρήστη $u_j$ και εξετάζουμε το διάνυσμα ταιριάσματος του. Η πρώτη Pareto συνάθροιση εφαρμόζεται σε όλες τις τιμές του διανύσματος ταιριάσματος, ορίζοντας μια μερική και αυστηρά μερική διάταξη. Ένα αντικείμενο $o_a$ είναι προτιμότερο από ένα αντικείμενο $o_b$, για τον χρήστη $u_j$, και συμβολίζεται ως $o_a \succeq^j o_b$, αν και μόνο αν, για κάθε χαρακτηριστικό $A_k$ ισχύει ότι $m_a^j.A_k \geq m_b^j.A_k$. Επιπλέον, ένα αντικείμενο $o_a$ είναι αυστηρά προτιμότερο από ένα αντικείμενο $o_b$, για τον χρήστη $u_j$, και συμβολίζεται ως $o_a \succeq^j o_b$, αν και μόνο αν, για κάθε καθορισμένο χαρακτηριστικό $A_k$ ισχύει ότι $m_a^j.A_k \geq m_b^j.A_k$.

Θεωρούμε τώρα σε όλους τους χρήστες στο $\mathcal{U}$. Η δεύτερη Pareto συνάθροιση εφαρμόζεται για όλους τους χρήστες, ορίζοντας μια αυστηρά μερική διάταξη. Ένα αντικείμενο $o_a$ είναι συλλογικά προτιμότερο έναντι του $o_b$, αν το $o_a$ προτιμάται έναντι του $o_b$ από όλους τους χρήστες, και υπάρχει ένα χρήστης $u_j$ για τον οποίο το $o_a$ είναι αυστηρά προτιμότερο έναντι του $o_b$. Τα συλλογικά ανώτατα αντικείμενα από τα $\mathcal{O}$ λαμβάνοντας υπόψη τους χρήστες $\mathcal{U}$, είναι τα αντικειμένων για τα οποία δεν υπάρχει αντικείμενο που να προτιμάται συλλογικά έναντι αυτών.

Στην συνέχεια ορίζουμε το MCP πρόβλημα.

**Πρόβλημα 1.** **[MCP]** Λαμβάνοντας υπόψη ένα σύνολο αντικειμένων $\mathcal{O}$ και ένα σύνολο χρηστών $\mathcal{U}$, το πρόβλημα των πολλαπλών κατηγορικών προτιμήσεων είναι να βρεθούν τα συλλογικά μέγιστα αντικείμενα του $\mathcal{O}$ σε σχέση με τους χρήστες $\mathcal{U}$.

**Algorithm 1.** BSL

**Input**: objects $\mathcal{O}$, users $\mathcal{U}$
**Output**: $CM$ the collectively maximal
**Variables**: $\mathcal{R}$ set of intermediate records

```
1 foreach o_i ∈ O do
2     foreach u_j ∈ U do
3         compute m_i^j
4         r_i[j] ← m_i^j
5     insert r_i into R
6 CM ← POSkylineAlgo (R)
```

### 2.2.2 Βασικός Αλγόριθμος

Το MCP πρόβλημα μπορεί να μετατραπεί σε πρόβλημα μέγιστων αντικειμένων (maximal objects problem) ή σε ένα ερώτημα κορυφογραμμής (skyline query), όπου τα στοιχεία εισόδου είναι τα διανύσματα ταιριάσματος. Ωστόσο, να σημειωθεί ότι το MCP πρόβλημα είναι διαφορετικό από τον υπολογισμό του συμβατικού ερωτήματος κορυφογραμμής.

Ο ψευδοκώδικας της βασικής μέθοδού (BSL) απεικονίζεται στον Αλγόριθμο 1. Το υπολογιστικό κόστος της BSL είναι το άθροισμα δύο τμημάτων. Το πρώτο τμήμα υπολογίζει τους βαθμούς ταιριάσματος και απαιτεί $O(|\mathcal{O}|\cdot|\mathcal{U}|)$ χρόνο. Το δεύτερο τμήμα υπολογίζει την κορυφογραμμή και απαιτεί $O(|\mathcal{O}|^2\cdot|\mathcal{U}|\cdot d)$, υποθέτοντας έναν αλγόριθμο κορυφογραμμής τετραγωνικής (quadratic) πολυπλοκότητας. Επομένως, ο BSL απαιτεί $O(|\mathcal{O}|^2\cdot|\mathcal{U}|\cdot d)$ χρόνο.

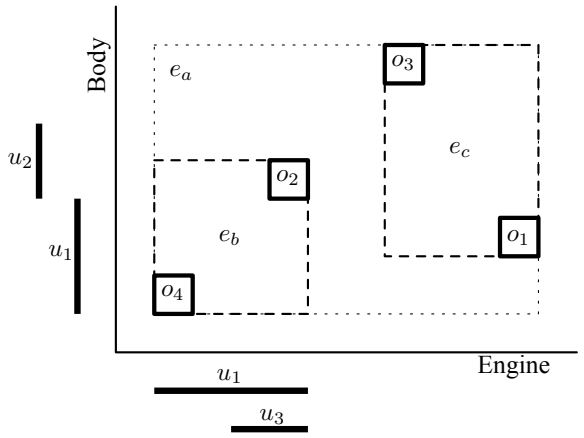
### 2.2.3 Αλγόριθμος Βασισμένος σε Ευρετήρια

Αυτή η ενότητα παρουσιάζει τον αλγόριθμο ο οποίος βασίζεται σε ευρετήρια (IND). Οι βασικές ιδέες του IND είναι: (1) εφαρμόζει τον μετασχηματισμό της ιεραρχία που περιγράφηκε προηγουμένως, και ευρετηριοποιεί τα διαστήματα τα οποία προκύπτουν, (2) ορίζει και κάνει χρήση άνω φραγμάτων (upper bounds) για τους βαθμούς ταιριάσματος, με αυτό τον τρόπο, κατευθύνει την αναζήτηση και αποκρύπτει γρήγορα αντικείμενα.

Υποθέτουμε ένα σύνολο αντικειμένων $\mathcal{O}$ και ένα σύνολο χρηστών $\mathcal{U}$ που μετασχηματίζονται έτσι ώστε κάθε χαρακτηριστικό $A_k$ να έχει τιμή από ένα διάστημα (interval) $I_k$. Ως εκ τούτου, κάθε αντικείμενο (και χρήστη) ορίζει ένα υπερ-ορθογώνιο που ορίζεται από το $d$-διάστατο καρτεσιανό γινόμενο, δηλαδή $[0,|A_1|)\times\cdots\times[0,|A_d|)$.

Το Σχήμα 2.2 απεικονίζει τον μετασχηματισμό των αντικειμένων και των χρηστών που παρουσιάζονται στον Πίνακα 2.1 και τις ιεραρχίες στο Σχήμα 2.1. Για παράδειγμα, το αντικείμενο $o_1$ παρουσιάζεται σαν το ορθογώνιο $[9,10)\times[1,2)$. Ομοίως, ο χρήστης $u_1$ παρουσιάζεται ως δύο διαστήματα, $[0,4)$, $[0,3)$.

Ο αλγόριθμος IND δεικτοδοτεί το σύνολο των αντικειμένων σε έναν $d$-διάστατο χώρο. Συγκεκριμένα, ο IND χρησιμοποιεί ένα R*-Tree $\mathcal{T}$ [48], το οποίο είναι κατάλληλο για δεικτοδότηση ορθογώνιων. Κάθε κόμβος του $\mathcal{T}$ αντιστοιχεί σε μια σελίδα στο δίσκο, και περιέχει μια σειρά καταχωρήσεων. Κάθε καταχώρηση $e_i$ περιλαμβάνει (1) έναν δείκτη $e_i.ptr$, και (2) ένα ελάχιστο ορθογώνιο (Minimum Bounding Rectangle-MBR) $e_i.mbr$. Μια καταχώρηση φύλλου $e_i$ αντιστοιχεί σε ένα αντικείμενο $o_i$, και ο δείκτη του $o_i.ptr$ είναι *null*, το $e_i.mbr$ είναι το ορθογώνιο που ορίζεται από τα διαστήματα του $o_i$. Ένα μη-φύλλο $e_i$ αντιστοιχεί σε ένα κόμβο-παιδί $N_i$, ο δείκτη του $e_i.ptr$ περιέχει τη διεύθυνση του $N_i$, και το $e_i.mbr$ είναι το MBR των MBRs που ορίζουν οι




**Σχήμα 2.2:** Μετασχηματισμός Αντικειμένων και Χρηστών

καταχωρίσεις του $N_i$.

Ο Αλγόριθμος 2 παρουσιάζει τον ψευδοκώδικα για τον IND. Ο IND πραγματοποιεί στη χειρότερη περίπτωση $O(|\mathcal{O}|^2 \cdot |\mathcal{U}| \cdot d)$ συγκρίσεις, και υπολογίζει τους βαθμούς ταιριάσματος με κόστος $O(|\mathcal{O}| \cdot |\mathcal{U}|)$. Συνολικά, ο IND απαιτεί $O(|\mathcal{O}|^2 \cdot |\mathcal{U}| \cdot d)$ χρόνο, το ίδιο όπως ο BSL. Ωστόσο, στην πράξη ο IND είναι κατά μια τάξη μεγέθους ταχύτερος από τον BSL.

## 2.3 Το $p$-Multiple Categorical Preference ($p$-MCP) Πρόβλημα

Η Ενότητα 2.3.1 εισάγει το $p$-MCP πρόβλημα, η Ενότητα 2.3.2 παρουσιάζει μια προσαρμογή της μεθόδου BSL, καθώς η Ενότητα 2.3.3 εισάγει μια index-based προσέγγιση.

### 2.3.1 Ορισμός Προβλήματος

Καθώς ο αριθμός των χρηστών αυξάνεται, γίνεται πιθανότερο οι χρήστες να εκφράσουν πολύ διαφορετικές και αντιφατικές προτιμήσεις. Έτσι, γίνεται δύσκολο να βρεθεί ένα ζευγάρι αντικειμένων τέτοιο ώστε οι χρήστες να συμφωνούν ποιο είναι το χειρότερο. Εν τέλει, ο αριθμός των πιθανότερα προτιμώμενων αντικειμένων αυξάνεται. Αυτό σημαίνει ότι η απάντηση σε ένα MCP πρόβλημα με μεγάλο αριθμό χρηστών χάνει σιγά σιγά το νόημα της.

Η ρίζα αυτού του προβλήματος είναι ότι απαιτούμε ομοφωνία για το αν το αντικείμενο είναι προτιμητέο για το σύνολο των χρηστών. Ο ακόλουθος ορισμός μας απαλλάσσει από αυτή την απαίτηση. Ένα αντικείμενο $o_a$ είναι *p-collectively prefered* σε σχέση με το $o_b$ και συμβολίζεται ως $o_a \succ_p o_b$, αν και μόνο αν υπάρχει ένα υποσύνολο $\mathcal{U}_p \subseteq \mathcal{U}$, τουλάχιστον $\lceil \frac{p}{100} \cdot |\mathcal{U}| \rceil$ χρηστών τέτοιο ώστε για κάθε χρήστη $u_i \in \mathcal{U}_p$ $o_a$ να είναι προτιμώμενο σε σχέση με το $o_b$, και υπάρχει ένας χρήστης $u_j \in \mathcal{U}_p$ για τον οποίο το $o_a$ είναι αυστηρά προτιμώμενο σε σχέση με το $o_b$. Με άλλα λόγια, απαιτούμε μόνο $p\%$ των χρηστών να αποφασίσουν αν ένα αντικείμενο είναι collectively prefered. Ομοίως, τα *p-collectively maximal αντικείμενα* των $\mathcal{O}$ με απήχηση στους χρήστες $\mathcal{O}$, ορίζεται ως η ομάδα των αντικειμένων στο $\mathcal{O}$ για τα οποία δεν υπάρχει άλλο αντικείμενο που είναι $p$-collectively maximal σε σχέση με αυτά. Οι παραπάνω ορισμοί δίνουν ώθηση

**Algorithm 2.** IND

**Input**: R*-Tree $\mathcal{T}$, users $\mathcal{U}$
**Output**: $CM$ the collectively maximal
**Variables**: $H$ a heap with $\mathcal{T}$ entries sorted by $score()$

```
1  CM ← ∅
2  read T root node
3  insert in H the root entries
4  while H is not empty do
5      e_x ← pop H
6      if e_x is non-leaf then
7          N_x ← read node e_x.ptr
8          foreach e_i ∈ N_x do
9              pruned ← false
10             foreach u_j ∈ U do
11                 compute M_i^j
12             foreach o_a ∈ CM do
13                 if ∀A_j : m_a^j ≥ M_i^j ∧ ∃A_k : m_a^k ≻ M_i^k then
14                     pruned ← true
15                     break
16             if not pruned then
17                 insert e_i in H
18     else
19         o_x ← e_x
20         result ← true
21         foreach o_a ∈ CM do
22             if o_a ≻ o_x then
23                 result ← false
24                 break
25         if result then
26             insert o_x in CM
```

στο $p$-MCP πρόβλημα.

**Πρόβλημα 2.** [$p$-**MCP**] Δοθείσας μιας ομάδας αντικειμένων $\mathcal{O}$ και μιας ομάδας χρηστών $\mathcal{U}$ ορισμένα σε κάποια κατηγορικά γνωρίσματα $\mathcal{A}$, το *p-Multiple Categorical Preference Problem* ($p$-MCP) είναι να βρεθούν τα $p$-collectively prefered αντικείμενα του $\mathcal{O}$ με βάση τους $\mathcal{U}$.

Ακολουθώντας τους ορισμούς, μπορούμε να κάνουμε κάποιες παρατηρήσεις, παρόμοιες με αυτές στην $k$-dominance έννοια [97]. Πρώτον, αν ένα αντικείμενο είναι collectively preferred σε σχέση με κάποιο άλλο αντικείμενο, είναι επίσης $p$-collectively preferred σε σχέση με αυτό το αντικείμενο για κάθε $p$. Ως αποτέλεσμα αυτού, ένα αντικείμενο που είναι $p$-collectively maximal είναι επίσης collectively maximal για κάθε $p$. Με άλλα λόγια, η απάντηση στο $p$-MCP πρόβλημα είναι ένα *υποσύνολο* της απάντησης στο αντίστοιχο MCP.

Δεύτερον, θεωρήστε ένα αντικείμενο $o$ που δεν είναι $p$-collectively maximal. Θα μπορούσε, κανένα από τα $p$-collectively maximal αντικείμενα να μην είναι $p$- collectively preferred σε σχέση με το $o$. Ως αποτέλεσμα αυτού, το να δούμε αν το $o$ είναι ένα αποτέλεσμα, λαμβάνοντας υπόψιν μας μόνο τα $p$- collectively maximal αντικείμενα θα μπορούσε να οδηγήσει σε λανθασμένα συμπεράσματα. Ισχύει όμως ότι πρέπει να υπάρχει ένα collectively maximal αντικείμενο που είναι $p$-collectively preferred σε σχέση με το $o$. Άρα αρκεί να εξετάσουμε το $o$ μόνο σε σχέση με τα collectively maximal αντικείμενα, (και όχι απλά το υποσύνολο που είναι $p$-collectively maximal).

### 2.3.2 Βασικός Αλγόριθμος ($p$-BSL)

Βασιζόμενοι στις παραπάνω παρατηρήσεις, περιγράφουμε ένα βασικό αλγόριθμο για το $p$-MCP πρόβλημα, βασιζόμενο στον BSL. Ο Αλγόριθμος 3 δείχνει τις αλλαγές με βάση τον BSL αλγόριθμο. Όλες οι παραλειπόμενες γραμμές είναι ίδιες με αυτές στον Αλγόριθμος 3. Ο $p$-BSL αλγόριθμος αρχικά υπολογίζει τα collectively maximal αντικείμενα εφαρμόζοντας τον BSL (*lines 1–6*). Έπειτα, κάθε collectively maximal αντικείμενο, συγκρίνεται με όλα τα άλλα collectively maximal αντικείμενα (*lines 7–14*). Συγκεκριμένα, για κάθε αντικείμενο $o_i$, εξετάζουμε αν υπάρχει άλλο αντικείμενο στο $CM$, που είναι $p$- collectively preferred σε σχέση με το $o_i$ (*lines 10–12*). Αν δεν υπάρχει τέτοιο αντικείμενο, το αντικείμενο $o_i$ εισάγεται στο $p$-$CM$ (*line 14*). όταν ο αλγόριθμος τερματίσει, το σύνολο $p$-$CM$ περιέχει τα $p$-collectively maximal αντικείμενα.

**Υπολογιστή ανάλυση** Αρχικά, ο αλγόριθμος υπολογίζει το collectively maximal σύνολο χρησιμοποιώντας τον BSL αλγόριθμο (*lines 1–6*), ο οποίος απαιτεί $O(|\mathcal{O}|^2 \cdot |\mathcal{U}| \cdot d)$. Ύστερα, βρίσκει τα $p$- collectively maximal αντικείμενα (*lines 7–14*), πραγματοποιώντας στη χειρότερη περίπτωση $O(|\mathcal{O}|^2)$ συγκρίσεις. Άρα στη χειρότερη περίπτωση έχουμε ότι $|CM| = |\mathcal{O}|$. Ως εκ τούτου, το υπολογιστικό κόστος του αλγορίθμου είναι $O(|\mathcal{O}|^2 \cdot |\mathcal{U}| \cdot d)$.

### 2.3.3 Αλγόριθμος Βασισμένος σε Ευρετήρια ($p$-IND)

Επίσης προτείνουμε μια επέκταση του IND για το $p$-MCP πρόβλημα, τον $p$-IND. Ο Αλγόριθμος 4 δείχνει τις αλλαγές με βάση τον IND αλγόριθμο.Όλες οι παραλειπόμενες γραμμές είναι παρόμοιες με αυτές του Αλγόριθμος 4.

**Algorithm 3.** $p$-BSL

**Input**: objects $\mathcal{O}$, users $\mathcal{U}$
**Output**: $p$-$CM$ the $p$-collectively maximal
**Variables**: $CM$ the collectively maximal

⋮
**foreach** $o_i \in CM$ **do**
  $inpCM \leftarrow true$
  **foreach** $o_j \in CM \backslash o_i$ **do**
    **if** $o_j \succ_p o_i$ **then**
      $inpCM \leftarrow false$
      **break**;
  **if** $inpCM$ **then**
    insert $o_i$ to $p$-$CM$

Συμπληρωματικά στο σύνολο $CM$, ο $p$-IND, διατηρεί το σύνολο $p$-$CM$ των $p$-collectively maximal αντικειμένων που ανακαλύψαμε μέχρι εδώ (*line 1*). Ισχύει ότι $p$-$CM \subseteq CM$. Έτσι ένα αντικείμενο μπορεί να εμφανίζεται και στα δύο σύνολα. Όταν εμφανίζεται μια leaf entry $o_x$ (*line 19*), συγκρίνεται με κάθε αντικείμενο $o_a$ στο $CM$ (*lines 21–30*) σε τρεις ελέγχους. Πρώτα, ο αλγόριθμος εξετάζει αν το $o_a$ είναι collectively preferred σε σχέση με το $o_x$ (*lines 22–24*). Σε αυτή την περίπτωση, το αντικείμενο $o_x$ δεν είναι στο $CM$ και έτσι ούτε στο $p$-$CM$. Δεύτερον, εξετάζει αν το $o_a$ είναι $p$-collectively preferred σε σχέση με το $o_x$ (*lines 25–27*). Σε αυτή την περίπτωση, το αντικείμενο $o_x$ δεν είναι στο $p$-$CM$, αλλά είναι στο $CM$. Τρίτον, ο αλγόριθμος εξετάζει αν το αντικείμενο $o_x$ είναι $p$-collectively preferred σε σχέση με το $o_a$ (*lines 28–30*). Σε αυτή την περίπτωση, το αντικείμενο $o_a$ μετακινείται από το $p$-collectively maximal αντικείμενα (*line 30*), αλλά παραμένει στο $CM$.

Μετά τους τρεις ελέγχους, αν το $o_x$ είναι collectively maximal (*line 31*) εισάγεται στο $CM$ (*line 32*). Στη συνέχεια, αν το $o_x$ είναι p-collectively maximal (*line 33*), εισάγεται και στο $p$-$CM$ (*line 34*). Όταν ο $p$- IND αλγόριθμος τερματίσει, το σύνολο $p$-$CM$ περιέχει την απάντηση στο $p$-MCP πρόβλημα.

**Υπολογιστική Ανάλυση.** $p$-IND πραγματοποιεί το πολύ τρεις φορές περισσότερες συγκρίσεις μεταξύ αντικείμενων, σε σχέση με τον IND. Έτσι η περιπλοκότητα εκτέλεσης του παραμένει $O(|\mathcal{O}|^2 \cdot |\mathcal{U}| \cdot d)$.

## 2.4 Το Group-Ranking Categorical Objects (GRCO) Πρόβλημα

Η Ενότητα 2.4.1 εισάγει το GRCO πρόβλημα, η Ενότητα2.4.2 περιγράφει έναν αλγόριθμο για τα GRCO. Στη συνέχεια η Ενότητα 2.4.3 συζητούνται κάποιες θεωρητικές ιδιότητες του προτεινόμενου σχήματος ταξινόμησης.

### 2.4.1 Ορισμός Προβλήματος

Όπως αναφέραμε στην Ενότητα 2.1, θα μπορούσε να οριστεί μια κατάταξη ανάμεσα στα αντικείμενα συνθέτοντας τους βαθμούς συσχέτισης για όλους τους χρήστες. Ωστόσο, οποιαδήποτε μια τέτοια κατάταξη θα ταν άδικη, καθώς δεν υπάρχει αντικειμενικός τρόπος να συναθροίσουμε τους βαθμούς συσχέτισης. Αντιθέτως, στην προσέγγιση

**Algorithm 4.** $p$-IND

**Input**: R*-Tree $\mathcal{T}$, users $\mathcal{U}$
**Output**: $p$-$CM$ the $p$-collectively maximal
**Variables**: $H$ a heap with $\mathcal{T}$ entries sorted by $score()$, $CM$ the collectively maximal object

```
1  CM ← ∅; p-CM ← ∅
   ⋮
4  while H is not empty do
   |  ⋮
18 |  else
19 |  |  o_x ← e_x
20 |  |  inCM ← true; inpCM ← true
21 |  |  foreach o_a ∈ CM do
22 |  |  |  if o_a ≻ o_x then
23 |  |  |  |  inCM ← false
24 |  |  |  |  break
25 |  |  |  if inpCM then
26 |  |  |  |  if o_a ≻_p o_x then
27 |  |  |  |  |  inpCM ← false
28 |  |  |  if o_a ∈ p-CM then
29 |  |  |  |  if o_x ≻_p o_a then
30 |  |  |  |  |  remove o_a from p-CM
31 |  |  if inCM then
32 |  |  |  insert o_x to CM
33 |  |  |  if inpCM then
34 |  |  |  |  insert o_x to p-CM
```

μας ακολουθείται μια αντικειμενική μέθοδος κατάταξης, βασισμένη στην έννοια του $p$-collectively preference. Η συγκεκριμένη κατάταξη δεν είναι αυστηρή, με την έννοια ότι μπορούν αντικείμενα να λαμβάνουν την ίδια θέση. Ορίζουμε την *θέση* ενός αντικειμένου $o$ ως το μικρότερο ακέραιο $\tau$, όπου $1 \leq \tau \leq |\mathcal{U}|$, τέτοιο ώστε $o$ είναι $p$- collectively maximal για κάθε $p \geq \frac{\tau}{|\mathcal{U}|} \cdot 100$. Τα μη collectively maximal αντικείμενα παίρνουν τη μικρότερη δυνατή θέση $|\mathcal{U}| + 1$. Διαισθητικά,, η θέση $\tau$ για ένα αντικείμενο $o$ σημαίνει ότι κάθε ομάδα $\mathcal{U}' \subseteq \mathcal{U}$ από τουλάχιστον $\tau$ χρήστες (δηλαδή, $|\mathcal{U}'| \geq \tau$) θα θεωρούσε το $o$ προτιμητέο, δηλαδή, το $o$ θα ήταν collectively maximal γι αυτούς τους $\mathcal{U}'$ χρήστες. Στην υψηλότερη θέση 1, ένα αντικείμενο $o$, προτιμάται από κάθε χρήστη, δηλαδή, το $o$ εμφανίζεται σε όλα τα πιθανά σύνολα $p$-collectively maximal αντικειμένων.

**Πρόβλημα 3.** **[GRCO]** Δοθέντος ενός συνόλου αντικειμένων $\mathcal{O}$ και ενός συνόλου χρηστών $\mathcal{U}$ ορισμένα σε ένα σύνολο από categorical attributes $\mathcal{A}$, το *Group-Ranking Categorical Objects (GRCO)* πρόβλημα είναι να βρεθεί η ταξινόμηση όλων των collectively maximal αντικειμένων του $\mathcal{O}$ με βάση τους $\mathcal{U}$.

### 2.4.2 Αλγόριθμος Ταξινόμησης (RANK-CM)

Ο Αλγόριθμος 5 (RANK-CM), υπολογίζει την κατάταξη όλων των collectively maximal αντικειμένων. Ο αλγόριθμος δέχεται ως είσοδο, τα collectively maximal αντικείμενα $CM$, καθώς και τον αριθμό των χρηστών $|\mathcal{U}|$. Αρχικά, σε κάθε αντικείμενο αποδίδεται, η υψηλότερη θέση στην κατάταξη, δηλαδή, $rank(o_i) \leftarrow 1$ (*line 2*). Έπειτα, κάθε αντικείμενο, συγκρίνεται με όλα τα άλλα στο $CM$ (*loop in line 3*). Μέσα από τις

συγκρίσεις των αντικειμένων, αυξάνουμε $\tau$ (*lines 5–11*) από την τωρινή θέση δηλαδή, $\text{rank}(o_x)$) (*line 4*) μέχρι την $|\mathcal{U}|$. Άν το $o_i$ δεν είναι $p$-collectively maximal (*line 7*), για $p = \frac{\tau}{|\mathcal{U}|} \cdot 100$ (*line 6*), τότε το $o_x$ δεν μπορεί να είναι στην $p$-$CM$ και μπορεί μόνο να έχει θέση το πολύ $\tau + 1$ (*line 8*). Τελικά κάθε αντικείμενο εισάγεται στο $rCM$ με βάση τη κατάταξή του (*line 12*).

**Υπολογιστική ανάλυση.** Ο αλγόριθμος συγκρίνει κάθε collective maximal αντικείμενο με όλα τα άλλα collective maximal αντικείμενα. Ανάμεσα σε δύο αντικείμενα ο αλγόριθμος κάνει το πολύ $|\mathcal{U}|-1$ συγκρίσεις. Από τη στιγμή λοιπόν που στη χειρότερη περίπτωση έχουμε ότι $|CM| = |\mathcal{O}|$, το υπολογιστικό κόστος του Αλγόριθμου 5 είναι $O(|\mathcal{O}|^2 \cdot |\mathcal{U}|)$.

**Algorithm 5.** RANK-CM

**Input**: $CM$ the collectively maximal objects, $|\mathcal{U}|$ the number of users
**Output**: $rCM$ the ranked collectively maximal objects

```
1  foreach o_i ∈ CM do
2      rank(o_i) ← 1
3      foreach o_j ∈ CM\o_i do
4          τ ← rank(o_i)
5          while τ ≤ |U| − 1 do
6              p ← (τ / |U|) · 100
7              if o_j ≻_p o_i then
8                  rank(o_i) = τ + 1
9              else
10                 break;
11             τ ← τ + 1
12     insert o_i in rCM at rank(o_i)
```

### 2.4.3 Ιδιότητες Ταξινόμησης

Σε αυτή την ενότητα, συζητάμε μερικές θεωρητικές ιδιότητες στο γενικό πλαίσιο του προβλήματος αθροιστικής ταξινόμησης rank aggregation problem. Αυτές οι ιδιότητες, χρησιμοποιούνται ευθέως στη voting theory ως κριτήρια αξιολόγησης για την ¨ορθότητα' ενός voting system [320, 35, 285]. Δείχνουμε ότι το προτεινόμενο σχήμα ταξινόμησης, ικανοποιεί αρκετές από αυτές τις ιδιότητες.

**Ιδιότητα 1.** [**Πλειοψηφία**] Εάν ένα αντικείμενο είναι αυστηρά προτιμώμενο, σε σχέση με όλα τα άλλα αντικείμενα από την *πλειοψηφία* των χρηστών, τότε αυτό το αντικείμενο κατατάσσεται πάνω από όλα τα άλλα αντικείμενα.

**Ιδιότητα 2.** [**Ανεξαρτησία των ασυσχέτιστων εναλλακτικών (Independence of Irrelevant Alternatives)**] Η θέση κάθε αντικειμένου δεν επηρεάζεται αν εισάγονται ή εξάγονται non-collectively maximal αντικείμενα.

**Ιδιότητα 3.** [**Ανεξαρτησία των παρόμοιων εναλλακτικών**] Η θέση κάθε αντικειμένου δεν επηρεάζεται αν εισέρχονται non- collectively maximal αντικείμενα παρόμοια με ένα υπάρχον αντικείμενο

**Ιδιότητα 4.** [**Ισότητα των χρηστών**] Το αποτέλεσμα θα παραμείνει το ίδιο αν δυο χρήστες αλλάξουν τις προτιμήσεις τους. Αυτή η ιδιότητα είναι επίσης γνωστή

ως *Ανωνυμία*.

**Ιδιότητα 5.** [**Μονοτονία**] Αν ένα αντικείμενο $o_a$ κατατάσσεται πάνω από ένα άλλο αντικείμενο $o_b$, και ένας χρήστης αυξάνει το ενδιαφέρον του για το $o_a$, τότε το $o_a$ διατηρεί τη θέση του πάνω από το $o_b$.

**Ιδιότητα 6.** [**Συμμετοχή**] *1η Εκδοχή:* Αν ένα αντικείμενο $o_a$ κατατάσσεται πάνω από ένα αντικείμενο $o_b$, τότε αν προσθέσουμε ένα ή παραπάνω χρήστες, που αυστηρά προτιμούν το $o_a$ σε σχέση με τα άλλα αντικείμενα, το αντικείμενο $o_a$ διατηρεί τη θέση του πάνω από το $o_b$.

*2η εκδοχή:* Υποθέτουμε ένα αντικείμενο $o_a$ που κατατάσσεται πάνω από ένα αντικείμενο $o_b$, και ότι υπάρχει τουλάχιστον ένας χρήστης $u \in \mathcal{U}$ που δεν έχει δηλώσει καμία προτίμηση. Τότε αν ο $u$ εκφράσει ότι προτιμά αυστηρά το $o_a$ σε σχέση με όλα τα άλλα αντικείμενα, το αντικείμενο $o_a$ διατηρεί τη θέση του πάνω από το $o_b$.

**Ιδιότητα 7.** [**Πρόσμιξη**] *1η Εκδοχή:* Αν δύο αντικείμενα κατατάσσονται στην ίδια θέση, η πρόσθεση ενός καινούργιου χρήστη, μπορεί να προκαλέσει την άνοδο του ενός πάνω από το άλλο.

*2η εκδοχή:* Υποθέτουμε ότι δύο αντικείμενα κατατάσσονται στην ίδια θέση, και ότι υπάρχει τουλάχιστον ένας χρήστης $u$ που δεν έχει δηλώσει τις προτιμήσεις του. Αν ο $u$ εκφράσει προτιμήσεις, μπορεί να προκληθεί η άνοδος ενός αντικειμένου πάνω από το άλλο.

**Ιδιότητα 8.** [**Ουδετερότητα των προτιμήσεων του χρήστη**] Οι χρήστες με διαφορετικό αριθμό προτιμήσεων ή διαφορετική λεπτομέρεια προτιμήσεων preference granularity είναι το ίδιο "σημαντικοί".

**Ιδιότητα 9.** [**Ουδετερότητα της περιγραφής των αμτικειμένων**] Αντικείμενα με διαφορετική λεπτομέρεια περιγραφής description granularity (δηλ. τιμές χαρακτηριστικών) είναι το ίδιο "σημαντικά".

Οι αποδείξεις των παραπάνω μπορούν να βρεθούν στο [59].

## 2.5 Πειραματική Ανάλυση

Η Ενότητα 2.5.1 περιγράφει τα σύνολα δεδομένων που χρησιμοποιούνται για την αξιολόγηση. Η Ενότητες 2.5.2 και 2.5.3 εξετάζουν την αποτελεσματικότητα των αλγορίθμων MCP και $p$-MCP αντίστοιχα. Τέλος, στην Ενότητα 2.5.4 ερευνούμε την αποτελεσματικότητα της ταξινόμησης στο GRCO πρόβλημα.

### 2.5.1 Datasets & Προτιμήσεις χρηστών

Χρησιμοποιούμε πέντε σύνολα δεδομένων, ένα συνθετικό και τέσσερις πραγματικά. Το πρώτο είναι συνθετικό (Synthetic Dataset), στο οποίο τα αντικείμενα και οι χρήστες δημιουργούνται συνθετικά. Πιο συγκεκριμένα, όλα τα γνωρίσματα έχουν την ίδια ιεραρχία, ένα δυαδικό δένδρο ύψους $\log|A|$, και ως εκ τούτου όλα τα χαρακτηριστικά έχουν τον ίδιο αριθμό φύλλων $|A|$. Για να δημιουργήσουμε το σύνολο των αντικειμένων, καθορίζουμε ένα επίπεδο, $\ell_o$ (όπου $\ell_o = 1$ αντιστοιχεί στα φύλλα). Στη συνέχεια, επιλέγουμε τυχαία κόμβους από αυτό το επίπεδο. Ο αριθμός των αντικειμένων συμβολίζεται ως $|\mathcal{O}|$, ενώ ο αριθμός των χαρακτηριστικά για κάθε αντικείμενο συμβολίζεται με $d$. Ομοίως, για να αποκτήσει το σύνολο των χρηστών, καθορίζουμε ένα επίπεδο $\ell_u$. Ο αριθμός των χρηστών συμβολίζεται με $|\mathcal{U}|$.

Το δεύτερο σύνολο δεδομένων είναι το RestaurantsF, το οποίο περιέχει 85,681 αμερικάνικα εστιατόρια από το site Factual[1]. Θεωρούμε *τρία* κατηγορικά χαρακτηριστικά , *Κουζίνα*, *Εμφάνιση* και *Πάρκινγκ*. Συγκεκριμένα για τις κατηγορίες Κουζίνα, Εμφάνιση και Πάρκινγκ έχουμε 6, 3, 3 επίπεδα και 126, 5, 5 κόμβους φύλλου, αντίστοιχα.

Το τρίτο σύνολο δεδομένων είναι το ACM, που περιέχει 281,476 ACM δημοσιεύσεις από το datahub[2]. Το χαρακτηριστικό *Category* είναι κατηγορικό και χρησιμοποιείται από την ACM για να ταξινομήσει τις δημοσιεύσεις. Η ιεραρχία καθορίζεται από το ACM Computing Classification System[3], και οργανώνεται σε 4 επίπεδα και έχει 325 κόμβους φύλλων.

Το τέταρτο σύνολο δεδομένων είναι το Cars, που περιέχει 30,967 περιγραφές αυτοκινήτων από το διαδίκτυο[4]. Θεωρούμε τις κατηγορίες *Μηχανή*, *Αμάξωμα* ανδ *Μετάδοση ταχύτητας*, που έχουν 3, 4, 3 επίπεδα, και 11, 23, 5 κόμβους φύλλων, αντίστοιχα.

Το πέμπτο σύνολο δεδομένων είναι το RestaurantsR, από ένα πρότυπο σύστημα συστάσεων[5]. Αυτό το dataset περιλαμβάνει ένα σύνολο από 130 περιγραφές εστιατορίων και 138 χρηστών μαζί με τις προτιμήσεις τους. Για τους σκοπούς μας θεωρούμε *τέσσερις* κατηγορίες *Κουζίνα*, *Κάπνισμα*, *Εμφάνιση*, και *Περιβάλλον*, που έχουν 5, 3, 3, 3 επίπεδα, και 83, 3, 3, 3 κόμβους φύλλων, αντίστοιχα. Αυτό το dataset χρησιμοποιείται για την ανάλυση αποτελεσματικότητας του GRCO προβλήματος, ενώ τα άλλα datasets χρησιμοποιούνται για την αξιολόγηση των MCP αλγορίθμων.

Για την αξιολόγηση αποτελεσματικότητας, οι προτιμήσεις των χρηστών για τα πραγματικά datasets εξάγονται, ακολουθώντας δυο διαφορετικές προσεγγίσεις. Στην πρώτη προσέγγιση, γνωστή ως *πραγματικές προτιμήσεις* (Real preferences), επιχειρούμε να προσομοιώσουμε πραγματικές προτιμήσεις των χρηστών. Συγκεκριμένα, για το dataset RestaurantF, χρησιμοποιούμε ως προτιμήσεις των χρηστών τις περιγραφές εστιατορίων από την λίστα των καλύτερων εστιατορίων της Νέας Υόρκης [6]. Για το dataset Car, οι προτιμήσεις των χρηστών προκύπτουν από τα πιο δημοφιλή αμάξια [7]. Τέλος, για το dataset ACM, οι προτιμήσεις των χρηστών λαμβάνονται, με βάση τις δημοσιεύσεις από μια ερευνητική ομάδα[8]. Στη δεύτερη προσέγγιση, που αποκαλείται *Συνθετικές προτιμήσεις* (Synthetic preferences), οι προτιμήσεις του χρήστη λαμβάνονται χρησιμοποιώντας μια μέθοδο παρόμοια με εκείνη που ακολουθήθηκε στο Synthetic dataset. Συγκεκριμένα, οι προτιμήσεις των χρηστών προσδιορίζονται, επιλέγοντας τυχαία ιεραρχικούς κόμβους από το δεύτερο επίπεδο ιεραρχίας (δηλαδή, $\ell_u = 2$). Ο Πίνακας 2.3 συνοψίζει τα βασικά χαρακτηριστικά των χρησιμοποιούμενων πραγματικών datasets.

.

### 2.5.2 Αποδοτικότητα MPC Αλγορίθμων

Για το MPC πρόβλημα, εξετάζουμε τον IND αλγόριθμο και τρεις εκδοχές του BSL αλγορίθμου, που ονομάζονται BSL-BNL, BSL-SFS και BSL-BBS, που χρησιμοποιούν τους skyline αλγόριθμους BNL [86], SFS [108], BBS [269], αντίστοιχα.

[1] www.factual.com
[2] datahub.io/dataset/rkb-explorer-acm
[3] www.acm.org/about/class/ccs98-html
[4] www.epa.gov
[5] archive.ics.uci.edu/ml/datasets/Restaurant+&+consumer +data
[6] www.yelp.com
[7] www.edmunds.com/car-reviews/top-rated.html
[8] www.dblab.ntua.gr/pubs

**Table 2.3:** Real datasets basic characteristics

| Dataset | Number of Objects | Attributes (Hierarchy height) |
|---|---|---|
| RestaurantsF | 85,691 | *Cuisine* (6), *Attire* (3), *Parking* (3) |
| ACM | 281,476 | *Category* (4) |
| Cars | 30,967 | *Engine* (3), *Body* (4), *Transmission* (3) |
| RestaurantsR | 130 | *Cuisine* (5), *Smoke* (3), *Dress* (3), *Ambiance* (3) |

**Table 2.4:** Parameters (Synthetic)

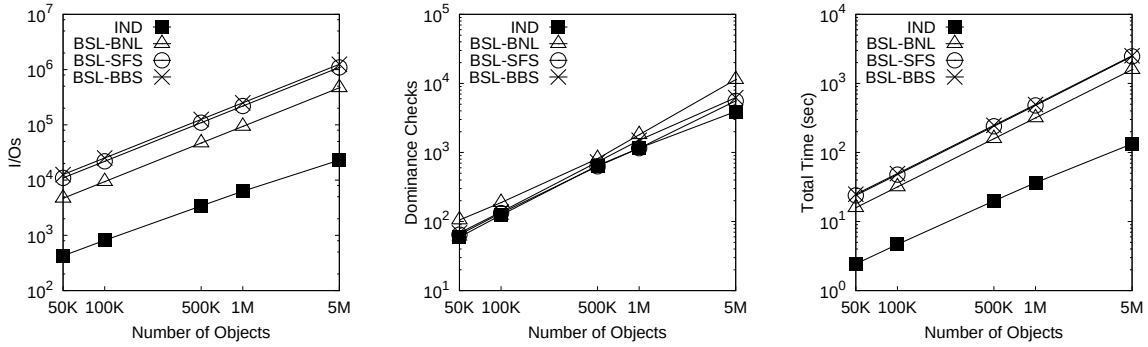
| Description | Symbol | Values |
|---|---|---|
| Number of objects | $\|\mathcal{O}\|$ | 50K, 100K, **500K**, 1M, 5M |
| Number of attribute | $d$ | 2, 3, **4**, 5, 6 |
| Group size | $\|\mathcal{U}\|$ | 2, 4, **8**, 16, 32 |
| Hierarchy height | $\log\|A\|$ | 4, 6, **8**, 10, 12 |
| Hierarchy level for objects | $\ell_o$ | **1**, 2, 3, 4, 5 |
| Hierarchy level for users | $\ell_u$ | **2**, 3, 4, 5, 6 |

Για να μετρηθεί η αποδοτικότητα των αλγορίθμων, μετράμε: (1) τον αριθμό των I/O πράξεων (2) ο αριθμός των ελέγχων κυριαρχίας και (3) τον συνολικό το χρόνο εκτέλεσης, το οποίο μετράται σε δευτερόλεπτα.

#### 2.5.2.1 Αποτελέσματα στο Συνθετικά Δεδομένα

Σε αυτήν την ενότητα μελετάμε την αποδοτικότητα των MCP αλγόριθμων κάνοντας χρήση του **Συνθετικού** dataset. Ο Πίνακας 2.4 περιγράφει τις παραμέτρους που μεταβάλλουμε και το εύρος των τιμών που εξετάζονται για το Synthetic.

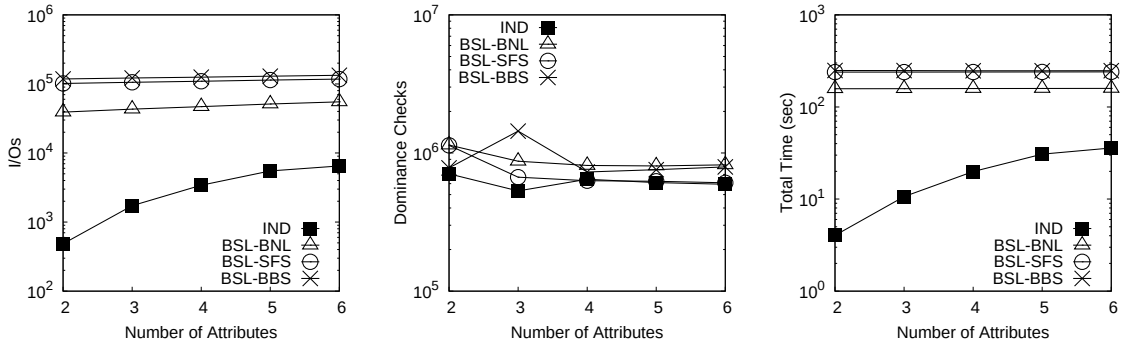
υσινγ τηε Synthetic δαταση δεσχριβεδ ιν συβσεχτιον 2.5.1.




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

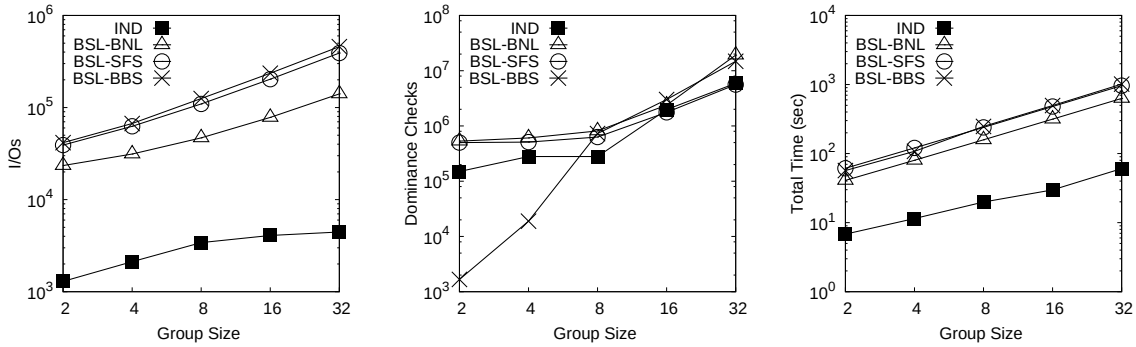
**Σχήμα 2.3:** MCP αλγόριθμοι, Synthetic: μεταβάλλοντας $|\mathcal{O}|$

**Μεταβάλλοντας τον αριθμός των αντικειμένων.** Στο πρώτο πείραμα, μελετάμε την απόδοση των αλγορίθμων σε σχέση με τον αριθμό των αντικείμενων. Πιο συγκεκριμένα, μεταβάλλουμε τον αριθμό των αντικειμένων από 50K έως 5M και μετράμε τον αριθμό των I/Os, τον αριθμό των ελέγχων κυριαρχίας, και το συνολικό χρόνο επεξεργασίας. Όταν ο αριθμός των αντικειμένων αυξάνεται, η απόδοση όλων των μεθόδων μειώνεται. Ο αριθμός των I/O (Σχήμα 2.3α′) που εκτελούνται από τον IND είναι πολύ μικρότερος από τις BSL παραλλαγές, ο λόγος είναι ότι ο BSL πρέπει να κατασκευάσει ένα αρχείο που περιέχει τους βαθμούς ταιριάσματος. Επιπλέον, οι




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.4:** MCP αλγόριθμοι, Synthetic: μεταβάλλοντας $d$




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.5:** MCP αλγόριθμοι, Synthetic: μεταβάλλοντας $|\mathcal{U}|$

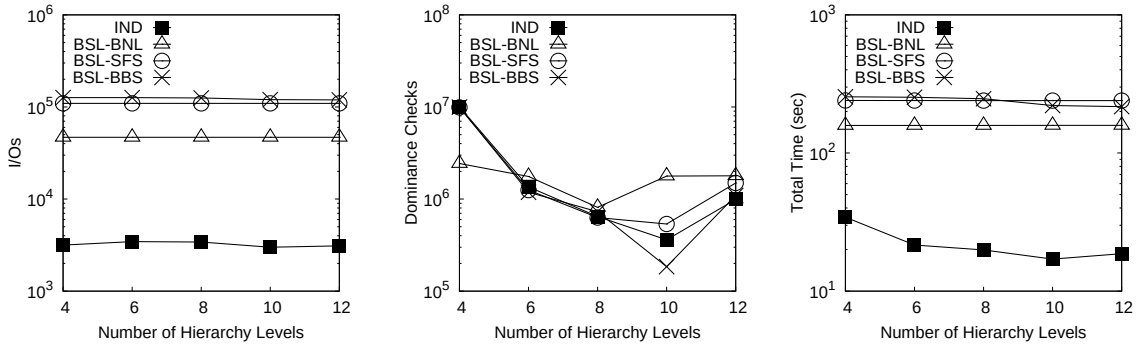
παραλλαγές SFS και BBS πρέπει να προ-επεξεργαστουν το αρχείο αυτό, και να ταξινομήσουν τα αντικείμενα ή να χτίσουν το R-δέντρο, αντίστοιχα. Ως εκ τούτου, η BSL-BNL απαιτεί τα λιγότερα I/O μεταξύ των BSL παραλλαγών.

**Μεταβάλλοντας τον αριθμός των χρηστών.** Στο επόμενο πείραμα, μεταβάλλουμε τον αριθμό των χρηστών $|\mathcal{U}|$ από 2 έως 32. Τα αποτελέσματα απεικονίζονται στο Σχήμα 2.5γ′. Η απόδοση όλων των μεθόδους μειώνεται όσο αυξάνεται το $|\mathcal{U}|$. Το Σχήμα 2.5γ′ δείχνει ότι ο IND είναι κάτι περισσότερο από μια τάξη μεγέθους γρηγορότερος από όλες τις BSL παραλλαγές, μεταξύ των οποίων η BSL-BNL είναι η πιο γρήγορη.

**Μεταβάλλοντας το ύψος της ιεραρχίας.** Σε αυτό το πείραμα, μεταβάλλουμε το ύψους της ιεραρχία $\log|A|$ από 4 έως 12 επίπεδα. Το Σχήμα 2.6γ′ απεικονίζει τα αποτελέσματα. Όλες οι μέθοδοι δεν επηρεάζονται σημαντικά από αυτή την παράμετρο. Συνολικά, η IND είναι κάτι περισσότερο από μια τάξη μεγέθους ταχύτερη από όλες τις BSL παραλλαγές.

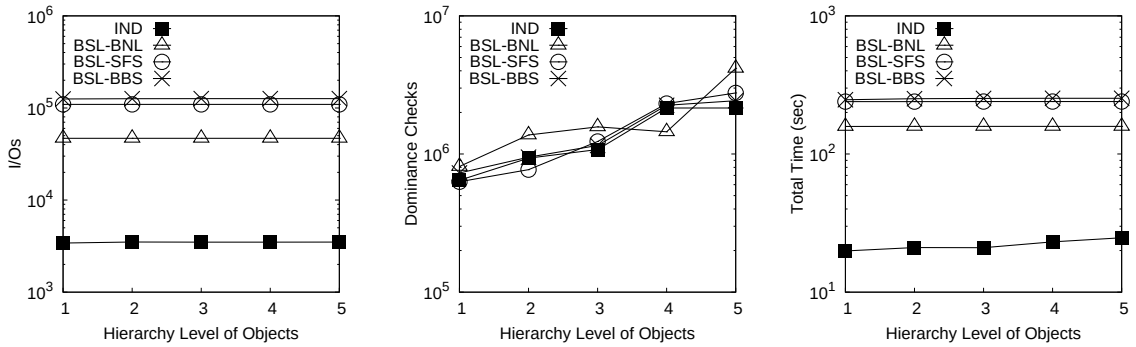
**Μεταβάλλοντας το επίπεδο των αντικειμένων.** Το Σχήμα 2.7γ′ απεικονίζει τα αποτελέσματα μεταβάλλοντας το επίπεδο $\ell_o$. Η απόδοση όλων των μεθόδων δεν επηρεάζεται σημαντικά από το $\ell_o$.

**Μεταβάλλοντας το επίπεδο των χρηστών.** Σε αυτό το πείραμα, μεταβάλουμε το επίπεδο $\ell_u$. Ο συνολικός χρόνος για την μέθοδο IND παίρνει την υψηλότερη τιμή




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.6:** MCP αλγόριθμοι, Synthetic: μεταβάλλοντας $\log|A|$



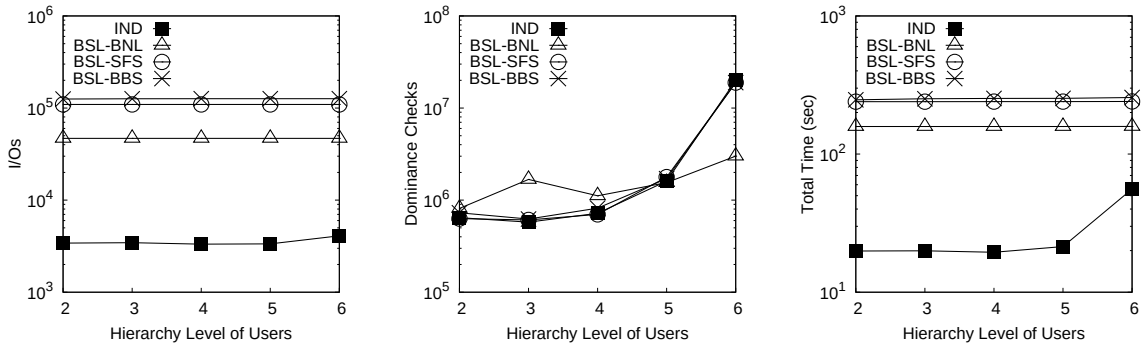

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.7:** MCP αλγόριθμοι, Synthetic: μεταβάλλοντας $\ell_o$

του για $\ell_u = 6$, καθώς ο αριθμός των ελέγχων κυριαρχία αυξάνει απότομα για αυτή τη τιμή.

### 2.5.2.2 Αποτελέσματα σε Πραγματικά Δεδομένα

Σε αυτή την ενότητα μελετάμε την αποδοτικότητα των MCP αλγορίθμων κάνοντας χρήση τριών πραγματικών datasets όπως περιγράφονται στην Ενότητα 2.5.1. Για κάθε dataset, εξετάζουμε τόσο πραγματικές, όσο και συνθετικές προτιμήσεις, που λαμβάνονται, όπως περιγράψαμε στην Ενότητα 2.5.1. Επίσης, επιλέγουμε το μέγεθος της ομάδας $|\mathcal{U}|$ από 2 μέχρι 32 χρήστες.




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.8:** MCP αλγόριθμοι, Synthetic: μεταβάλλοντας $\ell_u$

Τα διαγράμματα στα Σχήμα 2.9 & 2.10 παρουσιάζουν τα αποτελέσματα για το dataset ΡεσταυραντσΦ, για πραγματικές και συνθετικές προτιμήσεις, αντίστοιχα. Ομοίως, τα διαγράμματα στα Σχήμα 2.11 & 2.12 παρουσιάζουν τα αποτελέσματα για το dataset ACM, και τα διαγράμματα Σχήμα 2.13 & 2.14 για το Cars. Όπως παρατηρούμε, η απόδοση των εξεταζόμενων μεθόδων είναι παρόμοια για όλα τα datasets, πραγματικά και συνθετικά. Επίσης παρόμοια απόδοση παρατηρείται σε πραγματικές και συνθετικές προτιμήσεις χρηστών. Στις περισσότερες περιπτώσεις, ο IND έχει καλύτερη επίδοση κατά τουλάχιστον μια τάξη μεγέθους σε ότι αφορά τα I/Os και το συνολικό χρόνο. Επιπλέον, ο IND πραγματοποιεί λιγότερους κύριους ελέγχους σε σχέση με τη μέθοδο BSL σχεδόν σε όλες τις περιπτώσεις.

Όσον αφορά τις μεθόδους BSL, ο BSL-BNL υπερνικά τους υπόλοιπους σε ότι αφορά τα I/Os και το συνολικό χρόνο, ενώ οι BSL- SFS και BNL-BBS έχουν σχεδόν τις ίδιες επιδόσεις. Σε ότι αφορά τον αριθμό των ελέγχων, για λιγότερους από 16 χρήστες ο BSL-BNL πραγματοποιεί περισσότερους ελέγχους από άλλες μεθόδους BSL, ενώ για 32 χρήστες, σε πολλές περιπτώσεις (Σχήμα 2.9β′, 2.10β′, 2.11β′), ο BSL πραγματοποιεί τους λιγότερους κύριους ελέγχους από τις μεθόδους BSL. Τέλος, για λιγότερους από 8 χρήστες, ο BNL-BBS πραγματοποιεί λιγότερους κύριους ελέγχους από τις υπόλοιπες BSL μεθόδους.

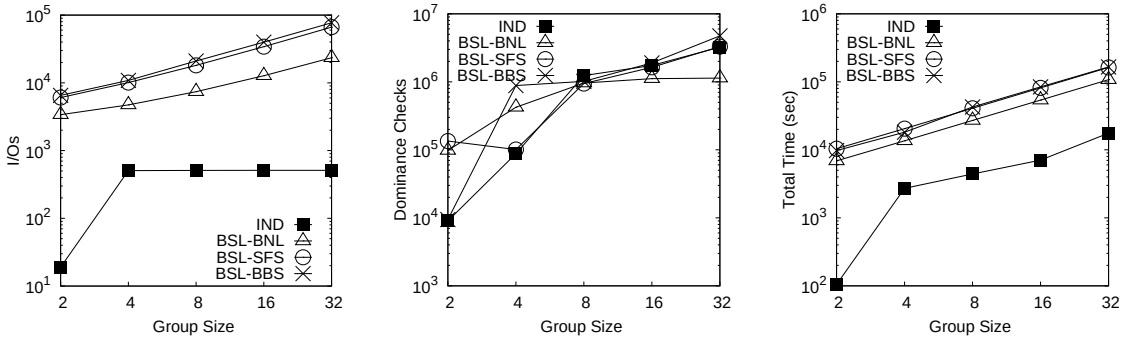
### 2.5.3 Αποδοτικότητα $p$-MCP Αλγορίθμων

Σε αυτή την ενότητα, ερευνούμε τις επιδόσεις των $p$-MCP αλγορίθμων (Ενότητα 2.3). Για το $p$-MCP πρόβλημα, εφαρμόζουμε τις αντίστοιχες επεκτάσεις όλων των αλγορίθμων IND και BSL, προσθέτοντας ένα πρόθεμα $p$. Όπως προηγουμένως, μετράμε τον αριθμό των I/O λειτουργιών, κύριων ελέγχων και του συνολικού χρόνου. Στα ακόλουθα πειράματα, χρησιμοποιούμε τα τρία πραγματικά datasets και μεταβάλουμε τον αριθμό των χρηστών από 2 έως 1024, καθώς $p$ = 30%. Επίσης μεταβάλουμε την παράμετρο $p$ από 10% μέχρι 50%. Ωστόσο, η επίδοση όλων των μεθόδων, (σε σχέση με τα I/Os και το συνολικό χρόνο) παραμένει ανεπηρέαστη από το $p$. Έτσι τα σχετικά διαγράμματα παραλείπονται.

Τα διαγράμματα Σχήμα 2.15 & 2.16 παρουσιάζουν το αποτέλεσμα για το RestaurantsF dataset, για πραγματικές και συνθετικές προτιμήσεις αντίστοιχα. Παρόμοια, τα διαγράμματα Σχήμα 2.17 & 2.18 αντιστοιχούν στο dataset ACM, και τα διαγράμματα Σχήμα 2.19 & 2.20 στο Cars.

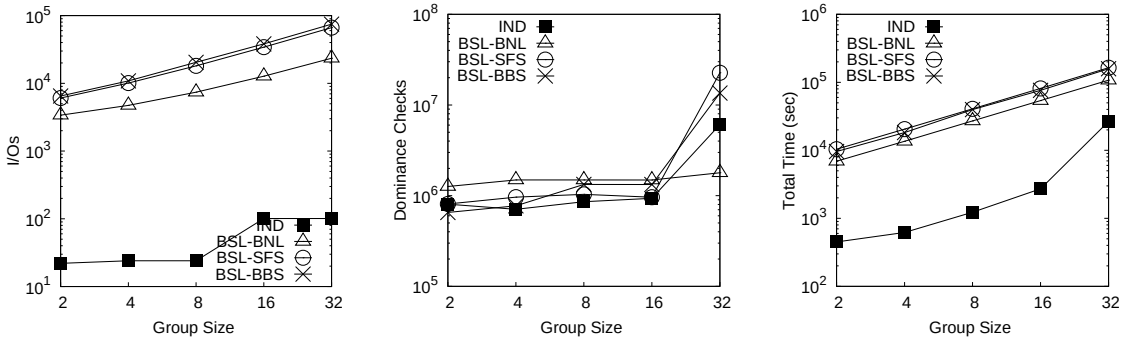
Όπως παρατηρούμε, ο IND υπερνικά τη μέθοδο BSL, σε όλες σχεδόν τις περιπτώσεις. Συγκεκριμένα, ο αριθμός των I/O λειτουργιών στον IND, είναι αρκετές τάξεις μεγέθους λιγότερες από τις παραλλαγές του BSL. Επιπλέον, σε σχεδόν όλες τις περιπτώσεις, ο IND πραγματοποιεί λιγότερους ελέγχους (dominance checks) από τις μεθόδους BSL. Ο αριθμός των I/Os που πραγματοποιούνται από τον IND παραμένει σταθερός για πάνω από 16 χρήστες, ενώ για τις BSL μεθόδους, οι λειτουργίες I/O συνεχώς αυξάνονται μέχρι τους 256 χρήστες. Όσον αφορά τους ελέγχους ο αριθμός τους αυξάνεται με $|\mathcal{U}|$ ακολουθώντας μια σχεδόν παρόμοια τάση για όλες τις μεθόδους.

Τέλος, όσον αφορά τις μεθόδους BSL, ο BSL-BNL υπερνικά τις άλλες BSL μεθόδους στα I/Os και το συνολικό χρόνο. ενώ οι BSL-SFS , BNL-BBS έχουν σχεδόν τις ίδιες επιδόσεις. Ως προς τους ελέγχους, σε μερικές περιπτώσεις (Σχήμα 2.15β′ & 2.17β′) ο BSL-BNL υπερνικά όλες τις BSL μεθόδους, ενώ σε άλλες περιπτώσεις (Σχήμα 2.16β′, 2.18β′, 2.19β′, 2.20β′),) ο BSL-BNL πραγματοποιεί περισσότερους ελέγχους από τις άλλες μεθόδους BSL.



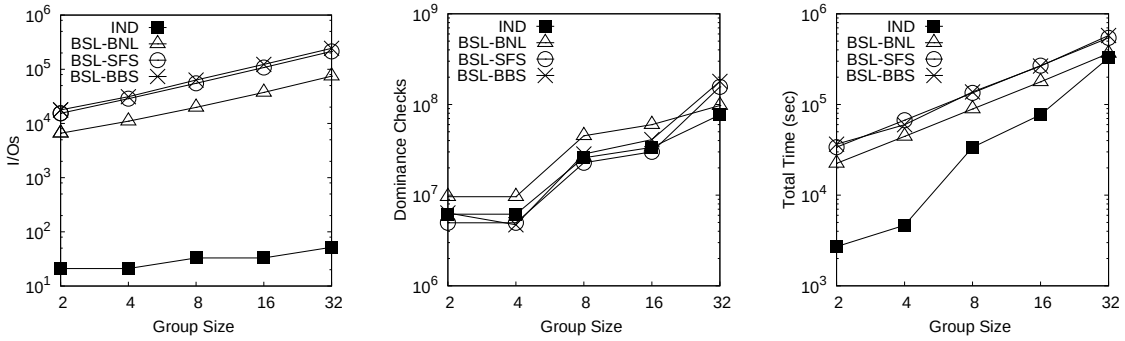

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.9:** MCP αλγόριθμοι, RestaurantsF (Real preferences): μεταβάλλοντας $|\mathcal{U}|$



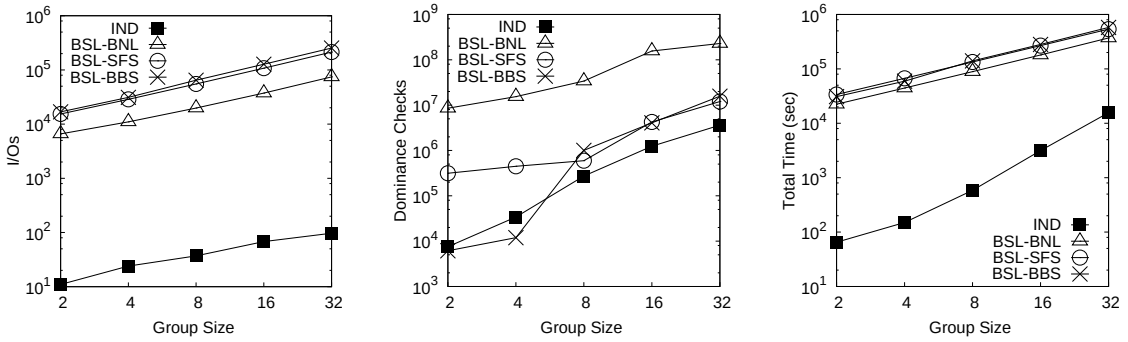

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.10:** MCP αλγόριθμοι, RestaurantsF (Synthetic preferences): μεταβάλλοντας $|\mathcal{U}|$



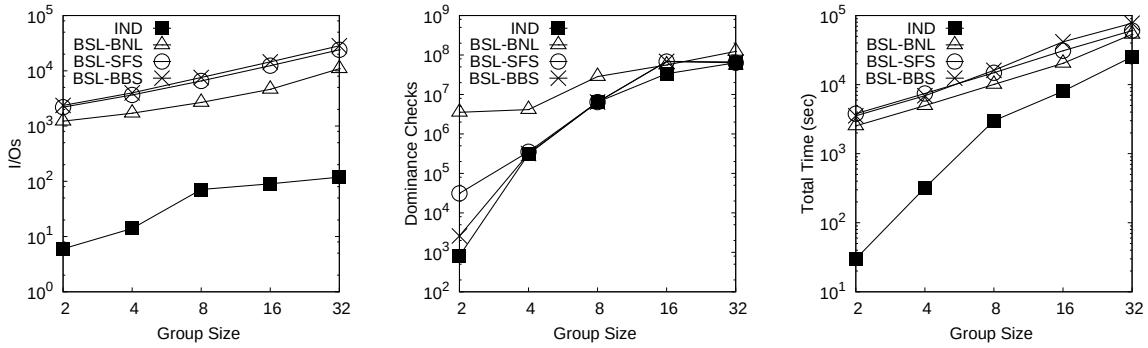

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.11:** MCP αλγόριθμοι, ACM (Real preferences): μεταβάλλοντας $|\mathcal{U}|$



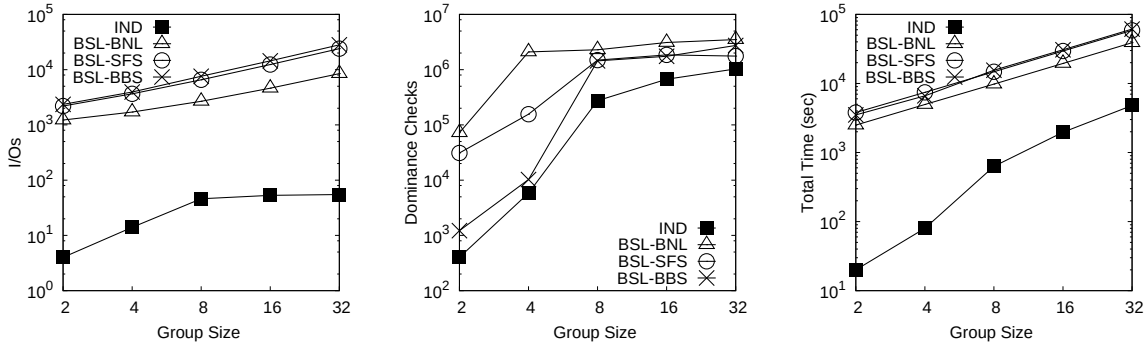

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.12:** MCP αλγόριθμοι, ACM (Synthetic preferences): μεταβάλλοντας $|\mathcal{U}|$



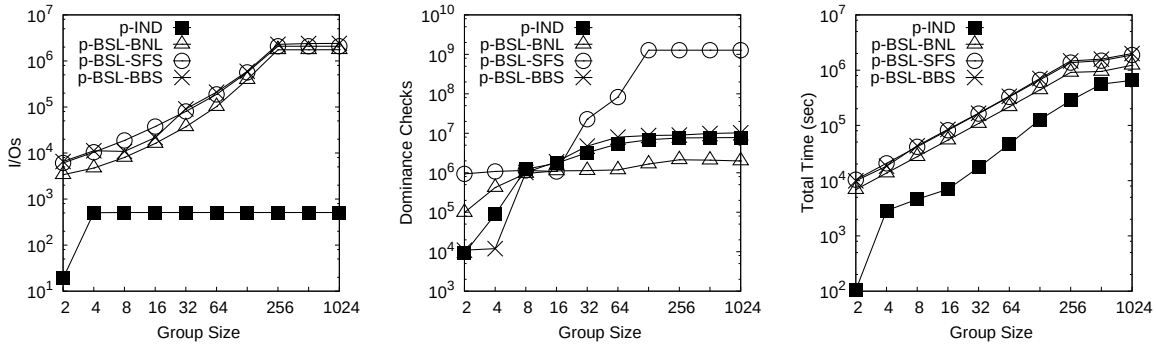

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.13:** MCP αλγόριθμοι, Cars (Real preferences): μεταβάλλοντας $|\mathcal{U}|$



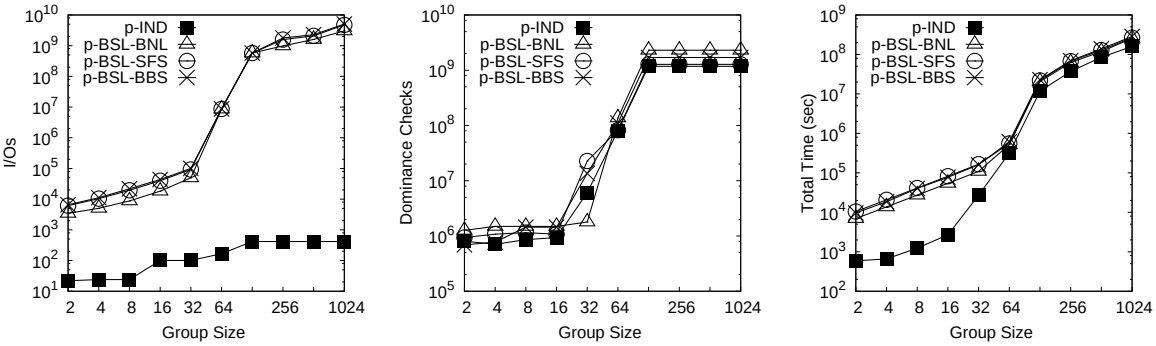

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.14:** MCP αλγόριθμοι, Cars (Synthetic preferences): μεταβάλλοντας $|\mathcal{U}|$



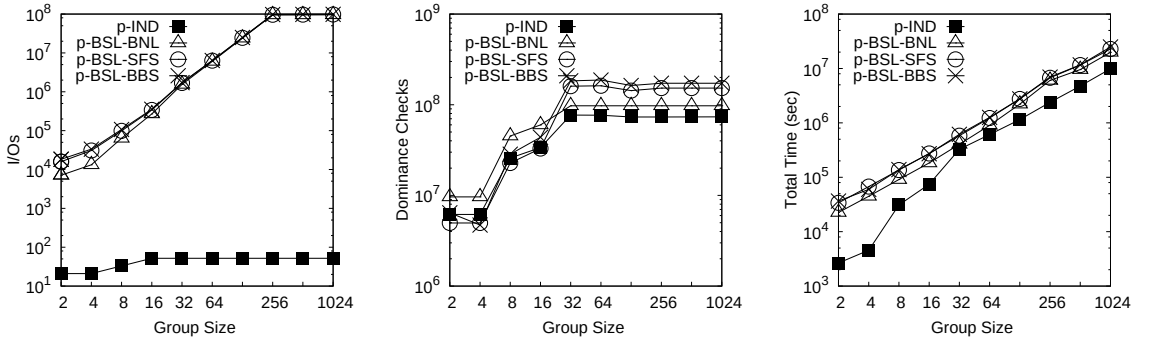

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.15:** $p$-MCP αλγόριθμοι, RestaurantsF (Real preferences): μεταβάλλοντας $|\mathcal{U}|$



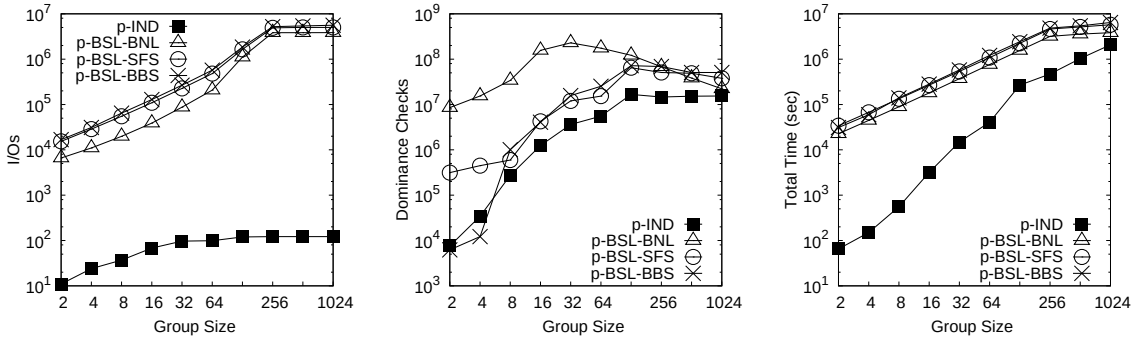

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.16:** $p$-MCP αλγόριθμοι, RestaurantsF (Synthetic preferences): μεταβάλλοντας $|\mathcal{U}|$



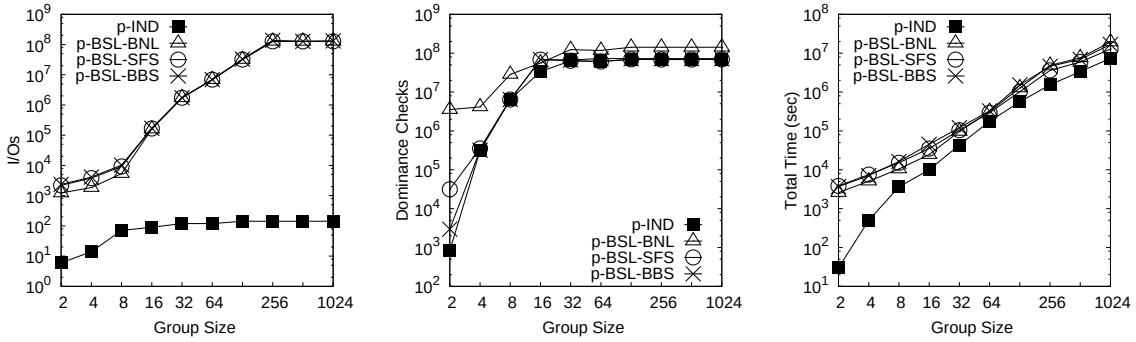

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.17:** $p$-MCP αλγόριθμοι, ACM (Real preferences): μεταβάλλοντας $|\mathcal{U}|$



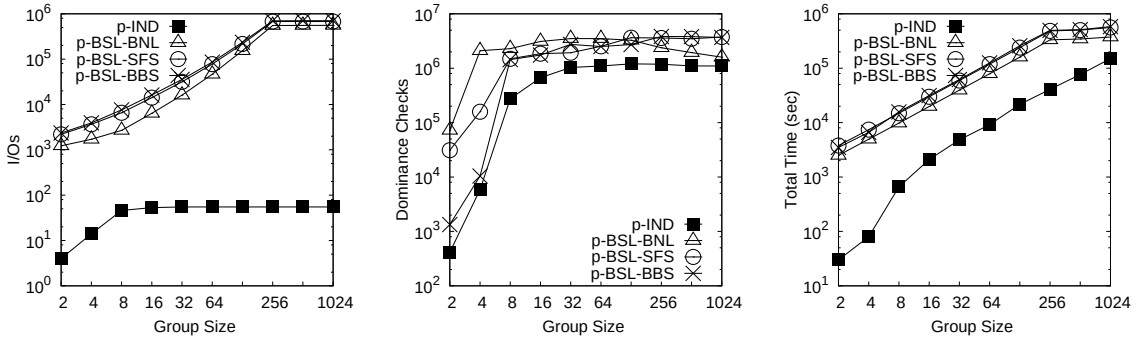

(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.18:** $p$-MCP αλγόριθμοι, ACM (Synthetic preferences): μεταβάλλοντας $|\mathcal{U}|$




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.19:** $p$-MCP αλγόριθμοι, Cars (Real preferences): μεταβάλλοντας $|\mathcal{U}|$




(α′) I/O Πράξεις (β′) Έλεγχοι (γ′) Συνολικός Χρόνος

**Σχήμα 2.20:** $p$-MCP αλγόριθμοι, Cars (Synthetic preferences): μεταβάλλοντας $|\mathcal{U}|$

### 2.5.4 Αποτελεσματικότητα του GRCO

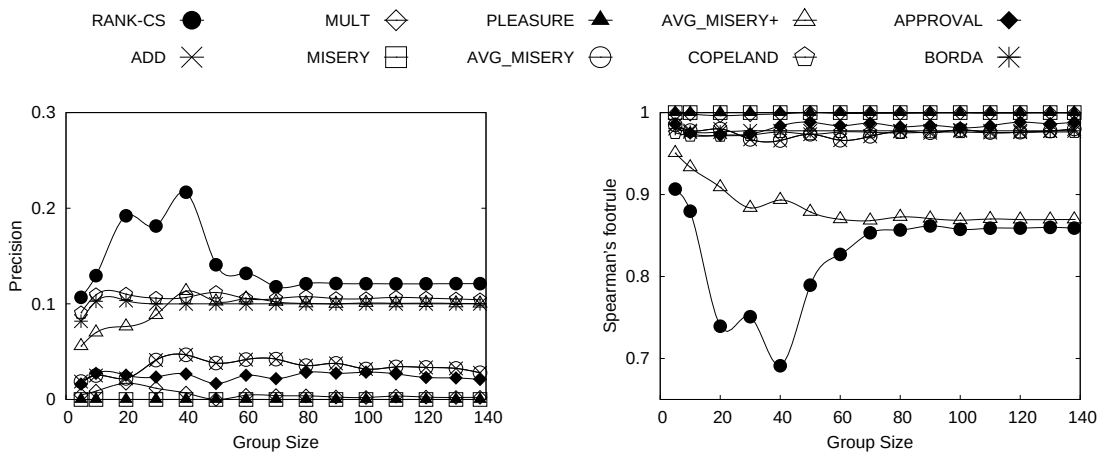
Σε αυτή την ενότητα μελετάμε την αποτελεσματικότητα του προβλήματος GRCO (σεστιον 2.4). Συγκρίνουμε τον RANK-CM αλγόριθμο με εννιά διάσημες στρατηγικές συνάθροισης (aggregations strategies), που χρησιμοποιούνται από τα περισσότερα συστήματα ομαδικών προτάσεων (group recommender systems) [93]. Συγκεκριμένα, εφαρμόζουμε τις ακόλουθες στρατηγικές:

- *Additive (ADD)*: προσθέτει τους ατομικούς βαθμούς συσχέτισης individual matching degrees
- *Multiplicative* (MULT): πολλαπλασιάζει τους ατομικούς βαθμούς συσχέτισης
- *Least Misery* (MISERY): εξάγει τον μικρότερο από τους ατομικούς βαθμούς συσχέτισης
- *Most Pleasure* (PLEASURE): εξάγει τον μέγιστο από τους ατομικούς βαθμούς συσχέτισης
- *Average Without Misery* (AVG_MISERY): παίρνει τους μέσους ατομικούς βαθμούς συσχέτισης, αποκλείει τους ατομικούς βαθμούς συσχέτισης κάτω από ένα κατώφλι
- *Average Without Misery Threshold-free* (AVG_MISERY+): είναι μια στρατηγική που εισάγεται εδώ, παρόμοια με AVG_MISERY, με τη διαφορά ότι το κατώφλι τίθεται στο ελάχιστο των ατομικών βαθμών συσχέτισης
- *Copeland Rule* (COPELAND): υπολογίζει για ένα αντικείμενο, τον αριθμό, από πόσα αντικείμενα έχει μεγαλύτερο βαθμό συσχέτισης, μείον από πόσα έχει μικρότερο.
- *Approval Voting* (APPROVAL): μετράει τον αριθμό των ατομικών βαθμών συσχέτισης, με τιμή μεγαλύτερη ή ίση από ένα κατώφλι.
- *Borda Count* (BORDA): προσθέτει το "σκορ" που υπολογίζεται για κάθε βαθμό συσχέτισης σύμφωνα με την κατάταξη του σε μια λίστα προτιμήσεων χρηστών (ο βαθμός με τη μικρότερη αξία παίρνει μηδέν πόντους, ο επόμενος ένα κ.ο.κ.)

Σημειώστε πως, το κατώφλι στις στρατηγικές AVG_MISERY και APPROVAL τίθεται στο 0.5.

Για να αξιολογήσουμε την αποτελεσματικότητα του σχήματος ταξινόμησης μας, χρησιμοποιούμε το dataset RestaurantsR. Χρησιμοποιούμε τις αξιολογήσεις όλων των χρηστών και εξάγουμε μια λίστα κατάταξης των διασημότερων εστιατορίων, που παίζει το ρόλο της "απόλυτης αλήθειας" (ground truth). Έπειτα, συγκρίνουμε τις λίστες που μας δίνουν η RANK-CM και οι άλλες στρατηγικές συνάθροισης aggregation strategies, με την ground truth. Υπολογίζουμε την *Ακρίβεια* (Precision) και το *Generalized Spearman's Footrule* [150], για διάφορα ranks και μεγέθη ομάδων. Για να κατασκευάσουμε τις ομάδες χρηστών, για κάθε μέγεθος ομάδας, επιλέγουμε τυχαία χρήστες, συνθέτοντας 500 ομάδες του ίδιου μεγέθους. Έτσι, στα πειράματα εμφανίζονται οι μέσες τιμές.

**Μεταβάλλοντας το Μέγεθος της Ομάδας.** Στο πρώτο πείραμα (Σχήμα 2.21 & 2.22), θεωρούμε διαφορετικά μεγέθη ομάδων, μεταβάλλοντας τον αριθμό των χρηστών, από 5 μέχρι 138. Υπολογίζουμε την ακρίβεια και το Spearman's footrule για




(α′) Precision (β′) Spearman's footrule

**Σχήμα 2.21:** RestaurantsR (Rank 10): μεταβάλλοντας $|\mathcal{U}|$

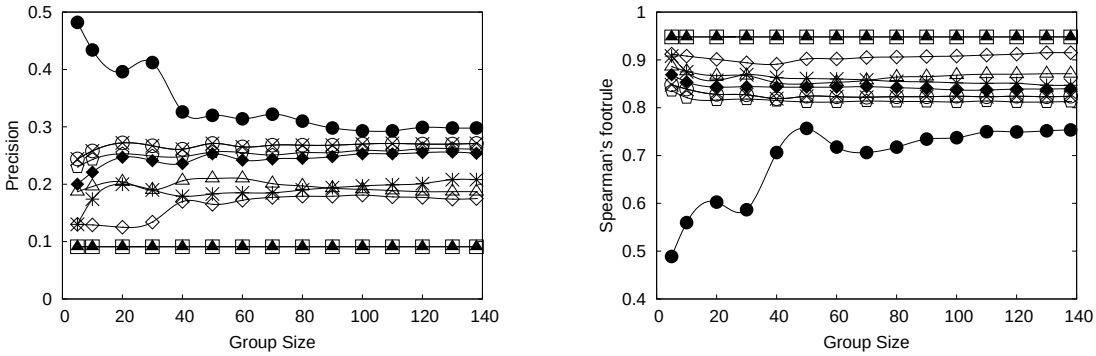
όλες τις μεθόδους συγκρινόμενη με την απόλυτη αλήθεια, στη θέση 10 (Σχήμα 2.21) και θέση 20 (Σχήμα 2.22).

Στο Σχήμα 2.21, θεωρούμε τα πρώτα δέκα εστιατόρια (δηλ., τη θέση 10), η ακρίβεια για κάθε μέθοδο, ορίζεται ως ο αριθμός των κοινών εστιατορίων ανάμεσα στην απόλυτη αλήθεια, και τη λίστα που μας επιστρέφει η εκάστοτε μέθοδος, δια του δέκα. Για παράδειγμα, στο Σχήμα 2.21α′, για τις ομάδες των 20 χρηστών, η RANK-CM έχει ακρίβεια περίπου 0.2, που σημαίνει ότι για τα πρώτα δέκα εστιατόρια, ο RANK-CM ανακτά κατά μέσο όρο δύο διάσημα εστιατόρια. Απ′ την άλλη, οι BORDA και COPELAND ανακτούν κατά μέσο όρο ένα διάσημο εστιατόριο, και έχουν ακρίβεια περίπου 0.1.

Όσον αφορά τα αποτελέσματα στη θέση 10, όπως παρατηρούμε από το Σχήμα 2.21, η RANK-CM ξεπερνάει σε επίδοση όλες τις άλλες μεθόδους και στις δύο μετρήσεις. Σημειώστε ότι, οι τιμές του Spearman's footrule κινούνται από 0 εώς 1, όπου οι χαμηλότερες αξίες υποδεικνύουν μια καλύτερη προσέγγιση της απόλυτης αλήθειας (το 0 σημαίνει ότι οι δύο λίστες είναι ίδιες). Όσον αφορά τις υπόλοιπες στρατηγικές συνάθροισης, τα καλύτερα αποτελέσματα δίνονται από τις COPELAND, BORDA AVG_MISERY+, ενώ οι MISERY και PLEASURE έχουν τα χειρότερα.

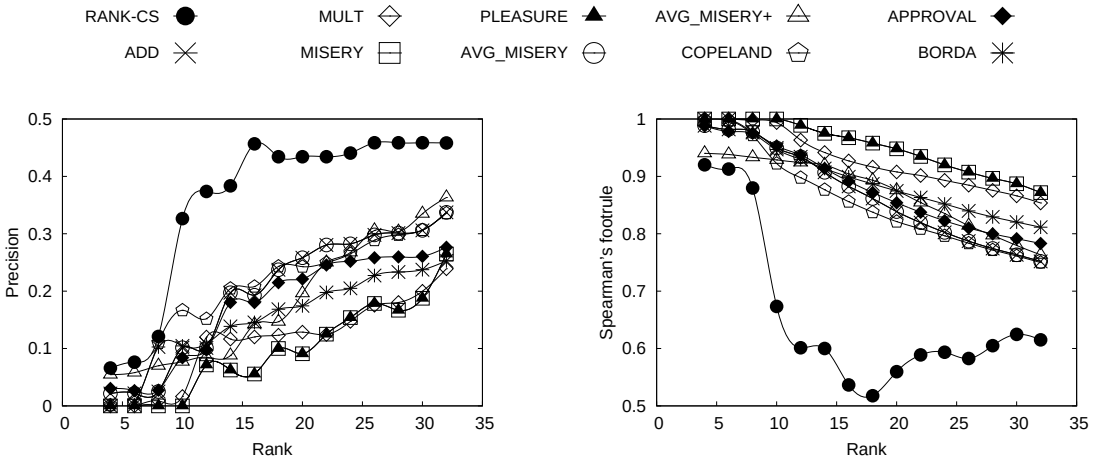
Παρόμοια αποτελέσματα και παρατηρήσεις ισχύουν στη θέση 20 (Σχήμα 2.22), όπου η RANK-CM ξεπερνάει όλες τις υπόλοιπες μεθόδους, με τις COPELAND, ADD καιAVG_MISERY να είναι οι καλύτερες εναλλακτικές. Συνολικά,η RANK-CM αποδίδει καλύτερα από άποψη ακρίβειας και pearman's footrule , από τις υπόλοιπες στρατηγικές, σε όλες τις περιπτώσεις. Η στρατηγική COPELAND φαίνεται να είναι η καλύτερη εναλλακτική, ενώ οι MISERY και PLEASURE οι χειρότερες.

**Μεταβάλλοντας τη Θέση** Σε αυτό το πείραμα, θεωρούμε διαφορετικά μεγέθη ομάδας (δηλ. 10, 20, 30) και υπολογίζουμε την ακρίβεια και το Spearman's footrule από τις θέσεις 4 μέχρι 32. Όπως μπορούμε να παρατηρήσουμε από τα διαγράμματα Σχήμα 2.23, 2.24 & 2.25, η επίδοση όλων των μεθόδων είναι σχεδόν παρόμοια για το εξεταζόμενο μέγεθος ομάδας. Η RANK-CM πετυχαίνει καλύτερη επίδοση σε ότι αφορά την ακρίβεια και το Spearman's footrule σε σχεδόν όλες τις εξεταζόμενες θέσεις. Με εξαιρέσεις στις θέσεις 4 και 6 για τα μεγέθη ομάδας 10 και 30, όπου η COPELAND πετυχαίνει σχεδόν την ίδια επίδοση. Όσον αφορά τις άλλες μεθόδους, η καλύτερη επίδοση είναι από τις COPELAND, ADD και AVG_MISERY+.



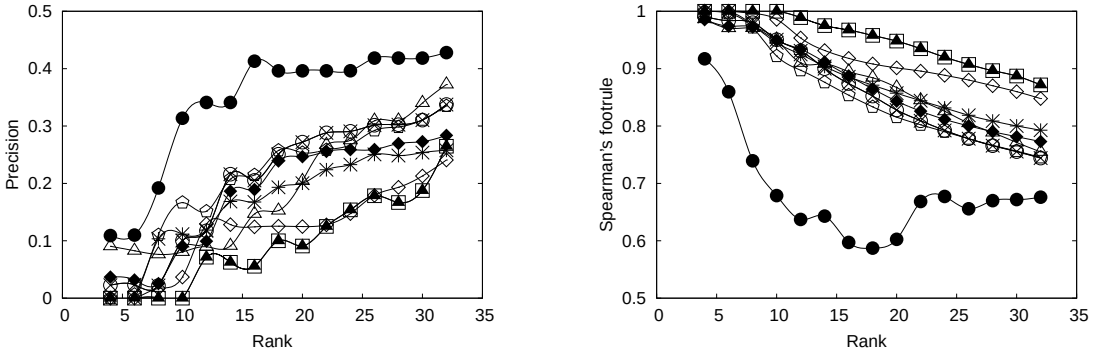

(α′) Precision (β′) Spearman's footrule

**Σχήμα 2.22:** RestaurantsR (Rank 20): μεταβάλλοντας $|\mathcal{U}|$



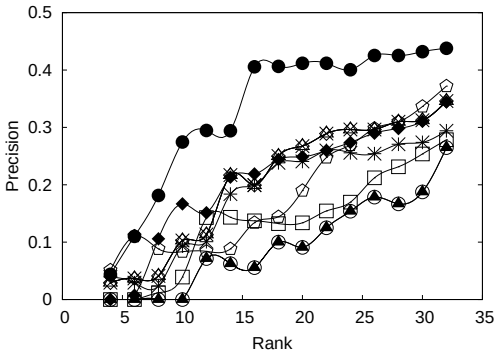

(α′) Precision (β′) Spearman's footrule

**Σχήμα 2.23:** RestaurantsR ($|\mathcal{U}| = 10$): μεταβάλλοντας rank

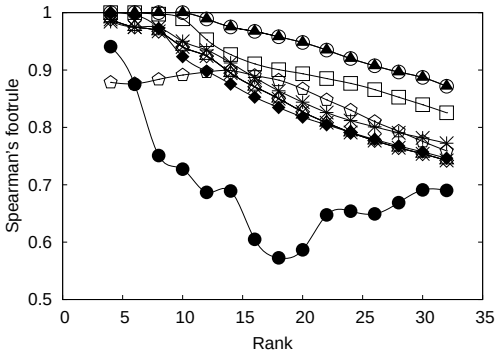



(α′) Precision (β′) Spearman's footrule

**Σχήμα 2.24:** RestaurantsR ($|\mathcal{U}| = 20$): μεταβάλλοντας rank




(α′) Precision




(β′) Spearman's footrule

**Σχήμα 2.25:** RestaurantsR ($|\mathcal{U}| = 30$): μεταβάλλοντας rank

## 2.6 Σχετικές Εργασίες

### 2.6.1 Συστήματα Συστάσεων

Υπάρχουν διάφορες τεχνικές για την μοντελοποίηση *προτιμήσεων* [310, 225]. Υπάρχουν οι ποσοτικές προτιμήσεις (quantitative preferences), π.χ., [23, 187, 219], κατά τις οποίες οι προτιμήσεις ορίζονται με αριθμητικά σκορ στα χαρακτηριστικά. Επίσης υπάρχουν οι ποιοτικές προτιμήσεις (qualitative preferences), π.χ., [211, 107], κατά τις οποίες οι προτιμήσεις ορίζονται χρησιμοποιώντας δυαδικές σχέσεις.

Ο γενικός στόχος των συστημάτων σύστασης (recommendation systems) [19, 80, 351, 204] είναι να εντοπιστούν εκείνα τα αντικείμενα που ταιριάζουν περισσότερο με τις προτιμήσεις του χρήστη. Συνήθως, τα συστήματα αυτά παρέχουν μια κατάταξη των αντικειμένων συγκεντρώνοντας τις προτιμήσεις των χρηστών π.χ., [23, 99, 187, 211, 107].

Πρόσφατα, διάφορες μέθοδοι για συστάσεις σε ομάδες χρηστών (group recommendations) [193, 253, 93, 85], έχουν προταθεί. Αυτές οι μέθοδοι, προτείνουν τα στοιχεία, προσπαθώντας να ικανοποιήσει όλα τα μέλη της ομάδας [291, 266] και ταξινομούνται σε δύο προσεγγίσεις. Στην πρώτη, οι προτιμήσεις του κάθε μέλους της ομάδας συνδυάζονται για να δημιουργήσουν ένα εικονικό χρήστη, οι συστάσεις της ομάδας πρότεινε με βάση τον εικονικό χρήστη. Στο δεύτερο, ξεχωριστές συστάσεις για κάθε μέλος υπολογίζεται και στη συνέχεια συγχωνεύονται σε μια ενιαία σύσταση. Αρκετές μέθοδοι να συνδυαστούν διαφορετικές λίστες κατάταξης έχουν προταθει στην περιοχή της ανάκτηση πληροφορίας (information retrieval) [36, 139, 261, 152, 156].

Ένας μεγάλος αριθμός μεθόδων συστάσεων για ομάδες χρηστών έχουν αναπτυχθεί σε αρκετούς τομείς όπως: μουσική [305, 115, 277, 255, 100, 355], ταινίες [267], τηλεοπτικά προγράμματα [252, 352, 338, 42], εστιατόρια [270, 254], περιηγήσεις σε αξιοθέατα [160, 33, 209], πακέτα διακοπών [257, 192], τρόφιμα [141], ειδησιογραφικά νέα [278], και διαδικτυακές κοινότητες [163, 213, 39]. Τέλος, αρκετές εργασίες έχουν μελετήσει το πρόβλημα της συνάθροισης ταξινόμησης (rank aggregation) στο πλαίσιο των συστάσεων για ομάδες χρηστών [291, 41, 55, 256, 266].

### 2.6.2 Συναθροίσεις Βασισμένες σε Pareto

Η εργασία [86] αναζωπυρώσει το ενδιαφέρον για το πρόβλημα της εύρεσης των μέγιστων αντικειμένων *maximal objects* [224, 53, 52], επανεισάγοντας το ως τελεστή κορυφογραμμής skyline operator. Σε αυτή την περίπτωση, τα μέγιστα αντικείμενα αναφέρονται ως αντικείμενα κορυφογραμμής. Τα ερωτήματα κορυφογραμμής αναζητούν εκείνες τις πλειάδες μιας σχέσης για τις οποίες δεν υπάρχει κάποια άλλη πλειάδα που είναι καλύτερη από αυτές σε όλες τις διαστάσεις- γνωρίσματα, ή όπως συνηθίζεται να λέγεται δεν κυριαρχεί (dominates) πάνω στις πρώτες.

Η πιο γνωστή μέθοδος είναι η Block Nested Loops (BNL) [86], η οποία ελέγχει κάθε σημείο με όλα τα σημεία του συνόλου δεδομένων. Επίσης υπάρχουν μέθοδοι οι οποίες βασίζονται στην ταξινόμηση των αντικειμένων π.χ., SFS [108], LESS [168], SaLSa [43] και SOAD [300]. Οι μέθοδοι αυτοί προσπαθούν να μειώσουν τον αριθμό των συγκρίσεων κυριαρχίας (dominance checks) ταξινομώντας τα δεδομένα εισόδου. Τελος, ο RAND [295] αλγόριθμος βασίζεται σε πολλαπλά περάσματα του αρχείου εισόδου και δειγματοληψία (βλ. Ενότητα 3.4 για περισσότερες λεπτομέρειες).

Σε άλλες προσεγγίσεις, πολυδιάστατες τεχνικές δεικτοδοτήσεις χρησιμοποιούνται για την καθοδήγηση της αναζήτησης. Ο πιο γνωστός αλγόριθμος σε αυτή την κατηγορία είναι ο BBS [269] όπου χρησιμοποιεί ένα R-δέντρο.

Ομοίως, ο *Nearest Neighbor* αλγόριθμος (NN) [218] χρησιμοποιεί επίσης ένα R-δέντρο για την εκτέλεση πολλαπλών nearest neighbor αναζητήσεων. Μια δομή bitmap χρησιμοποιείται από τον *Bitmap* [316] αλγόριθμο, ενώ στον *Index* [316] χρησιμοποιείται ένα B-δέντρο ανά διάσταση. Αλλες μεθόδους, π.χ. , [229, 237], χρησιμοποιούν space-filling curves. Τέλος, ο *Lattice Skyline* (LS) αλγοριθμος [262] χτίζει μια εξειδικευμένη δομή δεδομένων για low-cardinality domains.

Ειδικοί αλγόριθμοι για τον αποδοτικό υπολογισμό των αντικειμένων κορυφογραμμής πάνω από μερικώς διατεταγμένα πεδία τιμών [96, 345, 293, 354], μετρικούς χώρους [104], ή σε μη-μετρικών χώρους [268], έχουν προταθεί.

Επιπλέον, έρευνας έχει πραγματοποιηθεί στην προσπάθεια να αντιμετωπιστεί το ζήτημα να περιοριστεί το μέγεθος των σκψλινε αντικειμένων [349, 97, 243, 234, 317, 228]. Αυτές οι προσεγγίσεις βασίζονται στην εισαγωγή νέων εννοιών και/ή στην ταξινόμηση των αποτελεσμάτων (βλ. [240] για περισσότερες λεπτομέρειες) .

## 2.7 Επίλογος

Αυτό το κεφάλαιο περιγράφει αντικειμενικές τεχνικές ταξινόμησης λαμβάνοντας υπόψιν τις προτιμήσεις από ένα σύνολο χρηστών. Ο στόχος είναι να ταξινομηθούν τα αντικείμενα με βάση το τι θεωρείται ιδανικό από όλους τους χρήστες. Συγκεκριμένα, εξετάζουμε τρία σχετικά προβλήματα βασιζόμενα σε μια διπλή Pareto συνάθροιση (aggregation). Το πρώτο πρόβλημα είναι να βρεθούν τα αντικείμενα που θεωρούνται ομόφωνα ιδανικά από όλους τους χρήστες. Στο δεύτερο πρόβλημα, "χαλαρώνουμε" την απαίτηση για ομοφωνία και απαιτούμε μόνο ένα ποσοστό των χρηστών να συμφωνεί. Έπειτα, στο τρίτο πρόβλημα, δημιουργούμε ένα αποτελεσματικό σχήμα ταξινόμησης βασιζόμενο στην διπλή Pareto συνάθροιση. Οι προτεινόμενες μέθοδοι, χρησιμοποιούν έναν μετασχηματισμό στις τιμές των κατηγορικών χαρακτηριστικών, ώστε να είναι δυνατή η χρήση συνηθισμένων δομών δεικτοδότησης. Τέλος, μια λεπτομερής, πειραματική ανάλυση πιστοποιεί την αποδοτικότητα και αποτελεσματικότητα των τεχνικών μας.

# Κεφάλαιο 3

# Αλγόριθμοι Κορυφογραμμής Δευτερεύουσας Μνήμης

Μια ιδιαίτερα χρήσιμη και διαδεδομένη κατηγορία ερωτημάτων που τίθενται σε πολυδιάστατα δεδομένα είναι τα λεγόμενα *ερωτήματα κορυφογραμμής* (*skyline queries*). Τα ερωτήματα κορυφογραμμής αναζητούν εκείνα τα αντικείμενα για τα οποία δεν υπάρχει κάποιο άλλο αντικείμενο που είναι καλύτερο από αυτά σε όλες τις διαστάσεις-γνωρίσματα, ή όπως συνηθίζεται να λέγεται δεν *κυριαρχεί* (*dominates*) πάνω στο πρώτο. Τα τελευταία χρόνια το πρόβλημα έχει μελετηθεί εκτενώς, και έχει προταθεί ένας μεγάλος αριθμός αλγορίθμων δευτερεύουσας (εξωτερικής) μνήμης.

Σε αυτό το κεφάλαιο εξετάζουμε λεπτομερώς τους scan-based αλγορίθμους κορυφογραμμής, οι οποίοι πραγματοποιούν έναν αριθμό από περάσματα στην βάση δεδομένων προκειμένου να εξάγουν τα αντικείμενα της κορυφογραμμής. Παρόλο που αυτοί οι αλγόριθμοι έχουν σχεδιαστεί να λειτουργούν σε δευτερεύουσα μνήμη, υπάρχουν πολλές λεπτομέρειες υλοποίησης που έχουν παραμεληθεί. Επιπλέον, δεν έχουν μελετηθεί αρκετές σχεδιαστικές επιλογές οι οποίες έχουν ως αποτέλεσμα διαφορετικές εκδοχές των βασικών μεθόδων. Σε αυτή την εργασία πραγματοποιούμε μια εκτεταμένη πειραματική μελέτη χρησιμοποιώντας πραγματικά και συνθετικά δεδομένα. Η εκτεταμένη μελέτη ανέδειξε νέα συμπεράσματα σχετικά με την σχεδίαση και την απόδοση των αλγορίθμων κορυφογραμμής. Πιο συγκεκριμένα, συμπεράναμε ότι συγκεκριμένες επιλογές σχεδίασης μπορούν να έχουν σημαντικό αντίκτυπο στην επίδοση των αλγορίθμων. Όπως επίσης δείχνουμε ότι, αντίθετα με την κοινή πεποίθηση, οι πιο απλοί αλγόριθμοι, μπορούν να είναι πολύ πιο γρήγοροι από μεθόδους που είναι βασισμένες σε προ-επεξεργασία.

## 3.1 Εισαγωγή

Οι *ερωτήσεις κορυφογραμμής*, ή όπως είχε παρουσιαστεί στο [86] ο *skyline operator*, έχουν συγκεντρώσει μεγάλη προσοχή τα τελευταία χρόνια στην κοινότητα διαχείρισης δεδομένων. Δεδομένης μιας βάσης αντικειμένων, οι ερωτήσεις κορυφογραμμής επιστρέφουν τα αντικείμενα τα οποία δεν κυριαρχούνται. Ένα αντικείμενο *κυριαρχεί* σε ένα άλλο, εάν έχει καλύτερες τιμές σε όλες τις ιδιότητες, και αυστηρά καλύτερη τιμή σε τουλάχιστον μία. Η εύρεση της κορυφογραμμής είναι γνωστή και ως το πρόβλημα *Pareto-optimal set*, ή *maximal vectors*.

Αυτή η εργασία μελετά λεπτομερώς μια σημαντική κλάση αλγορίθμων, τους *scan-based αλγορίθμους κορυφογραμμής*. Ένας αλγόριθμος αυτής της κλάσης πραγματο-

ποιεί πολλαπλά περάσματα ενός αρχείο εισόδου, όπου το αρχείο εισόδου στο πρώτο πέρασμα είναι η βάση δεδομένων. Σε ένα μεταγενέστερο πέρασμα, το αρχείο εισόδου είναι το αποτέλεσμα του προηγούμενου περάσματος. Ο αλγόριθμος τερματίζει όταν το αρχείο εξόδου παραμένει άδειο μετά από την ολοκλήρωση ενός περάσματος. Γενικά, κατά τη διάρκεια ενός περάσματος, ο αλγόριθμος διατηρεί στην κύρια μνήμη σε ένα μικρό *παράθυρο* (window) μη συγκρίσιμων αντικειμένων, το οποίο αφαιρεί κυριαρχούμενα αντικείμενα από το αρχείο εισόδου. Κάθε αντικείμενο που δεν κυριαρχείται εγγράφεται στο αρχείο εξόδου.

Παρόλο που οι αλγόριθμοι που μελετάμε έχουν σχεδιαστεί ειδικά να λειτουργούν σε δευτερεύουσα μνήμη, έχει δοθεί λίγη προσοχή σε σημαντικές λεπτομέρειες σχετικά με τη διαχείριση μνήμης. Για παράδειγμα, όλοι οι αλγόριθμοι υποθέτουν ότι η μονάδα μεταφοράς I/O (εισόδου/εξόδου) είναι το αντικείμενο, ενώ σε ένα αληθινό σύστημα είναι ένα σύνολο αντικειμένων. Η εργασία μας αντιμετωπίζει τέτοιες ελλείψεις, εισάγοντας ένα πιο ρεαλιστικό I/O μοντέλο. Επιπλέον, μελετώντας τη διαχείριση των αντικειμένων μέσα στην κύρια μνήμη, εισάγουμε μερικές καινούργιες ενδιαφέρουσες πολιτικές (policies).

Οι κύριες συνεισφορές αυτής της εργασίας συνοψίζονται ως εξής:

1. Με βάση το μοντέλο δευτερεύουσα μνήμης (external memory), προσαρμόσαμε τέσσερις δημοφιλείς αλγόριθμους κορυφογραμμής βασισμένους στην σάρωση, και εξετάζουμε λεπτομερώς διάφορες παραμέτρους υλοποίησης καθώς και θέματα τη διαχείριση μνήμης.

2. Επικεντρωθήκαμε στην μελέτη ενός κύριου χαρακτηριστικού των αλγορίθμων αυτής της κατηγόριας, την διαχείριση των αντικειμένων που διατηρούνται στο παράθυρο της κύριας μνήμης. Πιο συγκεκριμένα, εισάγαμε και μελετήσαμε ένα σύνολο από διαφορετικές πολίτικες (policies) σχετιζόμενες με δυο κύριες εργασίες (tasks): την διάσχιση (traverse) και την απομάκρυνση (eviction) των αντικειμένων του παραθύρου. Και οι δύο αυτές εργασίες μπορούν να έχουν σημαντικές επιπτώσεις στον αριθμό των I/Os (Εισόδων/Εξόδων) καθώς και στον χρόνο επεξεργασίας (CPU Time).

3. Πραγματοποιήσαμε πειραματική αξιολόγηση των αλγορίθμων, χρησιμοποιώντας υλοποιήσεις αυστηρά βασισμένες σε δευτερεύουσα μνήμη και όχι σε προσομοιώσεις. Για την αξιολόγηση χρησιμοποιήθηκαν τόσο συνθετικά όσο και πραγματικά σύνολα δεδομένων. Από την αξιολόγηση προέκυψαν χρήσιμα συμπεράσματα, τα οποία σε αρκετές περιπτώσεις ήταν αντίθετα με την κοινή πεποίθηση, βάση την οποία οι αλγόριθμοι που πραγματοποιούν προ-επεξεργασία (preprocessing) των δεδομένων είναι πιο αποδοτικοί.

4. Τέλος, πραγματοποιήσαμε εκτενή μελέτη των προτεινόμενων πολιτικών. Από την μελέτη προέκυψε, ότι σε δεδομένα με συγκεκριμένα χαρακτηριστικά οι πολιτικές αυτές μπορούν να μειώσουν τον αριθμό των ελέγχων κυριαρχίας (dominance checks) παραπάνω από 50%.

## 3.2 Εισαγωγικά

### 3.2.1 Ορισμοί

Έστω $\mathcal{O}$ ένα σύνολο από *d-dimensional αντικείμενα*. Κάθε αντικείμενο $o \in \mathcal{O}$ αναπαρίσταται από τις *ιδιότητες* του $o = (o^1, o^2, \ldots, o^d)$. Το *domain* κάθε ιδιότητας, είναι το σύνολο των θετικών πραγματικών αριθμών $\mathbb{R}^+$. Υποθέτουμε ότι ένα αντικείμενο $o_1$ είναι *καλύτερο* από ένα άλλο αντικείμενο $o_2$ σε μία ιδιότητα $j$, εάν $o_1^j < o_2^j$. Ένα αντικείμενο $o_1$ *κυριαρχεί* σε ένα άλλο αντικείμενο $o_2$, και συμβολίζεται με $o_1 \succ o_2$, εάν (1) $\forall i \in [1,d]$, $o_1^i \leqslant o_2^i$ και (2) $\exists j \in [1,d]$, $o_1^j < o_2^j$. Η *κορυφογραμμή* ενός συνόλου αντικειμένου $\mathcal{O}$, που συμβολίζεται με $SL(\mathcal{O})$, είναι το σύνολο των αντικειμένων στο $\mathcal{O}$ που δεν κυριαρχούνται από κανένα άλλο αντικείμενο του $\mathcal{O}$. Τυπικά, $SL(\mathcal{O}) = \{o_i \in \mathcal{O} \mid \nexists o_k \in \mathcal{O} : o_k \succ o_i\}$.

### 3.2.2 I/O Μοντέλο Δευτερεύουσα Μνήμης

Αυτή η παράγραφος περιγράφει ένα *μοντέλο δευτερεύουσας μνήμης*, παρόμοιο με αυτό του [21]. Η μονάδα μεταφοράς μεταξύ της κύριας και της δευτερεύουσας μνήμης (δηλαδή του δίσκου) είναι ένα *block*[1]. Κάθε αλγόριθμος δευτερεύουσας μνήμης, όπως οι μέθοδοι κορυφογραμμής, διαβάζει/γράφει blocks από/στα *αρχεία* δίσκου. Υποθέτουμε ότι τα αρχεία αποθηκεύονται συνεχόμενα στον δίσκο, και συνεπώς ένα καινούργιο block γράφεται πάντα στο τέλος του αρχείου.

Συμβολίζουμε ως $N = |\mathcal{O}|$ το μέγεθος της βάσης δεδομένων δηλαδή $N$ είναι ο συνολικός αριθμός των αντικειμένων προς επεξεργασία. Μετράμε το μέγεθος $B$ ενός block σε αντικείμενα (tuples). Παρόμοια, η κύρια μνήμη μπορεί να χωρέσει $M$ αντικείμενα, με τις προϋποθέσεις ότι $M < N$ (και συχνά πολύ μικρότερο) για να δικαιολογηθεί η ανάγκη για αλγορίθμους δευτερεύουσας μνήμης, και $M > 2B$ για να υποστηρίζονται βασικές in-memory λειτουργίες.

Σχετικά με τις Input/Output (I/O) λειτουργίες, υποθέτουμε ότι δεν υπάρχουν buffers εισόδου ή εξόδου, έτσι ώστε τα blocks από τον δίσκο να μεταφέρονται απευθείας στον (αντίστοιχα από) δίσκό από (αντίσχτοιχα στην) την κύρια μνήμη. Ισοδύναμα, οι buffers εισόδου ή εξόδου μοιράζονται με τον αλγόριθμο την ίδια μνήμη μεγέθους $M$.

Κατηγοριοποιούμε τις I/O λειτουργίες σε δύο τρόπους. Ένα *read* μεταφέρει τα δεδομένα από τον δίσκο, όπου ένα *write* μεταφέρει δεδομένα στον δίσκο. Η δεύτερη κατηγοριοποίηση βασίζεται στον αριθμό των blocks που μεταφέρονται. Σημειώνουμε ότι μια λειτουργία read (αντίστοιχα write) μεταφέρει τουλάχιστον ένα block και το πολύ $\lfloor \frac{M}{B} \rfloor$ blocks στην κύρια μνήμη (αντίστοιχα στον δίσκο).

## 3.3 Ένα Μοντέλο για Αλγορίθμους Κορυφογραμμής Βασισμένους στη Σάρωση

Όλοι οι αλγόριθμοι κορυφογραμμής περιέχουν ένα σύνολο αντικειμένων, που ονομάζεται *παράθυρο* (window), το οποίο αποτελείται από πιθανά αντικείμενα κορυφογραμμής, αντικείμενα κορυφογραμμών, ή κάποια αυθαίρετα αντικείμενα γενικά. Μια συνήθη διαδικασία που μπορεί να παρατηρηθεί στους αλγορίθμους είναι η εξής: Με δεδομένο ένα

[1] Το [21] υποθέτει ότι $P$ blocks μπορούν να μεταφερθούν ταυτόχρονα. Σε αυτήν την εργασία ορίζουμε $P = 1$

υποψήφιο αντικείμενο (candidate object) που δε βρίσκεται στο παράθυρο, *διέσχισε* το παράθυρο και καθόρισε αν το υποψήφιο αντικείμενο κυριαρχείται από κάποιο αντικείμενο στο παράθυρο, και αν όχι, καθόρισε επιπρόσθετα τα αντικείμενα στο παράθυρο που κυριαρχεί. Με την ολοκλήρωση της διάσχισης και αν το υποψήφιο αντικείμενο δεν κυριαρχείται, ο αλγόριθμος κορυφογραμμής μπορεί να επιλέξει να το εισάγει στο παράθυρο, πιθανότατα *απομακρύνοντας* κάποια αντικείμενα από το παράθυρο.

Στην προηγούμενη γενική διαδικασία, ταυτοποιούμε και επικεντρωνόμαστε σε δύο επιλογές σχεδίασης. Η πρώτη είναι η *πολιτική διάσχισης* (traversal policy) που καθορίζει τη σειρά με την οποία τα αντικείμενα στο παράθυρο εξετάζονται, και συνεπώς τη σειρά με την οποία γίνονται οι έλεγχοι κυριαρχίας. Αυτή η επιλογή σχεδίασης επηρεάζει άμεσα τον αριθμό των ελέγχων κυριαρχίας που γίνονται και έτσι τον χρόνο εκτέλεσης του αλγορίθμου. Μια ιδανική (αλλά μη ρεαλιστική) πολιτική διάσχισης θα απαιτούσε μόνο έναν έλεγχο κυριαρχίας στην περίπτωση που το υποψήφιο αντικείμενο κυριαρχείται, δηλαδή, την επίσκεψη μόνο σε ένα αντικείμενο παραθύρου που κυριαρχείται, και/ή την επίσκεψη σε αυτά τα αντικείμενα παραθύρου που το υποψήφιο αντικείμενο κυριαρχεί.

Η δεύτερη επιλογή σχεδίασης είναι η *πολιτική απομάκρυνσης* (eviction policy) που καθορίζει ποια αντικείμενα παραθύρου θα απομακρυνθούν προκειμένου να δημιουργηθεί χώρος για το υποψήφιο αντικείμενο. Αυτή η επιλογή βασικά καθορίζει τη δύναμη κυριαρχίας του παραθύρου (dominance power), και έτσι μπορεί έμμεσα να επηρεάσει τόσο τον αριθμό των μελλοντικών ελέγχων κυριαρχίας καθώς και τον αριθμό των μελλοντικών I/O λειτουργιών.

Ορίζουμε τρεις βασικές *πολιτικές διάσχισης παραθύρου*:

- Την *πολιτική σειριακής διάσχισης* (*sequential traversal policy - sqT*), όπου τα αντικείμενα παραθύρου διασχίζονται *σειριακά*, δηλαδή με την σειρά που αποθηκεύονται. Αυτή η πολιτική ακολουθείται από όλους τους υπάρχοντες αλγορίθμους.
- Την *πολιτική τυχαίας διάσχισης* (*random traversal policy - rdT*), όπου τα αντικείμενα παραθύρου διασχίζονται με *τυχαία* σειρά. Αυτή η πολιτική χρησιμοποιείται για να εκτιμηθεί το αποτέλεσμα των άλλων.
- Την *πολιτική διάσχισης που βασίζεται στην εντροπία* (*entropy-based traversal policy - enT*), όπου τα αντικείμενα παραθύρου διασχίζονται σε αύξουσα σειρά με βάση τις τιμές *εντροπίας* τους (δηλαδή, $\sum_{i=1}^{d} \ln(o^i + 1)$). Διαισθητικά, ένα αντικείμενο με χαμηλή τιμή εντροπίας έχει μεγαλύτερη πιθανότητα κυριαρχίας καθώς κυριαρχεί σε μεγάλο όγκο του χώρου.

Επιπρόσθετα με τις βασικές πολιτικές διάσχισης, ορίζουμε διάφορα σχήματα ταξινόμησης (ranking schemes) για τα αντικείμενα, στα οποία θα αναφερθούμε αργότερα. Αυτά τα schemes προσπαθούν να αποτιμήσουν τη πιθανότητα κυριαρχίας ενός αντικειμένου, με την υψηλότερη κατάταξη να υποδηλώνει μεγαλύτερη πιθανότητα.

Συγκεκριμένα, θεωρούμε τις εξής πολιτικές διάσχισης παραθύρου:

- Την *πολιτική διάσχισης που βασίζεται στην ταξινόμηση* (*ranked-based traversal policy - rkT*), όπου αντικείμενα παραθύρου διασχίζονται σε φθίνουσα σειρά βασισμένη στις τιμές της *κατάταξής* τους. Επιπλέον, θεωρούμε τρεις υβριδικές πολιτικές διάσχισης βασισμένες στη κατάταξη (hybrid random-, rank-based traversal policies).

- Την *πολιτική τυχαίας υψηλότερης διάσχισης* (*highest-random traversal policy - hgRdT*), όπου τα $k$ αντικείμενα με τον μεγαλύτερο βαθμό διασχίζονται πρώτα, σε φθίνουσα σειρά με βάση τον βαθμό τους. Μετά, υιοθετείται η πολιτική τυχαίας διάσχισης.
- Την *πολιτική τυχαίας χαμηλότερης διάσχισης* (*lowest- random traversal policy - lwRdT*), όπου τα $k$ αντικείμενα με την χαμηλότερη τιμή κατάταξης συγκρίνονται πρώτα, πριν συνεχίσουμε με την τυχαία διάσχιση.
- Την *πολιτική τυχαίας πρόσφατης διάσχισης* (*recent-random traversal policy - rcRdT*), όπου τα $k$ πιο προσφάτως αναγνωσμένα αντικείμενα συγκρίνονται πρώτα, πριν συνεχίσουμε με την τυχαία διάσχιση.

Επιπλέον, ορίζουμε τρεις *πολιτικές απομάκρυνσης*:

- Την *append πολιτική απομάκρυνσης* (*append eviction policy - apE*), με την οποία το *τελευταίο* αντικείμενο που εισήχθη απομακρύνεται. Αυτή υιοθετείται από την πλειοψηφία των υπάρχον αλγορίθμων.
- Την *πολιτική απομάκρυνσης που βασίζεται στην εντροπία* (*entropy-based eviction policy - enE*), όπου το αντικείμενο με την *υψηλότερη τιμή εντροπίας* απομακρύνεται.
- Την *πολιτική απομάκρυνσης που βασίζεται στην ταξινόμηση* (*ranked-based eviction policy - rkE*), όπου το αντικείμενο με την *χαμηλότερη τιμή κατάταξης* αφαιρείται. Σε περίπτωση ισοπαλίας των τιμών εντροπίας με τις τιμές κατάταξης, το πιο πρόσφατο αντικείμενο απομακρύνεται.

Στη συνέχεια αναφερόμαστε στα ranking schemes που χρησιμοποιούνται στη διάσχιση που βασίζεται στην ταξινόμηση και στις πολιτικές απομάκρυνσης. Σε κάθε αντικείμενο παραθύρου αναθέτετε μια τιμή κατάταξης, που αρχικά ανατίθεται μηδενική τιμή. Διαισθητικά, η κατάταξη χρησιμεύει για να τακτοποιηθούν τα "υποσχόμενα" αντικείμενα με υψηλή δύναμη κυριαρχίας, δηλαδή, αντικείμενα που κυριαρχούν σε μεγάλο αριθμό άλλων αντικειμένων. Ο αλγόριθμος κορυφογραμμής μπορεί να χρησιμοποιήσει αυτή τη πληροφορία προκειμένου να μειώσει τους απαιτούμενους ελέγχους κυριαρχίας ξεκινώντας τη διάσχιση του παραθύρου από υποσχόμενα αντικείμενα, και/ή να απομακρυνθούν μη υποσχόμενα αντικείμενα.

Ορίζουμε τα εξής *ranking schemes*:

- *r0R*: Η κατάταξη ενός αντικειμένου $o$ σε μια χρονική στιγμή $t$, ισούται με τον αριθμό των αντικειμένων που κυριαρχούνται από το $o$ μέχρι την $t$. Δηλαδή, αυτό το ranking scheme μετράει τον αριθμό των αντικειμένων που κυριαρχείται από το $o$.
- *r1R*: αυτή η κατάταξη είναι παρόμοια με την *r0R*. Όμως, παίρνει υπόψιν της τον αριθμό των αντικειμένων που κυριαρχούνται από τα αντικείμενα που κυριαρχεί το $o$. Έστω ότι το $rank(o)$ συμβολίζει την κατάταξη του αντικειμένου $o$. Υποθέτουμε ότι το αντικείμενο $o_1$ κυριαρχεί στο $o_2$. Τότε, η κατάταξη $o_1$ μετά που έχει κυριαρχήσει στο $o_2$ είναι ίση με το $rank(o_1) + rank(o_2) + 1$.
- *r2R*: αυτή η κατάταξη αναθέτει δύο τιμές για κάθε αντικείμενο $o$, την *r1R* τιμή του, καθώς και τον αριθμό που το $o$ συγκρίθηκε με άλλο αντικείμενο και κανένα δεν ήταν κυριαρχούμενο (δηλαδή, τον αριθμό των μη-συγκρίσιμων έλεγχων

κυριαρχίας). Η *r1R* τιμή λαμβάνεται υπόψιν για την κατάταξη των αντικειμένων παραθύρου, ενώ ο αριθμός των μη συγκρίσιμων ελέγχων λαμβάνεται υπόψιν στην περίπτωση ισοβαθμίας: όσο περισσότερους μη συγκρίσιμων ελέγχων έχει κάθε αντικείμενο, τόσο χαμηλότερη είναι η κατάταξη του.

Τέλος, κάποιοι αλγόριθμοι κορυφογραμμής που βασίζονται στην σάρωση πραγματοποιούν ένα στάδιο προεπεξεργασίας με το οποίο τα αντικείμενα εισόδων *ταξινομούνται* σύμφωνα με μια μονότονη monotone συνάρτηση.

Σε αυτή την εργασία, θεωρούμε τις τρεις πιο συνήθεις *συναρτήσεις ταξινόμησης*:

- Τη *συνάρτηση ταξινόμησης εντροπίας* (*entropy sorting function - Entr*) που ορίζεται ως $Entr(o) = \sum_{i=1}^{d} \ln(o^i + 1)$.
- Τη *συνάρτηση ταξινόμησης αθροίσματος* (*sum sorting function - Sum*), που ορίζεται ως το άθροισμα των τιμών των ιδιοτήτων των αντικειμένων.
- Τη *συνάρτηση ελάχιστης ταξινόμησης* (*minimum sorting function - minC*), το οποίο ταξινομεί τα αντικείμενα σε αύξουσα σειρά με βάση την ελάχιστη τιμή ιδιότητας. Η *Sum* συνάρτηση χρησιμοποιείται για να επιλύσει ισοπαλίες και να εξασφαλίσει τη μονοτονία.

## 3.4 Προσαρμογή Αλγορίθμων με βάση το I/O Μοντέλο

### 3.4.1 Block Nested Loop Algorithm (BNL)

Ο αλγόριθμος *Block Nested Loop* (BNL) [86] είναι ένας από τους πρώτους αλγορίθμους δευτερεύουσας μνήμης για υπολογισμό κορυφογραμμής. Όλοι οι υπολογισμοί στον BNL πραγματοποιούνται κατά τη διάρκεια της διάσχισης του παραθύρου. Έτσι, ο BNL χρησιμοποιεί ένα παράθυρο τόσο μεγάλο όσο επιτρέπει η μνήμη. Συγκεκριμένα, έστω $W$ ο αριθμός των αντικειμένων που είναι αποθηκευμένα σε ένα παράθυρο, και έστω $O_b$ ο αριθμός των αντικειμένων που είναι προγραμματισμένα να γράφουν στον δίσκο (δηλαδή, στο buffer εξόδου). Η υπόλοιπη μνήμη μεγέθους $I_b = M - W - O_b$ χρησιμεύει ως buffer εισόδου, για να ανακτήσει αντικείμενα από τον δίσκο. Σημειώνουμε ότι το μέγεθος των I/O buffers $I_b$ και $O_b$ ποικίλει κατά την εκτέλεση του BNL, και εξαρτάται από τον περιορισμό ότι το μέγεθος του buffer εισόδου είναι πάντα τουλάχιστον ένα disk block, δηλαδη, $I_b \geq B$, και ότι το μέγεθος του buffer εξόδου υπερβαίνει ένα disk block, δηλαδή, $O_b \leq B$. Αναφερόμαστε αργότερα στο πως ο BNL εφαρμόζει αυτούς τους περιορισμούς.

Στη συνέχεια περιγράφουμε τη διαχείριση μνήμης στον BNL αλγόριθμο. Ο BNL πραγματοποιεί έναν αριθμό από περάσματα, σε καθένα από τα οποία διαβάζεται ένα αρχείο εισόδου. Για το πρώτο πέρασμα, το αρχείο εισόδου είναι η βάση δεδομένων, ενώ το αρχείο εισόδου σε επόμενα περάσματα δημιουργείται σε προηγούμενα περάσματα. Ο BNL τερματίζει όταν το αρχείο εισόδου είναι άδειο. Κατά τη διάρκεια ενός περάσματος, το αρχείο εισόδου διαβάζεται σε *chunks*, δηλαδή σε σύνολα από blocks. Συγκεκριμένα, κάθε λειτουργία ανάγνωσης μεταφέρει στην κύρια μνήμη ακριβώς $\lfloor \frac{I_b}{B} \rfloor$ blocks από τον δίσκο, περιλαμβάνοντας έτσι 1 τυχαίο και $\lfloor \frac{I_b}{B} \rfloor - 1$ διαδοχικά I/Os. Όμως, όταν γεμίζει το buffer εξόδου, δηλαδή, $O_b = B$, μια λειτουργία γραφής (write operation) μεταφέρει στον δίσκο ακριβώς 1 block και υφίσταται 1 τυχαίο I/O.

Τώρα θα αναφερθούμε στο τι συμβαίνει όταν ένα chunk από αντικείμενα μεταφέρεται στο buffer εισόδου στην κύρια μνήμη. Για κάθε αντικείμενο $o$ στο buffer εισόδου, ο BNL διασχίζει το παράθυρο, υιοθετώντας την πολιτική διαδοχικής διάσχισης ($sqT$). Τότε ο BNL έναν διπλό (two-way) έλεγχο κυριαρχίας μεταξύ του $o$ και ενός αντικειμένου παραθύρου $w$. Αν το $o$ κυριαρχείται από το $w$, το $o$ απορρίπτεται και η διάσχιση σταματάει. Διαφορετικά, αν το $o$ κυριαρχεί στο $w$, το αντικείμενο $w$ απλά αφαιρείται από το παράθυρο.

Στο τέλος της διάσχισης, αν το $o$ δεν έχει απορριφθεί, επισυνάπτεται στο παράθυρο. Αν το $W$ γίνει μεγαλύτερο από το $M - O_b - B$, ο BNL πρέπει να απομακρύνει ένα αντικείμενο από το παράθυρο στο buffer εξόδου προκειμένου να διασφαλίσει την ύπαρξη αρκετού χώρου στο buffer εισόδου. Συγκεκριμένα, ο BNL εφαρμόζει την πολιτική απομάκρυνσης ($apE$), και επιλέγει τα αντικείμενα που έχουν εισαχθεί τελευταία, τα οποία είναι $o$, για να μετακινηθούν buffer εξόδου. Αν μετά από αυτήν την απομάκρυνση, το buffer εξόδου περιέχει $O_b = B$ αντικείμενα, τα περιεχόμενά του εγγράφονται στο αρχείο, το οποίο θα αποτελέσει το αρχείο εισόδου του επόμενου περάσματος.

Ένα τελευταίο ζήτημα είναι πώς ο BNL ταυτοποιεί ότι ένα αντικείμενο $o$ είναι αντικείμενο κορυφογραμμής, καθώς ο BNL πρέπει να διασφαλίσει ότι το $o$ έχει περάσει από έλεγχο κυριαρχίας με όλα τα εναπομείναντα αντικείμενα που είναι στο αρχείο εισόδου. Όταν αυτό μπορεί να εγγυηθεί, το $o$ αφαιρείται από το παράθυρο και επιστρέφεται ως αποτέλεσμα. Αυτή η διαδικασία εφαρμόζεται μέσω ενός μηχανισμού *timestamp*, λεπτομέρειες για τον οποίο μπορούν να βρεθούν στο [86].

### 3.4.2 Sort Filter Skyline Algorithm (SFS)

Ο *Sort Filter Skyline* (SFS) [108] αλγόριθμος είναι παρόμοιος με τον BNL με μια βασική εξαίρεση: η βάση δεδομένων πρώτα ταξινομείται με μια εξωτερική διαδικασίας ταξινόμησης με βάση μια μονοτονική συνάρτηση βαθμολόγησης (monotonic scoring function). Ο SFS μπορεί να χρησιμοποιεί οποιαδήποτε συνάρτηση έχει οριστεί στην Ενότητα 3.3.

Παρομοίως με τον BNL, ο SFS αλγόριθμος χρησιμοποιεί μια πολιτική σειριακής διάσχισης παραθύρου ($sqT$) και την πολιτική απομάκρυνσης $apE$. Υπάρχουν όμως, δύο διαφοροποιήσεις σε σχέση με τον BNL. Λόγω της ταξινόμηση, οι έλεγχοι κυριαρχίας κατά τη διάρκεια της διάσχισης παραθύρου είναι μονοί (one-way). Δηλαδή ένα αντικείμενο $o$ ελέγχεται μόνο για κυριαρχία από ένα αντικείμενο παραθύρου $w$. Επιπλέον, η ταυτοποίηση κορυφογραμμής στον SFS είναι πολύ πιο απλή από ότι στον BNL. Στο τέλος κάθε περάσματος, όλα τα αντικείμενα παραθύρων είναι σίγουρο ότι αποτελούν αποτελέσματα και επομένως αφαιρούνται και επιστρέφονται.

### 3.4.3 Linear Elimination Sort for Skyline Algorithm (LESS)

Ο *Linear Elimination Sort for Skyline* (LESS) [168] αλγόριθμος βελτιώνει μια βασική ιδέα των SFS, πραγματοποιώντας ελέγχους κυριαρχίας κατά τη διάρκεια της διαδικασίας εξωτερικής ταξινόμησης. Ας θυμηθούμε ότι η καθιερωμένη εξωτερική ταξινόμηση πραγματοποιεί έναν αριθμό από περάσματα στα δεδομένα εισόδου. Το αποκαλούμενο zero pass ή sort pass μεταφέρει στην κύρια μνήμη $M$ αντικείμενα, τα ταξινομεί in-memory και τα εγγράφει σε δίσκο. Τότε το $k$-ο πέρασμα της εξωτερικής ταξινόμησης, διαβάζει στα blocks της κύριας μνήμης μέχρι $\lfloor M/B \rfloor - 1$ αρχεία που έχουν δημιουργηθεί σε προηγούμενα περάσματα, συγχωνεύει τα αντικείμενα και εγγράφει τα αποτελέσματα

στον δίσκο.

Ο LESS τροποποιεί τη διαδικασία εξωτερικής ταξινόμησης με δύο τρόπους. Πρώτον, κατά τη διάρκεια του zero pass, ο LESS διατηρεί ένα παράθυρο μεγέθους $W_0$ αντικειμένων ως ένα φίλτρο απαλοιφής κατά τη διάρκεια της ταξινόμησης. Έτσι η υπολειπόμενη μνήμη $M - W_0$ χρησιμοποιείται για την in-memory ταξινόμηση. Το παράθυρο αρχικά είναι γεμάτο μετά την ανάγνωση των πρώτων $M - W_0$ αντικειμένων επιλέγοντας εκείνα με τις μικρότερες τιμές εντροπίας. Τότε για κάθε ανάγνωση αντικειμένου $o$ από τον δίσκο και πριν την in-memory ταξινόμησή του, ο LESS πραγματοποιεί μια διάσχιση παραθύρου. Συγκεκριμένα, ο LESS χρησιμοποιεί την πολιτική διαδοχικής διάσχισης ($sqT$) πραγματοποιώντας έναν μονό (one-way) έλεγχο κυριαρχίας, δηλαδή ελέγχει μόνο αν το $o$ κυριαρχείται. Συγκρίνοντας τα αντικείμενα εισόδου με το παράθυρο, ταυτοποιείται το αντικείμενο με την μικρότερη εντροπία $o_h$. Τότε, ξεκινά μια άλλη διαδοχική διάσχιση παραθύρου ($sqT$), που όμως αυτή τη φορά ελέγχει αν το $o_h$ κυριαρχεί στα αντικείμενα του παραθύρου. Αν το $o_h$ επιβιώσει, προστίθεται στο παράθυρο, απομακρύνοντας ένα αντικείμενο με υψηλότερη τιμή εντροπίας, δηλαδή εφαρμόζεται η πολιτική απομάκρυνσης που βασίζεται στην εντροπία ($enE$).

Η δεύτερη πρόκληση στη διαδικασία εξωτερικής ταξινόμησης είναι κατά τη διάρκεια του τελευταίου περάσματος, όπου ο LESS διατηρεί ένα παράθυρο μεγέθους $W$ αντικειμένων. Σε αυτό το πέρασμα, όπως και σε κάθε επακόλουθο πέρασμα επεξεργασίας κορυφογραμμής, ο LESS λειτουργεί ακριβώς όπως ο SFS. Δηλαδή χρησιμοποιείται η πολιτική διαδοχικής διάσχισης ($sqT$), γίνονται μονοί έλεγχοι κυριαρχίας, και τα αντικείμενα παραθύρου απομακρύνονται σύμφωνα με την πολιτική ($epE$).

### 3.4.4 Randomized Multi-pass Streaming Algorithm (RAND)

Στον *Randomized multi-pass streaming* (RAND) αλγόριθμο [295], κάθε πέρασμα αποτελείται από τρία στάδια, όπου κάθε στάδιο διατρέχει τα αρχεία εισόδου του προηγούμενο σταδίου. Για αυτό, κάθε πέρασμα ουσιαστικά αντιστοιχεί σε τρεις αναγνώσεις του αρχείου εισόδου. Στο πρώτο στάδιο, διαβάζεται το αρχείο εισόδου και έπειτα γεμίζει ένα παράθυρο μέγιστου μεγέθους $W = M - B$ με τυχαίως επιλεγμένα αντικείμενα εισόδου (χρησιμοποιώντας reservoir sampling).

Στο δεύτερο στάδιο, το αρχείο εισόδου διαβάζεται πάλι αλλά αυτή τη φορά διαβάζονται ένα ένα τα blocks, ενώ το παράθυρο των $W$ αντικειμένων παραμένει στη μνήμη. Για κάθε αντικείμενο εισόδου $o$, ο αλγόριθμος διασχίζει το παράθυρο σε σειριακή σειρά ($sqT$), πραγματοποιώντας μονούς έλεγχους κυριαρχίας. Αν ένα αντικείμενο παραθύρου $w$ κυριαρχείται από $o$, το $w$ αντικαθίσταται από το $o$. Σημειώνουμε ότι, στο τέλος κάθε σταδίου, όλα τα αντικείμενα παραθύρου είναι αντικέιμενα κορυφογραμμής, και μπορούν να επιστραφούν. Όμως, δεν αφαιρούνται από την μνήμη.

Στο τρίτο στάδιο, για κάθε αντικείμενο εισόδου $o$, ο RAND πραγματοποιεί άλλη μία σειριακή διάσχιση του παραθύρου ($sqT$), αυτή τη φορά πραγματοποιώντας έναν αντίστροφο μονό (inverse one-way) έλεγχο κυριαρχίας. Αν ο $o$ κυριαρχείται από ένα αντικείμενο παραθύρου $w$, ή αν το $o$ και το $w$ αντιστοιχούν στο ίδιο αντικείμενο, ο RAND απορρίπτει το $o$. Διαφορετικά εγγράφεται σε ένα αρχείο στον δίσκο, χρησιμεύοντας ως αρχείο εισόδου για το επόμενο πέρασμα. Στο τέλος αυτού του σταδίου, η μνήμη αδειάζει.

## 3.5 Πειραματική Ανάλυση

### 3.5.1 Περιβάλλον

#### 3.5.1.1 Σύνολα Δεδομένων

Η πειραματική μας αξιολόγηση περιλαμβάνει τόσο συνθετικά όσο και πραγματικά σύνολα δεδομένων.

Για να κατασκευάσουμε συνθετικά σύνολα δεδομένων, θεωρούμε τους τρεις καθιερωμένους τύπους κατανομής που χρησιμοποιούνται στην βιβλιογραφία. Συγκεκριμένα, οι κατανομές είναι: anti- correlated (ANT), correlated (CORR), και η ανεξάρτητη ή independent (IND). Τα συνθετικά σύνολα δεδομένων δημιουργούνται χρησιμοποιώντας τον generator που αναπτύχθηκε από τους συγγραφείς στο [86].

Πραγματοποιούμε επίσης πειράματα σε τρία πραγματικά σύνολα δεδομένων. Το *NBA* σύνολο δεδομένων αποτελείται από 17,264 αντικείμενα, που περιέχουν στατιστικά για παίκτες μπάσκετ. Για κάθε παίκτη θεωρούμε 5 στατιστικά στοιχεία (π.χ., βαθμούς, ριμπάουντ, ασίστ, κλεψίματα, και μπλοκ). Το *House* είναι ένα 6-διάστατο σύνολο δεδομένων που αποτελείται από 127,931 αντικείμενα. Κάθε αντικείμενο, αντιπροσωπεύει τα χρήματα που δαπανά το χρόνο μια αμερικάνικη οικογένεια για έξι διαφορετικούς τύπους δαπανών (π.χ., αέριο, ηλεκτρισμός, νερό, θέρμανση κτλ). Τέλος, το *Colour* είναι ένα 9-διάστατο σύνολο δεδομένων, το οποίο περιέχει 68,040 αντικείμενα, αντιπροσωπεύοντας τις πρώτες τρεις στιγμές μιας RGB χρωματικής κατανομής της εικόνας.

#### 3.5.1.2 Υλοποίηση

Όλοι οι αλγόριθμοι που περιγράφονται στην Ενότητα 3.3, γράφτηκαν σε C++, μεταγλωττίστηκαν με το πρόγραμμα gcc, τα πειράματα πραγματοποιήθηκαν σε 2.6GHz CPU. Προκειμένου να μελετήσουμε με ακρίβεια την επίδραση των I/O λειτουργιών, απενεργοποιούμε το operating system caching, και πραγματοποιούμε απευθείας και συγχρονισμένες I/O's.

Το μέγεθος του κάθε αντικειμένου είναι 100 bytes, όπως και στην πειραματική αξιολόγηση των εργασιών που είσηγαγαν τους αλγορίθμους που εξετάζουμε. Τέλος, το default μέγεθος του block είναι 2048 bytes. Επομένως, με το default setting, κάθε block περιέχει 20 αντικείμενα.

#### 3.5.1.3 Μετρικές

Για να μετρήσουμε την αποδοτικότητα των αλγορίθμων, μετράμε: (1) τον αριθμό των λειτουργιών I/O του δίσκου, οι οποίες διαχωρίζονται σε τέσσερις κατηγορίες, λειτουργίες ανάγνωσης και γραφής (read, write operations) που πραγματοποιούνται κατά τη διάρκεια του σταδίου προ-επεξεργασίας (δηλαδή, ταξινόμηση) εάν υπάρχουν, και οι λειτουργίες ανάγνωσης και γραφής που πραγματοποιούνται κατά τη διάρκεια του κύριου υπολογιστικού σταδίου, (2) τον αριθμό των ελέγχων κυριαρχίας (3) τον χρόνο που απαίτησε μόνο η CPU επεξεργασία, που αναφέρεται ως CPU Time και μετριέται σε seconds, (4) τον συνολικό χρόνο εκτέλεσης, που αναφέρεται ως Total Time και μετριέται σε seconds. Σε κάθε περίπτωση οι χρονικές τιμές που αναφέρονται είναι οι μέσες τιμές 5 εκτελέσεων.

**Πίνακας 3.1:** Παράμετροι

| Description | Parameter | Values |
|---|---|---|
| Number of Objects | $N$ | 50k, 100K, **500K**, 1M, 5M |
| Number of Attributes | $d$ | 3, **5**, 7, 9, 15 |
| Memory Size | $M/N(\%)$ | 0.15%, 0.5% **1%**, 5%, 10% |
| Block Size (Bytes) | $B \cdot$ object size | 1024, **2048**, 4096 |
| Distribution | - | INT, CORR, ANT |

### 3.5.2 Σύγκριση Αλγορίθμων

Ο Πίνακας 3.1 παραθέτει τις παραμέτρους και τα εύρη των τιμών που εξετάζονται. Σε κάθε πείραμα, μεταβάλλουμε μία μόνο παράμετρο και ορίζουμε τις υπόλοιπες στις default (bold) τιμές. Οι SFS και LESS ταξινομούν με βάση την συνάρτηση εντροπίας. Κατά το μηδενικό πέρασμα στον LESS, το παράθυρο ρυθμίζεται στο ένα block.

#### 3.5.2.1 Μεταβάλλοντας τον Αριθμό των Αντικειμένων

Σε αυτό το πείραμα, μεταβάλλουμε τον αριθμό των αντικειμένων από 50K μέχρι και 5M και μετράμε τον συνολικό χρόνο, τον αριθμό των I/O's, των ελέγχων κυριαρχίας, και τον CPU χρόνο, στα Σχήμα 3.1~3.4.

Τα σημαντικά συμπεράσματα από το Σχήμα 3.1 είναι δύο. Πρώτον, οι RAND και BNL έχουν καλύτερες επιδόσεις από τις άλλες μεθόδους σε anti-correlated σύνολα δεδομένων. Αυτό εξηγείται ως εξής. Σημειώνουμε ότι ο CPU χρόνος περιλαμβάνει τον χρόνο που ξοδεύτηκε για τους ελέγχους κυριαρχίας, τη ταξινόμηση δεδομένων στη περίπτωση των LESS/SFS, και της ταυτοποίησης κορυφογραμμής (skyline identification), στη περίπτωση του BNL. Από το Σχήμα 3.4 μπορούμε να εξάγουμε ότι ο BNL ξοδεύει πολύ χρόνο στην ταυτοποίηση κορυφογραμμής. Ο BNL απαιτεί τον ίδιο ή περισσότερο CPU χρόνου από τον RAND, ενώ ο BNL πραγματοποιεί λιγότερους ελέγχους κυριαρχίας από τον RAND. Αυτό είναι πιο προφανές στην περίπτωση ανεξάρτητων και συσχετιζόμενων συνόλων δεδομένων όπου το κόστος των ελέγχων κυριαρχίας είναι χαμηλότερο σε σχέση με το anti-correlated σύνολο δεδομένων. Σε αυτά τα σύνολα δεδομένων, ο BNL CPU χρόνος αυξάνεται απότομα καθώς η πληθικότητα αυξάνεται.

Το δεύτερο συμπέρασμα είναι ότι σε ανεξάρτητα και συσχετιζόμενα σύνολα δεδομένων, η απόδοση του BNL γρήγορα πέφτει καθώς αυξάνει η πληθικότητα. Αυτό οφείλεται στην αύξηση του μεγέθους του παράθυρου, το οποίο με τη σειρά του κάνει τη διατήρηση του παραθύρου και την ταυτοποίηση κορυφογραμμής πιο δύσκολη.

Το Σχήμα 3.2 δείχνει τις I/O λειτουργίες που πραγματοποιούνται από τους αλγορίθμους. Παρατηρούμε ότι ο BNL έχει καλύτερες επιδόσεις από άλλες μεθόδους σε σχεδόν όλα τα περιβάλλοντα. Συγκεκριμένα, στο συσχετισμένο σύνολο, ο LESS είναι πολύ κοντά στον BNL. Επίσης, παρατηρούμε ότι, γενικά, το ποσοστό των λειτουργιών γραφής στον LESS και στον SFS είναι πολύ μεγαλύτερο από ότι στον BNL και RAND. Πρέπει να σημειώσουμε ότι, οι λειτουργίες γραφής είναι γενικά πιο εκτεταμένες σε σύγκριση με τις λειτουργίες ανάγνωσης. Τέλος, για τους LESS και SFS, μπορούμε να παρατηρήσουμε ότι το μεγαλύτερο μέρος των I/O λειτουργιών πραγματοποιείται κατά το στάδιο της ταξινόμησης.

Σχετικά με τον αριθμό των ελέγχων κυριαρχίας, που φαίνεται στο Σχήμα 3.3, οι LESS και SFS πραγματοποιούν τους λιγότερους, ενώ ο RAND τους περισσότερους, σε όλες τις περιπτώσεις. Το Σχήμα 3.4 δείχνει τον CPU time που δαπανάται από

τις μεθόδους. Ο SFS δαπανά περισσότερο CPU time από τον LESS παρόλο που πραγματοποιούν παρόμοιο αριθμό ελέγχων κυριαρχίας. Αυτό εξηγείται από το γεγονός ότι ο SFS ταξινομεί έναν μεγαλύτερο αριθμό αντικειμένων από ότι ο LESS. Τέλος, όπως αναφέρθηκε προηγουμένως, ο BNL δαπανά αρκετό CPU time για την ταυτοποίηση κορυφογραμμής.

#### 3.5.2.2 Μεταβάλλοντας τον Αριθμό των Διαστάσεων

Σε αυτό το πείραμα διερευνούμε την απόδοση καθώς μεταβάλλουμε τον αριθμό των διαστάσεων από 3 σε 15. Στο Σχήμα 3.5 όπου απεικονίζεται ο συνολικός χρόνος, η απόδοση όλων των μεθόδων είναι περίπου η ίδια για τα anti-correlated και τα independent σύνολα δεδομένων, καθώς αυξάνονται οι διαστάσεις. Σε correlated σύνολα δεδομένων, η κορυφογραμμή μπορεί να χωρέσει στην κύρια μνήμη, και για αυτό το λόγο οι BNL και RAND απαιτούν λίγα περάσματα, ενώ οι SFS και LESS σπαταλούν χρόνο ταξινομώντας τα δεδομένα.

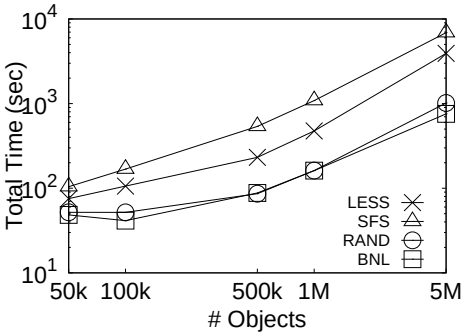
Σχετικά με τις I/O's (Σχήμα 3.6), ο BNL έχει καλύτερες επιδόσεις από άλλες μεθόδους σε όλες τις περιπτώσεις, ενώ ο LESS είναι η δεύτερη καλύτερη μέθοδος. Ομοίως, όπως φαίνεται στο Σχήμα 3.2, οι LESS και SFS πραγματοποιούν εμφανώς περισσότερες λειτουργίες γραφής σε σχέση με τους BNL και RAND. Το Σχήμα 3.7 δείχνει ότι οι LESS και SFS έχουν καλύτερες επιδόσεις από την άλλη μέθοδο, πραγματοποιώντας τον ίδιο αριθμό ελέγχων κυριαρχίας. Τέλος, ο CPU time παρουσιάζεται στο Σχήμα 3.8, όπου και πάλι το κόστος της ταυτοποίησης κορυφογραμμής είναι παρατηρήσιμο για τον BNL.

#### 3.5.2.3 Μεταβάλλοντας το Μέγεθος της Μνήμης

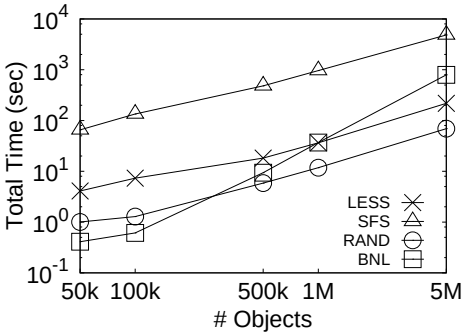
Στο Σχήμα 3.9, μεταβάλλουμε το μέγεθος της διαθέσιμης μνήμης. Γενικά, ο συνολικός χρόνος εδώ ακολουθεί τη τάση των I/O λειτουργιών. Παρατηρούμε ότι ο απαιτούμενος χρόνος για όλες τις μεθόδους μειώνεται απότομα για μεγέθη μνήμης μέχρι και 1%. Όμως, πέρα από αυτό το σημείο, ο χρόνος είναι σχεδόν σταθερός καθώς το μέγεθος της μνήμης αυξάνεται, με την εξαίρεση του BNL, όπου ο χρόνος αυξάνεται ελαφρά (λόγω του κόστους ταυτοποίησης κορυφογραμμής με βάση και το μέγεθος του παραθύρου).

#### 3.5.2.4 Μεταβάλλοντας το Μέγεθος του Block

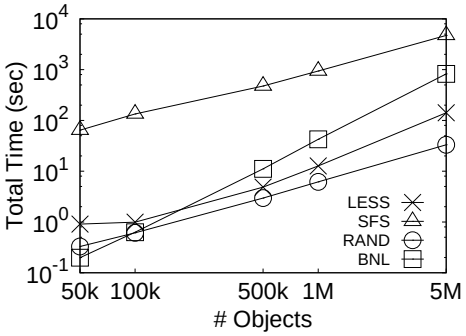
Σε αυτό το πείραμα μεταβάλλουμε το μέγεθος του block, από τα 1024 μέχρι και 4096 bytes. Σημειώνουμε ότι, όπως αναφέραμε και προηγουμένως, το μέγεθος του αντικειμένου είναι 100 bytes και συνεπώς, ένα block μεγέθους 1024 bytes περιέχει 1024/100 = 10 αντικείμενα. Το Σχήμα 3.10 δείχνει τον συνολικό χρόνο, ο οποίος ακολουθεί την I/O τάση, αφού όλοι οι άλλες παράμετροι παραμένουν σταθεροί.




(α′) Anti-correlated

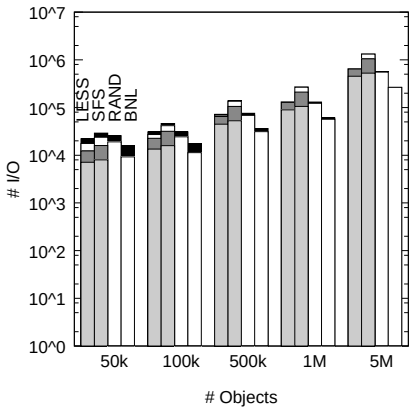



(β′) Independent

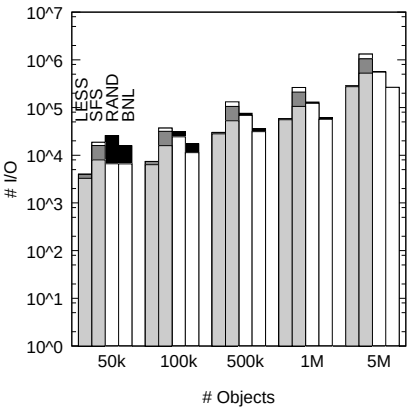



(γ′) Correlated

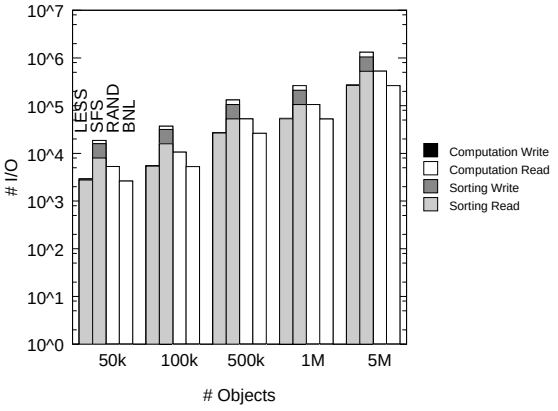
**Σχήμα 3.1:** Total Time: μεταβάλλοντας τον αριθμό των αντικειμένων




(α′) Anti-correlated

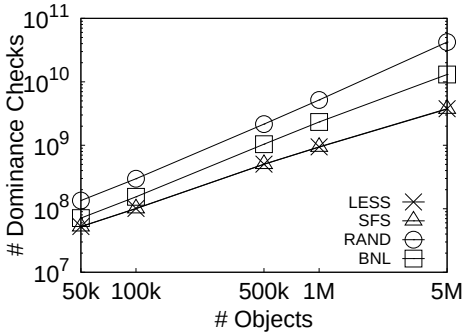



(β′) Independent

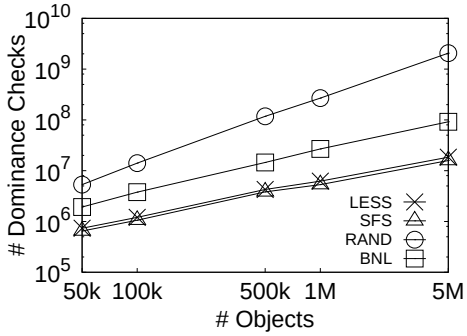



(γ′) Correlated

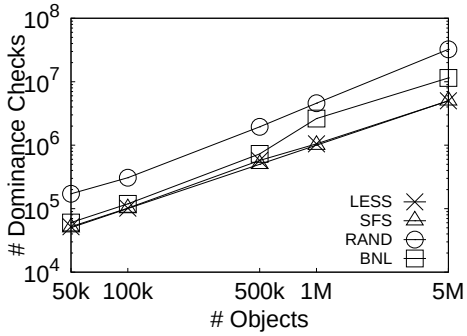
**Σχήμα 3.2:** I/O Operations: μεταβάλλοντας τον αριθμό των αντικειμένων




(α′) Anti-correlated

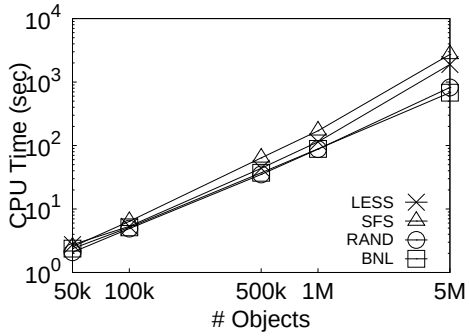



(β′) Independent

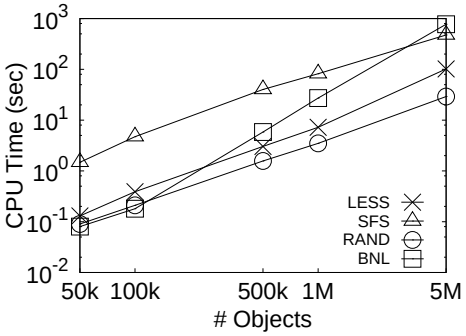



(γ′) Correlated

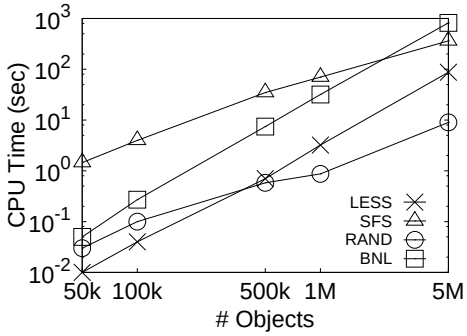
**Σχήμα 3.3:** Dominance Checks: μεταβάλλοντας τον αριθμό των αντικειμένων




(α′) Anti-correlated

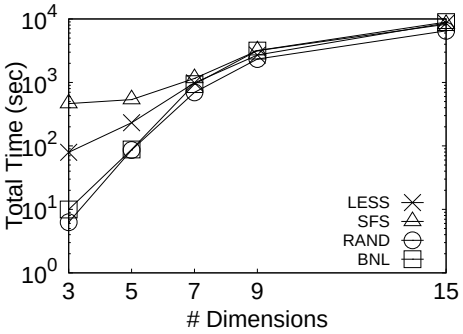



(β′) Independent

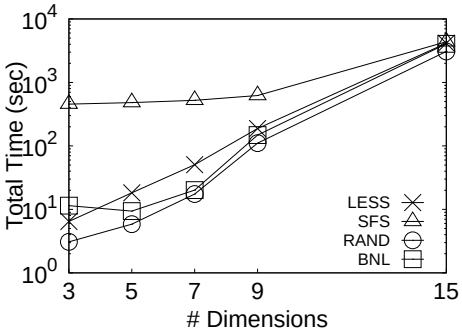



(γ′) Correlated

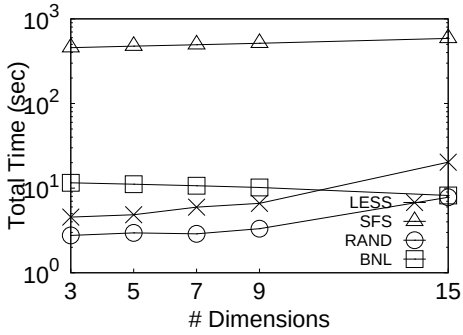
**Σχήμα 3.4:** CPU Time: μεταβάλλοντας τον αριθμό των αντικειμένων




(α′) Anti-correlated

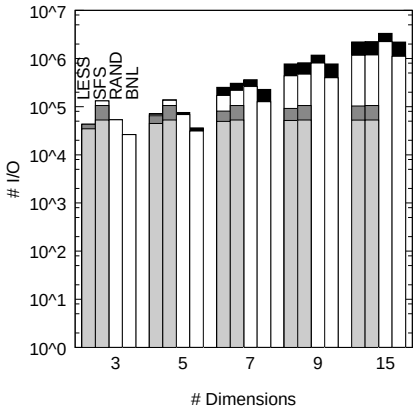



(β′) Independent

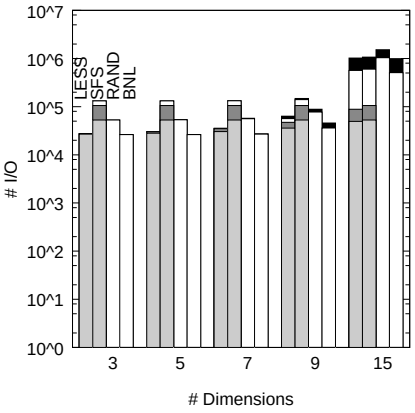



(γ′) Correlated

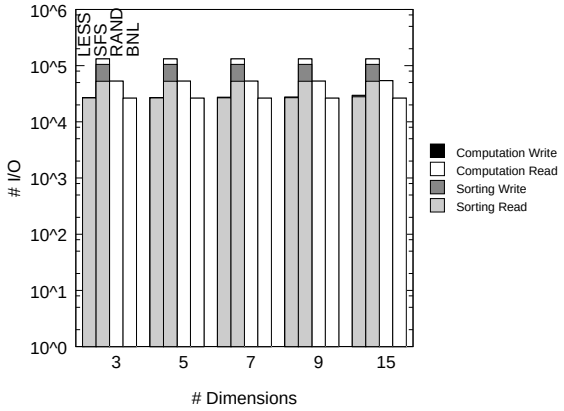
**Σχήμα 3.5:** Total Time: μεταβάλλοντας τον αριθμό των διαστάσεων




(α′) Anti-correlated

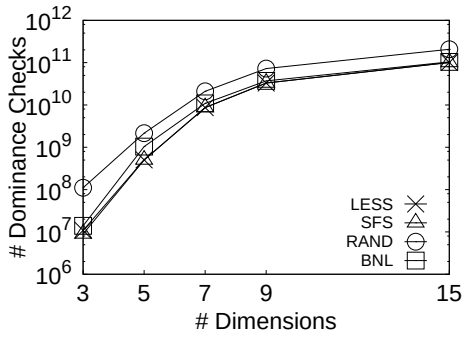



(β′) Independent

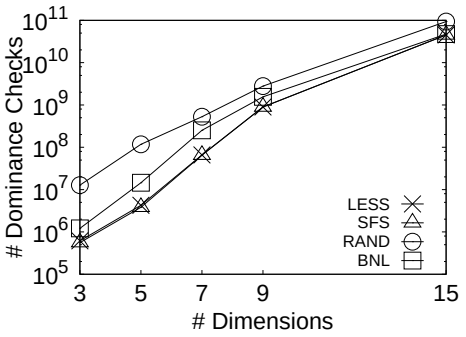



(γ′) Correlated

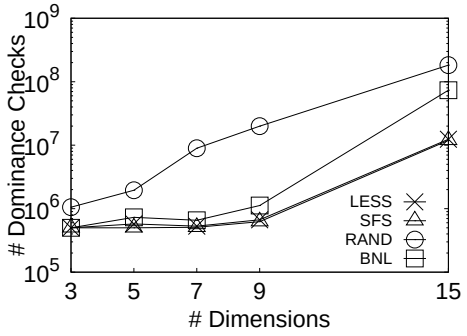
**Σχήμα 3.6:** I/O Operations: μεταβάλλοντας τον αριθμό των διαστάσεων




(α′) Anti-correlated

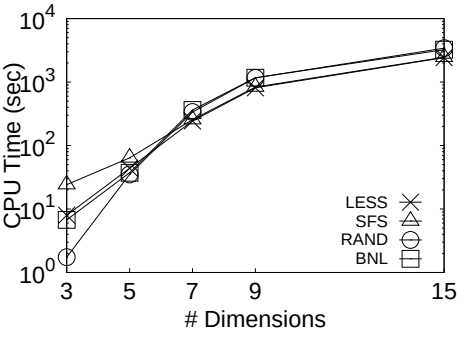



(β′) Independent

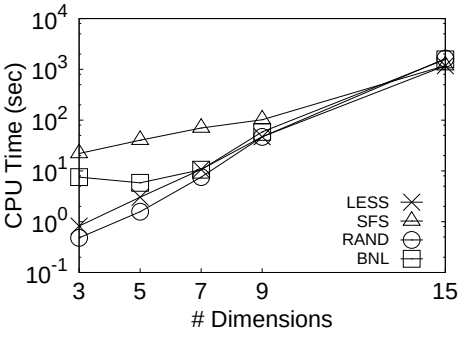



(γ′) Correlated

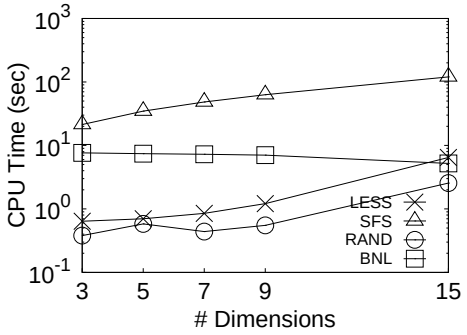
**Σχήμα 3.7:** Dominance Checks: μεταβάλλοντας τον αριθμό των διαστάσεων




(α′) Anti-correlated

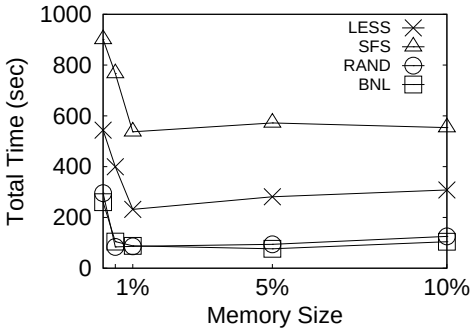



(β′) Independent

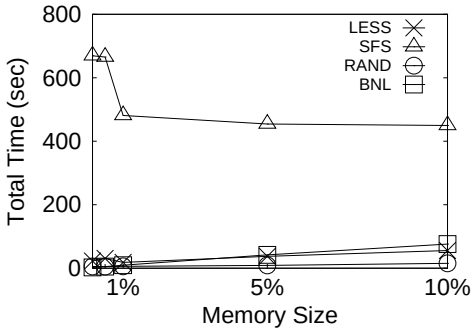



(γ′) Correlated

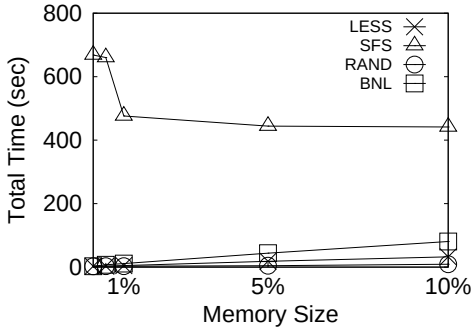
**Σχήμα 3.8:** CPU Time: μεταβάλλοντας τον αριθμό των διαστάσεων




(α′) Anti-correlated




(β′) Independent




(γ′) Correlated

**Σχήμα 3.9:** Total Time: varying memory size

#### 3.5.2.5 Πραγματικά Σύνολα Δεδομένων

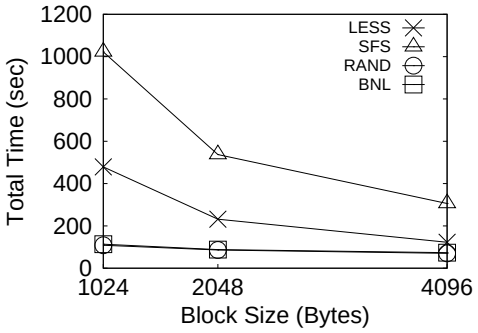
Σε αυτό το πείραμα, αξιολογούμε τις μεθόδους χρησιμοποιώντας τα πραγματικά σύνολα δεδομένων που αναφέρθηκαν στην Ενότητα 3.5.1. Ο Πίνακας 3.2 συνοψίζει τα αποτελέσματα, παρουσιάζοντας τον συνολικό χρόνο που απαιτήθηκε από όλες τις μεθόδους. Παρατηρούμε ότι ο BNL έχει καλύτερες επιδόσεις από τις άλλες μεθόδους σε όλα τα σύνολα δεδομένων αναφορικά με τον συνολικό χρόνο. Ο RAND έχει καλύτερες επιδόσεις με τις άλλες μεθόδους σε όλες τις περιπτώσεις, ενώ ο SFS έχει τις χειρότερες. Σημειώνουμε ότι, σε House και Colour σύνολα δεδομένων, ο RAND πραγματοποιεί περισσότερους ελέγχους κυριαρχίας, και περισσότερες I/O λειτουργίες, από ότι ο LESS. Όμως, ο LESS απαιτεί περισσότερο συνολικό χρόνο, λόγω του μεγαλύτερου αριθμού των λειτουργιών γραφής, και του CPU time που δαπανάται στην ταξινόμηση.

**Πίνακας 3.2:** Πραγματικά Δεδομένα: Total Time (sec)

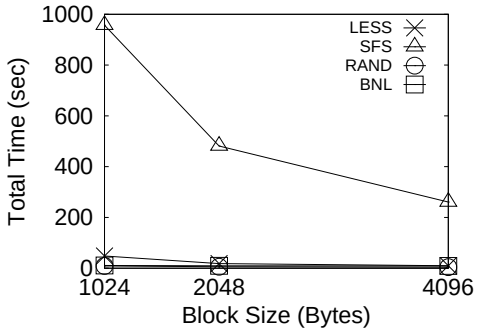
| **Dataset** | **LESS** | **SFS** | **RAND** | **BNL** |
|---|---|---|---|---|
| House | 30.11 | 178.21 | 15.25 | 4.98 |
| Colour | 14.43 | 90.73 | 3.70 | 1.28 |
| NBA | 9.45 | 26.68 | 0.71 | 0.41 |

### 3.5.3 Αξιολόγηση Πολιτικών

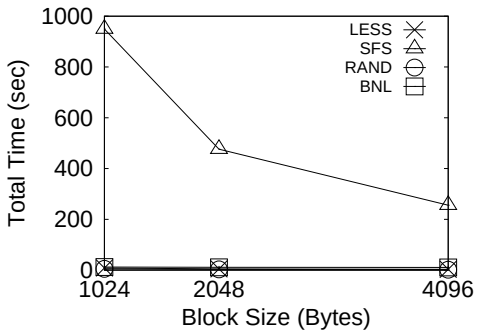
Σε αυτό το πείραμα, μελετάμε την επίδραση των διαφορετικών πολιτικών παραθύρου σε αλγορίθμους κορυφογραμμής που βασίζονται στην σάρωση. Συγκεκριμένα, χρησιμοποιούμε τους BNL και SFS αλγορίθμους και χρησιμοποιούμε αρκετές πολιτικές




(α′) Anti-correlated




(β′) Independent




(γ′) Correlated

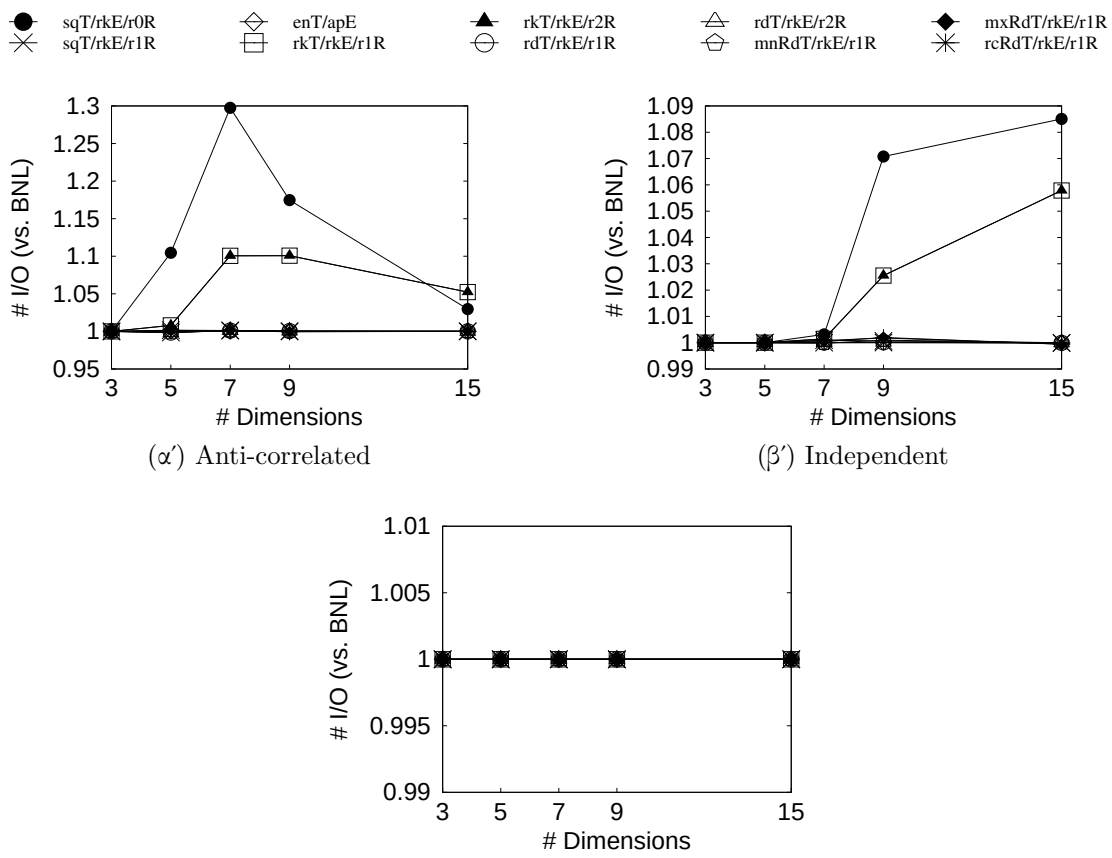
**Σχήμα 3.10:** Total Time: varying block size

διάσχισης και απομάκρυνσης, σε συνδυασμό με διαφορετικά σχήματα κατάταξης. Η επίδραση των πολιτικών στον LESS είναι παρόμοια με την επίδραση στον SFS και δεν παρουσιάζεται. Σχετικά με τον RAND, μόνο η πολιτική διάσχισης του παραθύρου επηρεάζει την επίδοση. Η επίδραση του δεν είναι δραματική και έτσι δεν παρουσιάζεται.

Όλα τα αποτελέσματα παρουσιάζονται με βάση τους αρχικούς αλγορίθμους. Δηλαδή, έστω $m$ η μέτρηση του αρχικού αλγορίθμου, και $m'$ η αντίστοιχη μέτρηση της εξεταζόμενης παραλλαγής. Σε αυτή τη περίπτωση, η μέτρηση που παρουσιάζεται για την παραλλαγή είναι $1 + (m' - m)/m$.

#### 3.5.3.1 BNL

Πρώτα μελετάμε την επίδοση του BNL με βάση τις 10 πιο σημαντικές πολιτικές και σχήματα ταξινόμησης. Το Σχήμα 3.11 δείχνει τις I/O λειτουργίες που πραγματοποιούνται από τις εκδόσεις του BNL. Όπως μπορούμε να δούμε, καμία από τις εξεταζόμενες παραλλαγές δεν έχει σημαντικά καλύτερη απόδοση από τον αρχικό αλγόριθμο. Σχεδόν σε όλες τις περιπτώσεις, η I/O απόδοση των περισσότερων παραλλαγών είναι πολύ κοντά στην αρχική. Ο λόγος είναι ότι η append eviction policy (apE), η οποία υιοθετείται από τον αρχικό BNL ήδη αποδίδει πολύ καλά για δύο λόγους. Πρώτα, η apE πολιτική πάντα αφαιρεί αντικείμενα που δεν έχουν κυριαρχήσει σε άλλα αντικείμενα. Με αυτόν τον τρόπο, η πολιτική αυτή έμμεσα αφαρμόζει κριτήρια σχετικά με την κυριαρχία (dominance-oriented criterion). Δεύτερον, η apE πολιτική πάντα αφαιρεί το πιο πρόσφατα διαβασμένο αντικείμενο, το οποίο είναι πολύ σημαντικό στην BNL. Ένα αντικείμενο που έχει μόλις διαβαστεί, απαιτεί περισσότερο χρόνο (σε σχέση με τα άλλα αντικείμενα του παραθύρου) προκειμένου να ταυτοποιηθεί ως κορυφογραμμή, και συνεπώς να μεταβιβαστεί στα αποτελέσματα και να απελευθερώσει μνήμη. Έτσι, κρα-




(α′) Anti-correlated

(β′) Independent

(γ′) Correlated

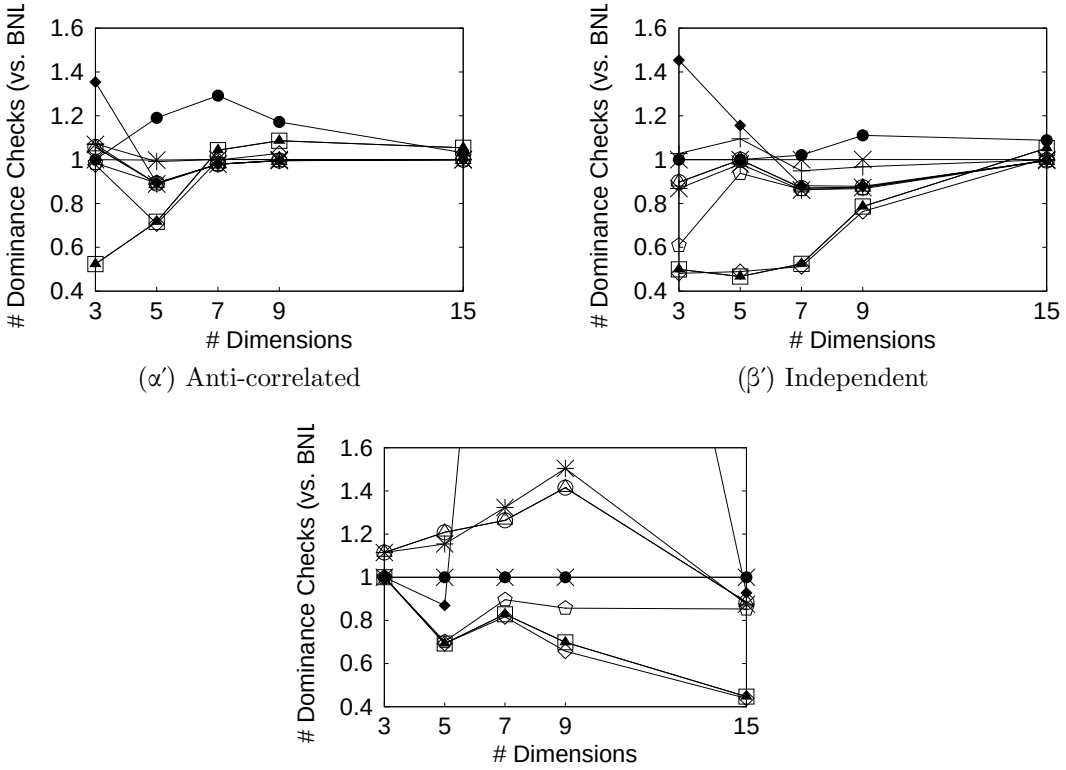
**Σχήμα 3.11:** BNL Policies (I/O Operations): μεταβάλλοντας τον αριθμό των διαστάσεων

τώντας τα "παλιότερα" αντικείμενα αυξάνουμε την πιθανότητα απελευθέρωσης μνήμης στο κοντινό μέλλον. Και ακόμα είναι πιθανό να μειώσουμε οριακά τον αριθμό των I/Os.

Το Σχήμα 3.12 δείχνει τον αριθμό των ελέγχων κυριαρχίας που πραγματοποιούνται. Μπορούμε να παρατηρήσουμε ότι, σε αρκετές περιπτώσεις, οι παραλλαγές που υιοθετούν τη πολιτική διάσχισης που βασίζεται στην κατάταξη, πραγματοποιούν σημαντικά λιγότερους ελέγχους κυριαρχίας από ότι ο αρχικός αλγόριθμος. Συγκεκριμένα, οι rkT/rkE/r1R και rkT/rkE/r2R παραλλαγές έχου καλύτερες επιδόσεις από τις άλλες σχεδόν σε όλες τις περιπτώσεις, σε independent και correlated σύνολα δεδομένων, μέχρι και κατά 50%. Παρόμοια αποτελέσματα ισχύουν για την περίπτωση λίγων διαστάσεων στα anti-correlated σύνολα δεδομένων. Όμως, αυτό δεν ισχύει για περισσότερες διαστάσεις, λόγω της μεγάλης αύξησης των αντικειμένων κορυφογραμμής στα anti-correlated σύνολα δεδομένων.

### 3.5.3.2 SFS

Εδώ, όπως στο προηγούμενο πείραμα, εξετάζουμε την επίδοση του SFS αλγορίθμου καθώς υιοθετεί διαφορετικές πολιτικές. Ομοίως με τον BNL, καμία παραλλαγή του SFS δε πραγματοποιεί σημαντικά λιγότερες I/O λειτουργίες (Σχήμα 3.13). Σχετικά με τους ελέγχους κυριαρχίας (Σχήμα 3.14), σε anti-correlated και independent σύνολα δεδομένων, οι περισσότερες παραλλαγές έχουν παρόμοια επίδοση με τον αρχικό αλγόριθμο. Μόνο για τα correlated σύνολα δεδομένων, οι πολιτικές που βασίζονται στην ταξινόμηση επιδεικνύουν σημαντικά κέρδη στην επίδοση.




(α′) Anti-correlated

(β′) Independent

(γ′) Correlated

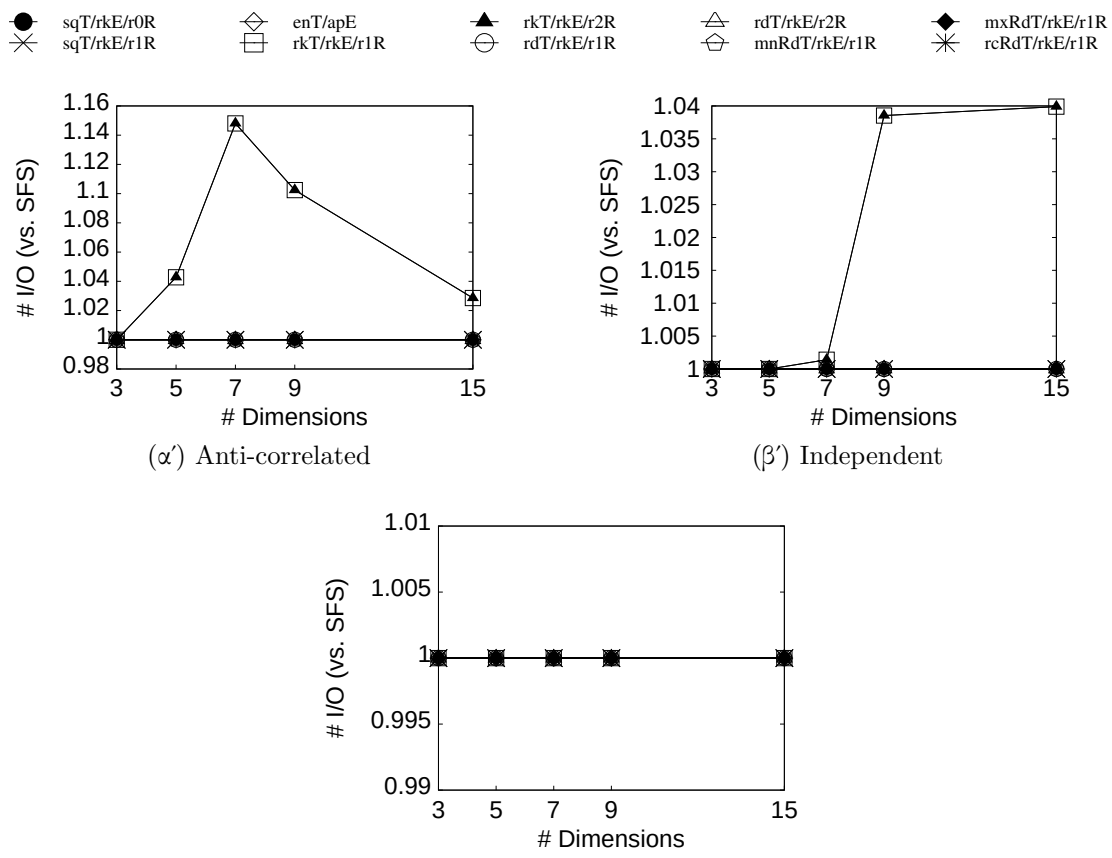
**Σχήμα 3.12:** BNL Policies (Dominance Checks): μεταβάλλοντας τον αριθμό των διαστάσεων

### 3.5.4 Αξιόλογη της Συνάρτησης Ταξινόμησης

Σε αυτό το πείραμα μελετάμε την επίδοση των SFS και των LESS αλγορίθμων χρησιμοποιώντας διαφορετικές συναρτήσεις ταξινόμησης. Χρησιμοποιούμε τρεις συναρτήσεις ταξινόμησης: την *Entr*, την *Sum* και την *minC*, όπως περιγράφονται στη Ενότητα 3.3. Όπως μπορούμε να δούμε στο Σχήμα 3.15, οι συναρτήσεις ταξινόμησης *Entr* και *Sum* έχουν παρόμοια επίδοση σε όλες τις περιπτώσεις, και για τους δύο αλγορίθμους SFS και LESS. Μια άλλη ενδιαφέρουσα παρατήρηση είναι ότι ο SFS, σε αντίθεση με τον LESS, έχει παρόμοια επίδοση σε όλες τις συναρτήσεις ταξινόμησης.

### 3.5.5 Σχολιασμός

Σε ένα I/O-sensitive περιβάλλον, δηλαδή όταν οι I/O λειτουργίες κοστίζουν σημαντικά περισσότερο από τα CPU cycles, ο BNL φαίνεται να είναι η ιδανική επιλογή, καθώς πραγματοποιεί λιγότερες I/O λειτουργίες από ότι οι άλλες μέθοδοι σχεδόν σε όλα τα περιβάλλοντα. Επιπλέον, οι BNL και RAND πραγματοποιούν λιγότερες λειτουργίες γραφής από ότι οι άλλες μέθοδοι. Από την άλλη, σε ένα CPU-sensitive περιβάλλον, οι LESS και RAND φαίνονται να είναι καλές επιλογές. Ο LESS πραγματοποιεί τους λιγότερους ελέγχους κυριαρχίας, ενώ ο RAND δεν δαπανά χρόνο για την ταξινόμηση δεδομένων, ούτε για την ταυτοποίηση κορυφογραμμών. Τέλος, σχετικά με τις πολιτικές που δοκιμάζονται, αυτές που βασίζονται στην κατάταξη παρουσιάζουν οφέλη αλλά μόνο σε CPU-sensitive περιβάλλοντα.




(α′) Anti-correlated

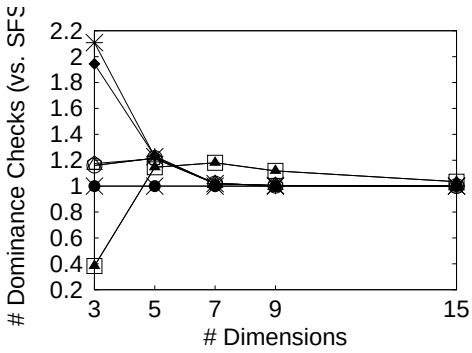
(β′) Independent

(γ′) Correlated

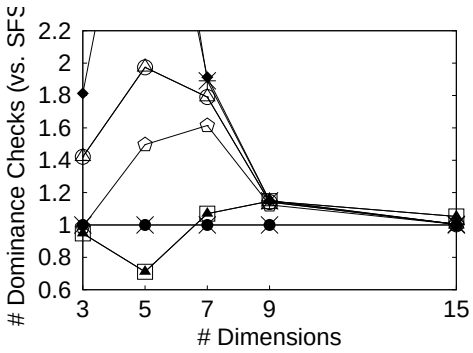
**Σχήμα 3.13:** SFS Policies (I/O Operations): μεταβάλλοντας τον αριθμό των διαστάσεων

## 3.6 Επίλογος

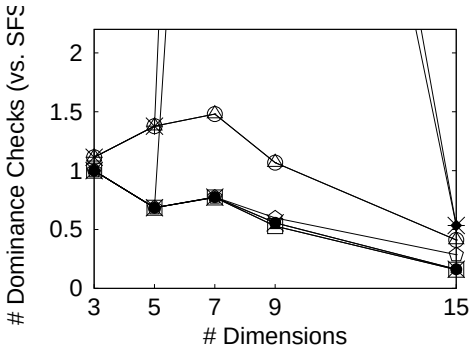
Σε αυτή την εργασία μελετήσαμε μια σημαντική κατηγορία αλγορίθμων κορυφογραμμών δευτερεύουσας μνήμης. Συγκεκριμένα, υποθέσαμε το καθιερωμένο μοντέλο δευτερεύουσας μνήμης και μελετήσαμε λεπτομέρειες υλοποίησης που προκύπτουν στις υλοποιήσεις πραγματικών συστημάτων. Επιπλέον, επικεντρωθήκαμε στα ζητήματα της διαχείρισης των in-memory αντικειμένων, και της συγκρότηση του παραθύρου. Τέλος, πραγματοποιήσαμε μια εκτεταμένη πειραματική αξιολόγηση σε πραγματικά και συνθετικά δεδομένα χρησιμοποιώντας πραγματικές υλοποιήσεις που βασίζονται στον δίσκο. Η εκτεταμένη μελέτη ανέδειξε νέα συμπεράσματα σχετικά με την σχεδίαση και την απόδοση των αλγορίθμων κορυφογραμμής.




(α′) Anti-correlated

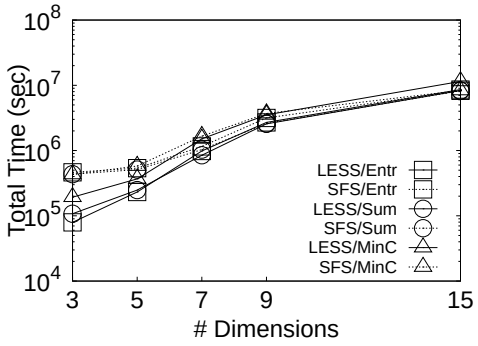



(β′) Independent

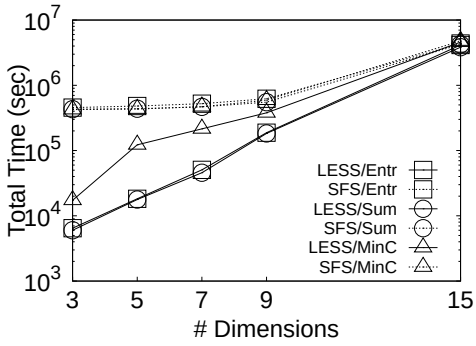



(γ′) Correlated

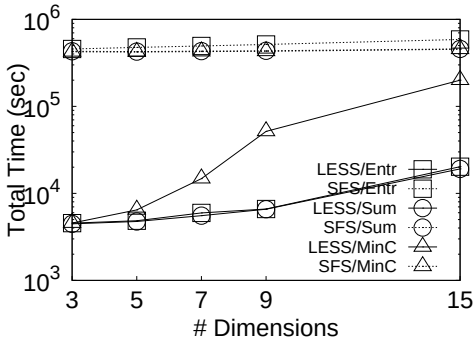
**Σχήμα 3.14:** SFS Policies (Dominance Checks): μεταβάλλοντας τον αριθμό των διαστάσεων




(α′) Anti-correlated




(β′) Independent




(γ′) Correlated

**Σχήμα 3.15:** Sorting Functions (Total Time): μεταβάλλοντας τον αριθμό των διαστάσεων

# Μέρος II

# Διερευνητική Ανάλυση Δεδομένων

# Κεφάλαιο 4

# Οπτική Διερεύνηση και Ανάλυση Μεγάλων Δεδομένων

Τα συστήματα οπτικοποίησης και διερεύνησης δεδομένων είναι μείζονος σημασίας στην εποχή των Μεγάλων Δεδομένων (Big Data), στην οποία ο όγκος και η ποικιλία των προσβάσιμων πληροφοριών, δυσκολεύουν τη χειροκίνητη διερεύνηση και ανάλυση δεδομένων από τους ανθρώπους. Τα περισσότερα παραδοσιακά συστήματα λειτουργούν offline, περιορισμένα έτσι στο να έχουν πρόσβαση σε προ-επεξεργασμένα (στατιστικά) δεδομένα. Ακόμα, περιορίζουν τις δυνατότητές τους καθώς διαχειρίζονται δεδομένα μικρού μεγέθους, τα οποία είναι εύκολα διαχειρίσιμα με συμβατικές τεχνικές. Από την άλλη, στην εποχή των Big Data είναι σύνηθες η ύπαρξη μεγάλων όγκων και ποικιλία πηγών δεδομένων, που είναι στη φύση τους κυρίως δυναμικές. Καθώς οι περισσότερες προσφέρουν API ή σημεία πρόσβασης (query endpoints) για online πρόσβαση, ή σε άλλες περιπτώσεις παραλαμβάνουν τα δεδομένα σε συνεχή ροή (streaming). Για αυτό το λόγο, τα σύγχρονα συστήματα πρέπει να ανταποκριθούν στη πρόκληση της άμεσης (on-the-fly) και κλιμακούμενης (scalable) οπτικοποίησης μεγάλων και δυναμικών συνόλων δεδομένων, προσφέροντας επαρκείς τεχνικές διερεύνησης, καθώς και μηχανισμούς αφαίρεσης και περίληψης δεδομένων. Τέλος, πρέπει να λαμβάνουν υπόψιν διαφορετικά σενάρια διερεύνησης ανάλογα με τις προτιμήσεις του χρήστη.

Σε αυτό το κεφάλαιο μελετάμε δύο προβλήματα. Το πρώτο πρόβλημα αφορά την άμεσης οπτικής διερεύνησης μεγάλων συνόλων δεδομένων. Για αυτό το πρόβλημα παρουσιάζουμε ένα γενικό μοντέλο για προσωποποιημένη διερεύνηση πολλών επιπέδων και ανάλυσης. Το μοντέλο μας αναπτύσσεται πάνω σε μια ελαφριά δεντροειδή δομή δεδομένων (lightweight tree-based structure) η οποία μπορεί να κατασκευαστεί αποδοτικά και άμεσα για δεδομένο σύνολο δεδομένων. Αυτή η δομή ομαδοποιεί (aggregate) τα αντικείμενα εισόδου σε ένα ιεραρχικό πολυεπίπεδο μοντέλο. Υπολογίζοντας διαφορετικά σενάρια διερεύνησης μεγάλων συνόλων δεδομένων, το προτεινόμενο μοντέλο επιτρέπει την αποδοτική πολυεπίπεδη διερεύνηση, προσφέροντας σταδιακή κατασκευή (incremental construction) καθώς και prefetching, βασιζόμενα στην αλληλεπίδραση με τον χρήστη, καθώς και δυναμική προσαρμογή της ιεραρχία βάση των προτιμήσεων του χρήστη. Μια λεπτομερή θεωρητική ανάλυση παρουσιάζεται, απεικονίζοντας την αποδοτικότητα της προτεινόμενης μεθόδου. Το παραπάνω μοντέλο πραγματώνεται σε ένα πρωτότυπο web-based εργαλείο, που ονομάζεται SynopsViz και προσφέρει πολυεπίπεδη οπτική διερεύνηση και ανάλυση διασυνδεδεμένων δεδομένων. Τέλος, πραγματοποιούμε αξιολόγηση της απόδοσης καθώς και μελέτη με πραγματικούς χρήστες.

Το δεύτερο πρόβλημα αφορά τη διερεύνηση και οπτικοποίηση πολύ μεγάλων γράφων.

Για αυτό το πρόβλημα παρουσιάζουμε μια καινοτόμα πλατφόρμα η οποία επιτρέπει στον χρήστη να αλληλεπιδρά με το οπτικοποιημένο γράφο με τρόπο που μοιάζει με τη διερεύνηση γεωγραφικών χαρτών σε πολλαπλά επίπεδα. Η προσέγγιση μας περιλαμβάνει μία φάση προεπεξεργασίας των δεδομένων κατά την διάρκεια της οποίας υπολογίζεται η διάταξη του γράφου με την απόδοση συντεταγμένων στους κόμβους του με βάση έναν Ευκλείδειο χώρο. Τα σημεία που αντιστοιχούν στους κόμβους εισάγονται σε μία χωρική δομή (R-tree) και αποθηκεύονται στη βάση δεδομένων. Πολλαπλά αφαιρετικά επίπεδα του γράφου, βασισμένα σε διαφορετικά κριτήρια, δημιουργούνται επίσης στην φάση της προεπεξεργασίας και αποθηκεύονται με τον ίδιο τρόπο όπως ο βασικός γράφος, επιτρέποντας στον χρήστη να εξερευνά τα δεδομένα στα διαφορετικά αφαιρετικά επίπεδα ανάλογα με τις ανάγκες του. Στην συνέχεια το σύστημα μας αντιστοιχίζει τις ενέργειες του χρήστη σε χωρικά ερωτήματα (window queries) που μπορούν να αποτιμηθούν αποδοτικά από την υποδομή που δημιουργήσαμε στην φάση της προεπεξεργασίας. Με αυτήν την τεχνική έχουμε τόσο στοχευμένη πρόσβαση σε πάρα πολύ μεγάλους γράφους χωρίς διαστήματα αδράνειας, όσο και χαμηλές απαιτήσεις σε μνήμη. Το on-line εργαλείο μας υποστηρίζει τέσσερις βασικές λειτουργίες: (1) διαδραστική περιήγηση, (2) διερεύνηση πολλαπλών επιπέδων, (3) αναζήτηση με λέξεις-κλειδιά, και (4) επιλογή και διαχείριση υπογράφων.

## 4.1 Αποδοτική Πολυεπίπεδη Διερεύνηση

Η διερεύνηση, η οπτικοποίηση και η ανάλυση δεδομένων αποτελεί βασική εργασία των επιστημόνων δεδομένων (data scientists) και των αναλυτών σε πολλές εφαρμογές. Η διερεύνηση και οπτικοποίηση δεδομένων επιτρέπει στους χρήστες να ανακαλύπτουν ενδιαφέροντα μοτίβα, να εξάγουν συσχετισμούς και σχέσεις αιτιατότητας, καθώς επίσης υποστηρίζει και sense-making δράσεις πάνω στα δεδομένα που δεν είναι πάντα δυνατές με παραδοσιακές τεχνικές εξόρυξης δεδομένων [189, 122]. Τα παραπάνω είναι μείζονος σημασίας στην εποχή των Μεγάλων Δεδομένων (Big Data), στην οποία ο όγκος και η ποικιλομορφία των προσβάσιμων πληροφοριών δυσκολεύουν τη χειροκίνητη διερεύνηση και ανάλυση μεγάλων συνόλων δεδομένων από τους ανθρώπους.

Μία από τις μεγάλες προκλήσεις της οπτικής διερεύνησης σχετίζεται με το *μεγάλο μέγεθος* που χαρακτηρίζει πολλά σύνολα δεδομένων στις μέρες μας. Αναλογιζόμενοι το ρητό: *"overview first, zoom and filter, then details on demand"* [302], η απόκτηση επισκόπησης overview είναι βασικό καθήκον στη διαδικασία της οπτικής διερεύνησης. Όμως, η επισκόπηση σε ένα μεγάλο σύνολο δεδομένων είναι δύσκολο να επιτευχθεί. Η *υπερφόρτωση πληροφοριών* (information overloading, overplotting) είναι συχνό πρόβλημα για αυτό και η βασική απαίτηση στις προτεινόμενες προσεγγίσεις είναι η αφαίρεση και η περίληψη πληροφοριών.

Μια δεύτερη πρόκληση σχετίζεται με τη διαθεσιμότητα των API και των σημείων πρόσβασης query endpoints (π.χ. SPARQL) για online πρόσβαση στα δεδομένα, καθώς και σε περιπτώσεις στις οποίες τα δεδομένα μεταδίδονται σε συνεχή ροή (streaming). Τα παραπάνω επιβάλουν την διαχείρισης μεγάλων συνόλων δεδομένων σε δυναμικό περιβάλλον (dynamic setting), και συνεπώς το στάδιο της προεπεξεργασίας (preprocessing phase) δεν είναι εφικτό, όπως η παραδοσιακή ευρετηρίαση (indexing). Συνεπώς, οι σύγχρονες τεχνικές οφείλουν να παρέχουν κλιμακωσιμότητα (scalability) και αποδοτική επεξεργασία (efficient processing) για άμεση ανάλυση και οπτικοποίηση δυναμικών συνόλων δεδομένων.

Τέλος, η απαίτηση για άμεση οπτικοποίηση πρέπει να συνδυαστεί με την ποικιλία

των προτιμήσεων και απαιτήσεων που προκύπτουν από τους διαφορετικούς χρήστες και τα διαφορετικά σενάρια. Συνεπώς, οι προτεινόμενες προσεγγίσεις πρέπει να παρέχουν στον χρήστη την ικανότητα να προσαρμόζει την διερεύνηση, επιτρέποντας να οργανώνει τα δεδομένα με διαφορετικούς τρόπους ανάλογα με το τύπο της πληροφορίας ή τις λεπτομέρειες που θέλουν να διερευνήσουν.

Λαμβάνοντας υπ' όψιν το γενικό πρόβλημα της διερεύνησης μεγάλων δεδομένων [303, 178, 263, 346, 166], οι περισσότερες προσεγγίσεις στοχεύουν στην ικανότητα κατάλληλης περίληψης και αφαίρεσης μεγάλων συνόλων δεδομένων. Από αυτήν την άποψη, μεγάλος αριθμός συστημάτων υιοθετούν *τεχνικές προσεγγίσεων* (approximation techniques), δηλαδή τεχνικές αφαίρεσης δεδομένων (data reduction techniques) στις οποίες υπολογίζονται μερικά αποτελέσματα. Οι υφιστάμενες προσεγγίσεις βασίζονται κυρίως: (1) στη δειγματοληψία (sampling) και στο φιλτράρισμα (filtering) [155, 271, 212, 20, 190, 46] και/ή στη (2) συνάθροιση (aggregation) (π.χ. binning, clustering) [144, 198, 197, 167, 239, 344, 45, 236, 18, 195]. Ομοίως, μερικά σύγχρονα συστήματα βάσεων δεδομένων (database-oriented systems) υιοθετούν τεχνικές προσεγγίσεις χρησιμοποιώντας προσεγγίσεις ερωτήσεων (π.χ.query translation, query rewriting) [46, 198, 197, 337, 346]. Πρόσφατα, σταδιακές τεχνικές προσεγγίσεων (incremental approximation techniques) έχουν υιοθετηθεί: σε αυτές τις μεθόδους υπολογίζονται προσεγγιστικές απαντήσεις σε προοδευτικά μεγαλύτερα σύνολα δεδομένων [155, 20, 190]. Σε διαφορετικό πλαίσιο, μια προσαρμοζόμενη ευρετηριακή προσέγγιση (adaptive indexing), χρησιμοποιείται στην [357], όπου οι δείκτες δημιουργούνται σταδιακά και προσαρμοσμένα σε όλη τη διάρκεια της διερεύνησης. Ακόμα, για να βελτιώσουν την απόδοση, πολλά συστήματα χρησιμοποιούν τεχνικές καταχώρησης και ανάκτησης (caching and prefetching techniques).[319, 200, 194, 45, 98, 210, 133]. Τέλος, σε άλλες προσεγγίσεις, χρησιμοποιούνται παράλληλες αρχιτεκτονικές (parallel architectures) [142, 202, 201, 190].

Για να αντιμετωπίσουμε τις παραπάνω προκλήσεις, σε αυτή την εργασία, παρουσιάζουμε ένα νέο μοντέλο που συνδυάζει προσωποποιημένη, πολυεπίπεδη διερεύνηση (personalized multilevel exploration) με online ανάλυση αριθμητικών και χρονικών δεδομένων (numeric and temporal data). Στον πυρήνα έχουμε ένα ελαφρύ ιεραρχικό μοντέλο συνάθροισης (lightweight hierarchical aggregation model), που κατασκευάζεται άμεσα (on-the-fly) για δεδομένο σύνολο δεδομένων. Το προτεινόμενο μοντέλο είναι μια δεντροειδής δομή που αθροίζει (aggregate) τα δεδομένα σε πολλαπλά ιεραρχικά επίπεδα σχετιζόμενων ομάδων. Το μοντέλο μας επίσης εμπλουτίζει τις ομάδες (δηλαδή τις συναθροίσεις, περιλήψεις) με στατιστικά στοιχεία σχετικά με το περιεχόμενό τους, προσφέροντας πλουσιότερη εποπτεία πάνω στα δεδομένα. Ένα επιπλέον χαρακτηριστικό είναι ότι επιτρέπει στους χρήστες να οργανώνουν τη διερεύνηση με διαφορετικούς τρόπους, μέσω της παραμετροποίησης του αριθμού των ομάδων, του εύρους και του πλήθους των στοιχείων τους, του αριθμού των ιεραρχικών επιπέδων, κτλ. Ακόμα, παρουσιάζουμε τρία σενάρια διερεύνησης και παρουσιάζουμε δύο μεθόδους για αποδοτική διερεύνηση μεγάλων συνόλων δεδομένων: η πρώτη επιτυγχάνει τη σταδιακή δημιουργία του μοντέλου βασιζόμενη στην αλληλεπίδραση με τον χρήστη, ενώ η δεύτερη επιτρέπει τη δυναμική και αποδοτική προσαρμογή του μοντέλου στις προτιμήσεις του χρήστη. Η αποδοτικότητα του προτεινόμενου μοντέλου παρουσιάζεται μέσω μιας λεπτομερούς θεωρητικής ανάλυσης, καθώς και από μία πειραματική αξιολόγηση. Τέλος, το προτεινόμενο μοντέλο έχει υλοποιηθεί σε ένα web-based εργαλείο, το *Σψνοπσΐζ*, το οποίο προσφέρει ποικιλία τεχνικών οπτικοποίησης (π.χ. διαγράμματα) για πολυεπίπεδη οπτική διερεύνηση και ανάλυση σε Διασυνδεδεμένα Δεδομένα (Linked Data datasets).

**Συνεισφορά.** Οι κύριες συνεισφορές της παρούσας εργασίας είναι οι εξής.

1. Εισάγουμε ένα γενικό μοντέλο για την οργάνωση, τη διερεύνηση και την ανάλυση αριθμητικών και χρονικών δεδομένων με πολυεπίπεδο τρόπο.
2. Υλοποιούμε το μοντέλο ως μια ελαφριά δεντροειδή δομή κύριας μνήμης (lightweight, main memory tree-based structure), η οποία μπορεί να κατασκευαστεί αποδοτικά και άμεσα.
3. Προτείνουμε δύο εκδόσεις δεντροειδών δομών, οι οποίες μπορούν να υιοθετούν διαφορετικές προσεγγίσεις για την οργάνωση δεδομένων.
4. Περιγράφουμε μια απλή μέθοδο εκτίμησης των παραμέτρων για την κατασκευή του δέντρου, όταν δεν είναι διαθέσιμες προτιμήσεις του χρήστη.
5. Ορίζουμε διαφορετικά σενάρια διερεύνησης ανάλογα με τις προτιμήσεις του χρήστη.
6. Εισάγουμε μια μέθοδο που σταδιακά δημιουργεί ένα ιεραρχικό δέντρο μέσω της αλληλεπίδρασης με τον χρήστη.
7. Προτείνουμε μια αποδοτική μέθοδο που δυναμικά προσαρμόζει την υπάρχουσα ιεραρχία σε μια νέα, ανάλογα με τις προτιμήσεις του χρήστη.
8. Παρουσιάζουμε μια λεπτομερή θεωρητική ανάλυση, που αναδεικνύει την αποδοτικότητα του προτεινόμενου μοντέλου.
9. Αναπτύσσουμε ένα πρωτότυπο σύστημα το οποίο εφαρμόζει το παρόν μοντέλο, προσφέροντας μία πολυεπίπεδη οπτική διερεύνηση στα Διασυνδεδεμένα Δεδομένα (Linked Data)
10. Πραγματοποιούμε μία λεπτομερή πειραματική αξιολόγηση και μία μελέτη με χρήστες, χρησιμοποιώντας το σύνολο δεδομένων DBpedia 2014.

### 4.1.1 Το Μοντέλο HETree

Σε αυτό το τμήμα της εργασίας παρουσιάζουμε το HETree (**H**ierarchical **E**xploration **Tree** - Δέντρο Ιεραρχικής Διερεύνησης), ένα γενικό μοντέλο για την οργάνωση, τη διερεύνηση, και την ανάλυση αριθμητικών και χρονικών δεδομένων με πολυεπίπεδο τρόπο. Συγκεκριμένα, το HETree ορίζεται στο πλαίσιο της πολυεπίπεδης οπτικής διερεύνησης και ανάλυσης. Το προτεινόμενο μοντέλο οργανώνει ιεραρχικά αριθμητικά και χρονικά δεδομένα, χωρίς να απαιτεί αυτό να περιγράφεται μέσω ενός ιεραρχικού σχήματος.

Η βασική ιδέα του μοντέλου μας είναι η ιεραρχική ομαδοποίηση συνόλων δεδομένων βασισμένοι στις τιμές μίας από τις ιδιότητές τους. Τα αντικείμενα εισόδου αποθηκεύονται στα φύλλα, ενώ οι εσωτερικοί κόμβοι αθροίζουν τους θυγατρικούς κόμβους τους. Η ρίζα του δέντρου αντιπροσωπεύει ολόκληρο το σύνολο δεδομένων. Η κυρια ιδέα του μοντέλου μας προσεγγίζει μια απλουστευμένη εκδοχή του στατικού (static) 1D R-Tree [170].

Σχετικά με την οπτική αναπαράσταση του μοντέλου και της διερεύνησης δεδομένων, τόσο τα αντικείμενα συνόλων δεδομένων ( data objects sets) δηλαδή τα περιεχόμενα των κόμβων-φύλλων, όσο και οι οντότητες που αντιπροσωπεύουν ομάδες αντικειμένων (φύλλα ή εσωτερικοί κόμβοι) αναπαριστώνται οπτικά, επιτρέποντας στον χρήστη να διερευνάει τα δεδομένα με ιεραρχικό τρόπο. Σημειώνουμε ότι η δεντροειδής δομή οργανώνει τα δεδομένα με ιεραρχικό τρόπο, χώρις να θέτει περιορισμούς στον τρόπο που

ο χρήστης αλληλεπιδρά με αυτήν την ιεραρχία. Έτσι, καθίσταται δυνατό να υιοθετηθούν διαφορετικές στρατηγικές σχετικά τη διάσχιση, καθώς και με τους κόμβους του δέντρου που οριστικοποιούμε σε κάθε στάδιο.

#### 4.1.1.1 Εισαγωγικά

Σε αυτήν την εργασία μοντελοποιήσαμε τα αντικείμενα δεδομένων ως RDF triples. Όμως οι παρουσιαζόμενοι μέθοδοι είναι γενικοί και μπορούν να εφαρμοστούν σε οποιοδήποτε σύνολο δεδομένων με αριθμητικά και χρονικά χαρακτηριστικά. Έτσι, οι όροι triple (τριπλέτα) και data object θα χρησιμοποιούνται εξίσου.

Θεωρούμε ένα *RDF dataset* $R$ που αποτελείται από ένα σύνολο από *RDF triples*. Ως *δεδομένα εισόδου*, θεωρούμε ένα σύνολο από RDF triples $D$, όπου $D \subseteq R$ και οι τριπλέτες στο $D$ έχουν ως αντικείμενα είτε αριθμητικές (π.χ., ακέραιες, δεκαδικές) είτε χρονικές τιμές (π.χ., ημερομηνία, ώρα). Έστω $tr$ μία RDF triple, $tr.s$, $tr.p$ και $tr.o$ αντιπροσωπεύουν αντίστοιχα, ένα *υποκείμενο-subject*, *κατηγόρημα-predicate* και *αντικείμενο-object* της RDF triple $tr$.

Με δεδομένα εισόδου $D$, $S$ είναι ένα *διατεταγμένο σύνολο* (ordered set) από RDF triples, παραγόμενα από $D$, όπου οι τριπλέτες ταξινομούνται με βάση τις τιμές του αντικειμένου, με αύξουσα διάταξη. Υποθέτουμε ότι το $S[i]$ υποδηλώνει την $i$-η τριπλέτα, με $S[1]$ να είναι η πρώτη τριπλέτα. Τότε, για κάθε $i < j$, έχουμε ότι $S[i].o \leq S[j].o$. Επίσης, $D = S$, δηλαδή, για κάθε $tr, tr \in D$ αν $tr \in S$.

Το Σχήμα 4.1 παρουσιάζει ένα σύνολο από 10 RDF triples, αντιπροσωπεύοντας άτομα και την ηλικία τους. Στο Σχήμα 4.1, υποθέτουμε ότι τα αντικείμενα $p0$-$p9$ είναι στιγμιότυπα της κλάσης *Person* και το χαρακτηριστικό *age* είναι datatype property με εύρος ακεραίων (integer range).

| | |
|---|---|
| $p0$ *age* 35 | $p5$ *age* 35 |
| $p1$ *age* 100 | $p6$ *age* 45 |
| $p2$ *age* 55 | $p7$ *age* 80 |
| $p3$ *age* 37 | $p8$ *age* 20 |
| $p4$ *age* 30 | $p9$ *age* 50 |

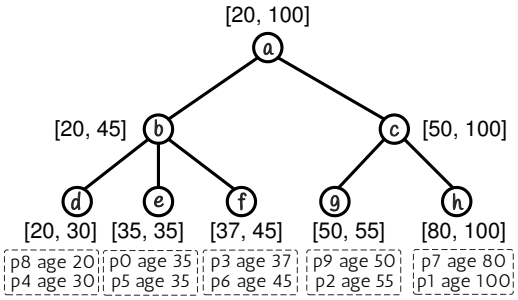
**Σχήμα 4.1:** Παράδειγμα δεδομένα εισόδου

Υποθέτουμε ένα *διάστημα* (interval) $I = [a, b]$, όπου $a, b \in \mathbb{R}$· τότε, $I = \{k \in \mathbb{R} \mid a \leq k \leq b\}$. Ομοίως, για $I = [a, b)$, έχουμε ότι $I = \{k \in \mathbb{R} \mid a \leq k < b\}$. Έστω $I^-$ και $I^+$ ο συμβολισμός του κατώτερου και ανώτερου ορίου του διαστήματος $I$, αντίστοιχα. Δηλαδή, αν $I = [a, b]$, τότε $I^- = a$ και $I^+ = b$. Το *μήκος* του διαστήματος $I$ ορίζεται ως $|I^+ - I^-|$.

#### 4.1.1.2 Η Δομή HETree

Σε αυτή την παράγραφο, παρουσιάζουμε πιο λεπτομερώς τη δομή *HETree*. Η HETree οργανώνει ιεραρχικά αριθμητικά και χρονικά[1] δεδομένα σε ομάδες ενώ διαστήματα χρησιμοποιούνται για να αντιπροσωπεύουν αυτές τις ομάδες. Η HETree ορίζεται από το

[1]Σημειώνουμε ότι η δομή μας χειρίζεται αριθμητικά και χρονικά δεδομένα με παρόμοιο τρόπο. Επίσης, άλλοι τύποι μονοδιάστατων δεδομένων μπορούν να υποστηριχτούν, με την προϋπόθεση ότι μια συνολική διάταξη μπορεί να οριστεί στα δεδομένα.




**Σχήμα 4.2:** Content-based HETree (HETree-C)

βαθμό του δέντρου και από τον αριθμό των κόμβων-φύλλων[2]. Ουσιαστικά,ο αριθμός των κόμβων-φύλλων αντιστοιχεί στον αριθμό των ομάδων στις οποίες τα αντικείμενα δεδομένων εισόδων ταξινομούνται. Ο βαθμός του δέντρου αντιστοιχεί στον (μέγιστο) αριθμό ομάδων, όπου μια ομάδα χωρίζεται στο κατώτερο επίπεδο.

Με δεδομένο ένα σύνολο από αντικείμενα δεδομένων (RDF triples) $D$, έναν θετικό ακέραιο $\ell$ που δηλώνει τον αριθμό των φύλλων, ένα θετικό ακέραιο $d$ που δηλώνει τον βαθμό του δέντρου, ένα *HETree* $(D, \ell, d)$ είναι ένα *ορδερεδ δ-αρψ τρεε*, με τις ακόλουθες βασικές ιδιότητες.

- Το δέντρο έχει ακριβώς $\ell$ αριθμό φύλλων.
- Όλα τα φύλλα βρίσκονται στο ίδιο επίπεδο.
- Κάθε φύλλο περιέχει ένα σύνολο από αντικείμενα δεδομένων, που είναι ταξινομημένα σε αύξουσα σειρά με βάση τις τιμές τους. Με δεδομένο ένα φύλλο $n$, τα $n.data$ δηλώνουν τα αντικείμενα δεδομένων που περιέχονται στο $n$.
- Κάθε εσωτερικός κόμβος έχει το πολύ $d$ θυγατρικούς κόμβους. Έστω $n$ ένας εσωτερικός κόμβος, $n.c_i$ υποδηλώνει το $i$-ο παιδί για τον κόμβο $n$, με $n.c_1$ να είναι το αριστερότερο παιδί.
- Κάθε κόμβος αντιστοιχεί σε ένα διάστημα. Με δεδομένο έναν κόμβο $n$, το $n.I$ δηλώνει το διάστημα για τον κόμβο $n$.
- Σε κάθε επίπεδο, όλοι οι κόμβοι ταξινομούνται με βάση τα κατώτερα όρια των διαστημάτων τους. Δηλαδή, αν $n$ ένας εσωτερικός κόμβος, για κάθε $i < j$, έχουμε ότι $n.c_i.I^- \leq n.c_j.I^-$.
- Για κάθε φύλλο, το διάστημα του οριοθετείται από τις τιμές των αντικειμένων που περιέχονται στο φύλλο. Έστω $n$ ο αριστερότερος κόμβος, υποθέτουμε ακόμα ότι $n$ περιέχει $x$ αντικείμενα από $D$. Τότε, έχουμε ότι $n.I^- = S[1].o$ και $n.I^+ = S[x].o$, όπου $S$ είναι το διατεταγμένο σύνολο αντικείμενο που προκύπτει από το $D$.
- Για κάθε εσωτερικό κόμβο, το διάστημα του οριοθετείται από την ένωση των διαστημάτων των παιδιών του. Δηλαδή, αν $n$ είναι ένας εσωτερικός κόμβος, με $k$ παιδιά, τότε, έχουμε $n.I^- = n.c_1.I^-$ και $n.I^+ = n.c_k.I^+$.

Ακόμα, παρουσιάζουμε δύο διαφορετικές προσεγγίσεις για την οργάνωση του HETree. Στη πρώτη προσέγγιση, ταξινομούμε τα αντικείμενα δεδομένων σε ομάδες, όπου οι τιμές των αντικειμένων κάθε ομάδας καλύπτει ίδιο εύρος τιμών. Στη δεύτερη προσέγγιση,

[2]Σημειώνουμε ότι ακολουθώντας παρόμοια προσέγγιση, η HETree μπορεί επίσης να οριστεί μέσω του ύψους του δέντρου αντί για το βαθμό του και τον αριθμό των φύλλων του.

**Algorithm 6.** createHETree-C/R ($D$, $\ell$, $d$)

**Input**: $D$: set of objects; $\ell$: number of leaf nodes; $d$: tree degree
**Output**: $r$: root node of the HETree tree

1 $S \leftarrow$ sort $D$ based on objects values
2 $L \leftarrow$ constrLeaves-C/R($S, \ell$)
3 $r \leftarrow$ constrtInterlNodes($L, d$)
4 **return** $r$

ταξινομούμε αντικείμενα σε ομάδες, όπου κάθε ομάδα περιέχει ίσο αριθμό αντικειμένων. Στα παρακάτω τμήματα, παρουσιάζουμε με λεπτομέρεια τις δύο προσεγγίσεις για την οργάνωση δεδομένων στο HETree.

#### 4.1.1.3 Content-based HETree (HETree-C)

Σε αυτή την παράγραφο παρουσιάζουμε μια εκδοσή του HETree, που ονομάζεται HETree-C (*Content-based* HETree). Αυτή η έκδοση του HETree ταξινομεί δεδομένα σε ομάδες ίδιου μεγέθους. Η βασική ιδιότητα του HETree-C είναι ότι κάθε φύλλο περιέχει περίπου τον ίδιο αριθμό αντικειμένων και ότι το περιεχόμενο του (δηλαδή τα αντικείμενα) κάθε φύλλου προσδιορίζουν το διάστημα του. Για τη δημιουργία του δέντρου, τα αντικείμενα πρώτα αναθέτονται στα φύλλα και μετά ορίζονται τα διαστήματα.

Ένα *HETree-C* $(D, \ell, d)$ είναι ένα HETree, με την εξής επιπρόσθετη ιδιότητα. Κάθε φύλλο περιέχει $\lambda$ ή $\lambda-1$ αντικείμενα, όπου[3] $\lambda = \left\lceil \frac{|D|}{\ell} \right\rceil$. Συγκεκριμένα, τα $\ell - (\lambda \cdot \ell - |D|)$ αριστερότερα φύλλα περιέχουν $\lambda$ αντικείμενα, ενώ τα υπόλοιπα φύλλα[4] περιέχουν $\lambda - 1$. Μπορούμε ισοδύναμα να ορίσουμε το HETree-C μέσω του αριθμού των αντικειμένων ανά φύλλο $\lambda$, αντί του αριθμού των φύλλων $\ell$.

##### 4.1.1.3.1 Η Κατασκευή του HETree-C

Το HETree-C κατασκευάζεται από κάτω προς τα πάνω. Ο Αλγόριθμος 6 περιγράφει τη δημιουργία του HETree-C. Ο αλγόριθμος παίρνει ως εισόδους: (1) ένα σύνολο από αντικείμενα δεδομένων $D$, (2) τον αριθμό των φύλλων $\ell$, και (3) το βαθμό του δέντρου $d$. Αρχικά, ο αλγόριθμος ταξινομεί το σύνολο αντικειμένων $D$ σε αύξουσα σειρά, με βάση τις τιμές των αντικειμένων (*line 1*). Τότε, ο αλγόριθμος χρησιμοποιεί δύο διαδικασίες για να δημιουργήσει τους κόμβους του δέντρου. Η πρώτη διαδικασία, με το όνομα constrLeaves-C δημιουργεί τα φύλλα του δέντρου (*line 2*). Η δεύτερη διαδικασία, με το όνομα constrtInterNodes, δημιουργεί τους εσωτερικούς κόμβους του δέντρου. (*line 3*). Τέλος, η ρίζα του δέντρου επιστρέφεται (*line 4*).

#### 4.1.1.4 Range-based HETree (HETree-R)

Η δεύτερη έκδοση του HETree ονομάζεται HETree-R (*Range-based* HETree). Το HETree-R οργανώνει τα δεδομένα σε ομάδες ίσου εύρους. Η βασική ιδιότητα του HETree-R είναι ότι κάθε φύλλο καλύπτει ίσο εύρος τιμών. Έτσι, στο HETree-R, ο χώρος των δεδομένων (data space) που ορίζεται από τις τιμές των αντικειμένων είναι

[3] Υποθέτουμε ότι, ο αριθμός των αντικειμένων είναι τουλάχιστον ίσος με τον αριθμό των φύλλων δηλαδή, $|D| \geq \ell$.

[4] Ως εναλλακτική μπορούμε να δημιουργήσουμε το HETree-C, έτσι ώστε κάθε φύλλο να περιέχει $\lambda$ αντικείμενα, εκτός από τα δεξιότερα φύλλα που μπορούν να περιέχουν από 1 ως $\lambda$ αντικείμενα.

**Procedure 1:** constrLeaves-C($S$, $\ell$)

**Input**: $S$: ordered set of objects; $\ell$: number of leaf nodes
**Output**: $L$: ordered set of leaf nodes

$\lambda \leftarrow \left\lceil \frac{|S|}{\ell} \right\rceil$
$k \leftarrow \ell - (\lambda \cdot \ell - |S|)$
$beg \leftarrow 1$
**for** $i \leftarrow 1$ **to** $\ell$ **do**
    create an empty leaf node $n$
    **if** $i \leq k$ **then**
        $num \leftarrow \lambda$
    **else**
        $num \leftarrow \lambda - 1$
    $end \leftarrow beg + num$
    **for** $t \leftarrow beg$ **to** $end$ **do**
        $n.data \leftarrow S[t]$
    $n.I^- \leftarrow S[beg].o$
    $n.I^+ \leftarrow S[end].o$
    $L[i] \leftarrow n$
    $beg \leftarrow end + 1$
**return** $L$

**Procedure 2:** constrtInterNodes($H$, $d$)

**Input**: $H$: ordered set of nodes; $d$: tree degree
**Output**: $r$: root node for $H$
**Variables**: $P$: ordered set of $H$'s parent nodes

$p_{num} \leftarrow \left\lceil \frac{|H|}{d} \right\rceil$ //number of parents nodes
$t \leftarrow d - (p_n \cdot d - |H|)$ //last parent's number of children
$c_{beg} \leftarrow 1$ //first child node
**for** $p \leftarrow 1$ **to** $p_{num}$ **do**
    create an empty internal node $n$
    **if** $p = p_{num}$ **then**
        $c_{num} \leftarrow t$ //number of children
    **else**
        $c_{num} \leftarrow d$
    $c_{end} \leftarrow c_{beg} + c_{num}$ //last child node
    **for** $j \leftarrow c_{beg}$ **to** $c_{end}$ **do**
        $n.c[j] \leftarrow H[j]$
    $n.I^- \leftarrow H[c_{beg}].I^-$
    $n.I^+ \leftarrow H[c_{end}].I^+$
    $P[p] \leftarrow n$
    $c_{beg} \leftarrow c_{end} + 1$
**if** $p_{num} = 1$ **then**
    $r \leftarrow P$
    **return** $r$
**else**
    **return** constrtInterlNodes($P, d$)

ίσα μοιρασμένος ανάμεσα στα φύλλα. Σε αντίθεση με το HETree-C, στο HETree-R, το εσωτερικό ενός φύλλου προσδιορίζει το περιεχόμενό του. Έτσι, για τη δημιουργία του HETree-R, πρώτα ορίζεται το εσωτερικό όλων των φύλλων και μετά εισάγονται τα αντικείμενα.

Ένα *HETree-R* $(D, \ell, d)$ είναι ένα HETree, με την εξής επιπρόσθετη ιδιότητα. Το εσωτερικό κάθε φύλλου έχει το ίδιο μήκος, δηλαδή καλύπτει ίσο εύρος τιμών. Έστω $S$ ένα σύνολο sorted RDF που προέρχεται από $D$, ενώ για κάθε φύλλο, το περιεχόμενο του έχει μήκος $\rho$, όπου[5] $\rho = \frac{|S[1].o - S[|S|].o|}{\ell}$. Έτσι, για ένα φύλλο $n$, έχουμε ότι $|n.I^- - n.I^+| = \rho$. Για παράδειγμα, για το αριστερότερο φύλλο, το εσωτερικό του είναι $[S[1].o, S[1].o + \rho)$. Το HETree-R ισοδύναμα ορίζεται μέσω του διαστήματος μήκους (interval length) $\rho$, αντί για τον αριθμό των φύλλων $\ell$.

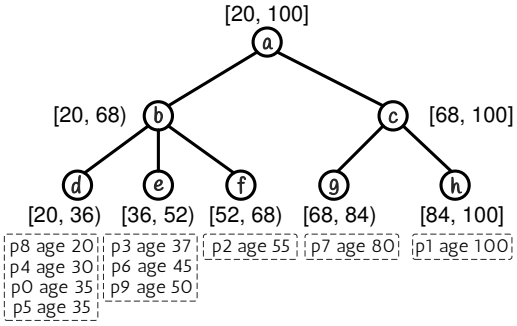



**Figure 4.3:** Range-based HETree (HETree-R)

#### 4.1.1.4.1 Η Κατασκευή του HETree-R

Αυτή η παράγραφος εξετάζει τη δημιουργία της δομής του HETree-R. Το HETree-R επίσης δημιουργείται από πάνω προς τα κάτω. Ομοίως με το τμήμα του HETree-C, ο Αλγόριθμος 6 χρησιμοποιείται για την δημιουργία του HETree-R. Η μόνη διαφορά είναι η constrLeaves-R διαδικασία (*line 2*), η οποία δημιουργεί τους κόμβους-φύλλα του HETree-R και παρουσιάζεται στην Διαδικασία 3.

Η διαδικασία παίρνει ως είσοδο ένα ταξινομημένο σύνολο αντικειμένων δεδομένων $S$, καθώς και τον αριθμό των φύλλων $\ell$. Πρώτα, υπολογίζει το εύρος $\rho$ των φύλλων (*line 1*). Η διαδικασία δημιουργεί $\ell$ φύλλα (*lines 2–9*) και αναθέτει ίδια διαστήματα για όλα (*lines 4–8*), διασχίζει όλα τα αντικείμενα $S$ (*λινες 10–12*) και τα τοποθετεί στα κατάλληλα φύλλα (*line 12*). Τέλος, αφαιρεί άδεια φύλλα (*lines 13–15*) και επιστρέφει τα φύλλα που δημιουργήθηκαν (*line 13*).

### 4.1.1.5 Υπολογισμός των Παραμέτρων του HETree

Σε ένα σενάριο εργασίας, ο χρήστης προσδιορίζει τις παραμέτρους που απαιτούνται για τη δημιουργία του HETree (π.χ., αριθμός των φύλλων $\ell$). Σε αυτή τη παράγραφο, περιγράφουμε την προσέγγιση μας για τον αυτόματο υπολογισμό των παραμέτρων του HETree βασισμένοι στα δεδομένα εισόδου, όταν δεν παρέχονται οι προτιμήσεις του χρήστη. Ο στόχος είναι να εξάγονται οι παράμετροι από τα δεδομένα εισόδου,

[5] Υποθέτουμε εδώ ότι, υπάρχει τουλάχιστον ένα αντικείμενο στο $D$ με διαφορετική τιμή από τα υπόλοιπα αντικείμενα.

**Procedure 3:** constrLeaves-R($S$, $\ell$)

**Input**: $S$: ordered set of objects; $\ell$: number of leaf nodes
**Output**: $L$: ordered set of leaf nodes

$\rho \leftarrow \frac{|S[1].o - S[|S|].o|}{\ell}$
**for** $i \leftarrow 1$ **to** $\ell$ **do**
  create an empty leaf node $n$
  **if** $i = 1$ **then**
    $n.I^- \leftarrow S[1].o$
  **else**
    $n.I^- \leftarrow L[i-1].I^+$
  $n.I^+ \leftarrow n.I^- + \rho$
  $L[i] \leftarrow n$
**for** $t \leftarrow 1$ **to** $|S|$ **do**
  $j \leftarrow \left\lfloor \frac{S[t].o - S[1].o}{\rho} \right\rfloor + 1$
  $L[j].data \leftarrow S[t]$
**return** $L$

έτσι ώστε το HETree που προκύπτει να μπορεί να ανταποκριθεί στις βασικές κατευθυντήριες γραμμές που καθορίζονται από το περιβάλλον οπτικοποίησης. Παρακάτω, παρουσιάζουμε την προτεινόμενη προσέγγιση.

Μια σημαντική παράμετρος στις ιεραρχικές οπτικοποιήσεις είναι ο ελάχιστος και ο μέγιστος αριθμός αντικειμένων που μπορούν να είναι αποδοτικά διαχειρίσιμες στο πιο λεπτομερές επίπεδο[6]. Στη περίπτωση μας, οι παραπάνω αριθμοί αντιστοιχούν στον αριθμό των αντικειμένων που περιέχονται στα φύλλα. Ο σωστός υπολογισμός αυτών των αριθμών είναι ζωτικής σημασίας προκειμένου να αποφευχθεί η υπερφόρτωση και οι διασκορπισμένες οπτικοποιήσεις.

Έτσι, στη κατασκευή του HETree, η προσέγγιση μας θεωρεί τον ελάχιστο και τον μέγιστο αριθμό αντικειμένων ανά φύλλο, που συμβολίζονται ως $\lambda_{min}$ και $\lambda_{max}$, αντίστοιχα. Εκτός του αριθμού αντικειμένων που βρίσκονται στο χαμηλότερο επίπεδο, η προσεγγισή μας θεωρεί τέλεια $m$-ary δέντρα, τέτοια ώστε να προκύπτει μια πιο ομοιόμορφη δομή (δηλαδή όλες οι ομάδες να χωρίζονται στον ίδιο αριθμό ομάδων).

### 4.1.2 Αποδοτική Πολυεπίπεδη Διερεύνηση

Σε αυτή τη παράγραφο, χρησιμοποιούμε τη δομή HETree ώστε να χειριζόμαστε αποδοτικά διαφορετικές περιπτώσεις πολυεπίπεδης διερεύνησης. Ουσιαστικά, προτείνουμε δύο μεθόδους αποδοτικής ιεραρχικής διερεύνησης μεγάλων συνόλων δεδομένων. Η πρώτη μέθοδος σταδιακά δημιουργεί την ιεραρχία μέσω της αλληλεπίδρασης με τον χρήστη, ενώ η δεύτερη επιτυγχάνει τη δυναμική προσαρμογή της οργάνωσης των δεδομένων βασισμένη στις προτιμήσεις του χρήστη.

Σενάρια Διερεύνησης

#### 4.1.2.1 Σενάρια Διερεύνησης

Σε μια τυπική περίπτωση πολυεπίπεδης διερεύνησης, η οποία εδώ θα αναφέρεται ώς *Basic exploration scenario* (BSC), ο χρήστης διερευνά ένα σύνολο δεδομένων από πάνω προς τα κάτω. Ο χρήστης πρώτα αποκτά μια αποπτεία των δεδομένων μέσω του

[6]Παρόμοια όρια μπορούν να οριστούν και για άλλα επίπεδα δέντρου.

επιπέδου της ρίζας (root level), και έπειτα προχωρά σε πιο λεπτομερή περιεχόμενα για να αποκτήσει πρόσβαση στα πραγματικά αντικείμενα δεδομένων στα φύλλα. Στο BSC, η ρίζα της ιεραρχίας είναι το σημείο εκκίνησης της διερεύνησης και έτσι, το πρώτο στοιχείο που παρουσιάζεται.

Το παραπάνω σενάριο προσφέρει βασικές δυνατότητες διερεύνησης, όμως δεν υποστηρίζει περιπτώσεις όπου ο χρήστης επιθυμεί να ορίζει ως σημεία εκκίνησης σημεία άλλα εκτός από τη ρίζα, όπως η περίπτωση να επιθυμεί να ξεκινήσει την συγκεκριμένο πόρο (resource), ή από συγκεκριμένο εύρος τιμών.

Ας δούμε το επόμενο παράδειγμα, στο οποίο ο χρήστης επιθυμεί να διερευνήσει το *DBpedia* infoboxes dataset ώστε να εντοπίσει τοποθεσίες με πολύ μεγάλο πληθυσμό. Αρχικά, επιλέγει την ιδιότητα *populationTotal* και ξεκινά τη διερεύνηση από τη ρίζα, ενώ μετά κινείται στο δεξίο μέρος του δέντρου και καταλήγει στο δεξιότερο φύλλο που περιέχει τις πιο πυκνοκατοικημένες τοποθεσίες. Τότε, ο χρήστης ενδιαφέρεται να δει την έκταση (δηλαδή ιδιότητα *areaTotal*) της πιο πυκνοκατοικημένης περιοχής και επίσης να διερευνήσει περιοχές με παρόμοια έκταση. Τέλος, αποφασίζει να διερευνήσει περιοχές βασισμένος στην έκταση των περιοχών νερού που περιέχουν (*areaWater*). Σε αυτή τη περίπτωση, προτιμά να ξεκινήσει τη διερεύνηση εξετάζοντας τις περιοχές των οποίων η έκταση των περιοχών νερού να βρίσκεται σε ένα δεδομένο εύρος τιμών.

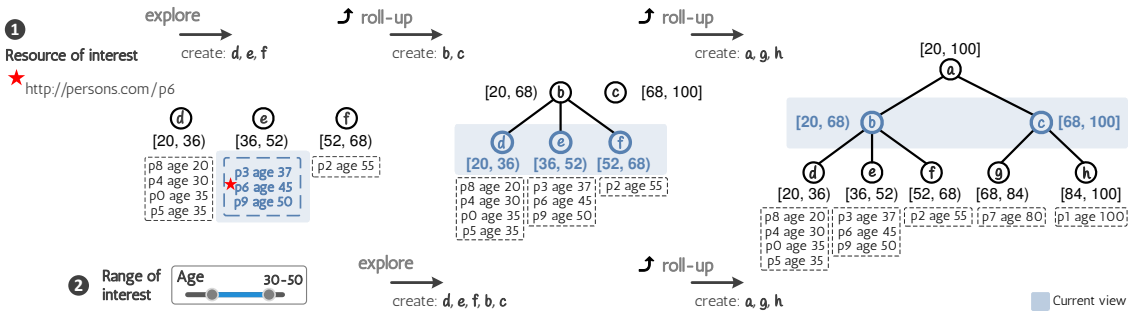
Σε αυτό το παράδειγμα, πέρα από το BSC, υπάρχουν δύο επιπρόσθετα σενάρια. Στο σενάριο της *Resource-based* (RES) διερεύνησης, ο χρήστης προσδιορίζει έναν πόρο ενδιαφέροντος (resource of interest) (δηλαδή, ένα IRI) και μια συγκεκριμένη ιδιότητα: η διερεύνηση ξεκινά από το φύλλο που περιέχει τον συγκεκριμένο πόρο και συνεχίζει από κάτω προς τα πάνω. Τέλος, το φύλλο που περιέχει τον πόρο ενδιαφέροντος, αναφερόμαστε σε αυτό ως *φύλλο ενδιαφέροντος* (leaf of interest).

Στο τρίτο σενάριο, που ονομάζεται *Range-based* (RAN) διερεύνηση, επιτρέπει στον χρήστη να ξεκινήσει τη διερεύνηση από ένα αυθαίρετο σημείο παρέχοντας ένα εύρος τιμών. Ο χρήστης ξεκινά από μια ομάδα από εσωτερικούς κόμβους και έπειτα συνεχίζει διασχίζοντας πάνω ή κάτω την ιεραρχία. Το RAN σενάριο ξεκινά παρουσιάζοντας όλα τα αδέρφια-κόμβους (sibling nodes) που είναι παιδιά του κόμβου που καλύπτει το συγκεκριμένο εύρος ενδιαφέροντος. Αναφερόμαστε σε αυτούς τους κόμβους ως *nodes of interest* ή κόμβοι ενδιαφέροντος.

Αναφορικά με τις λειτουργίες πλοήγησης ("navigation-related" operations), ο χρήστης μπορεί να κινηθεί πάνω ή κάτω στην ιεραρχία πραγματοποιώντας λειτουργία *drill-down* ή λειτουργία *roll-up*, αντίστοιχα. Μια λειτουργία drill-down σε έναν κόμβο $n$ επιτρέπει στον χρήστη να επικεντρωθεί στον $n$ και να απεικονίσει τα παιδιά του. Αν ο κόμβος $n$ είναι φύλλο, τότε απεικονίζεται η ομάδα των δεδομένων που περιέχεται στο $n$. Από την άλλη, ο χρήστης μπορεί να πραγματοποιήσει μια λειτουργία *roll-up* σε μια ομάδα κόμβων-αδερφών $S$. Ο κόμβος γονέας του $S$ καθώς και οι κόμβοι-γονείς των αδερφών απεικονίζονται. Τέλος, η λειτουργία roll-up όταν εφαρμόζεται σε μια ομάδα δεδομένων $O$ θα παρουσιάσει το φύλλο που περιέχει $O$ καθώς και τα φύλλα-αδέρφια, ενώ μια λειτουργία drill-down δεν εφαρμόζεται σε ένα αντικείμενο δεδομένων.

#### 4.1.2.2 Σταδιακή Κατασκευή του HETree

Στον Ιστό Δεδομένων, τα δεδομένα μπορούν να ανακτηθούν δυναμικά από ένα απομακρυσμένο site (π.χ. μέσω ενός SPARQL endpoint), και συνεπώς, σε όλες τις περιπτώσεις διερεύνησης υποθέτουμε ότι το HETree κατασκευάζεται άμεσα όταν ο χρήστης ξεκινά τη διερεύνηση. Όμως, η δημιουργία ιεραρχιών για μεγάλα σύνολα δεδομένων έχει ως αποτέλεσμα μεγάλο χρόνο απόκρισης για την έναρξη της διερεύνηση




**Σχήμα 4.4:** Παράδειγμα σταδιακής κατασκευής HETree. ❶ Resource-based (RES) exploration scenario; ❷ Range-based (RAN) exploration scenario

όπως φαίνεται και στο πειραματικό τμήμα της εργασίας.

Σε αυτή την ενότητα, παρουσιάζουμε τη μέθοδο ICO (**I**ncremental HETree **Co**nstruction), η οποία σταδιακά κατασκευάζει το HETree, βασιζόμενη στην αλληλεπίδραση με τον χρήστη. Η προτεινόμενη μέθοδος προχωρά παραπέρα από τη σταδιακή κατασκευή του δέντρου, στοχεύοντας στη παραπέρα μείωση του χρόνου απόκρισης κατά τη διαδικασία της διερεύνησης μέσω της δημιουργίας από πριν (preconstructing-prefetching) των τμημάτων του δέντρου τα οποία θα επισκεφτεί ο χρήστης στην επόμενη λειτουργία roll-up ή drill-down. Έτσι, ένας κόμβος $n$ δεν δημιουργείται όταν ο χρήστης τον επισκέπτεται για πρώτη φορά, αφού αντιθέτως έχει δημιουργηθεί σε προηγούμενο στάδιο της διερεύνησης, όπου ο χρήστης ήταν σε έναν κόμβο, στον οποίο ο $n$ είναι προσπελάσιμος με μια λειτουργία roll-up ή drill-down. Με αυτόν τον τρόπο, η μέθοδος μας προσφέρει σταδιακή δημιουργία του δέντρου, προσαρμοσμένη στη διερεύνηση του κάθε χρήση. Τέλος, δείχνουμε ότι κατά τη διαδικασία της διερεύνησης, η μέθοδος ICO δημιουργεί τον ελάχιστο αριθμό στοιχείων του HETree.

Στην αρχή κάθε περίπτωσης διερεύνησης, η ICO δημιουργεί μια ομάδα *αρχικών κόμβων* (initial nodes), οι οποίοι είναι οι κόμβοι που παρουσιάζονται αρχικά, καθώς και οι κόμβοι οι οποίοι μπορούν να προσεγγιστούν από τη πρώτη λειτουργία του χρήστη (δηλαδή, τα *απαιτούμενα στοιχεία* του HETree). Τα *απαιτούμενα στοιχεία του HETree* ενός σταδίου διερεύνησης είναι κόμβοι που μπορούν να προσεγγιστούν από τον χρήστη μέσω μίας λειτουργίας διερεύνησης. Έτσι, στη περίπτωση του RES, οι αρχικοί κόμβοι είναι το φύλλο ενδιαφέροντος και τα αδέρφια-φύλλα του. Στο RAN, οι αρχικοί κόμβοι είναι οι κόμβοι ενδιαφέροντος, τα παιδιά τους, ο κόμβος-γονέας τους καθώς και τα αδέρφια τους. Τέλος, στη περίπτωση του BSC οι αρχικοί κόμβοι η ρίζα και τα παιδιά της. Στο Σχήμα 4.4 παρουσιάζονται παραδείγματα σταδιακής κατασκευής.

Στην συνέχεια, περιγράφουν τους *κανόνες δημιουργίας* που ακολουθεί η ICO κατά τη διάρκεια της διαδικασίας διερεύνησης. Αυτοί οι κανόνες προσφέρουν αντιστοιχίσεις μεταξύ των ειδών των στοιχείων που παρουσιάζονται σε κάθε στάδιο διερεύνησης και των στοιχείων που δημιουργεί η ICO. Σημειώνουμε ότι αυτοί οι κανόνες εφαρμόζονται μετά τη δημιουργία των αρχικών κόμβων, και στα τρία σενάρια διερεύνησης.

**Κανόνας 1:** *Αν παρουσιάζεται ένα σύνολο εσωτερικών αδερφικών κόμβων $C$*, η ICO δημιουργεί: $(i)$ τον κόμβο-γονέα του $C$ μαζί με τους αδερφούς-κόμβους των γονέων και $(ii)$ τα παιδιά κάθε κόμβου στο $C$.

**Κανόνας 2:** *Αν παρουσιάζεται ένα σύνολο από φύλλα αδερφών κόμβων $L$*, η ICO δεν δημιουργεί τίποτα (οι απαιτούμενοι κόμβοι έχουν ήδη δημιουργηθεί προηγουμένως).

**Κανόνας 3:** *Αν παρουσιάζεται ένα σύνολο από αντικείμενα δεδομένων $O$*, η ICO δεν δημιουργεί τίποτα (οι απαιτούμενοι κόμβοι έχουν ήδη δημιουργηθεί προηγουμένως).

**Σχόλιο 1.** Κάθε φορά που η ICO δημιουργεί ένα κόμβο (είτε αρχικό κόμβο είτε λόγω των κανόνων δημιουργίας), δημιουργεί επίσης και όλους τους αδελφικούς του κόμβους.

Η ακόλουθη πρόταση δείχνει ότι, σε κάθε περίπτωση, τα απαιτούμενα HETree στοιχεία έχουν ήδη δημιουργηθεί νωρίτερα από την ICO.

**Πρόταση 1.** Σε κάθε περίπτωση διερεύνησης, τα HETree στοιχεία που μπορεί να φτάσει ο χρήστης μέσω μιας λειτουργίας (δηλαδή τα απαιτούμενα στοιχεία), έχουν προηγουμένως δημιουργηθεί από την ICO.

Επίσης, το ακόλουθο θεώρημα δείχνει ότι σε κάθε περίπτωση διερεύνησης η ICO δημιουργεί μόνο τα απαιτούμενα HETree στοιχεία.

**Θεώρημα 1.** Η ICO δημιουργεί τον ελάχιστο αριθμό HETree στοιχείων σε κάθε περίπτωση διερεύνησης.

#### 4.1.2.2.1 Ο αλγόριθμος ICO

Σε αυτή την ενότητα, παρουσιάζουμε τον αλγόριθμο σταδιακής δημιουργίας του HETree. Σημειώνουμε ότι εδώ περιλαμβάνουμε τον ψευδοκώδικα μόνο για την έκδοση HETree-R, αφού η μόνη διαφορά με την HETree-C έκδοση έγκειται στο ότι τα διαστήματα των κόμβων έχουν υπολογιστεί και στο ότι το σύνολο δεδομένων είναι αρχικά ταξινομημένο.

Ο αλγόριθμος ICO-R (Αλγόριθμος 7) υλοποιεί την σταδιακή μέθοδο για το HETree-R. Ο αλγόριθμος χρησιμοποιεί δύο διαδικασίες για να δημιουργήσει όλους τους απαραίτητους κόμβους. Η πρώτη διαδικασία constrRollUp-R (Procedure 4) δημιουργεί τους κόμβους οι οποίοι προσεγγίζονται μέσω μιας λειτουργίας roll-up, όπου η constrDrillDown-R (Procedure 5) δημιουργεί τους κόμβους οι οποίοι προσεγγίζονται μέσω μιας λειτουργίας drill-down. Επιπρόσθετα, οι διαδικασίες που αναφέρθηκαν πριν χρησιμοποιούν δύο δευτερεύοντες διαδικασίες: την computeSiblingInterv-R (Procedure 6) και την constrSiblingNodes-R (Procedure 7), οι οποίες χρησιμοποιούνται για τον υπολογισμό των διαστημάτων των κόμβων και τη δημιουργία των κόμβων.

**Algorithm 7.** ICO-R($D$, $\ell$, $d$, $U$, $cur$, $H$)

**Input**: $D$: set of objects; $\ell$: number of leaf nodes; $d$: tree degree;
$U$: interval representing user's starting point; $cur$: currently presented elements;
$H$: currently created HETree-R
**Output**: $H$: updated HETree-R
**Variables**: $len$: the length of the leaf's interval

```
1  if cur = null then                                          //first ICO call
2      len ← (D.maxv − D.minv) / ℓ
3      from U compute I_0, h_0                                 //used for constructing initial nodes
4      cur, H ← constrSiblingNodes-R(I_0, null, D, h_0)
5      if RES then return H
6  if cur[1].p = null and D ≠ ∅ then
7      H ← constrRollUp-R(D, d, cur, H)
8      if cur[1].h > 0 then                                    //cur are not leaves
9          H ← constrDrillDown-R(D, d, cur, H)
10 return H
```

**Procedure 4:** constrRollUp-R($D$, $d$, $cur$, $H$)

**Input**: $D$: set of objects; $d$: tree degree; $cur$: currently presented elements; $H$: currently created HETree-R

**Output**: $H$: updated HETree-R

//Computed in ICO-R: $len$: the length of the leaf's interval

create an empty node $par$ //$cur$ parent node
$par.h \leftarrow cur[1].h + 1$
$par.I^- \leftarrow cur[1].I^-$
$par.I^+ \leftarrow cur[|cur|].I^+$
**for** $i \leftarrow 1$ **to** $|cur|$ **do** //create parent-child relations
    $par.c[i] \leftarrow cur[i]$
    $cur[i].p \leftarrow par$
insert $par$ into $H$
$l_p \leftarrow par.I^+ - par.I^-$ //$par$ interval length
$I^-_{ppar} \leftarrow D.minv + d \cdot l_p \cdot \left\lfloor \frac{par.I^- - D.minv}{d \cdot l_p} \right\rfloor$
$I^+_{ppar} \leftarrow \min(D.maxv, I^-_{ppar} + d \cdot l_p)$ //compute interval for $par$ parent, $I_{ppar}$
$l_{sp} \leftarrow (len \cdot d^{cur[1].h})$ //interval length for a $par$ sibling node
$I_{spar} \leftarrow$ computeSiblingInterv-R($I^-_{ppar}, I^+_{ppar}, l_{sp}, d$) //compute intervals for all $par$ sibling nodes
remove $par.I$ from $I_{spar}$ //remove $par$ interval, $par$ already constructed
$S \leftarrow$ constrSiblingNodes-R($I_{spar}$, null, $D, cur[1].h + 1$)
insert $S$ into $H$
**return** $H$

#### 4.1.2.3 Προσαρμοστική Κατασκευή του HETree

Σε ένα σενάριο οπτικής διερεύνησης, ο χρήστης επιθυμεί να τροποποιήσει την οργάνωση των δεδομένων, παρέχοντας τις προτιμήσεις του για όλη την ιεραρχία ή κάποιο κομμάτι της. Οι χρήστες μπορούν να επιλέξουν συγκεκριμένο ένα υπόδεντρο και να αλλάξουν τον αριθμό των ομάδων που παρουσιάζονται σε κάθε επίπεδο (δηλαδή τον βαθμό του δέντρου) ή το μέγεθος των ομάδων (δηλαδή τον αριθμό των φύλλων). Σε αυτή τη περίπτωση, πρέπει να δημιουργηθεί άμεσα ένα νέο δέντρο (ή ένα κομμάτι του) που ανταποκρίνεται στις νέες παραμέτρους που έδωσε ο χρήστης.

Για παράδειγμα, ας σκεφτούμε το HETree-C στο Σχήμα 4.5 που αντιπροσωπεύει ηλικίες προσώπων[7]. Ένας χρήστης μπορεί να περιηγηθεί στον κόμβο $b$, όπου επιθυμεί

[7]Για απλότητα, το Σχήμα 4.5 παρουσιάζει μόνο τη τιμή των αντικειμένων

**Procedure 5:** constrDrillDown-R($D$, $d$, $cur$, $H$)

**Input**: $D$: set of objects; $d$: tree degree; $cur$: currently presented elements; $H$: currently created HETree-R

**Output**: $H$: updated HETree-R

//Computed in ICO-R: $len$: the length of the leaf's interval

$l_c = len \cdot d^{cur[1].h-1}$ //length of the children's intervals
**for** $i \leftarrow 1$ **to** $|cur|$ **do**
    **if** $cur[i].c[0] =$ null **then continue** //nodes previously constructed
    $I_{ch} \leftarrow$ computeSiblingInterv-R($cur[i].I^-, cur[i].I^+, l_c, d$) //compute intervals for $cur[i]$ children
    $S \leftarrow$ constrSiblingNodes-R($I_{ch}, cur[i], cur[i].data, cur[1].h - 1$)
    **for** $k \leftarrow 1$ **to** $|S|$ **do**
        $cur[i].c[k] \leftarrow S[k]$
    insert $S$ into $H$
**return** $H$

**Procedure 6:** computeSiblingInterv-R($low$, $up$, $len$, $n$)

**Input**: $low$: intervals' lower bound; $up$: intervals' upper bound; $len$: intervals' length; $n$: number of siblings
**Output**: $I$: an ordered set with at most $n$ equal length intervals

```
1 I_t^-, I_t^+ ← low
2 for i ← 1 to n do
3 |   I_t^- ← I_t^+
4 |   I_t^+ ← min(up, len + I_t^-)
5 |   append I_t to I
6 |   if I_t^+ = up then break
7 return I
```

**Procedure 7:** constrSiblingNodes-R($I$, $p$, $A$, $h$)

**Input**: $I$: an ordered set with equal length intervals $p$: nodes' parent node; $A$: available data objects; $h$: nodes' height
**Output**: $S$: a set of HETree-R sibling nodes

```
1 l = I[1]^+ − I[1]^−                                   //intervals' length
2 T[ ] ← ∅
3 foreach tr ∈ A do                                     //indicate enclosed data for each node
4 |   j ← ⌊(tr.o − I[1]^−) / l⌋ + 1
5 |   if j ≥ 0 and j ≤ |I| then
6 |   |   insert object tr into T[j]
7 |   |   remove object tr from A
8 for i ← 1 to |I| do                                   //construct nodes
9 |   if T[i] = ∅ then continue
10|   create a new node n
11|   n.I^− ← I[i]^−
12|   n.I^+ ← I[i]^+
13|   n.p ← p
14|   n.c ← null
15|   n.data ← T[i]
16|   n.h ← h
17|   if h = 0 then                                     //node is a leaf
18|   |   sort n.data based on objects values
19|   append n to S
20 return S
```

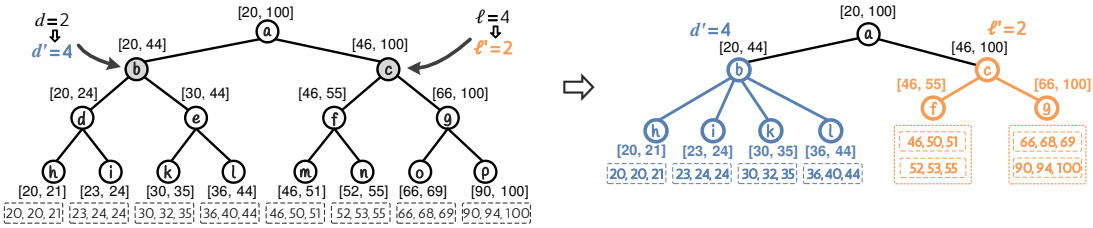
να αυξήσει τον αριθμό των ομάδων που παρουσιάζονται σε αυτό το επίπεδο. Έτσι, τροποποιεί το βαθμό $b$ από 2 σε 4 και το υπόδεντρο προσαρμόζεται στη νέα παράμετρο όπως απεικονίζεται στο κάτω δέντρο στο Σχήμα 4.5. Από την άλλη, ο χρήστης επιλέγει να διερευνήσει το δεξί υπόδεντρο (ξεκινώντας από τον κόμβο $c$) με λιγότερες λεπτομέρειες. Επιλέγει να αυξήσει το μέγεθος των ομάδων μειώνοντας (από 4 σε 2) τον αριθμό των φύλλων για το υπόδεντρο $c$. Σε κάθε περίπτωση, η δημιουργία ενός υποδέντρου από το μηδέν, βασισμένοι στις παραμέτρους που έδωσε ο χρήστη, καθώς και οι εκ νέου υπολογισμοί των στατιστικών, επιφέρουν μεγάλη χρονική καθυστέρηση, ειδικά όταν οι προτιμήσεις του χρήστη εφαρμόζονται σε ένα μεγάλο μέρος της ιεραρχίας.

Σε αυτό το τμήμα, παρουσιάζουμε τη μέθοδο ADA (**Ada**ptive HETree Construction), η οποία δυναμικά προσαρμόζει ένα υπάρχον HETree με βάση ένα σύνολο νέων παραμέτρων που δίνεται από τον χρήστη. Η μέθοδος μας, αντί να δημιουργεί το δέντρο και να υπολογίζει τα στατιστικά των κόμβων από το μηδέν, ανακατασκευάζει τα νέα

**Πίνακας 4.1:** Περίληψη της προσαρμοστικής κατασκευή του HETree ⋆

| | | **Modify Degree** | | | | **Modify Num. of Leaves** | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Full Construction** | $d' = d^k$ | $d' = k \cdot d$ | $d' = \sqrt[k]{d}$ | *elsewhere* | $\ell' > \ell$ | $\ell' = \frac{\ell}{d^k}$ | $\ell' = \frac{\ell}{k}$ | $\ell' = \ell - k$ |
| **Tree Construction** | | | | | | | | | |
| **Complexity** | $O(mlogm+d'e)$ | $O(mlog_{\sqrt[k]{d'}}m)$ | $O(d'e)$ | $O(d'^k r)$ | $O(d'e)$ | $O(m+d'e)$ | $O(m)$ | $O(m+d'e)$ | $O(mlogm+d'e)$ |
| **#leaves**$_0$ | $\ell'$ | 0 | 0 | 0 | 0 | $\ell'$ | 0 | 0 | $\ell'$ |
| **#leaves**$_+$ | 0 | 0 | 0 | 0 | 0 | 0 | $\ell'$ | $\ell'$ | 0 |
| **#internals**$_0$ | $e$ | 0 | $e$ | $e-r$ | $e$ | $e$ | 0 | $e$ | $e$ |
| **#internals**$_+$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Statistics Computations** | | | | | | | | | |
| **Complexity** | $O(m+d'e)$ | $O(1)$ | $O(\frac{k\ell'}{d'}+d'e)$ | $O(d'(e-r))$ | $O(d'e)$ | $O(m+d'e)$ | $O(1)$ | $O(m+d'e)$ | $O(m+d'e-\ell'-k)$ |
| **#leaves**$_0$ | $\ell'$ | 0 | 0 | 0 | 0 | $\ell'$ | 0 | 0 | $\ell' - \frac{\ell'^2}{d'}$ |
| **#leaves**$_+$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | $\ell'$ | $\frac{\ell'^2}{d'}$ |
| **#internals**$_0$ | $e$ | 0 | $e - \left\lceil \frac{\ell'}{d'} \right\rceil$ | $e-r$ | $e$ | $e$ | 0 | $e$ | $e$ |
| **#internals**$_+$ | 0 | 0 | $\left\lceil \frac{\ell'}{d'} \right\rceil$ | 0 | 0 | 0 | 0 | 0 | 0 |

⋆$m = |D|$, $e = \frac{d'\ell'-1}{d'-1}$ (maximum number of internal nodes), and $r = \frac{d'^k\ell'-1}{d'^k-1}$




**Σχήμα 4.5:** Παράδειγμα προσαρμοστικής κατασκευή του HETree

κομμάτια της ιεραρχίας αξιοποιώντας τα υπάρχοντα στοιχεία (δηλαδή κόμβοι, στατιστικά) του δέντρου. Με αυτό τον τρόπο, η ADA καταφέρνει να μειώσει το συνολικό κόστος δημιουργίας και επιτρέπει την άμεση αναδιοργάνωση των οπτικοποιημένων δεδομένων. Στο παράδειγμα στο Σχήμα 4.5, το νέο υπόδεντρο του $b$ μπορεί να παραχθεί από το παλιό, αφαιρώντας τους εσωτερικούς κόμβους $d$ και $e$, ενώ ένα νέο υπόδεντρο του $c$ προκύπτει από συγχωνεύσεις φύλλων και αθροίζοντας τα στατιστικά τους.

Έστω ότι το $\mathcal{T}(D, \ell, d)$ δηλώνει το υπάρχον HETree και $\mathcal{T}'(D, \ell', d')$ το νέο HETree που αντιστοιχεί στις καινούργιες προτιμήσεις του χρήστη για βαθμό δέντρου $d'$ και αριθμό φύλλων $\ell'$. Σημειώνουμε ότι ο συμβολισμός $\mathcal{T}$ μπορεί επίσης να δηλώνει ένα υπόδεντρο ενός υπάρχοντος HETree (στη περίπτωση που ο χρήστης τροποποιεί μόνο ένα τμήμα του). Σε αυτή τη περίπτωση, ο χρήστης υποδηλώνει την *ρίζα αναδημιουργίας* (reconstruction root) του $\mathcal{T}$.

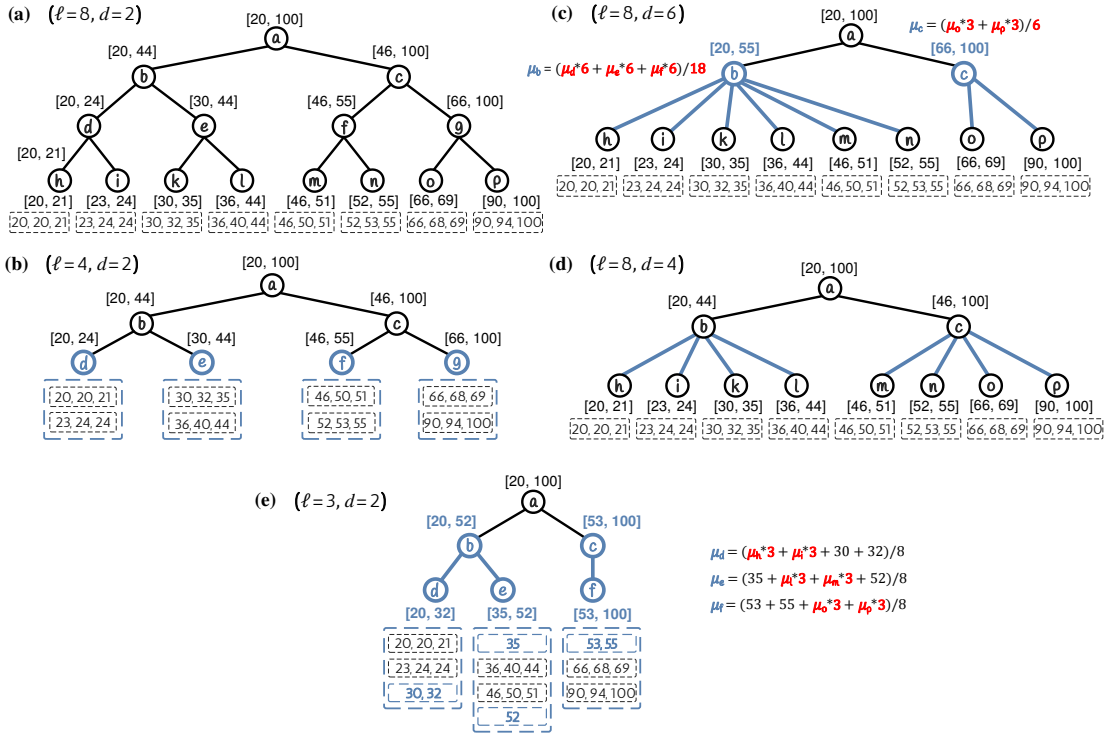
Τότε, η ADA αναγνωρίζει τα ακόλουθα στοιχεία του $\mathcal{T}$: (1) Τα στοιχεία του $\mathcal{T}$ που υπάρχουν επίσης στο $\mathcal{T}'$. Για παράδειγμα, ας σκεφτούμε τις εξής δύο περιπτώσεις: Πρώτον, τα φύλλα του $\mathcal{T}'$ να είναι εσωτερικοί κόμβοι του $\mathcal{T}$ σε επίπεδο $x$. Δεύτερον, τα στατιστικά των $\mathcal{T}'$ κόμβων σε επίπεδο $x$ να είναι ίδια με τα στατιστικά των $\mathcal{T}$ κόμβων σε επίπεδο $y$. (2) Τα στοιχεία του $\mathcal{T}$ που μπορούν να ξαναχρησιμοποιηθούν για να δημιουργηθούν στοιχεία στο $\mathcal{T}'$. Για παράδειγμα, ας σκεφτούμε τις εξής δύο περιπτώσεις: Πρώτον, κάθε φύλλο του $\mathcal{T}'$ να δημιουργείται μέσω της συγχώνευσης $x$ φύλλων του $\mathcal{T}$. Δεύτερον, τα στατιστικά του κόμβου $n$ του $\mathcal{T}'$ να μπορούν να υπολογιστούν αθροίζοντας τα στατιστικά των κόμβων $q$ και $w$ του $\mathcal{T}$.

Συνεπώς, θεωρούμε ότι ένα στοιχείο (δηλαδή ένας κόμβος ή τα στατιστικά ενός κόμβου) στο $\mathcal{T}'$ μπορεί: (1) να δημιουργηθεί/υπολογιστεί από την αρχή[8], (2) να ξαναχρησιμοποιηθεί όπως είναι από το $\mathcal{T}$ ή (3) να προέλθει συναθροίζοντας στοιχεία από το $\mathcal{T}$.

Ο Πίνακας 4.1 συνοψίζει την ADA διαδικασία. Συγκεκριμένα, ο πίνακας περιλαμβάνει: (1) την υπολογιστική πολυπλοκότητα για τη δημιουργία του $\mathcal{T}'$, που συμβολίζεται ως: *Complexity*, (2) τον αριθμό των φύλλων και των εσωτερικών κόμβων του $\mathcal{T}'$ που δημιουργήθηκαν από την αρχή, που συμβολίζεται ως $\#leaves_0$ και $\#internals_0$, αντίστοιχα και (3) τον αριθμό των φύλλων και των εσωτερικών κόμβων του $\mathcal{T}'$ που προκύπτουν από κόμβους του $\mathcal{T}$, που συμβολίζονται ως $\#leaves_+$ και $\#internals_+$, αντίστοιχα. Το χαμηλότερο σημείο του πίνακα παρουσιάζει τα αποτελέσματα για τον υπολογισμό των στατιστικών των κόμβων στο $\mathcal{T}'$. Τέλος, η δεύτερη στήλη του πίνακα, που συμβολίζεται ως *Full Construction*, παρουσιάζει τα αποτελέσματα από τη δημιουργία του $\mathcal{T}'$ από το μηδέν.

Στης επόμενες παραγράφους, παρουσιάζουμε λεπτομερώς την διαδικασία της ανακατασκευής μέσω των δέντρων του Σχήμα 4.6. Το Σχήμα 4.6a παρουσιάζει το αρχικό

[8]Σημειώνουμε ότι είναι πιθανό για έναν κόμβο στο $\mathcal{T}'$ που δεν έχει δημιουργηθεί από το μηδέν να αθροίσει τα στατιστικά του από τους κόμβους στο $\mathcal{T}$.




**Σχήμα 4.6:** Παραδείγματα προσαρμοστικής κατασκευή

δέντρο $\mathcal{T}$ το οποίο είναι ένα δέντρο HETree-C, με $\ell = 8$ και $d = 2$. Οι εικόνες 4.6b ~ 4.6e παρουσιάζουν αρκετά ανακατασκευασμένα δέντρα $\mathcal{T}'$. Το μπλε χρώμα χρησιμοποιείται για να υποδείξει τα στοιχεία (κόμβοι, ακμές, στατιστικά) του $\mathcal{T}'$ που δεν υπάρχουν στο $\mathcal{T}$. Σχετικά με τα στατιστικά, υποθέτουμε ότι σε κάθε κόμβο υπολογίζουμε τη μέση τιμή. Σε κάθε $\mathcal{T}'$, παρουσιάζουμε μόνο τις μέσες τιμές που δεν είναι γνωστές από το $\mathcal{T}$.

#### 4.1.2.3.1 Ο χρήστης τροποποιεί τον βαθμό του δέντρου

Σχετικά με τη τροποποίηση της παραμέτρου του βαθμού του δέντρου, διακρίνουμε τις εξής περιπτώσεις:

**Ο χρήστης αυξάνει τον βαθμό του δέντρου.** Έχουμε ότι $d' > d$, με βάση τον βαθμό $d'$ έχουμε τις ακόλουθες περιπτώσεις:

(1) $d' = d^k$, με $k \in \mathbb{N}^+$ και $k > 1$

Το Σχήμα 4.6a παρουσιάζει $\mathcal{T}$ με $d = 2$ και το Σχήμα 4.6d παρουσιάζει το ανακατασκευασμένο $\mathcal{T}'$ με $d' = 4$ (δηλαδή, $k = 2$).

Το $\mathcal{T}'$ προκύπτει απλά αφαιρώντας τους κόμβους με ύψος 1 (δηλαδή, $d$, $e$, $f$, $g$) και ενώνοντας τους κόμβους με ύψος 2 (δηλαδή, $b$, $c$) με τα φύλλα. Γενικά, το $\mathcal{T}'$ προκύπτει από το $\mathcal{T}$ απλά αφαιρώντας βαθμούς δέντρου από το $\mathcal{T}$. Ακόμα, δεν υπάρχει ανάγκη για υπολογισμό κανενός νέου στατιστικού, αφού τα στατιστικά για όλους τους κόμβους του $\mathcal{T}'$ παραμένουν τα ίδια όπως και στο $\mathcal{T}$.

(2) $d' = k \cdot d$, με $k \in \mathbb{N}^+$, $k > 1$ και $k \neq d^\nu$ όπου $\nu \in \mathbb{N}^+$

Ένα παράδειγμα με $k = 3$ παρουσιάζεται στο Σχήμα 4.6c, όπου έχουμε ότι $d' = 6$. Σε αυτή τη περίπτωση, τα φύλλα του $\mathcal{T}$ (Σχήμα 4.6a) παραμένουν φύλλα στο $\mathcal{T}'$ και όλοι οι εσωτερικοί κόμβοι μέχρι και την ρίζα αναδημιουργίας (reconstruction root) του $\mathcal{T}$ δημιουργούνται από το μηδέν. Αναφορικά με τα στατιστικά των κόμβων, μπορούμε να υπολογίσουμε τις μέσες τιμές των κόμβων του $\mathcal{T}'$ με ύψος 1 (δηλαδή, $\mu_b$, $\mu_c$) αθροίζοντας ήδη υπολογισμένες μέσες τιμές (δηλαδή, $\mu_d$, $\mu_e$, κτλ.) από $\mathcal{T}$.

Γενικά, εκτός από τα φύλλα, δημιουργούμε όλους τους εσωτερικούς κόμβους από το μηδέν. Για τους εσωτερικούς κόμβους με ύψος 1, υπολογίζουμε τα στατιστικά τους αθροίζοντας τα στατιστικά των φύλλων του $\mathcal{T}$, ενώ για τους εσωτερικούς κόμβους με ύψος μεγαλύτερο του 1, υπολογίζουμε από το μηδέν τα στατιστικά τους.

(3) *elsewhere*

Σε κάθε άλλη περίπτωση όπου ο χρήστης αυξάνει τον βαθμό του δέντρου, όλοι οι εσωτερικοί κόμβοι στο $\mathcal{T}'$ εκτός από τα φύλλα δημιουργούνται από το μηδέν. Σε αντίθεση με την προηγούμενη περίπτωση, τα στατιστικά του φύλλου από το $\mathcal{T}$ δεν μπορούν να ξαναχρησιμοποιηθούν και για αυτό, για όλους τους εσωτερικούς κόμβους στο $\mathcal{T}'$ τα στατιστικά ξανά-υπολογίζονται.

**Ο χρήστης μειώνει τον βαθμό του δέντρου.** Εδώ έχουμε ότι $d' < d$, βασισμένοι στον βαθμό $d'$ έχουμε τις εξής δύο περιπτώσεις:

(1) $d' = \sqrt[k]{d}$, με $k \in \mathbb{N}^+$ και $k > 1$

Υποθέτουμε ότι το Σχήμα 4.6d απεικονίζει το $\mathcal{T}$, με $d = 4$, ενώ το Σχήμα 4.6a παρουσιάζει το $\mathcal{T}'$ με $d' = 2$. Μπορούμε να παρατηρήσουμε ότι το $\mathcal{T}'$ περιέχει όλους τους κόμβους του $\mathcal{T}$, καθώς και μια ομάδα επιπλέον εσωτερικών κόμβων (δηλαδή, $d$, $e$, $f$, $g$). Έτσι, το $\mathcal{T}'$ προκύπτει από το $\mathcal{T}$ δημιουργώντας μερικούς νέους εσωτερικούς κόμβους.

(2) *elsewhere*

Αυτή η περίπτωση είναι όμοια με την προηγούμενη περίπτωση (3) όπου ο χρήστης αυξάνει τον βαθμό του δέντρου.

#### 4.1.2.3.2 Ο χρήστης τροποποιεί τον αριθμό των φύλλων

Σχετικά με τη τροποποίηση της παραμέτρου για τον αριθμό των φύλλων, διακρίνουμε τις εξής περιπτώσεις:

**Ο χρήστης αυξάνει τον αριθμό των φύλλων.** Σε αυτή τη περίπτωση έχουμε ότι $\ell' > \ell$, έτσι, κάθε φύλλο του $\mathcal{T}$ χωρίζεται σε αρκετά φύλλα στο $\mathcal{T}'$ και τα αντικείμενα που περιέχονται σε ένα φύλλο του $\mathcal{T}$ πρέπει να μετακινηθούν στα νέα φύλλα του $\mathcal{T}'$. Ως αποτέλεσμα, τόσο οι εσωτερικοί κόμβοι όσο και τα φύλλα στο $\mathcal{T}'$ έχουν διαφορετικά περιεχόμενα σε σύγκριση με τους κόμβους στο $\mathcal{T}$ και πρέπει να δημιουργούνται από το μηδέν μαζί με τα στατιστικά τους. Σε αυτή τη περίπτωση, η δημιουργία του $\mathcal{T}'$ απαιτεί $O(|D| + \frac{d^2 \cdot \ell' - d}{d-1})$ (αποφεύγοντας το στάδιο της ταξινόμησης).

**Ο χρήστης μειώνει τον αριθμό των φύλλων.** Σε αυτή τη περίπτωση έχουμε ότι $\ell' < \ell$, βασισμένοι στην $\ell'$ τιμή έχουμε τις ακόλουθες τρεις περιπτώσεις:

(1) $\ell' = \dfrac{\ell}{d^k}$, με $k \in \mathbb{N}^+$

Το Σχήμα 4.6α παρουσιάζει το $\mathcal{T}$ με $\ell = 8$ και $d = 2$. Ένα παράδειγμα αυτής της περίπτωσης με $k = 1$, παρουσιάζεται στο Σχήμα 4.6b, όπου έχουμε το $\mathcal{T}'$ με $\ell' = 4$. Στο Σχήμα 4.6b, παρατηρούμε ότι τα φύλλα στο $\mathcal{T}'$ προκύπτουν από τη συγχώνευση $d^k$ φύλλων του $\mathcal{T}$. Για παράδειγμα, το φύλλο $d$ του $\mathcal{T}'$ προκύπτει από τη συγχώνευση των φύλλων $h$ και $i$ του $\mathcal{T}$. Τότε, το $\mathcal{T}'$ προκύπτει από το $\mathcal{T}$, αντικαθιστώντας τους κόμβους του $\mathcal{T}$ με ύψος $k$ (δηλαδή, $b$, $e$, $f$, $g$), με τα φύλλα του $\mathcal{T}'$. Τέλος, οι κόμβοι του $\mathcal{T}$ με ύψος μικρότερου του $k$ δεν περιλαμβάνονται στο $\mathcal{T}'$.

Έτσι, σε αυτή τη περίπτωση, το $\mathcal{T}'$ δημιουργείται με τη συγχώνευση των φύλλων του $\mathcal{T}'$ και με την αφαίρεση των εσωτερικών κόμβων του $\mathcal{T}'$ που έχουν ύψος μικρότερο ή ίσο του $k$. Επίσης, δεν επαναυπολογίζουμε τις στατιστικές των νέων φύλλων του $\mathcal{T}'$ αφού αυτές παράγονται από τα στατιστικά των αφαιρούμενων κόμβων με ύψος $k$.

(2) $\ell' = \dfrac{\ell}{k}$, με $k \in \mathbb{N}^+$, $k > 1$ και $k \neq d^\nu$, όπου $\nu \in \mathbb{N}^+$

Όπως στην προηγούμενη περίπτωση, τα φύλλα στο $\mathcal{T}'$ δημιουργούνται από τη συγχώνευση φύλλων από το $\mathcal{T}$ και τα στατιστικά τους υπολογίζονται βάσει των στατιστικών των συγχωνευμένων φύλλων. Σε αυτή τη περίπτωση, όμως, όλοι οι εσωτερικοί κόμβοι στο $\mathcal{T}'$ πρέπει να δημιουργηθούν από το μηδέν.

(3) $\ell' = \ell - k$, με $k \in \mathbb{N}^+$, $k > 1$ και $\ell' \neq \dfrac{\ell}{\nu}$, οπου $\nu \in \mathbb{N}^+$

Οι δύο προηγούμενες περιπτώσεις περιγράφουν ότι κάθε φύλλο στο $\mathcal{T}'$ περιέχει *πλήρως* $k$ φύλλα από το $\mathcal{T}$. Σε αυτή τη περίπτωση, ένα φύλλο στο $\mathcal{T}'$ μπορεί *μερικώς* να περιέχει φύλλα από το $\mathcal{T}$. Ένα φύλλο στο $\mathcal{T}'$ περιέχει *πλήρως* ένα φύλλο από το $\mathcal{T}$ όταν όλα τα αντικείμενα δεδομένων που περιέχει το $\mathcal{T}'$ φύλλο ανήκουν στο $\mathcal{T}$ φύλλο. Διαφορετικά, ένα φύλλο στο $\mathcal{T}'$ *μερικώς* περιέχει ένα φύλλο από το $\mathcal{T}$ όταν το $\mathcal{T}'$ φύλλο περιέχει ένα υποσύνολο των αντικειμένων δεδομένων από το $\mathcal{T}$ φύλλο.

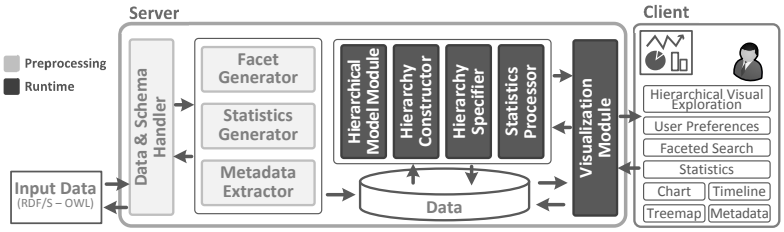
Ένα παράδειγμα αυτής της περίπτωσης φαίνεται στο Σχήμα 4.6e που απεικονίζει ένα $\mathcal{T}'$ που προκύπτει από το $\mathcal{T}$ που παρουσιάζεται στο Σχήμα 4.6a. Το $d$ φύλλο του $\mathcal{T}'$ περιέχει πλήρως τα φύλλα $h$, $i$ του $\mathcal{T}$ και μερικώς το φύλλο $k$ για το οποίο η τιμή 35 ανήκει σε διαφορετικό φύλλο (δηλαδή στο $e$).

### 4.1.3 Το Πλαίσιο SynopsViz

#### 4.1.3.1 Παρουσίαση Πλαισίου

Η αρχιτεκτονική του *rdf:SynopsViz* παρουσιάζεται στο Σχήμα 4.11, ενώ στο Σχήμα 4.8 παρουσιάζεται η βασική γραφική διεπαφή της εφαρμογής και στο Σχήμα 4.9 παραδείγματα οπτικοποίησης.

Το σενάριό μας περιλαμβάνει τρία κύρια μέρη: την γραφική διεπαφή του χρήστη, το πλαίσιο rdf:SynopsViz, και τα δεδομένα εισόδου. Το μέρος του χρήστη, αντιστοιχεί στην πρόσοψη (front-end) του πλαισίου, προσφέροντας πολλές λειτουργίες για τους τελικούς χρήστες (π.χ., στατιστική ανάλυση, αναζήτηση με όψεις, κλπ). Το rdf:SynopsViz δέχεται ως *Δεδομένα Εισόδου (Input Data)* RDF δεδομένα. Προαιρετικά, OWL-RDF/S λεξιλόγια ή οντολογίες που περιγράφουν τα δεδομένα εισόδου μπορούν να φορτωθούν. Στη συνέχεια, θα περιγράψουμε τα βασικά συστατικά του πλαισίου rdf:SynopsViz.




**Σχήμα 4.7:** Η αρχιτεκτονική του πλαισίου rdf:SynopsViz

Στην φάση της προ-επεξεργασίας, το *Data and Schema Handler* αναλύει τα δεδομένα εισόδου και εξάγει (inferes) πληροφορίες σχήματος. Το *Facets Generator* παράγει τις όψεις για τις κλάσεις και τις ιδιότητες των δεδομένων εισόδου. Το *Statistics Generator* υπολογίζει διάφορα στατιστικά στοιχεία σχετικά με το σχήμα, τον γράφο και τα στιγμιότυπα του συνόλου δεδομένων εισόδου. Το *Metadata Extractor* συλλέγει μεταδεδομένα τα οποία μπορούν να χρησιμοποιηθούν για την αξιολόγηση της ποιότητας των δεδομένων. Το *Hierarchical Model Module* υιοθετεί το ιεραρχικό μας μοντέλο και αποθηκεύει τα αρχικά δεδομένα εμπλουτισμένα με τις πληροφορίες που υπολογίζεται κατά το στάδιο της προ-επεξεργασίας.

Κατά τη διάρκεια της εκτέλεσης εμπλέκονται τα παρακάτω τμήματα. Το *Hierarchy Specifier* είναι υπεύθυνο για τη διαχείριση των παραμέτρων της ιεραρχίας (π.χ., τον αριθμός των επίπεδων, τον αριθμός των κόμβων ανά επίπεδο) και την παροχή αυτών των πληροφοριών στο Hierarchy Constructor. Το *Hierarchy Constructor* εφαρμόζει το μοντέλο της ιεραρχίας. Με βάση τις επιλεγμένες όψεις και τις παραμέτρους της ιεραρχίας: καθορίζει τις ομάδες της ιεραρχίας και τις τριπλέτες που περιέχονται σε αυτές και υπολογίζει στατιστικά στοιχεία σχετικά με το περιεχόμενό των ομάδων (π.χ., το εύρος, τη διακύμανση, κλπ). Το *Visualization Module* επιτρέπει την αλληλεπίδραση μεταξύ του χρήστη και του πλαισίου, παρέχοντας πολλαπλές λειτουργίες (π.χ., πλοήγηση, φιλτράρισμα, αναζήτηση) πάνω στα οπτικοποιημένα δεδομένα.

#### 4.1.3.2 Υλοποίηση

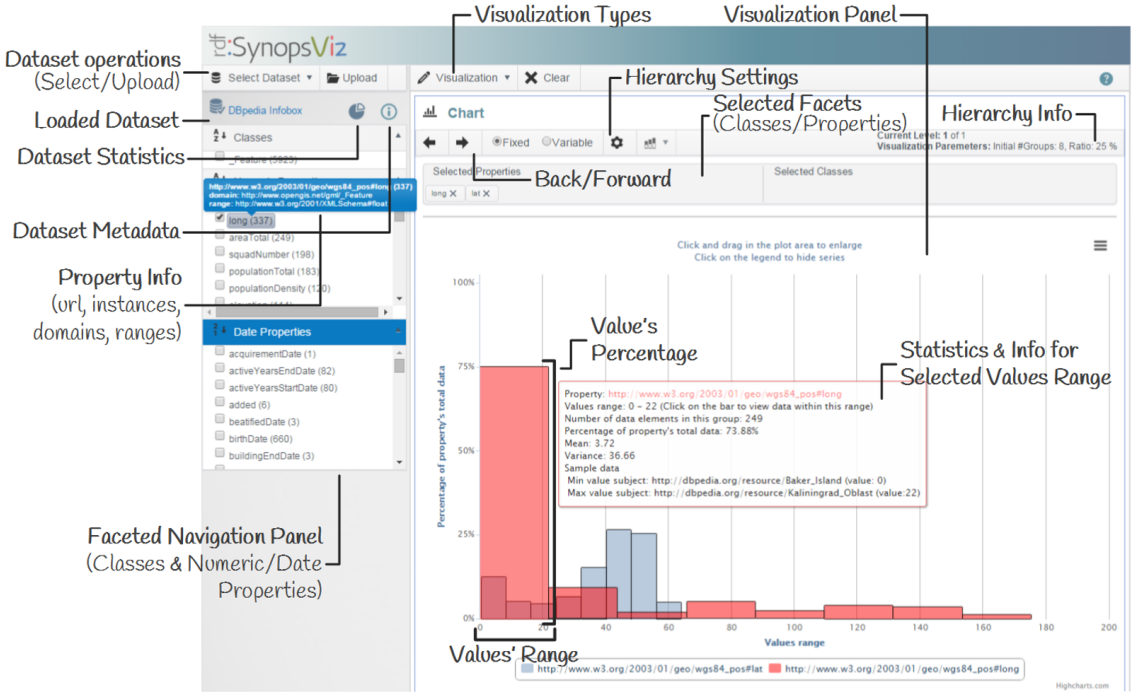
Το *rdf:SynopsViz* έχει υλοποιηθεί πάνω σε διάφορα εργαλεία ανοικτού κώδικα και βιβλιοθήκες. Όσον αφορά τις βιβλιοθήκες οπτικοποίηση, χρησιμοποιούμε την Highcharts[9], για γραφήματα περιοχής (Area charts) καθώς και χρονοδιάγραμμα (Timeline) και τα Google Charts[10] για δεντρικό χάρτη (treemap) και πίτες γραφημάτων (pie charts). Επιπλέον, έχουμε χρησιμοποιήσει το πλαίσιο Jena[11] για την διαχείριση και την αποθήκευση των RDF δεδομένων.

Μια πρωτότυπη εφαρμογή διαδικτύου του *rdf:SynopsViz* είναι διαθέσιμη στο `http://synopsviz.imis.athena-innovation.gr`. Επίσης, ένα βίντεο που παρουσιάζει το σενάριο που περιγράφεται παρακάτω είναι διαθέσιμο στο `http://youtu.be/8v-He1U4oxs`.

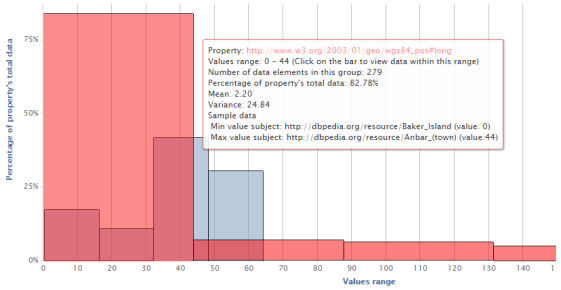
### 4.1.4 Πειραματική Ανάλυση

Σε αυτή την παράγραφο παρουσιάζουμε την αξιολόγηση της προσέγγιση μας. Στην Ενότητα 4.1.4.1, παρουσιάζουμε το σύνολο δεδομένων και την πειραματική διάταξη.

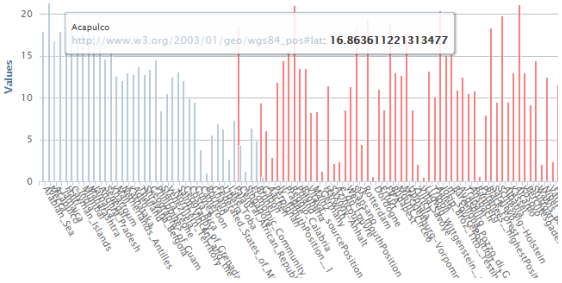
[9] www.highcharts.com
[10] developers.google.com/chart
[11] jena.apache.org



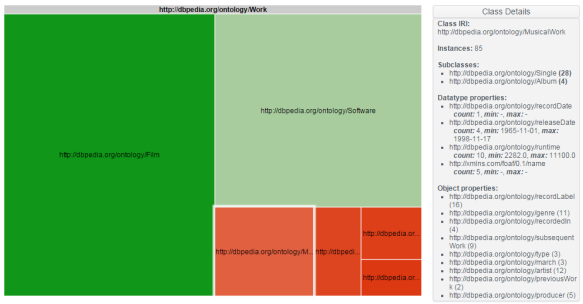

**Σχήμα 4.8:** Η βασική διεπαφή του πλαισίου rdf:SynopsViz



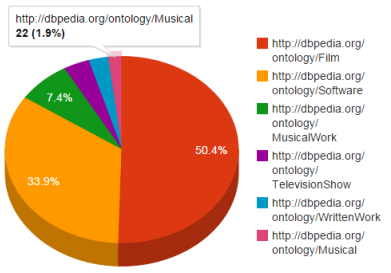

(α′) Γράφημα Περιοχής (Area chart)




(β′) Γράφημα Στηλών (Column chart)



(γ′) Γράφημα Δεντρικού Χάρτη (Treemap chart)




(δ′) Γράφημα Πίτας (Pie chart)

**Σχήμα 4.9:** Παραδείγματα οπτικοποίησης

Έπειτα, στην Ενότητα 4.1.4.2 παρουσιάζουμε τα αποτελέσματα επίδοσης και στην Ενότητα 4.1.4.3 την αξιολόγηση με χρήστες που πραγματοποιήσαμε.

#### 4.1.4.1 Πειραματική Διάταξη

Στην αξιολόγηση μας, χρησιμοποιούμε το γνωστό σύνολο δεδομένων *DBpedia* 2014. Ειδικότερα, χρησιμοποιούμε το *Mapping-based Properties (cleaned)* dataset[12] το οποίο περιέχει υψηλής ποιότητας δεδομένα, εξαγόμενα από τα Wikipedia Infoboxes. Αυτό το σύνολο δεδομένων περιέχει 33.1M triples και περιλαμβάνει ένα μεγάλο αριθμό αριθμητικών και χρονικών ιδιοτήτων ποικίλων μεγεθών. Το σύστημα αποθήκευσης (backend system) φιλοξενείται σε ένα server με τετραπύρηνη CPU στα 2GHz και 8GB RAM που τρέχει σε Windows Server 2008. Ως πελάτης, χρησιμοποιούμε ένα laptop με i5 CPU στα 2.5GHz με 4G RAM, που τρέχει σε Windows 7, Firefox 38.0.1 και με σύνδεση δικτύου ADSL2+ . Ακόμα, στην αξιολόγηση χρήστη, ο πελάτης είναι εφοδιασμένος με οθόνη 24″ (1920×1200).

#### 4.1.4.2 Αξιολόγηση Επίδοσης

Σε αυτή την παράγραφο, εξετάζουμε την επίδοση του προτεινόμενου μοντέλου, καθώς και την συμπεριφορά του εργαλείου μας, σχετικά με το χρόνο δημιουργίας και τον χρόνο απόκρισης αντίστοιχα.

##### 4.1.4.2.1 Περιβάλλον

Για να μελετήσουμε την επίδοση, οπτικοποιείται ένας αριθμός αριθμητικών και χρονικών ιδιοτήτων χρησιμοποιώντας τις ιεραρχικές προσεγγίσεις HETree-C/R και την μη-ιεραρχική προσέγγιση FLAT.

Διαλέγουμε μια ομάδα από κάθε τύπο ιδιοτήτων. Κάθε ομάδα περιέχει 15 ιδιότητες με ποικίλα μεγέθη, ξεκινώντας, από μικρές ιδιότητες που έχουν 50-100 triples μέχρι τις μεγαλύτερες ιδιότητες. Στη πειραματική μελέτη, για κάθε από τις τρεις προσεγγίσεις, μετράμε τον χρόνο απόκρισης. Επιπλέον, για τις δύο ιεραρχικές προσεγγίσεις μετράμε ακόμα τον χρόνο που απαιτείται για τη δημιουργία του HETree.

##### 4.1.4.2.2 Αποτελέσματα

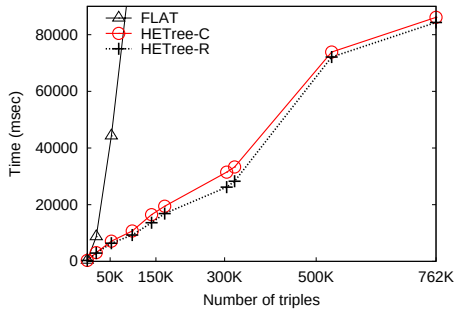
Ο Πίνακας 4.2 παρουσιάζει τα αποτελέσματα της αξιολόγησης σχετικά με τις αριθμητικές (πάνω μισό) και χρονικές (κάτω μισό) ιδιότητες. Οι ιδιότητες ταξινομούνται σε αύξουσα σειρά βάσει του αριθμού των triples. Για κάθε ιδιότητα, ο πίνακας περιέχει τον αριθμό των triples, τα χαρακτηριστικά της δημιουργημένης HETree δομής (δηλαδή, αριθμός φύλλων, βαθμός, ύψος, και αριθμός κόμβων) καθώς και ο χρόνος δημιουργίας και ο χρόνος απόκρισης κάθε προσέγγισης. Οι παρούσες μετρήσεις χρόνου είναι οι μέσες τιμές από 50 εκτελέσεις.

Σχετικά με τη σύγκριση σε σχέση με το HETree και με το FLAT, η FLAT προσέγγιση δεν μπορεί να οπτικοποιήσει αποτελέσματα για τις ιδιότητες που έχουν πάνω από 305K triples, που υποδεικνύονται στις τελευταίες σειρές τόσο για τις αριθμητικές όσο και για τις χρονικές ιδιότητες με '–' στον χρόνο απόκρισης της FLAT. Για τις υπόλοιπες ιδιότητες, μπορούμε να παρατηρήσουμε ότι οι HETree προσεγγίσεις έχουν καλύτερες αποδόσεις από την FLAT σε όλες τις περιπτώσεις, ακόμα και στη μικρότερη

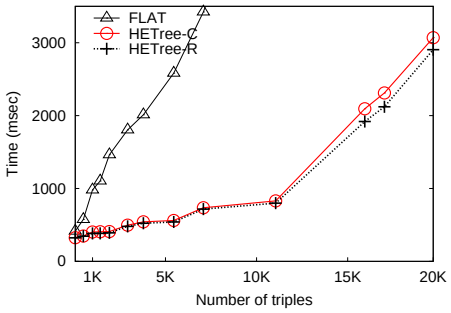
[12]downloads.dbpedia.org/2014/en/mappingbased_properties_ cleaned_en.nt.bz2

**Πίνακας 4.2:** Αποτελέσματα επίδοσης

| Property (#Triples) | Tree Characteristics | | | | HETree-C | | HETree-R | | FLAT |
|---|---|---|---|---|---|---|---|---|---|
| | #Leaves | Degree | Height | #Nodes | Construction Time (msec) | Response Time (msec) | Construction Time (msec) | Response Time (msec) | Response Time (msec) |
| **Numeric Properties** | | | | | | | | | |
| rankingWins (50) | 9 | 3 | 2 | 13 | 5 | 324 | 1 | 323 | 415 |
| distanceToBelfast (104) | 9 | 3 | 2 | 13 | 7 | 337 | 4 | 329 | 419 |
| waistSize (241) | 16 | 4 | 2 | 21 | 10 | 346 | 9 | 336 | 440 |
| fileSize (492) | 27 | 3 | 3 | 40 | 18 | 347 | 16 | 345 | 575 |
| hsvCoordinateValue (995) | 81 | 3 | 4 | 121 | 74 | 403 | 50 | 383 | 980 |
| lineLength (1,923) | 81 | 3 | 4 | 121 | 77 | 409 | 55 | 391 | 1,463 |
| powerOutput (5,453) | 243 | 3 | 5 | 364 | 234 | 560 | 217 | 540 | 2,583 |
| width (11,049) | 729 | 3 | 6 | 1,093 | 506 | 830 | 467 | 799 | 6,135 |
| numberOfPages (21,743) | 729 | 3 | 6 | 1,093 | 2,888 | 3,219 | 2,403 | 2,722 | 12,669 |
| inseeCode (36,780) | 2,187 | 3 | 7 | 3,280 | 4,632 | 4,962 | 4,105 | 4,436 | 19,119 |
| areaWater (40,564) | 2,187 | 3 | 7 | 3,280 | 4,945 | 5,134 | 5,274 | 5,457 | 29,538 |
| populationDensity (52,572) | 2,187 | 3 | 7 | 3,280 | 6,803 | 7,127 | 6,080 | 6,404 | 44,262 |
| areaTotal (140,408) | 6,561 | 3 | 8 | 9,841 | 16,158 | 16,482 | 13,298 | 13,627 | 219,018 |
| populationTotal (304,522) | 19,683 | 3 | 9 | 29,524 | 31,141 | 31,473 | 25,866 | 26,196 | 1,523,675 |
| lat (533,900) | 19,683 | 3 | 9 | 29,524 | 73,528 | 73,862 | 71,784 | 72,106 | — |
| **Temporal Properties** | | | | | | | | | |
| retired (155) | 9 | 3 | 2 | 13 | 8 | 330 | 4 | 327 | 425 |
| endDate (341) | 27 | 3 | 3 | 40 | 17 | 339 | 16 | 339 | 468 |
| lastAirDate (704) | 64 | 4 | 3 | 85 | 34 | 359 | 30 | 359 | 853 |
| buildingStartDate (1,415) | 81 | 3 | 4 | 121 | 73 | 406 | 53 | 384 | 1,103 |
| latestReleaseDate (2,925) | 243 | 3 | 5 | 364 | 162 | 496 | 146 | 480 | 1,804 |
| orderDate (3,788) | 243 | 3 | 5 | 364 | 210 | 542 | 195 | 523 | 2,011 |
| decommissioningDate (7,082) | 243 | 3 | 5 | 364 | 405 | 735 | 383 | 717 | 3,423 |
| shipLaunch (15,938) | 729 | 3 | 6 | 1,093 | 1,772 | 2,094 | 1,595 | 1,919 | 6,935 |
| completionDate (17,017) | 729 | 3 | 6 | 1,093 | 1,987 | 2,311 | 1,793 | 2,121 | 7,814 |
| foundingDate (19,694) | 729 | 3 | 6 | 1,093 | 2,745 | 3,069 | 2,583 | 2,905 | 8,699 |
| added (44,227) | 2,187 | 3 | 7 | 3,280 | 5,912 | 5,943 | 6,244 | 6,265 | 33846 |
| activeYearsStartDate (98,160) | 6,561 | 3 | 8 | 9,841 | 10,368 | 10,702 | 8,952 | 9,282 | 107,587 |
| releaseDate (169,156) | 6,561 | 3 | 8 | 9,841 | 19,122 | 19,451 | 16,526 | 16,856 | 950,545 |
| deathDate (321,883) | 19,683 | 3 | 9 | 29,524 | 32,990 | 33,313 | 27,936 | 28,271 | — |
| birthDate (761,830) | 59,049 | 3 | 10 | 88,573 | 85,797 | 86,120 | 83,982 | 84,314 | — |




(α′) Ιδιότητες: από 50 μέχρι 762K τριπλες




(β′) Ιδιότητες: από 50 μέχρι 20K τριπλες

**Σχήμα 4.10:** Χρόνος Απόκρισης σε σχέση με τον αριθμό τριπλες

ιδιότητα (δηλαδή *rankingWin*, 50 triples). Καθώς το μέγεθος των ιδιοτήτων αυξάνει, η διαφορά μεταξύ της HETree προσέγγισης και της FLAT αυξάνει επίσης. Πιο αναλυτικά, για ιδιότητες που έχουν περισσότερες από 1.400 triples (δηλαδή, οι αριθμητικές ιδιότητες που είναι μεγαλύτερες από τη *hsvCoordinateValue* -5η σειρά-, και οι χρονικές ιδιότητες που είναι μεγαλύτερες από τη *lastAirDate* -4η σειρά-), οι HETree προσεγγίσεις έχουν καλύτερες αποδόσεις απο τις FLAT κατά μία τάξη μεγέθους περίπου. Τελικά, για την μεγαλύτερη ιδιότητα που μπορεί να διαχειριστεί η FLAT (δηλαδή, *populationTotal*,305K triples), η διαφορά μεταξύ της HETree και της FLAT είναι περίπου δύο τάξεις μεγέθους.

Το Σχήμα 4.10 συνοψίζει τα αποτελέσματα του Πίνακα 4.2, παρουσιάζοντας τον χρόνο απόκρισης για όλες τις μεθόδους σε συνάρτηση με τον αριθμό των triples. Συγκεκριμένα, το Σχήμα 4.10α περιλαμβάνει όλα τα μεγέθη ιδιοτήτων, ενώ το Σχήμα 4.10β επικεντρώνεται στις ιδιότητες που έχουν μέχρι 20K triples. Παρατηρούμε επίσης εδώ ότι το HETree-R λειτουργεί ελαφρώς καλύτερα από το HETree-C. Επιπλέον, από το Σχήμα 4.10β μπορούμε να δούμε ότι για μέχρι 10K triples η επίδοση της HETree προσέγγισης είναι περίπου η ίδια. Μπορούμε επίσης να παρατηρήσουμε τη σημαντική διαφορά μεταξύ της FLAT και της HETree προσέγγισης.

#### 4.1.4.3 Μελέτη Χρηστών

Σε αυτή την ενότητα παρουσιάζουμε την χρηστική αξιολόγηση του εργαλείου με πραγματικούς χρήστες, εξετάζοντας τρεις προσεγγίσεις: δύο ιεραρχικές και την FLAT.

##### 4.1.4.3.1 Εργασίες

Σε αυτή την ενότητα περιγράφουμε τους διαφορετικούς τύπους εργασιών που χρησιμοποιούνται στη διαδικασία της χρηστικής αξιολόγησης.

**Τύπος 1 [Βρες πόρους με συγκεκριμένη τιμή]:** Αυτός ο τύπος εργασίας απαιτεί πόρους που έχουν τιμή $v$ (ως αντικείμενα). Για αυτό το είδος εργασίας, ορίζουμε την εργασία T1 επιλέγοντας μια τιμή $v$ που αντιστοιχεί σε 5 πόρους.

**Τύπος 2 [Βρες πόρους με εύρος τιμών]:** Αυτός ο τύπος εργασίας απαιτεί πόρους που έχουν τιμή μεγαλύτερη από $v_{min}$ και μικρότερη από $v_{max}$. Ορίζουμε δύο εργασίες αυτού του είδους, επιλέγοντας διαφορετικούς συνδυασμούς των $v_{min}$ και $v_{max}$ τιμών, έτσι ώστε να ορίζονται εργασίες με διαφορετικούς αριθμούς πόρων. Συγκεκριμένα, η πρώτη εργασία, που ονομάζεται T2.1, εξετάζουμε τις τιμές $v_{min}$ και $v_{max}$ έτσι

ώστε να περιλαμβάνεται μια μικρή ομάδα πόρων (περίπου 10), ενώ η δεύτερη εργασία, T2.2, εξετάζει μια μεγαλύτερη ομάδα (περίπου 50) πόρων.

**Τύπος 3 [Σύγκρινε κατανομές]:** Αυτός ο τύπος εργασίας απαιτεί από τον συμμετέχοντα να αναγνωρίσει εάν εμφανίζονται περισσότεροι πόροι πάνω ή κάτω από μια δοθείσα τιμή $v$. Για αυτόν τον τύπο, ορίζουμε την εργασία T3, επιλέγοντας την τιμή $v$ κοντά στη μέση.

#### 4.1.4.3.2 Περιβάλλον

Προκειμένου να μελετήσουμε το αποτέλεσμα σε σχέση με το μέγεθος των ιδιοτήτων στα επιλεγμένα εργασίες, έχουμε επιλέξει δύο ιδιότητες διαφορετικών μεγεθών. Συγκεκριμένα, επιλέξαμε την *hsvCoordinateHue* αριθμητική ιδιότητα που περιέχει 970 triples, αναφέρεται ως *Small*, και η *maximumElevation* αριθμητική ιδιότητα, που περιέχει 37.936 triples, αναφέρεται ως *Large*.

Στην αξιολόγησή μας, πήραν μέρος 10 συμμετέχοντες. Οι συμμετέχοντες ήταν μεταπτυχιακοί φοιτητές και ερευνητές επιστήμης υπολογιστών. Κατά τη διάρκεια της αξιόλογης, κάθε συμμετέχοντας εκτέλεσε τις τέσσερις εργασίες που περιγράφηκαν προηγουμένως, χρησιμοποιώντας όλες τις προσεγγίσεις (δηλαδή, HETree-C/R και FLAT), τόσο για τις μικρές όσο και τις μεγάλες ιδιότητες.

Κατά τη διάρκεια της αξιολόγησης ο εκπαιδευτής μέτρησε τον χρόνο που απαιτούνταν για να τελειώσει κάθε συμμετέχοντας τις εργασίες, καθώς και τον αριθμό λάθος απαντήσεων. Ο Πίνακας 4.3 παρουσιάζει τον μέσο χρόνο που απαιτήθηκε από τους συμμετέχοντας για να ολοκληρώσουν τη κάθε εργασία. Ο πίνακας περιέχει τις μετρήσεις για όλες τις προσεγγίσεις, και για τις δύο ιδιότητες. Αν και αναγνωρίζουμε ότι ο αριθμός των συμμετεχόντων στην αξιολόγηση μας είναι μικρός, έχουμε υπολογίσει τη στατιστική σημασία των αποτελεσμάτων. Ουσιαστικά, για κάθε ιδιότητα, η $p$-τιμή του κάθε εργασίας παρουσιάζεται στην τελευταία στήλη. Η $p$-τιμή υπολογίζεται χρησιμοποιώντας το one-way repeated measures ANOVA. Τέλος, τα αποτελέσματα σχετικά με τον αριθμό των έργων που δεν απαντήθηκαν σωστά παρουσιάζονται στον Πίνακα 4.4.

#### 4.1.4.3.3 Αποτελέσματα

**Εργασία T1.** Σχετικά με την πρώτη εργασία, όπως μπορούμε να παρατηρήσουμε από τον Πίνακα 4.3, οι HETree προσεγγίσεις έχουν καλύτερες επιδόσεις από την FLAT, σε όλα τα μεγέθη ιδιοτήτων. Σημειώνουμε ότι τα χρονικά αποτελέσματα στο T1 είναι στατιστικά σημαντικά ($p < 0.01$).

Όπως αναμενόταν, όλες οι προσεγγίσεις απαιτούν περισσότερο χρόνο για τις μεγάλες ιδιότητες (Large property) σε σχέση με τις μικρές (Small). Αυτό το πρόβλημα στη FLAT προκαλείται από τον μεγάλο αριθμό από πόρους τους οποίους πρέπει να εξετάσουν οι συμμετέχοντες, μέχρι να υποδείξουν την απαιτούμενη τιμή. Από την άλλη πλευρά, στη HETree, το πρόβλημα προκαλείται από το μεγαλύτερο αριθμό επιπέδων που έχει η ιεραρχία της μεγάλης ιδιότητας Large property. Έτσι, οι συμμετέχοντες πρέπει να εκτελέσουμε περισσότερες λειτουργίες drill-down και να εξετάσουν περισσότερες ομάδες αντικειμένων, μέχρι να φτάσουν τους LD πόρους.

**Εργασία T2.1.** Στην επόμενη εργασία, όπου οι συμμετέχοντες έπρεπε να υποδείξουν μια μικρή ομάδα πόρων με μεγάλο εύρος τιμών, η επίδοση της FLAT είναι πολύ

**Πίνακας 4.3:** Μέσος Χρόνος Ολοκλήρωσης Εργασιών (sec)

| | Small Property | | | | Large Property | | | |
|---|---|---|---|---|---|---|---|---|
| | FLAT | HETree-C | HETree-R | $p$ | FLAT | HETree-C | HETree-R | $p$ |
| **T1** | 54 | 29 | 28 | ★★ | 85 | 52 | 47 | ★★ |
| **T2.1** | 63 | 57 | 64 | ◆ | 74 | 60 | 69 | ★ |
| **T2.2** | 120 | 69 | 74 | ★★ | 128 | 72 | 77 | ★★ |
| **T3** | 262 | 41 | 40 | ★★ | — | 64 | 62 | — |

★★ ($p < 0.01$) ★ ($p < 0.05$) ◆ ($p > 0.05$)

**Πίνακας 4.4:** Ποσοστό σφάλματος (%)

| | Small Property | | | | Large Property | | | |
|---|---|---|---|---|---|---|---|---|
| | FLAT | HETree-C | HETree-R | $p$ | FLAT | HETree-C | HETree-R | $p$ |
| **T1** | 0 | 0 | 0 | ◆ | 0 | 0 | 0 | ◆ |
| **T2.1** | 0 | 0 | 0 | ◆ | 0 | 0 | 0 | ◆ |
| **T2.2** | 20 | 0 | 0 | ◆ | 20 | 0 | 10 | ◆ |
| **T3** | 70 | 0 | 0 | ★★ | — | 0 | 0 | — |

★★ ($p < 0.01$) ★ ($p < 0.05$) ◆ ($p > 0.05$)

κοντινή με της HETree, ειδικά με τη Small property (Πίνακας 4.3). Επιπλέον, μπορούμε να παρατηρήσουμε ότι η HETree-C προσέγγιση έχει σχετικά καλύτερη επίδοση από τη HETree-R. Τέλος, σχετικά με τη στατιστική σημαντικότητα των αποτελεσμάτων, στις μικρές ιδιότητες έχουμε ότι $p > 0.05$, ενώ στις μεγάλες έχουμε ότι $p < 0.005$.

**Εργασία T2.2.** Σε αυτή την εργασία οι συμμετέχοντες πρέπει να υποδείξουν μία μεγαλύτερη ομάδα πόρων (σχετικά με την προηγούμενη εργασία) με δεδομένο εύρος τιμών. Οι HETree προσεγγίσεις έχουν εμφανώς καλύτερη επίδοση από τη FLAT προσέγγιση με στατιστική σημαντικότητα ($p < 0.01$), ενώ παρόμοια αποτελέσματα παρατηρούνται σε όλες τις ιδιότητες.

**Εργασία T3.** Στο τελευταία εργασία, ζητήθηκε από τους συμμετέχοντες να βρουν ποιά από τα δύο εύρη περιέχουν περισσότερους πόρους. Όπως αναμενόταν, ο Πίνακας 4.3 δείχνει ότι η προσέγγιση HETree έχει εμφανώς καλύτερη επίδοση από την FLAT προσέγγιση με στατιστική σημαντικότητα στη Small property. Σχετικά με τη Large property, όπως αναμενόταν, ήταν αδύνατο για τους συμμετέχοντες να επιλύσουν την εργασία με την FLAT, αφού αυτό θα απαιτούσε να μετρηθούν 19K πόρους. Συνεπώς, κανείς από τους συμμετέχοντες δεν ολοκλήρωσε την εργασία χρησιμοποιώντας τη FLAT (συμβολίζεται με '–' στον Πίνακα 4.3), αν αναλογιστούμε το όριο 5 λεπτών που ορίστηκε για στην εργασία.

### 4.1.5 Ρελατεδ Ωορκ

Σε αυτή την ενότητα γίνεται μια ανασκόπηση έργων που σχετίζονται με την προσέγγιση μας στην οπτικοποίηση και διερεύνηση στον ιστό δεδομένων ή Web of Data (WoD).

Στον Πίνακα 4.5 παρέχουμε μια σύνοψη και συγκρίνουμε αρκετά συστήματα οπτικοποίησης που έχουν παρόμοια χαρακτηριστικά με το δικό μας σύστημα, το SynopsViz.

**Πίνακας 4.5:** Επισκόπηση Συστημάτων

| System | WoD | Hierarchical | Data Types* | Vis. Types** | Statistics | Recomm. | Incr. | Preferences | Domain | App. Type |
|---|---|---|---|---|---|---|---|---|---|---|
| **Rhizomer** [89] | ✓ | | N, T, S, H, G | C, M, T, TL | | ✓ | | | generic | Web |
| **Payola** [216] | ✓ | | N, T, S, H, G | C, CI, G, M, T, TL, TR | | | | | generic | Web |
| **LDVM** [88] | ✓ | | S, H, G | B, M, T, TR | | ✓ | | | generic | Web |
| **Vis Wizard** [328] | ✓ | | N, T, S | B, C, M, P, PC, SG | | ✓ | | ✓ | generic | Web |
| **LDVizWiz** [37] | ✓ | | S, H, G | M, P, TR | | ✓ | | | generic | Web |
| **LinkDaViz** [322] | ✓ | | N, T, S | B, C, S, M, P | | ✓ | | ✓ | generic | Web |
| **VizBoard** [339] | ✓ | | N, H | C,S, T | | ✓ | | ✓ | generic | Web |
| **SemLens** [180] | ✓ | | N | S | | | | ✓ | generic | Web |
| **LODeX** [51] | ✓ | | G | G, M, P | ✓ | | | | generic | Web |
| **LODWheel** [313] | ✓ | | N, S, G | C, G, M, P | | | | | generic | Web |
| **RelFinder** [179] | ✓ | | G | G | | | | ✓ | generic | Web |
| **Fenfire** [177] | ✓ | | G | G | | | | | generic | Desktop |
| **Lodlive** [91] | ✓ | | G | G | | | | ✓ | generic | Web |
| **IsaViz** [276] | ✓ | | G | G | | | | ✓ | generic | Desktop |
| **graphVizdb** [65] | ✓ | | G | G | | | | ✓ | generic | Web |
| **ViCoMap** [286] | ✓ | | N, T, S | M | ✓ | ✓ | | ✓ | generic | Web |
| **EDT** [249] | | | N, T, H | C, CM, T, SP | ✓ | | | | OLAP | Desktop |
| **Polaris** [311] | | ✓ | N, T, S, H | C, M, S | ✓ | | ✓ | ✓ | OLAP | Desktop |
| **XmdvTool** [343, 158] | | ✓ | N | DS, PC, S, ST | | | | | generic | Desktop |
| **GrouseFlocks** [31] | | ✓ | G | G | | | ✓ | ✓ | generic | Desktop |
| **GMine** [195] | | ✓ | G | G | | | | | generic | Desktop |
| **Gephi** [44] | | ✓ | G | G | ✓ | | | ✓ | generic | Desktop |
| **CGV** [327] | | ✓ | G | G | | | ✓ | ✓ | generic | Desktop |
| **SynopsViz** | ✓ | ✓ | N, T, H | C, P, T, TL [1] | ✓ | ✓ | ✓ | ✓ | generic | Web |

* N: Numeric, T: Temporal, S: Spatial, H: Hierarchical (tree), G: Graph (network)

** B: *bubble chart*, C: *chart*, CI: *circles*, CM: *colormap*, DS: *dimensional stacking*, G: *graph*, M: *map*, P: *pie*, PC: *parallel coordinates*, S: *scatter*, SG: *streamgraph*, SP: *solarplot*, ST: *star glyphs*, T: *treemap*, TL: *timeline*, TR: *tree*

[1] The HETree model is not restricted to these visualization types.

#### 4.1.5.1 Συστήματα οπτικοποίησης και διερεύνησης

Ένας μεγάλος αριθμός εργασιών που εξετάζουν ζητήματα σχετικά με την οπτική διερεύνηση και ανάλυση στο WoD έχουν προταθεί στην βιβλιογραφία [123, 251, 26]. Ακολούθως, ταξινομούμε αυτές τις εργασίες στις επόμενες κατηγορίες: (1) Γενικά συστήματα οπτικοποίησης (Generic visualization systems), (2) Domain, vocabulary & device-specific συστήματα οπτικοποίησης, (3) Graph-based συστήματα οπτικοποίησης, και (4) Συστήματα οπτικοποίησης οντολογιών (Ontology visualization systems)

##### 4.1.5.1.1 Γενικά Συστήματα Οπτικοποίησης

Στα πλαίσια της οπτικής διερεύνησης στον WoD, υπάρχει ένας μεγάλος αριθμός πλαισίων γενικής οπτικοποίησης (generic visualization frameworks), που προσφέρουν ένα μεγάλο εύρος τύπων οπτικοποίησης και λειτουργιών. Ακολούθως, παραθέτουμε τα πιο γνωστά συστήματα σε αυτήν την κατηγορία.

Το *Rhizomer* [89] παρέχει WoD διερεύνηση βασιζόμενο σε overview, zoom και filter workflow. Το *VizBoard* [339, 340] είναι ένα information visualization workbench για τον WoD φτιαγμένο πάνω σε μια mashup πλατφόρμα. Το *Payola* [216] είναι ένα γενικό πλαίσιο για WoD οπτικοποίηση και ανάλυση. Το Μοντέλο Οπτικοποίησης Διασυνδεδεμένων Δεδομένων ή *Linked Data Visualization Model* (LDVM) [88] παρέχει μία αφηρημένη διαδικασία οπτικοποίησης για WoD σύνολα δεδομένων. Το LDVM επιτρέπει τη σύνδεση με δυναμικό τρόπο διαφορετικών συνόλων δεδομένων με διάφορα είδη οπτικοποίησης. Το *Vis Wizard* [328] είναι ένα Web-based σύστημα οπτικοποίησης, το οποίο χρησιμοποιεί σημασιολογία δεδομένων προκειμένου να απλοποιήσει τη διαδικασία της εγκατάστασης των οπτικοποιήσεων. Ομοίως, το *Linked Data Visualization Wizard* (LDVizWiz) [37] παρέχει έναν ημιαυτόματο τρόπο παραγωγής μιας πιθανής οπτικοποίησης για WoD σύνολα δεδομένων. Σε παρόμοια πλαίσια, το *LinkDaViz* [322] βρίσκει τις κατάλληλες οπτικοποιήσεις για δεδομένο τμήμα ενός συνόλου δεδομένων. Το *Balloon Synopsis* [298] παρέχει έναν WoD visualizer βασισμένο σε HTML και JavaScript. Το *SemLens* [180] είναι ένα οπτικό σύστημα που συνδυάζει διαγράμματα διασποράς (scatter plots) και σημασιολογικούς φακούς (semantic lenses), προσφέροντας οπτική ανακάλυψη συσχετίσεων και μοτίβων στα δεδομένα. Το *LODeX* [51] είναι ένα σύστημα που δημιουργεί μια αντιπροσωπευτική περίληψη μιάς WoD source. Το *LODWheel* [313] είναι ένα Web-based σύστημα οπτικοποίησης το οποίο συνδυάζει JavaScript βιβλιοθήκες (π.χ, MooWheel, JQPlot) προκειμένου να οπτικοποήσει RDF δεδομένα σε πίνακες και γραφήματα. Το *Hide the stack* [124] προτείνει μια προσέγγιση για την οπτικοποίηση του WoD για μέσους τελικούς χρήστες.

##### 4.1.5.1.2 Domain, Vocabulary & Device-specific Συστήματα Οπτικοποίησης

Σε αυτό το τμήμα, παρουσιάζουμε συστήματα που επικεντρώνονται σε ανάγκες οπτικοποίησης για συγκεκριμένους τύπους δεδομένων ή domains, RDF vocabularies ή συσκευές. Αρκετά συστήματα επικεντρώνονται στην οπτικοποίηση και στη διερεύνηση γεωχωρικών δεδομένων. Το *Map4rdf* [231] είναι ένα πολύπλευρο browsing tool που επιτρέπει RDF σύνολα δεδομένων να οπτικοποιούνται σε έναν OSM ή Google Map. Το *Facete* [307] είναι ένα εργαλείο διερεύνησης και οπτικοποίησης για SPARQL προσπελάσιμα δεδομένα (accessible data), προσφέροντας πολύπλευρες λειτουργίες φιλτραρίσματος. Το *SexTant* [54] και το *Spacetime* [335] επικεντρώνονται στην οπτικοποίηση

και στη διερεύνηση χρονικά μεταβαλόμενων γεωχωρικών δεδομένων. Ο *LinkedGeoData Browser* [306] είναι ένας πολύπλευρος browser και editor ο οποίος έχει αναπτυχθεί στα πλαίσια του LinkedGeoData project. Τέλος, στα ίδια πλαίσια, ο *DBpedia Atlas* [334] προσφέρει διερεύνηση σε DBpedia σύνολα δεδομένων χρησιμοποιώντας τα χωρικά δεδομένα του συνόλου δεδομένων. Μία ποικιλία εργαλείων επικεντρώνονται σε πολυδιάστατους WoD που μοντελοποιούνται με το Data Cube vocabulary. Ο *CubeViz* [146, 294] είναι ένας πολύπλευρος browser για τη διερεύνηση στατιστικών δεδομένων. Το *Payola Data Cube Vocabulary* [181] υιοθετεί τα LDVM στάδια [88] προκειμένου να οπτικοποιήσει RDF δεδομένα που περιγράφονται στο Data Cube vocabulary. Το *OpenCube Toolkit* [199] προσφέρει αρκετά εργαλεία σχετικά με τους στατικούς WoD. Επιπλέον, το *OpenCube Map View* προσφέρει διαδραστικές map-based οπτικοποιήσεις σε RDF data cubes βασισμένες στις γεωχωρικές διαστάσεις τους. Ο *Linked Data Cubes Explorer* (LDCE) [203] επιτρέπει στους χρήστες να διερευνούν και να αναλύουν στατιστικά σύνολα δεδομένων. Τέλος, το [273] προσφέρει αρκετές οπτικοποιήσεις χαρτών και πινάκων δημογραφικών, κοινωνικών και στατιστικών διασυνδεδεμένων κυβικών δεδομένων.

Σχετικά με τα device-specific συστήματα, η *DBpedia Mobile* [47] είναι μια location-aware εφαρμογή για κινητά για τη διερεύνηση και την οπτικοποίηση DBpedia πόρων. Η *Who's Who* [92] είναι μια εφαρμογή για τη διερεύνηση και την οπτικοποίηση πληροφοριών που επικεντρώνεται σε διάφορα ζητήματα που εμφανίζονται στα περιβάλλοντα των κινητών.

#### 4.1.5.1.3 Graph-based Συστήματα Οπτικοποίησης

Ένας μεγάλος αριθμός από συστήματα οπτικοποιεί WoD σύνολα δεδομένων υιοθετόντας μια *graph-based* (δηλαδή, node-link) προσέγγιση. Το *RelFinder* [179] είναι ένα Web-based εργαλείο που προσφέρει διαδραστική ανακάλυψη και οπτικοποίηση σχέσεων (δηλαδή, συνδέσεων) μεταξύ επιλεγμένων WoD πόρων. Το *Fenfire* [177] και το *Lodlive* [91] είναι εξερευνητικά εργαλεία που επιτρέπουν στους χρήστες να φυλλομετρούν στον WoD χρησιμοποιώντας διαδραστικά γραφήματα. Ξεκινώντας από ένα δεδομένο URI, ο χρήστης μπορεί να διερευνήσει τον WoD ακολουθώντας τα links. Το *IsaViz* [276] επιτρέπει στους χρήστες να μεγεθύνουν και να πλοηγηθούν σε ένα RDF γράφημα, και επίσης προσφέρει αρκετές 'edit' λειτουργίες (δηλαδή, διαγραφή/πρόσθεση/ μετονομασία κόμβων και ακμών). Στα ίδια πλαίσια, το *graphVizdb* [65] έχει φτιαχτεί πάνω σε χωρικές τεχνικές και τεχνικές βάσεων δεδομένων που προσφέρουν διαδραστική οπτικοποίηση σε πολύ μεγάλα (RDF) γραφήματα. Το *ZoomRDF* [353] χρησιμοποιεί έναν χωρικά βελτιστοποιημένο αλγόριθμο οπτικοποίησης προκειμένου να αυξήσει τον αριθμό των πόρων που μπορούν να απεικονιστούν. Το *Trisolda* [132] προτείνει μια ιεραρχική οπτικοποίηση RDF γραφήματος. Υιοθετεί τεχνικές ομαδοποίησης προκειμένου να συγχωνεύσει κόμβους γραφημάτων. Περισσότερες λεπτομέρειες σχετικά με την ιεραρχική οπτικοποίηση γραφημάτων μπορούν να βρεθούν στην Ενότητα 4.1.5.2. Η *Paged Graph Visualization* (PGV) [128] χρησιμοποιεί μια Ferris-Wheel προσέγγιση για να επιδείξει κόμβους με μεγάλο βαθμό. Ο *RDF graph visualizer* [296] υιοθετεί μια κομβό-κεντρική προσέγγιση για να οπτικοποιήσει RDF γραφήματα. Αντί να προσπαθήσει να οπτικοποιήσει ολόκληρο το γράφημα, ανακαλύπτονται κόμβοι ενδιαφέροντος (δηλαδή αρχικοί κόμβοι) ψάχνοντας ετικέτες κόμβων. Τότε ο χρήστης μπορεί διαδραστικά να πλοηγηθεί στο γράφημα. Τέλος, το *RDF-Gravity*[13] οπτικοποιεί τα RDF

[13] σεμωεβ.σαλζβυργρεσεαρση.ατ/αππς/ρδφ-γραιτψ

και τα OWL δεδομένα. Προσφέρει φιλτράρισμα, αναζήτηση με λέξεις-κλειδιά και σύνταξη της διάταξης του γραφήματος. Επίσης, οι κόμβοι μπορούν να απεικονιστούν με διαφορετικά χρώματα και σχήματα βασισμένοι στον RDF τύπο τους.

#### 4.1.5.1.4 Συστήμα Οπτικοποίησης Οντολογιών

Τα προβλήματα σχετιζομενα *οπτικοποίηση και διερεύνηση οντολογίας* έχουν μελετηθεί εκτεταμένα σε αρκετούς άλλους τομείς (π.χ. βιολογία, χημεία). Στα επόμενα, επικεντρωνόμαστε σε graph-based συστήματα οπτικοποίησης οντολογιών που έχουν αναπτυχθεί σε πλαίσια WoD [157, 138, 171, 227, 208]. Στα περισσότερα συστήματα, οι οντολογίες οπτικοποιούνται ακολουθώντας το node-link παράδειγμα [242, 241, 186, 264, 83, 151, 188, 233, 27, 223, 312].

Από την άλλη, οι *CropCircles* [342] χρησιμοποιούν μια προσέγγιση γεωμετρικής συγκράτησης (geometric containment approach), παρουσιάζοντας την ιεραρχία κλάσης ως ένα σύνολο ομόκεντρων κύκλων. Επιπλέον, υβριδικές προσεγγίσεις υιοθετούνται σε άλλα έργα. Το *Knoocks* [220] συνδυάζει containment-based και node-link προσεγγίσεις. Τέλος, το *OntoTrix* [40] και το *NodeTrix* [182] χρησιμοποιούν node-link αναπαραστάσεις και αναπαραστάσεις με πίνακες γειτνίασης (adjacency matrix).

### 4.1.5.2 Ιεραρχική Οπτική Διερεύνηση

Η ευρύτερη περιοχή οπτικοποίησης δεδομένων και πληροφοριών παρέχει ποικιλία προσεγγίσεων για ιεραρχική ανάλυση και παρουσίαση. Οι ιεραρχικές τεχνικές οπτικοποίησης έχουν ευρέως χρησιμοποιηθεί για την οπτικοποίηση μεγάλων γραφημάτων χρησιμοποιώντας το node-link παράδειγμα. Σε αυτές τις τεχνικές το γράφημα αποσυντίθεται επαναληπτικά σε μικρότερα υπογραφήματα για να σχηματίσει μια ιεραρχία αφηρημένων στρώσεων (hierarchy of abstraction layers). Στις περισσότερες περιπτώσεις, η ιεραρχία δημιουργείται χρησιμοποιώντας μεθόδους ομαδοποίησης και κατάτμησης (clustering and partitioning methods) [232, 32, 30, 18, 38, 44, 195, 327]. Σε άλλα έργα, η ιεραρχία ορίζεται με hub-based [235] και density-based [356] τεχνικές. Το *GrouseFlocks* [31] υποστηρίζει ad-hoc ιεραρχίες οι οποίες ορίζονται χειροκίνητα από τους χρήστες. Τέλος, υπάρχουν μερικές *edge bundling* τεχνικές οι οποίες ενώνουν ακμές γραφημάτων σε δέσμες. Οι ακμές πολλές φορές προσθέτονται με βάση τις τεχνικές ομαδοποίηση [159, 147, 274], τους βρόχους (mesh) [226, 119] ή ρητώς από την ιεραρχία [185].

Στα πλαίσια της της *online αναλυτικής επεξεργασίας* (online analytical processing-OLAP), αρκετές προσεγγίσεις παρέχουν ιεραρχική οπτική διερεύνηση, χρησιμοποιώντας τις προκαθορισμένες ιεραρχίες. Το [249] προτείνει μια κατηγορία OLAP-aware ιεραρχικών οπτικών διατάξεων. Ομοίως, το [321] χρησιμοποιεί OLAP-based ιεραρχικά στιβαγμένες μπάρες (hierarchical stacked bars). Το *Polaris* [311] προσφέρει οπτική διερευνητική ανάλυση αποθηκών δεδομένων με πλούσια ιεραρχική δομή.

Επιπλέον, αρκετές ιεραρχικές τεχνικές έχουν προταθεί στα πλαίσια της οντολογίας οπτικοποίησης και διερεύνσης, πχ. [342, 220] (βλέπε Ενότητα 4.1.5.1.4) Τέλος, στα πλαίσια της ιεραρχικής πλοήγησης, το [207] οργανώνει αποτελέσματα ερωτήσεων χρησιμοποιώντας MeSH concept hierarchy. Στο [94] μια ιεραρχική δομή έχει δυναμικά δημιουργηθεί για να κατηγοριοποιεί αριθμητικά και κατηγορικά αποτελέσματα ερωτήσεων. Ομοίως, το [105] δημιουργεί προσωποποιημένες ιεραρχίες υπολογίζοντας διαφορετικές προτιμήσεις χρηστών.

#### 4.1.5.3 Δομές Δεδομένων και Επεξεργασία Δεδομένων

Σε αυτή την ενότητα, παρουσιάζουμε τις δομές δεδομένων και τις τεχνικές προεπεξεργασίας δεδομένων, οι οποίες είναι και πιο σχετικές με την προσέγγιση μας.

Το *R-Tree* [170] είναι μια disk-based, multi-dimensional, indexing δομή, η οποία έχει χρησιμοποιείται προκειμένου να χειριστούμε αποδοτικά χωρικές ερωτήσεις. Το R-Tree υιοθετεί την ιδέα των minimum bounding rectangles (MBRs) προκειμένου να οργανώσει ιεραρχικά πολυδιάστατα αντικείμενα.

Η *διακριτοποίηση δεδομένων* (Data discretization) [162, 134] είναι μια διαδικασία όπου συνεχείς ιδιότητες μετατρέπονται σε διακριτές. Ένας μεγάλος αριθμός μεθόδων (π.χ. με επίβλεψη, χωρίς επίβλεψη) διακριτοποίησης δεδομένων έχει προταθεί.

Το *Binning* είναι μια απλή μέθοδος διακριτοποίησης χωρίς επίβλεψη με την οποία δημιουργείται ένας προκαθορισμένος αριθμός από bins. Ευρέως γνωστές binning μέθοδοι είναι οι *equal-width* και οι *equal-frequency*. Στην προσέγγιση ίσου πλάτους (equal-width approach), το εύρος μιας ιδιότητας χωρίζεται σε διαστήματα που έχουν ίσο πλάτος και κάθε διάστημα αντιπροσωπεύει ένα bin. Στην προσέγγιση ίσης συχνότητας, ένας ίσος αριθμός από τιμές τοποθετείται σε κάθε bin. Με την επαναλαμβανόμενη εφαρμογή τεχνικών διακριτοποίησης, μπορεί να παραχθεί μια ιεραρχική διακριτοποίηση των τιμών των χαρακτηριστικών (δηλαδή *concept/generalization hierarchies*). [301, 174, 111].

### 4.1.6 Επίλογος

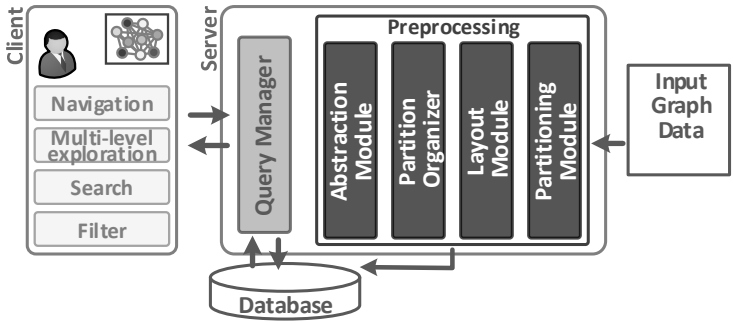
Σε αυτή την ενότητα παρουσιάσαμε το HETree, ένα γενικό μοντέλο που συνδυάζει τη προσωπική πολυεπίπεδη διερεύνηση με την online ανάλυση αριθμητικών και χρονικών δεδομένων. Το μοντέλο μας έχει δημιουργηθεί σε μια ελαφριά δεντροειδή δομή (lightweight tree-based structure), η οποία μπορεί αποδοτικά να δημιουργηθεί άμεσα για ένα δεδομένο σύνολο δεδομένων. Έχουμε παρουσιάσει δύο παραλλαγές για τη δημιουργία του μοντέλου μας: η HETree-C δομή ταξινομεί τα δεδομένα εισόδου σε ομάδες σταθερού μεγέθους, ενώ η HETree-R δομή ταξινομεί τα δεδομένα εισόδου με ομάδες σταθερού εύρους. Με αυτό τον τρόπο ο χρήστης μπορεί να προσαρμόζει τη διερεύνηση, να οργανώνει τα δεδομένα με διαφορετικούς τρόπους, παραμετροποιώντας τον αριθμό των ομάδων, το εύρος και το πλήθος των περιεχομένων τους, τον αριθμό των ιεραρχικών επιπέδων κτλ. Παρουσιάζουμε επίσης ένα τρόπο αποδοτικού υπολογισμού στατιστικών, καθώς και μια μέθοδο για αυτόματη εξαγωγή των καταλληλότερων παραμέτρων για τη κατασκευή του μοντέλου μας. Σχετικά με την επίδοση της πολυεπίπεδης διερεύνησης σε μεγάλα σύνολα δεδομένων, το μοντέλο μας παρέχει σταδιακή δημιουργία του HETree και αποδοτική προσαρμογή του HETree βασισμένη στις προτιμήσεις του χρήστη. Επίσης αναπτύχθηκε ένα πρωτότυπο Web-based σύστημα, που ονομάζεται SynopsViz, βασισμένο στο προτεινόμενο μοντέλο. Τέλος, η αποδοτικότητα και η αποτελεσματικότητα της παρούσας προσέγγισης επιδεικνύονται μέσω μιας λεπτομερούς αξιολόγησης απόδοσης και μιας μελέτης με πραγματικούς χρήστες.

## 4.2 Κλιμακούμενη Εξερεύνηση Γράφων

Η οπτικοποίηση γράφων είναι μία βασική λειτουργία σε πολλές εφαρμογές που συσχετίζονται με την διαχείριση επιστημονικών δεδομένων, την ανάλυση κοινωνικών δικτύων αλλά και στα συστήματα λήψεων αποφάσεων. Με την ευρεία αποδοχή του RDF μοντέλου δεδομένων και την πρόσφατη πρωτοβουλία των ελεύθερων διασυνδεδεμένων δεδομένων (Linked Open Data), τα δεδομένα γράφων είναι πλέον πολύ διαδεδομένα. Η οπτικοποίηση αυτών των δεδομένων με την μορφή γράφων παρέχει στον μη-έμπειρο χρήστη την δυνατότητα να ερευνήσει διαισθητικά τα δεδομένα, να εντοπίσει τμήματα που τον ενδιαφέρουν. Για να μπορεί να γίνει αυτό απαιτείται μια διαδραστική οπτικοποίηση, και όχι μία στατική εικόνα, όπου τα στοιχεία του γράφου αντιστοιχούν σε διακριτά οπτικοποιημένα αντικείμενα, π.χ. DOM αντικείμενα. Με αυτόν τον τρόπο ο χρήστης μπορεί να διαχειριστεί έναν γράφο κατευθείαν από την διεπαφή που του επιτρέπει ανάμεσα σε άλλα να επιλέγει κόμβους και ακμές για να μπορεί να βλέπει επιπλέον πληροφορίες για αυτά και να επιλέγει τμήματα του γράφου. Δεδομένου ότι οι γράφοι που υπάρχουν στον πραγματικό κόσμο είναι τεράστιοι, η οπτικοποίηση όπως παρουσιάστηκε έχει σημαντικές τεχνικές δυσκολίες σε ότι αφορά την διαχείριση των δεδομένων.

Αρχικά, η οπτικοποίηση πρέπει να μπορεί να γίνεται χωρίς να χρειάζεται να φορτωθεί ολόκληρος ο γράφος στην κύρια μνήμη. Τέτοιες ¨ολοκληρωτικές' (holistic) προσεγγίσεις [44, 177] καταλήγουν σε απαγορευτικές απαιτήσεις μνήμης και βασίζονται σε συγκεκριμένες αρχιτεκτονικές πελάτη-εξυπηρετητή (client-server) που δεν είναι πάντα οικονομικά προσιτές από μικρές εταιρείες. Ακόμα, το εργαλείο οπτικοποίησης πρέπει να εξασφαλίζει πολύ μικρό χρόνο ανταπόκρισης ακόμα και σε περιβάλλον με πολλούς χρήστες με απλά μηχανήματα που έχουν περιορισμένες υπολογιστικές ικανότητες. Τέλος, η οπτικοποίηση πρέπει να είναι ελαστική και κατανοητή για την χρήστη, επιτρέποντας του να εξερευνά τον γράφο με πολλούς τρόπους και σε διαφορετικά επίπεδα λεπτομερειών.

Οι πιο πρόσφατες μελέτες στον χώρο [18, 31, 38, 196, 235, 327, 356] αντιμετωπίζουν τα προηγούμενα προβλήματα καταφεύγοντας στην ιεραρχική οπτικοποίηση. Αυτό σημαίνει ότι συγχωνεύουν τμήματα του γράφου και τα παρουσιάζουν ως έναν περιληπτικό κόμβο, επαναληπτικά, μέχρι να δημιουργήσουν μια δομή δέντρου που αποτελείται από αφαιρετικά επίπεδα. Με αυτήν την προσέγγιση καταλήγουμε σε μία αποσύνθεση του γράφου σε πολλούς και σημαντικά μικρότερους υπο-γράφους που οπτικοποιούνται αυτόνομα εάν ζητηθεί από τον χρήστη, αν ο χρήστης επιλέξει να επεκτείνει τον περιληπτικό κόμβο για να δει το προηγούμενο αφαιρετικό επίπεδο. Στις περισσότερες περιπτώσεις, η ιεραρχία κατασκευάζεται με την χρήση μεθόδων συσταδοποίησης (clustering) και κατάτμησης (partitioning) [18, 38, 44, 196, 327]. Σε άλλες εργασίες, η ιεραρχία καθορίζεται με βάση τεχνικές που βασίζονται σε κέντρα (hub-based) [235] και στην πυκνότητα του γράφου (density-based) [356]. Το [31] υποστηρίζει ιεραρχίες που ορίζονται από τον χρήστη. Μία διαφορετική προσέγγιση βλέπουμε στο [314] που εκμεταλλεύεται τεχνικές δειγματοληψίας (sampling). Τέλος, σε ότι αφορά τα δεδομένα Ιστού, υπάρχει ένας μεγάλος αριθμός εργαλείων για την οπτικοποίηση RDF γράφων [69] που όμως απαιτούν την φόρτωση ολόκληρου του γράφου στην διεπαφή χρήστη. Παρόλο που η ιεραρχική προσέγγιση δίνει όμορφες οπτικοποιήσεις με χαμηλές απαιτήσεις μνήμης δεν επιτρέπει την ενστικτώδη εξερεύνηση ενός γράφου, δηλαδή το να ακολουθήσει ο χρήστης ένα μονοπάτι του. Ακόμα, με την ιεραρχική προσέγγιση είναι πολύ δύσκολο να εξερευνηθούν πυκνά τμήματα του γράφου με όλες τις λεπτομέρειες και




**Σχήμα 4.11:** Αρχιτεκτονική της Πλατφόρμας

όχι σε κάποιο ιεραρχικό επίπεδο. Τέλος, η δυνατότητα εφαρμογής ιεραρχίας εξαρτάται πλήρως από τα ιδιαίτερα χαρακτηριστικά του συνόλου των δεδομένων που είναι προς οπτικοποίηση, χαρακτηριστικά παραδείγματα είναι η ύπαρξη μικρών και συνεκτικών συστάδων [18, 31, 38, 196, 327], ή η ομαλή κατανομή του βαθμού συνεκτικότητας των κόμβων [235, 356]. Μια εκτεταμένη παρουσίαση των σχετικών εργασιών παρατίθεται στην Ενότητα 4.1.5

Σε αυτή την εργασία παρουσιάζουμε μια πλατφόρμα για κλιμακούμενη πολυεπίπεδη οπτική εξερεύνηση μεγάλων γράφων. Η προτεινόμενη πλατφόρμα μπορεί εύκολα να υποστηρίξει ποικίλες οπτικοποιήσεις συμπεριλαμβανομένων και όλων των προτάσεων στην βιβλιογραφία και βασίζει την αποτελεσματικότητα της σε μία καινοτόμα τεχνική για την αποθήκευση με βάση ευρετήριο του γράφου σε πολλά αφαιρετικά επίπεδα. Συγκεκριμένα, η προσέγγιση μας περιλαμβάνει μία φάση προεπεξεργασίας όπου υπολογίζεται η διάταξη του γράφου με την απόδοση συντεταγμένων στους κόμβους του με βάση εναν Ευκλείδειο χώρο. Η ίδια προεπεξεργασία ακολουθείται για όλα τα αφαιρετικά επίπεδα. Οι συντεταγμένες αποθηκεύονται με την βοήθεια μίας δομής χωρικών δεδομένων σε μία ΒΔ. Με αυτόν τον τρόπο ο ενέργειες του χρήστη αντιστοιχίζονται σε χωρικά ερωτήματα στην ΒΔ. Η πλατφόρμα που παρουσιάζουμε εδώ είναι η απόδειξη ότι διαδραστικές οπτικοποιήσεις μπορούν να είναι αποτελεσματικές σε απλούς υπολογιστές επιτρέποντας παράλληλα στον χρήστη να εξερευνήσει πλήρως τον γράφο ανεξάρτητα από το αφαιρετικό επίπεδο και το μέγεθος του.

### 4.2.1 Επισκόπηση Συστήματος

Η αρχιτεκτονική της πλατφόρμας μας, γραπῐ́ζδβ Σχήμα 4.11, αποτελείται από τρία βασικά τμήματα: (1) τον Πελάτη (client) (2) το βασικό υποσύστημα graphVizdb και (3) την βάση δεδομένων. Ο Πελάτης είναι η διεπαφή χρήστη που παρέχει μία σειρά από λειτουργίες στον χρήστη, έναν διαδραστικό καμβά, δυνατότητες αναζήτησης, πολύ-επίπεδη αναζήτηση και άλλα. Το βασικό υποσύστημα περιέχει το τμήμα της προεπεξεργασίας και των Διαχειριστή Ερωτημάτων που είναι υπεύθυνος για την επικοινωνία ανάμεσα στον Πελάτη και την βάση δεδομένων. Το τμήμα της προεπεξεργασίας χωρίζεται στα υποσυστήματα Κατάτμηση, Διάταξη γράφου, Οργάνωση κατατμήσεων και Δημιουργία Επιπέδων. Τέλος η βάση δεδομένων περιέχει όλα τα απαραίτητα δεδομένα για την οπτικοποίηση μαζί με όλα τα απαραίτητα ευρετήρια.

### 4.2.2 Προεπεξεργασία Δεδομένων

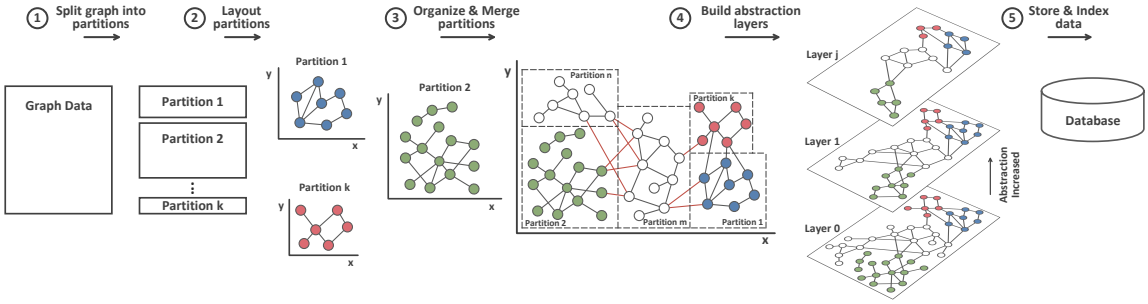
Στην πλατφόρμα μας, η διάταξη του γράφου υπολογίζεται μόνο μία φορά στον Εξυπηρετητή με την χρήση οποιουδήποτε επιθυμητού αλγορίθμου για σχεδιασμό γράφων. Το αποτέλεσμα αυτής της διαδικασίας είναι να δοθούν σε κάθε κόμβο του γράφου συντεταγμένες με βάση τον Ευκλείδειο χώρο. Οι πιο σύγχρονοι αλγόριθμοι διάταξης γράφων παρέχουν εξαιρετικά αποτελέσματα αλλά απαιτούν στην πράξη πολύ μνήμη ακόμα και για μικρούς γράφους με μερικές χιλιάδες ακμές και κόμβους. Για να αντιμετωπίσουμε αυτό το πρόβλημα το προσεγγίσαμε με μία τεχνική κατάτμησης όπως δείχνει το Σχήμα 4.12.

Αρχικά, ο γράφος χωρίζεται από το υποσύστημα Κατάτμηση σε ένα σύνολο $k$ υπογράφων (*Βήμα 1*) όπου το $k$ καθορίζεται από το μέγεθος του γράφου και την διαθέσιμη μνήμη. Ο διαχωρισμός γίνεται με στόχο να περιοριστούν όσο γίνεται οι ακμές ανάμεσα στους υπογράφους [206]. Στην συνέχεια, το υποσύστημα *Διάταξη Γράφου* εφαρμόζει τον αλγόριθμο διάταξης που έχει επιλεγεί, μπορεί να είναι οποιοσδήποτε αλγόριθμος, και δίνει συντεταγμένες στους κόμβους κάθε υπογράφου ανεξάρτητα από τους άλλους υπογράφους και τις όποιες ακμές προς αυτούς (*Βήμα 2*). Οι ακμές ανάμεσα στους διαφορετικούς υπογράφους λαμβάνονται υπόψιν από το υποσύστημα *Οργάνωση Κατατμήσεων* όταν οργανώνει τις κατατμήσεις στον χώρο (*Βήμα 3*). Πολλαπλά αφαιρετικά επίπεδα του γράφου δημιουργούνται από το υποσύστημα *Δημιουργία Επιπέδων* (*Βήμα 4*). Στο *Βήμα 5* ο γράφος μαζί με τα αφαιρετικά του επίπεδα αποθηκεύεται στην βάση δεδομένων με την βοήθεια ευρετήριων.

#### 4.2.2.1 Οργάνωση Κατατμήσεων

Οι κατατμήσεις οργανώνονται στον χώρο με την βοήθεια ενός άπληστου (greedy) αλγορίθμου που έχει διπλό στόχο. Ο πρώτος είναι να εξασφαλίσει ότι οι υπογράφοι δεν θα επικαλύπτονται στον χώρο, ενώ παράλληλα προσπαθεί να ελαχιστοποιήσει το συνολικό μήκος των ακμών που υπάρχουν ανάμεσα στους υπογράφους (*εξωτερικές ακμές*).

Αρχικά ο αλγόριθμος υπολογίζει τις εξωτερικές ακμές για κάθε κατάτμηση. Στην συνέχεια επιλέγει την κατάτμηση με τις περισσότερες εξωτερικές ακμές προς όλες τις άλλες και την τοποθετεί στο κέντρο του χώρου, ενημερώνει δηλαδή κατάλληλα τις συντεταγμένες των κόμβων που αυτή περιέχει. Αυτή είναι η $m$ κατάτμηση στο Σχήμα 4.12 που έχει 9 τέτοιες ακμές, με κόκκινο χρώμα. Οι υπόλοιπες κατατμήσεις κρατούνται σε ουρά προτεραιότητας με βάση τον αριθμό των εξωτερικών ακμών που έχουμε με τις κατατμήσεις που ήδη βρίσκονται τοποθετημένες στον χώρο. Σε κάθε επόμενο βήμα, ο αλγόριθμος τοποθετεί την πρώτη κατάτμηση στην ουρά σε μία διαθέσιμη θέση στον χώρο ελαχιστοποιώντας το ελάχιστο μήκος των εξωτερικών ακμών. Η κατάτμηση αφαιρείται από την ουρά προτεραιότητας, η οποία με την σειρά της ενημερώνεται κατάλληλα, και οι συντεταγμένες των κόμβων που περιέχει ενημερώνονται με βάση την δοθείσα θέση στον χώρο. Η διαδικασία επαναλαμβάνεται μέχρι η ουρά προτεραιότητας να αδειάσει. Η αποτελεσματικότητα του αλγορίθμου στηρίζεται τόσο στον περιορισμένο αριθμό των κατατμήσεων όσο και στην περιοχή που πρέπει να ελέγξουμε για την τοποθέτηση κάθε κατάτμησης που βρίσκεται μόνο γύρω από τις περιοχές που έχουν ήδη καταληφθεί σε προηγούμενα βήματα.




**Σχήμα 4.12:** Επισκόπηση Προεπεξεργασίας

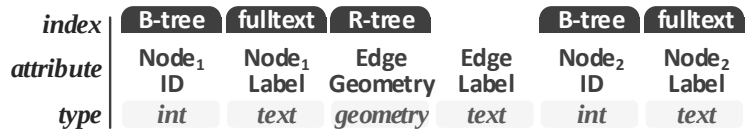
#### 4.2.2.2 Δημιουργία Αφαιρετικών Επιπέδων

Εφόσον οι κατατμήσεις οργανωθούν στον χώρο, κατασκευάζονται μία σειρά από αφαιρετικά επίπεδα (abstraction levels) με βάση τον αρχικό γράφο (Σχήμα 4.12). Κάθε επίπεδο $i$ ($i > 0$) αντιστοιχεί σε έναν νέο γράφο που προέρχεται από τον γράφο του επιπέδου $(i-1)$ με την εφαρμογή μίας αφαιρετικής μεθόδου. Η ιεραρχία των επιπέδων κατασκευάζεται από κάτω προς τα πάνω ξεκινώντας από τον αρχικό γράφο στο επίπεδο 0. Κάθε φορά που κατασκευάζουμε έναν νέο γράφο για το επίπεδο ι στηριζόμαστε στην διάταξη του γράφου του επιπέδου $i-1$. Η αφαιρετική μέθοδος μπορεί να είναι οποιοσδήποτε αλγόριθμος που παράγει μία πιο συμπυκνωμένη μορφή του αρχικού γράφου είτε συνενώνοντας τμήματα του γράφου σε έναν κόμβο είτε φιλτράροντας τμήματα του γράφου με βάση κάποια μετρική όπως είναι κριτήρια βαθμολόγησης κόμβων σαν το PageRank. Είναι σημαντικό να επισημάνουμε ότι η προσέγγιση μας δεν θέτει κανέναν περιορισμό στον αριθμό των επιπέδων ή στο μέγεθος του γράφου σε κάθε επίπεδο. Τα επίπεδα που δημιουργούνται αποθηκεύονται σαν ξεχωριστοί γράφοι όπως περιγράφουμε στην συνέχεια.

#### 4.2.2.3 Αποθηκευτικό Σχήμα

Η βάση δεδομένων μας περιλαμβάνει μόνο έναν πίνακα για κάθε αφαιρετικό επίπεδο στον οποίο αποθηκεύονται όλες οι πληροφορίες για τον γράφο του επιπέδου. Όλοι οι πίνακες έχουν το ίδιο σχήμα όπως παρουσιάζεται στο Σχήμα 4.13. Ο γράφος αποθηκεύεται σαν ένα σύνολο από τριπλέτες της μορφής (κόμβος1, ακμή, κόμβος2). Μία γραμμή του πίνακα του Σχήμα 4.13 περιέχει: (1) τον μοναδικό αναγνωριστικό αριθμό για τον πρώτο κόμβο, (2) την τιμή του πρώτου κόμβου, (3) την γεωμετρία της ακμής, ένα αντικείμενο που αναπαριστά την γραμμή ανάμεσα στους δύο κόμβους στον χώρο, (4) την τιμή της ακμής, (5) τον μοναδικό αναγνωριστικό αριθμό για τον δεύτερο κόμβο και (6) την τιμή του δεύτερου κόμβου. Στην περίπτωση που έχουμε κατευθυνόμενο γράφο, ο πρώτος κόμβος είναι η πηγή και ο δεύτερος ο στόχος.

Χρησιμοποιούνται ευρετήρια B+δέντρων για τα γνωρίσματα (1) και (5) ώστε να εντοπίζονται αποτελεσματικά όλοι οι γείτονες ενός κόμβου, ευρετήρια κειμένου πάνω στα γνωρίσματα (2), (4) και (6) ώστε να είναι εύκολη η αναζήτηση με λέξη κλειδί στα δεδομένα του γράφου και ένα ευρετήριο R-δέντρου χρησιμοποιείται στο γνώρισμα (3) για να είναι εύκολος ο εντοπισμός των ακμών στον χώρο. Κάθε γεωμετρία αναγνωρίζεται από τις συντεταγμένες των δύο κόμβων των οποίων οι μοναδικοί αναγνωριστικοί αριθμοί και οι τιμές είναι αποθηκευμένες στην ίδια γραμμή του πίνακα.




**Σχήμα 4.13:** Αποθηκευτικό Σχήμα

**Πίνακας 4.6:** Χρόνος για κάθε Βήμα της Προεπεξεργασίας (min)

| Dataset | #Edges | #Nodes | Step 1 | Step 2 | Step 3 | Step 4 | Step 5 |
|---|---|---|---|---|---|---|---|
| Wikidata | 151M | 146M | 1.8 | 4.5 | 25.5 | 16.5 | 670.1 |
| Patent | 16.5M | 3.8M | 5.1 | 2.8 | 9.7 | 8.2 | 41.2 |

### 4.2.3 Υλοποίηση

Έχουμε υλοποιήσει ένα πρότυπο σύστημα για την πλατφόρμα graphVizdb[14] το οποίο παρέχει δια δραστική οπτικοποίηση πολύ μεγάλων γράφων. Το πρωτότυπο διαθέτει τρεις βασικές λειτουργίες: (1) δια δραστική πλοήγηση, (2) εξερεύνησης πολλαπλών επιπέδων, και (3) αναζήτηση λέξεων-κλειδιών. Χρησιμοποιούμε MySQL για την αποθήκευση δεδομένων και ευρετηρίαση, το πλαίσιο Θενα για τον χειρισμό των RDF δεδομένων, το Metis[15] για τον διαχωρισμό των γράφων, και το Graphviz[16] για την απότύπωση. Στην πρόσοψη (front-end), χρησιμοποιούμε την βιβλιοθήκη mxgraph[17], για την οπτικοποίηση των γράφων στο πρόγραμμα περιήγησης (browser). Στο Σχήμα 4.14 παρουσιάζεται η γραφική διεπαφή της εφαρμογής.

### 4.2.4 Πειραματική Ανάλυση

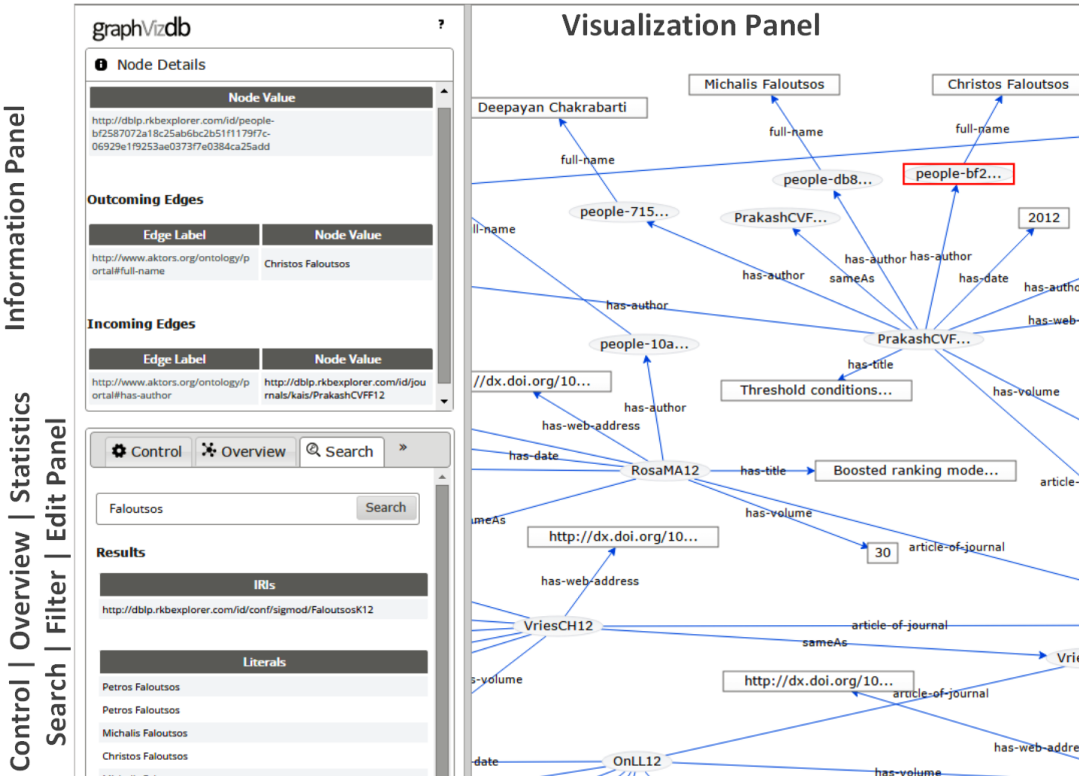
#### 4.2.4.1 Περιβάλλον

Τα πειράματα που παρουσιάζουμε εδώ έγιναν στον Okeanos cloud με την χρήση μίας εικονικής μηχανής με τετραπύρηνη ΚΠΥ στα 2GHz και 8GB RAM που τρέχει το λειτουργικό LINUX. Για να τρέχουμε την εφαρμογή χρήστη χρησιμοποιήσαμε Google Chrome σε φορητό υπολογιστή με i7 CPU στα 1.8GHz και 4GB RAM. Η διαθέσιμη μνήμη για την MySQL ήταν 6GB.

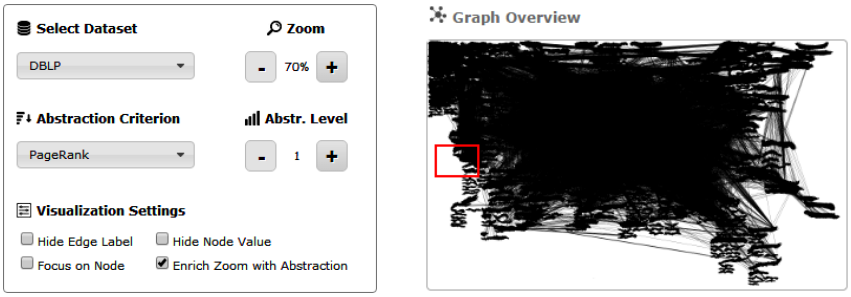
#### 4.2.4.2 Σύνολα Δεδομένων

Για να αξιολογήσουμε τον χρόνο ανταπόκρισης του συστήματος μας χρησιμοποιήσαμε πολλά πραγματικά γραφικά σύνολα δεδομένων με ποικίλα χαρακτηριστικά. Εδώ παρουσιάζουμε αποτελέσματα από δύο τέτοια σύνολα: το Wikidata RDF σύνολο δεδομένων και το Patent Citation γράφο. Το πρώτο είναι μία εξαγωγή RDF δεδομένων από την Wikidata με 151M ακμές και 146M κόμβους. Το δεύτερο είναι από το SNAP αποθετήριο για μεγάλα γραφικά σύνολα δεδομένων με 16.5M ακμές και 3.8M κόμβους.

[14] graphvizdb.imis.athena-innovation.gr
[15] glaros.dtc.umn.edu/gkhome/views/metis
[16] www.graphviz.org
[17] www.jgraph.com




(α′) Μαιν ωινδοω

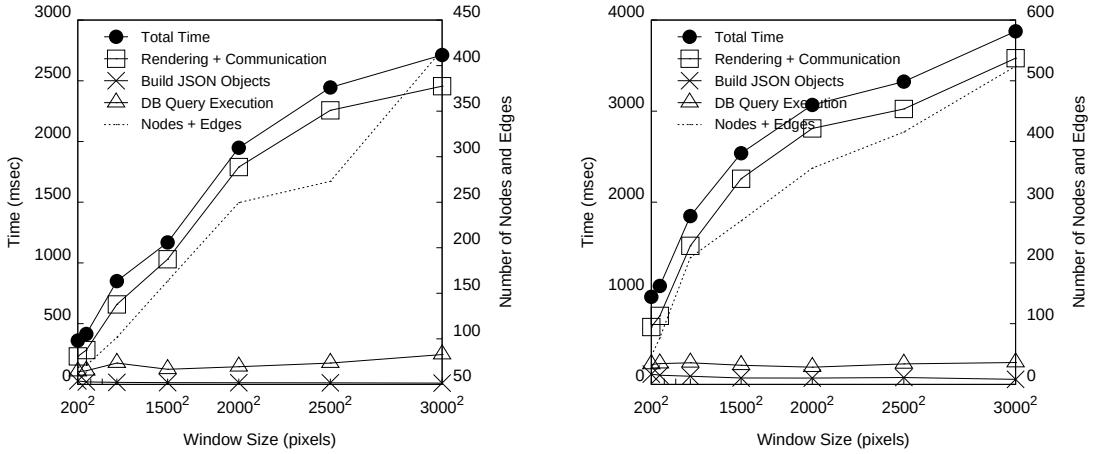



(β′) δντρολ πανελ (γ′) Βιρδιεω

**Σχήμα 4.14:** Ωεβ υσερ ιντερφαςε

### 4.2.4.3 Προεπεξεργασία

Στον Πίνακα 4.6 παρουσιάζεται ο χρόνος που χρειάζεται για κάθε βήμα της προεπεξεργασίας όπως αυτή παρουσιάστηκε στο Σχήμα 4.12. Οι χρόνοι αυτοί είναι μεγαλύτεροι για το σύνολο δεδομένων Wikidata δεδομένου ότι είναι πολύ μεγαλύτερο από το Patent. Εξαίρεση σε αυτό είναι το Βήμα 1, η κατάτμηση σε $k$ μέρη, που θέλει μεγαλύτερο χρόνο για το Πατεντ λόγο του μεγαλύτερου μέσου βαθμού ανά κόμβο. Παρατηρούμε ότι το πιο χρονοβόρο τμήμα της προεπεξεργασίας είναι η δημιουργία των ευρετηρίων που όμως αντιστοιχούν στον χρόνο που χρειάστηκε για να δημιουργηθούν τα ευρετήρια και για τα πέντε επίπεδα του γράφου αθροιστικά. Στην πράξη αυτή η διαδικασία μπορεί να παραλληλοποιηθεί και ο χρόνος που να χρειάζεται για το Βήμα 5 να είναι αυτός για τον μεγαλύτερο γράφο, τον αρχικό του επιπέδου 0 που είναι 274.5 και 17.4 λεπτά για το Wikidata και το Patent αντίστοιχα.




(α′) Wikidata (β′) Patent

**Σχήμα 4.15:** Χρόνος vs. Μέγεθος Παραθύρου

#### 4.2.4.4 Εξερεύνηση

Το σενάριο για τα πειράματα μας περιλαμβάνει την μέτρηση του χρόνου ανταπόκρισης της εφαρμογής για ερωτήματα σε παράθυρα με διαφορετικά μεγέθη. Τα ερωτήματα αυτά υπολογίζονται στον διακομιστή και στέλνονται στην διεπαφή χρήστη για την οπτικοποίηση. Χρησιμοποιούμε παράθυρα με μέγεθος από $200^2$ έως $3000^2$ pixels και καταγράφουμε τους χρόνους για τον αρχικό γράφο του επιπέδου 0.

Για κάθε μέγεθος παραθύρου εκτελούμε 100 τυχαία ερωτήματα. Τα αποτελέσματα παρουσιάζονται στο Σχήμα 3 και περιλαμβάνουν τους μέσους χρόνους των ερωτημάτων σε μσες: (1) *DB Query Execution*: ο χρόνος που χρειάζεται η βάση δεδομένων για να απαντήσει στο ερώτημα, (2) *Build JSON Objects*: ο χρόνος που χρειάζεται ο διακομιστής για να επεξεργαστεί την απάντηση της βάσης δεδομένων και να κατασκευάσει τα ΘΣΟΝ αντικείμενα που στέλνει στον πελάτη, (3) *Communication + Rendering*: ο χρόνος που διαρκεί η επικοινωνία διακομιστή-πελάτη μαζί με τον χρόνο που κάνει να οπτικοποιηθεί ο γράφος στον χρήστη, και (4) *Total Time*: το άθροισμα όλων των προηγούμενων χρόνων. Η τιμή *Nodes + Edges* του Σχήματος 4.15 αναφέρονται στην μέση τιμή των κόμβων και των ακμών που περιλαμβάνονται σε 1K τυχαία παράθυρα κάθε μεγέθους.

Η πρώτη παρατήρηση είναι ότι η απόδοση της εφαρμογής μας είναι γραμμική ως προς το μέγεθος του παραθύρου και του συνολικού αριθμού αντικειμένων σε αυτό. Η συμπεριφορά είναι παρόμοια και για τα δύο σύνολα δεδομένων. Όπως βλέπουμε στο Σχήμα 4.15 ο χρόνος ανταπόκρισης της εφαρμογής εξαρτάται κυρίως από τον χρόνο *Communication + Rendering*. Οι δύο αυτές τιμές παρουσιάζονται ενιαία γιατί ο γράφος φτάνει στην διεπαφή του χρήστη σε μικρά τμήματα και οι χρόνοι δεν είναι εύκολα διακριτοί. Τέλος, πρέπει να επισημάνουμε ότι ο χρόνος που χρειάζεται η βάση δεδομένων για να επιστρέψει το αποτέλεσμα είναι αμελητέος και δεν αυξάνεται σημαντικά με την αύξηση του μεγέθους του παραθύρου.

### 4.2.5 Επίλογος

Σε αυτή την ενότητα παρουσιάσαμε μία καινοτόμα πλατφόρμα για την διαδραστική απεικόνιση πολύ μεγάλων γράφων. Η πλατφόρμα που παρουσιάζεται επιτρέπει στον χρήστη να αλληλεπιδρά με τον οπτικοποιημένο γράφο με έναν τρόπο όμοιο με τον τρόπο που εξερευνώνται οι χάρτες πολλαπλών επιπέδων. Για να είναι δυνατή η οπτικοποίηση πολύ μεγάλων γράφων, προτείνουμε μία τεχνική οπτικοποίησης που βασίζεται στην κατάτμηση. Η προσέγγιση μας περιλαμβάνει μία φάση προεπεξεργασίας των δεδομένων κατά την διάρκεια της οποίας υπολογίζεται η διάταξη του γράφου με την απόδοση συντεταγμένων στους κόμβους του με βάση έναν Ευκλείδειο χώρο. Στην συνέχεια το σύστημα μας αντιστοιχίζει τις ενέργειες του χρήστη σε χωρικά ερωτήματα που μπορούν να αντιμετωπιστούν εύκολα και αποδοτικά από την υποδομή που δημιουργήσαμε στην φάση της προεπεξεργασίας. Με αυτήν την τεχνική έχουμε τόσο στοχευμένη πρόσβαση σε πάρα πολύ μεγάλους γράφους χωρίς διαστήματα αδράνειας όσο και χαμηλές απαιτήσεις σε μνήμη. Τέλος αναπτύξαμε μία διαδικτυακή εφαρμογή που υποστηρίζει τέσσερις βασικές λειτουργίες: (1) διαδραστική εξερεύνηση, (2) εξερεύνηση πολλαπλών επιπέδων, (3) διαχείριση υπογράφων και (4) αναζήτηση με λέξεις κλειδιά.

# Μέρος III

# Σημασιολογική Ανάλυση Δεδομένων

# Κεφάλαιο 5

# Διαλειτουργικότητα μεταξύ XML και Σημασιολογικού Περιβάλλοντος

Ο *Ιστός Δεδομένων* (Web of Data - WoD) είναι ένα ανοιχτό περιβάλλον που αποτελείται από ένα σημαντικό αριθμό από μεγάλα σύνολα αλληλοσυνδεδεμένων RDF δεδομένων από διαφορετικούς τομείς. Σε αυτό το περιβάλλον, οργανώσεις και εταιρίες υιοθετούν τις πρακτικές *Διασυνδεδεμένων Δεδομένων* (Linked Data) χρησιμοποιώντας τεχνολογίες *Σημασιολογικού Ιστού* (Semantic Web - SW), προκειμένου να δημοσιεύουν τα δεδομένα και να προσφέρουν SPARQL σημεία πρόσβασης (endpoints). Από την άλλη μεριά, στο Διαδίκτυο σήμερα, το κυρίαρχο πρότυπο ανταλλαγής δεδομένων είναι το XML. Ακόμα, πολλά διεθνή πρότυπα (π.χ., *Dublin Core*, *MPEG-7*, *METS*) σε αρκετούς τομείς (π.χ. Digital Libraries, GIS, Multimedia) έχουν διατυπωθεί σε XML Schema. Τα προηγούμενα οδήγησαν στο να δίνεται όλο και μεγαλύτερη έμφαση στα XML δεδομένα, στα οποία έχουμε πρόσβαση μέσω της γλώσσας ερωτήσεων XQuery. Οι SW και XML κόσμοι, καθώς και οι αναπτυγμένες υποδομές τους βασίζονται σε διαφορετικά μοντέλα δεδομένων, σημασιολογία και γλώσσες ερωτήσεων. Επομένως, είναι σημαντικό να αναπτύσσονται μηχανισμοί διαλειτουργικότητας (interoperability mechanisms) που επιτρέπουν στους χρήστες του Ιστού Δεδομένων να έχουν πρόσβαση σε XML σύνολα δεδομένων XML datasets, χρησιμοποιώντας την SPARQL. Δεν είναι ρεαλιστικό να περιμένουμε ότι τα υπάρχοντα δεδομένα (π.χ.,Relational, XML) θα μετατραπούν σε SW δεδομένα. Για αυτό, η έκδοση legacy δεδομένων ως διασυνδεδεμένων δεδομένων και η παροχή SPARQL σημείων πρόσβασης έχει γίνει σημαντική ερευνητική πρόκληση. Σε αυτή τη κατεύθυνση, παρουσιάζουμε το πλαίσιο *SPARQL2XQuery* το οποίο δημιουργεί ένα διαλειτουργικό περιβάλλον, όπου οι SPARQL ερωτήσεις μεταφράζονται αυτόματα σε XQuery ερωτήσεις, προκειμένου να αποκτήσουν πρόσβαση σε XML δεδομένα. Το πλαίσιο SPARQL2XQuery παρέχει ένα μοντέλο αντιστοιχίσεων (mapping model) για την αναπαράσταση αντιστοιχίσεων μεταξύ OWL–RDF/S και XML Schema, καθώς και μιά μέθοδο για την μετάφραση SPARQL ερωτήσεων σε XQuery. Επιπλέον, το πλαίσιο υποστηρίζει τόσο τον χειροκίνητο όσο και τον αυτόματο προσδιορισμό των αντιστοιχήσεων (mapping specification) μεταξύ οντολογιών (ontologies) και XML σχημάτων. Στη περίπτωση του αυτόματου προσδιορισμού αντιστοιχίσεων, το SPARQL2XQuery εκμεταλλεύεται το στοιχείο XS2OWL το οποίο μετατρέπει XML σχήματα σε OWL οντολογίες. Τέλος, πραγματοποιήθηκαν εκτεταμένα πειράματα για να αξιολογηθεί ο μετασχηματισμός του σχήματος (schema transfor-

mation), η δημιουργία αντιστοιχίσεων (mapping generation), μετάφραση ερωτημάτων (query translation) και η αποδοτικότητα της αποτίμησης των ερωτήσεων (query evaluation efficiency), χρησιμοποιώντας πραγματικά και συνθετικά σύνολα δεδομένων.

## 5.1 Εισαγωγή

Πρωτοβουλίες όπως τις Linked Open Data, Open-Government και Linked Life Data έχουν διαδραματίσει σημαντικό ρόλο στην ανάπτυξη του λεγόμενου Ιστού Δεδομένων (Web of Data). Στον Ιστό Δεδομένων, ένας μεγάλος αριθμός οργανισμών, ιδρυμάτων και επιχειρήσεων (π.χ., DBpedia, geonames, PubMed, Data.gov, ACM, NASA, BBC, MusicBrainz, IEEE, κλπ.) εφαρμόζουν τις πρακτικές των Διασυνδεμένων Δεδομένων (Linked Data). Αξιοποιώντας τις τεχνολογίες του Σημασιολογικό Ιστού (Semantic Web), δημοσιεύουν τα δεδομένα τους και προσφέρουν SPARQL σημεία πρόσβασης (SPARQL endpoints). Σήμερα στον Ιστό Δεδομένων, υπάρχουν εκατοντάδες αλληλοσυνδεόμενα (inter-linked) σύνολα RDF δεδομένων από διάφορους τομείς. Μια μεγάλη πρόκληση για τον Ιστό Δεδομένων, είναι η διάθεση και δημοσιοποίηση παραδοσιακών πηγών δεδομένων (π.χ., σχεσιακές βάσεις, XML δεδομένα, κλπ).

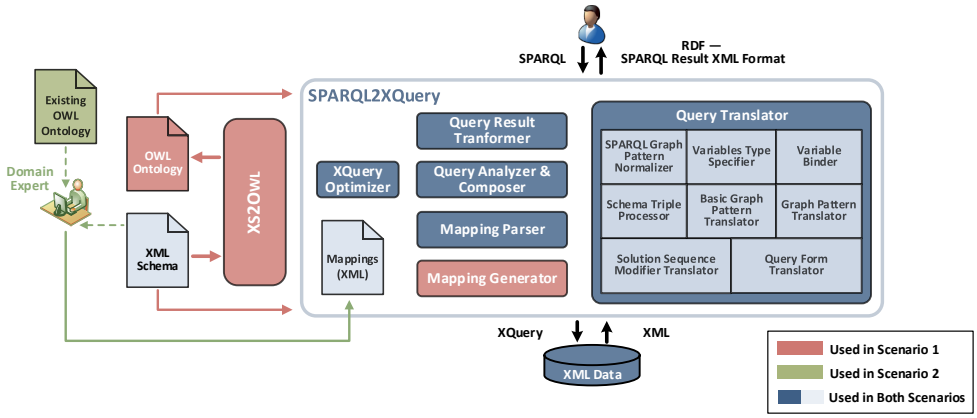
Οι τεχνολογίες του Σημασιολογικού Ιστού παρέχουν την δυνατότητα διαχείρισης RDF δεδομένων [87, 248, 258], προφέροντας πρόσβαση μέσω της γλώσσα ερωτήσεων SPARQL [281]. Από την άλλη μεριά, στο Διαδίκτυο σήμερα, το κυρίαρχο πρότυπο ανταλλαγής δεδομένων είναι το XML/XML Schema, το οποίο βασίζεται στο ημιδομημένο μοντέλο δεδομένων, με κυρίαρχη γλώσσα ερωτήσεων την XQuery. Λόγω των δυνατοτήτων δόμησης που παρέχει η γλώσσα XML Schema και του κεντρικού ρόλου που παίζει κατά την ανταλλαγή δεδομένων στο Διαδίκτυο, σημαντικά πρότυπα περιγραφής δεδομένων και μεταδεδομένων (metadata) για πολλές διαφορετικές περιοχές εφαρμογών έχουν εκφραστεί στη γλώσσα XML Schema [1, 6, 13, 9, 8, 4, 7, 14, 2, 15, 10, 3, 11], συμπεριλαμβανομένων προτύπων στην περιοχή των πολυμέσων (multimedia), προτύπων ηλεκτρονικής εκπαίδευσης (e-learning), προτύπων για ψηφιακές βιβλιοθήκες (digital libraries) κ.α. Έτσι, οι XML και XML Schema έχουν αποτελέσει τη βάση της συντακτικής και δομικής διαλειτουργικότητας (Structural and Syntactic Interoperability) στο Διαδίκτυο.

Δεδομένου ότι ο Σημασιολογικός και ο XML κόσμος έχουν διαφορετικά μοντέλα δεδομένων, διαφορετική σημασιολογία και χρησιμοποιούν διαφορετικές γλώσσες ερωτήσεων [71], είναι ζωτικής σημασίας η ανάπτυξη μεθοδολογιών που θα διασφαλίζουν τη διαλειτουργικότητα μεταξύ των Σημασιολογικών και των XML υποδομών, διευκολύνοντας έτσι τη διαφανή πρόσβαση σε XML δεδομένα μέσα από τον Ιστό Δεδομένων, χρησιμοποιώντας Σημασιολογικές τεχνολογίες.

Αυτή η εργασία παρουσιάζει το πλαίσιο SPARQL2XQuery, το οποίο δημιουργεί ένα διαλειτουργικό περιβάλλον, όπου οι SPARQL ερωτήσεις μεταφράζονται αυτόματα σε XQuery, προκειμένου να έχουμε πρόσβαση στα XML δεδομένα. Το πλαίσιο SPARQL2XQuery παρέχει ένα μοντέλο αντιστοιχίσεων (mapping model) για την αναπαράσταση αντιστοιχίσεων μεταξύ OWL-RDF/S και XML Schema, καθώς και μια μέθοδο για την μετάφραση SPARQL ερωτήσεων σε XQuery.

### 5.1.1 Επισκόπηση Πλαισίου

Μια επισκόπηση της αρχιτεκτονικής του SPARQL2XQuery συστήματος παρουσιάζεται στο Σχήμα 6.1. Το πλαίσιο SPARQL2XQuery υποστηρίζει δύο διαφορετικά σενάρια.




**Σχήμα 5.1:** Η αρχιτεκτονική του πλαισίου SPARQL2XQuery

Όπως φαίνεται στο Σχήμα 6.1, και τα δυο σενάρια εργασίας περιλαμβάνουν τα υπάρχοντα XML δεδομένα που ακολουθούν ένα ή περισσότερα XML σχήματα.

**1ο Σενάριο.** *Πρόσβαση σε XML δεδομένα βασισμένη σε οντολογίες που έχουν παραχθεί αυτόματα.* Το σενάριο αυτό βασίζεται στο στοιχείο (component) XS2OWL το οποίο έχει αναπτυχθεί και ενσωματωθεί στο πλαίσιο SPARQL2XQuery. Ειδικότερα, το στοιχείο XS2OWL παράγει αυτόματα OWL οντολογίες που αναπαριστούν τα XML σχήματα. Στη συνέχεια, το πλαίσιο SPARQL2XQuery ανιχνεύει, δημιουργεί και διατηρεί τις αντιστοιχίσεις μεταξύ των XML σχημάτων και των OWL οντολογιών που παρήγαγε το XS2OWL. Συγκεκριμένα, σε αυτό το σενάριο, έχουμε τα ακόλουθα βήματα: (α) Χρησιμοποιώντας το XS2OWL, το XML Schema εκφράζεται ως μια OWL οντολογία. (β) Το στοιχείο Mapping Generator παίρνει ως είσοδο το XML Schema και την παραγόμενη οντολογία, και αυτόματα δημιουργεί, συντηρεί και αποθηκεύει τις μεταξύ τους αντιστοιχίσεις, σε μορφή XML. (γ) Τα SPARQL ερωτήματα που διατυπώνονται πάνω στην παραγόμενη οντολογία, μεταφράζονται από το στοιχείο Query Translator σε XQuery εκφράσεις. (δ) Τα αποτελέσματα του ερωτήματος μετασχηματίζεται από το Query Result Transformer στην επιθυμητή μορφή (SPARQL Query Result XML Format ή RDF format).

**2ο Σενάριο.** *Πρόσβαση σε XML δεδομένα βασισμένη σε υπάρχουσες οντολογίες.* Σε αυτό το σενάριο, τα XML σχήματα αντιστοιχίζονται χειροκίνητα (manually) σε υπάρχουσες οντολογίες. Οι αντιστοιχήσεις που προκύπτουν χρησιμοποιούνται στην μετάφραση SPARQL ερωτήσεων σε XQuery. Συγκεκριμένα, σε αυτό το σενάριο, τα ακόλουθα βήματα λαμβάνουν χώρα: (α) Το XML Schema αντιστοιχίζεται χειροκίνητα σε μια υπάρχον RDF/S-OWL οντολογία. (β) Τα SPARQL ερωτήματα που εκφράζονται πάνω στην οντολογία μεταφράζονται σε XQuery εκφράσεις. (γ) Τα αποτελέσματα του ερωτήματος μετασχηματίζεται στην επιθυμητή μορφή.

Το SPARQL2XQuery πλαίσιο υποστηρίζει τις ακόλουθες λειτουργίες:

- *Μετασχηματισμός Σχήματος* (Schema Transformation). Κάθε XML Schema μπορεί να μετατραπεί αυτόματα σε μία OWL οντολογία, χρησιμοποιώντας το στοιχείο XS2OWL.

- *Δημιουργία Αντιστοιχίσεων* (Mapping Generation). Οι αντιστοιχίσεις μεταξύ των XML Schemas και των OWL αναπαραστάσεων τους, ανιχνεύονται αυτόματα και αποθηκεύονται σε XML μορφή.

– *Μετάφραση Ερωτημάτων* (Query Translation). Κάθε SPARQL ερώτημα μεταφράζεται σε ένα XQuery ερώτημα.

– *Μετασχηματισμός Αποτελεσμάτων* (Query Result Transformation). Τα αποτελέσματα του ερωτήματος μετασχηματίζονται στην επιθυμητή μορφή.

### 5.1.2 Συνεισφορά

Οι κύριες συνεισφορές αυτής της εργασίας συνοψίζονται ως εξής:

1. Εισάγουμε το μοντέλο μετασχηματισμού XS2OWL, το οποίο πραγματοποιεί τη μετατροπή του XML Schema σε OWL οντολογία. Από όσο γνωρίζουμε, αυτή είναι η πρώτη εργασία που αποτυπώνει πλήρως τη σημασιολογία του XML Schema.

2. Εισάγουμε ένα μοντέλο αντιστοιχίσεων για τον ορισμό αντιστοιχίσεις μεταξύ RDF/S - OWL και XML Schema.

3. Προτείνουμε μια μέθοδο και μια σειρά αλγορίθμων που παρέχουν μια ολοκληρωμένη SPARQL σε XQuery μετάφραση. Από όσο γνωρίζουμε, αυτή είναι η πρώτη εργασία που ασχολήθηκε με το συγκεκριμένο ζήτημα.

4. Ενσωματώσαμε στο πλαίσιο SPARQL2XQuery το στοιχείο XS2OWL, διευκολύνοντας έτσι την αυτόματη δημιουργία και τη συντήρηση των αντιστοιχίσεων που χρησιμοποιούνται στην SPARQL σε XQuery μετάφραση.

5. Προτείνουμε ένα σύνολο από κανόνες επανεγγραφής/βελτιστοποίησης των XQuery ερωτήσεων που παράγονται από τη μετάφραση. Ο στόχος είναι η παραγωγή πιο αποδοτικών XQuery εκφράσεων. Επιπλέον, μελετήσαμε πειραματικά την επίδραση αυτών των κανόνων στην απόδοση των XQuery ερωτήσεων.

6. Περιγράφουμε την επέκταση του SPARQL2XQuery πλαισίου για την υποστήριξη ερωτήσεων ενημερώσεων (Update queries) .

7. Διεξάγουμε μια εκτενής πειραματική αξιολόγηση, μελετώντας τα παρακάτω: (α) χρόνο μετατροπής σχήματος (β) χρόνο παραγωγής αντιστοιχίσεων (γ) χρόνο μετάφρασης ερωτημάτων και (δ) χρόνο αποτίμησης ερωτημάτων, χρησιμοποιώντας τόσο πραγματικά όσο και συνθετικά δεδομένα.

## 5.2 Σχετικές Εργασίες

Ένας μεγάλος αριθμός συστημάτων *ολοκλήρωσης δεδομένων* (data integration) [245] και *ανταλλαγής δεδομένων* (επίσης γνωστά ως μετατροπής δεδομένων) (data exchange/ transformation) [149] έχουν προταθεί. Στα πλαίσια της XML, οι πρώτες ερευνητικές προσπάθειες έχουν αποπειραθεί να παρέχουν διαλειτουργικότητα και ολοκλήρωση μεταξύ των Σχεσιακών και XML κόσμων [259, 279, 350, 127, 323, 205, 238, 222, 217]. Επιπλέον, αρκετές προσεγγίσεις επικεντρώνονται στην ολοκλήρωση δεδομένων και στην ανταλλαγή μεταξύ ετερογενών XML πόρων δεδομένων [130, 81, 172, 173, 318, 191, 34, 24, 84]. Τέλος, μεγάλος αριθμός μεθόδων έχουν προταθεί για την ενοποίηση σχεσιακών και σημασιολογικών πηγών δεδομένων [221, 299, 78, 332, 290, 90, 103, 246, 101, 143, 148, 282].

Στα πλαίσια της υποστήριξης της διαλειτουργικότητας μεταξύ των SW και XML κόσμων [71], έχουν προταθεί πολλές προσεγγίσεις για τη μετατροπή XML σχημάτων σε οντολογίες, και/ή XML δεδομένα σε RDF δεδομένα και αντίστροφα. Οι πιο πρόσφατες συνδυάζουν τεχνολογίες SW και XML τεχνολογίες προκειμένου να μετατρέψουν τα XML δεδομένα σε RDF και αντίστροφα. Μεταξύ των δημοσιευμένων αποτελεσμάτων, τα πιο σχετικά με τη δική μας προσέγγιση είναι εκείνα που χρησιμοποιούν τη γλώσσα ερωτήσεων SPARQL.

### 5.2.1 Γεφυρώνοντας τον Σημασιολογικό με τον XML κόσμο — Μία επισκόπηση

Σε αυτή τη παράγραφο, συνοψίζουμε τη βιβλιογραφία σχετικά με την διαλειτουργικότητα και τα ζητήματα ολοκλήρωσης μεταξύ των SW και των XML κόσμων. Κατηγοροποιούμε αυτά τα συστήματα σε *συστήματα ολοκλήρωσης δεδομένων* (Πινάκας 6.2) και *συστήματα ανταλλαγής δεδομένων* (Πινάκας 6.3).

Τα συστήματα ολοκλήρωσης δεδομένων (Πινάκας 6.2) είναι παλιότερα και δεν υποστηρίζουν τις σημερινές καθιερωμένες τεχνολογίες (π.χ., XML Schema, OWL, RDF, SPARQL). Παρατηρούμε επίσης ότι, αν και τα συστήματα ανταλλαγής δεδομένων που φαίνονται στον Πίνακα 6.3 είναι πιο πρόσφατα, δεν υποστηρίζουν ένα σενάριο ολοκλήρωσης ούτε παρέχουν μεθόδους μετάφρασης ερωτήσεων. Αντιθέτως, επικεντρώνονται στη μετατροπή δεδομένων και σχημάτων, μελετώντας πώς τα RDF δεδομένα μπορούν να μετατραπούν σε XML σύνταξη και/ή ή πώς τα XML σχήματα μπορούν να εκφραστούν ως οντολογίες και αντίθετα.

### 5.2.2 Πρόσφατες προσεγγίσεις

Σε αυτή την ενότητα, παρουσιάζουμε τις πιο πρόσφατες προσεγγίσεις σχετικά με την υποστήριξη της διαλειτουργικότητας και την ολοκλήρωση μεταξύ των SW και των XML κόσμων. Αυτές οι προσεγγίσεις χρησιμοποιούν τις τωρινές W3C καθιερωμένες τεχνολογίες (π.χ., XML Schema, RDF/S, OWL, XQuery, SPARQL). Οι περισσότερες από τις πιο πρόσφατες προσπάθειες (Πίνακας 6.3) επικεντρώνονται στον συνδυασμό των XML και SW τεχνολογιών προκειμένου να παρέχεται ένα διαλειτουργικού περιβάλλον. Συγκεκριμένα, συγχωνεύουν χαρακτηριστικά των SPARQL, XQuery, XPath και XSLT για να μετατρέψουν XML δεδομένα RDF και αντίστροφα.

Το W3C *Semantic Annotations for WSDL* (SAWSDL) Working Group [153] χρησιμοποιεί XSLT για να μετατρέψει XML δεδομένα σε RDF, και χρησιμοποιεί έναν συνδυασμό SPARQL και XSLT για την αντίστροφη μετατροπή. Επιπλέον, το W3C *Gleaning Resource Descriptions from Dialects of Languages* (GRDDL) Working Group [121] χρησιμοποιεί XSLT για να εξάγει RDF δεδομένα από XML.

Η XSPARQL [25, 76, 75] συνδυάζει SPARQL και XQuery προκειμένου να πετύχει τη μετατροπή των XML σε RDF και αντίστροφα. Στη περίπτωση της μετατροπής από XML σε RDF, η XSPARQL χρησιμοποιεί έναν συνδυασμό από XQuery εκφράσεις και SPARQL Construct queries. Οι XQuery εκφράσεις χρησιμοποιούνται για πρόσβαση σε XML δεδομένα, και οι SPARQL Construct queries χρησιμοποιούνται για να μετατρέψουν τα XML δεδομένα σε RDF. Στη περίπτωση μετατροπής RDF σε XML, η XSPARQL χρησιμοποιεί έναν συνδυασμό SPARQL και XQuery όρων. Οι SPARQL όροι χρησιμοποιούνται για πρόσβαση στα RDF δεδομένα, και οι XQuery όροι χρησιμοποιούνται για τη μορφοποίηση των αποτελεσμάτων σε XML σύνταξη. Ο-

**Πίνακας 5.1:** Παρουσίαση συστημάτων SW - XML ολοκλήρωσης

| **Data Integration Systems** | | | | | |
|---|---|---|---|---|---|
| **System** | **Environment Characteristics** | | | **Operations** | |
| | **Data Models** | **Schema Definition Languages** | **Query Languages** | **Query Translation** | **Schema Transformation** |
| STYX (2002) [29, 56] | XML | DTD / Graph | OQL / XQuery | OQL → XQuery | |
| ICS–FORTH SWIM (2003) [110, 109, 217] | Relational / XML | DTD / Relational / RDF Schema | SQL / XQuery / RQL | RQL → SQL & RQL → XQuery | |
| PEPSINT (2004) [348, 347, 118, 117] | XML | XML Schema / RDF Schema | XQuery / RDQL | RDQL → Xquery | XML Schema → RDF Schema |
| Lehti & Fankhauser (2004) [230] | XML | XML Schema / OWL | XQuery / SWQL | SWQL → XQuery | XML Schema → OWL |
| SPARQL2XQuery | XML | XML Schema / OWL | XQuery / SPARQL | SPARQL → XQuery | XML Schema → OWL (XS2Owl) |

**Πίνακας 5.2:** Παρουσίαση συστημάτων ανταλλαγής SW - XML δεδομένων

| Data Integration Systems | | | | | |
|---|---|---|---|---|---|
| **System** | **Environment Characteristics** | | **Operations** | | |
| | **Data Models** | **Schema Definition Languages** | **Schema Transformation** | **Use Existing Ontology** | **Data Transformation** |
| Klein (2002) [244] | XML / RDF | XML Schema / RDF Schema | | | XML → RDF |
| WEESA (2004) [284] | XML / RDF | XML Schema / OWL | | ✓ | XML → RDF |
| Ferdinand et al. (2004) [154] | XML / RDF | XML Schema / OWL–DL | XML Schema → OWL–DL | | XML → RDF |
| Garcia & Celma (2005) [161] | XML / RDF | XML Schema / OWL–FULL | XML Schema → OWL–FULL | | XML → RDF |
| Bohring & Auer (2005) [82] | XML / RDF | XML Schema / OWL–DL | XML Schema → OWL–DL | | XML → RDF |
| Gloze (2006) [292] | XML / RDF | XML Schema / OWL | | | XML ↔ RDF |
| JXML2OWL (2006 & 2008) [288, 289] | XML / RDF | XML Schema / OWL | | ✓ | XML → RDF |
| GRDDL (2007) [121] | XML / RDF | not specified | | | XML ↔ RDF ⋆ |
| SAWSDL (2007) [153] | XML / RDF | not specified | | | XML ↔ RDF ⋆ |
| Thuy et al. (2007 & 2008) [325, 326] | XML / RDF | DTD / OWL–DL | DTD → OWL–DL ⋆ | | XML → RDF ⋆ |
| Janus (2008 & 2011) [49, 50] | XML / RDF | XML Schema / OWL–DL | XML Schema → OWL–DL | | |
| Deursen et al. (2008) [129] | XML / RDF | XML Schema / OWL | | ✓ | XML → RDF ⋆ |
| XSPARQL (2008) [25, 76, 75] | XML / RDF | not specified | | | XML ↔ RDF ⋆ |
| Droop et al. (2007 & 2008) [137, 136, 135] | XML / RDF | not specified | | | XML → RDF ⋆ |
| Cruz & Nicolle (2008) [116] | XML / RDF | XML Schema / OWL | | ✓ | XML → RDF |
| XSLT+SPARQL (2008) [57] | XML / RDF | not specified | | | RDF → XML |
| DTD2OWL (2009) [324] | XML / RDF | DTD / OWL–DL | DTD → OWL–DL | | XML → RDF |
| Corby et al. (2009) [113] | XML / RDF / Relational | not specified | | | XML → RDF ⋆ Relational → RDF |
| TopBraid Composer (Maestro Edition) – TopQuadrant (Commercial Product) | XML / RDF | not specified / OWL | XML → OWL | | XML ↔ RDF ⋆ |
| XS2OWL | XML / RDF | XML Schema 1.1 / OWL 2 | XML Schema → OWL | | XML ↔ RDF |

⋆ The transformation is performed in a semi-automatic way that requires user intervention.

μοίως, στο [113] XPath, XSLT και SQL είναι ενσωματωμένα σε SPARQL ερωτήσεις προκειμένου να μετατρέψουν XML και σχετικά δεδομένα (relational data) σε RDF. Στο XSLT+SPARQL [57] η γλώσσα XSLT επεκτείνεται προκειμένου να ενσωματώσει SPARQL SELECT και ASK ερωτήσεις. Οι SPARQL ερωτήσεις αποτιμούνται πάνω σε RDF δεδομένα και τα αποτελέσματα μετατρέπονται σε XML χρησιμοποιώντας XSLT εκφράσεις.

Σε μερικές άλλες προσεγγίσεις, οι SPARQL ερωτήσεις είναι ενσωματωμένες σε XQuery και XSLT ερωτήσεις [169]. Στο [137, 136, 135], οι XPath εκφράσεις είναι ενσωματωμένες σε SPARQL ερωτήσεις. Αυτές οι προσεγγίσεις προσπαθούν να επεξεργαστούν παράλληλα XML και RDF δεδομένα, και να επωφεληθούν από τον συνδυασμό των χαρακτηριστικών των γλωσσών SPARQL, XQuery, XPath και XSLT. Τέλος, προτείνεται μια μέθοδος στο [137, 136, 135] για τη μετατροπή XML δεδομένων σε RDF και μετάφρασης XPath ερωτήσεων σε SPARQL.

### 5.2.3 Σχολιασμός

Σε αυτό το τμήμα, αναφερόμαστε στις υπάρχουσες προσεγγίσεις, δίνοντας έμφαση στα κύρια ελαττώματα και περιορισμούς. Τα υπάρχοντα συστήματα ολοκλήρωσης δεδομένων (Πίνακας 6.2) δεν υποστηρίζουν τις σημερινές καθιερωμένες τεχνολογίες (π.χ., XML σχήματα, OWL, RDF, SPARQL. Από την άλλη, τα συστήματα ανταλλαγής δεδομένων (Πίνακας 6.3) είναι πιο πρόσφατα και υποστηρίζουν τις σημερινές καθιερωμένες τεχνολογίες, όμως δεν υποστηρίζουν περιπτώσεις ολοκλήρωσης και μηχανισμούς μετάφρασης ερωτήσεων. Αντιθέτως, επικεντρώνονται στη μετάδοση δεδομένων και δεν παρέχουν μηχανισμούς για να εκφράσουν XML ερωτήσεις χρησιμοποιώντας τη γλώσσα ερωτήσεων SPARQL.

Οι πρόσφατες προσεγγίσεις [121, 153, 25, 169, 137, 136, 135, 57, 113] παρουσιάζουν χρηστικά προβλήματα για τους τελικούς χρήστες. Συγκεκριμένα, οι χρήστες αυτών των συστημάτων είναι αναγκασμένοι να: (α) είναι εξοικειωμένοι τόσο με τα SW όσο και με τα XML μοντέλα και γλώσσες, (β) είναι γνώστες των οντολογιών και των XML σχημάτων προκειμένου να εκφράζουν τις ερωτήσεις, και (γ) να είναι γνώστες τις σύνταξης και της σημασιολογίας της κάθε προσέγγισης προκειμένου να εκφράζει τις ερωτήσεις. Επιπλέον, κάθε μια από αυτές τις προσεγγίσεις έχει υιοθετήσει δική της σύνταξη και σημασιολογία τροποποιώντας και/ή συγχωνεύοντας τις καθιερωμένες τεχνολογίες. Αυτές οι τροποποιήσεις μπορεί να έχουν ως αποτέλεσμα προβλήματα συμβατότητας, χρήσης και επεκτασιμότητας. Είναι άξιο παρατήρησης ότι, ως αποτέλεσμα των περιπτώσεων που υιοθετούνται από αυτές τις προσεγγίσεις, έχουν αποτιμηθεί μόνο πάνω σε μικρά σύνολα δεδομένων.

Συγκριτικά με τις πρόσφατες προσεγγίσεις, στο πλαίσιο SPARQL2XQuery που παρουσιάζεται σε αυτήν την εργασία, οι χρήστες (α) δουλεύουν μόνο σε SW τεχνολογίες, (β) δεν χρειάζεται να γνωρίζουν το XML σχήμα ούτε την ύπαρξη των XML δεδομένων, και (γ) εκφράζουν τις ερωτήσεις τους μόνο σε καθιερωμένη (δηλαδή, χωρίς τροποποιήσεις) SPARQL σύνταξη. Τέλος, το πλαίσιο SPARQL2XQuery έχει αποτιμηθεί σε μεγάλα σύνολα δεδομένων.

## 5.3 Μετασχηματισμός Σχήματος

Σε αυτή την παράγραφο, περιγράφουμε τη διαδικασία μετασχηματισμού σχήματος (Σχήμα 5.2), την οποία εκμεταλλευόμαστε στο πρώτο σενάριο χρήσης, προκειμένου

να μετατραπούν αυτόματα τα XML σχήματα σε OWL οντολογίες. Ακολουθώντας τον αυτόματο μετασχηματισμό σχήματος, οι αντιστοιχίσεις μεταξύ XML σχημάτων και OWL οντολογιών δημιουργούνται επίσης αυτόματα και διατηρούνται στο πλαίσιο SPARQL2XQuery. Αυτές οι αντιστοιχίσεις χρησιμοποιούνται αργότερα από άλλα τμήματα του SPARQL2XQuery πλαισίου, για αυτόματη μετάφραση από SPARQL σε XQuery.


**Σχήμα 5.2:** Η διαδικασία μετασχηματισμού με το XS2OWL

Αυτός ο μετασχηματισμός σχήματος πραγματοποιείται χρησιμοποιώντας το τμήμα XS2OWL [329, 309], το οποίο χρησιμοποιεί το *XS2OWL μοντέλο μετασχηματισμού*. Το XS2OWL μοντέλο μετασχηματισμού επιτρέπει την αυτόματη έκφραση του XML σχήματος σε σύνταξη OWL. Επιπλέον, επιτρέπει το μετασχηματισμό των XML δεδομένων σε RDF μορφή και αντίστροφα. Η καινούργια έκδοση του XS2OWL μοντέλου μετασχηματισμού που παρουσιάζεται εδώ, εκμεταλλεύεται τη σημασιολογία του OWL 2 προκειμένου να πετύχει μία πιο ακριβή αναπαράσταση των δομών του XML σχήματος σε OWL σύνταξη. Επιπλέον, υποστηρίζει τις πιο πρόσφατες εκδόσεις των προτύπων (δηλαδή, XML Schema 1.1 και OWL 2). Συγκεκριμένα, οι περιορισμοί ταυτότητας του XML σχήματος (δηλαδή, *key*, *keyref*, *unique*), μπορούν να αναπαρασταθούν με ακρίβεια σε OWL 2 σύνταξη (που δεν ήταν εφικτό με OWL 1.0). Αυτό αντιμετωπίζει τον πιο σημαντικό περιορισμό της προηγούμενης έκδοσης του XS2OWL μοντέλου μετασχηματισμού.

Μια περίληψη της διαδικασίας μετασχηματισμού του XS2OWL απεικονίζεται στο Σχήμα 5.2. Όπως φαίνεται στο Σχήμα 5.2, το τμήμα XS2OWL παίρνει ως είσοδο ένα XML σχήμα $XS$ και δημιουργεί: (α) Μια οντολογία OWL σχήματος $O_S$ που εσωκλείει τις σημασιολογίες του XML Schema, και (β) μια Backwards Compatibility οντολογία $O_{BC}$ η οποία κρατάει τις αντιστοιχίσεις μεταξύ των $O_S$ δομών και των $XS$ δομών. Το $O_{BC}$ περιέχει επίσης τις σημασιολογίες των XML Schema constructs που δεν μπορούν να περιέχονται στο $O_S$ (αφού δεν μπορούν να αναπαρασταθούν από τις OWL σημασιολογίες).

### 5.3.1 Το XS2OWL Μοντέλο Μετασχηματισμού

Σε αυτή την ενότητα, περιγράφουμε το XS2OWL μοντέλο μετασχηματισμού. Μια επίσημη περιγραφή του XS2OWL μοντέλου μετασχηματισμού και λεπτομέρειες εφαρμογής μπορούν να βρεθούν στο [73]. Μια λίστα των αντιστοιχήσεων μεταξύ των δομών του σχήματος XML και των δομών OWL, όπως περιγράφονται στο XS2OWL μοντέλο μετασχηματισμού XS2OWL, παρουσιάζεται στον Πίνακα 6.4.

### 5.3.2 Παράδειγμα Μετασχηματισμού του XML Schema

Παρουσιάζουμε ένα παράδειγμα που δείχνει την έκφραση του XML σχήματος στο OWL χρησιμοποιώντας στοιχεία XS2OWL. Εισάγουμε ένα XML σχήμα (στο οποίο θα αναφέρεται ως *Persons* XML Schema), το οποίο θα χρησιμοποιηθεί σε όλο το υπόλοιπο κεφάλαιο. Το Persons XML Schema παρουσιάζεται στην Εικόνα 6.2 και περιγράφει

**Πίνακας 5.3:** Αντιστοιχίες μεταξύ XML Schema και OWL constructs στο μοντέλο XS2OWL

| XML Schema Construct | OWL Construct |
|---|---|
| Complex Type | Class |
| Simple Datatype | Datatype Definition |
| Element | (Datatype or Object) Property |
| Attribute | Datatype Property |
| Sequence | Unnamed Class – Intersection |
| Choice | Unnamed Class – Union |
| Annotation | Comment |
| Extension, Restriction | subClassOf axiom |
| Unique (*Identity Constraint*) | HasKey axiom ⋆ |
| Key (*Identity Constraint*) | HasKey axiom – ExactCardinality axiom ⋆ |
| Keyref (*Identity Constraint*) | In the Backwards Compatibility Ontology |
| Substitution Group | SubPropertyOf axioms |
| Alternative + | In the Backwards Compatibility Ontology |
| Assert + | In the Backwards Compatibility Ontology |
| Override, Redefine + | In the Backwards Compatibility Ontology |
| Error + | Datatype |

Note. The + indicates the new XML Schema constructs introduced by the XML Schema 1.1 specification. The ⋆indicates the OWL 2 constructs.

**Πίνακας 5.4:** Persons XML Schema complex types στην Schema Ontology ($O_S$)

| XML Schema Complex Types | Ontology Classes | |
|---|---|---|
| | rdf:ID | rdfs:subClassOf |
| Person_Type | Person_Type | owl:Thing |
| Student_Type | Student_Type | Person_Type |
| Persons (*unnamed complex type*) | NS_Persons_UNType | owl:Thing |

τις προσωπικές πληροφορίες για άτομα (Persons) (οι οποίοι μπορεί να είναι φοιτητές). Το στοιχείο ρίζα (root element) *Persons* μπορεί να περιέχει οποιοδήποτε αριθμό στοιχείων Person τύπου *Person_Type*, και οποιοδήποτε αριθμό στοιχείων Student τύπου *Student_Type*.

Οι δομές της οντολογίας σχήματος (Schema ontology) $O_S$ που δημιουργούνται αυτόματα από το XS2OWL για τα Persons XML Schema (θα αναφέρεται ως *Persons* Ontology) παρουσιάζεται στον Πίνακα 6.5 και Πίνακα 6.6.

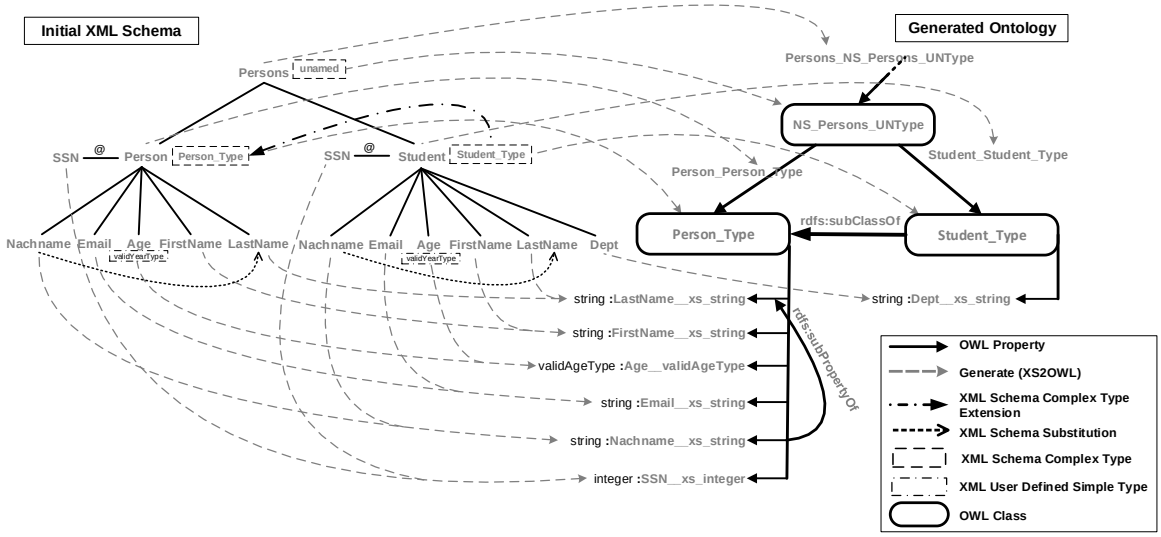
Οι δομές της Backwards Compatibility ontology που δημιουργούνται από το XS2OWL είναι διαθέσιμα στο [73]. Το XML Schema του Σχήμα 6.2 και η Schema ontology $O_S$ που δημιουργήθηκε από το XS2OWL απεικονίζονται στο Σχήμα 6.3.

## 5.4 Μοντέλο Αντιστοιχίσεων

Στο SW, τα OWL–RDF/S έχουν υιοθετηθεί ως schema definition γλώσσες, ενώ στον κόσμο XML, χρησιμοποιείται η γλώσσα XML Schema. Το προτεινόμενο μοντέλο αντιστοιχίσεων ορίζεται στο πλαίσιο της μετάφρασης από SPARQL σε XQuery, για τον ορισμό αντιστοιχίσεων μεταξύ οντολογιών και XML Schemas. Συγκεκριμένα, το μοντέλο αντιστοιχίσεων SPARQL2XQuery προσδιορίζει: (α) τις υποστηριζόμενες αντιστοιχίσεις, (β) την αναπαράσταση των αντιστοιχίσεων, και (γ) τους απαραίτητους

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:complexType name="Person_Type">
    <xs:sequence>
      <xs:element ref="LastName" minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="FirstName" type="xs:string" minOccurs="1" maxOccurs="unbounded"/>
      <xs:element name="Age" type="validAgeType" minOccurs="1" maxOccurs="1" />
      <xs:element name="Email" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="SSN" type="xs:integer"/>
  </xs:complexType>

  <xs:complexType name="Student_Type">
    <xs:complexContent>
      <xs:extension base="Person_Type">
        <xs:sequence>
          <xs:element name="Dept" type="xs:string"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>

  <xs:element name="Persons">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="Person" type="Person_Type" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element name="Student" type="Student_Type" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="LastName" type="xs:string"/>

  <xs:element name="Nachname" substitutionGroup="LastName" type="xs:string"/>

  <xs:simpleType name="validAgeType" >
    <xs:restriction base="xs:float">
        <xs:minInclusive value="0.0"/>
        <xs:maxInclusive value="150.0"/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

**Σχήμα 5.3:** Ένα XML Schema που περιγράφει άτομα (*Persons* XML Schema)




**Σχήμα 5.4:** Το Persons XML Schema (Σχήμα 6.2), η Persons Schema Ontology και οι αντιστοιχίσεις τους

**Πίνακας 5.5:** Persons XML Schema elements και attributes στην Schema Ontology ($O_S$)

| XML Schema Elements & Attributes | Ontology Properties | | | | |
|---|---|---|---|---|---|
| | **Type** | **rdf:ID** | **rdfs:subPropertyOf** | **rdfs:domain** | **rdfs:range** |
| LastName | DTP | LastName__xs_string | — | Person_Type | xs:string |
| FirstName | DTP | FirstName__xs_string | — | Person_Type | xs:string |
| Age | DTP | Age__validAgeType | — | Person_Type | validAgeType |
| Nachname | DTP | Nachname__xs_string | LastName__xs_string | Person_Type | xs:string |
| Email | DTP | Email__xs_string | — | Person_Type | xs:string |
| SSN | DTP | SSN__xs_integer | — | Person_Type | xs:integer |
| Dept | DTP | Dept__xs_string | — | Student_Type | xs:string |
| Person | OP | Person__Person_Type | — | NS_Persons_UNType | Person_Type |
| Student | OP | Student__Student_Type | — | NS_Persons_UNType | Student_Type |
| Persons | OP | Persons__NS_Persons_UNType | — | owl:Thing | NS_Persons_UNType |

τελεστές (operators) για το χειρισμό των αντιστοιχίσεων.

Ορίζουμε το μοντέλο αντιστοιχίσεων προκειμένου να παρέχονται διάφανη πρόσβαση σε XML δεδομένα (transparent XML querying) από το SW περιβάλλον. Στο προτεινόμενο μοντέλο, οι αντιστοιχίσεις μπορούν απλά να θεωρηθούν ως ζευγάρια των ontology constructs (δηλαδή κλάσεις, ιδιότητες) και path expressions πάνω στα XML δεδομένα (δηλαδή, XPath). Οι αντιστοιχίσεις που έχουν οριστεί χρησιμοποιούνται για την μετάφραση των SPARQL ερωτήσεων σε XQuery εκφράσεων. Η χρήση τωνXPath [16] εκφράσεων στο μοντέλο αντιστοιχίσεων, εκτός από την ευρεία αποδοχή του XPath, έχει στόχο την εκμετάλλευση των ιδιοτήτων του (δηλαδή, ευελιξία, εκφραστικότητα), όπως υπογραμμίζονται παρακάτω.

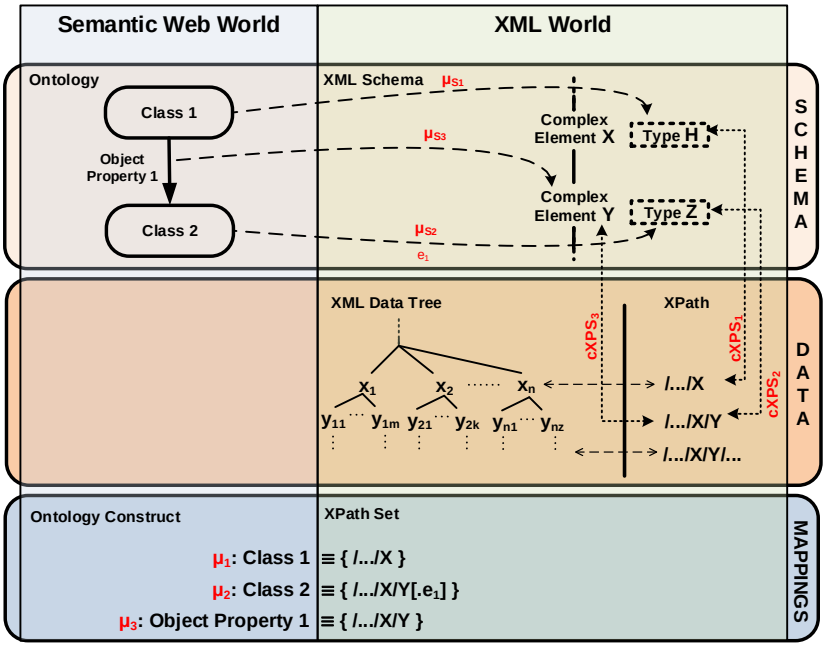
Χρησιμοποιώντας XPath εκφράσεις μπορούμε να υποδείξουμε *με ακρίβεια* τους κόμβους XML. Για παράδειγμα, ας θεωρήσουμε μια αντιστοίχιση που στοχεύει να υποδείξει τα πρόσωπα των οποίων η ηλικία είναι μεταξύ 20 και 30 χρονών (οι ορισμοί των προσώπων ακολουθούν τα σχήματα Persons της Εἰκονας 6.2). Χρησιμοποιώντας το XPath, αυτή η αντιστοιχία μπορεί να εκφραστεί ως $/Persons/Person[./age > 20\ and\ ./age < 30]$.

Ακόμα, η *εκφραστικότητα* του XPath *εξπρεσσιτψ* μέσα από τις συναρτήσεις και τους τελεστες που προσφερει [17], επιτρέπει στο μοντέλο αντιστοιχίσεων να υποστηρίζει ευέλικτες και εκφραστικές εκφράσεις αντιστοιχίσεων.

Η έκφραση των αντιστοιχίσεων ως XPath εκφράσεις μας επιτρέπει να *περιλαμβάνουμε τόσο schema information όσο και data information.* Ως schema information, θεωρούμε την ιεραρχική δομή των δεδομένων που επιβάλλεται από τις εκφράσεις XPath. Ως data information, θεωρούμε συνθήκες πάνω στις τιμές δεδομένων (π.χ., $age > 20$). Η χρήση της ιεραρχικής δομής των δεδομένων επιτρέπει την ελαχιστοποίηση του αριθμού των αντιστοιχίσεων, που έχει ως αποτέλεσμα την μη δημιουργία περιττών ή άσχετων ερωτήσεων.

Τέλος, η χρήση εκφράσεων XPath επιτρέπει τον ορισμό των αντιστοιχίσεων με χρήση άλλων αντιστοιχίσεων (ως "building blocks"). Αυτό το χαρακτηριστικό μπορεί να αξιοποιηθεί στις XQuery εκφράσεις για (α) τη συσχέτιση διαφορετικών μεταβλητών και/ή (β) τη χρήση ήδη αποτιμημένων αποτελεσμάτων. Τα προηγούμενα μπορούν να οδηγήσουν στη δημιουργία αποδοτικών XQuery ερωτήσεων.

Το Σχήμα 6.4 περιγράφει τις σχέσεις μεταξύ των SW (αριστερή πλευρά) και XML (δεξιά πλευρά) worlds. Συγκεκριμένα, παρουσιάζει μια οντολογία, ένα XML Schema και συνδέσεις μεταξύ τους, τόσο στα επίπεδα σχημάτων όσο και δεδομένων (schema and data levels). Στο *schema level* (Ontology/XML Schema), παρατηρούνται οι συνδέσεις μεταξύ των ontology constructs (δηλαδή, κλάσεις, ιδιότητες) και των XML




**Σχήμα 5.5:** Συσχετίσεις μεταξύ XML και SW

Schema constructs (δηλαδή, στοιχείων, σύνθετων τύπων). Επιπλέον, στο *επίπεδο δεδομένων*, τα XML δεδομένα ακολουθούν το XML Schema. Ως αποτέλεσμα, μπορούμε να αναγνωρίσουμε τις XML Schema δομές στα XML data, και να τα αναφερόμαστε σε αυτά χρησιμοποιώντας ένα σύνολο από XPath εκφράσεις. Τέλος, οι αντιστοιχίσεις στα πλαίσια της μετάφρασης από SPARQL σε XQuery) μπορούν απλά να θεωρηθούν ως συνδέσεις μεταξύ των ontology constructs και των XPath εκφράσεων (στο κάτω μέρος του Σχήμα 6.4).

### 5.4.1 Αντιστοιχίσεις Σχήματος

Σε αυτή τη παράγραφο ορίζουμε τα *Schema Mappings*, τα οποία χρησιμοποιούνται για να ορίσουμε συνδέσεις μεταξύ δομών πάνω σε οντολογίες και XML Schemas, στα πλαίσια της μετάφρασης από SPARQL σε XQuery. Στο μοντέλο αντιστοιχίσεων μας, οι αντιστοιχίσεις σχημάτων μπορούν επίσης να εμπλουτιστούν με πληροφορίες δεδομένων επιπέδων (π.χ., συνθήκες σε τιμές δεδομένων), που έχουν ως αποτέλεσμα ακριβείς και ευέλικτες αντιστοιχίσεις. Σημειώνουμε ότι αφού στο πλαίσιο μας οι ερωτήσεις SPARQL που εκφράζονται πάνω σε οντολογίες μεταφράζονται σε XQuery ερωτήσεις που εκφράζονται πάνω σε XML Schemas, τα schema mappings ορίζονται με έναν κατευθυντήριο τρόπο από οντολογίες προς XML Schemas.

Δεδομένης μιας οντολογίας $OL$ και ενός XML Schema $XS$, έστω **oc** να είναι μια ομάδα από $OL$ δομές και **xc** μια ομάδα από $XS$ δομές. Ένα *Schema Mapping* ($\mu_S$) μεταξύ $OL$ και $XS$ είναι μια έκφραση της μορφής:

$$\mu_S: OE \overset{\mathbf{E}}{\mapsto} XE$$

όπου $OE$ είναι μια έκφραση που περιέχει **oc** δομές, τομές ($\wedge$) ή/και ενώσεις ($\vee$), XE είναι μια έκφραση που περιέχει **xc** δομές, τομές ή/και ενώσεις και **E** είναι μια ομάδα συνθηκών που εφαρμόζεται σε **xc** μέλη.

Ένα schema mapping αναπαριστά μια συσχέτιση μεταξύ **oc** και **xc** κάτω από τις συνθήκες που περιγράφονται στο **E**. Μπορούμε απλά να πούμε ότι τα **oc** μέλη αντιστοιχίζονται στα **xc** μέλη υπό τις συνθήκες που περιγράφονται στο **E**. Οι **E** συνθήκες μπορούν απλά να θεωρηθούν ως εκφράσεις δέντρου (tree expressions) που εφαρμόζονται στις **xc** δομές.

Αναλυτικά, μια συνθήκη αντιστοίχισης $e \in \mathbf{E}$ είναι μια *tree expression* που αναφέρεται στις $XS$ δομές και/ή XML δεδομένα που ακολουθούν το $XS$. Συγκεκριμένα, μια συνθήκη αντιστοίχισης $e$ *εφαρμόζεται* σε ένα σύνολο XML Schema constructs $xca \subseteq \mathbf{xc}$ και μπορεί ακόμα να *αναφέρεται* (δηλαδή, να περιλαμβάνει) σε αρκετές δομές ανεξάρτητες στο $xca$. Επιπλέον, μια συνθήκη $e$ μπορεί να περιέχει (α) μονοπάτια δέντρου, (β) operators και συναρτήσεις (π.χ, *intersection*, *union*, <, >, =, ≠, *ends-with*, *concat*), καθώς και (γ) σταθερές (π.χ, 25, 3.4, "John"). Είναι άξιο παρατήρησης το γεγονός ότι κάθε XML Schema construct μπορεί να αναφέρεται σε μια έκφραση συνθήκης (condition expression). Επιπλέον, μια συνθήκη αντιστοίχισης $e$ μπορεί να εφαρμοστεί σε συγκεκριμένες δομές ή σε ολόκληρη $XE$ έκφραση. Ανακεφαλαιώνοντας, ένα schema mapping condition $e$ μπορεί να είναι όποια συνθήκη μπορεί να εκφραστεί σε XPath σύνταξη [16]. Έτσι, η υψηλή εκφραστικότητα των XPath εκφράσεων (συμπεριλαμβανομένου και των ενσωματωμένων συναρτήσεων [17]) μπορεί να χρησιμοποιηθεί σε μια συνθήκη αντιστοίχισης, και, μαζί με την ευελιξία της εφαρμογής ανεξάρτητων συνθηκών σε διαφορετικές XML constructs, οδηγεί σε πλούσια, ευέλικτα και εκφραστικά schema mappings.

Σχετικά με τις ιδιότητες οντολογιών, έστω $pr$ μια ιδιότητα οντολογίας και q ένα στοιχείο XML Schema ή ιδιότητα. Το schema mapping $\mu_S{:}pr \mapsto q$ αντιστοιχεί $pr.domain \mapsto d$ και $pr.range \mapsto q$, όπου $d$ είναι ένα (σύνθετο) XML στοιχείο στο οποίο έχει οριστεί το $q$. Επιπλέον, το domain και το εύρος της ιδιότητας οντολογίας $pr$, μπορεί να αντιστοιχηθεί ατομικά σε διαφορετικά XML Schema στοιχεία/ιδιότητες. Για παράδειγμα, έστω $q$, $v$ XML Schema στοιχεία/ιδιότητες, τότε $\mu_{S_1}{:}pr.domain \mapsto q$ και $\mu_{S_2}{:}pr.range \mapsto v$.

### 5.4.2 Αντιστοιχίσεις μεταξύ XML Schema Constructs και XPath Sets — Συσχέτιση Σχήματος και Δεδομένων

Έχουμε ήδη ορίσει τα schema mappings μεταξύ των ontology constructs και XML Schema constructs. Αφού θέλουμε να μεταφράσουμε SPARQL ερωτήσεις σε XQuery εκφράσεις που θα αποτιμηθούν σε XML δεδομένα, πρέπει να αναγνωρίσουμε αντιστοιχίσεις μεταξύ των ontology constructs (που αναφέρονται στις SPARQL ερωτήσεις) και των XML δεδομένων, με βάση τα schema mappings που ορίστηκαν. Σε αυτή τη παράγραφο προσπαθούμε να εκφράσουμε τις σχέσεις μεταξύ των XML Schema constructs και των XML κόμβων δεδομένων χρησιμοποιώντας εκφράσεις συνόλων XPath.

Στο επίπεδο των δεδομένων, τα XML δεδομένα είναι έγκυρα με βάση το XML Schema(s) που ακολουθούν. Έτσι, για κάθε XML Schema construct μπορούμε να αναγνωρίσουμε τους αντίστοιχους XML κόμβους δεδομένων και να απευθυνθούμε σε αυτούς χρησιμοποιώντας XPath εκφράσεις. Με αυτόν τον τρόπο, μπορούμε να ορίσουμε τις συσχετίσεις μεταξύ των XML schema constructs και XML δεδομένων.

Με δεδομένη μια SPARQL ερώτηση, για όλα τα ontology constructs που αναφέρονται στην ερώτηση: (α) αναγνωρίζουμε τα XML Schema constructs που αναφέρονται στα schema mappings που έχουν οριστεί, και (β) καθορίζουμε τα αντίστοιχα XPath σύνολα για τα αναγνωρισμένα XML Schema constructs. Ως αποτέλεσμα, τα ontology

constructs που αναφέρεται στην SPARQL ερώτηση σχετίζονται άμεσα με τα XML δεδομένα μέσω των XPaths.

Τυπικά, έστω $D$ ένα XML σύνολο δεδομένων, έγκυρο με βάση ένα XML Schema $XS$. Μια συσχέτιση του XML Schema construct $xc$ με ένα XPath σύνολο $xps$ είναι μια συνάρτηση $cXPS$: $\mathbf{XC} \longrightarrow \mathbf{XPS}$ που αναθέτει το XPath σύνολο $xps \in \mathbf{XPS}$ στην XML construct $xc \in \mathbf{XC}$, όπου το $xps$ απευθύνεται σε όλους τους αντίστοιχους XML κόμβους του $xc$ στο $D$.

Ο Πίνακας 6.7 συνοψίζει τις αντιστοιχίσεις μεταξύ του "XML part" (που αναφέρεται δηλαδή στις XML Schema constructs) των schema mapping εκφράσεων και των XPath συνόλων.

**Πίνακας 5.6:** Αντιστοιχίσεις μεταξύ Schema mapping και XPath Sets

| **Schema Mapping Expression** | **XPath Set Correspondences** |
|---|---|
| $w$ | $cXPS(w)$ |
| $e \bigvee p$ | $xe \bigcup xp$ |
| $e \bigwedge p$ | $xe \bigcap xp$ |
| $w\langle\!\langle ce \rangle\!\rangle$ | $cXPS(w)[xce]$ |
| $w \bigvee z$ | $cXPS(w) \bigcup cXPS(z)$ |
| $w \bigwedge z$ | $cXPS(w) \bigcap cXPS(z)$ |
| $(w \bigvee z)\langle\!\langle ce \rangle\!\rangle$ | $cXPS(w)[xce] \bigcup cXPS(z)[xce]$ |
| $(w \bigwedge z)\langle\!\langle ce \rangle\!\rangle$ | $cXPS(w)[xce] \bigcap cXPS(z)[xce]$ |

### 5.4.3 Αναπαράσταση της Αντιστοίχισης Σχήματος

Σε προηγούμενες παραγράφους έχουμε ορίσει τις σχέσεις στο επίπεδο των δεδομένων (δηλαδή, Schema Mappings – Ενότητα 5.4.1), καθώς και σχέσεις μεταξύ των XML Schema και των XML δεδομένων (δηλαδή, αντιστοιχίσεις μεταξύ των XML Schema Constructs και των XPath Sets – Ενότητα 5.4.2). Εδώ χρησιμοποιούμε αυτές τις σχέσεις προκειμένου να ορίσουμε και να αναπαραστήσουμε τα schema mappings στα πλαίσια της μετάφρασης από SPARQL σε XQuery .

Τυπικά, δεδομένης μιας οντολογίας $OL$ και ενός XML Schema $XS$, έστω **oc** ΄ένα σύνολο από $OL$ δομές, **xc** ένα σύνολο από $XS$ δομές και $\mu_S$ μια αντιστοίχιση σήματος μεταξύ **oc** και **xc**. Μία *αντιστοίχιση* ($\mu$) μεταξύ $OL$ και $XS$ στα πλαίσια της μετάφρασης από SPARQL σε XQuery είναι μια έκφραση της μορφής:

$\mu$: $oc \equiv sxps$, όπου $sxps$ είναι ένα XPath Set που αντιστοιχεί σε **xc** δομές κάτω από τις αντιστοιχίσεις σχήματος $\mu_S$.

Στο υπόλοιπο κεφάλαιο, για κάθε κλάση οντολογίας c, το συσχετιζόμενο XPath Set συμβολίζεται ως $\mathbf{X}_c$ (*Class XPath Set*). Επιπλέον, για κάθε ιδιότητα οντολογίας $pr$, το συσχετιζόμενο XPath Set συμβολίζεται ως $\mathbf{X}_{pr}$ (*Property XPath Set*). Επίσης, για τους $pr$ domains και τα εύρη, τα συσχετιζόμενα XPath Sets συμβολίζονται ως $\mathbf{X}_{prD}$ (*Property Domains XPath Set*) και $\mathbf{X}_{prR}$ (*Property Ranges XPath Set*) αντίστοιχα.

### 5.4.4 Αυτόματη Παραγωγή Αντιστοιχίσεων

Στο πρώτο SPARQL2XQuery σενάριο, οι αντιστοιχίσεις δημιουργούνται αυτόματα. Συγκεκριμένα, η δημιουργία αντιστοιχίσεων διεξάγεται από το στοιχείο *Mapping Generator*. Το στοιχείο αυτό καθορίζει τις *αντιστοιχίες μεταξύ των XML Schema construct και του XPath Set* για όλα τα XML constructs. Τέλος, το στοιχείο δημιουργεί

ένα XML αρχείο που περιέχει τις σχέσεις όλων των ontology constructs με τα XPath Sets. Συγκεκριμένα, δημιουργεί τα σύνολα $\mathbf{X}_c$, $\mathbf{X}_{pr}$, $\mathbf{X}_{prD}$ και $\mathbf{X}_{prR}$ για όλες τις κλάσεις και τις ιδιότητες.

**Παράδειγμα 1.** Θεωρούμε το XML Schema του Σχήμα 6.2 και την αντίστοιχη οντολογία που δημιουργείται από το XS2OWL (Πίνακας 6.5 και Πίνακας 6.6). Βασισμένο στις αυτομάτως καθορισμένες αντιστοιχίσεις σχημάτων (Σχήμα 6.3), το στοιχείο Mapping Generator δημιουργεί τις αναπαραστάσεις αντιστοιχίσεων που παρατίθενται παρακάτω.

**Generated Mappings between the XML Schema and the Ontology of Figure 4**

***Classes:***

Person_Type ≡ $X_{Person\_Type}$ = { /Persons/Person }

Student_Type ≡ $X_{Student\_Type}$ = { /Persons/Student }

NS_Persons_UNType ≡ $X_{NS\_Persons\_UNType}$ = { /Persons }

***Object Properties:***

Persons__NS_Persons_UNType ≡ $X_{Persons\_\_NS\_Persons\_UNType}$ = { /Persons }
Persons__NS_Persons_UNType.domain ≡ $X_{Persons\_\_NS\_Persons\_UNTypeD}$ = = { /Persons }
Persons__NS_Persons_UNType.range ≡ $X_{Persons\_\_NS\_Persons\_UNTypeR}$ = = { /Persons }

Person__Person_Type ≡ $X_{Person\_\_Person\_Type}$ = { /Persons/Person }
Person__Person_Type.domain ≡ $X_{Person\_\_Person\_TypeD}$ = { /Persons }
Person__Person_Type.range ≡ $X_{Person\_\_Person\_TypeR}$ = { /Persons/Person }

Student__Student_Type ≡ $X_{Student\_\_Student\_Type}$ = { /Persons/Student }
Student__Student_Type.domain ≡ $X_{Student\_\_Student\_Type\ D}$ = { /Persons }
Student__Student_Type.range ≡ $X_{Student\_\_Student\_TypeR}$ = = { /Persons/Student }

***Datatype Properties:***

FirstName__xs_string ≡ $X_{FirstName\_\_xs\_string}$={ /Persons/Person/FirstName, /Persons/Student/FirstName }
FirstName__xs_string.domain ≡ $X_{FirstName\_\_xs\_stringD}$={ /Persons/Person, /Persons/Student }
FirstName__xs_string.range ≡ $X_{FirstName\_\_xs\_stringR}$={ /Persons/Person/FirstName, /Persons/Student/FirstName }

LastName__xs_string ≡ $X_{LastName\_\_xs\_string}$ = { /Persons/Person/LastName, /Persons/Student/LastName }
LastName__xs_string.domain ≡ $X_{LastName\_\_xs\_stringD}$ = { /Persons/Person, /Persons/Student }
LastName__xs_string.range ≡ $X_{LastName\_\_xs\_stringR}$ ={ /Persons/Person/LastName, /Persons/Student/LastName }

Age__xs_integer ≡ $X_{Age\_\_xs\_integer}$ = { /Persons/Person/Age, /Persons/Student/Age}
Age__xs_integer.domain ≡ $X_{Age\_\_xs\_integerD}$ = { /Persons/Person, /Persons/Student }
Age__xs_integer.range ≡ $X_{Age\_\_xs\_integerR}$ = { /Persons/Person/Age, /Persons/Student/Age}

Email__xs_string ≡ $X_{Email\_\_xs\_string}$ = { /Persons/Person/Email, /Persons/Student/Email}
Email__xs_string.domain ≡ $X_{Email\_\_xs\_stringD}$ = { /Persons/Person, /Persons/Student }
Email__xs_string.range ≡ $X_{Email\_\_xs\_stringR}$ = { /Persons/Person/Email, /Persons/Student/Email }

Nachname__xs_string ≡ $X_{Nachname\_\_xs\_string}$ = { /Persons/Person/Nachname, /Persons/Student/Nachname }
Nachname__xs_string.domain ≡ $X_{Nachname\_\_xs\_stringD}$ = { /Persons/Person, /Persons/Student }
Nachname__xs_string.range ≡ $X_{Nachname\_\_xs\_stringR}$ ={ /Persons/Person/Nachname, /Persons/Student/Nachname }

SSN__xs_integer ≡ $X_{SSN\_\_xs\_integer}$ = { /Persons/Person/@SSN, /Persons/Student/@SNN}
SSN__xs_integer.domain ≡ $X_{SSN\_\_xs\_integerD}$ = { /Persons/Person, /Persons/Student }
SSN__xs_integer.range ≡ $X_{SSN\_\_xs\_integerR}$ ={ /Persons/Person/@SSN, /Persons/Student/@SNN}

Dept__xs_string ≡ $X_{Dept\_\_xs\_string}$ = { /Persons/Student/Dept }
Dept__xs_string.domain ≡ $X_{Dept\_\_xs\_stringD}$ = { /Persons/Student }
Dept__xs_string.range ≡ $X_{Dept\_\_xs\_stringR}$ = { /Persons/Student/Dept }

□

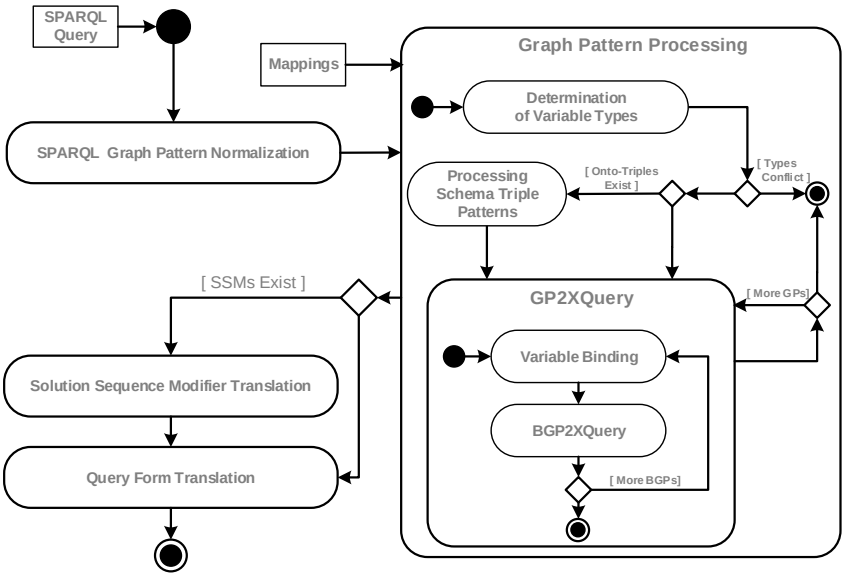
## 5.5 Εισαγωγή στη Διαδικασία Μετάφρασης Ερωτήσεων

### 5.5.1 Εισαγωγικά

Εδώ εισάγουμε κάποιες σημαντικές έννοιες της μετάφρασης ερωτήσεων. Έστω $\mathbf{I}_{RDF}$ ένα σύνολο που περιέχει τα IRIs του RDF vocabulary (π.χ, rdf:type, rdf:Property), $\mathbf{I}_{RDFS}$ το σύνολο που περιέχει τα IRIs του RDF Schema vocabulary (π.χ, rdfs:subClassOf, rdfs:domain) και $\mathbf{I}_{OWL}$ το σύνολο που περιέχει τα IRIs του OWL vocabulary (π.χ, owl:equivalentClass, owl:FunctionalProperty). Επιπλέον, έστω $\mathbf{I}_{CL}$ το σύνολο που περιέχει τα IRIs των κλάσεων μιας οντολογίας και $\mathbf{I}_{PR}$ το σύνολο που περιέχει τα IRIs των ιδιοτήτων μιας οντολογίας.

Από τα παραπάνω σύνολα, ορίζουμε το σύνολο $\mathbf{I}_{VC}$, που περιέχει όλα τα IRIs των RDF/S και των OWL vocabularies $\mathbf{I}_{VC} = \mathbf{I}_{RDF} \bigcup \mathbf{I}_{RDFS} \bigcup \mathbf{I}_{OWL}$. Επιπλέον, ορίζουμε το σύνολο $\mathbf{I}_{OL}$, που περιέχει τα IRIs που αναφέρονται σε κλάσεις οντολογιών και ιδιότητες $\mathbf{I}_{OL} = \mathbf{I}_{CL} \bigcup \mathbf{I}_{PR}$.

**Ορισμός 1. (Schema Triple Pattern)** Το *Schema Triple Pattern* είναι ένα triple pattern το οποίο αναφέρεται σε σε δομή και/ή σε σημασιολογίας της οντολογίας. Συγκεκριμένα, το Schema Triple Pattern είναι ένα triple pattern που περιέχει




**Σχήμα 5.6:** UML διάγραμμα το οποίο περιγράφει την διαδικασία της μετάφρασης

ιδιότητες των RDF/S και OWL vocabularies, ή ένα triple pattern που έχει IRIs που αναφέρονται σε κλάσεις οντολογιών ή ιδιότητες. Τυπικά, ένα Schema Triple Pattern ορίζεται ως εξής:

Μια τριπλέτα $\langle s, p, o\rangle \in (\mathbf{I}_{VC} \cup \mathbf{I}_{OL} \cup \mathbf{B} \cup \mathbf{V}) \times (\mathbf{I}_{VC} \cup \mathbf{I}_{OL}) \times (\mathbf{I}_{VC} \cup \mathbf{I}_{OL} \cup \mathbf{B} \cup \mathbf{L} \cup \mathbf{V})$ που ονομάζεται *Schema Triple Pattern* (ή απλά *Schema Triple*).

Χρησιμοποιούμε το $schemaTr(gp)$ για να δηλώσουμε ένα σύνολο από Schema Triples που προκύπτουν σε ένα graph pattern $gp$.

### 5.5.2 Σύνοψη της μετάφρασης ερωτήσεων

Σε αυτή τη παράγραφο παρουσιάζουμε μια σύνοψη της διαδικασίας μετάφρασης από SPARQL σε XQuery, η οποία πραγματοποιείται από το στοιχείο Query Translator.

Το Σχήμα 5.6 παρουσιάζει, στη μορφή διαγράμματος UML, ολόκληρη τη διαδικασία μετάφρασης. Όπως φαίνεται στο Σχήμα 5.6, κατά τη διαδικασία της μετάφρασης έχουμε ως είσοδο την SPARQL ερώτηση και τις αντιστοιχίσεις μεταξύ της οντολογίας και XML Schema. Η δραστηριότητα *"The SPARQL Graph Pattern Normalization"* (Παράγραφος 5.6.1) μετασχηματίζει το Graph Pattern (GP) της SPARQL ερωτήσεις σε μια ισοδύναμη normal form, που έχει ως αποτέλεσμα πιο απλή και πιο αποδοτική διαδικασία μετάφρασης. Ακολουθεί η επεξεργασία του *Graph Pattern*, προκειμένου να μεταφραστεί το Graph Pattern της SPARQL ερώτησης (ο όρος Where της ερώτησης) σε XQuery εκφράσεις. Έπειτα, οι solution sequence modifiers (SSMs) που ενδεχομένως να περιλαμβάνονται στην ερώτηση μεταφράζονται (Παράγραφος 5.9.1). Τέλος, με βάση τη μορφή της SPARQL ερώτησης, το δημιουργημένο XQuery εμπλουτίζεται με κατάλληλες εκφράσεις προκειμένου να επιτευχθεί η επιθυμητή δομή των αποτελεσμάτων (π.χ., είτε να δημιουργηθεί ένα RDF γράφημα είτε να επιστραφεί μία SPARQL ακολουθία επίλυσης) (Παράγραφος 5.9.2).

Η επεξεργασία του Graph Pattern είναι μια σύνθετη δραστηριότητα με αρκετές υπο-δραστηριότητες (Σχήμα 5.6). Αρχικά, μία δραστηριότητα τακτοποιεί τους τύπους των SPARQL μεταβλητών, προκειμένου να καθορίσει τη μορφή των αποτελεσμάτων καθώς και να πραγματοποιήσει τους έλεγχους συνοχής στη χρήση των μεταβλητών

(variable usage) (Παράγραφος 5.6.2). Έπειτα, μία δραστηριότητα (Παράγραφος 5.6.3) επεξεργάζεται τα Schema Triples (Ορισμός 1) που ενδεχομένως να υπάρχουν στο pattern και καθορίζει τις συνδέσεις μεταβλητών (δηλαδή, αναθέτει τα κατάλληλα XPaths στις μεταβλητές). Αυτές οι συνδέσεις θα χρησιμοποιηθούν στα επόμενα στάδια ως αρχικές συνδέσεις μεταβλητών. Τέλος, χρησιμοποιείται ένας αλγόριθμος μετάφρασης (*GP2XQuery*) που μεταφράζει τα GPs σε XQuery εκφράσεις (Παράγραφος 5.8). Κατά τη διάρκεια της GP2XQuery μετάφρασης, για κάθε Basic Graph Pattern (BGP) που περιέχεται στο GP, πραγματοποιείται ένα στάδιο σύνδεσης μεταβλητών (Παράγραφος 5.7) και μια μετάφραση από BGP σε XQuery (Παράγραφος 5.8.1).

## 5.6 Κανονικοποίηση Ερωτήσεων, Τύποι Μεταβλητών & Schema Triples

### 5.6.1 Κανονικοποίηση Σχηματομορφών Γράφων (Graph Pattern

Σε αυτή τη παράγραφο, περιγράφουμε το στάδιο της κανονικοποίσης του *SPARQL Graph Pattern*, που μετασχηματίζει το graph pattern (GP) μιας ερώτησης SPARQL, και το μετατρέπει σε μια ισοδύναμη φυσιολογική μορφή (normal form). Η κανονικοποίηση του SPARQL graph pattern βασίζεται στις GP εκφράσεις ισοδυναμίας που αποδείχτηκαν στο [272] και σε τεχνικές μετασχηματισμού ερωτήσεων.

Η νέα GP φυσιολογική μορφή (normal form) επιτρέπει μια ευκολότερη και πιο αποδοτική διαδικασία μετάφρασης, καθώς και τη δημιουργία πιο αποδοτικών ερωτήσεων XQuery αφού: (α) Η normal form περιέχει μια σειρά από union–free graph patterns, κάθε ένα από τα οποία μπορεί να τύχει επεξεργασίας ανεξάρτητα. (β) Η normal form περιέχει μεγαλύτερα Basic Graph Patterns. Μεγαλύτερα basic graph patterns συνεπάγονται πιο αποδοτική διαδικασία μετάφρασης, καθώς μειώνουν τον αριθμό των συνδέσεων μεταβλητών (variable bindings), καθώς και των διαδικασιών μετάφρασης από BGP σε XQuery που απαιτούνται (περισσότερες λεπτομέρειες μπορούν να βρεθούν στο τμήμα 5.8). (γ) Τα μεγαλύτερα basic graph patterns έχουν ως αποτέλεσμα περισσότερες διαδοχικές συζεύξεις (sequential conjunctions) (δηλαδή ANDs) οι οποίες χειρίζονται εγγενώς από εκφράσεις XQuery, επομένως πιο αποδοτικές ερωτήσεις XQuery (περισσότερες λεπτομέρειες μπορούν να βρεθούν στο τμήμα 5.8). Σημειώνουμε ότι στις περισσότερες περιπτώσεις, τα "real-world" (δηλαδή, user defined) SPARQL graph patterns εκφράζονται αρχικά σε normal form [275], άρα αυτή η φάση συχνά αποφεύγεται.

### 5.6.2 Προσδιορισμός των Τύπων των Μεταβλητών

Σε αυτό το τμήμα περιγράφουμε το στάδιο του *προσδιορισμού των τύπων των μεταβλητών*. Σε αυτό το στάδιο ταυτοποιείται ο τύπος των SPARQL μεταβλητών που αναφέρονται στο union–free graph pattern (UF–GP). Οι καθορισμένοι τύποι μεταβλητών χρησιμοποιούνται για να προσδιορίσουν τη μορφή των αποτελεσμάτων, και επομένως τη σύνταξη του Return XQuery clause. Επιπλέον, οι τύποι μεταβλητών χρησιμοποιούνται για τη δημιουργία πιο αποδοτικών XQuery εκφράσεων. Συγκεκριμένα, οι τύποι μεταβλητών χρησιμοποιούνται από τα στάδιο επεξεργασίας των Schema Triple patterns και το στάδιο σύνδεσης μεταβλητών, προκειμένου να μειωθούν οι συνδέσεις περικόπτοντας

τις περιττές. Τέλος, μέσα από τον προσδιορισμό των τύπων των μεταβλητών, πραγματοποιείται έλεγχος συνέπειας στη χρήση των μεταβλητών, προκειμένου να εντοπιστούν πιθανά προβλήματα (π.χ. η ίδια μεταβλητή να ορίζεται πολλές φορές ως άλλου τύπου στο ίδιο UF–GP). Σε μια τέτοια περίπτωση, η UF– GP δεν μπορεί να αντιστοιχηθεί με κανένα RDF dataset, επομένως, αυτό το UF–GP περικόπτεται και δεν μεταφράζεται με αποτέλεσμα πιο αποδοτικές XQuery ερωτήσεις που επιταχύνουν τη διαδικασία της μετάφρασης. Στον Πίνακα 5.7 ορίζουμε τον τύπο μεταβλητών που μπορεί να προκύψουν στα triple patterns.

**Πίνακας 5.7:** Τύποι Μεταβλητών

| Notation | Name | Description |
|---|---|---|
| *CIVT* | Class Instance Variable Type | Represents class instance variables |
| *LVT* | Literal Variable Type | Represents literal value variables |
| *UVT* | Unknown Variable Type | Represents unknown type variables |
| *DTPVT* | Data Type Predicate Variable Type | Represents data type predicate variables |
| *OPVT* | Object Predicate Variable Type | Represents object predicate variables |
| *UPVT* | Unknown Predicate Variable Type | Represents unknown predicate variables |

#### 5.6.2.1 Κανόνες Προσδιορισμού Τύπου Μεταβλητών

Εδώ περιγράφουμε τους κανόνες που χρησιμοποιούνται για τον προσδιορισμό του τύπου των μεταβλητών. Έστω $OL$ μια οντολογία, UF–GP ένα Union–Free Graph Pattern που εκφ ράζεται σε $OL$, έστω **mDTP** ένα σύνολο από αντιστοιχισμένες ιδιότητες συνόλου δεδομένων (mapped datatype properties) του $OL$, έστω **mOP** ένα σύνολο από mapped object properties του $OL$, έστω $\mathbf{V}_{UFGP}$ ένα σύνολο από UF–GP μεταβλητές[1] και έστω $\mathbf{L}_{UFGP}$ (*UF–GP Literal Set*) ένα σύνολο από τα literals που αναφέρονται στην UF–GP.

Ο προσδιορισμός του τύπου των μεταβλητών είναι μια συνάρτηση $VarType$: $\mathbf{V}_{UFGP} \longrightarrow \mathbf{VT}$ που αναθέτει ένα τύπο μεταβλητής $vt \in \mathbf{VT}$ σε κάθε μεταβλητή $v \in \mathbf{V}_{UFGP}$, όπου $\mathbf{VT} = \{CIVT, LVT, UVT, DTPVT, OPVT, UPVT\}$ περιλαμβάνει όλους τους τύπους μεταβλητών. Η σχέση μεταξύ του domain και του εύρους της συνάρτησης $VarType$ προσδιορίζονται από τους κανόνες προσδιορισμού που παρουσιάζονται παρακάτω.

Εδώ, απαριθμούμε τους κανόνες προσδιορισμού που εφαρμόζονται επαναληπτικά για κάθε τριπλέτα στο δοθέν UF–GP. Το τελικό αποτέλεσμα των κανόνων δεν επηρεάζεται από τη σειρά με την οποία εφαρμόζονται οι κανόνες ούτε και από τη σειρά με την οποία αναλύονται τα triple patterns. Ως $T_x$ συμβολίζουμε τον τύπο της μεταβλητής $x$.

Με δεδομένο ένα (non-Schema) triple pattern $t \in \langle s, p, o \rangle$, όπου $s$ το μέρος υποκειμένου , $p$ το μέρος κατηγορούμενου και $o$ το μέρος αντικειμένου, ορίζουμε τους ακόλουθους κανόνες:

**Κανόνας 1:** Αν $s \in \mathbf{V}_{UFGP} \Longrightarrow T_s = CIVT$. Αν το υποκείμενο είναι μεταβλητή, τότε ο τύπος της θα είναι Μεταβλητή Τύπου Στιγμιότυπου Κλάσης (*Class Instance Variable Type-CIVT*).

[1]Το σύνολο $\mathbf{V}_{UFGP}$ δεν περιλαμβάνει τις μεταβλητές που προκύπτουν μόνο στα Schema triple patterns, αφού τα Schema triple patterns προκύπτουν από το στάδιο προσδιορισμού του τύπου των μεταβλητών)

**Κανόνας 2:** Αν $p \in \mathbf{DTP}$, και $o \in \mathbf{V}_{UFGP} \Longrightarrow T_o = LVT$.Αν το κατηγόρημα είναι ιδιότητα τύπων δεδομένων, και το αντικείμενο είναι μεταβλητή, ο τύπος της θα είναι Μεταβλητή Τύπου Σταθεράς (*Literal Variable Type-LVT*).

**Κανόνας 3:** Αν $p \in \mathbf{mOP}$, και $o \in \mathbf{V}_{UFGP} \Longrightarrow T_o = CIVT$. Αν το κατηγόρημα είναι ιδιότητα αντικειμένων, και το αντικείμενο είναι μεταβλητή, ο τύπος της θα είναι Μεταβλητή Τύπου Στιγμιότυπου Κλάσης (*Class Instance Variable Type- CIVT*).

**Κανόνας 4:** $T_p = DTPVT \Longleftrightarrow T_o = LVT \mid p, o \in \mathbf{V}_{UFGP}$. Αν το κατηγόρημα είναι μεταβλητή και είναι τύπου Μεταβλητή Ιδιότητας Τύπου Θεδομένων, τότε αν και το αντικείμενο είναι μεταβλητή ο τύπος της θα είναι Μεταβλητή Τύπου Σταθεράς (*Literal Variable Type-LVT*). Επίσης ισχύει και το αντίστροφο.

**Κανόνας 5:** $T_p = OPVT \Longleftrightarrow T_o = CIVT \mid p, o \in \mathbf{V}_{UFGP}$. Αν το κατηγόρημα είναι μεταβλητή και είναι τύπου Μεταβλητή Ιδιότητας Τύπου Αντικειμένων, τότε αν και το αντικείμενο είναι μεταβλητή θα είναι τύπου Μεταβλητή Τύπου Στιγμιότυπου Κλάσης (*Class Instance Variable Type-CIVT*). Επίσης ισχύει και το αντίστροφο.

**Κανόνας 6:** Αν $o \in \mathbf{L}_{UFGP}$ και $p \in \mathbf{V}_{UFGP} \Longrightarrow Tp = DTPVT$. Αν το αντικείμενο είναι σταθερά, τότε αν το κατηγόρημα είναι μεταβλητή, θα είναι τύπου Μεταβλητή Τύπου Στιγμιότυπου Σταθεράς (*Data Type Predicate Variable Type- DTPVT*).

Το στάδιο του καθορισμού του τύπου δεδομένων, συμπεριλαμβανομένου της αρχικοποίησης των μεταβλητών, των κανόνων καθορισμού και τον έλεγχο, παρουσιάζεται στον αλγόριθμο στο [73].

### 5.6.3 Επεξεργασία Schema Triple Pattern

Σε αυτή την ενότητα παρουσιάζουμε την *επεξεργασία του schema triple pattern*. Αυτό το στάδιο πραγματοποιείται προκειμένου να υποστηριχτούν οι schema-based ερωτήσεις. Αφού οι schema-based ερωτήσεις θεωρούνται ερωτήσεις που περιέχουν triple patterns που αναφέρονται στη δομή της οντολογίας και/ή στη σημασιολογία (δηλαδή, *Schema Triple Patterns*, Ορισμος 1). Στα πλαίσια της επεξεργασίας schema triple pattern, τα Schema Triple Patterns που περιέχονται στην ερώτηση επεξεργάζονται με βάση την οντολογία έτσι ώστε οι πληροφορίες του σχήματος να μπορούν να χρησιμοποιηθούν κατά τη διάρκεια της μετάφρασης.

Αρχικά, ontology constructs αντιστοιχίζονται στις μεταβλητές που περιέχονται στα Schema Triples. Έπειτα, χρησιμοποιώντας τις προκαθορισμένες αντιστοιχίσεις, τα ontology constructs αντικαθίστανται με τα αντίστοιχα XPath Sets. Ως αποτέλεσμα αυτής της επεξεργασίας, τα XPaths είναι αντιστοιχισμένα με τις μεταβλητές που περιέχονται στα Schema Triples. Αυτές οι συνδέσεις θα χρησιμοποιηθούν ως αρχικές συνδέσεις (bindings) για το στάδιο της σύνδεσης μεταβλητών (variable binding) (Ενότητα 5.7). Σημειώνουμε ότι όπως αναφέρεται στον Ορισμό 1, τα triple patterns που έχουν μια μεταβλητή στο κατηγόρημα δεν ορίζονται ως schema triples, αφού μπορούν να χειρίζονται τόσο δεδομένα όσο και πληροφορίες σχήματος. Έτσι, αυτά τα triples θεωρούνται non-schema triple patterns.

## 5.7 Σύνδεση Μεταβλητών

Σε αυτή την ενότητα, περιγράφουμε το στάδιο της *σύνδεσης μεταβλητών* (Variables Binding). Στο δικό μας πλαίσιο, ο όρος σύνδεση μεταβλητών (variable bindings) χρησιμοποιείται για να περιγράψει την ανάθεση XPaths στις μεταβλητές που αναφέρονται σε ένα BGP, επιτρέποντας έτσι τη μετάφραση των BGPs σε XQuery εκφράσεις.

Διαισθητικά, αυτό το στάδιο θεωρεί τις δομές γράφων που δημιουργήθηκε από τα triples patterns που ορίζονται από τα BGP, καθώς και τις αντιστοιχίσεις, προκειμένου να καθοριστεί το κατάλληλο σύνολο από συνδέσεις. Αυτό το σύνολο από συνδέσεις πρόκειται να χρησιμοποιηθεί στη δημιουργία των XQuery εκφράσεων. Πρέπει να σημειωθεί ότι, λόγω της μορφής των αντιστοιχίσεων (δηλαδή, XPaths Sets) η (ιεραρχική) δομή του XML data λαμβάνεται επίσης υπόψιν από το στάδιο σύνδεσης των μεταβλητών.

Επιπλέον, πληροφορίες σχήματος και/ή σημασιολογίες που πιθανόν να εκφράζονται στις ερωτήσεις SPARQL χρησιμοποιούνται στο στάδιο σύνδεσης μεταβλητών χρησιμοποιώντας τις συνδέσεις που καθορίζονται από το στάδιο επεξεργασίας Schema Triple (Ενότητα 5.6.3). Για αυτό το λόγο, τα Schema Triples παραλείπονται (δηλαδή, περικόπτονται) από αυτό το στάδιο και οι καθορισμένες Schema Triple συνδέσεις χρησιμοποιούνται ως αρχικές συνδέσεις.

### 5.7.1 Αλγόριθμος Σύνδεσης Μεταβλητών

#### 5.7.1.1 Εισαγωγικά

Ένα RDF triple $\langle s, p, o \rangle$ είναι ένα υπογράφημα του κατευθυνόμενου RDF γράφου, όπου $s$, $o$ είναι κόμβοι και $p$ είναι μια ακμή του κατευθυνόμενου γράφου, με κατεύθυνση από το $s$ στο $o$. Ως $\mathbf{X}_s$, $\mathbf{X}_p$ και $\mathbf{X}_o$ συμβολίζουμε τα XPath Set που αντιστοιχούν στο υποκείμενο, κατηγόρημα και αντικείμενο των XPath Sets, αντίστοιχα. Επιπλέον, έστω $\mathbf{X}_{pD}$ και $\mathbf{X}_{pR}$ τα XPath Sets που αντιστοιχούν, στα predicate domains και στα range αντίστοιχα.

Σκεπτόμενοι την ιεραρχική δομή των XML data και τη δομή του κατευθυνόμενου RDF γράφου, πρέπει να τηρούνται οι ακόλουθες σχέσεις για τα XPath Sets των triple pattern τμημάτων:

(α′) $\exists x_s \in \mathbf{X}_s$ και $\exists x_{pD} \in \mathbf{X}_{pD} : x_s \,\tilde{\sqsubset}\, x_{pD}$. Το XPath Set του υποκειμένου ($\mathbf{X}_s$) περιέχει XPaths που προτάσσουν (prefix) τα XPaths που περιέχονται στα predicate domains XPath Set ($\mathbf{X}_{pD}$).

(β′) $\exists x_{pD} \in \mathbf{X}_{pD}$ ανδ $\exists x_{pR} \in \mathbf{X}_{pR} : x_{pD} \,\tilde{\sqsubset}\, x_{pR}$. Τα predicate domains XPath Set ($\mathbf{X}_{pD}$) περιέχουν XPaths που προτάσσουν τα XPaths που περιέχονται στα predicate ranges XPath Set ($\mathbf{X}_{pR}$).

(γ′) $\exists x_{pR} \in \mathbf{X}_{pR}$ ανδ $\exists x_o \in \mathbf{X}_o : x_{pR} \,\tilde{\sqsubset}\, x_o$. Τα predicate ranges XPath Set ($\mathbf{X}_{pR}$) περιέχουν XPaths που προτάσσουν τα XPaths που περιέχονται στα object XPath Set ($\mathbf{X}_o$).

Έτσι, από τα (α), (β) ανδ (γ), καταλήγουμε στη *Σχέση Υποκείμενο-Κατηγόρημα-Αντικείμενο*, που ορίζεται στην (2):

$$\exists x_s \in \mathbf{X}_s, \exists x_{pD} \in \mathbf{X}_{pD}, \exists x_{pR} \in \mathbf{X}_{pR}, \exists x_o \in \mathbf{X}_o : x_s \,\tilde{\sqsubset}\, x_{pD} \,\tilde{\sqsubset}\, x_{pR} \,\tilde{\sqsubset}\, x_o \tag{2}$$

Η σχέση Υποκείμενο-Κατηγόρημα-Αντικείμενο πρέπει να ισχύει για κάθε triple pattern. Έτσι, ο αλγόριθμος σύνδεσης μεταβλητών χρησιμοποιεί αυτή τη σχέση προκειμένου να καθορίσει τις κατάλληλες συνδέσεις για ολόκληρο το σύνολο των conjunctive triple patterns (δηλαδή, BGP), ξεκινώντας από τις συνδέσεις κάθε επιμέρους τμήματος του triple pattern (δηλαδή, υποκείμενο, κατηγόρημα, αντικείμενο).

#### 5.7.1.2 Σύνοψη Αλγορίθμου

Εδώ περιγράφουμε τον Αλγόριθμο *Σύνδεσης Μεταβλητών* (Αλγόριθμος 1), ο οποίος παίρνει ως είσοδο (α) ένα Basic Graph Pattern ($BGP$), (β) ένα σύνολο από αρχικές συνδέσεις ($\mathbf{X}^{Sch}$), (γ) τους τύπους των μεταβλητών που υπάρχουν στο BGP ($varTypes$), και (δ) οι αντιστοιχίσεις των BGP ontology constructs ($\mathbf{M}$). Οι τύποι των μεταβλητών καθορίζονται στο στάδιο καθορισμού των τύπων μεταβλητών και οι αρχικές συνδέσεις είναι εκείνες που προκύπτουν από το στάδιο της Schema Triple επεξεργασίας.

Ο καθορισμός των συνδέσεων μιας τριπλέτας πραγματοποιείται χρησιμοποιώντας τους κανόνες σύνδεσης (*γραμμές* 13, 16 & 19). Κάθε μέρος της τριπλέτας (υποκείμενο-κατηγόρημα-αντικείμενο) χρησιμοποιεί έναν κανόνα σύνδεσης. Οι κανόνες σύνδεσης παρουσιάζονται αναλυτικά στο [9].

### 5.7.2 Σχέσεις XPath Set για Triple Patterns

Σε αρκετές περιπτώσεις, τα XPath Sets που αντιστοιχούν σε διαφορετικές SPARQL μεταβλητές πρέπει να συσχετιστούν. Για παράδειγμα, έστω triple pattern $?x\ FirstName\_xs\_string\ ?y$, η μεταβλητή $x$ αντιστοιχεί σε Άτομα και Μαθητές και η μεταβλητή $y$ στα ονόματά τους. Το στάδιο σύνδεσης μεταβλητών θα έχει ως αποτέλεσμα δύο XPath Sets: ένα για τα Άτομα και τους μαθητές που αντιστοιχούν στη μεταβλητή $x$ και ένα για όλα τα ονόματα που αντιστοιχούν στη μεταβλητή $y$. Όμως, η συσχέτιση των ατόμων και των ονομάτων τους πρέπει να πραγματοποιηθεί. Εισάγουμε την Σχέση επέκτασης (Extension Relation) η οποία ισχύει για διαφορετικά XPath Sets και μπορεί να χρησιμοποιηθεί για να τα συσχετίσει.

**Ορισμός 2. (Extension Relation)** Ένα XPath Set $\mathbf{D}$ είναι μια επέκταση ενός XPath Set $\mathbf{A}$ εάν όλα τα XPaths στο $\mathbf{D}$ είναι απόγονοι των XPaths του $\mathbf{A}$. Αυτή η σχέση μπορεί να επιτευχθεί αν ο XPath Set Concatenation ($\oplus$) operator [9] εφαρμοστεί στο XPath Set $\mathbf{A}$ έχοντας ως δεξιό όρο ένα XPath Set $\mathbf{C}$, και ως αποτέλεσμα το XPath Set $\mathbf{D}$, το οποίο θα είναι μια *extension* του $\mathbf{A}$ (δηλαδή, $\mathbf{A}\oplus\mathbf{C}=\mathbf{D}$, $\mathbf{D}$ είναι μια *επέκταση* του $\mathbf{A}$).

## 5.8 Μετάφραση των Graph Pattern

Σε αυτή την ενότητα, περιγράφουμε το στάδιο μετάφρασης του graph pattern, κατά το οποίο μεταφράζεται ένα graph pattern σε σημασιολογικά αντίστοιχες XQuery εκφράσεις. Σημειώνουμε ότι σε αυτό το τμήμα υιοθετούμε τη σημασιολογία των SPARQL graph patterns (που ορίζονται στο [272]). Η έννοια του graph pattern ορίζεται αναδρομικά. Το στάδιο μετάφρασης του basic graph pattern (Ενότητα 5.8.1) μεταφράζει τα βασικά στοιχεία ενός GP (δηλαδή, BGPs) σε XQuery εκφράσεις, το οποίο σε αρκετές περιπτώσεις πρέπει να συσχετιστούν στα πλαίσια ενός GP. Δηλαδή, να εφαρμοστούν

**Algorithm 1: Variable Binding Algorithm**

**Input:** Basic Graph Pattern *BGP*, Initial Bindings $\mathbf{X}^{Sch}$,
Variable Types *varTypes*, Mappings **M**

**Output:** Variable Bindings $\mathbf{X}_v$

1. **for each** variable *v* **in** *BGP* *//initialize the bindings*
2. **if** $v \in var(schemaTr(BGP))$ *//if the variable v are included at schema triples*
3. $\mathbf{X}_v^0 = \mathbf{X}_v^{Sch}$
*//initialize the bindings from the bindings determined the from schema triple processing*
4. **else**
5. $\mathbf{X}_v^0 = \{\ominus\}$ *//initialize with the "special" value "⊖"*
6. **end if**
7. **end for**
8. it = 0 *//iteration counter initialization*
9. **repeat**
11. **for each** triple *t* **in** *BGP* *//loop over all the BGP triples*
12. **if** $s \in \mathbf{V}$ *//if the subject is a variable*
13. $\mathbf{X}_s^{i+1} = B_s(\ t, \mathbf{X}_s^i, \mathbf{X}_{pD}^i, \mathbf{X}_o^i, \mathbf{M}\ )$
*//determine the subject bindings of the current iteration (i.e., t+1)*
14. **end if**
15. **if** $p \in \mathbf{V}$ *//if the predicate is a variable*
16. $\mathbf{X}_p^{i+1} = B_p(\ t, \mathbf{X}_s^i, \mathbf{X}_p^i, \mathbf{M}, varTypes\ )$
*//determine the predicate bindings of the current iteration (i.e., i+1)*
17. **end if**
18. **if** $o \in \mathbf{V}$ *//if the object is a variable*
19. $\mathbf{X}_o^{i+1} = B_o(\ t, \mathbf{X}_s^i, \mathbf{X}_p^i, \mathbf{X}_o^i, \mathbf{M}, varTypes\ )$
*//determine the object bindings of the current iteration (i.e., i+1)*
20. **end if**
21. **end for**
22. i = i + 1 *//increase the counter*
23 **until (**$\forall\ v \in var(BGP) \Rightarrow \mathbf{X}_v^i = \mathbf{X}_v^{i-1}$ **)**
*//loop until the bindings of the previous iteration are equal with the bindings of this iteration*
24. **return** $\mathbf{X}_v\ \forall\ v \in var(BGP)$ *//return all the variable bindings for this basic graph pattern*

**Algorithm 2: For or Let XQuery Clause Selection** (*QF*, **RV**, *v*)

**Input:** SPARQL query form *QF*, *Return Variables* **RV**, SPARQL variable *v*
**Output:** XQuery Clause Type

```
1.  if QF ≠ Ask
2.      if (v ∈ RV) or ( ∃K ∈ RV | K is extension of v )
3.          return Create a For XQuery Clause
4.      else
5.          return Create a Let XQuery Clause
6.      end if
7.    else
8.      return Create a Let XQuery Clause
9.  end if
```

οι SPARQL operators (δηλαδή, τα UNION, AND, OPT και FILTER) που μπορεί να προκύψουν έξω από τα BGPs. Ο αλγόριθμος *GP2XQuery* διασχίζει το δέντρο αξιολόγησης SPARQL που προκύπτει από το GP, προκειμένου να χειριστεί τους SPARQL operators.

## 5.8.1 Μετάφραση των Basic Graph Pattern

Αυτή η παράγραφος περιγράφει τη μετάφραση των basic graph pattern σε XQuery εκφράσεις.

### 5.8.1.1 Σύνοψη BGP2XQuery Αλγορίθμου

Εδώ περιγράφουμε τον BGP2XQuery αλγόριθμο, ο οποίος μεταφράζει τα BGPs σε XQuery εκφράσεις. Ο αλγόριθμος δεν εκτελείται τριπλέτα ανά τριπλέτα. Αντίθετα, επεξεργάζεται τα υποκείμενα, τα κατηγορήματα και τα αντικείμενα όλων των τριπλετών ξεχωριστά. Για κάθε μεταβλητή SPARQL που περιλαμβάνεται στο BGP, ο αλγόριθμος δημιουργεί τους XQuery όρους For ή Let, χρησιμοποιώντας τις συνδέσεις μεταβλητών (variable binding), τις αντιστοιχίσεις (input mappings), και την σχέση επέκτασης (extension relation) (Ορισμός 2). Η μετάφραση των BGPs περιγράφεται λεπτομερώς στις ακόλουθες παραγράφους.

### 5.8.1.2 For or Let Clause?

Ένα ζήτημα κομβικής σημασίας στη δημιουργία των XQuery εκφράσεων είναι η παραγωγή της κατάλληλης ακολουθίας λύσης (solution sequence) βασισμένοι στη SPARQL σημασιολογία. Για να το επιτύχουμε αυτό, για μια SPARQL μεταβλητή $v$, δημιουργούμε έναν όρο For ή έναν όρο Let σύμφωνα με τον αλγόριθμο που παρουσιάζεται παρακάτω (Αλγόριθμος 2). Διαισθητικά, ο αλγόριθμος διαλέγει μεταξύ της δημιουργίας των όρων For και Let προκειμένου να παράξει την επιθυμητή ακολουθία λύσης.

### 5.8.1.3 Μετάφραση Υποκειμένου

Ο αλγόριθμος *μετάφρασης υποκειμένου* (Subject Translation algorithm) (Αλγόριθμος 3) μεταφράζει το υποκείμενο όλων των triple patterns ενός BGP σε XQuery εκφράσεις. Πρέπει να σημειώσουμε ότι, για το υπόλοιπο κεφάλαιο, το σύμβολο $N_X$ συμβολίζει το όνομα της SPARQL μεταβλητής $X$ και τα triple patterns συμβολίζονται με $s\ p\ o$, όπου $s$ είναι το υποκείμενο, $p$ το κατηγόρημα και $o$ το αντικείμενο του triple pattern.

**Algorithm 3: Subject Translation (** *BGP*, *QF*, **R V** , *bindings* **)**

**Input:** Basic Graph Pattern *BGP*, SPARQL query form *QF*,
SPARQL Return Variables **R V** , Variable Bindings *bindings*
**Output:** For or Let XQuery Clause *xC*

1. **for each** *triple* **in** *BGP*
2. **if** $s \in \mathbf{V}$ *// If subject is a variable*
3. $xC.type \leftarrow$ **For or Let XQuery Clause Selection** ( *QF*, **R V** , *s* )
*//Create a For or Let XQuery Clause*
4. $xC.var \leftarrow N_s$ *// Define an XQuery Variable with the name of SPARQL Variable s*
5. $xC.expr \leftarrow \$doc/x_1$ union $\$doc/x_2$ union … union $\$doc/x_n$ , $\forall x_i \in \mathbf{X}_s$
*// Set expr equal to the XPath Set of the Subject prefixed with the $doc variable*
*//* $\mathbf{X}_s$ *is the binding XPath Set for the variable s*
6. **end if**
7. **end for**
8. **return** *xC*

**Algorithm 4: Predicate Translation (** *BGP*, *QF*, **R V** , *bindings* **)**

**Input:** Basic Graph Pattern *BGP*, SPARQL query form *QF*,
SPARQL Return Variables **R V** , Variable Bindings *bindings*
**Output:** For or Let XQuery Clause *xC*

1. **for each** *triple* **in** *BGP*
2. **if** $p \in \mathbf{V}$ *// If predicate is a variable*
3. $xC.type \leftarrow$ **For or Let XQuery Clause Selection** ( *QF*, *RV*, *p* )
*//Create a For or Let XQuery Clause*
4. $xC.var \leftarrow N_p$ *// Define an XQuery Variable with the same name with the SPARQL Variable p*
5. $xC.expr \leftarrow \$ N_s /x_1$ union $\$ N_s /x_2$ union … union $\$ N_s /x_n$ , $\forall x_i \in \mathbf{X}_s \gg \mathbf{X}_p$
*// Set expr equal to the variable corresponding to the triple subject variable suffixed with XPaths that have resulted from the* $\mathbf{X}_s \gg \mathbf{X}_p$ *operation. The XPath Set.* $\mathbf{X}_p$ *is the binding XPath Set for the variable p and* $\mathbf{X}_s$ *is the binding XPath Set for the subject s*
6. **end if**
7. **end for**
8. **return** *xC*

#### 5.8.1.4 Μετάφραση Κατηγορήματος

Ο αλγόριθμος *μετάφρασης κατηγορήματος* (Predicate Translation algorithm) (Αλγόριθμος 4) μεταφράζει το κατηγόρημα όλων των triple patterns ενός BGP σε XQuery εκφράσεις.

#### 5.8.1.5 Μετάφραση Αντικειμένου

Ο αλγόριθμος *μετάφρασης αντικειμένου* (Object Translation algorithm) (Αλγόριθμος 5) μεταφράζει το αντικείμενο όλων των triple patterns ενός BGP σε XQuery εκφράσεις.

#### 5.8.1.6 Κατασκευή του Return όρου

Ο αλγόριθμος *Construct Return Clause* (Αλγόριθμος 7) δημιουργεί τον XQuery όρο επιστροφής (Return Clause).

## 5.9 Solution Sequence Modifiers & Query Forms

### 5.9.1 Μεταφράζοντας τους Solution Sequence Modifiers

Αυτή η ενότητα περιγράφει το στάδιο μετάφρασης των *Solution Sequence Modifier*, το οποίο μεταφράζει τους SPARQL Solution Sequence Modifiers (SSMs) σε XQuery

**Algorithm 5: Object Translation (** *BGP*, *QF*, **R V** , *bindings*, *mappings* **)**

**Input:** Basic Graph Pattern *BGP*, SPARQL query form *QF*, SPARQL Return Variables **R V** , Variable Bindings *bindings*, mappings between the ontology and the XML schema *mappings*

**Output:** For or Let XQuery Clause *xC*

1. **for each** *triple* **in** *BGP*
2. **if** $o \in \mathbf{I}$ *// If the object is a literal*
3. **if** $p \in \mathbf{V}$ *// If the predicate is a variable*
4. Create *XPredicate* over the *xC.expr* where *xC* is the For/Let clause created for the predicate *p*
5. *XPredicate* ← [.= "*o*"]
6. **if** Let XQuery Clause created for *p*
7. Create *"Bindings Assurance Condition"* for *p* *//see "Biding Assurance Condition" Section*
8. **end if**
9. **else** *// The predicate is not a variable – it is an IRI*
10. Create *XPredicate* $\forall x_i \in \mathbf{X}_s$ in *xC.expr*, where *xC* is the For/Let clause created for the subject *s*
11. *XPredicate* ← $[./y_1 = \text{"o"}$ or $./y_2 = \text{"o"}$ or … or $./y_n = \text{"o"}] \forall y_i \in \{x_i\} \gg \mu_p$
*// $\mathbf{X}_s$ is the bindings XPath Set for the subject S and $\mu_P$ is the mappings XPath Set for the property p*
12. **end if**
13. **else if** $o \in \mathbf{V}$ *// If the object is a variable*
14. **if** $p \in \mathbf{V}$ *// If the predicate is a variable*
15. *xC.type* ← Create a Let XQuery Clause
16. *xC.var* ← $N_o$ *// Define an XQuery Variable with the name of the SPARQL Variable o*
17. *xC.expr* ← $\$ N_p$ *// Set expr equal to the predicate Variable*
18. **if** Let XQuery Clause created for *p*
19. Create *"Bindings Assurance Condition"* for *o* *//see "Biding Assurance Condition" Section*
20. **end if**
21. **else** *// The predicate is not a variable – it is an IRI*
22. *xC.type* ← **For or Let XQuery Clause Selection (** *QF*, **R V** , *o* **)** *//Create a For or Let XQuery Clause*
23. *xC.var* ← $N_o$ *// Define an XQuery Variable with the name of the SPARQL Variable o*
24. *xC.expr* ← $\$ N_s / x_1$ union $\$ N_s / x_2$ union … union $\$ N_s / x_n \forall x_i \in \mathbf{X}_s \gg \mu_p$
*// Set expr equal to the variable corresponding to the triple subject suffixed with some of the XPath of the Predicate XPath Set*
*// $\mathbf{X}_s$ is the bindings XPath Set for the subject s and $\mu_P$ is the mappings XPath Set for the property p.*
25. **if** Let XQuery Clause created for *o*
26. Create *"Bindings Assurance Condition"* for *o* *//see "Biding Assurance Condition" Section*
27. **end if**
28. **end if**
29. **end if**
30. **end for**
31. **return** *xC*

εκφράσεις. Οι SSMs που μπορεί να περιέχονται στις SPARQL ερωτήσεις μεταφράζονται χρησιμοποιώντας XQuery όρους και ενσωματωμένες συναρτήσεις. Οι SSMs που υποστηρίζονται από το τωρινό SPARQL πρότυπο είναι οι Distinct, Reduced, Order By, Limit, και Offset.

Ο Πίνακας 5.8 συνοψίζει τις XQuery εκφράσεις και τις ενσωματωμένες XQuery συναρτήσεις που χρησιμοποιούνται για τη μετάφραση των solution sequence modifiers.

**Πίνακας 5.8:** Μετάφρασης των Solution Sequence Modifier σε XQuery εκφράσεις

| Solution Sequence Modifier | XQuery Expressions |
|---|---|
| **LIMIT** $n$ | **return**( $Results[**position**( )<= $n$ ] ) |
| **OFFSET** $n$ | **return**( $Results[**position**( )> $n$ ] ) |
| **LIMIT** $n$ && **OFFSET** $m$ | **return**( $Results[ **position**( )> $m$ **and position**( )<= $n + m$ ] ) |
| **ORDER BY DESC**(?x) **ASC**(?y) | **for** $res **in** $Results<br>**order by** $res/x **descending empty least**, $res/y **empty least**<br>**return** $res |

## 5.9.2 Μεταφράζοντας τους Τύπους των Ερωτήσεων

Η *μετάφραση των Query Form* είναι το τελικό στάδιο της μετάφρασης από SPARQL σε XQuery. Το τωρινό πρότυπο της γλώσσας ερωτήσεων SPARQL υποστηρίζει τέσ-

**Algorithm 6: Filter Translation (** *BGP* **)**

**Input:** Basic Graph Pattern *BGP*

**Output:** Where XQuery Clause *xC* or Create *XPredicates* over XQuery clauses

1. **for each** *Filter* **in** *BGP*
2. Translate the SPARQL Operators of the *Filter* expression
3. **if** (*Filter* **is** *safe* )
4. Create *XPredicates* for the Filter expressions
5. **else**
6. *xC* ← Create an XQuery Where Clause Condition
7. **end if**
8. **end for**
9. **return** *xC*

**Algorithm 7: Construct Return Clause (** *BGP, QF,* **R V** , *varTypes* **)**

**Input:** Basic Graph Pattern *BGP*, SPARQL query form *QF*, SPARQL Return Variables **R V** , Variable Types *varTypes*

**Output:** Return XQuery Clause *xC*

1. **if** *QF* = Ask
2. *xC* ← return("yes") *//Create an XQuery Return clause*
3. **else** *//The query form is not Ask*
   *//Create an XQuery Return clause*
4. *xC* ← return(<Result>
   <$var_1$>...</$var_1$> , <$var_2$>...<$var_2$>,…,<$var_r$>...</$var_r$></Result>)
   $\forall\ var_i \in \mathbf{R\,V} \cap var(BGP)$
   *// Each Return Variable included in the given BGP is inserted in the XQuery return clause*
5. $\forall\ var_i \in \mathbf{R\,V} \cap var(BGP)$ use the *varTypes* to determine the result form of $var_i$
6. **end if**
7. **return** *xC*

$$
Q_X = \begin{cases}
\textbf{let } \$\text{Results} := (xE_Q) & \textbf{if } QF = \textit{Select} \\
\textbf{return } (\ \langle\text{Results}\rangle\ \$\text{Results}\ \langle/\text{Results}\rangle\ ) & \\
\\
\textbf{let } \$\text{Results} := (xE_Q) & \textbf{if } QF = \textit{Ask} \\
\textbf{return } (\textbf{ if } (\textbf{ empty } (\$\text{Results})\ )\ \textbf{then } \text{“no”}\ \textbf{else } \text{“yes”}\ ) & \\
\\
\textbf{let } \$\text{Results} := (xE_Q) & \textbf{if } QF = \textit{Construct} \\
\textbf{for } \$\text{res}\ \textbf{at } \$\text{iter}\ \textbf{in } \$\text{Results} & \\
\textbf{return } (\textbf{ if } (\textbf{ exists}(\ \$\text{res/x}\ )\ )\ \textbf{then} & \\
\quad \textbf{concat } (\textbf{ concat } (\ \text{“\_:a”}\ ,\ \$\text{iter}\ ),\ \text{“ iri:property ”}\ ,\ \textbf{string}(\ \$\text{res/x}\ )\ ,\ \text{“.”}\ ) & \\
\quad \textbf{else } (\ ) & \\
\quad \textbf{if } (\textbf{ exists}(\ \$\text{res/p}\ )\ \textbf{and exists}(\ \$\text{res/y}\ )\ )\ \textbf{then} & \\
\quad \textbf{concat } (\textbf{ concat}(\ \text{“\_:a”}\ ,\ \$\text{iter}\ ),\ \textbf{string}(\ \$\text{res/p}\ )\ ,\ \textbf{string}(\ \$\text{res/y}\ )\ ,\ \text{“.”}\ ) & \\
\quad \textbf{else } (\ )\ ) &
\end{cases} \tag{5}
$$

σερις μορφές ερωτήσεων: Select, Ask, Construct και Describe. Σύμφωνα με τη μορφή ερωτήσεων, ο τύπος των αποτελεσμάτων που επιστρέφονται είναι διαφορετικός. Συγκεκριμένα, μετά τη μετάφραση οποιουδήποτε solution sequence modifier, το XQuery που προκύπτει εμπλουτίζεται από τις κατάλληλες, για αυτή τη μορφή ερωτήσεων, XQuery εκφράσεις προκειμένου να διαμορφωθούν τον κατάλληλο τύπο αποτελεσμάτων (δηλαδή, ένα RDF γράφο, μια ακολουθία αποτελέσματος (result sequence), ή μια Boolean τιμή).

Η μετάφραση των query forms σκιαγραφείται στο (5), όπου $Q_X$ είναι ένα σύνολο XQuery εκφράσεων που προκύπτουν μετά από τη μετάφραση της SPARQL query form. Έστω η SPARQL query $Q_S = \langle QF, GP, SSM \rangle$, όπου $QF$ είναι η query form, $GP$ είναι το query graph pattern και $SSM$ οι solution sequence modifiers. Έστω $xE_Q$ οι XQuery εκφράσεις που παράγονται από τη μετάφραση από των graph pattern ($GP$) και των solution sequence modifier ($SSM$).

## 5.10 XQuery Βελτιστοποίηση - Αναδιατύπωση

Σε αυτή την ενότητα, παρουσιάζουμε ένα μικρό αριθμό απλών κανόνων αναδιατύπωσης που στόχο έχουν την παροχή πιο αποδοτικών XQuery εκφράσεων. Αυτοί οι κανόνες εφαρμόζονται στις XQueries που δημιουργούνται από τη μετάφραση της SPARQL σε XQuery.

### 5.10.1 Κανόνες Αναδιατύπωσης

Οι προτεινόμενοι κανόνες αναδιατύπωσης στοχεύουν στη παροχή πιο αποδοτικών XQuery εκφράσεων που επωφελούνται από τη γνώση του τρόπου δημιουργίας των XQuery εκφράσεων κατά τη διάρκεια της μετάφρασης από SPARQL σε XQuery, καθώς και από τη σημασιολογία του XML Schema. Οι κανόνες χρησιμοποιούν τα προηγούμενα, προκειμένου να αφαιρέσουν τους πλεονάζοντες XQuery όρους και μεταβλητές, να απελευθερώσουν εμφωλευμένους For XQuery όρους και τέλος να ελαχιστοποιήσουν τα loops που εκτελούνται από τους For XQuery όρους.

Οι κανόνες αναδιατύπωσης εφαρμόζονται διαδοχικά στις δημιουργούμενες XQuery ερωτήσεις. Πρώτα, εφαρμόζεται ο κανόνας *Rule* 1, μετά ο κανόνας *Rule* 2 εφαρμόζεται στην ερώτηση XQuery που προκύπτει, και τελικά εφαρμόζεται ο κανονας *Rule* 3.

**Κανόνας Αναδιατύπωσης 1 [Αλλάζοντας τους For Όρους σε Let Όρους]:** Έστω $xC$ ένας For XQuery όρος, $A$ το σύνολο των XML στοιχείων και/ή γνωρισμάτων που αντιστοιχούν στις XPath εκφράσεις που περιέχονται στο $xC.expr$. Αν κάθε από τα XML στοιχεία/γνωρίσματα που περιέχονται στο $A$ μπορούν να εμφανιστούν το πολύ μία φορά, τότε το $xC$ αλλάζει από For XQuery όρο σε Let XQuery όρο. Τυπικά, η αλλαγή των όρων For σε όρους Let εκφράζεται ως εξής:

$$\forall a \in A : a.cardinality \in [0,1] \Rightarrow xC.type \leftarrow Let$$

Ο κανόνας *αλλαγής των For όρων σϵ Let όρους* εφαρμόζεται στους For XQuery όρους, από τα πάνω προς τα κάτω (ή και το αντίστροφο). Διαισθητικά, αυτός ο κανόνας χρησιμοποιεί τις πληροφορίες σχήματος προκειμένου να μετατρέψει τους For όρους σε Let όρους σε περιπτώσεις όπου δεν μπορούν να υπάρχουν πολλαπλές τιμές. Ο στόχος αυτού του κανόνα είναι να αποφύγει τους μη απαραίτητους ελέγχους για πιθανές πολλαπλές τιμές που εκτελούνται από τους For όρους, σε περιπτώσεις όπου Let όροι μπορούν επίσης να χρησιμοποιηθούν. Η χρήση αυτού του κανόνα έχει ως αποτέλεσμα περισσότερους Let όρους που μπορούν να αφαιρεθούν αργότερα, όταν εφαρμοστεί ο Κανόνας 2.

**Κανόνας Αναδιατύπωσης 2 [Μειώνοντας τους Let Όρους]:** Έστω $xCl_1$ ένας Let XQuery όρος. Αν η μεταβλητή $xCl_1.var$ του Let όρου, είναι *επέκταση* (Ορισμός 2) μιας XQuery μεταβλητής $xCl_2.var$, όπου $xCl_2$ είναι ένας For ή Let XQuery όρο, τότε ο όρος Let αφαιρείται και ο $xCl_1.var$ αντικαθίστανται παντού με το $xCl_1.epr$. Επιπλέον, αν έχει οριστεί μια Biding Assurance συνθήκη για το $xCl_1.var$ δηλαδή, αν υπάρχει ένα ($xCl_1.var$) statement στον Where XQuery όρο (Ενότητα 5.8.1.5). Τότε, η exists function αφαιρείται και αντικαθίσταται από μια συνθήκη στο $xCl_2.expr$. Η συνθήκη ορίζεται χρησιμοποιώντας τα XPredicates και τα XPaths του $xCl_1.expr$.

Ο κανόνας *μείωσης των Let όρων* εφαρμόζεται επαναληπτικά στους Let XQuery όρους, από κάτω προς τα πάνω. Διαισθητικά, αυτός ο κανόνας αφαιρεί τους μη απαραίτητους Let clauses που έχουν παραχθεί από τη triple pattern μετάφραση και μπορούν να περικοπούν. Ο στόχος αυτού του κανόνα είναι να εξαλειφθούν οι μη απαραίτητοι XQuery όροι και μεταβλητές. Επίσης, σε περίπτωση ύπαρξης της Biding Assurance συνθήκης, πραγματοποιείται ένα *predicate pushdown*. Συγκεκριμένα, η exists condition που υπάρχει στον Where XQuery όρο αποτιμάται σε ένα νωρίτερο στάδιο επεξεργασίας των ερωτήσεων αφού εφαρμόζεται στα XPaths χρησιμοποιώντας τα XPath κατηγορούμενα.

**Κανόνας Αναδιατύπωσης 3 ["Ελευθερώνοντας" Εμφωλευμένους For όρους]:** Έστω $xCl_1$ ένας For XQuery όρος. Αν η μεταβλητή του For όρου $xCl_1.var$ δεν είναι Return μεταβλητή ($xCl_1.var \notin \mathbf{RV}$) και μόνο μία XQuery μεταβλητή $xCl_2.var$ είναι *επέκταση* (Ορισμός 2) του $xCl_1.var$. Τότε, ο For όρος αφαιρείται και $xCl_1.var$ αντικαθίστανται από το $xCl_1.expr$.

Ο κανόνας *απελευθέρωσης των εμφωλευμένων For όρων* εφαρμόζεται επαναληπτικά για τους For XQuery όρους, από πάνω προς τα κάτω. Διαισθητικά, αυτός ο κανόνας απελευθερώνει εμφωλευμένους For όρους που μπορούν να εκφραστούν ως ένας For

όρος. Ο στόχος αυτού του κανόνα είναι να μειώσει τους εμφωλευμένους For όρους, και με αυτό το τρόπο, αφαιρούνται επίσης μερικοί XQuery όροι και μεταβλητές.

Επισήμως, ο κανόνας απελευθέρωσης των εμφωλευμένων For όρων περιγράφεται ως:

*Initial XQuery Expressions*

**for** $v$₁ **in** expr$_1$
…
**for** $v$₂ **in** $v$₁/xp$_1$
…
**where** ( … funcX($v$₁) … )
**return** ( … )

⇒

*Rewritten XQuery Expressions*

**for** $v$₂ **in** expr$_1$/xp$_1$
…
**where** ( … funcX(expr$_1$) … )
**return** ( … )

## 5.11 Υποστήριξη SPARQL Ερωτήσεων Ενημέρωσης

Σε αυτή την ενότητα, περιγράφουμε την επέκταση του πλαισίου SPARQL2XQuery για την υποστήριξη SPARQL ερωτημάτων ενημέρωσης [297]. Η υποστήριξη ενημερώσεων τόσο στην γλώσσα SPARQL όσο και στην γλώσσα XQuery έχουν πρόσφατα προτυποποιηθεί στις εκδόσεις SPARQL 1.1 και XQuery Update Facility [287], αντίστοιχα. Σε αυτή την ενότητα έχουμε μελετήσει τις αντιστοιχίες μεταξύ των δυο αυτών εκδόσεων, και περιγράφουμε την επέκταση του μοντέλου αντιστοιχίσεων (mapping mode) καθώς και των αλγορίθμων μετάφρασης από SPARQL σε XQuery, για την υποστήριξη των τελεστών ενημέρωσης (update operations).

Σχετικές εργασίες έχουν πρόσφατα προταθεί. Στο σενάριο διαλειτουργικότητας μεταξύ RDB και RDF, το D2R/Update [140] (μια επέκταση D2R) και το OntoAccess [183] υποστηρίζουν SPARQL ερωτήματα ενημέρωσης πάνω από σχεσιακές βάσεις δεδομένων. Όσον αφορά το σενάριο διαλειτουργικότητας μεταξύ XML, RDB και RDF, η εργασία που παρουσιάζεται στο [28] επεκτείνει τη γλώσσα XSPARQL [75] , προκειμένου να υποστηρίξει τα ερωτήματα ενημέρωσης.

### 5.11.1 Μετάφραση SPARQL Ερωτήσεων Ενημέρωσης σε XQuery

Ο Πίνακας 5.9 παρουσιάζει του τελεστές ενημέρωσης της SPARQL και συνοψίζει τη μετάφρασή τους στην XQuery. Συγκεκριμένα, υπάρχουν τρεις βασικές κατηγορίες τελεστών SPARQL ενημερώσεων: a) Delete Data; b) Insert Data; and c) Delete/Insert. Για κάθε τελεστή ενημέρωσης, μια απλοποιημένη σύνταξη SPARQL παρουσιάζεται, καθώς και οι αντίστοιχες εκφράσεις σε XQuery. Για την SPARQL υποθέτουμε τα ακόλουθα σύνολα, ως $tr$ θεωρούμε μια τριπλέτα RDF, $tp$ είναι μια τριπλέτα σχηματομορφής (triple pattern), $trp$ είναι είτε τριπλέτα ή τριπλέτα σχηματομορφής, και gp είναι μια σχηματομορφή γράφου. Επιπλέον, στην XQuery, ορίζουμε ως $xE_W$, $xE_I$ και $xE_D$ τις εκφράσεις XQuery (δηλαδή, εκφράσεις FLOWR) που προκύπτουν από τη μετάφραση των SPARQL Where, Insert and Delete, αντίστοιχα. Το $xE$ είναι ένα σύνολο XQuery εκφράσεων, το $xE(\$v_1, \$v_2, \ldots \$v_n)$ δηλώνει ότι το $xE$ χρησιμοποιεί τις τιμές των XQuery μεταβλητών $\$v_1, \$_v2, \ldots \$v_n$. Τέλος, το $xn$ υποδηλώνει ένα τμήμα (fragment) της XML, δηλαδή, ένα σύνολο από XML κόμβους, και $xp$ υποδηλώνει μια XPath έκφραση.

**Πίνακας 5.9:** Μετάφραση SPARQL Ερωτήσεων Ενημέρωσης σε XQuery

| **SPARQL** | | **Translated XQuery Expressions** |
|---|---|---|
| **SPARQL Update Operation** | **Syntax Template** [1] | |
| DELETE DATA | **Delete data**{<br>*tr*<br>} | **delete nodes collection**("http://dataset...")/$xp_1$<br>**...**<br>**delete nodes collection**("http://dataset...")/$xp_n$ |
| INSERT DATA | **Insert data**{<br>*tr*<br>} | **let** $n$1 := $xn_1$<br>**…**<br>**let** $n$n := $xn_n$<br>**let** $data1 := ($nk, $nm,…) // k, m,… ∈[1,n]<br>**…**<br>**let** $datap := ($nj, $nv,…) // j, y,… ∈[1,n]<br>**let** $insert_location1 **:= collection**("http://xmldataset...")/$xp_1$<br>**…**<br>**let** $insert_locationp **:= collection**("http://xmldataset...")/$xp_p$<br>**return(**<br>**insert nodes** $data1 **into** $insert_location1 ,<br>**…**<br>**insert nodes** $datap **into** $insert_locationp<br>) |
| DELETE / INSERT | (a) **Delete**{<br>*trp*<br>}**Where**{<br>*gp*<br>}<br><br>(b) **Insert**{<br>*trp*<br>}**Where**{<br>*gp*<br>}<br><br>(c) **Delete**{<br>*trp*<br>}**Insert**{<br>*trp*<br>}**Where**{<br>*gp*<br>} | (a)<br>**let** $where_gp := $xE_W$<br>**let** $delete_gp:= $xE_D$ ($where_gp)<br>**return delete nodes** $delete_gp<br><br>(c)<br>Translate Delete Where same as (a),<br>then translate Insert Where same as (b)<br><br>(b) **let** $where_gp := $xE_W$<br>**let** $insert_location1 := $xp_1$<br>**for** $it1 **in** $insert_location1<br>$xE_I$ ($where_gp, $it1)<br>**return insert nodes into** $it1<br>**…**<br>**let** $where_gp := $xE_W$<br>**let** $insert_ location n := $xp_n$<br>**for** $itn **in** $insert_locationn<br>$xE_I$ ($where_gp, $itn)<br>**return insert nodes into** $itn |

[1] For simplicity, the WITH, GRAPH and USING clauses are omitted.

**Delete Data.** Ο SPARQL τελεστής Delete Data αφαιρεί ένα σύνολο από RDF τριπλέτες. Αυτός ο τελεστής μπορεί να μεταφραστεί σε XQuery χρησιμοποιώντας τον XQuery τελεστή Delete Nodes. Συγκεκριμένα, χρησιμοποιώντας τις αντιστοιχίσεις, το σύνολο των τριπλετών tp που ορίζονται στην SPARQL μετατρέπονται (χρησιμοποιώντας μια παρόμοια προσέγγιση όπως είναι ο αλγόριθμος BGP2XQuery) σε μια σειρά από εκφράσεις XPath **XP**. Για κάθε $xp_i \in$ **XP** ένα XQuery Delete Nodes ορίζεται.

**Insert Data.** Ο SPARQL τελεστής Insert Data, προσθέτει μια σειρά από νέες τριπλέτες σε RDF γράφους. Αυτός ο SPARQL τελεστής μπορεί να μεταφραστεί σε XQuery χρησιμοποιώντας τον XQuery τελεστή Insert Nodes. Στη μετάφραση του Insert Data, το σύνολο των τριπλετών $tr$ που ορίζονται στην SPARQL μετατρέπονται σε ένα σύνολο XML κόμβων $xn_i$, χρησιμοποιώντας τις προκαθορισμένες αντιστοιχίσεις. Ειδικότερα, αυτό πραγματοποιείται με μια σειρά από Let XQuery ορισμούς. Στη συνέχεια, η θέση εισαγωγής των XML κόμβων μπορεί εύκολα να προσδιοριστεί λαμβάνοντας υπόψη τις τριπλέτες και τις αντιστοιχίσεις.

**Insert / Delete.** Οι SPARQL τελεστές Delete/Insert χρησιμοποιούνται για την αφαίρεση ή/και την προσθήκη ενός συνόλου τριπλετών από/σε RDF γράφους. Σύμφωνα με την σημασιολογία της SPARQL 1.1, ο Where ορισμός που εμφανίζετε πρώτος είναι αυτός που αρχικά αποτιμάται. Στη συνέχεια, οι τελεστές Delete/Insert εφαρμόζονται πάνω στα παραγόμενα αποτελέσματα. Οι SPARQL τελεστές Delete/Insert μπορούν να μεταφραστούν σε XQuery χρησιμοποιώντας τους τελεστές Delete Nodes και Insert Nodes, αντίστοιχα. Εν συντομία, αρχικά ο γράφος σχηματομορφής που χρησιμοποιείται στην Where δήλωση του SPARQL μεταφράζεται σε XQuery εκφράσεις $xE_W$ (ομοίως όπως και στον αλγόριθμο GP2XQuery). Στη συνέχεια, ο γράφος σχηματομορφής

που χρησιμοποιείται στην Delete/Insert δήλωση, μεταφράζεται σε XQuery εκφράσεις $xE_D/xE_I$ (όπως στον αλγόριθμο BGP2XQuery).

## 5.12 Πειραματική Ανάλυση

Σε αυτή τη παράγραφο παρουσιάζουμε τα αποτελέσματα της πειραματικής αξιολόγησης που έχουμε πραγματοποιήσει για το SPARQL2XQuery Framework χρησιμοποιώντας τόσο συνθετικά όσο και πραγματικά σύνολα δεδομένων. Ο στόχος ήταν να αξιολογήσουμε την αποδοτικότητα των: (α) μετασχηματισμού σχήματος, (β) δημιουργίας αντιστοιχίσεων, (γ) μετάφρασης ερωτήσεων, και (δ) αποτίμηση ερωτήσεων. Έχουμε χρησιμοποιήσει αρκετά σύνολα ερωτήσεων προσπαθώντας να καλύψουμε σχεδόν όλες τις παραλλαγές της SPARQL σύνταξης, τις ιδιότητες και τις ειδικές περιπτώσεις.

Το SPARQL2XQuery Framework έχει αναπτυχθεί χρησιμοποιώντας τεχνολογίες σχετικές με την Java (Java 2SE και Jena) πάνω στην open source, native XML βάση δεδομένων. Έχουμε χρησιμοποιήσει δύο native XML βάσεις δεδομένων (και τις δικές τους XQuery engines) που αναφέρονται ως "XML Store Y" και "XML Store Z". Επιπλέον, έχουμε χρησιμοποιήσει μια memory-based XQuery engine που αναφέρονται ως "Memory- based XQuery Engine". Για RDF αποθήκευση, έχουμε χρησιμοποιήσει το Jena TDB 0.10.1 και την Jena ARQ 2.10.1 SPARQL engine. Τέλος, για την αξιολόγηση του XS2OWL χρησιμοποιήσαμε δύο επεξεργαστές XSLT, έναν freeware XSLT επεξεργαστή που αναφέρεται ως "Freeware XSLT Processor", και έναν XSLT επεξεργαστή που είναι ενσωματωμένος σε ένα εμπορικό εργαλείο, που χαρακτηρίζεται ως "Commercial XSLT Tool".

### 5.12.1 Μετασχηματισμός Σχήματος & Απόδοση Δημιουργίας Αντιστοιχίσεων

Προκειμένου να αξιολογήσουμε το SPARQL2XQuery Framework, έχουμε χρησιμοποιήσει αρκετά διεθνή πρότυπα από διαφορετικούς τομείς (δηλαδή, ψηφιακές βιβλιοθήκες, πολιτιστική κληρονομιά, πολυμέσα) που έχουν εκφραστεί σε XML Schema. Το Persons XML Schema που ορίστηκε στην Ενότητα 5.3.2 έχει επίσης χρησιμοποιηθεί. Αυτά τα XML Schemas έχουν χρησιμοποιηθεί για να αξιολογηθεί ο μετασχηματισμός σχήματος και οι διαδικασίες δημιουργίας αντιστοιχίσεων.

Ακολούθως, παρουσιάζουμε τα αποτελέσματα του πειράματος που προκειμένου να μελετήσουμε την απόδοση του μετδιεξήγαμεασχηματισμού σχήματος και της δημιουργίας αντιστοιχίσεων.

Για κάθε ένα από τα XML Schemas, έχουμε χρησιμοποιήσει το στοιχείο XS2OWL, προκειμένου αυτόματα να μετασχηματιστεί ένα XML Schema σε OWL οντολογία, μετρώντας παράλληλα τον χρόνο που απαιτείται για αυτόν τον μετασχηματισμό (*Schema Transformation Time*). Τότε, χρησιμοποιώντας την οντολογία σχήματος που δημιουργήθηκε και το XML Schema, και μετράμε τον χρόνο που απαιτείται από το στοιχείο παραγωγής αντιστοιχίσεων (Mapping Generator component) του SPARQL2XQuery Framework προκειμένου να ανακαλύψει αυτόματα και να παράγει τις αντιστοιχίσεις (*Mapping Generation Time*).

Ο Πίνακας 5.10 παρουσιάζει τον χρόνο μετασχηματισμού σχήματος και τον χρόνο παραγωγής αντιστοιχίσεων για κάθε XML Schema. Σημειώνουμε ότι ο χρόνος μετασχηματισμού σχήματος παρουσιάζεται τόσο για τον επεξεργαστή *Freeware XSLT* και

**Πίνακας 5.10:** Χρόνος Μετασχηματισμού Σχήματος και Χρόνος Παραγωγής Αντιστοιχίσεων (msec)

| XML Schema | Schema Transformation Time | | Mapping Generation Time |
|---|---|---|---|
| | Freeware XSLT Processor | Commercial XSLT Tool | |
| Persons (Section 5.3.2) | 2.4 | 17.9 | 8.7 |
| DBLP ⋆ | 62.5 | 22.5 | 360.9 |
| METS [6] | 58.2 | 270.5 | 388.9 |
| Text MD [13] | 7.7 | 45.1 | 14.5 |
| MPEG-7 [9] | 730.7 | 3500.6 | 1954.2 |
| SCORM 12 [11] | 132.7 | 415.2 | 421.1 |
| MARC 21 [4] | 6.3 | 51.4 | 12.5 |
| MODS [7] | 191.3 | 594.8 | 482.3 |
| TEI [14] | 840 | 980.1 | 2208.4 |
| TEI Lite [14] | 418 | 932.6 | 1288.3 |
| EAD [2] | 402.7 | 3305.7 | 1052 |
| VRA Core 4 [15] | 47.3 | 290 | 304.3 |
| VRA Core 4 Strict [15] | 3.3 | 122.1 | 10 |
| MIX [10] | 200 | 601.3 | 495.5 |
| MADS [5] | 50.1 | 393.4 | 345.6 |

⋆Note that in our experiments, the DTD that originally describes the DBLP dataset has been expressed in XML Schema syntax.

το *Commercial XSLT Tool*.

## 5.12.2 Αποδοτικότητα Μετάφρασης

Σε αυτή την ενότητα παρουσιάζουμε τα πειραματικά αποτελέσματα που σχετίζονται με την αποδοτικότητα της διαδικασίας μετάφρασης από SPARQL σε XQuery. Για να αξιολογήσουμε την αποδοτικότητα της διαδικασίας της μετάφρασης, μετράμε τον χρόνο μετάφρασης που απαιτείται από το SPARQL2XQuery Framework προκειμένου να μεταφράσει μια SPARQL ερώτηση σε XQuery ερώτηση. Παρακάτω, παρουσιάζουμε τρία πειράματα. Στο πρώτο πείραμα (Ενότητα 5.12.2.1), έχουμε δημιουργήσει αρκετές SPARQL ερωτήσεις τροποποιώντας το μέγεθος και τον τύπου του graph pattern. Στο δεύτερο πείραμα (Ενότητα 5.12.2.1), έχουμε τροποποιήσει στις προηγούμενες ερωτήσεις τον αριθμό των αντιστοιχίσεων μεταξύ της οντολογίας και του XML Schema. Τέλος, στο τρίτο πείραμα, έχουμε χρησιμοποιήσει τρια σύνολα SPARQL ερωτήσεων προσπαθώντας να καλύψουμε όλες τις παραλλαγές της SPARQL γραμματικής (Ενότητα 5.12.2.2).

### 5.12.2.1 Χρόνος Μετάφρασης για διαφορετικά Graph Patterns & Αντιστοιχίσεις

Εδώ, εξετάζουμε την αποδοτικότητα της διαδικασίας μετάφρασης ερωτήσεων.

#### 5.12.2.1.1 Μεταβάλλοντας τους Τύπους και τα Μεγέθη του Graph Pattern

Σε αυτό το πείραμα, δημιουργήσαμε πολλές διαφορετικές SPARQL ερωτήσεις τροποποιώντας τον τύπο και το μέγεθος του graph pattern. Για αυτό το λόγο, αλλάξαμε

**Πίνακας 5.11:** Χαρακτηριστικά Μετάφρασης σε σχέση με τον Αριθμό των Tripple Patterns ($n$)

| Graph Pattern Type | Characteristics w.r.t. Number of Triple Patterns ($n$) | | |
|---|---|---|---|
| | SPARQL Variables | Generated XQuery Clauses | XPath Set Operations |
| $GP_1$ | $2n$ | $2n$ For/Let, 1 Where and 1 Return | $2n$ ($\bar{\cap}$) and $n$ ($\leqslant$) |
| $GP_2$ | $n$ | $n$ For/Let, 1 Where and 1 Return | $n$ ($\bar{\cap}$) |
| $GP_3$ | $3n$ | $3n$ For/Let, 1 Where and 1 Return | $2n$ ($\bar{\cap}$), $n$ ($\geqslant$) and $n$ ($\leqslant$) |
| $GP_4$ | $2n$ | $2n$ For/Let, 1 Where and 1 Return | $2n$ ($\bar{\cap}$) and $n$ ($\geqslant$) |

**Πίνακας 5.12:** Query translation time & SPARQL parsing time vs. Graph Pattern

| Query Translation Time [*SPARQL Parsing Time*] | | | | | | | |
|---|---|---|---|---|---|---|---|
| Graph Pattern Type | Number of Triple Patterns ($n$) | | | | | | |
| | 1 | 3 | 7 | 10 | 15 | 20 | 30 |
| $GP_1$ | 2.09 [*0.14*] | 2.13 [*0.15*] | 2.17 [*0.17*] | 2.20 [*0.66*] | 2.37 [*0.69*] | 2.91 [*0.70*] | 3.93 [*0.72*] |
| $GP_2$ | 2.07 [*0.38*] | 2.07 [*0.37*] | 2.11 [*0.38*] | 2.13 [*0.39*] | 2.29 [*0.42*] | 2.79 [*0.46*] | 3.81 [*0.65*] |
| $GP_3$ | 3.22 [*0.22*] | 3.26 [*0.24*] | 3.28 [*0.29*] | 3.39 [*0.32*] | 3.74 [*0.37*] | 3.89 [*0.41*] | 4.25 [*0.46*] |
| $GP_4$ | 3.21 [*0.21*] | 3.26 [*0.24*] | 3.29 [*0.28*] | 3.35 [*0.30*] | 3.64 [*0.31*] | 3.76 [*0.34*] | 4.04 [*0.40*] |
| Average | 2.65 [*0.24*] | 2.68 [*0.25*] | 2.71 [*0.28*] | 2.76 [*0.42*] | 3.01 [*0.45*] | 3.34 [*0.48*] | 4.01 [*0.56*] |

(α) τον αριθμό, και (β) τον τύπο των triple patterns που περιλαμβάνονται στο graph pattern.

Έχουμε ορίσει τέσσερις τύπους graph patterns ($GP_1$, $GP_2$, $GP_3$ και $GP_4$) τροποποιώντας τους τύπος των περιλαμβανομένων triple patterns:

(α) $GP_1 = ?x_1\ P_1\ ?y_1.\ ?x_2\ P_2\ ?y_2 \cdots ?x_n\ P_n\ ?y_n$

(β) $GP_2 = ?x_1\ P_1\ \text{“}abc\text{”}.\ ?x_2\ P_2\ \text{“}abc\text{”} \cdots ?x_n\ P_n\ \text{“}abc\text{”}$

(γ) $GP_3 = ?x_1\ ?y_1\ ?z_1.\ ?x_2\ ?y_2\ ?z_2. \cdots ?x_n\ ?y_n\ ?z_n$ και

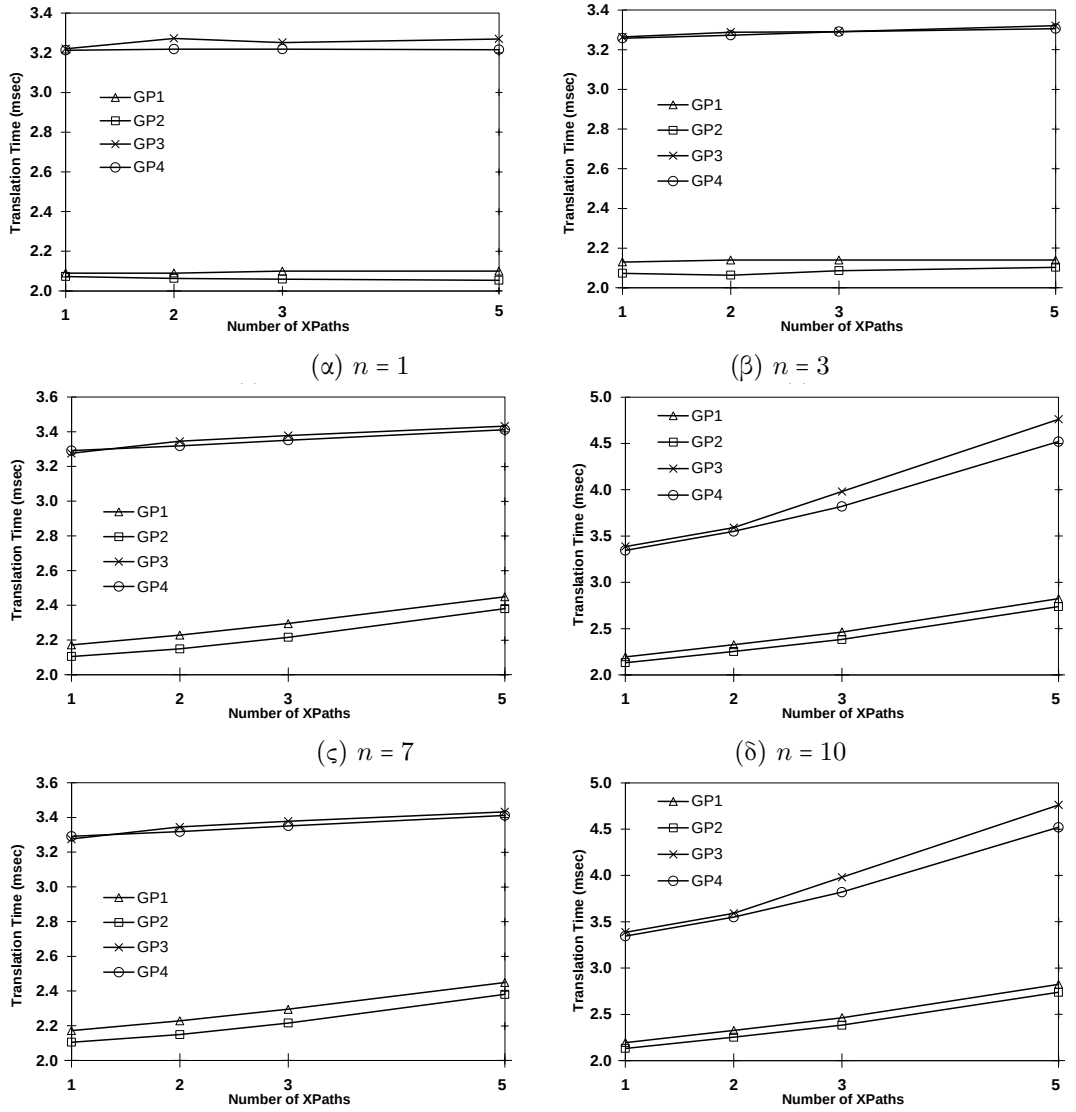
(δ) $GP_4 = ?x_1\ ?y_1\ \text{“}abc\text{”}.\ ?x_2\ ?y_2\ \text{“}abc\text{”} \cdots ?x_n\ ?y_n\ \text{“}abc\text{”}$,

όπου $n$ είναι ο αριθμός των triple patterns. Ο Πίνακας 5.11 παρουσιάζει τα βασικά χαρακτηριστικά της μετάφρασης από SPARQL σε XQuery για τους προηγούμενους τύπους του graph pattern. Η τελευταία στήλη αναφέρεται στις λειτουργίες του XPath Set που προκύπτουν στο στάδιο της σύνδεσης μεταβλητών.

Για καθέναν από τους παραπάνω τύπους graph pattern ($GP_1$, $GP_2$, $GP_3$ και $GP_4$), έχουμε δημιουργήσει graph patterns που περιέχουν n triple patterns με $n = 1, 3, 7, 10, 15, 20, 30$. Τέλος, για κάθε ιδιότητα οντολογίας $P_i$, έχουμε υποθέσει την αντιστοιχία $P_i \equiv \{/a/b/i\}$. Ο απαιτούμενος από το SPARQL2XQuery Framework χρόνος μετάφρασης από SPARQL σε XQuery για κάθε ερώτηση παρουσιάζεται στον Πίνακα 5.12. Ο SPARQL χρόνος ανάλυσης (parsing time) παρουσιάζεται επίσης στον Πίνακα 5.12.

#### 5.12.2.1.2 Μεταβάλλοντας τον αριθμό των Αντιστοιχίσεων

Σε αυτό το πείραμα έχουμε χρησιμοποιήσει τις SPARQL ερωτήσεις που έχουν δημιουργηθεί στο προηγούμενο πείραμα. Επιπλέον, έχουμε τροποποιήσει τον αριθμό των προκαθορισμένων αντιστοιχίσεων μεταξύ της οντολογίας και του XML Schema. Συγκεκριμένα, έχουμε τροποποιήσει τον αριθμό των αντιστοιχίσεων (δηλαδή, το μέγεθος των XPath Sets) για όλες τις ιδιότητες οντολογίας $P_i$. Με αυτό τον τρόπο, έχουμε




**Σχήμα 5.7:** Χρονάς Μετάφρασης vs. Αριθμό Αντιστοιχίσεων

αλλάξει την πολυπλοκότητα του σταδίου σύνδεσης μεταβλητών.

Το Σχήμα 5.7 παρουσιάζει τον χρόνο μετάφρασης ερωτήσεων με διαφορετικά πλήθη XPath εκφράσεων ανά αντιστοίχιση. Κάθε διάγραμμα στο Σχήμα 5.7 αντιστοιχεί σε έναν συγκεκριμένο αριθμό ($n$) triple patterns και απεικονίζει όλους τους τύπους των graph pattern ($GP_1$, $GP_2$, $GP_3$ και $GP_4$), ενώ ποικίλει ο αριθμός των XPath εκφράσεων από 1 σε 5.

#### 5.12.2.2 Χρόνος Μετάφρασης για τα Persons, DBLP & Berlin Query Sets

##### 5.12.2.2.1 Σύνολα Ερωτήσεων

Σε αυτή την ενότητα παρουσιάζουμε τα τρία σύνολα ερωτήσεων (query sets) που έχουν χρησιμοποιηθεί στα πειράματά μας. Το πρώτο σύνολο ερωτήσεων αποτελείται από 12 SPARQL ερωτήσεις του Berlin SPARQL Benchmark [79]. Μια σύνοψη των SPARQL χαρακτηριστικών που χρησιμοποιούνται από το Berlin query set μπορούν να βρεθούν στο [73]. Το δεύτερο σύνολο ερωτήσεων (Persons Queries) περιέχει 15 SPARQL

**Πίνακας 5.13:** Χρόνος Μετάφρασης & Χρόνος SPARQL Ανάλυσης

**(a) Person Query Set**

| Persons Query | Translation Time | SPARQL Parsing Time |
|---|---|---|
| $Q_1$ | 3.35 | 0.90 |
| $Q_2$ | 3.35 | 0.80 |
| $Q_3$ | 3.31 | 0.97 |
| $Q_4$ | 3.32 | 0.74 |
| $Q_5$ | 3.34 | 0.62 |
| $Q_6$ | 3.30 | 0.50 |
| $Q_7$ | 3.32 | 0.87 |
| $Q_8$ | 6.23 | 0.49 |
| $Q_9$ | 6.46 | 0.68 |
| $Q_{10}$ | 3.26 | 0.34 |
| $Q_{11}$ | 3.30 | 0.39 |
| $Q_{12}$ | 3.29 | 0.39 |
| $Q_{13}$ | 3.28 | 0.50 |
| $Q_{14}$ | 3.26 | 0.32 |
| $Q_{15}$ | 3.26 | 0.29 |
| **Avg.** | **3.71** | **0.59** |

**(b) DBLP Query Set**

| DBLP Query | Translation Time | SPARQL Parsing Time |
|---|---|---|
| $Q_1$ | 5.73 | 1.2 |
| $Q_2$ | 4.19 | 1.4 |
| $Q_3$ | 7.70 | 1.2 |
| $Q_4$ | 7.62 | 1.0 |
| $Q_5$ | 3.89 | 0.6 |
| **Avg.** | **5.83** | **1.1** |

**(c) Berlin Query Set**

| Berlin Query | Translation Time | SPARQL Parsing Time |
|---|---|---|
| $Q_1$ | 4.04 | 1.29 |
| $Q_2$ | 13.82 | 0.90 |
| $Q_3$ | 10.54 | 0.88 |
| $Q_4$ | 7.26 | 0.82 |
| $Q_5$ | 3.85 | 0.99 |
| $Q_6$ | 3.61 | 0.50 |
| $Q_7$ | 16.11 | 0.79 |
| $Q_8$ | 19.02 | 0.71 |
| $Q_9$ | 3.55 | 0.28 |
| $Q_{10}$ | 3.70 | 0.51 |
| $Q_{11}$ | 6.63 | 0.29 |
| $Q_{12}$ | 3.72 | 0.48 |
| **Avg.** | **7.99** | **0.70** |

ερωτήσεις βασισμένες στην οντολογία Persons (Πίνακας 6.5 και Πίνακας 6.6). Το τρίτο σύνολο ερωτήσεων (DBLP Queries) περιέχει 5 SPARQL ερωτήσεις βασισμένες στην DBLP οντολογία. Τα δύο τελευταία σύνολα ερωτήσεων έχουν χρησιμοποιηθεί για την αξιολόγηση του συστήματός μας σε σχέση με τον : (α) χρόνο μετάφρασης, και (β) χρόνο αποτίμησης ερωτήσεων.

#### 5.12.2.2.2 Αποτελέσματα

Σε αυτό το πείραμα έχουμε αξιολογήσει την αποδοτικότητα της διαδικασίας μετάφρασης χρησιμοποιώντας αρκετές διαφορετικές SPARQL ερωτήσεις. Για κάθε ερώτηση, έχουμε μετρήσει τον χρόνο μετάφρασης που απαιτείται από το SPARQL2XQuery Framework προκειμένου να μεταφράσει τις SPARQL ερωτήσεις σε XQuery εκφράσεις. Ο χρόνος μετάφρασης ερωτήσεων και ο SPARQL χρόνος ανάλυσης καθώς επίσης και ο μέσος χρόνος ανάλυσης και μετάφρασης για κάθε σύνολο ερωτήσεων παρουσιάζεται στον Πίνακα 5.13.

Ο Πίνακας 5.13α παρουσιάζει τους χρόνους μετάφρασης για 15 ερωτήσεις του συνόλου ερωτήσεων Persons. Επίσης, ο χρόνος μετάφρασης για τις ερωτήσεις του DBLP συνόλου ερωτήσεων παρουσιάζονται στον Πίνακα 5.13β. Τέλος, ο Πίνακας 5.13γ παρουσιάζει τους χρόνους μετάφρασης για το Berlin query set.

### 5.12.3 Αποδοτικότητα της Αποτίμησης Ερωτήσεων

Σε αυτή την ενότητα παρουσιάζουμε τα πειραματικά αποτελέσματα που αναφέρονται στην αποδοτικότητα της αποτίμησης XQuery εκφράσεων που δημιουργούνται από το SPARQL2XQuery Framework.

#### 5.12.3.1 Μεθοδολογία

**Σύνολα Δεδομένων.** Προκειμένου να αξιολογήσουμε το SPARQL2XQuery Framework σε σχέση με την αποδοτικότητα της αποτίμησης ερωτήσεων, έχουμε χρησιμοποιήσει τόσο πραγματικά όσο και συνθετικά σύνολα δεδομένων. Το πραγματικό σύνολο δεδομένων που χρησιμοποιήσαμε είναι το XML DBLP dataset.

**Πίνακας 5.14:** Χαρακτηριστικά των Συνόλων Δεδομένων Persons

| XML Dataset Characteristics | | | | Corresponding RDF Dataset Characteristics | |
|---|---|---|---|---|---|
| **Dataset Name** | ***N*** | **XML Nodes** | **Size** (Kb) | **Triples** | **Size** (Kb) |
| **DT$_1$** | $10^2$ | 1450 | 20 | $6{\cdot}10^2$ | 40 |
| **DT$_2$** | $5{\cdot}10^2$ | 7250 | $10^2$ | $3{\cdot}10^3$ | $2{\cdot}10^2$ |
| **DT$_3$** | $10^3$ | $145{\cdot}10^2$ | $2{\cdot}10^2$ | $6{\cdot}10^3$ | $4{\cdot}10^2$ |
| **DT$_4$** | $5{\cdot}10^3$ | $725{\cdot}10^2$ | $10^3$ | $3{\cdot}10^4$ | $2{\cdot}10^3$ |
| **DT$_5$** | $10^4$ | $145{\cdot}10^3$ | $2{\cdot}10^3$ | $6{\cdot}10^4$ | $4{\cdot}10^3$ |
| **DT$_6$** | $5{\cdot}10^4$ | $725{\cdot}10^3$ | $10^4$ | $3{\cdot}10^5$ | $2{\cdot}10^4$ |
| **DT$_7$** | $10^5$ | $145{\cdot}10^4$ | $2{\cdot}10^4$ | $6{\cdot}10^5$ | $4{\cdot}10^4$ |
| **DT$_8$** | $5{\cdot}10^5$ | $725{\cdot}10^4$ | $10^5$ | $3{\cdot}10^6$ | $2{\cdot}10^5$ |
| **DT$_9$** | $10^6$ | $145{\cdot}10^5$ | $2{\cdot}10^5$ | $6{\cdot}10^6$ | $4{\cdot}10^5$ |
| **DT$_{10}$** | $5{\cdot}10^6$ | $725{\cdot}10^5$ | $10^6$ | $3{\cdot}10^7$ | $2{\cdot}10^6$ |

Το συνθετικό σύνολο δεδομένων είναι δομημένα σύμφωνα με το Persons XML Schema (Σχήμα 6.2). Ο Πίνακας 5.14 συνοψίζει τα βασικά χαρακτηριστικά των συνόλων δεδομένων Persons XML, συμπεριλαμβανομένου του μεγέθους τους σε Kilobytes, τον κατά προσέγγιση αριθμό των XML κόμβων, κτλ.

**Ερωτήσεις.** Στη δική μας περίπτωση αξιολόγησης, κάθε SPARQL ερώτηση $Q_S$ των Persons και DBLP συνόλων ερωτήσεων, έχει αυτόματα μεταφραστεί από το SPARQL2XQuery Framework σε XQuery ερώτηση $Q_{X_a}$. Επιπλέον, η $Q_S$ έχει ανεξάρτητα μεταφραστεί χειροκίνητα από έναν ειδικό στην XQuery $Q_{X_m}$. Οι $Q_{X_m}$ ερωτήσεις έχουν εκφραστεί σύμφωνα με τις σημασιολογίες του XML Schema και αφού εφαρμόστηκαν οι τεχνικές που στόχευαν στην παροχή αποδοτικών XQuery ερωτήσεων. Τέλος, οι κανόνες αναδιατύπωσης που ορίστηκαν στη Ενότητα 5.10 έχουν εφαρμοστεί στις αυτομάτως δημιουργούμενες XQuery ερωτήσεις ($Q_{X_a}$), προκειμένου να αποκτήσουμε τις αυτόματα αναδιατυπωμένες XQuery ερωτήσεις $Q_{X_a\text{-}Rw}$.

**Μετρικές Αξιολόγησης.** Προκειμένου να μελετήσουμε την αποδοτικότητα των XQuery ερωτήσεων, έχουμε μετρήσει και συγκρίνει τους χρόνους αποτίμησης ερωτήσεων για τις (α) αυθεντικές SPARQL ερωτήσεις, εκτελεσμένες χρησιμοποιώντας το SPARQL engine, (β) τις αυτόματα παραγόμενες ($Q_{X_a}$) XQuery ερωτήσεις, (γ) τις αυτόματα αναδιατυπώμενες ($Q_{X_a\text{-}Rw}$) XQuery ερωτήσεις, και (δ) τις χειροκίνητα μεταφρασμένες ($Q_{X_m}$) XQuery ερωτήσεις. Σημειώνουμε ότι οι XQuery χρόνοι αποτίμησης στηρίζονται αρκετά από τα χαρακτηριστικά του συστήματος διαχείρισης των XML δεδομένων (π.χ., αποθήκευση, ευρετηρίαση (indexing), query engine, query optimizer, κτλ.).

#### 5.12.3.2 Συνθετικό Σύνολο Δεδομένων

Σε αυτό το πείραμα μελετάμε την αποδοτικότητα των XQuery ερωτήσεων που έχουν δημιουργηθεί από το SPARQL2XQuery Framework χρησιμοποιώντας συνθετικά σύνολα δεδομένων (Πίνακας 5.14).

**Πίνακας 5.15:** Χρόνος Αποτίμησης Ερωτήσεων στο Σύνολο Δεδομένων $DT_8$ (XML Store Y)

| **Query Evaluation Time** (sec) | | | | | | |
|---|---|---|---|---|---|---|
| **Query** | **SPARQL** ($Q_S$) | **Manual** ($Q_{Xm}$) | **Auto-Rw** ($Q_{Xa\text{-}Rw}$) | **Auto** ($Q_{Xa}$) | **Auto-Rw vs. Auto** | **Auto-Rw vs. Manual** |
| **$Q_1$** | 1.66 | 5.95 | 4.30 | 6.78 | 57.7 % | 27.7 % |
| **$Q_2$** | 1.69 | 5.96 | 4.28 | 6.76 | 57.8 % | 28.1 % |
| **$Q_3$** | 1.53 | 0.41 | 0.42 | 0.45 | 7.6 % | -1.0 % |
| **$Q_4$** | 2.78 | 10.79 | 11.00 | 11.08 | 0.7 % | -1.9 % |
| **$Q_5$** | 10.83 | 55.70 | 55.77 | 63.97 | 14.7 % | -0.1 % |
| **$Q_6$** | 1.55 | 6.55 | 6.49 | 6.89 | 6.1 % | 0.9 % |
| **$Q_7$** | 1.36 | 0.91 | 0.92 | 0.93 | 1.2 % | -0.2 % |
| **$Q_8$** | 6.03 | 12.93 | 13.09 | 13.11 | 0.2 % | -1.3 % |
| **$Q_9$** | 5.34 | 3.21 | 3.22 | 5.76 | 79.1 % | -0.3 % |
| **$Q_{10}$** | 0.00 | 6.63 | 5.74 | 6.91 | 20.4 % | 13.4 % |
| **$Q_{11}$** | 21.74 | 14.89 | 15.07 | 16.47 | 9.3 % | -1.2 % |
| **$Q_{12}$** | 2.44 | 15.47 | 15.49 | 15.74 | 1.6 % | -0.1 % |
| **$Q_{13}$** | 0.00 | 0.23 | 0.24 | 0.25 | 5.5 % | 2.1 % |
| **$Q_{14}$** | 1.37 | 3.69 | 3.61 | 3.80 | 5.2 % | 2.2 % |
| **$Q_{15}$** | 2.74 | 9.14 | 15.69 | 15.88 | 1.2 % | -71.7 % |
| **Average** | **4.07** | **10.17** | **10.36** | **11.65** | **12.5 %** | **-1.9 %** |

#### 5.12.3.2.1 Ανάλυση του Χρόνου Αποτίμησης Ερωτήσεων

Έχουμε χρησιμοποιήσει το συνθετικό Persons dataset $DT_8$ και το Persons σύνολο ερωτήσεων. Το $DT_8$ σύνολο περιλαμβάνει $5 \cdot 10^5$ καταχωρήσεις ατόμων (persons) και μαθητών (students) (250K persons και 250K students), έχει μέγεθος 105Kb και έχει περίπου $725 \cdot 10^4$ XML κόμβους.

Ο Πίνακας 5.15 συνοψίζει τα αποτελέσματα της σύγκρισης της εκτέλεσης του SPARQL, με τις XQuery ερωτήσεις που δημιουργούνται αυτόματα, αναδιατυπώνονται και μεταφράζονται χειροκίνητα. Συγκεκριμένα, για κάθε ερώτηση, ο Πίνακας 5.15περιέχει τους χρόνους αποτίμησης για (α) τις SPARQL ερωτήσεις (που υποδηλώνεται ως SPARQL), (β) τις χειροκίνητα μεταφρασμένες XQuery ερωτήσεις (που υποδηλώνεται ως *Manual*), (γ) τις αυτόματα αναδιατυπώμενες XQuery ερωτήσεις (που υποδηλώνεται ως *Auto-Rw*), και (δ) τις αυτόματα δημιουργημένες (χωρίς αναδιατύπωση) XQuery ερωτήσεις (που υποδηλώνεται ως *Auto*). Επιπρόσθετα, ο Πίνακας 5.15 παρουσιάζει τη βελτίωση των αναδιατυπωμένων ερωτήσεων σε σχέση με τις αυτομάτως δημιουργημένες ερωτήσεις (που υποδηλώνεται ως *Auto-Rw vs. Auto*) καθώς και σύγκριση μεταξύ των αυτομάτως αναδιατυπωμένων και των χειροκίνητα μεταφρασμένων XQuery ερωτήσεων (που υποδηλώνεται ως *Auto-Rw vs. Manual*). Η μονάδα μέτρησης για τον χρόνο αποτίμησης είναι το δευτερόλεπτο (sec).

**Αυτομάτως Αναδιατυπωμένες vs. Αυτομάτως Δημιουργημένες Ερωτήσεις (Auto-Rw vs. Auto)**. Μπορούμε να παρατηρήσουμε από τον Πινακα 5.15 ότι για όλες σχεδόν τις ερωτήσεις οι χρόνοι αποτίμησης για τις αναδιατυπωμένες ερωτήσεις παρουσιάζουν μια αξιοσημείωτη βελτίωση στην απόδοση σε σχέση με τις αυτομάτως δημιουργημένες. Η μέση μείωση στον χρόνο αποτίμησης για τις αναδιατυπωμένες ερωτήσεις ήταν 12.5%, και το μέγιστο 79.1%.

Για πέντε ($Q_4$, $Q_7$, $Q_8$, $Q_{12}$ και $Q_{15}$) από τις δεκαπέντε ερωτήσεις, ο χρόνος αποτίμησης ερωτήσεων ήταν περίπου ίδιος για τις αναδιατυπωμένες και τις αυτομάτως δημιουργημένες ερωτήσεις (με μία μείωση χρόνου μεταξύ 0.2% και 1.6 %). Για εφτά ερωτήσεις ($Q_3$, $Q_5$, $Q_6$, $Q_{10}$, $Q_{11}$, $Q_{13}$ και $Q_{14}$), οι αναδιατυπωμένες ερωτήσεις έχουν παρουσιάσει μια ελαφριά βελτίωση με μια μείωση χρόνου αποτίμησης μεταξύ 5.2% και 20.4% σε σύγκριση με τια αυτομάτως δημιουργημένες. Τέλος, τρεις ερωτήσεις ($Q_1$, $Q_2$ και $Q_9$) έχουν παρουσιάσει μια σημαντική βελτίωση επίδοσης με μια μείωση χρόνου της τάξης του 57.7% με 79.1%.

**Αυτομάτως Αναδιατυπωμένες vs. Χειροκίνητα Μεταφρασμένες Ερωτήσεις (Auto-Rw vs. Manual).** Οι χρόνοι αποτίμησης των αυτομάτως αναδιατυπωμένων ερωτήσεων είναι παραπλήσιοι με εκείνους των χειροκίνητα μεταφρασμένων ερωτήσεων όπως φαίνεται και στον Πίνακα 5.15, με μια μέση αύξηση της τάξης του 1.9%.

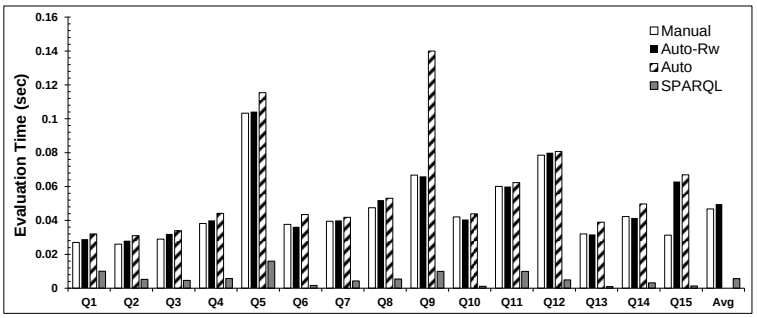
Για τρεις ($Q_1$, $Q_2$ και $Q_{10}$) από τις δεκαπέντε ερωτήσεις, οι αναδιατυπώμενες ερωτήσεις έχουν καλύτερες επιδόσεις από τις χειροκίνητα μεταφρασμένες, με μείωση χρόνου αποτίμησης της τάξης του 13.4% με 28.1%. Επιπλέον, σε άλλες περιπτώσεις ($Q_6$, $Q_9$, $Q_{13}$ και $Q_{14}$), οι αναδιατυπώμενες ερωτήσεις έχουν δείξει μια στοιχειώδη βελτίωση (με μείωση χρόνου αποτίμησης μεταξύ του 0.9% και 13.4%) σε σύγκριση με τις χειροκίνητα μεταφρασμένες. Για τις υπόλοιπες ερωτήσεις, (με την εξαίρεση της ερώτησης $Q_{15}$), ο χρόνος αποτίμησης των αναδιατυπωμένων και των αυτομάτως μεταφρασμένων ερωτήσεων ήταν περίπου ο ίδιος. Για την ερώτηση $Q_{15}$, η χειροκίνητη μετάφραση έχει δείξει μια αξιοσημείωτη αύξηση του χρόνου αποτίμησης (71.7%).

#### 5.12.3.2.2 Μεταβάλλοντας το Μέγεθος του Συνόλου Δεδομένων

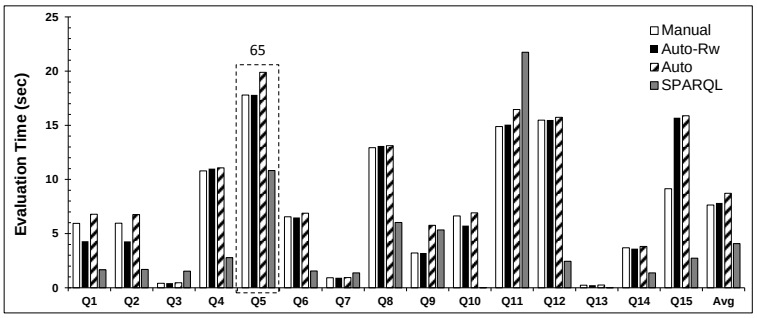
Προκειμένου να μελετήσουμε την αποδοτικότητα της αποτίμησης ερωτήσεων πάνω στο μέγεθος του συνόλου δεδομένων, έχουμε χρησιμοποιήσει 10 συνθετικά Persons XML σύνολα δεδομένων. Πρώτα παρουσιάζουμε μια σύνοψη της επίδρασης του μεγέθους του συνόλου δεδομένων στον χρόνο αποτίμησης. Στα επόμενα σχήματα, παρουσιάζουμε τα αποτελέσματα που αποκτήθηκαν χρησιμοποιώντας διαφορετικές XQuery engines. Συγκεκριμένα, το Σχήμα 5.8 αντιστοιχεί στο XML Store Y, το Σχήμα 5.9 αντιστοιχούν στο XML Store Z, και το Σχήμα 5.10 αντιστοιχεί στην Memory-based XQuery Engine. Τέλος, το Σχήμα 5.11 παρέχει μια λεπτομερή θεώρηση στο χρόνο αποτίμησης ερώτησης πάνω στο μέγεθος του συνόλου δεδομένων.

Παρατηρούμε ότι οι χρόνοι αποτίμησης τόσων των χειροκίνητων όσο και των αυτομάτως αναδιατυπωμένων ερωτήσεων έχουν παρόμοια επίδοση. Όσο μεγαλώνει το μέγεθος του συνόλου δεδομένων, ο χρόνος αποτίμησης αυξάνει υπο-γραμμικά (sublinearly) για δεδομένο σύνολο δεδομένων. Για κάποιες ερωτήσεις, η αύξηση είναι λιγότερο έντονη από ότι σε άλλες (π.χ, ερωτήσεις 3, 7, 9), λόγω της υψηλής επιλεκτικότητας (δηλαδή, μικρό σύνολο αποτελεσμάτων) αυτών των ερωτήσεων. Όμως, για όλες τις ερωτήσεις, η αύξηση είναι πιο έντονη για σύνολα δεδομένων μεγαλύτερα των $10^5$ καταχωρήσεων. Τέλος, με εξαίρεση τις ερωτήσεις 7, 10, 11 και 12 όπου οι χρόνοι αποτίμησης είναι σχεδόν ίσοι από το μικρότερο σύνολο δεδομένων στο μεγαλύτερο, καθώς το μέγεθος του συνόλου δεδομένων αυξάνει, η διαφορά μεταξύ των χρόνων αποτίμησης μειώνεται, με τις περισσότερες ερωτήσεις να έχουν σχεδόν τους ίδιους χρόνους για τα μεγαλύτερα σύνολα δεδομένων ($DT_7$ σε $DT_{10}$).

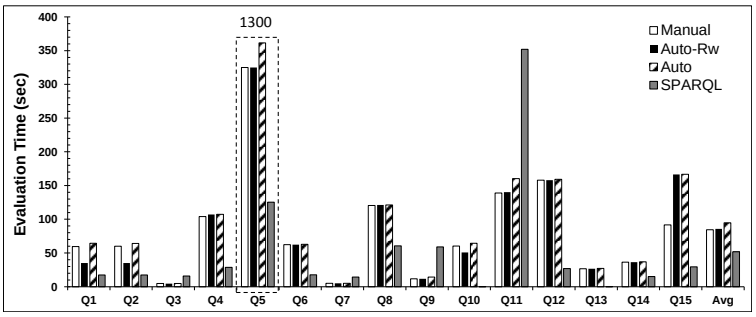
Οι μέσοι χρόνοι αποτίμησης (Σχήμα 5.11χ) αυξάνουν πολύ γρήγορα με την αύξηση να είναι πιό έντονη για σύνολα δεδομένων μεγαλύτερα των $10^5$ καταχωρήσεων




(α) $DT_1$ Persons (XML Store Y)




(β) $DT_8$ Persons (XML Store Y)




(γ) $DT_{10}$ Persons (XML Store Y)

**Σχήμα 5.8:** Χρόνος Αποτίμησης Ερωτήσεων στα Persons Σύνολα Δεδομένων $DT_1$, $DT_8$ και $DT_{10}$ (XML Store Y)

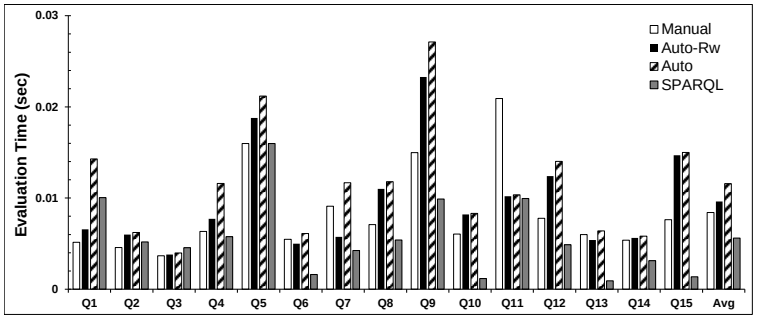
(records). Επιπλέον, καθώς το μέγεθος του συνόλου δεδομένων αυξάνει, η διαφορά μεταξύ των χρόνων μειώνεται.

Τα αποτελέσματα δείχνουν ότι ακόμα και χωρίς εκτεταμένη βελτιστοποίηση, μπορεί να επιτευχθεί μια αξιοσημείωτη βελτίωση επίδοσης. Ο χρόνος αποτίμησης ερωτήσεων μειώνεται κατά μέσο όρο κατά 13% σε σύγκριση με τις μη αναδιατυπωμένες, με μέγιστη μείωση της τάξης του 83% σε μερικές περιπτώσεις. Ακόμα και οι αυτομάτως δημιουργημένες ερωτήσεις έχουν λογική επίδοση και κλιμακώνουν (scale) αρκετά καλά για μεγέθη μέχρι $725 \cdot 10^5$ XML κόμβους.

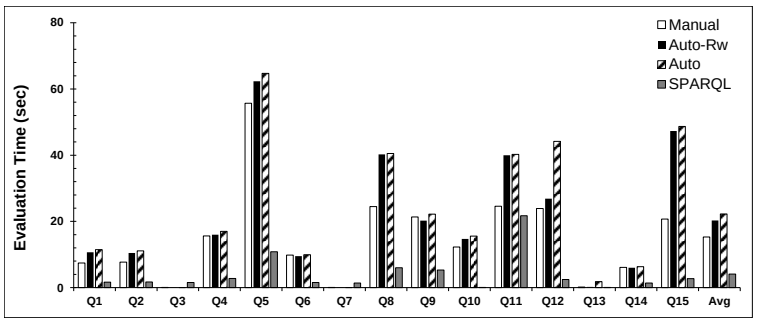
#### 5.12.3.3 Πραγματικά Σύνολα Δεδομένων

Σε αυτό το πείραμα έχουμε μελετήσει την αποδοτικότητα των αυτομάτως δημιουργημένων XQuery ερωτήσεων χρησιμοποιώντας ένα πραγματικό σύνολο δεδομένων. Ο Πίνακας 5.16 συνοψίζει τα πειραματικά αποτελέσματα.

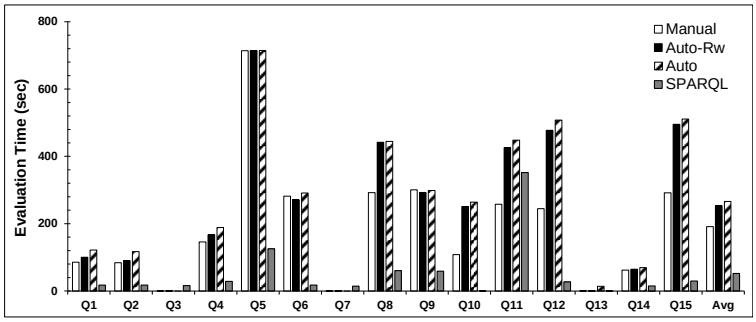
**Αυτομάτως Αναδιατυπωμένες vs. Αυτομάτως Δημιουργημένες (Auto-Rw vs. Auto) ερωτήσεις.** Ο Πίνακας 5.16 δείχνει ότι οι χρόνοι αποτίμησης για τις




(α) $DT_1$ Persons (XML Store Z)




(β) $DT_8$ Persons (XML Store Z)




(γ) $DT_{10}$ Persons (XML Store Z)

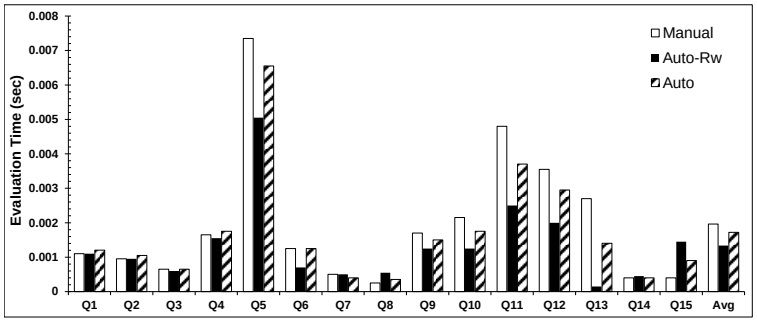
**Σχήμα 5.9:** Χρόνος Αποτίμησης Ερωτήσεων στα Persons Σύνολα Δεδομένων $DT_1$, $DT_8$ και $DT_{10}$ (XML Store Z)

αναδιατυπώμενες ερωτήσεις παρουσιάζουν μια σημαντική βελτίωση επίδοσης σε σύγκριση με τις αυτομάτως δημιουργημένες ερωτήσεις, με μέση μείωση χρόνου της τάξης του 13.8%.

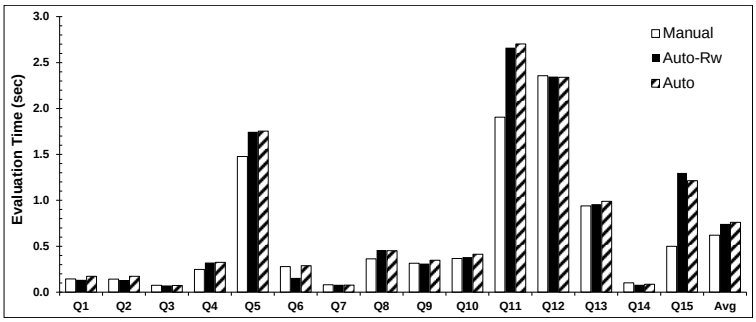
**Αυτομάτως Αναδιατυπωμένες vs. Χειροκίνητα Μεταφρασμένες (Auto-Rw vs. Manual) ερωτήσεις**. Μπορούμε να δούμε από τον Πίνακα 5.16 ότι οι χρόνοι αποτίμησης των αυτομάτως δημιουργημένων ερωτήσεων είναι σχεδόν ίδιοι με εκείνον των χειροκίνητα μεταφρασμένων ερωτήσεων, με μέση αύξηση χρόνου της τάξης του 2.2%.

Τέλος, μπορούμε να παρατηρήσουμε από τον Πίνακα 5.16 ότι η επίδοση της αποτίμησης ερωτήσεων για τα DBLP σύνολα δεδομένων είναι παρόμοια με εκείνη των συνθετικών συνόλων δεδομένων του ίδιου μεγέθους.

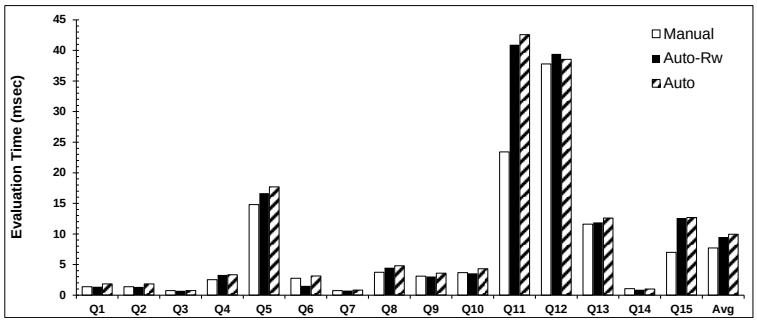
Στο σχήμα που ακολουθεί, παρουσιάζουμε τα αποτελέσματα που αποκτήθηκαν χρησιμοποιώντας διαφορετικές XQuery engines. Συγκεκριμένα, το Σχήμα 5.12α αντιστοιχεί στο XML Store Y, το Σχήμα 5.12β αντιστοιχεί στο XML Store Z, και το




(α) $DT_1$ Persons (Memory-based XQuery Engine)




(β) $DT_8$ Persons (Memory-based XQuery Engine)




(γ) $DT_{10}$ Persons (Memory-based XQuery Engine)

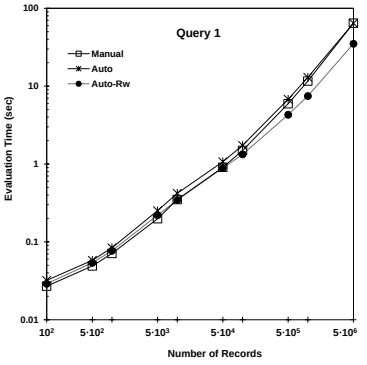
**Σχήμα 5.10:** Χρόνος Αποτίμησης Ερωτήσεων στα Persons Σύνολα Δεδομένων $DT_1$, $DT_8$ και $DT_{10}$ (Memory-based XQuery Engine)

Σχήμα 5.12γ αντιστοιχεί σε μια Memory-based XQuery Engine.

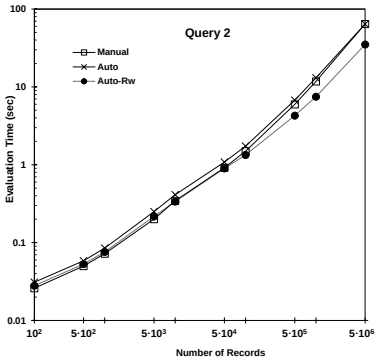
### 5.12.4 Επίλογος

Ο Ιστός δεδομένων (Web of Data-WoD) είναι ένα ανοιχτό περιβάλλον που αποτελείται από εκατοντάδες μεγάλα interlinked, user contributed σύνολα δεδομένων. Ο WoD βασίζεται σε τεχνολογίες και πρότυπα που αναπτύχθηκαν από την κοινότητα του Σημασιολογικού Ιστού (Semantic Web-SW) (δηλαδή, OWL, RDF/S, SPARQL) για αναπαράσταση και διαχείριση διαδικτυακών πληροφοριών. Αντίθετα, στην τωρινή διαδικτυακή υποδομή, τα XML/XML Schema είναι τα κυρίαρχα πρότυπα για την ανταλλαγή πληροφοριών, και για την αναπαράσταση ημιδομημένων πληροφοριών. Επιπλέον, πολλά διεθνή πρότυπα (π.χ, Dublin Core, MPEG-7) εκφράζονται σε XML Schema. Τα παραπάνω έχουν οδηγήσει σε αυξανόμενη έμφαση στα XML δεδομένα.

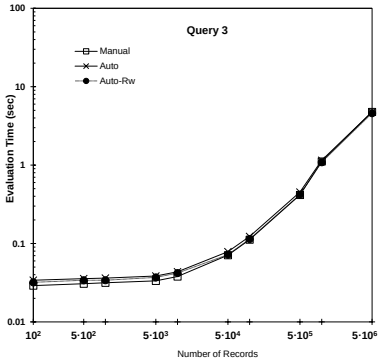
Στον WoD, οι χρήστες δεν πρέπει να αλληλεπιδρούν με διαφορετικά μοντέλα πληροφοριών και γλώσσες έκφρασης των ερωτήσεων τους. Επιπλέον, δεν είναι ρεαλιστικό



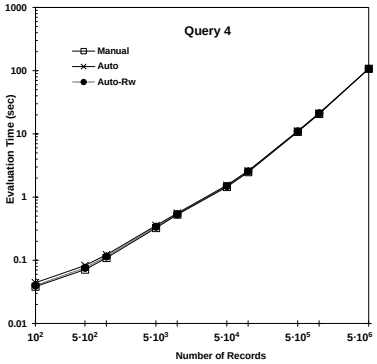

(α) Query 1



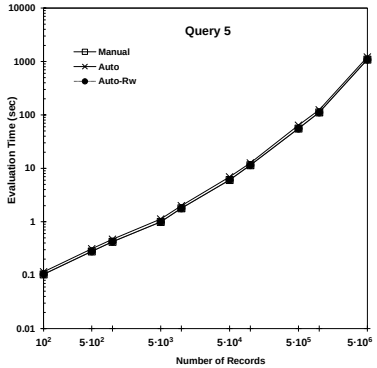

(β) Query 2



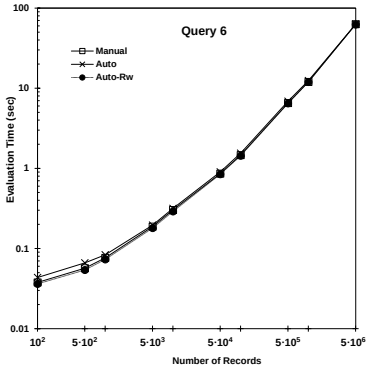

(ς) Χυερψ 3



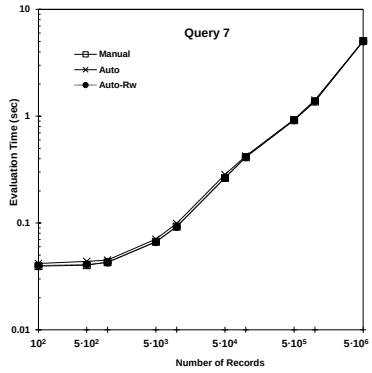

(δ) Query 4



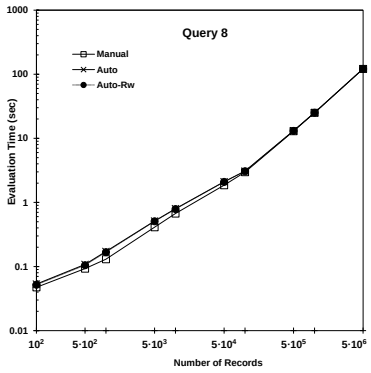

(ε) Query 5



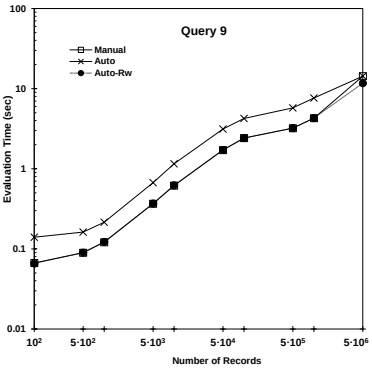

(φ) Query 6



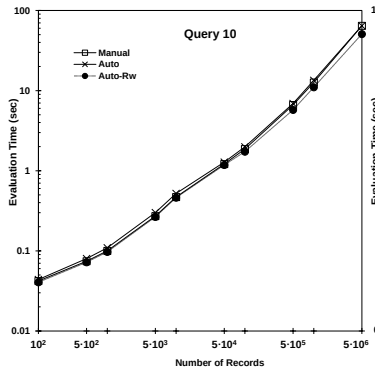

(γ) Query 7



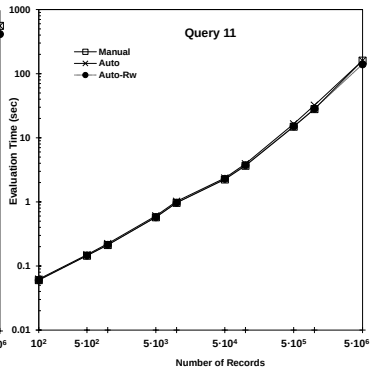

(η) Query 8



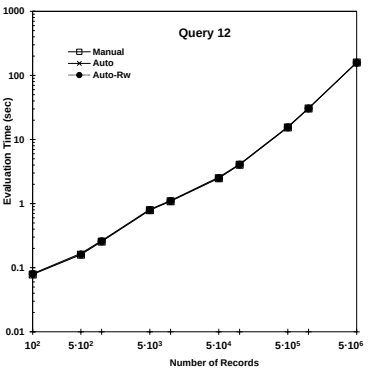

(ι) Query 9



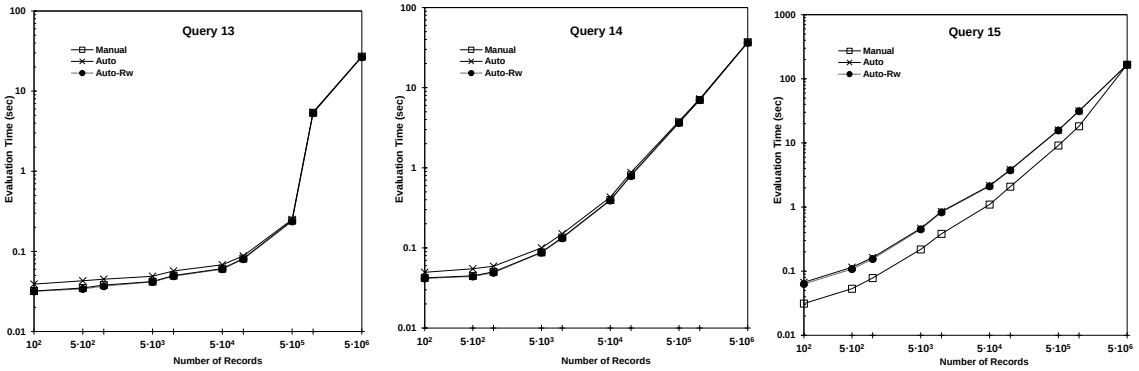

(κ) Query 10



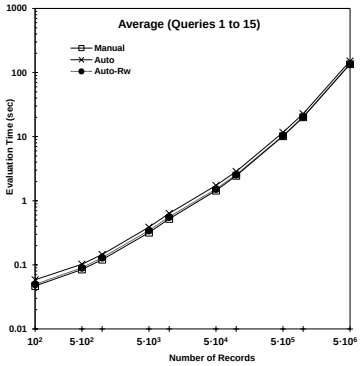

(λ) Query 11




(μ) Query 12




(ν) Query 13 (ο) Query 14 (π) Query 15

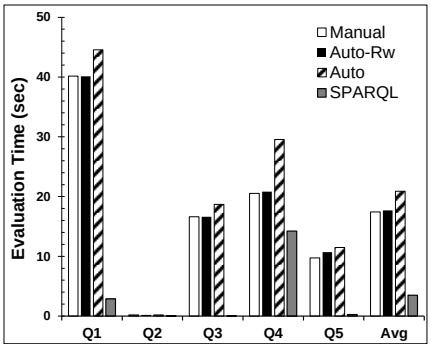



(χ) Average (Queries 1~15)

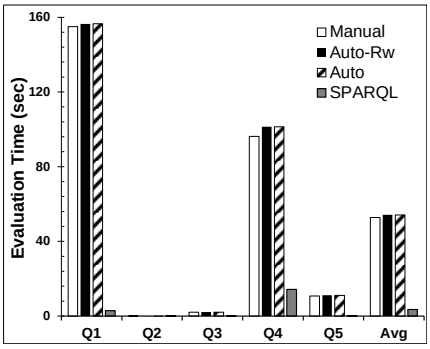
**Σχήμα 5.11:** Χρόνος Αποτίμησης Ερωτήσεων vs. Μέγεθος Δεδομένων (XML Store Y)

**Πίνακας 5.16:** Χρόνος Αποτίμησης Ερωτήσεων στο DBLP Σύνολο Δεδομένων (XML Store Y)

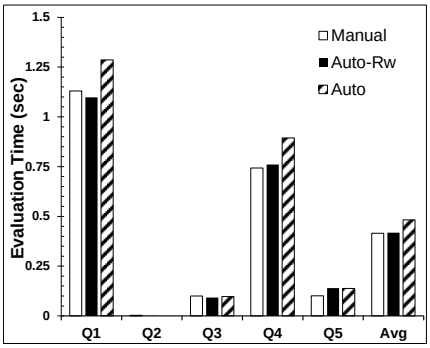
| **Query Evaluation Time** (sec) | | | | | | |
|---|---|---|---|---|---|---|
| **Query** | **SPARQL** ($Q_S$) | **Manual** ($Q_{Xm}$) | **Auto-Rw** ($Q_{Xa\text{-}Rw}$) | **Auto** ($Q_{Xa}$) | **Auto-Rw vs. Auto** | **Auto-Rw vs. Manual** |
| **$Q_1$** | 2.88 | 40.14 | 40.12 | 44.56 | 10.0 % | 0.1 % |
| **$Q_2$** | 0.07 | 0.19 | 0.19 | 0.21 | 11.2 % | 0.5 % |
| **$Q_3$** | 0.06 | 16.61 | 16.63 | 18.72 | 11.2 % | -0.1 % |
| **$Q_4$** | 14.24 | 20.52 | 20.82 | 29.57 | 29.6 % | -1.5 % |
| **$Q_5$** | 0.26 | 9.73 | 10.69 | 11.51 | 7.1 % | -9.9 % |
| **Average** | **3.50** | **17.44** | **17.69** | **20.92** | **13.8 %** | **-2.2 %** |




(α) XML Store Y




(β) ΞΜΛ Στορε Ζ




(γ) Memory-based ΞΧυερψ Ενγινε

**Σχήμα 5.12:** Χρόνος Αποτίμησης Ερωτήσεων στο DBLP Σύνολο Δεδομένων (Using different XQuery Engines)

να περιμένουμε ότι τα παραδοσιακά δεδομένα legacy data (δηλαδή, Relational, XML) θα μετατραπούν σε RDF data. Έτσι, είναι ζωτικής σημασίας να παρέχουμε μηχανισμούς διαλειτουργικότητας που επιτρέπουν στους WoD χρήστες να έχουν πρόσβαση με διαφάνεια σε εξωτερικούς ετερογενείς πόρους δεδομένων από το δικό τους περιβάλλον εργασίας. Τέλος, στην εποχή των διασυνδεδομένων δεδομένων, η παροχή SPARQL endpoints (δηλαδή, SPARQL-based υπηρεσιών αναζήτησης) σε παραδοσιακά δεδομένα αποτελεί μεγάλη ερευνητική πρόκληση.

Σε αυτήν την εργασία προτείναμε το SPARQL2XQuery Framework, το οποίο γεφυρώνει το κενό της ετερογένειας και δημιουργεί ένα διαλειτουργικό περιβάλλον μεταξύ των SW και XML worlds. Το SPARQL2XQuery Framework μπορεί να αποτελέσει βασικό στοιχείο για αρκετές WoD εφαρμογές, επιτρέποντας την εγκατάσταση SPARQL endpoints σε XML δεδομένα, καθώς και θεμελιώδες στοιχείο σε ontology-based integration frameworks που περιλαμβάνουν XML πόρους.

Το SPARQL2XQuery Framework επιτρέπει στις SPARQL ερωτήσεις που θέτονται σε οντολογίες να μεταφράζονται αυτόματα σε XQuery εκφράσεις που αποτιμούνται σε XML data με βάση ένα σύνολο προκαθορισμένων αντιστοιχίσεων. Για αυτό το λόγο, το Framework μας επιτρέπει τόσο τον χειροκίνητο όσο και τον αυτόματο ορισμό αντιστοιχίσεων (mapping specification) μεταξύ οντολογιών και XML Schemas. Τέλος, τα αποτελέσματα ερωτήσεων επιστρέφονται είτε σε μορφή RDF είτε σε SPARQL Query Result XML Format. Έτσι, οι WoD χρήστες δεν χρειάζεται να αλληλεπιδρούν με παραπάνω από ένα μοντέλο ή γλώσσες ερωτήσεων.

Πιο συγκεκριμένα, παρουσιάσαμε ένα μοντέλο αντιστοιχίσεων για την έκφραση των OWL–RDF/S σε XML Schema αντιστοιχίσεις, καθώς και μια μέθοδο για SPARQL σε XQuery μετάφραση. Από ότι γνωρίζουμε, αυτή είναι η πρώτη εργασία που επιλύει

αυτά τα ζητήματα. Επιπλέον, παρουσιάσαμε το XS2OWL, το οποίο επιτρέπει το μετασχηματισμό των XML Schemas σε OWL οντολογίες, χρησιμοποιώντας τις τελευταίες εκδόσεις των προτύπων (XML Schema 1.1. και OWL 2). Όσο γνωρίζουμε, αυτή είναι η πρώτη εργασία που εμπεριέχει πλήρως τις σημασιολογίες των XML Schema και υποστηρίζει τις XML Schema 1.1 constructs. Το XS2OWL έχει ενσωματωθεί στο SPARQL2XQuery framework προκειμένου να παρέχει αυτόματη δημιουργία και ανανέωση αντιστοιχίσεων.

Τέλος, διεξήγαγε και παρουσιάσαμε μια λεπτομερή πειραματική ανάλυση του SPARQL2XQuery framework προκειμένου να αναδειχτεί η αποδοτικότητα (α) του μετασχηματισμού σχήματος, (β) της δημιουργίας αντιστοιχίσεων, (γ) της μετάφρασης ερωτήσεων, και (δ) της αποτίμησης ερωτήσεων.

# Κεφάλαιο 6

# Σημασιολογική Ανάκτηση και Διερεύνηση

Σε αυτό το κεφάλαιο μελετάμε δυο προβλήματα. Το πρώτο πρόβλημα αφορά την σημασιολογικής ανάκτησης πληροφοριών. Για το πρόβλημα αυτό, προτείνουμε το πλαίσιο *GoNTogle* το οποίο υποστηρίζει ένα σημασιολογικό μοντέλο επισημειώσεων Το πλαίσιο παρέχει τόσο αυτόματο όσο και χειροκίνητο μηχανισμό επισημειώσεων. Ο μηχανισμό αυτόματων επισημειώσεων βασίζεται σε μια μέθοδο εκμάθησης η οποία χρησιμοποιεί το ιστορικό επισημειώσεων του χρήστη, καθώς και πληροφορίες κειμένου, προκειμένου να προτείνει αυτόματα επισημειώσεις για νέα κείμενα. Επιπλέον, εισάγουμε μια υβριδική μέθοδο ανάκτησης (hybrid retrieval method) που παρέχει έναν ευέλικτο συνδυασμό textual-based και semantic-based ανάκτησης σε συνδυασμό με ανεπτυγμένες σημασιολογικές λειτουργίες. Οι προτεινόμενες μέθοδοι εφαρμόζονται σε ένα πλήρως λειτουργικό εργαλείο και η αποτελεσματικότητά τους επιβεβαιώνεται πειραματικά.

Στην συνέχεια μελετάμε το πρόβλημα της μοντελοποίησης, δημοσίευσης και διερεύνησης εξελισσόμενων επιστημονικών δεδομένων, υιοθετώντας τεχνικές των Διασυνδεδεμένων Δεδομένων. Για το συγκεκριμένο πρόβλημα, προτείνουμε ένα RDF μοντέλο αλλαγών για να την περιγραφή των εξελισσόμενων δεδομένων. Βασισμένοι σε αυτό το μοντέλο, μετατρέπουμε παραδοσιακά βιολογικά δεδομένα σε εξελισσόμενα Διασυνδεδεμένα Δεδομένα. Η υποδομή διασυνδεδεμένων δεδομένων που αναπτύξαμε μπορεί να βοηθήσει τους βιολόγους να διερευνήσουν βιολογικές οντότητες καθώς και να μελετήσουν την εξέλιξή τους.

## 6.1 Σημασιολογική Ανάκτηση Πληροφορίας

Η επισημείωση εγγράφων έχει προσελκύσει, τα τελευταία χρόνια, την προσοχή των κοινοτήτων του Σημασιολογικού Ιστού [175] και των Ψηφιακών Βιβλιοθηκών [22]. Η σημασιολογική επισημείωση συνίσταται στον χαρακτηρισμό κειμένων με έννοιες (concepts), για παράδειγμα κλάσεις (classes) μίας οντολογίας (ontology), έτσι ώστε το περιεχόμενο να αποκτά σαφή και οργανωμένη σημασιολογία. Οι επισημειώσεις βοηθούν τους χρήστες να οργανώνουν καλύτερα τα έγγραφά τους, αλλά και βελτιώνουν τις δυνατότητες αναζήτησής τους: μέσω των επισημειώσεων, οι χρήστες μπορούν να αναζητούν πληροφορίες στα έγγραφά τους, όχι μόνο μέσω της αναζήτησης με λέξεις κλειδιά, αλλά και επιλέγοντας σαφώς ορισμένες έννοιες που περιγράφουν τα πεδία αναζήτησης τών χρηστών.

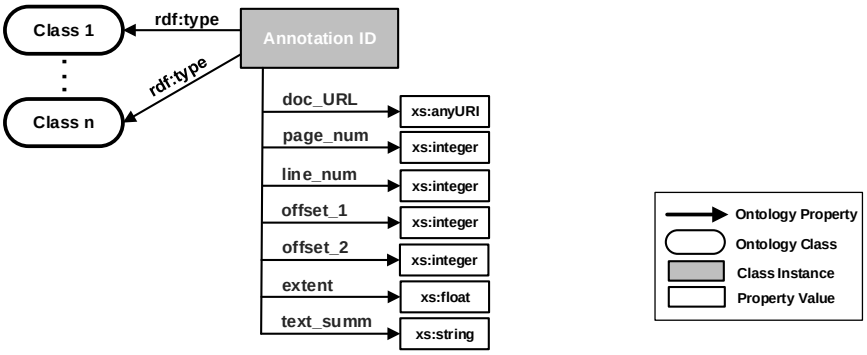
Αν και οι παραδοσιακές τεχνικές ανάκτησης πληροφορίας έχουν καθιερωθεί και χρησιμοποιούνται από πληθώρα εφαρμογών, αποδεικνύονται λιγότερο αποτελεσματικές σε σενάρια ασάφειας ή συνωνυμίας εννοιών. Από την άλλη πλευρά, αναζήτηση βασισμένη μόνο στα σημασιολογικά μεταδεδομένα των εγγράφων αναμένεται επίσης να μην είναι αποτελεσμάτική, αφού: (α) δε λαμβάνει υπόψη το ίδιο το κειμενικό περιεχόμενο, (β) σε πολλές περιπτώσεις τα σημασιολογικά μεταδεδομένα δεν είναι διαθέσιμα και (γ) σημασιολογικές επισημειώσεις μπορεί να καλύπτουν ένα μικρό μέρος του κειμένου των εγγράφων.

Υβριδικές μέθοδοι, οι οποίες συνδυάζουν αναζήτηση βασισμένη σε λέξεις κλειδιά και σημασιολογική αναζήτηση/περιήγηση σε έννοιες, μπορούν να ξεπεράσουν τα παραπάνω προβλήματα. Η ανάπτυξη μεθόδων και εργαλείων που ολοκληρώνουν σημασιολογική επισημείωση και αναζήτηση είναι ιδιαίτερα σημαντική. Για παράδειγμα, ερευνητές έχουν την ανάγκη να οργανώνουν, να κατηγοριοποιούν και να αναζητούν επιστημονικό υλικό (π.χ. δημοσιεύσεις) με αποτελεσματικό και αποδοτικό τρόπο. Παρόμοια, μία υπηρεσία αποδελτίωσης έχει την ανάγκη εντοπισμού ειδησεογραφικών άρθρων, επισημείωσης σημαντικών θεμάτων και αναζήτησης πληροφορίας σε αυτά.

Σε αυτό το κεφάλαιο παρουσιάζουμε το εργαλείου GoNTogle, ένα πλαίσιο σημασιολογικής επισημείωσης και ανάκτησης κειμένων, το οποίο συνδυάζει τεχνολογίες Σημασιολογικού Ιστού και κλασσικής Ανάκτησης Πληροφορίας. Το GoNTogle δίνει τη δυνατότητα χειροκίνητης και ημι-αυτόματης επισημείωσης βασιζόμενης σε έννοιες οντολογίας. Η επισημείωση βασίζεται σε καθιερωμένες τεχνολογίες Σημασιολογικού Ιστού, όπως η γλώσσα οντολογιών OWL -Web Ontology Language. Ταυτόχρονα, μία μέθοδος μηχανικής μάθησης (k-NN) η οποία εκμεταλλεύεται κειμενική πληροφορία και το ιστορικό επισημείωσης του χρήστη προτείνεται για την υποστήριξη του μηχανισμού αυτόματης επισημείωσης.

**Συνεισφορά.** Η κύρια συνεισφορά της εργασίας δίνεται παρακάτω.

1. Σχεδιάσαμε και υλοποιήσαμε ένα φιλικό προς το χρήστη πλαίσιο επισημείωσης εγγράφων που υποστηρίζει αρκετούς γνωστούς μορφότυπους κειμένων και προσφέρει προχωρημένες δυνατότητες αναζήτησης.
2. Το πλαίσιο ακολουθεί λογική βασισμένη σε εξυπηρετητή (server), όπου οι επισημειώσεις αποθηκεύονται σε ένα κεντρικό αποθετήριο, ξεχωριστά από το ίδιο το έγγραφο. Με αυτόν τον τρόπο, επιτυγχάνεται ένα συνεργατικό περιβάλλον, όπου κάθε χρήστης μπορεί να εκμεταλλευτεί τις επισημειώσεις των άλλων χρηστών, είτε για επισημείωση, είτε για αναζήτηση εγγράφων.




**Σχήμα 6.1:** Μοντέλο επισημείωσης

3. Προτείνουμε μία μέθοδο μηχανικής μάθησης για αυτοματοποιημένη επισημείωση κειμένων, με βάση μοντέλα εκπαιδευμένα πάνω στο ιστορικό επισημείωσης προηγούμενων κειμένων.
4. Εισάγουμε μία υβριδική μέθοδο αναζήτησης που συνδυάζει κλασσική αναζήτηση με λέξεις κλειδιά με σημασιολογική αναζήτηση.
5. Παρουσιάζουμε μία μελέτη χρηστών, η οποία καταδεικνύει την αποτελεσματικότητα της αυτοματοποιημένης επισημείωσης. Επίσης, δείχνουμε ότι η προτεινόμενη υβριδική αναζήτηση υπερισχύει των απλών, επιμέρους τρόπων αναζήτησης (με λέξεις κλειδιά και με περιήγηση στην οντολογία), τόσο σε ακρίβεια, όσο και σε ανάκληση

### 6.1.1 Σημασιολογική Επισημείωση

Το GoNTogle υποστηρίζει σημασιολογική επισημείωση για διάφορους, ευρέως χρησιμοποιούμενους μορφότυπους κειμένων (doc, pdf, txt, rtf, odt, sxw). Επιτρέπει την επισημείωση ολόκληρου του κειμένου ή κομματιών του, καθώς και χειροκίνητη ή ημιαυτόματη επισημείωση.

Το μοντέλο επισημείωσης είναι κοινό για όλους τους μορφότυπους κειμένων, παρόλο που διαφορετικοί μορφότυποι ενδέχεται να επιτρέπουν την εξαγωγή διαφορετικής μορφής μεταδεδομένων επισημείωσης. Οι επισημειώσεις αποθηκεύονται σε έναν κεντρικό εξυπηρετητή οντολογίας, ανεξάρτητα από τα έγγραφα. Οι επισημειώσεις από κάθε είδος εγγράφου ορίζονται και οργανώνονται/αποθηκεύονται ακριβώς με τον ίδιο τρόπο. Κάθε επισημείωση κωδικοποιείται ως στιγμιότυπο μίας κλάσης της οντολογίας, μαζί με την μεταπληροφορία της επισημείωσης για το έγγραφο. Αυτή η μεταπληροφορία οργανώνεται με τη βοήθεια ενός ελάχιστου συνόλου από ιδιότητες που ορίζουμε, έτσι ώστε η επισημείωση να μπορεί να «αναπαραχθεί» και να αναπαρασταθεί κάθε φορά που φορτώνεται, πάνω στο κείμενο του εγγράφου. Οι ιδιότητες αυτές περιέχουν πληροφορία όπως: document URL, annotation offsets, page number, extent of annotation over the document κλπ.

Η Εικόνα 6.1 δείχνει το μοντέλο επισημείωσης του συστήματος. Οι επισημειώσεις είναι οντότητες που μπορεί να ανήκουν σε μία ή περισσότερες κλάσεις. Μέσω των ιδιοτήτων που χαρακτηρίζουν τις οντότητες επισημείωσης, κωδικοποιούμε όλη την απαραίτητη μεταπληροφορία. Η ιδιότητα $doc\_URL$ αποθηκεύει το URL του εγγράφου. Οι ιδιότητες $page\_num$ και $line\_num$ περιέχουν τον αριθμό σελίδας και γραμμής μέσα στη σελίδα, στις οποίες ξεκινά η επισημείωση. Η $offset\_1$ αντιστοιχεί στη θέση της

αρχής της επισημείωσης από την αρχή του εγγράφου, ενώ η $offset\_2$ στη θέση του τέλους της επισημείωσης από την αρχή του εγγράφου. Η $extent$ αποθηκεύει το μέγεθος της επισημείωσης και η $text\_sum$ μία μικρή περίληψη του κειμένου της επισημείωσης, για λόγους παρουσίασης στη γραφική διεπιφάνεια.

#### 6.1.1.1 Αυτόματη Σημασιολογική Επισημείωση

Σε αυτή την ενότητα, παρουσιάζουμε το μοντέλο εκμάθησης που εφαρμόζουμε για αυτοματοποιημένη επισημείωση εγγράφων. Προτείνουμε μία μέθοδο βασισμένη στον αλγόριθμο kNN (k Nearest Neighbor) [260] που εκμεταλλεύεται το ιστορικό επισημείωσης των χρηστών για να προτείνει αυτόματα επισημειώσεις για νέα έγγραφα. Το σύνολο εκπαίδευσης του αλγορίθμου αποτελείται από προηγούμενες επισημειώσεις χρηστών. Συγκεκριμένα, όταν πραγματοποιείται μία επισημείωση, το αντίστοιχο κείμενο εξάγεται και ευρετηριοποιείται σε ένα ανεστραμμένο ευρετήριο (inverted index). Μαζί με την κειμενική πληροφορία, στο ευρετήριο κρατείται και πληροφορία σχετικά με τις κλάσεις επισημείωσης.

**Algorithm 8.** Annotation Suggestion Algorithm ($st$, $\mathbf{I}$)

**Input**: $st$: selected text; $\mathbf{I}$: index
**Output**: $cl_i$: suggested class; $Scr_{cli_i}$: suggested class score

1 **foreach** annotated text $at$ in $\mathbf{I}$ **do**
2  calculate $ts_{st,at}$
3 insert the $k$ most similar annotated texts in $S$
4 **foreach** $at$ in $S$ **do**
5  **foreach** class $cl$ annotate $at$ **do**
6  $Scr_{cl} = Scr_{cl} + (w_1 \cdot ts_{st,at}) \cdot (w_2 \cdot e_{cl,at})$
7 **return** $cl_i, Scr_{cl_i}$

Ο Αλγόριθμος 8 παρουσιάζει τη διαδικασία προτάσεων επισημείωσης. Είσοδος είναι το επιλεγμένο κείμενο $st$ και το ευρετήριο $\mathbf{I}$. Βασιζόμενοι στο σκορ ομοιότητας $ts_{st,at}$ μεταξύ του επιλεγμένου κειμένου $st$ και οποιουδήποτε ευρετηριασμένου κειμένου $at$ ήδη υπάρχουσας επισημείωσης, κρατάμε τις $k$ πιο όμοιες επισημειώσεις στο σύνολο $S$ (γραμμές 1-3). Στη συνέχεια, για κάθε επισημείωση του συνόλου $S$, εξάγουμε τις κλάσεις που της αντιστοιχούν. Σε κάθε κλάση $cl$ ανατίθεται ένα σκορ $Scr_{cl}$ που συνδυάζει: (α) την κειμενική ομοιότητα μεταξύ του κειμένου της επιλεγμένης και της ήδη υπάρχουσας επισημείωσης και (β) ένα σκορ $e_{cl,at}$ που αντιπροσωπεύει την έκταση κατά την οποία η κλάση $cl$ καλύπτει ένα επισημειωμένο κείμενο $at$ (γραμμή 6). Το $e_{cl,at}$ ορίζεται ως ο λόγος των όρων που είναι επισημειωμένοι με την κλάση $cl$, προς το συνολικό αριθμό όρων του επισημειωμένου κειμένου $at$.

$$e_{cl,at} = \frac{\text{number of tokens of } cl \text{ annotations over } at}{\text{number of tokens in } at}$$

Τα βάρη $w1$ και $w2$ χρησιμοποιούνται για να σταθμίσουν τη βαρύτητα των δύο σκορ. Τελικά, μία ταξινομημένη λίστα από προτεινόμενες κλάσεις $cl_i$ και τα αντίστοιχα σκορ τους $Scr_{cl_i}$ παρουσιάζεται στο χρήστη.

### 6.1.2 Αναζήτηση

Στη συνέχεια, παρουσιάζουμε τις δυνατότητες αναζήτησης του πλαισίου. Αρχικά, ορίζουμε τυπικά τους υποστηριζόμενους τύπους αναζήτησης και έπειτα αναλύουμε τις,

**Πίνακας 6.1:** Σύμβολα

| Symbol | Description |
|---|---|
| $q_{key}$ | Keyword query, consisting of search term $\{t_1, t_2, \ldots t_m\}$ |
| $S_{key}(q_{key})$ | Keyword-based search |
| $RS_{key}$ | Keyword-based search result set |
| $Scr_{key}(q_{key}, d)$ | Keyword-based similarity score |
| $q_{sem}$ | Semantic query, consisting of search classes $\{cl_1, cl_2, \ldots cl_n\}$ |
| $S_{sem}(q_{sem})$ | Semantic-based search |
| $RS_{sem}$ | Semantic-based search result set |
| $Scr_{sem}(q_{sem}, d)$ | Semantic-based similarity score |
| $S_{hybr}(q_{sem}, q_{key})$ | Hybrid search |
| $RS_{hybr}$ | Hybrid search result set |
| $Scr_{hybr}(q_{sem}, q_{key}, d)$ | Hybrid similarity score |

βασιζόμενες στην οντολογία, προχωρημένες λειτουργίες αναζήτησης. Ο Πίνακας 6.1 επεξηγεί τα σύμβολα που χρησιμοποιούμε.

#### 6.1.2.1 Τύποι Αναζήτησης

Κατηγοριοποιούμε τις βασικές δυνατότητες του πλαισίου σε τρεις τύπους: (α) Αναζήτηση με λέξεις κλειδιά, (β) Σημασιολογική αναζήτηση και (γ) Υβριδική αναζήτηση:

##### 6.1.2.1.1 Αναζήτηση με λέξεις κλειδιά

Ο χρήστης παρέχει τους όρους αναζήτησης και το σύστημα ανακτά σχετικά έγγραφα βασισμένο μόνο στην κειμενική ομοιότητα όρων αναζήτησης/κειμένων. Υιοθετούμε τη συνάρτηση βαθμολόγησης - ταξινόμησης αποτελεσμάτων της μηχανής αναζήτησης Lucene.

Έστω ένα ερώτημα $q_{key} = t_1, t_2, ...t_m$, όπου $t_i$ οι όροι (λέξεις κλειδιά) του ερωτήματος. Συμβολίζουμε τα αποτελέσματα της παραπάνω αναζήτησης ως ένα ταξινομημένο σύνολο $RS_{key}$ από πλειάδες $< d, Scr_{key}(q_{key}, d) >$, όπου $d$ όλα τα έγγραφα που ανακτήθηκαν, με σκορ $Scr_{key}(q_{key}, d)$.

##### 6.1.2.1.2 Σημασιολογική αναζήτηση

Ο χρήστης περιηγείται στις κλάσεις της οντολογίας και επικεντρώνει την αναζήτηση σε μία ή περισσότερες από αυτές. Ομοίως με την κλασσική αναζήτηση, ορίζουμε ένα σημασιολογικό ερώτημα ως $q_{sem} = cl_1, cl_2, ...cl_n$, όπου $cl_i$ οι κλάσεις αναζήτησης. Ορίζουμε τα αποτελέσματα της παραπάνω αναζήτησης ως ένα ταξινομημένο σύνολο $RS_{sem}$ από πλειάδες $< d, Scr_{sem}(q_{sem}, d) >$, όπου $d$ όλα τα έγγραφα που έχουν χαρακτηριστεί με τουλάχιστον μία από τις κλάσεις του ερωτήματος, με σημασιολογικό σκορ $Scr_{sem}(q_{sem}, d)$. Προκειμένου να ορίσουμε το σημασιολογικό σκορ, εξετάζουμε την έκταση της κάλυψης ενός εγγράφου από κάθε κλάση, δηλαδή το λόγο των όρων ενός εγγράφου που χαρακτηρίζονται με την κλάση, προς το συνολικό αριθμό όρων του εγγράφου.

Το τελικό σημασιολογικό σκορ ορίζεται ως:

$$Scr_{sem}(q_{sem}, d) = \sum_{i=1}^{n} \frac{ss_{cl_i,d}}{n},$$

$$ss_{cl_i,d} = \frac{\text{number of tokens of } cl_i \text{ annotations over } d}{\text{number of tokes in } d}$$

όπου $n$ οι κλάσεις που συμμετέχουν στο σημασιολογικό ερώτημα και $ss_{cli,d}$ ο βαθμός έκτασης κάθε κλάσης $cl_i$ σε ένα έγγραφο $d$.

#### 6.1.2.1.3 Υβριδική αναζήτηση

Ο χρήστης συνδυάζει αναζήτηση με λέξεις κλειδιά και περιήγηση/επιλογή κλάσεων της οντολογίας, έχοντας τη δυνατότητα να διαλέξει αν το τελικό αποτέλεσμα θα είναι η τομή ή η ένωση των αποτελεσμάτων. Και στις δύο περιπτώσεις, το τελικό σκορ είναι ένα σταθμισμένο άθροισμα των δύο επιμέρους σκορ.

$$Scr_{hybr}(q_{sem}, q_{key}, d) = w_{sem} \cdot Scr_{sem}(q_{sem}, d) + w_{key} \cdot Scr_{key}(q_{key}, d)$$

### 6.1.2.2 Προχωρημένες Δυνατότητες Αναζήτησης

Σε αυτήν την ενότητα παρουσιάζουμε λειτουργίες αναζήτησης που δύνανται να επιλεστούν, αφού έχει προηγηθεί μία αρχική αναζήτηση:

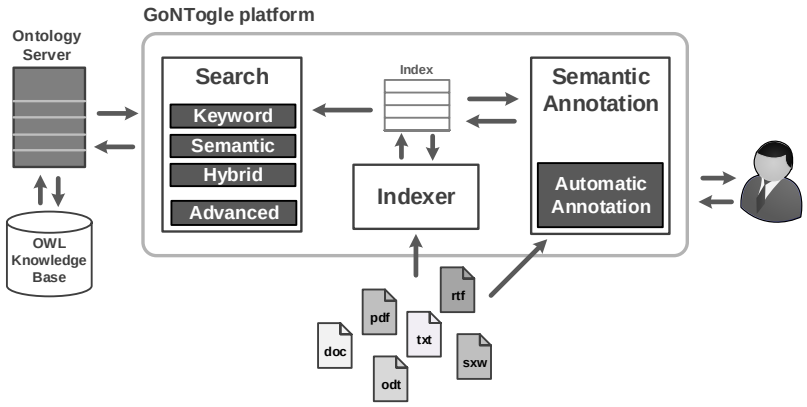
**Εύρεση σχετικών εγγράφων.** Ξεκινώντας από ένα έγγραφο-αποτέλεσμα $d$, ο χρήστης μπορεί να ψάξει όλα τα έγγραφα που έχουν επισημειωθεί με οποιαδήποτε από τις κλάσεις που χαρακτηρίζουν το έγγραφο.

**Εύρεση παρόμοιων εγγράφων.** Η συγκεκριμένη είναι παραλλαγή της προηγούμενης δυνατότητας. Ξεκινώντας από ένα έγγραφο-αποτέλεσμα $d$, ο χρήστης μπορεί να ψάξει όλα τα έγγραφα που έχουν επισημειωθεί με οποιαδήποτε από τις κλάσεις που χαρακτηρίζουν το έγγραφο και, παράλληλα, ανήκουν ήδη στα αποτελέσματα της προηγούμενης αναζήτησης.

**Εύρεση επόμενης «γενιάς» εγγράφων.** Η συγκεκριμένη δυνατότητα επιτρέπει τη μετάδοση της αναζήτησης σε χαμηλότερα επίπεδα της οντολογίας, αναζητώντας έγγραφα σε υποκλάσεις της αρχικής κλάσης αναζήτησης. Η λειτουργικότητα αυτή βρίσκει εφαρμογή, όταν η αρχική αναζήτηση είναι πολύ γενική.

**Εύρεση προηγούμενης «γενιάς» εγγράφων.** Είναι η αντίστροφη λειτουργικότητα της παραπάνω. Η συγκεκριμένη δυνατότητα επιτρέπει τη μετάδοση της αναζήτησης σε υψηλότερα επίπεδα της οντολογίας, αναζητώντας έγγραφα σε υπερκλάσεις της αρχικής κλάσης αναζήτησης. Η λειτουργικότητα αυτή βρίσκει εφαρμογή, όταν η αρχική αναζήτηση είναι πολύ ειδική.

**Αναζήτηση εγγύτητας.** Η συγκεκριμένη λειτουργία λαμβάνει υπόψη το επίπεδο των κλάσεων στην ιεραρχία της οντολογίας, μεταδίδοντας την αναζήτηση σε υποκλάσεις της αρχικής κλάσης.




**Σχήμα 6.2:** Αρχιτεκτονική συστήματος

### 6.1.3 Επισκόπηση Συστήματος

#### 6.1.3.1 Αρχιτεκτονική

Χάρη στην κεντρικοποιημένη αρχιτεκτονική, το GoNTogle εξασφαλίζει ένα συνεργατικό περιβάλλον στους χρήστες. Οι επισημειώσεις είναι ορατές και επαναχρησιμοποιήσιμες από διάφορες ομάδες χρηστών. Η Εικόνα 6.2 παρουσιάζει την αρχιτεκτονική του συστήματος, το οποίο χωρίζεται σε τέσσερα υποσυστήματα:

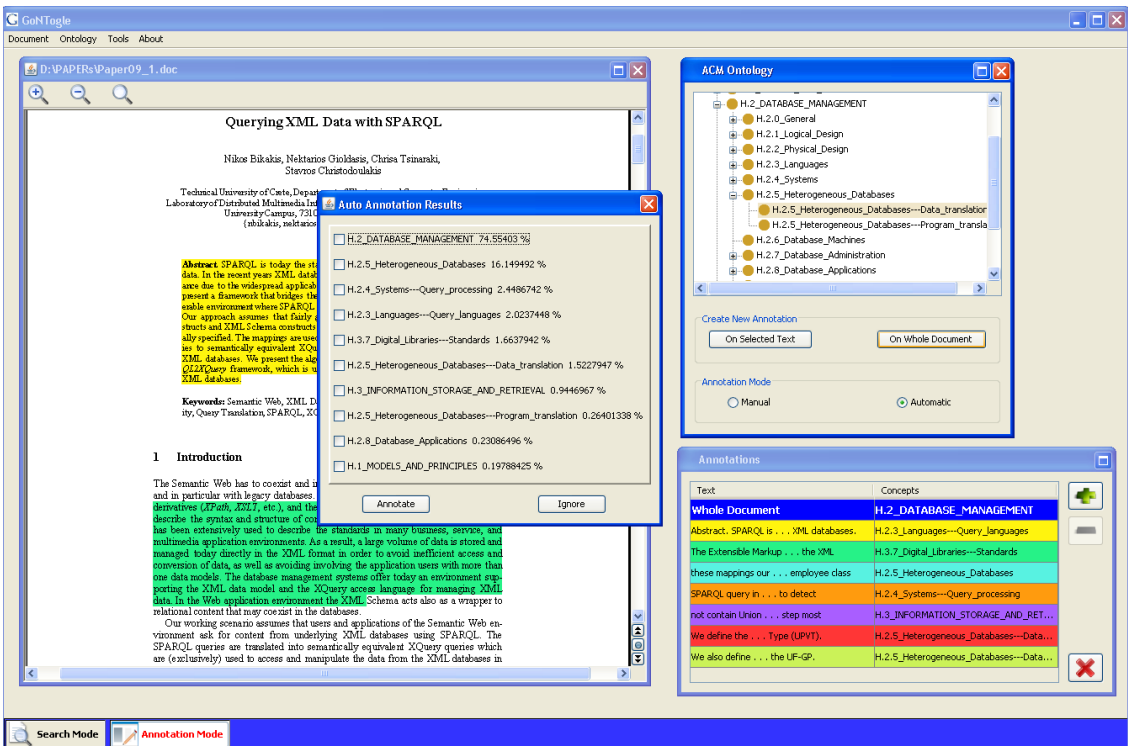
- Το Semantic Annotation Υποσύστημα, το οποίο αποτελείται από τις εξής μονάδες: (i) Document Viewer, (ii) Ontology Viewer και (iii) Annotation Editor.
- Το Ontology Server Υποσύστημα, το οποίο αποθηκεύει την οντολογία και τις επισημειώσεις. Αποτελείται από δύο μονάδες: (i) Ontology Manager και (ii) Ontology Knowledge Base.
- Το Indexing Υποσύστημα, υπεύθυνο για την ευρετηρίαση των εγγράφων.
- Το Search Υποσύστημα, με το οποίο επιτελείται αναζήτηση των εγγράφων.

#### 6.1.3.2 Λειτουργία

Η Εικόνα 6.3 παρουσιάζει μία οθόνη της εφαρμογής, που αφορά τη σημασιολογική επισημείωση εγγράφων. Ο χρήστης μπορεί να ανοίξει ένα έγγραφο στον Document Viewer. Επιπλέον, μπορεί να φορτώσει μία οντολογία μέσω του Ontology Viewer, επιλέγοντας μία ή περισσότερες κλάσεις για να χαρακτηρίσει όλο το έγγραφο ή την επιλεγμένη περιοχή κειμένου. Για κάθε επισημείωση εμφανίζεται στον Annotation Editor μία εγγραφή που αντιστοιχεί σε μία αποθηκευμένη στον Ontology Server επισημείωση. Ο χρήστης μπορεί να προσθαφαιρέσει κλάσεις σε μία επισημείωση ή να τη διαγράψει εντελώς. Με την επιλογή μίας επισημείωσης από τη λίστα του Annotation Editor, η εμφανιζόμενη περιοχή του εγγράφου μετακυλίεται στην αντίστοιχη επισημειωμένη περιοχή, η οποία επισημαίνεται με ειδικό χρωματισμό.

### 6.1.4 Πειραματική Ανάλυση

Σε αυτήν ην Ενότητα παρουσιάζεται η πειραματική αξιολόγηση (α) του μηχανισμού αυτόματης επισημείωσης εγγράφων και (β) της αποτελεσματικότητας της υβριδικής




**Σχήμα 6.3:** Γραφική διεπαφή συστήματος

αναζήτησης εγγράφων, συγκρινόμενης με τις επιμέρους: αναζήτηση με λέξεις κλειδιά και σημασιολογική αναζήτηση.

#### 6.1.4.1 Αυτόματη επισημείωση

Προκείμένου να αποτιμήσουμε την αποτελεσματικότητα του αλγορίθμου παραγωγής προτάσεων επισημείωσης, πραγματοποιούμε μία μελέτη χρηστών, κατά την οποία μετράμε την ακρίβεια σε κάθε θέση κατάταξης (P@n) και την ανάκληση (Recall).

##### 6.1.4.1.1 Διαμόρφωση πειράματος

Αρχικά μετατρέψαμε την ταξονομία της ACM[1] σε OWL οντολογία. Η οντολογία που προέκυψε είχε τέσσερα επίπεδα και 1463 κλάσεις.

Στη συνέχεια ζητήσαμε από 15 χρήστες να συμμετέχουν στην πειραματική μελέτη. Κάθε χρήστης επέλεξε δύο περιοχές ερευνητικού ενδιαφέροντος και, για κάθε περιοχή, επέλεξε δέκα δημοσιεύσεις τις οποίες είχε διαβάσει. Προκειμένου να εκπαιδεύσουμε το σύστημα, ζητήσαμε από κάθε χρήστη να επισημειώσει κομμάτια ή όλο το κείμενο για 12 από τις συνολικά 20 δημοσιεύσεις που του αναλογούσαν, με τουλάχιστον μία κλάση της χρησιμοποιούμενης οντολογίας. Αφού κάθε χρήστης εκτέλεσε τις επισημειώσεις, οι οποίες απετέλεσαν τη φάση εκπαίδευσης της πειραματικής διαδικασίας, ζητήσαμε από τον καθένα να αξιολογήσει τις αυτόματες προτάσεις κλάσεων του συστήματος, για κάθε μία από τις 8 εναπομείνασες δημοσιεύσεις. Πριν επισκοπήσει τις προτάσεις του συστήματος, ζητήθηκε από το χρήστη να σημειώσει κάποιες κλάσεις που περίμενε να του προτείνει το σύστημα, δηλαδή πολύ σχετικές κλάσεις με τη δημοσίευση, σύμφωνα με την άποψη του χρήστη. Στη συνέχεια, ο χρήστης σημείωσε ποιες από τις προτάσεις

[1] http://www.acm.org/about/class/2012

**Πίνακας 6.2:** Μέση Ακρίβεια στη θέση $n$ για κάθε χρήστη

| User | P@1 | P@2 | P@3 | P@4 | P@5 |
|---|---|---|---|---|---|
| **1** | 0.82 | 0.79 | 0.79 | 0.75 | 0.68 |
| **2** | 1.00 | 0.94 | 0.80 | 0.65 | 0.60 |
| **3** | 0.80 | 0.80 | 0.70 | 0.70 | 0.76 |
| **4** | 1.00 | 1.00 | 0.80 | 0.84 | 0.80 |
| **5** | 1.00 | 0.90 | 0.90 | 0.82 | 0.81 |
| **6** | 0.80 | 0.90 | 0.73 | 0.70 | 0.64 |
| **7** | 1.00 | 1.00 | 0.93 | 0.85 | 0.84 |
| **8** | 0.93 | 1.00 | 0.73 | 0.71 | 0.69 |
| **9** | 0.90 | 0.90 | 0.87 | 0.80 | 0.76 |
| **10** | 0.91 | 0.87 | 0.80 | 0.75 | 0.71 |
| **11** | 1.00 | 1.00 | 0.87 | 0.84 | 0.78 |
| **12** | 0.80 | 0.77 | 0.72 | 0.70 | 0.66 |
| **13** | 0.95 | 0.92 | 0.83 | 0.75 | 0.68 |
| **14** | 1.00 | 0.90 | 0.87 | 0.80 | 0.76 |
| **15** | 0.80 | 0.80 | 0.73 | 0.65 | 0.56 |
| **Avg** | 0.91 | 0.90 | 0.81 | 0.75 | 0.72 |

**Πίνακας 6.3:** Ανάκληση και τιμή $UVCS$ για κάθε χρήστη

| User | Recall | UVCS |
|---|---|---|
| **1** | 0.80 | 0.40 |
| **2** | 0.92 | 0.20 |
| **3** | 0.98 | 0.20 |
| **4** | 0.97 | 0.40 |
| **5** | 0.98 | 0.40 |
| **6** | 1.00 | 1.20 |
| **7** | 0.97 | 0.20 |
| **8** | 0.82 | 0.20 |
| **9** | 1.00 | 0.20 |
| **10** | 0.89 | 1.00 |
| **11** | 0.88 | 0.80 |
| **12** | 0.95 | 0.65 |
| **13** | 0.87 | 0.40 |
| **14** | 0.95 | 1.60 |
| **15** | 1.00 | 0 |
| **Avg** | 0.93 | 0.52 |

του συστήματος ήταν σωστές, ακόμα κι αν αυτός δεν είχε σκεφτεί από πριν τις αντίστοιχες κλάσεις. Στη συνέχεια, υπολογίσαμε τις τιμές ακρίβειας και ανάκλησης για κάθε χρήστη ξεχωριστά, καθώς και μέσες τιμές. Επιπλέον, με τη μετρική *απρόσμενων κλάσεων* ($UVCS$) μετράμε το μέσο αριθμό κλάσεων που δεν είχε προβλέψει ο χρήστης, αλλά προτάθηκαν σωστά από το σύστημα.

#### 6.1.4.1.2 Αποτελέσματα

Ο Πίνακας 6.2 παρουσιάζει, για κάθε χρήστη, τις μέσες τιμές ακρίβειας στη θέση $n$ (P@n) πάνω στις 8 αποτιμημένες δημοσιεύσεις. Ο Πίνακας 6.3 παρουσιάζει τις αντίστοιχες τιμές ανάκλησης και απρόσμενων κλάσεων. Περιορίζουμε την ανάλυσή μας στις 5 πρώτες θέσεις, αφού στο συγκεκριμένο σενάριο δεν περιμένουμε μία δημοσίευση να αφορά πολύ περισσότερες θεματικές περιοχές.

Παρατηρούμε ότι η μέθοδός μας επιτυγχάνει πολύ υψηλές τιμές ακρίβειας και ανάκλησης, με μέση ανάκληση 93%. Οι χαμηλές τιμές που παρατηρούνται για την ακρίβεια στις θέσεις 4 και 5 (P@4, P@5) δικαιολογούνται από το γεγονός ότι, σε αρκετές περιπτώσεις, δεν υπήρχαν περισσότερες από 3 κλάσεις που να χαρακτηρίζουν μία δημοσίευση. Τέλος, φαίνεται από τη μετρική $UVCS$ ότι το σύστημα καθοδηγεί τους χρήστες στην εύρεση νέων κλάσεων επισημείωσης, τι οποίες δεν είχαν προηγουμένως σκεφτεί.

### 6.1.4.2 Αναζήτηση

Σε αυτήν την ενότητα αξιολογούμε την αποτελεσματικότητα των τύπων αναζήτησης που προσφέρονται από το πλαίσιο του GoNTogle. Η σύγκριση γίνεται με τη βοήθεια των μετρικών της ακρίβειας στη θέση n (Precision@n), ανάκλησης (Recall) και F-

**Πίνακας 6.4:** Ερωτήματα λέξεων κλειδιών

| ID | Keywords |
|---|---|
| $q_{key}1$ | knowledge discovery and privacy |
| $q_{key}2$ | stream mining |
| $q_{key}3$ | RDF indexing |
| $q_{key}4$ | spatial databases |
| $q_{key}5$ | clustering |
| $q_{key}6$ | spatial access |
| $q_{key}7$ | query language |
| $q_{key}8$ | data model |
| $q_{key}9$ | XML interoperability |
| $q_{key}10$ | information integration |

**Πίνακας 6.5:** Σημασιολογικά ερωτήματα

| ID | Classes |
|---|---|
| $q_{sem}1$ | K.4.1 [Public Policy Issues]: Privacy |
| $q_{sem}2$ | H.2.8 [Database Applications]: Data mining |
| $q_{sem}3$ | H.3.1 [Content Analysis and Indexing]: Indexing methods |
| $q_{sem}4$ | H.2.8 [Database Applications]: Spatial databases and GIS |
| $q_{sem}5$ | H.3.3 [Information Search and Retrieval]: Clustering |
| $q_{sem}6$ | H.2.2 [Physical Design]: Access Methods |
| $q_{sem}7$ | H.2.3 [Languages]: Query languages |
| $q_{sem}8$ | H.2.1 [Logical Design]: Data models |
| $q_{sem}9$ | D.2.12 [Interoperability] |
| $q_{sem}10$ | H.2.5 [Heterogeneous Databases] |

measure. Σε κάθε περίπτωση, η προτεινόμενη υβριδική μέθοδος αναζήτησης επιτυγχάνει καλύτερα αποτελέσματα από τις επιμέρους μεθόδους αναζήτησης με λέξεις κλειδιά και σημασιολογικής αναζήτησης.

#### 6.1.4.2.1 Διαμόρφωση πειράματος

Τα βάρη που χρησιμοποιούνται για την υβριδική αναζήτηση παίρνουν τις ακόλουθες τιμές: $w_{sem} = 0.7, w_{key} = 0.3$ για τον τελεστή $AND$ και $w_{sem} = 0.6, w_{key} = 0.4$ για τον τελεστή $OR$ μετά από δοκιμαστική βελτιστοποίησή τους. Διαισθητικά, αυτές οι τιμές καταδεικνύουν ότι, στο σενάριό μας, η σημασιολογική αναζήτηση είναι ελαφρά πιο σημαντική από την αναζήτηση με λέξεις κλειδιά.

Το πειραματικό σύνολο δεδομένων αποτελείται από 300 χειροκίνητα και αυτόματα επισημειωμένα ερευνητικά κείμενα από το προηγούμενο πείραμα. Πρώτα, δημιουργήσαμε μία δεξαμενή από λέξεις κλειδιά που χαρακτηρίζουν/απαντώνται στα κείμενα και επιλέξαμε τυχαία 10 από αυτές, ώστε να τις θεωρήσουμε ως ερωτήματα. Σημειώνουμε ότι τα ερωτήματα μπορεί να περιέχουν περισσότερες από μία λέξεις. Επιπλέον, αντιστοιχίζουμε τα ερωτήματα λέξεων κλειδιών σε σημασιολογικά ερωτήματα, επιλέγοντας τις κατάλληλες (παρόμοιες) κλάσεις της οντολογίας. Με αυτόν τον τρόπο, μπορο-

**Πίνακας 6.6:** Μέσες τιμές των μετρικών Precision@n, Recall, F-measure για όλα τα ερωτήματα, για τέσσερις διαφορετικές εκδοχές του κάθε ερωτήματος

| | **P@1** | **P@2** | **P@3** | **P@4** | **P@5** | **P@6** | **P@7** | **P@8** | **P@9** | **P@10** | **Recall** | **F-measure** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $q_{key}$ | 0.73 | 0.73 | 0.70 | 0.70 | 0.60 | 0.53 | 0.49 | 0.49 | 0.46 | 0.45 | 0.55 | 0.50 |
| $q_{sem}$ | 1.00 | 0.95 | 0.91 | 0.89 | 0.91 | 0.88 | 0.83 | 0.78 | 0.74 | 0.68 | 0.84 | 0.75 |
| $q_{hybrA}$ | 1.00 | 1.00 | 1.00 | 1.00 | 0.98 | - | - | - | - | - | 0.66 | 0.79 |
| $q_{hybrO}$ | 1.00 | 1.00 | 0.97 | 0.98 | 0.96 | 0.95 | 0.95 | 0.90 | 0.83 | 0.76 | 0.98 | 0.86 |

ύμε να εφαρμόσουμε τους δύο τύπους αναζήτησης για τα ίδια ερωτήματα. Επιπλέον, μπορούμε να συνδυάσουμε τους δύο τύπους, σταθμίζοντας τα επιμέρους σκορ που προκύπτουν από τον κάθε τύπο, παράγοντας έτσι αποτελέσματα υβριδικής αναζήτησης. Ο συνδυασμός μπορεί να πραγματοποιηθεί τόσο με τελεστή AND (τομή των επιμέρους αποτελεσμάτων) όσο και με τελεστή OR (ένωση των επιμέρους αποτελεσμάτων). Οι Πίνακες 6.4 και 6.5 παρουσιάζουν τα ερωτήματα λέξεων κλειδιών και τα αντίστοιχα σημασιολογικά ερωτήματα.

Για κάθε ερώτημα μετράμε την ποιότητα της ανάκτησης χρησιμοποιώντας την ακρίβεια στη θέση n και την ανάκληση (Precision@n, Recall). Όπως αναφέραμε παραπάνω, αξιολογούμε τέσσερις τύπους αναζήτησης: (α) αναζήτηση με λέξεις κλειδιά (qkey), (β) σημασιολογική αναζήτηση (qsem), (γ) υβριδική με τελεστή AND (hybrA) και (δ) υβριδική με τελεστή OR (hybrO). Τέλος, για κάθε τύπο αναζήτησης εξετάζουμε συνολικά τέσσερις μετρικές: Precision@n, Recall, F-measure, Precision-Recall curve.

#### 6.1.4.2.2 Αποτελέσματα για όλα τα ερωτήματα

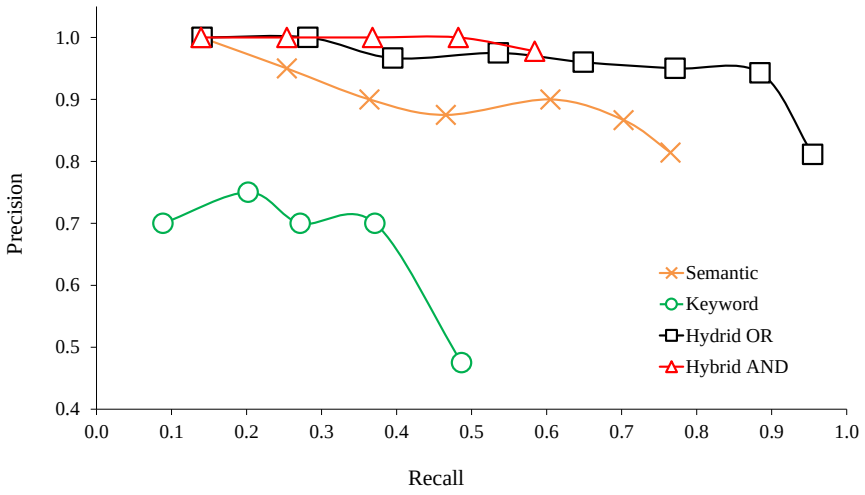
Ο Πίνακας 6.6 παρουσιάζει τις μέσες τιμές των μετρικών Precision@n, Recall, F-measure πάνω σε όλα τα ερωτήματα. Σημειώνουμε ότι τα περισσότερα ερωτήματα στην υβριδική αναζήτηση με τελεστή AND δεν επιστρέφουν πάνω από $5-6$ αποτελέσματα (όπως μπορεί να φανεί και από τον Πίνακα 6.7). Κατά συνέπεια, η ακρίβεια για αυτόν τον τύπο αναζήτησης υπολογίζεται μόνο για τις θέσεις 1 έως 5, για όλα τα ερωτήματα.

**Ακρίβεια.** Όπως παρατηρούμε στον Πίνακα 6.6, η υβριδική αναζήτηση (και για τους δύο τελεστές) αποδίδει καλύτερα από τις επιμέρους αναζητήσεις, σε κάθε θέση κατάταξης, με τον τύπο qhybrA να επιτυγχάνει ελαφρά καλύτερες τιμές για τις θέσεις 4 και 5. Επιπλέον, παρατηρούμε ότι η ακρίβεια της αναζήτησης με λέξεις κλειδιά μειώνεται δραστικά μετά τη θέση 4, ενώ η ακρίβεια των υπολοίπων τύπων αρχίζει να μειώνεται μετά από τη θέση 6. Η υβριδική αναζήτηση, συγκρινόμενη με την αναζήτηση με λέξεις κλειδιά, επιτυγχάνει μέγιστη αύξηση κατά 100% στη θέση 7 και ελάχιστη αύξηση 33.3% στη θέση 2. Συγκρινόμενη με τη σημασιολογική αναζήτηση, επιτυγχάνει μέγιστη αύξηση κατά 17.2% στη θέση 10 και ελάχιστη αύξηση 0% στη θέση 1.

**Ανάκληση.** Όπως μπορούμε να δούμε, ο τύπος qhybrO αποδίδει καλύτερα από τις επιμέρους μεθόδους, επιτυγχάνοντας ανάκληση κοντά στο 100% (98%). Ο qhybrA αποδίδει ανάμεσα στην αναζήτηση με λέξεις κλειδιά και στη σημασιολογική. Αυτό οφείλεται στο ότι ο συγκεκριμένος τύπος είναι πολύ περιοριστικός, επιστρέφοντας αρκετά λιγότερα έγγραφα από τους υπόλοιπους τύπους, κάτι το οποίο επηρεάζει αρνητικά την ανάκληση.

**Φ-μεασυρε.** Και σε αυτή τη μετρική, η υβριδική αναζήτηση ξεπερνά τους επιμέρους τύπους αναζήτησης. Συγκρίνοντας την υβριδική με τελεστή OR με αναζήτηση με λέξεις κλειδιά και με σημασιολογική, επιτυγχάνει αύξηση 77% και 16.4% αντίστοιχα. Συγκρίνοντας την υβριδική με τελεστή AND με αναζήτηση με λέξεις κλειδιά και με σημασιολογική, επιτυγχάνει αύξηση 52% και 0% αντίστοιχα.

**Καμπύλη ακρίβειας - ανάκλησης.** Η Εικόνα 6.4 παρουσιάζει τις καμπύλες ακρίβειας - ανάκλησης στο σύνολο των ερωτημάτων. Βλέπουμε ότι και οι δύο τύποι υβριδικής αναζήτησης έχουν πολύ σταθερή συμπεριφορά, επιτυγχάνοντας υψηλή ακρίβεια (κοντά στο 100%) ακόμα και για τιμές ανάκλησης μεγαλύτερες του 80%. Η υβριδική qhybrA παρουσιάζεται ελαφρά καλύτερη της qhybrO για τιμές ανάκλησης μικρότερες του 60%. Οι επιμέρους τύποι αναζήτησης συμπεριφέρονται χειρότερα, με αισθητά χειρότερη συμπεριφορά της καμπύλης αναζήτησης με λέξεις κλειδιά.




**Σχήμα 6.4:** Καμπύλη ακρίβειας -ανάκλησης για το σύνολο των ερωτημάτων

#### 6.1.4.2.3 Αποτελέσματα για κάθε ερώτημα

Ο Πίνακας 6.7 παρουσιάζει, για κάθε ερώτημα τις τιμές ακρίβειας στη θέση n και ανάκλησης. Οι επιμέρους αυτές τιμές επιβεβαιώνουν τις παραπάνω παρατηρήσεις μας: οι δύο υβριδικοί τύποι αναζήτησης ξεπερνούν σε αποτελεσματικότητα τους επιμέρους τύπους αναζήτησης, επιτυγχάνοντας, σε αρκετές περιπτώσεις, πολύ υψηλές τιμές ακρίβειας και ανάκλησης ταυτόχρονα, ενώ η πιο φτωχή συμπεριφορά παρουσιάζεται από την αναζήτηση με λέξεις κλειδιά.

### 6.1.5 Σχετικές Εργασίες

Ένας μεγάλος αριθμός προσεγγίσεων όσον αφορά τη σημασιολογική επισημείωση έχει προταθεί στη βιβλιογραφία [283, 333]. Οι περισσότερες επικεντρώνονται στην επισημείωση διαδικτυακών πόρων (web resources), όπως HTML ιστοσελίδες [215, 184, 112, 131, 12, 176, 336]. Στην επισημείωση απλού κειμένου, υπάρχουν προσεγγίσεις που διαφέρουν όσον αφορά τόσο στην επισημείωση, όσο και στην αναζήτηση κειμένου. Το GATE [120] είναι μία πλατφόρμα που ενσωματώνει συγκεκριμένη αρχιτεκτονική,

**Πίνακας 6.7:** Τιμές των μετρικών Precision@n, Recall για κάθε ερώτημα

| | Query 1 | | | | Query 2 | | | | Query 3 | | | | Query 4 | | | | Query 5 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ |
| **P@1** | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| **P@2** | 1.00 | 1.00 | 1.00 | 1.00 | 0.50 | 0.50 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| **P@3** | 1.00 | 1.00 | 1.00 | 1.00 | 0.33 | 0.67 | 1.00 | 0.67 | 0.67 | 0.67 | 1.00 | 1.00 | 1.00 | 0.67 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| **P@4** | 1.00 | 1.00 | 1.00 | 1.00 | 0.50 | 0.50 | - | 0.75 | 0.75 | 0.75 | 1.00 | 1.00 | 1.00 | 0.75 | 1.00 | 1.00 | 1.00 | 0.75 | 1.00 | 1.00 |
| **P@5** | 1.00 | 1.00 | 1.00 | 1.00 | 0.40 | 0.60 | - | 0.60 | 0.60 | 0.80 | 1.00 | 1.00 | 1.00 | 0.80 | 1.00 | 1.00 | 0.80 | 0.80 | 1.00 | 1.00 |
| **P@6** | 0.83 | 1.00 | - | 1.00 | 0.33 | 0.67 | - | 0.67 | 0.50 | 0.67 | - | 0.83 | 0.83 | 0.83 | 1.00 | 1.00 | 0.67 | 0.83 | - | 1.00 |
| **P@7** | 0.71 | 1.00 | - | 1.00 | 0.29 | 0.57 | - | 0.57 | 0.43 | 0.71 | - | 0.86 | 0.71 | 0.71 | - | 1.00 | 0.57 | 0.71 | - | 1.00 |
| **P@8** | 0.63 | 1.00 | - | 1.00 | 0.25 | 0.50 | - | 0.50 | 0.38 | 0.63 | - | 0.75 | 0.75 | 0.75 | - | 1.00 | 0.63 | 0.63 | - | 0.88 |
| **P@9** | 0.56 | 1.00 | - | 1.00 | 0.22 | 0.44 | - | 0.44 | 0.33 | 0.56 | - | 0.67 | 0.67 | 0.78 | - | 0.89 | 0.56 | 0.56 | - | 0.78 |
| **P@10** | 0.50 | 0.90 | - | 1.00 | 0.20 | 0.40 | - | 0.40 | 0.40 | 0.50 | - | 0.60 | 0.60 | 0.80 | - | 0.80 | 0.50 | 0.50 | - | 0.70 |
| **Recall** | 0.45 | 0.82 | 0.45 | 0.91 | 0.50 | 1.00 | 0.25 | 1.00 | 0.67 | 0.83 | 0.83 | 1.00 | 0.75 | 1.00 | 0.75 | 1.00 | 0.63 | 0.63 | 0.63 | 0.88 |

| | Query 6 | | | | Query 7 | | | | Query 8 | | | | Query 9 | | | | Query 10 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ | $q_{key}$ | $q_{sem}$ | $q_{hybrA}$ | $q_{hybrO}$ |
| **P@1** | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| **P@2** | 1.00 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |
| **P@3** | 1.00 | 1.00 | 1.00 | 1.00 | 0.33 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 | 0.67 | 1.00 | 1.00 | 1.00 |
| **P@4** | 1.00 | 1.00 | 1.00 | 1.00 | 0.25 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 0.75 | 1.00 | 1.00 | 1.00 | 0.75 | 1.00 | 1.00 | 1.00 |
| **P@5** | 0.80 | 1.00 | 0.80 | 1.00 | 0.20 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 0.60 | 1.00 | 1.00 | 1.00 | 0.60 | 1.00 | 1.00 | 1.00 |
| **P@6** | 0.67 | 0.83 | 0.67 | 1.00 | 0.33 | 1.00 | 1.00 | 1.00 | 0 | 1.00 | 1.00 | 1.00 | 0.50 | 1.00 | - | 1.00 | 0.50 | 0.83 | 1.00 | 1.00 |
| **P@7** | 0.71 | 0.71 | 0.57 | 1.00 | 0.29 | 1.00 | 1.00 | 1.00 | 0.14 | 1.00 | - | 1.00 | 0.43 | 0.86 | - | 1.00 | 0.43 | 0.86 | - | 1.00 |
| **P@8** | 0.75 | 0.63 | 0.50 | 1.00 | 0.38 | 1.00 | 1.00 | 1.00 | 0.13 | 0.88 | - | 0.88 | 0.38 | 0.75 | - | 0.88 | 0.50 | 0.88 | - | 1.00 |
| **P@9** | 0.67 | 0.56 | 0.44 | 0.89 | 0.44 | 0.89 | 1.00 | 1.00 | 0.11 | 0.78 | - | 0.78 | 0.33 | 0.67 | - | 0.78 | 0.56 | 0.78 | - | 0.89 |
| **P@10** | 0.60 | 0.50 | 0.40 | 0.80 | 0.50 | 0.80 | 0.90 | 1.00 | 0.10 | 0.70 | - | 0.70 | 0.30 | 0.60 | - | 0.70 | 0.60 | 0.70 | - | 0.80 |
| **Recall** | 0.67 | 0.63 | 0.50 | 1.00 | 0.50 | 0.80 | 0.90 | 1.00 | 0.14 | 0.70 | 0.89 | 1.00 | 0.43 | 0.86 | 0.71 | 1.00 | 0.75 | 0.88 | 0.75 | 1.00 |

πλαίσιο και γραφικό εργαλείο για γλωσσική επεξεργασία. Ταυτόχρονα προσφέρει εργαλεία και πόρους για κειμενική επισημείωση, τόσο χειροκίνητη όσο και αυτόματη, χρησιμοποιώντας τεχνολογίες εξαγωγής πληροφορίας (Information Extraction - IE).

Η δουλειά που περιγράφεται στο [214] προσφέρει μία υποδομή για σημασιολογική επισημείωση κειμένων, περιοριζόμενη, όμως, από τη δική της οντολογία, ονομαζόμενη KIMO. Τα υποσυστήματα για εξαγωγή πληροφορίας, διαχείριση εγγράφων και επισημείωση βασίζονται στο GATE. Σκοπός του υποσυστήματος εξαγωγής πληροφορίας είναι η αναγνώριση ονοματικών οντοτήτων που να σχετίζονται με τις έννοιες της οντολογίας KIMO. Συγκρινόμενη με τις παραπάνω προσεγγίσεις, η δουλειά που περιγράφεται στην παρούσα διατριβή υλοποιεί προχωρημένες δυνατότητες αναζήτησης, συνδυάζοντας κλασσική αναζήτηση με λέξεις κλειδιά και σημασιολογική αναζήτηση με περιήγηση στις κλάσεις της οντολογίας. Επιπλέον, εισάγει δυνατότητες αυτόματης επισημείωσης κειμένου, στηριζόμενες σε μοντέλα μηχανικής μάθησης που εκπαιδεύονται με βάση το ιστορικό επισημείωσης των χρηστών.

Η δουλειά του [95] (AKTiveMedia) υποστηρίζει επισημείωση κειμένου, εικόνων και ιστοσελίδων, χρησιμοποιώντας τόσο οντολογίες, όσο και επισημειώσεις ελεύθερου κειμένου (tags). Για την υποστήριξη αυτόματης επισημείωσης χρησιμοποιείται ένα υποκείμενο σύστημα εξαγωγής πληροφορίας, το οποίο εκπαιδεύεται από προηγούμενες επισημειώσεις για να προτείνει επισημειώσεις στο χρήστη. Παρόλα αυτά, η συγκεκριμένη δουλειά δεν υποστηρίζει δυνατότητες αναζήτησης, ενώ ο μηχανισμός αυτόματης επισημείωσης έχει διάφορους περιορισμούς: έχει χαμηλή απόδοση, όταν η επισημείωση επεκτείνεται πέραν του ενός όρου, ενώ δεν υποστηρίζεται επισημείωση με περισσότερες της μίας έννοιες.

Τα παραπάνω εργαλεία περιορίζονται σε επισημείωση απλού κειμένου ή HTML κειμένου. Όσον αφορά άλλους μορφότυπους κειμένου, το PDFTab [145] είναι ένα plug-in του Protege[2] που υποστηρίζει επισημείωση pdf εγγράφων με κλάσεις από οντολογίες OWL. Οι επισημειώσεις αποθηκεύονται εσωτερικά στο κάθε έγγραφο. Ομοίως, το SemanticWord [315] είναι ένα plug-in του MS Word που υποστηρίζει επισημείωση word εγγράφων με κλάσεις από οντολογίες DAML+OIL. Συγκριτικά με τη δική μας προσέγγιση, τα δύο παραπάνω εργαλεία δεν υποστηρίζουν αναζήτηση ή αυτόματη επισημείωση.

Όσον αφορά στη σημασιολογική αναζήτηση, τα τελευταία χρόνια έχουν, επίσης, προταθεί αρκετές μέθοδοι στη βιβλιογραφία [247]. Μία προσέγγιση αρκετά κοντινή στη δική μας παρουσιάζεται στο [58], όπου ένας συνδυασμός αναζήτησης με λέξεις κλειδιά και σημασιολογικής αναζήτησης σε διαδικτυακούς πόρους υλοποιείται πάνω στο AKTiveMedia [95]. Ένα σημαντικό μειονέκτημα αυτής της μεθόδου είναι ότι η ταξινόμηση αποτελεσμάτων στηρίζεται μόνο στην αναζήτηση με λέξεις κλειδιά, ενώ η σημασιολογική αναζήτηση χρησιμοποιείται μόνο για να φιλτραριστούν αποτελέσματα. Επιπλέον, [58] δεν υποστηρίζονται προχωρημένες επιλογές σημασιολογικής αναζήτησης βασισμένες στη σημασιολογία των οντολογιών. Τέλος, μία ενδιαφέρουσα αλλά λιγότερο σχετική προσέγγιση παρουσιάζεται στο [165], όπου γίνεται ανάλυση της σημασίας των λέξεων και φράσεων, ώστε να οριστούν σημασιολογικές σχέσεις μεταξύ εννοιών. Έτσι, η αναζήτηση επεκτείνεται με σημασιολογία, μετατρέποντας τις απλές λέξεις σε έννοιες, ώστε να γίνει δυνατή η εκμετάλλευση των μεταξύ τους σημασιολογιών.

[2]http://protege.stanford.edu/

### 6.1.6 Επίλογος

Σε αυτήν την ενότητα, παρουσιάσαμε το GoNTogle, ένα πλαίσιο σημασιολογικής επισημείωσης και ανάκτησης εγγράφων, το οποίο συνδυάζει τεχνολογίες Ανάκτησης Πληροφορίας και Σημασιολογικού ιστού. Το GoNTogle υποστηρίζει χειροκίνητη και αυτόματη επισημείωση εγγράφων, χρησιμοποιώντας οντολογίες εννοιών. Η λειτουργικότητα αυτόματης επισημείωσης βασίζεται σε έναν υποκείμενο μηχανισμό εκμάθησης, που εκμεταλλεύεται κειμενική πληροφορία και πληροφορία επισημειώσεων (κλάσεις της οντολογίας).

Προκειμένου να ξεπεράσουμε τα μειονεκτήματα της κλασικής αναζήτησης με λέξεις κλειδιά (πολυσημία νοημάτων, συνωνυμία) και της σημασιολογικής αναζήτησης (ανυπαρξία επισημειώσεων με κλάσεις της οντολογίας), προτείνουμε μία υβριδική μέθοδο αναζήτησης η οποία συνδυάζει τις δύο παραπάνω επιμέρους κατηγορίες αναζήτησης. Επιπλέον, ορίζουμε προχωρημένες λειτουργίες σημασιολογικής αναζήτησης, που διευκολύνουν το χρήστη, όταν θέλει να επεκτείνει την αναζήτηση σε ήδη ανακτημένα αποτελέσματα.

Επίσης, πραγματοποιήσαμε μία μελέτη χρηστών η οποία καταδεικνύει την αποτελεσματικότητα της μεθόδου αυτόματης επισημείωσης εγγράφων, καθώς και των μεθόδων υβριδικής αναζήτησης. Επόμενα βήματα βελτίωσης της συγκεκριμένης δουλειάς θα μπορούσαν να περιλαμβάνουν τα εξής: (α) Εκμετάλλευση της διαδικασίας συλλογιστικής σε οντολογίες για τη βελτίωση της αναζήτησης, (β) Ενσωμάτωση σημασιολογικών προσεγγίσεων επεξεργασίας φυσικής γλώσσας, (γ) Ενσωμάτωση υποστήριξης ελεύθερων επισημειώσεων (tagging) ταυτόχρονα με δομημένες σε οντολογίες επισημειώσεις και (δ) βελτίωση της λειτουργικότητας προβολής εγγράφων και επισημείωσης του υλοποιημένου εργαλείου.

## 6.2 Δημοσιοποίηση και Διερεύνηση Εξελισσόμενων Διασυνδεδεμένων Δεδομένων

Τα Διασυνδεδεμένα Δεδομένα (Linked Data) είναι μια ευρέως διαδεδομένη τεχνική για τη δημοσίευση και διάδοση των δεδομένων που διευκολύνει ιδιαίτερα στην επαναχρησιμοποίηση από πιθανούς καταναλωτές, όπως άτομα, κοινότητες ακόμα και εφαρμογές. Για να μπορεί αυτό να επιτευχθεί, το παράδειγμα των διασυνδεδεμένων δεδομένων περιλαμβάνει μια σειρά από γενικές πρακτικές δημοσίευσης, διάδοσης και σύνδεσης. Οι πρακτικές αυτές όμως δεν εξασφαλίζουν τις απαραίτητες προϋποθέσεις για επαναχρησιμοποίηση των δεδομένων όταν τα δεδομένα εξελίσσονται. Στην περίπτωση αυτή προκύπτουν νέες προκλήσεις που σχετίζονται με την δυναμικότητα των δεδομένων και οι οποίες πρέπει να αντιμετωπιστούν.

Χαρακτηριστικό παράδειγμα δεδομένων που αλλάζουν είναι τα επιστημονικά δεδομένα. Για να εξασφαλίσουμε την επαναχρησιμοποίηση των δεδομένων, αλλά και για να επιτρέψουμε την περαιτέρω εκμετάλλευση και επαλήθευση των επιστημονικών αποτελεσμάτων, οι χρήστες των δεδομένων (είτε άνθρωποι, είτε υπηρεσίες) θα πρέπει να μπορούν: (1) να έχουν πρόσβαση όχι μόνο στις τελευταίες εκδόσεις των διασυνδεδεμένων δεδομένων, αλλά και σε όλες τις προηγούμενες και (2) να έχουν πρόσβαση και στις αλλαγές που συντελέστηκαν ανάμεσα στις διαφορετικές εκδόσεις, καθώς και στις αιτίες και στα αποτελέσματα των αλλαγών αυτών.

Για να μπορέσουμε να υποστηρίξουμε τις παραπάνω απαιτήσεις, σε αυτή την εργασία

υιοθετούμε οντότητες RDF που εμπεριέχουν πληροφορίες σχετικά με την έκδοσή τους και ιδιότητες που δηλώνουν τις εκδόσεις, καθώς επίσης μοντελοποιούμε τις αλλαγές ως RDF πόρους στα διασυνδεδεμένα δεδομένα. Με βάση το προτεινόμενο μοντέλο, οι υπηρεσίες επερωτήσεων (π.χ. SPARQL) και πλοήγησης (δηλαδή η πλοήγηση μέσω URIs που μοντελοποιούν τις διαφορετικές εκδόσεις των δεδομένων) μπορούν να αναπτυχθούν πάνω στα διαχρονικά σύνολα ΔΔ.

### 6.2.1 Υπόβαθρο

Τα τελευταία χρόνια, το Ερευνητικό Κέντρο «Αθηνά» σε συνεργασία με τον Πανεπιστήμιο Θεσσαλίας (και παλαιότερα με το ΕΚΕΒΕ «Α. Φλέμινγκ») έχει αναπτύξει τις υποδομές αποθήκευσης και ανάλυσης δεδομένων επιστημών ζωής DIANA οι οποίες παρέχουν εξειδικευμένες διαδικτυακές εφαρμογές με σκοπό την υποβοήθηση της έρευνας πάνω στα βιολογικά μόρια microRNA. Τα microRNA είναι μικρά μόρια RNA που έχουν ρυθμιστικό ρόλο μέσα στο κύτταρο. Επιπλέον, η βιογένεσή τους περιέχει δύο βασικά στάδια, τη μεταγραφή του miRNA γονιδίου στο πρόδρομο μόριο miRNA (hairpin miRNA) και την παραγωγή του ώριμου και ενεργού μορίου miRNA (mature miRNA) από το πρόδρομο.

Στις υποδομές DIANA βρίσκονται συγκεντρωμένα δεδομένα γύρω από τα miRNAs από διάφορες βάσεις, όπως τη βάση miRBase που καταγράφει πληροφορίες για τα ίδια τα miRNA μόρια, τη συλλογή KEGG pathways που περιέχει πληροφορίες βιολογικών μονοπατιών με περιγραφές κειμένων που αναπαριστούν την γνώση των βιολόγων σχετικά με τις μοριακές αλληλεπιδράσεις και τα δίκτυα αλληλεπίδρασης, και τη βάση mirGen που περιέχει πληροφορίες σχετικά με τη θέση των γονιδίων και των παράγωγων μορίων [250].

### 6.2.2 Σχήμα Δεδομένων

Σε αυτή την ενότητα περιγράφουμε το σχήμα της βάση δεδομένων που χρησιμοποιήσαμε. Τα δεδομένα έχουν συλλεχθεί στα πλαίσια ανάπτυξης των διαδικτυακών εφαρμογών DIANA, που υποστηρίζουν πλήθος πολύπλοκων ροών εργασιών επιτρέποντας σε χρήστες που δεν έχουν καλή γνώση πληροφορικής να επιτύχουν πολύπλοκες και εξειδικευμένες αναλύσεις γύρω από την λειτουργικότητα των miRNAs.

Τα δεδομένων συλλέχθηκαν από γνωστές επιστημονικές βάσεις δεδομένων, ανάμεσά τους η MirBase, η KEGG Pathways και η Ensembl, και είναι αποθηκευμένα σε μια σχεσιακή βάση δεδομένων. Η βάση δεδομένων αποτελείται από πίνακες στους οποίους αποθηκεύονται οι βασικές οντότητες του miRNA κόσμου (hairpin miRNAs, mature miRNAs, γονίδια, μετάγραφα, κτλ), καθώς και πίνακες που μοντελοποιούν τις σχέσεις των οντοτήτων. Στον Πίνακα 6.8 φαίνεται ένα μέρος της miRNA σχεσιακής βάσης.

Στους πίνακες που παρουσιάζονται οργανώνεται όλη η πληροφορία των βιολογικών πειραμάτων αλλά κωδικοποιούνται και όλες οι αλλαγές που υφίστανται τα βιομόρια στη διάρκεια της ζωής τους, κάνοντας αναγκαία και καίριας σημασίας την υποστήριξη της προελευσιμότητας, καθώς και της διατηρησιμότητας κατά την σχεδίαση του λεξιλογίου ως ανοιχτά διασυνδεδεμένα δεδομένα.

**Πίνακας 6.8:** Μέρος το σχήματος της miRNA βάσης δεδομένων

| **Core tables** | **Column Description** |
|---|---|
| Hairpins | id (`mima_id`), name, sequence, species, gene location info, etc. |
| Matures | id (`mimat`), name, sequence, species. |
| Transcripts | tid, id given from ensembl.org (`enstid`), species, DNA strand, gene location info, etc. |
| ProteinGenes | id given from ensembl.org (`ensgid`), name, description. |
| Keggs | id given from genome.jp (`kegg_id`), name. |
| Tissues | name, species. |

| **Join Tables** | **Column Description** |
|---|---|
| MatureHairpinConn | It relates matures and hairpins. |
| MicroT5Interactions | It contains all the experimentally verified gene-mature interactions (bindings). |
| ProteinGeneKeggConn | It relates genes to kegg pathways. |
| MatureTissueConn | It relates matures to tissues. |

### 6.2.3 Μοντέλο για την Διαχείριση Αλλαγών και Εκδόσεων

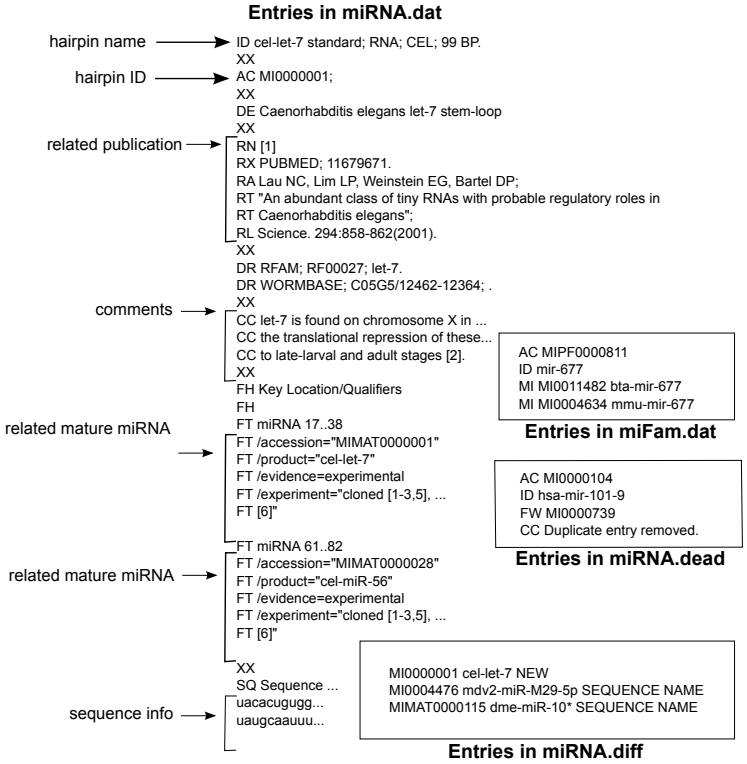
Η miRBase είναι μια βάση που δημοσιεύει πληροφορίες για τα miRNAs (hairpin και mature). Στην βάση εμπεριέχονται πληροφορίες για 18.589 hairpin miRNAs και για 21.881 mature miRNAs. Κάθε εγγραφή της miRBase αναπαριστά ένα ηαιρπιν miRNA με πληροφορίες για τη θέση του πάνω στο γονιδίωμα, την ακολουθία του καθώς και πληροφορίες για τα ματυρε miRNAs που προέρχονται από αυτό. Όλες αυτές οι πληροφορίες μεταβάλλονται με τον χρόνο, καθώς νέα πειραματικά δεδομένα γίνονται διαθέσιμα. Για τον λόγο αυτό η miRBase δημοσιεύεται σε εκδόσεις (versions), όπου κάθε φορά διατίθεται η τρέχουσα πεποίθησή μας για την επιστημονική πραγματικότητα. Επιπλέον, για ολοκληρωμένη εικόνα της εξέλιξης της γνώσης η miRBase διατηρεί μια λίστα αρχείων που καταγράφουν τις διαφορές ανάμεσα σε δύο διαδοχικές εκδόσεις, επιτρέποντάς μας την μοντελοποίηση της εξέλιξης των δεδομένων. Πληροφορίες που σχετίζονται με τις αλλαγές των hairpins και matures κωδικοποιούνται από το mirBase στα ακόλουθα αρχεία:

- **miRNA.dat** Το συγκεκριμένο αρχείο συγκεντρώνει όλες τις πληροφορίες που αναφέρονται σε κάθε hairpin miRNA της τρέχουσας έκδοσης (αναγνωριστικό, όνομα, σχετιζόμενα matures, σημαντικές σχετιζόμενες δημοσιεύσεις, ακολουθία κ.α.). Παράδειγμα από καταχωρίσεις του miRNA.dat παρουσιάζονται στο Σχήμα 6.5.

- **miRNA.diff** Το συγκεκριμένο αρχείο συγκεντρώνει γενικές πληροφορίες για τις αλλαγές που έχουν συμβεί σε σχέση με τις προηγούμενες εκδόσεις.

- **miRNA.dead** Το συγκεκριμένο αρχείο συγκεντρώνει όλα τα miRNA τα οποία έχουν πια διαγραφεί. Το αρχείο αυτό είναι αυξητικό, με την έννοια ότι σε κάθε νέα έκδοση το αρχείο αυτό περιέχει ό,τι περιείχε και το αντίστοιχο αρχείο στην προηγούμενη έκδοση, συν κάποιες επιπλέον εγγραφές.

- **miFam.dat** Το συγκεκριμένο αρχείο οργανώνει τα hairpin miRNAs σε οικογένειες, δηλαδή σε ομάδες των οποίων κοινό χαρακτηριστικό είναι τα παρόμοια matures.




**Σχήμα 6.5:** Παραδείγματα miRNA αρχείων

Από μελέτη των παραπάνω αρχείων οι αλλαγές που παρατηρούνται στα δεδομένα ανάμεσα στις διάφορες εκδόσεις της βάσης miRBase είναι οι παρακάτω.
Σχετικά με τα hairpin έχουμε τις ακόλουθες αλλαγές.

- NEW: Ένα νέο hairpin δημιουργείται.
- NAME: Το hairpin αλλάζει όνομα.
- SEQUENCE (SEQ): Το hairpin αλλάζει ακολουθία.
- NAME/SEQUENCE (NS): Το hairpin αλλάζει όνομα και ακολουθία.
- DELETE (DEL): Το hairpin διαγράφεται.

Ομοίως, για τα mature έχουμε τις ακόλουθες αλλαγές.

- NEW: Ένα νέο mature δημιουργείται.
- NAME: Το mature αλλάζει όνομα.
- SEQUENCE (SEQ): Το mature αλλάζει ακολουθία.
- NAME/SEQUENCE (NS): Το mature αλλάζει όνομα και ακολουθία.
- ADD PARENT HAIRPIN (APH): Ένα νεο hairpin προστίθεται στην λίστα των ηαιρπινς που παράγουν το mature.
- REMOVE PARENT HAIRPIN (RPH): Ένα hairpin αφαιρείται από τη λίστα των ηαιρπινς που παράγουν το mature.

**Πίνακας 6.9:** Πίνακας HairpinsHistory

| mimaid | change | name | seq | first_appearance | last_appearance |
|---|---|---|---|---|---|
| ..1364 | NEW | dre-mir-10b | ..xyz.. | 13 | 15 |
| ..1364 | NAME | dre-mir-10b-1 | ..yzx... | 16 | 17 |
| ..1364 | SEQ | dre-mir-10b-1 | ..sdf.. | 18 | 19 |
| ..1364 | SEQ | dre-mir-10b-1 | ..xxx.. | 20 | 32 |

**Πίνακας 6.10:** Πίνακας MaturesHistory

| mimat | change | name | seq | par. hairpin | first_appearance | last_appearance |
|---|---|---|---|---|---|---|
| ..9477 | NEW | bfl-miR-79 | ..yyx.. | ... | 28 | ... |
| ..9477 | APH | bfl-miR-79 | ..xyz.. | ..021 | 28 | 29 |
| ..9477 | NS | bfl-miR-9-3p | ..xzy.. | ... | 30 | 32 |

– DELETE (DEL): Το mature διαγράφεται.

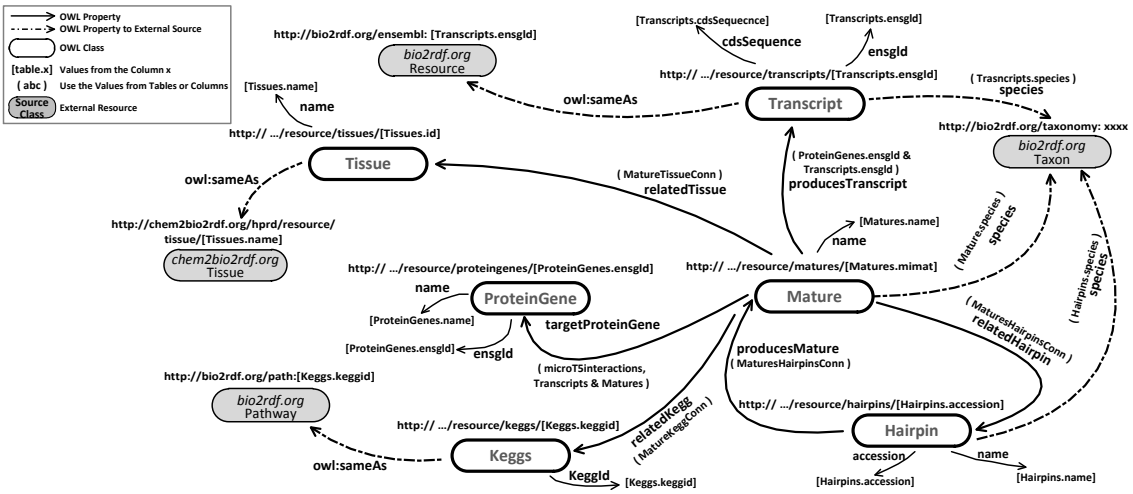
Για να διαχειριστούμε τις αλλαγές διατηρούμε τους σχεσιακούς πίνακες HairpinsHistory και MaturesHistory. Για κάθε αλλαγή κρατάμε τις ακόλουθες πληροφορίες: ένα κλειδί για την εγγραφή, το μοναδικό αναγνωριστικό όνομα του μορίου που προέρχεται από την miRBase που λειτουργεί ως ξένο κλειδί, τον τύπο της αλλαγής που υπέστη, την έκδοση την βάσης που έγινε η αλλαγή, την έκδοση της βάσης πριν από την επόμενη αλλαγή (ορίζοντας ουσιαστικά το διάστημα εγκυρότητας), καθώς και τις έγκυρες πληροφορίες ακολουθίας και ονόματος για το δεδομένο διάστημα.

Στον Πίνακα 6.9 φαίνεται ένα παράδειγμα εγγραφών από τον πίνακα hairpinHistory, στον οποίο αποθηκεύονται οι μεταβαλλόμενες πληροφορίες για ηαιρπιν miRNAs. Αναλυτικότερα, στον Πίνακα 6.9 το hairpin miRNA με μοναδικό αναγνωριστικό ...1364, όπως ορίζεται από την miRBase, καταχωρείται για πρώτη φορά στην έκδοση 5.0 της miRBase. Στην έκδοση 7.0 άλλαξε το όνομά του από dre-mir-10b σε dre-mir-10b-1. Καμία αλλαγή δεν υφίσταται μέχρι την έκδοση 8.0, κατά την οποία άλλαξε η αλληλουχία του. Τέλος, μια ακόμα αλλαγή στην αλληλουχία του συνέβη στην έκδοση 8.2. Από την έκδοση 8.2 μέχρι και την τρέχουσα έκδοση της miRBase, την έκδοση 18, δεν υφίσταται καμία αλλαγή όσο αφορά τα χαρακτηριστικά του. Παρόμοια δομή έχει και ο πίνακας matureHistory (Πίνακας 6.10) που διατηρεί τις μεταβαλλόμενες πληροφορίες για τα mature miRNAs.

### 6.2.4 Μοντελοποίηση Εξελισσόμενων miRNA Διασυνδεδεμένων Δεδομένων

#### 6.2.4.1 Υπόβαθρο

Για την δημοσίευση της επιμελημένης πληροφορίας των miRNA από την σχεσιακή βάση ως Διασυνδεδεμένα Δεδομένα ακολουθήσαμε την προσέγγιση του "εικονικού" RDF, δηλαδή την δυνατότητα μοντελοποίησης δεδομένων σε RDF μορφή χωρίς να είναι ρητά αποθηκευμένα σε RDF. Προσεγγίσεις αυτής της μορφής μας δίνουν το πλεονέκτημα να έχουμε πρόσβαση σε δεδομένα που είναι αποθηκευμένα σε μη RDF μορφή, χωρίς την ανάγκη διατήρησης δύο βάσεων, μια για την σχεσιακή μορφή και μια για την RDF μορφή. Το εργαλείο D2R server [77] είναι ένα δημοφιλές εργαλείο που μας δίνει την δυνατότητα δημοσίευσης δεδομένων από σχεσιακές βάσεις σε RDF. Τα περιεχόμενα




**Σχήμα 6.6:** Σχήμα και αντιστοιχίσεις: Τρέχουσα έκδοση

της σχεσιακής βάσης αντιστοιχίζονται σε RDF χρησιμοποιώντας τη δηλωτική γλώσσα D2RQ, ειδικά σχεδιασμένη για το σκοπό αυτό.

Μια αντιστοίχηση γραμμένη σε D2RQ δηλώνει πώς αναγνωρίζονται οι RDF πόροι, τις σχέσεις μεταξύ των RDF πόρων, όπως και πώς παράγονται τιμές (literals) για τις ιδιότητες των RDF πόρων από τα περιεχόμενα της βάσης. Οι αντιστοιχίσεις σε D2RQ δηλώνονται μέσα από δύο βασικά στοιχεία τα ClassMaps και τα PropertyBridges. Τα ClassMaps χρησιμοποιούνται για να ορίσουν τις RDF κλάσεις και τη δημιουργία των URIs χρησιμοποιώντας URI patterns. Για παράδειγμα, το πρότυπο `hairpins/@@diana_hairpins.mima_id@@` ορίζει ένα σχετικό URI, όπως το .../hairpins/MI0000005 χρησιμοποιώντας την τιμή της στήλης `mima_id` (το μοναδικό αναγνωριστικό ενός μορίου miRNA όπως δίνεται από την miRBase) από τον πίνακα hairpin της βάσης. Ο D2R σερερ μετατρέπει τα σχετικά URIs σε απόλυτα επεκτείνοντας τα με το βασικό URI του σερερ. Ο ακόλουθος ορισμός ClassMap ορίζει την κλάση των πόρων Hairpin:

```
map:Hairpins a d2rq:ClassMap;
    d2rq:dataStorage map:database;
    d2rq:uriPattern "hairpins/@@diana_hairpins.mima_id@@";
    d2rq:class diana:Hairpin;
    d2rq:classDefinitionLabel "Hairpin";
```

Κάθε ClassMap συνδέεται με ένα σύνολο από PropertyBridges που ορίζουν πως δημιουργούνται οι ιδιότητες ενός RDF πόρου. Οι ιδιότητες μπορεί να έχουν ως αντικείμενο είτε πραγματικές τιμές (literals), είτε άλλους RDF πόρους. Το ακόλουθο παράδειγμα ορίζει την ιδιότητα `diana:name`, μια ιδιότητα με αντικείμενο μια τιμή για τις οντότητες Hairpin με την χρήση PropertyBridges. Οι τιμές της ιδιότητας δημιουργούνται από τις τιμές της στήλη `name` του πίνακα `diana_haipins`:

```
map:diana_hairpins_name a d2rq:PropertyBridge;
    d2rq:belongsToClassMap map:Hairpins;
    d2rq:property diana:name;
    d2rq:propertyDefinitionLabel "Hairpins name";
    d2rq:column "diana_hairpins.name";
```

#### 6.2.4.2 Σχήμα και Αντιστοιχήσεις

Σε αυτή την ενότητα παρουσιάζουμε το μοντέλο που προτείνουμε για την μοντελοποίηση των ΔΔ. Το μοντέλο καταγράφει και μοντελοποιεί τις αλλαγές πάνω σε διαχρονικά RDF δεδομένα με στόχο να έχουμε πρόσβαση όχι μόνο στην τρέχουσα έκδοση των δεδομένων, αλλά και σε κάθε προηγούμενη έκδοσή τους. Επιπλέον, θέλουμε να μπορούμε να εντοπίζουμε όλες τις αλλαγές που διατρέχουν τα δεδομένα μας, όπως και την χρονική στιγμή που συνέβη η αλλαγή. Στο προτεινόμενο μοντέλο υιοθετούμε την προσέγγιση RDF οντοτήτων και ιδιοτήτων που μοντελοποιούν την πληροφορία των εκδόσεων. Επιπλέον, μοντελοποιούμε τις πολύπλοκες αλλαγές που υφίστανται τα δεδομένα ως RDF οντότητες.

Για να αναπαραστήσουμε ενημερωμένες οντότητες (π.χ. έναν miRNA πόρο που βρίσκεται στην τελευταία του έκδοση), χρησιμοποιούμε ένα γενικό URI που βασίζεται στην κλάση του πόρου, δηλαδή στην πληροφορία που παρέχεται από την ιδιότητα rdf:type (concept) και στο μοναδικό αναγνωριστικό του πόρου http://domain/concept/identifier. Με την χρήση τέτοιας μορφής URIs, μπορεί να ανακτηθεί η περιγραφή RDF της πιο σύγχρονης έκδοσης του RDF πόρου που ζητείται. Για την ανάκτηση μιας RDF περιγραφής του ίδιου RDF πόρου που βρίσκεται σε κάποια προηγούμενη έκδοση από το σύνολο των ΔΔ, αρκεί να επεκτείνουμε το γενικό URI με μια χρονική σήμανση (timestamp) που δηλώνει την ζητούμενη έκδοση των ΔΔ http://domain/concept/identifier/timestamp.

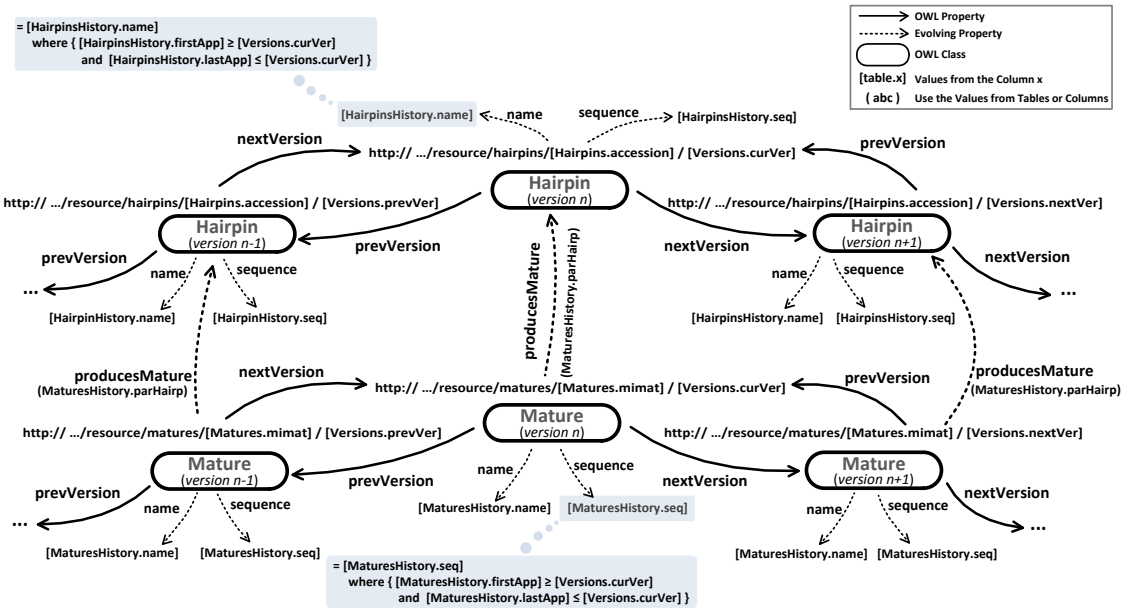
Για παράδειγμα, το . . . /matures/MIMAT0009477 είναι το URI ενός ώριμου μορίου miRNA με αναγνωριστικό MIMAT0009477 στην βάση της miRBase στην τρέχουσα έκδοσή της, ενώ το . . . /matures/MIMAT0009477/17 είναι το URI του ίδιου μορίου miRNA για την έκδοση "17" της miRBase. Ενώ η χρήση αυτής της μορφής URIs μας επιτρέπει την υποστήριξη πλοήγησης ανάμεσα σε διαφορετικές εκδόσεις των οντοτήτων απλά ακολουθώντας τα ίδια τα URIs, η χρήση ιδιοτήτων έκδοσης διευκολύνει τις επερωτήσεις διαχρονικών ΔΔ, δηλαδή την ανάκτηση RDF περιγραφών συγκεκριμένων εκδόσεων των ΔΔ.

Προτείνουμε την χρήση τεσσάρων ιδιοτήτων που σχετίζονται με τις εκδόσεις: (1) την ιδιότητα :label με σταθερή τιμή "now", (2) την ιδιότητα :version με τιμές χρονικής σήμανσης, (3) :prevVersion, και (4) :nextVersion. Η πρώτη ιδιότητα μπορεί να χρησιμοποιηθεί σε επερωτήσεις SPARQL για να ανακτηθούν RDF πόροι που βρίσκονται στην τρέχουσα έκδοση μόνο, ενώ η δεύτερη ιδιότητα μπορεί να χρησιμοποιηθεί για να μορφοποιηθούν διαχρονικές επερωτήσεις, δηλαδή επερωτήσεις που επεκτείνονται σε ένα διάφορες εκδόσεις των ΔΔ. Οι άλλες δύο ιδιότητες επιτρέπουν την πλοήγηση ανάμεσα σε προηγούμενη και επόμενη έκδοση του RDF πόρου, αντίστοιχα.

Τα Σχήματα 6.6 και 6.7 παρουσιάζουν το σχήμα τόσο της τρέχουσας έκδοσης όσο και το σχήμα για την διαχείριση διαφορετικών εκδόσεων.

### 6.2.5 Διερεύνηση Εξελισσόμενων miRNA Διασυνδεδεμένων Δεδομένων

Όπως αναφέρεται και προηγούμενος, στην περίπτωση των εξελισσόμενων δεδομένων, νέες προκλήσεις προκύπτουν που σχετίζονται με την δυναμικότητα των δεδομένων και οι οποίες πρέπει να αντιμετωπιστούν. Οι καταναλωτές των δεδομένων (1) πρέπει να έχουν πρόσβαση όχι μόνο στην τελευταία έκδοση των δεδομένων, αλλά και σε όλες τις προηγούμενες και (2) πρέπει να μπορεί να ιχνηλατηθούν οι αλλαγές ανάμεσα στις διαφορετικές εκδόσεις των δεδομένων, καθώς και οι αιτίες και τα αποτελέσματά τους.




**Σχήμα 6.7:** Σχήμα και αντιστοιχίσεις: Διαφορετικές εκδόσεις

| Property | Value |
|---|---|
| diana:changeName | <http://62.217.113.118:8080/resource/mchange/1612> |
| diana:changeNew | <http://62.217.113.118:8080/resource/mchange/34396> |
| diana:changeParentHairpin | <http://62.217.113.118:8080/resource/mchange/MIMAT0010008_28> |
| diana:mimat | MIMAT0010008 |
| diana:name | bfl-miR-129 |
| diana:nextVersion | <http://62.217.113.118:8080/resource/matures/MIMAT0010008/17.0> |
| diana:prevVersion | <http://62.217.113.118:8080/resource/matures/MIMAT0010008/15.0> |
| is diana:producesMature of | <http://62.217.113.118:8080/resource/hairpins/MI0010519/16.0> |
| diana:sequence | CCUUUUUGUGGUUUGGGGCUUUU |
| diana:species | <http://bio2rdf.org/taxonomy:7739> |
| rdf:type | diana:Matures |
| diana:version | 16.0 |

| Property | Value |
|---|---|
| diana:inVersion | 17.0 |
| diana:newName | bfl-miR-129a |
| rdf:type | diana:matureNameChange |

Generated by D2R Server

**Σχήμα 6.8:** Περιγραφή του mature MIMAT0010008 στην έκδοση 16.0

Το παρακάτω SPARQL ερώτημα ανακτά 10 hairpins και τις ακολουθίες τους, οι οποίες είναι τοποθετημένες στο χρωμόσωμα X, λαμβάνοντας υπόψιν την τρέχουσα έκδοση των δεδομένων.

```
SELECT ?h ?s WHERE {
  ?h rdf:type diana:Hairpin.
  ?h diana:sequence ?s.
  ?h diana:chromosome "X".
  ?h diana:label "now". } LIMIT 10
```

Το Σχήμα 6.8 παρουσιάζει την περιγραφή του mature MIMAT0010008 στην έκδοση 16. Το παρακάτω SPARQL ερώτημα ανακτά 10 hairpins τα οποία διαγράφηκαν ή αντικαταστάθηκαν στην έκδοση 1.3. της miRBase, καθώς και τα URIs των αλλαγών.

```
SELECT ?h ?d ?c WHERE {
  ?h rdf:type diana:Hairpin.
  {{?h diana:changeDelete ?d.} UNION {?h diana:changeForward ?c.}}
  ?h diana:version "1.3". } LIMIT 10
```

Τέλος, έχουμε την δυνατότητα να ανακτήσουμε ιστορικά στοιχεία σχετικά με την πραγματοποίηση των αλλαγών. Το παρακάτω SPARQL ερώτημα ανακτά τις αλλαγές που πραγματοποιήθηκαν στο όνομα ή στην ακολουθία του hairpin MI0001364.

```
SELECT ?h ?c ?v WHERE {
  ?h rdf:type diana:Hairpin.
  ?h diana:accession "MI0001364".
  {{?h diana:changeName ?c. ?c diana:inVersion ?v.}
     UNION
  {?h diana:changeSequence ?c. ?c diana:inVersion ?v.}}}
```

### 6.2.6 Σχετικές Εργασίες

Στο πλαίσιο των ΔΔ, πολυάριθμες προσεγγίσεις έχουν προταθεί για τη μελέτη των προβλημάτων της εξέλιξης, εκδόσεων, και ανίχνευση αλλαγών. Το [330], προτάσσει τον όρος δυναμική δεδομένων, μελετώντας την αντιμετώπιση των αλλαγών περιεχομένου και διασύνδεση σε ΔΔ. Στο [331] παρουσιάζετε μια συγκριτική μελέτη σχετικά με τις υπάρχον προσεγγίσεις και τα εργαλεία που χρησιμοποιούνται για την ανίχνευση και την περιγράφει της εξέλιξη των ΔΔ. Στο [280], οι συγγραφείς ασχολούνται με τις αλλαγές στη συνδέσεις μεταξύ συνόλων δεδομένων και συγκεκριμένα με το πρόβλημα των σπασμένων συνδέσεων (βροκεν λινκς).

Μια παρόμοια προσέγγιση είναι Silk linking framework [341], το οποίο χρησιμοποιείται για την ανακάλυψη και διατήρηση συνδέσεων μεταξύ πηγών δεδομένων στον Παγκόσμιο Ιστό. Αναφορικά με τις προσεγγίσεις που μελετάνε θέματα εκδόσεων, το πλαίσιο Memento [126] παρέχει μηχανισμό εκδόσεων σε πόρους ΔΔ . Τέλος, στο [114] προτείνονται τα χρονοδιαγράμματα σαν έναν τρόπο χρονική αναπαράστασης και διαχείρισης των ΔΔ.

Αρκετές προσπάθειες έχουν γίνει πρόσφατα για την παροχή υπηρεσιών επιστημονικών ΔΔ. Το W3C έχει δημιουργήσει το *Semantic Web Health Care and Life Sciences Interest Group* (HCLS)[3], με στόχο να εκμεταλλευτεί τεχνολογίες του Σημασιολογικού Ιστού για τη διαχείριση και την αναπαράσταση των βιολογικών και ιατρικών δεδομένων. Η ομάδα HCLS εργάζεται πάνω στο έργο *Linking Open Drug Data* (LODD), το οποίο παρέχει ΔΔ που εξάγονται από διάφορες πηγές δεδομένων, όπως *ClinicialTriasl.gov*, *DrugBank*, *DailyMed*, κλπ. Επιπλέον, το *Bio2RDF*[4] παρέχει ΔΔ που παράγονται από πάνω από 30 πηγές βιολογικών δεδομένων. Ορισμένες παλαιότερες προσπάθειες περιλαμβάνουν *YeastHub* [106], *LinkHub* [304], *BioDash* [265] και *BioGateway*[5]. Τέλος, *Chem2Bio2RDF* [102] ενοποιεί χημικές και βιολογικές πληροφορίες.

### 6.2.7 Επίλογος

Σε αυτή την ενότητα παρουσιάσαμε την εργασία μας σχετικά με την μοντελοποίησης, δημοσίευσης και διερεύνησης εξελισσόμενων επιστημονικών δεδομένων. Συγκεκριμένα, προτείνουμε ένα RDF μοντέλο αλλαγών για να την περιγραφή των εξελισσόμενων δεδομένων. Βασισμένοι σε αυτό το μοντέλο, μετατρέψαμε παραδοσιακά βιολογικά δεδομένα σε εξελισσόμενα Διασυνδεδεμένα Δεδομένα. Τέλος, αναπτύξαμε μια υποδομή διασυνδεδεμένων δεδομένων για την διερεύνηση, ανάκτηση και μελέτη εξελισσόμενων βιολογικών οντοτήτων

[3] www.w3.org/blog/hcls

[4] bio2rdf.org

[5] www.semantic-systems-biology.org/biogateway

# Κεφάλαιο 7

# Επίλογος

## 7.1 Σύνοψη

Αρχικά, μελετήσαμε το πρόβλημα της εύρεσης και της ταξινόμησης αντικειμένων που θεωρούνται προτιμητέα από ένα σύνολο χρηστών. Για το πρόβλημα αυτό, εισάγαμε και προτείναμε μια αντικειμενική και δίκαιη ερμηνεία αυτού του προβλήματος, βασισμένη σε Pareto συνάθροιση (aggregation). Λαμβάνοντας υπόψιν αυτή την ερμηνεία, μελετήσαμε τρία σχετικά προβλήματα. Το πρώτο είναι η εύρεση των αντικειμένων που θεωρούνται ομόφωνα ιδανικά από όλο το σύνολο των χρηστών. Στο δεύτερο πρόβλημα, χαλαρώνεται η απαίτηση της ομοφωνίας και απαιτείται μόνο ένα ποσοστό των χρηστών να συμφωνεί. Τέλος, στο τρίτο πρόβλημα, προτείνεται ένα αποτελεσματικό σχήμα ταξινόμησης (ranking scheme) βασισμένο σε Pareto συνάθροιση. Επιπλέον, μελετήθηκαν μερικές ενδιαφέρουσες προεκτάσεις των προαναφερθέντων προβλημάτων που περιλαμβάνουν τα παρακάτω ζητήματα: ιδιότητες με πολλαπλές τιμές (multi-values attributes), μη-δεντροειδείς ιεραρχίες (non-tree hierarchies), δεικτοδότηση υποχώρων subspace indexing, και αντικειμενικά χαρακτηριστικά (objective attributes). Μια λεπτομερή πειραματική αξιολόγηση επιβεβαιώνει την αποδοτικότητα και την αποτελεσματικότητα των προτεινόμενων μεθόδων.

Στην συνέχεια, μελετήσαμε λεπτομερειακά μερικούς από τους πιο γνωστούς αλγορίθμους κορυφογραμμής. Εισάγαμε ένα ρεαλιστικό μοντέλο για τις I/O λειτουργίες. Επιπλέον, μελετήσαμε λεπτομερώς τη διαχείριση των αντικειμένων μέσα στη μνήμη ( in-memory objects). Συγκεκριμένα, εισάγαμε διάφορες πολιτικές για δύο βασικές λειτουργίες: τη διάσχιση και την απομάκρυνση των in-memory αντικειμένων. Και οι δύο αυτές λειτουργίες έχουν σημαντικές επιπτώσεις τόσο στον αριθμό των απαιτούμενων I/Os όσο και στον απαιτούμενο CPU χρόνο. Η πειραματική αξιολόγηση των αλγορίθμων, πραγματοποιήθηκε χρησιμοποιώντας υλοποιήσεις, αυστηρά βασισμένες σε δευτερεύουσα μνήμη και όχι σε προσομοιώσεις. Από την αξιολόγηση πάνω σε συνθετικά και αληθινά σύνολα δεδομένων, προέκυψαν χρήσιμα συμπεράσματα. Συγκεκριμένα, δείχνουμε ότι, σε πολλές περιπτώσεις και αντίθετα με την κοινή πεποίθηση, οι αλγόριθμοι που πραγματοποιούν προ-επεξεργασία (τυπικά ταξινομούν) την βάση δεδομένων δεν είναι πιο αποδοτικοί.

Επιπλέον, μελετήσαμε το πρόβλημα της άμεσης οπτικής διερεύνησης σε μεγάλα σύνολα δεδομένων. Ως αποτέλεσμα, προτείναμε ένα πλαίσιο που προσφέρει προσωποποιημένη πολυεπίπεδη διερεύνηση και ανάλυση αριθμητικών και χρονικών δεδομένων. Το πλαίσιό μας βασίζεται σε μια ελαφριά δεντροειδούς δομή δεδομένων η οποία ομαδοποιεί τα αντικείμενα εισόδου σε ένα ιεραρχικό πολυεπίπεδο μοντέλο. Επιπλέον, ορίσαμε

διαφορετικά σενάρια διερεύνησης, υποθέτοντας διαφορικές προτιμήσεις των χρηστών. Προκειμένου να επιτευχθεί η αποδοτική διερεύνηση σε μεγάλα σύνολα δεδομένων, το πλαίσιό μας προσφέρει σταδιακή κατασκευή (incremental construction) καθώς και prefetching, βασιζόμενα στην αλληλεπίδραση με τον χρήστη. Ακόμα, το πλαίσιο παρέχει μια μέθοδο η οποία δυναμικά και αποδοτικά προσαρμόζει την υπάρχουσα ιεραρχία σε μια νέα, υιοθετώντας τις προτιμήσεις του χρήστη. Μια λεπτομερή θεωρητική ανάλυση, μια αξιολόγηση επίδοσης και μια μελέτη με πραγματικούς χρήστες, αναδεικνύουν την αποδοτικότητα και την αποτελεσματικότητα του προτεινόμενου πλασίου. Το πλαίσιο υλοποιήθηκε σαν ένα πρωτότυπο web-based εργαλείο, που ονομάζεται *synopsViz* και προσφέρει πολυεπίπεδη οπτική διερεύνηση και ανάλυση σε σύνολα Διασυνδεδεμένων Δεδομένων.

Στην συνέχεια, μελετήσαμε το πρόβλημα της οπτικοποίησης και της διερεύνησης πολύ μεγάλων γράφων. Για αυτό το πρόβλημα εισάγαμε το *graphVizdb*, μια νέα πλατφόρμα που προσφέρει διαδραστική οπτικοποίηση μεγάλων γράφων. Η προτεινόμενη πλατφόρμα βασίζεται σε ένα νέο τρόπο αλληλεπίδρασης με το οπτικοποιημένο γράφο που είναι παρόμοιος με εκείνον της διερεύνησης γεωγραφικών χαρτών. Η πλατφόρμα οφείλει την αποδοτικότητά της σε μια καινοτόμα τεχνική για την ευρετηρίαση (indexing) και την αποθήκευση του γράφου. Προκειμένου να οπτικοποιήσουμε πολύ μεγάλους γράφους, προτείναμε μια partition-based προσέγγιση οπτικοποίησης γραφήματος. Αξιολογήσαμε την απόδοση των μεθόδων μας χρησιμοποιώντας αρκετά πραγματικά σύνολα γραφικών δεδομένων. έλος, αναπτύξαμε ένα web-based πρότυπο το οποίο υποστηρίζει τέσσερις κύριες λειτουργίες: διαδραστική πλοήγηση, πολυεπίπεδη διερεύνηση, επιλογή και διαχείριση υπογράφων και αναζήτηση με λέξεις-κλειδιά.

Επιπρόσθετα, μελετήσαμε το πρόβλημα της διαλειτουργικότητας μεταξύ του Σημασιολογικού και του XML περιβάλλοντος. Για αυτό το πρόβλημα, προτείναμε το πλαίσιο *SPARQL2XQuery* το οποίο γεφυρώνει το κενό ετερογένειας και δημιουργεί ένα διαλειτουργικό περιβάλλον. Πιο συγκεκριμένα, ορίζουμε ένα μοντέλο αντιστοιχίσεων για την διατύπωση αντιστοιχίσεων μεταξύ OWL–RDF/S και XML Schema, καθώς και μια μέθοδο για μετάφραση από SPARQL σε XQuery. Τέλος, πραγματοποιήθηκε μια λεπτομερή πειραματική αξιολόγηση προκειμένου να μελετηθεί η αποδοτικότητα των προτεινόμενων μεθόδων.

Στην συνέχεια, μελετήσαμε το πρόβλημα της σημασιολογικής ανάκτησης πληροφοριών. Για το πρόβλημα αυτό, προτείναμε το πλαίσιο *GoNTogle* το οποίο υποστηρίζει ένα σημασιολογικό μοντέλο επισημειώσεων. Το πλαίσιο παρέχει τόσο αυτόματο όσο και χειροκίνητο μηχανισμό επισημειώσεων. Ο μηχανισμό αυτόματων επισημειώσεων βασίζεται σε μια μέθοδο εκμάθησης η οποία χρησιμοποιεί το ιστορικό επισημειώσεων του χρήστη, καθώς και πληροφορίες κειμένου, προκειμένου να προτείνει αυτόματα επισημειώσεις για νέα κείμενα. Επιπλέον, εισάγαμε μια υβριδική μέθοδο ανάκτησης (hybrid retrieval method) που παρέχει έναν ευέλικτο συνδυασμό textual-based και semantic-based ανάκτησης σε συνδυασμό με ανεπτυγμένες σημασιολογικές λειτουργίες. Οι προτεινόμενες μέθοδοι εφαρμόστηκαν σε ένα πλήρως λειτουργικό εργαλείο και η αποτελεσματικότητά τους επιβεβαιωθείτε πειραματικά.

Τέλος, μελετάμε το πρόβλημα της μοντελοποίησης, δημοσίευσης και διερεύνησης εξελισσόμενων επιστημονικών δεδομένων, υιοθετώντας τεχνικές των Διασυνδεδεμένων Δεδομένων. Για το συγκεκριμένο πρόβλημα, προτείναμε ένα RDF μοντέλο αλλαγών για να την περιγραφή των εξελισσόμενων δεδομένων. Βασισμένοι σε αυτό το μοντέλο, μετατρέψαμε παραδοσιακά βιολογικά δεδομένα σε εξελισσόμενα Διασυνδεδεμένα Δεδομένα. Η υποδομή διασυνδεδεμένων δεδομένων που αναπτύξαμε μπορεί να βοηθήσει

τους βιολόγους να διερευνήσουν βιολογικές οντότητες καθώς και να μελετήσουν την εξέλιξή τους.

## 7.2 Μελλοντικές Εργασίες

Κατά την εκπόνηση της παρούσας διατριβής, αναγνωρίσαμε τα ακόλουθα ενδιαφέροντα θέματα τα οποία προτείνουμε για μελλοντική εργασία.

- Πρόσφατα, υπήρξε αρκετό ενδιαφέρον γύρω από του partitioning-based αλγορίθμους κορυφογραμμής. Αν και αυτές οι προσεγγίσεις μειώνουν σημαντικά τον αριθμό των ελέγχων μεταξύ των αντικειμένων, δεν λαμβάνουν υπόψιν τον αριθμό των I/Os που είναι πιθανό να απαιτούνται. Αυτό έχει ως αποτέλεσμα, σε περιπτώσεις όπου το μέγεθος της κορυφογραμμής υπερβαίνει το μέγεθος της μνήμης, οι αλγόριθμοι αυτοί να πραγματοποιούν ένα μεγάλο αριθμό I/Os, γεγονός που επηρεάζει σημαντικά τη συνολική απόδοση. Μια καλή εναλλακτική θα ήταν να σχεδιάσουμε έναν απλό scan-based αλγόριθμό (δηλαδή, BLN-like) ο οποίος να λειτουργεί καλά σε I/Os, και να είναι εμπλουτισμένος με ένα ελαφρύ σχήμα διαχωρισμού χώρου (lightweight space partition scheme), το οποίο μπορεί να χρησιμοποιηθεί για να μειωθούν οι έλεγχοι αντικειμένων λαμβάνοντας υπόψιν τόσο την έννοια της κυριαρχίας όσο και της μη-συγκρισιμότητας.

- Σε αυτήν την εργασία, προτείναμε μια partition-based μέθοδο για την οπτικοποίηση πολύ μεγάλων γράφων. Αυτή η μέθοδος υιοθετεί μια προσέγγιση όπου οι οπτικοποιημένες διαμερίσεις συνδυάζονται και οργανώνονται σε μία "global" διαμέριση. Σε αυτά τα πλαίσια, παρουσιάσαμε έναν άπληστο αλγόριθμο (greedy algorithm) που προσπαθεί να αποφύγει τις επικαλύψεις κόμβων, καθώς και να ελαχιστοποιήσει το μήκος των ακμών που συνδέουν διαφορετικές διαμερίσεις. Επί του παρόντος, δουλεύουμε στην αξιολόγηση της αποτελεσματικότητας του προτεινόμενου αλγορίθμου, καθώς και στην ανάπτυξη και στην σύγκριση αρκετών εναλλακτικών μεθόδων. Μια προέκταση που παρουσιάζει πρόκληση θα ήταν να ορίσουμε ένα πιο ευέλικτο και σύνθετο πρόβλημα οργάνωσης των διαμερίσεων (partition organization problem). Μια πιθανή λύση σε αυτό το πρόβλημα θα ήταν να εξετάσουμε επίσης περιστροφή των διαμερίσεων, και την επανατοποθέτηση των κόμβων μέσα στις διαμερίσεις. Τέλος, ένα ενδιαφέρον ζήτημα είναι να αποδείξουμε το κατά πόσο ένα πρόβλημα οργάνωσης των διαμερίσεων είναι NP-hard.

- Ένα άλλο ενδιαφέρον πρόβλημα είναι η άμεση οπτικοποίηση μεγάλων γράφων. Σε αυτό το περιβάλλον, δεν υπάρχει στάδιο προεπεξεργασίας και ο γράφος αποθηκεύεται σε ένα αρχείο ως raw data. Αρχικά, ο χρήστης επιλέγει έναν κόμβο για να ξεκινήσει τη διερεύνησή του. Αυτός ο εναρκήριος κόμβος μπορεί να καθοριστεί από αρκετές τεχνικές, δηλαδή παρέχοντας το όνομα του κόμβοι, χρησιμοποιώντας αναζήτηση με λέξεις-κλειδιά, συστάσεις κτλ. Κάθε φορά που ο χρήστης επισκέπτεται έναν κόμβο, οι κόμβοι που συνδέονται με αυτόν τον κόμβο μέσα με προκαθορισμένο μήκος μονοπατιού, ανακτώνται από το αρχείο και οπτικοποιούνται. Στο πρόβλημα που περιγράφεται, προκύπτουν αρκετά ζητήματα που παρουσιάζουν προκλήσεις υπό την προϋπόθεση διερεύνηση σε πραγματικό χρόνο. Για παράδειγμα, πώς να βρεις αποδοτικά και να ανακτήσεις από ένα μεγάλο raw

αρχείο, τα κομμάτια του γράφου που απαιτούνται. Τέλος, σε περιπτώσεις που οπτικοποιείται ένας μεγάλος αριθμός από ακμές και κόμβους, απαιτείται ένα μεγάλο ποσό μνήμης από το user interface. Έτσι, σε τέτοιες περιπτώσεις, παρουσιάζεται το πρόβλημα της επιλογής τον κομματιών του οπτικοποιημένου γραφήματος που θα αφαιρεθεί από τον καμβά.

– Σχετικά με την πολυεπίπεδη διερεύνηση, το πλαίσιο που προτείνεται σε αυτήν την εργασία θεωρεί ότι τα αντικείμενα οργανώνονται και διερευνούνται με βάση ένα χαρακτηριστικό. Μια προέκταση που παρουσιάζει προκλήσεις θα ήταν να παρέχονται μέθοδοι και δομές που υποστηρίζουν διερεύνηση σε παραπάνω από μία διαστάσεις. Για παράδειγμα, επέκταση του πλαισίου μας προκειμένου να προσφέρει πολυεπίπεδη διερεύνηση (πάνω σε δύο ιδιότητες) χρησιμοποιώντας διαγράμματα διασποράς (scatter plots). Επιπλέον, ένα ενδιαφέρον πρόβλημα θα ήταν η τροποποίηση των μεθόδων προκειμένου να λαμβάνουν υπόψιν ζητήματα σχετικά με την αποδοτική διαχείριση αντικειμένων. Για παράδειγμα, να ξανασχεδιάσουμε τη μέθοδο μας προκειμένου να μειώσουμε το I/O κόστος σε raw δεδομένα, να ελαχιστοποιήσουμε τον αριθμό των αντικειμένων στα οποία πρέπει να έχει πρόσβαση σε κάθε στάδιο της κλιμακωτή κατασκευής, κτλ.

– Σε μια περίπτωση διερεύνηση, είναι σύνηθες οι χρήστες να ενδιαφέρονται να βρουν κάτι ενδιαφέρον και χρήσιμο χωρίς να ξέρουν ακριβώς τι ψάχνουν, μέχρι να έρθει η στιγμή να το ταυτοποιήσουν. Σε αυτή τη περίπτωση, οι χρήστες πραγματοποιούν μια σειρά από λειτουργίες (π.χ. ερωτήσεις), όπου το αποτέλεσμα κάθε λειτουργίας καθορίζει τη σύνθεση της επόμενης λειτουργίας. Σε αυτό το περιβάλλον, η προεπιλογή και η ανάκτηση των συνόλων δεδομένων (caching and prefetching) στα οποία είναι πιθανό να αποκτήσει πρόσβαση ο χρήστης στο κοντινό μέλλον μπορεί να μειώσει σημαντικά τον χρόνο απόκρισης. Αρκετές εργασίες έχουν μελετήσει πρόσφατα αυτό το πρόβλημα. Μια ενδιαφέρουσα κατεύθυνση θα ήταν να αναπτυχθούν τεχνικές caching και prefetching για διαφορετικές περιπτώσεις διερεύνησης. Αυτές οι περιπτώσεις μπορεί να χαρακτηρίζονται από τις υποστηριζόμενες διαδραστικές λειτουργίες και/ή τον τύπο οπτικοποίησης. Για παράδειγμα να αναπτύξουμε "operation-aware" τεχνικές caching και prefetching για μια συγκεκριμένη λειτουργία π.χ. pan, drill-down, roll-up, zoom. Σε συνδυασμό με τις "operation- aware" τεχνικές, θα ήταν ενδιαφέρον να αναπτυχθούν "type-aware" τεχνικές βασισμένες στον τύπο οπτικοποίησης π.χ.graph, line chart, scatter, histograms. Τέλος, ένα ενδιαφέρον θέμα θα ήταν η προσαρμογή τεχνικών από location-based και spatial-based επεξεργασία ερωτήσεων στα τα πλαίσια της οπτικής διερεύνησης.

– Στα πλαίσια της πολυεπίπεδης διερεύνησης, μια ενδιαφέρουσα κατεύθυνση θα ήταν να επεκτείνουμε το παρουσιαζόμενο πλαίσιο προκειμένου να υποστηρίξουμε περισσότερες πολιτικές παρουσίασης (rendering policies) και λειτουργίες αλληλεπίδρασης. Συγκεκριμένα, θα παρουσίαζε ενδιαφέρον η ανάπτυξη πιο ευέλικτων πολιτικών παρουσίασης π.χ., να παρουσιάζουμε όλους ή έναν αριθμό κόμβων στο παρόν επίπεδο ή να παρουσιάζουμε όλους τους κόμβους παρακάτω (και περιλαμβανομένου) του παρόντος επιπέδου, κτλ. Σχετικά με τις λειτουργίες αλληλεπίδρασης, οι παρούσες υποστηριζόμενες λειτουργίες (δηλαδή roll-up, drill-down) επιτρέπουν στους χρήστες να πλοηγηθούν σε μια ιεραρχία με έναν κάθετο τρόπο, μετακινούμενοι κάτω ή πάνω στα ιεραρχικά επίπεδα. Θα ήταν επίσης χρήσιμο να

υποστηριχτούν λειτουργίες οι οποίες θα επιτρέπουν στους χρήστες να διερευνούν την ιεραρχία οριζόντια. Για παράδειγμα, θα ήταν πολύτιμο οι χρήστες να έχουν πρόσβαση σε διαφορετικά σύνολα από αδελφικούς κόμβους χωρίς την ανάγκη αλλαγής επιπέδου στην ιεραρχία.

– Το ιεραρχικό μοντέλο συνάθροισης (hierarchical aggregation model) που παρουσιάστηκε σε αυτήν την εργασία, οργανώνει τα δεδομένα βασισμένο σε binning μεθόδους. Συγκεκριμένα, προκειμένου να πετύχουμε αποδοτική και άμεση επεξεργασία δεδομένων, υιοθετήθηκαν απλές μέθοδοι binning (δηλαδή, ίσου πλάτος και ίσης συχνότητα). Όμως, είναι γνωστό ότι αυτές οι μέθοδοι είναι ευάλωτες σε skewed data distributions και σε outliers. Έτσι, προκειμένου να χειριστούμε αποδοτικά μη-ομοιόμορφες κατανομές δεδομένων, απαιτούνται πιο προηγμένες discretization μέθοδοι. Όμως, στο περιβάλλον όπου απαιτείται η άμεση επεξεργασία δεδομένων, η χρήση προηγμένων discretization μεθόδων δεν είναι εφικτή (λόγω στη μεγάλη υπολογιστική πολυπλοκότητα). Σε αυτό το πλαίσιο και λαμβάνοντας υπόψιν τη μεγάλη σημασία της μείωσης των δεδομένων (data reduction) στο γενικό πρόβλημα της οπτικοποίησης μεγάλων δεδομένων, η ανάπτυξη τόσο αποτελεσματικών όσο και αποδοτικών τεχνικών μείωσης δεδομένων θα ήταν ενδιαφέρον θέμα.

# Bibliography


[1] *Dublin Core Metadata Element Set.* Dublin Core Metadata Initiative. dublincore.org/documents/dces.

[2] *Encoded Archival Description (EAD).* Library of Congress. www.loc.gov/ead.

[3] *IEEE WG-12: IEEE Standard for Learning Object Metadata (LOM).* ltsc.ieee.org/wg12.

[4] *MARC 21 concise format for bibliographic metadata.* Library of Congress. www.loc.gov/marc/bibliographic/ecbdhome.html.

[5] *Metadata Authority Description Standard (MADS).* Library of Congress. www.loc.gov/standards/mads.

[6] *Metadata Encoding and Transmission Standard (METS).* Library of Congress. www.loc.gov/standards/mets.

[7] *Metadata Object Description Schema (MODS).* Library of Congress. www.loc.gov/standards/mods.

[8] *MPEG-21 Multimedia framework, ISO 21000-17:2003-2007.* Intl. Standardization Organization.

[9] *MPEG-7 Multimedia content description interface, ISO 15938-1-11:2002-2007.* Intl. Standardization Organization.

[10] *Niso Metadata for Images in XML (MIX).* Library of Congress. www.loc.gov/standards/mix.

[11] *Sharable Content Reference Model (SCORM).* Advanced Distributed Learning Initiative (ADL). www.adlnet.gov/scorm/index.aspx.

[12] *SMORE: Create OWL Markup for HTML Web Pages.* http://www.mindswap.org/2005/SMORE.

[13] *Technical Metadata for Text (TextMD).* Library of Congress. www.loc.gov/standards/textMD/.

[14] *Text Encoding and Interchange (TEI).* TEI Consortium. www.tei-c.org.

[15] *Vra Core 4.0.* Visual Resources Association (VRA). www.vraweb.org/projects/vracore4/index.html.

[16] B. A. and et al., editors. *XML Path Language (XPath) 2.0*. W3C Rec., 2007. `www.w3.org/TR/xpath20`.

[17] M. A. and et al., editors. *XQuery 1.0 and XPath 2.0 Functions and Operators*. W3C Rec., 2010. `www.w3.org/TR/xpath-functions`.

[18] J. Abello, F. van Ham, and N. Krishnan. ASK-GraphView: A Large Scale Graph Visualization System. *IEEE Trans. Vis. Comput. Graph.*, 12(5), 2006.

[19] G. Adomavicius and A. Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 17(6), 2005.

[20] S. Agarwal, B. Mozafari, A. Panda, H. Milner, S. Madden, and I. Stoica. BlinkDB: queries with bounded errors and bounded response times on very large data. In *EuroSys*, 2013.

[21] A. Aggarwal and J. S. Vitter. The input/output complexity of sorting and related problems. *Commun. ACM*, 31(9), 1988.

[22] M. Agosti and N. Ferro. A Formal Model of Annotations of Digital Content. *ACM Transactions on Information Systems (TOIS)*, 26(1), 2008.

[23] R. Agrawal and E. L. Wimmers. A Framework for Expressing and Combining Preferences. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2000.

[24] V. Aguilera, S. Cluet, T. Milo, P. Veltri, and D. Vodislav. Views in a large-scale XML repository. *The Intl. Journal on Very Large Data Bases (VLDBJ)*, 11:3, 2002.

[25] W. Akhtar, J. Kopecký, T. Krennwallner, and A. Polleres. XSPARQL: Traveling between the XML and RDF Worlds - and Avoiding the XSLT Pilgrimage. In *Extended Semantic Web Conference (ESWC)*, 2008.

[26] F. Alahmari, J. A. Thom, L. Magee, and W. Wong. Evaluating Semantic Browsers for Consuming Linked Data. In *Australasian Database Conference (ADC)*, 2012.

[27] H. Alani. TGVizTab: An Ontology Visualisation Extension for Protege. In *Workshop on Visualizing Information in Knowledge Engineering*, 2003.

[28] M. I. Ali, N. Lopes, O. Friel, and A. Mileo. Update semantics for interoperability among xml, RDF and RDB - A case study of semantic presence in cisco's unified presence systems. In *Web Technologies and Applications (APWeb)*, 2013.

[29] B. Amann, C. Beeri, I. Fundulaki, and M. Scholl. Querying XML Sources Using an Ontology-Based Mediator. In *Cooperative Information Systems (CoopIS)*, 2002.

[30] D. Archambault, T. Munzner, and D. Auber. Grouse: Feature-Based, Steerable Graph Hierarchy Exploration. In *EuroVis07*, 2007.

[31] D. Archambault, T. Munzner, and D. Auber. GrouseFlocks: Steerable Exploration of Graph Hierarchy Space. *IEEE Trans. Vis. Comput. Graph.*, 14(4), 2008.

[32] D. Archambault, T. Munzner, and D. Auber. Tugging Graphs Faster: Efficiently Modifying Path-Preserving Hierarchies for Browsing Paths. *IEEE Trans. Vis. Comput. Graph.*, 17(3), 2011.

[33] L. Ardissono, A. Goy, G. Petrone, M. Segnan, and P. Torasso. Intrigue: Personalized Recommendation of Tourist Attractions for Desktop and Hand Held Devices. *Applied Artificial Intelligence*, 17(8-9), 2003.

[34] M. Arenas and L. Libkin. XMLdata Exchange: Consistency and Query Answering. *Journal of ACM (JACM)*, 55:2, 2008.

[35] K. J. Arrow. *Social Choice and Individual Values*. Yale University Press, 2nd edition, 1963.

[36] J. A. Aslam and M. H. Montague. Models for Metasearch. In *Intl. ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2001.

[37] G. A. Atemezing and R. Troncy. Towards a linked-data based visualization wizard. In *Workshop on Consuming Linked Data*, 2014.

[38] D. Auber. Tulip - A Huge Graph Visualization Framework. In *Graph Drawing Software*. 2004.

[39] E.-A. Baatarjav, S. Phithakkitnukoon, and R. Dantu. Group Recommendation System for Facebook. In *OTM Workshops*, 2008.

[40] B. Bach, E. Pietriga, and I. Liccardi. Visualizing Populated Ontologies with OntoTrix. *Intl. J. Semantic Web Inf. Syst.*, 9(4), 2013.

[41] L. Baltrunas, T. Makcinskas, and F. Ricci. Group recommendations with rank aggregation and collaborative filtering. In *ACM conference on Recommender systems, RecSys*, 2010.

[42] D. G. Bar and O. Glinansky. Family Stereotyping - A Model to Filter TV Programs for Multiple Viewers. In *Workshop on Personalization in Future TV*, 2002.

[43] I. Bartolini, P. Ciaccia, and M. Patella. Efficient Sort-based Skyline Evaluation. *ACM Transactions on Database Systems (TODS)*, 33(4), 2008.

[44] M. Bastian, S. Heymann, and M. Jacomy. Gephi: An Open Source Software for Exploring and Manipulating Networks. In *Conference on Weblogs and Social Media (ICWSM)*, 2009.

[45] L. Battle, R. Chang, and M. Stonebraker. Dynamic Prefetching of Data Tiles for Interactive Visualization. Technical Report, 2015.

[46] L. Battle, M. Stonebraker, and R. Chang. Dynamic reduction of query result sets for interactive visualizaton. In *IEEE Conference on Big Data*, 2013.

[47] C. Becker and C. Bizer. Exploring the Geospatial Semantic Web with DBpedia Mobile. *J. Web Sem.*, 7(4), 2009.

[48] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 1990.

[49] I. Bedini, G. Gardarin, and B. Nguyen. Deriving Ontologies from XML Schema. In *EDA*, 2008.

[50] I. Bedini, C. Matheus, P. Patel-Schneider, A. Boran, and B. Nguyen. Transforming XML Schema to OWL Using Patterns. In *ICSC*, 2011.

[51] F. Benedetti, L. Po, and S. Bergamaschi. A Visual Summary for Linked Open Data sources. In *Intl. Semantic Web Conference (ISWC)*, 2014.

[52] J. L. Bentley, K. L. Clarkson, and D. B. Levine. Fast Linear Expected-Time Algorithms for Computing Maxima and Convex Hulls. In *ACM-SIAM Symposium on Discrete Algorithms*, 1990.

[53] J. L. Bentley, H. T. Kung, M. Schkolnick, and C. D. Thompson. On the Average Number of Maxima in a Set of Vectors and Applications. *Journal of ACM (JACM)*, 25(4), 1978.

[54] K. Bereta, C. Nikolaou, M. Karpathiotakis, K. Kyzirakos, and M. Koubarakis. SexTant: Visualizing Time-Evolving Linked Geospatial Data. In *Intl. Semantic Web Conference (ISWC)*, 2013.

[55] S. Berkovsky and J. Freyne. Group-based recipe recommendations: analysis of data aggregation strategies. In *ACM conference on Recommender systems, RecSys*, 2010.

[56] A. Bernd, C. Beeri, I. Fundulaki, and M. Scholl. Ontology-Based Integration of XML Web Resources. In *Intl. Semantic Web Conference (ISWC)*, 2002.

[57] D. Berrueta, J. E. Labra, and I. Herman. XSLT+SPARQL: Scripting the Semantic Web with SPARQL embedded into XSLT stylesheets. In *Workshop on Scripting for the Semantic Web*, 2008.

[58] R. Bhagdev, S. Chapman, F. Ciravegna, V. Lanfranchi, and D. Petrelli. Hybrid search: Effectively combining keywords and semantic searches. In *Extended Semantic Web Conference (ESWC)*, 2008.

[59] N. Bikakis. Personalized, Semantic and Exploratory Data Analysis, 2016. PhD Thesis, National Technical University of Athens, Greece.

[60] N. Bikakis, K. Benouaret, and D. Sacharidis. Reconciling Multiple Categorical Preferences with Double Pareto-Based Aggregation. In *Intl. Conference on Database Systems for Advanced Applications (DASFAA)*, 2014.

[61] N. Bikakis, K. Benouaret, and D. Sacharidis. Finding Desirable Objects under Group Categorical Preferences. *Knowledge and Information Systems Journal (KAIS)*, 2015.

[62] N. Bikakis, G. Giannopoulos, T. Dalamagas, and T. K. Sellis. Integrating Keywords and Semantics on Document Annotation and Search. In *Intl. Conference on Ontologies, DataBases, and Applications of Semantics (ODBASE)*, 2010.

[63] N. Bikakis, N. Gioldasis, C. Tsinaraki, and S. Christodoulakis. Querying XML data with SPARQL. In *Intl. Conference on Database and Expert Systems Applications (DEXA)*, 2009.

[64] N. Bikakis, N. Gioldasis, C. Tsinaraki, and S. Christodoulakis. Semantic Based Access over XML Data. In *World Summit on the Knowledge Society (WSKS)*, 2009.

[65] N. Bikakis, J. Liagouris, M. Kromida, G. Papastefanatos, and T. K. Sellis. Towards Scalable Visual Exploration of Very Large RDF Graphs. In *Extended Semantic Web Conference (ESWC)*, 2015.

[66] N. Bikakis, J. Liagouris, M. Krommyda, G. Papastefanatos, and T. K. Sellis. graphVizdb: A Scalable Platform for Interactive Large Graph Visualization. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2016.

[67] N. Bikakis, G. Papastefanatos, M. Skourla, and T. K. Sellis. A Hierarchical Framework for Efficient Multilevel Visual Exploration and Analysis. *Semantic Web Journal*, 2016.

[68] N. Bikakis, D. Sacharidis, and T. K. Sellis. A study on external memory scan-based skyline algorithms. In *Intl. Conference on Database and Expert Systems Applications (DEXA)*, 2014.

[69] N. Bikakis and T. Sellis. Exploration and Visualization in the Web of Big Linked Data: A Survey of the State of the Art. In *International Workshop on Linked Web Data Management (LWDM)*, 2016.

[70] N. Bikakis, M. Skourla, and G. Papastefanatos. rdf: SynopsViz - A Framework for Hierarchical Linked Data Visual Exploration and Analysis. In *Extended Semantic Web Conference (ESWC)*, 2014.

[71] N. Bikakis, C. Tsinaraki, N. Gioldasis, I. Stavrakantonakis, and S. Christodoulakis. The XML and Semantic Web Worlds: Technologies, Interoperability and Integration: A Survey of the State of the Art. In *Semantic Hyper/Multimedia Adaptation - Schemes and Applications, Springer*. 2013.

[72] N. Bikakis, C. Tsinaraki, I. Stavrakantonakis, and S. Christodoulakis. Supporting SPARQL Update Queries in RDF-XML Integration. In *Intl. Semantic Web Conference (ISWC)*, 2014.

[73] N. Bikakis, C. Tsinaraki, I. Stavrakantonakis, N. Gioldasis, and S. Christodoulakis. The SPARQL2XQuery Interoperability Framework. Technical report, 2012. `www.dblab.ntua.gr/~bikakis/SPARQL2XQueryTR2012.pdf`.

[74] N. Bikakis, C. Tsinaraki, I. Stavrakantonakis, N. Gioldasis, and S. Christodoulakis. The SPARQL2XQuery interoperability framework - Utilizing Schema Mapping, Schema Transformation and Query Translation to Integrate XML and the Semantic Web. *World Wide Web*, 18(2), 2015.

[75] S. Bischof, S. Decker, T. Krennwallner, N. Lopes, and A. Polleres. Mapping between RDF and XML with XSPARQL. *J. Data Semantics*, 1:3, 2012.

[76] S. Bischof, N. Lopes, and A. Polleres. Improve Efficiency of Mapping Data between XML and RDF with XSPARQL. In *Intl. Conference on Web Reasoning and Rule Systems (RR)*, 2011.

[77] C. Bizer and R. Cyganiak. D2r server – publishing relational databases on the semantic web. In *Intl. Semantic Web Conference (ISWC)*, 2006.

[78] C. Bizer and R. Cyganiak. D2R Server - Publishing Relational Databases on the Semantic Web. In *Intl. Semantic Web Conference (ISWC)*, 2006.

[79] C. Bizer and A. Schultz. The Berlin SPARQL Benchmark. *Intl. Journal On Semantic Web and Information Systems - Special Issue on Scalability and Performance of Semantic Web Systems*, 2009.

[80] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez. Recommender systems survey. *Knowl.-Based Syst.*, 46, 2013.

[81] P. Bohannon, W. Fan, M. Flaster, and P. Narayan. Information Preserving XML Schema Embedding. In *Intl. Conference on Very Large Databases (VLDB)*, 2005.

[82] H. Bohring and S. Auer. Mapping XML to OWL Ontologies. In *Leipziger Informatik-Tage*, 2005.

[83] T. Boinski, A. Jaworska, R. Kleczkowski, and P. Kunowski. Ontology visualization. In *Conference on Information Technology*, 2010.

[84] A. Bonifati, E. Q. Chang, T. Ho, L. V. S. Lakshmanan, R. Pottinger, and Y. Chung. Schema Mapping and Query Translation in Heterogeneous P2P XML Databases. *The Intl. Journal on Very Large Data Bases (VLDBJ)*, 19:2, 2010.

[85] L. Boratto and S. Carta. State-of-the-Art in Group Recommendation and New Approaches for Automatic Identification of Groups. In *Information Retrieval and Mining in Distributed Environments*. 2011.

[86] S. Börzsönyi, D. Kossmann, and K. Stocker. The Skyline Operator. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2001.

[87] D. Brickley and G. R. V., editors. *RDF Vocabulary Description Language 1.0: RDF Schema*. W3C Rec., 2004. www.w3.org/TR/rdf-schema.

[88] J. M. Brunetti, S. Auer, R. Garcia, J. Klimek, and M. Necasky. Formal Linked Data Visualization Model. In *Intl. Conference on Information Integration and Web-based Applications & Services, (IIWAS)*, 2013.

[89] J. M. Brunetti, R. Gil, and R. Garcia. Facets and Pivoting for Flexible and Usable Linked Data Exploration. In *Interacting with Linked Data Workshop*, 2012.

[90] B. C. *Mapping Relational Data to RDF with Virtuoso's RDF Views*. OpenLink Software, 2007.

[91] D. V. Camarda, S. Mazzini, and A. Antonuccio. LodLive, exploring the web of data. In *Conference on Semantic Systems (I-SEMANTICS)*, 2012.

[92] A. E. Cano, A. Dadzie, and M. Hartmann. *Who's Who* - A Linked Data Visualisation Tool for Mobile Environments. In *Extended Semantic Web Conference (ESWC)*, 2011.

[93] I. Cantador and P. Castells. Group Recommender Systems: New Perspectives in the Social Web. In *Recommender Systems for the Social Web*. 2012.

[94] K. Chakrabarti, S. Chaudhuri, and S. Hwang. Automatic Categorization of Query Results. In *ACM Conference on Management of Data (SIGMOD)*, 2004.

[95] A. Chakravarthy, V. Lanfranchi, and F. Ciravegna. Cross-media document annotation and enrichment. In *Semantic Authoring and Annotation Workshop*, 2006.

[96] C. Y. Chan, P.-K. Eng, and K.-L. Tan. Stratified Computation of Skylines with Partially-Ordered Domains. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2005.

[97] C. Y. Chan, H. V. Jagadish, K.-L. Tan, A. K. H. Tung, and Z. Zhang. Finding k-dominant Skylines in High Dimensional Space. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2006.

[98] S. Chan, L. Xiao, J. Gerth, and P. Hanrahan. Maintaining interactivity while exploring massive time series. In *IEEE Symposium on Visual Analytics Science and Technology (VAST)*, 2008.

[99] Y.-C. Chang, L. D. Bergman, V. Castelli, C.-S. Li, M.-L. Lo, and J. R. Smith. The Onion Technique: Indexing for Linear Optimization Queries. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2000.

[100] D. L. Chao, J. Balthrop, and S. Forrest. Adaptive radio: achieving consensus using negative preferences. In *ACM Conference on Supporting Group Work*, 2005.

[101] A. Chebotko, S. Lub, and F. Fotouhib. Semantics preserving SPARQL-to-SQL translation. *Data & Knowl. Eng. (DKE)*, 68:10, 2009.

[102] B. Chen, X. Dong, D. Jiao, H. Wang, Q. Zhu, Y. Ding, and D. J. Wild. Chem2bio2rdf: a semantic framework for linking and data mining chemogenomic and systems chemical biology data. *BMC Bioinformatics*, 11, 2010.

[103] H. Chen, Z. Wu, H. Wang, and Y. Mao. RDF/RDFS-based Relational Database Integration. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2006.

[104] L. Chen and X. Lian. Efficient Processing of Metric Skyline Queries. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 21(3), 2009.

[105] Z. Chen and T. Li. Addressing diverse user preferences in SQL-query-result navigation. In *ACM Conference on Management of Data (SIGMOD)*, 2007.

[106] K.-H. Cheung, K. Y. Yip, A. K. Smith, R. de Knikker, A. Masiar, and M. Gerstein. Yeasthub: a semantic web use case for integrating data in life sciences domain. In *ISMB (Supplement of Bioinformatics)*, 2005.

[107] J. Chomicki. Preference formulas in relational queries. *ACM Transactions on Database Systems (TODS)*, 28(4), 2003.

[108] J. Chomicki, P. Godfrey, J. Gryz, and D. Liang. Skyline with Presorting. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2003.

[109] V. Christophides, G. Karvounarakis, I. Koffina, G. Kokkinidis, A. Magkanaraki, D. Plexousakis, G. Serfiotis, and V. Tannen. The ICS-FORTH SWIM: A Powerful Semantic Web Integration Middleware. In *Workshop on the Semantic Web, Ontologies and Databases (SWDB)*, 2003.

[110] V. Christophides, G. Karvounarakis, A. Magkanaraki, D. Plexousakis, and V. Tannen. The ICS-FORTH Semantic Web Integration Middleware (SWIM). *IEEE Data Eng. Bull.*, 26(4), 2003.

[111] W. W. Chu and K. Chiang. Abstraction of High Level Concepts from Numerical Values in Databases. In *AAAI Workshop on Knowledge Discovery in Databases*, 1994.

[112] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *World Wide Web Conference (WWW)*, 2004.

[113] O. Corby, L. Kefi-Khelif, H. Cherfi, F. Gandon, and K. Khelif. Querying the semantic web of data using SPARQL, RDF and XML. Technical report, INRIA, 2009.

[114] G. Correndo, M. Salvadores, I. Millard, and N. Shadbolt. Linked timelines: Temporal representation and management in linked data. In *Intl. Workshop on Consuming Linked Data (COLD)*, 2010.

[115] A. Crossen, J. Budzik, and K. J. Hammond. Flytrap: intelligent group music recommendation. In *Int. Conference on Intelligent User Interfaces*, 2002.

[116] C. Cruz and C. Nicolle. Ontology Enrichment and Automatic Population from XML Data. In *Intl. Workshop on Ontology-based Techniques*, 2008.

[117] I. Cruz, X. Huiyong, and F. Hsu. An Ontology-Based Framework for XML Semantic Integration. In *Intl. Database Engineering & Applications Symposium (IDEAS)*, 2004.

[118] I. Cruz, H. Xiao, and F. Hsu. Peer-to-peer semantic integration of XML and RDF data sources. In *Agents and Peer-to-Peer Computing Workshop (AP2PC)*, 2004.

[119] W. Cui, H. Zhou, H. Qu, P. C. Wong, and X. Li. Geometry-Based Edge Clustering for Graph Visualization. *IEEE Trans. Vis. Comput. Graph.*, 14(6), 2008.

[120] H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Annual Meeting of the Association for Computational Linguistics (ALC)*, 2002.

[121] C. D., editor. *Gleaning Resource Descriptions from Dialects of Languages*. W3C Rec., 2007. `www.w3.org/TR/grddl`.

[122] A. Dadzie, V. Lanfranchi, and D. Petrelli. Seeing is believing: Linking data with knowledge. *Information Visualization*, 8(3), 2009.

[123] A. Dadzie and M. Rowe. Approaches to visualising Linked Data: A survey. *Semantic Web*, 2(2), 2011.

[124] A. Dadzie, M. Rowe, and D. Petrelli. *Hide the Stack:* Toward Usable Linked Data. In *Extended Semantic Web Conference (ESWC)*, 2011.

[125] T. Dalamagas, N. Bikakis, G. Papastefanatos, Y. Stavrakas, and A. G. Hatzigeorgiou. Publishing life science data as linked open data: the case study of miRBase. In *Intl. Workshop on Open Data (WOD)*, 2012.

[126] H. V. de Sompel, R. Sanderson, M. L. Nelson, L. Balakireva, H. Shankar, and S. Ainsworth. An http-based versioning mechanism for linked data. In *Intl. Workshop on Linked Data on the Web, (LDOW)*, 2010.

[127] D. DeHaan, D. Toman, M. Consens, and T. Ozsu. A Comprehensive XQuery to SQL Translation using Dynamic Interval Encoding. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2003.

[128] L. Deligiannidis, K. Kochut, and A. P. Sheth. RDF data exploration and visualization. In *Workshop on CyberInfrastructure: Information Management in eScience*, 2007.

[129] D. V. Deursen, C. Poppe, G. Martens, E. Mannens, and R. V. Walle. XML to RDF conversion: a generic approach. In *AXMEDIS*, 2008.

[130] A. Deutsch and V. Tannen. Reformulation of XML Queries and Constraints. In *Intl. Conference on Database Theory (ICDT)*, 2003.

[131] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, K. S. McCurley, S. Rajagopalan, A. Tomkins, J. A. Tomlin, and J. Y. Zien. A Case for Automated Large-Scale Semantic Annotation. *Journal of Web Semantics*, 1(1), 2003.

[132] J. Dokulil and J. Katreniakova. Using Clusters in RDF Visualization. In *Advances in Semantic Processing*, 2009.

[133] P. R. Doshi, E. A. Rundensteiner, and M. O. Ward. Prefetching for Visual Data Exploration. In *Conference on Database Systems for Advanced Applications (DASFAA)*, 2003.

[134] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and Unsupervised Discretization of Continuous Features. In *Intl. Conference on Machine Learning*, 1995.

[135] M. Droop, M. Flarer, J. Groppe, S. Groppe, V. Linnemann, J. Pinggera, F. Santner, M. Schier, F. Schopf, H. Staffler, and S. Zugal. Translating XPath Queries into SPARQL Queries. In *OTM Workshops*, 2007.

[136] M. Droop, M. Flarer, J. Groppe, S. Groppe, V. Linnemann, J. Pinggera, F. Santner, M. Schier, F. Schopf, H. Staffler, and S. Zugal. Bringing the XML and Semantic Web Worlds Closer: Transforming XML into RDF and Embedding XPath into SPARQL. In *Intl. Conference on Enterprise Information Systems (ICEIS)*, 2008.

[137] M. Droop, M. Flarer, J. Groppe, S. Groppe, V. Linnemann, J. Pinggera, F. Santner, M. Schier, F. Schöpf, H. Staffler, and S. Zugal. Embedding XPATH Queries into SPARQL Queries. In *Intl. Conference on Enterprise Information Systems (ICEIS)*, 2008.

[138] M. Dudas, O. Zamazal, and V. Svatek. Roadmapping and Navigating in the Ontology Visualization Landscape. In *Conference on Knowledge Engineering and Knowledge Management (EKAW)*, 2014.

[139] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the Web. In *World Wide Web Conference (WWW)*, 2001.

[140] V. Eisenberg and Y. Kanza. D2rq/update: updating relational data via virtual RDF. In *World Wide Web Conference (WWW)*, 2012.

[141] M. Elahi, M. Ge, F. Ricci, D. Massimo, and S. Berkovsky. Interactive Food Recommendation for Groups. In *ACM Conference on Recommender Systems, RecSys*, 2014.

[142] A. Eldawy, M. Mokbel, and C. Jonathan. HadoopViz: A MapReduce Framework for Extensible Visualization of Big Spatial Data. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2016.

[143] B. Elliott, E. Cheng, C. Thomas-Ogbuji, and Z. M. Ozsoyoglu. A Complete Translation from SPARQL into Efficient SQL. In *Intl. Database Engineering & Applications Symposium (IDEAS)*, 2009.

[144] N. Elmqvist and J. Fekete. Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines. *IEEE Trans. Vis. Comput. Graph.*, 16(3), 2010.

[145] H. Eriksson. An annotation tool for semantic documents. In *Extended Semantic Web Conference (ESWC)*, 2007.

[146] I. Ermilov, M. Martin, J. Lehmann, and S. Auer. Linked Open Data Statistics: Collection and Exploitation. In *Knowledge Engineering and the Semantic Web*, 2013.

[147] O. Ersoy, C. Hurter, F. V. Paulovich, G. Cantareiro, and A. Telea. Skeleton-Based Edge Bundling for Graph Visualization. *IEEE Trans. Vis. Comput. Graph.*, 17(12), 2011.

[148] Z. F. Converting SPARQL to SQL. Technical report, 2006. `lists.w3.org/Archives/Public/public-rdf-dawg/2006OctDec/att-0058/sparql-to-sql.pdf`.

[149] R. Fagin, P. Kolaitis, R. Miller, and L. Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci. (TCS)*, 2005.

[150] R. Fagin, R. Kumar, and D. Sivakumar. Comparing Top k Lists. *SIAM J. Discrete Math.*, 17(1), 2003.

[151] S. Falconer, C. Callendar, and M.-A. Storey. A Visualization Service for the Semantic Web. In *Knowledge Engineering and Management by the Masses*. 2010.

[152] M. Farah and D. Vanderpooten. An Outranking Approach for Rank Aggregation in Information Retrieval. In *Intl. ACM Conference on Research and Development in Information Retrieval (SIGIR)*, 2007.

[153] J. Farrell and H. Lausen, editors. *Semantic Annotations for WSDL and XML Schema*. W3C Rec., 2007. `www.w3.org/TR/sawsdl`.

[154] M. Ferdinand, C. Zirpins, and D. Trastour. Lifting XML Schema to OWL. In *Intl. Conference on Web Engineering (ICWE)*, 2004.

[155] D. Fisher, I. O. Popov, S. M. Drucker, and m. c. schraefel. Trust me, i'm partially right: incremental visualization lets analysts explore large datasets faster. In *Conference on Human Factors in Computing Systems CHI*, 2012.

[156] E. A. Fox and J. A. Shaw. Combination of Multiple Searches. In *Text Retrieval Conference (TREC)*, 1993.

[157] B. Fu, N. F. Noy, and M.-A. Storey. Eye Tracking the User Experience - An Evaluation of Ontology Visualization Techniques. *Semantic Web Journal (to appear)*, 2015.

[158] Y. Fua, M. O. Ward, and E. A. Rundensteiner. Hierarchical parallel coordinates for exploration of large datasets. In *IEEE Visualization*, 1999.

[159] E. R. Gansner, Y. Hu, S. C. North, and C. E. Scheidegger. Multilevel agglomerative edge bundling for visualizing large graphs. In *IEEE Pacific Visualization Symposium (PacificVis)*, 2011.

[160] I. Garcia, L. Sebastia, and E. Onaindia. On the design of individual and group recommender systems for tourism. *Expert Syst. Appl.*, 38(6), 2011.

[161] R. Garcia and O. Celma. Semantic integration and retrieval of multimedia meta- data. In *Intl. Workshop on Knowledge Markup and Semantic Annotation*, 2005.

[162] S. Garcia, J. Luengo, J. A. Saez, V. Lopez, and F. Herrera. A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning. *IEEE Trans. Knowl. Data Eng.*, 25(4), 2013.

[163] M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, and K. Seada. Enhancing group recommendation by incorporating social relationship interactions. In *ACM Int. conference on Supporting group work, GROUP*, 2010.

[164] G. Giannopoulos, N. Bikakis, T. Dalamagas, and T. K. Sellis. GoNTogle: A Tool for Semantic Annotation and Search. In *Extended Semantic Web Conference (ESWC)*, 2010.

[165] F. Giunchiglia, U. Kharkevich, and I. Zaihrayeu. Concept search. In *Extended Semantic Web Conference (ESWC)*, 2009.

[166] P. Godfrey, J. Gryz, and P. Lasek. Interactive Visualization of Large Data Sets, 2015. Technical Report, York University.

[167] P. Godfrey, J. Gryz, P. Lasek, and N. Razavi. Visualization through inductive aggregation. In *Conference on Extending Database Technology (EDBT)*, 2016.

[168] P. Godfrey, R. Shipley, and J. Gryz. Algorithms and analyses for maximal vector computation. *The Intl. Journal on Very Large Data Bases (VLDBJ)*, 16(1), 2007.

[169] S. Groppe, J. Groppe, V. Linnemann, D. Kukulenzm, N. Hoeller, and C. Reinke. Embedding SPARQL into XQuery/XSLT. In *ACM Intl. Symposium on Applied Computing (SAC)*, 2008.

[170] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *ACM Conference on Management of Data (SIGMOD)*, 1984.

[171] F. Haag, S. Lohmann, S. Negru, and T. Ertl. OntoViBe: An Ontology Visualization Benchmark. In *Workshop on Visualizations and User Interfaces for Knowledge Engineering and Linked Data Analytics*, 2014.

[172] A. Halevy, Z. Ives, P. Mork, and I. Tatarinov. Piazza Data Management Infrastructure for Semantic Web Applications. In *World Wide Web Conference (WWW)*, 2003.

[173] A. Y. Halevy, Z. G. Ives, D. Suciu, and I. Tatarinov. Schema Mediation in Peer Data Management Systems. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2003.

[174] J. Han and Y. Fu. Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases. In *AAAI Workshop on Knowledge Discovery in Databases*, 1994.

[175] S. Handschuh and S. Staab, editors. *Annotation for the Semantic Web*. IOS Press, 2003.

[176] S. Handschuh, S. Staab, and F. Ciravegna. S-CREAM: Semi-automatic CREAtion of Metadata. In *Conference on Knowledge Engineering and Knowledge Management (EKAW)*, 2002.

[177] T. Hastrup, R. Cyganiak, and U. Bojars. Browsing Linked Data with Fenfire. In *World Wide Web Conference (WWW)*, 2008.

[178] J. Heer and S. Kandel. Interactive Analysis of Big Data. *ACM Crossroads*, 19(1), 2012.

[179] P. Heim, S. Lohmann, and T. Stegemann. Interactive Relationship Discovery via the Semantic Web. In *Extended Semantic Web Conference (ESWC)*, 2010.

[180] P. Heim, S. Lohmann, D. Tsendragchaa, and T. Ertl. SemLens: visual analysis of semantic data with scatter plots and semantic lenses. In *Conference on Semantic Systems (I-SEMANTICS)*, 2011.

[181] J. Helmich, J. Klimek, and M. Necasky. Visualizing RDF Data Cubes Using the Linked Data Visualization Model. In *Extended Semantic Web Conference (ESWC)*, 2014.

[182] N. Henry, J. Fekete, and M. J. McGuffin. NodeTrix: a Hybrid Visualization of Social Networks. *IEEE Trans. Vis. Comput. Graph.*, 13(6), 2007.

[183] M. Hert, G. Reif, and H. C. Gall. Updating relational data via sparql/update. In *EDBT/ICDT Workshops*, 2010.

[184] A. Hogue and D. Karger. Thresher: automating the unwrapping of semantic content from the World Wide Web. In *World Wide Web Conference (WWW)*, 2005.

[185] D. Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Trans. Vis. Comput. Graph.*, 12(5), 2006.

[186] W. Hop, S. de Ridder, F. Frasincar, and F. Hogenboom. Using Hierarchical Edge Bundles to visualize complex ontologies in GLOW. In *ACM Symposium on Applied Computing (SAC)*, 2012.

[187] V. Hristidis, N. Koudas, and Y. Papakonstantinou. PREFER: A System for the Efficient Execution of Multi-parametric Ranked Queries. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2001.

[188] A. Hussain, K. Latif, A. Rextin, A. Hayat, and M. Alam. Scalable Visualization of Semantic Nets using Power-Law Graphs. *AMIS*, 8(1), 2014.

[189] S. Idreos, O. Papaemmanouil, and S. Chaudhuri. Overview of Data Exploration Techniques. In *ACM Conference on Management of Data (SIGMOD)*, 2015.

[190] J. Im, F. G. Villegas, and M. J. McGuffin. VisReduce: Fast and responsive incremental information visualization of large datasets. In *IEEE Conference on Big Data*, 2013.

[191] Z. Ives, A. Halevy, P. Mork, and I. Tatarinov. Piazza: mediation and integration infrastructure for Semantic Web data. *J. Web Sem.*, 1:2, 2004.

[192] A. Jameson. More than the sum of its members: challenges for group recommender systems. In *Working conference on Advanced visual interfaces*, 2004.

[193] A. Jameson and B. Smyth. Recommendation to Groups. In *The Adaptive Web*, 2007.

[194] P. Jayachandran, K. Tunga, N. Kamat, and A. Nandi. Combining User Interaction, Speculative Query Execution and Sampling in the DICE System. *Proc. of the VLDB Endowment (PVLDB)*, 7(13), 2014.

[195] J. F. R. Jr., H. Tong, J. Pan, A. J. M. Traina, C. T. Jr., and C. Faloutsos. Large Graph Analysis in the GMine System. *IEEE Trans. Knowl. Data Eng.*, 25(1), 2013.

[196] J. F. R. Jr., H. Tong, A. J. M. Traina, C. Faloutsos, and J. Leskovec. GMine: A System for Scalable, Interactive Graph Visualization and Mining. In *Conference on Very Large Databases (VLDB)*.

[197] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl. Faster Visual Analytics through Pixel-Perfect Aggregation. *VLDB Endowment (PVLDB)*, 7(13), 2014.

[198] U. Jugel, Z. Jerzak, G. Hackenbroich, and V. Markl. VDDA: automatic visualization-driven data aggregation in relational databases. *Journal on Very Large Data Bases (VLDBJ)*, 2015.

[199] E. Kalampokis, A. Nikolov, P. Haase, R. Cyganiak, A. Stasiewicz, A. Karamanou, M. Zotou, D. Zeginis, E. Tambouris, and K. A. Tarabanis. Exploiting Linked Data Cubes with OpenCube Toolkit. In *Intl. Semantic Web Conference (ISWC)*, 2014.

[200] A. Kalinin, U. Çetintemel, and S. B. Zdonik. Interactive Data Exploration Using Semantic Windows. In *ACM Conference on Management of Data (SIGMOD)*, 2014.

[201] A. Kalinin, U. Çetintemel, and S. B. Zdonik. Searchlight: Enabling Integrated Search and Exploration over Large Multidimensional Data. *Proc. of the VLDB Endowment (PVLDB)*, 8(10), 2015.

[202] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi. Distributed and interactive cube exploration. In *IEEE Conference on Data Engineering (ICDE)*, 2014.

[203] B. Kampgen and A. Harth. OLAP4LD - A Framework for Building Analysis Applications Over Governmental Statistics. In *Extended Semantic Web Conference (ESWC)*, 2014.

[204] R. Kannan, M. Ishteva, and H. Park. Bounded matrix factorization for recommender system. *Knowl. Inf. Syst.*, 39(3), 2014.

[205] G. Kappel, E. Kapsammer, and W. Retschitzegger. Integrating XML and Relational Database Systems. *World Wide Web Journal (WWWJ)*, 7:4, 2004.

[206] G. Karypis and V. Kumar. Multilevel Graph Partitioning Schemes. In *Intl. Conference on Parallel Processing (ICPP)*, 1995.

[207] A. Kashyap, V. Hristidis, M. Petropoulos, and S. Tavoulari. Effective Navigation of Query Results Based on Concept Hierarchies. *IEEE Trans. Knowl. Data Eng.*, 23(4), 2011.

[208] A. Katifori, C. Halatsis, G. Lepouras, C. Vassilakis, and E. G. Giannopoulou. Ontology visualization methods - a survey. *ACM Comput. Surv.*, 39(4), 2007.

[209] J. Kay and W. Niu. Adapting Information Delivery to Groups of People. In *Workshop on New Technologies for Personalized Information Access*, 2005.

[210] H. A. Khan, M. A. Sharaf, and A. Albarrak. DivIDE: efficient diversification for interactive data exploration. In *Conference on Scientific and Statistical Database Management (SSDBM)*, 2014.

[211] W. Kießling. Foundations of Preferences in Database Systems. In *Intl. Conference on Very Large Databases (VLDB)*, 2002.

[212] A. Kim, E. Blais, A. G. Parameswaran, P. Indyk, S. Madden, and R. Rubinfeld. Rapid sampling for visualizations with ordering guarantees. *Proc. of the VLDB Endowment (PVLDB)*, 8(5), 2015.

[213] J. K. Kim, H. K. Kim, H. Y. Oh, and Y. U. Ryu. A group recommendation system for online communities. *Int. Journal of Information Management*, 30(3), 2010.

[214] A. Kiryakov, B. Popov, I. Terziev, D. Manov, and D. Ognyanoff. Semantic annotation, indexing, and retrieval. *Journal of Web Semantics*, 2(1), 2004.

[215] N. Kiyavitskaya, N. Zeni, J. R. Cordy, L. Mich, and J. Mylopoulos. Cerno: Light-weight tool support for semantic annotation of textual documents. *Data & Knowl. Eng. (DKE)*, 68(12), 2009.

[216] J. Klimek, J. Helmich, and M. Necasky. Payola: Collaborative Linked Data Analysis and Visualization Framework. In *Extended Semantic Web Conference (ESWC)*, 2013.

[217] I. Koffina, G. Serfiotis, V. Christophides, and V. Tannen. Mediating RDF/S Queries to Relational and XML Sources. *Intl. J. Semantic Web Inf. Syst. (IJSWIS)*, 2(4), 2006.

[218] D. Kossmann, F. Ramsak, and S. Rost. Shooting Stars in the Sky: An Online Algorithm for Skyline Queries. In *Intl. Conference on Very Large Databases (VLDB)*, 2002.

[219] G. Koutrika and Y. E. Ioannidis. Personalization of Queries in Database Systems. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2004.

[220] S. Kriglstein and R. Motschnig-Pitrik. Knoocks: New Visualization Approach for Ontologies. In *Conference on Information Visualisation*, 2008.

[221] K. Krishnamoorthy, R. Kumar, and S. R. Dua. Converting SPARQL Queries to SQL Queries. *Microsoft Corporation, U.S. Patent 7818352*, 2010.

[222] R. Krishnamurthy, R. Kaushik, and J. Naughton. XML-SQL Query Translation Literature: The State of the Art and Open Problems. In *Intl. XML Database Symposium (Xsym)*, 2003.

[223] S. Krivov, R. Williams, and F. Villa. GrOWL: A tool for visualization and editing of OWL ontologies. *J. Web Sem.*, 5(2), 2007.

[224] H. T. Kung, F. Luccio, and F. P. Preparata. On Finding the Maxima of a Set of Vectors. *Journal of ACM (JACM)*, 22(4), 1975.

[225] M. Lacroix and P. Lavency. Preferences: Putting More Knowledge into Queries. In *Intl. Conference on Very Large Databases (VLDB)*, 1987.

[226] A. Lambert, R. Bourqui, and D. Auber. Winding Roads: Routing edges into bundles. *Comput. Graph. Forum*, 29(3), 2010.

[227] M. Lanzenberger, J. Sampson, and M. Rester. Visualization in Ontology Tools. In *Intl. Conference on Complex, Intelligent and Software Intensive Systems (CISIS)*, 2009.

[228] J. Lee, G. won You, S. won Hwang, J. Selke, and W.-T. Balke. Interactive skyline queries. *Inf. Sci.*, 211, 2012.

[229] K. C. K. Lee, B. Zheng, H. Li, and W.-C. Lee. Approaching the Skyline in Z Order. In *Intl. Conference on Very Large Databases (VLDB)*, 2007.

[230] P. Lehti and P. Fankhauser. XML data integration with OWL: Experiences and challenges. In *Intl. Symposium on Applications and the Internet*, 2004.

[231] A. d. Leon, F. Wisniewki, B. Villazon-Terrazas, and O. Corcho. Map4rdf-Faceted Browser for Geospatial Datasets. In *Using Open Data: policy modeling, citizen empowerment, data journalism*, 2012.

[232] C. Li, G. Baciu, and Y. Wang. ModulGraph: Modularity-based Visualization of Massive Graphs. In *Visualization in High Performance Computing*, 2015.

[233] T. Liebig and O. Noppens. OntoTrack: A semantic approach for ontology authoring. *J. Web Sem.*, 3(2-3), 2005.

[234] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang. Selecting Stars: The k Most Representative Skyline Operator. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2007.

[235] Z. Lin, N. Cao, H. Tong, F. Wang, U. Kang, and D. H. P. Chau. Demonstrating Interactive Multi-resolution Large Graph Exploration. In *IEEE Conference on Data Mining Workshops*, 2013.

[236] L. D. Lins, J. T. Klosowski, and C. E. Scheidegger. Nanocubes for Real-Time Exploration of Spatiotemporal Datasets. *IEEE Trans. Vis. Comput. Graph.*, 19(12), 2013.

[237] B. Liu and C.-Y. Chan. ZINC: Efficient Indexing for Skyline Computation. *VLDB Endowment*, 4(3), 2010.

[238] C. Liu, M. Vincent, and J. Liu. Constraint Preserving Transformation from Relational Schema to XML Schema. *World Wide Web Journal (WWWJ)*, 9:1, 2006.

[239] Z. Liu, B. Jiang, and J. Heer. *imMens*: Real-time Visual Querying of Big Data. *Comput. Graph. Forum*, 32(3):421–430, 2013.

[240] C. Lofi and W.-T. Balke. On Skyline Queries and How to Choose from Pareto Sets. In *Advanced Query Processing (1)*. 2013.

[241] S. Lohmann, S. Negru, F. Haag, and T. Ertl. VOWL 2: User-Oriented Visualization of Ontologies. In *Conference on Knowledge Engineering and Knowledge Management (EKAW)*, 2014.

[242] S. Lohmann, S. Negru, F. Haag, and T. Ertl. Visualizing Ontologies with VOWL. *Semantic Web Journal*, 2015.

[243] H. Lu, C. S. Jensen, and Z. Zhang. Flexible and Efficient Resolution of Skyline Query Size Constraints. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 23(7), 2011.

[244] K. M. Interpreting XML via an RDF schema. In *Intl. Workshop on Electronicy Business Hubs*, 2002.

[245] L. M. Data Integration: A Theoretical Perspective. In *Symposium on Principles of Database Systems (PODS)*, 2002.

[246] L. Ma, C. Wang, J. Lu, F. Cao, Y. Pan, and Y. Yu. Effective and Efficient Semantic Web Data Management over DB2. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2008.

[247] C. Mangold. A survey and classification of semantic search approaches. *Int. J. Metadata Semantics and Ontology*, 2(1), 2007.

[248] F. Manola and M. E., editors. *RDF Primer*. W3C Rec., 2004. `www.w3.org/TR/rdf-primer`.

[249] S. Mansmann and M. H. Scholl. Exploring OLAP aggregates with hierarchical visualization techniques. In *ACM Symposium on Applied Computing (SAC)*, 2007.

[250] M. Maragkakis, M. Reczko, V. A. Simossis, P. Alexiou, G. L. Papadopoulos, T. Dalamagas, G. Giannopoulos, G. Goumas, E. Koukis, K. Kourtis, T. Vergoulis, N. Koziris, T. Sellis, P. Tsanakas, and A. G. Hatzigeorgiou. Diana-microt web server: elucidating microrna functions through target prediction. *Nucleic Acids Research*, 37(suppl 2), 2009.

[251] N. Marie and F. L. Gandon. Survey of Linked Data Based Exploration Systems. In *Workshop on Intelligent Exploration of Semantic Data (IESD)*, 2014.

[252] J. Masthoff. Group Modeling: Selecting a Sequence of Television Items to Suit a Group of Viewers. *User Model. User-Adapt. Interact.*, 14(1), 2004.

[253] J. Masthoff. Group Recommender Systems: Combining Individual Models. In *Recommender Systems Handbook*. 2011.

[254] J. F. McCarthy. Pocket Restaurant Finder: A situated recommender systems for groups. In *Workshop on Mobile Ad-Hoc Communication*, 2002.

[255] J. F. McCarthy and T. D. Anagnost. MusicFX: An Arbiter of Group Preferences for Computer Aupported Collaborative Workouts. In *ACM Conference on Computer Supported Cooperative Work*, 1998.

[256] K. McCarthy, L. McGinty, and B. Smyth. Case-Based Group Recommendation: Compromising for Success. In *Int. Conference on Case-Based Reasoning, ICCBR*, 2007.

[257] K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth, and P. Nixon. CATS: A Synchronous Approach to Collaborative Group Recommendation. In *Florida Artificial Intelligence Research Society Conference*, 2006.

[258] D. L. McGuinness and van Harmelen F., editors. *OWL Web Ontology Language: Overview*. W3C Rec., 2004. `www.w3.org/TR/owl-features`.

[259] R. J. Miller, L. M. Haas, and M. A. Hernández. Schema Mapping as Query Discovery. In *Intl. Conference on Very Large Databases (VLDB)*, 2000.

[260] T. M. Mitchell. *Machine Learning*. WCB/McGraw-Hill, 1997.

[261] M. H. Montague and J. A. Aslam. Condorcet Fusion for Improved Retrieval. In *Intl. Conference on Information and Knowledge Management*, 2002.

[262] M. D. Morse, J. M. Patel, and H. V. Jagadish. Efficient Skyline Computation over Low-Cardinality Domains. In *Intl. Conference on Very Large Databases (VLDB)*, 2007.

[263] K. Morton, M. Balazinska, D. Grossman, and J. D. Mackinlay. Support the Data Enthusiast: Challenges for Next-Generation Data-Analysis Systems. *VLDB Endowment (PVLDB)*, 7(6), 2014.

[264] E. Motta, P. Mulholland, S. Peroni, M. d'Aquin, J. M. Gomez-Perez, V. Mendez, and F. Zablith. A Novel Approach to Visualizing and Navigating Ontologies. In *Intl. Semantic Web Conference (ISWC)*, 2011.

[265] E. K. Neumann and D. Quan. Biodash: A semantic web dashboard for drug development. In *Pacific Symposium on Biocomputing*, 2006.

[266] E. Ntoutsi, K. Stefanidis, K. Nørvåg, and H.-P. Kriegel. Fast Group Recommendations by Applying User Clustering. In *Intl. Conference on Conceptual Modeling (ER)*, 2012.

[267] M. O'Connor, D. Cosley, J. A. Konstan, and J. Riedl. PolyLens: A recommender system for groups of user. In *European Conference on Computer Supported Cooperative Work, ECSCW*, 2001.

[268] D. P, P. M. Deshpande, D. Majumdar, and R. Krishnapuram. Efficient skyline retrieval with arbitrary similarity measures. In *Intl. Conference on Extending Database Technology (EDBT)*, 2009.

[269] D. Papadias, Y. Tao, G. Fu, and B. Seeger. Progressive skyline computation in database systems. *ACM Transactions on Database Systems (TODS)*, 30(1), 2005.

[270] M.-H. Park, H.-S. Park, and S.-B. Cho. Restaurant Recommendation for Group of People in Mobile Environments Using Probabilistic Multi-criteria Decision Making. In *Asia Pacific Conference on Computer Human Interaction*, 2008.

[271] Y. Park, M. J. Cafarella, and B. Mozafari. Visualization-Aware Sampling for Very Large Databases. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2016.

[272] J. Perez, M. Arenas, and C. Gutierrez. Semantics and Complexity of SPARQL. *ACM Transactions on Database Systems (TODS)*, 34:3, 2009.

[273] I. Petrou, M. Meimaris, and G. Papastefanatos. Towards a methodology for publishing Linked Open Statistical Data. *eJournal of eDemocracy & Open Government*, 6(1), 2014.

[274] D. Phan, L. Xiao, R. B. Yeh, P. Hanrahan, and T. Winograd. Flow Map Layout. In *IEEE Symposium on Information Visualization (InfoVis)*, 2005.

[275] F. Picalausa and S. Vansummeren. What are real SPARQL queries like? In *Intl. Workshop on Semantic Web Information Management*, 2011.

[276] E. Pietriga. IsaViz: a Visual Environment for Browsing and Authoring RDF Models. In *World Wide Web Conference (WWW)*, 2002.

[277] A. Piliponyte, F. Ricci, and J. Koschwitz. Sequential Music Recommendations for Groups by Balancing User Satisfaction. In *User Modeling, Adaptation, and Personalization*, 2013.

[278] S. Pizzutilo, B. De Carolis, G. Cozzolongo, and F. Ambruoso. Group Modeling in a Public Space: Methods, Techniques, Experiences. In *Int. Conference on Applied Informatics and Communications*, 2005.

[279] L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernández, and R. Fagin. Translating Web Data. In *Intl. Conference on Very Large Databases (VLDB)*, 2000.

[280] N. Popitsch and B. Haslhofer. Dsnotify: handling broken links in the web of data. In *World Wide Web Conference (WWW)*, 2010.

[281] E. Prud'hommeaux and S. A., editors. *SPARQL Query Language for RDF*. W3C Rec., 2008. `www.w3.org/TR/rdf-sparql-query`.

[282] C. R. A relational algebra for SPARQL. Technical report, Hewlett-Packard Laboratories, 2005. `www.hpl.hp.com/techreports/2005/HPL-2005-170.html`.

[283] L. Reeve and H. Han. Survey of semantic annotation platforms. ACM Symposium on Applied Computing (SAC), 2005.

[284] G. Reif, M. Jazayeri, and H. Gall. Towards semantic Web Engineering: WEESA-mapping XML schema to ontologies. In *Workshop on Application Design (WWW2004)*, 2004.

[285] W. H. Riker. *Liberalism Against Populism*. Waveland Press Inc, 1988.

[286] P. Ristoski and H. Paulheim. Visual Analysis of Statistical Data on Maps using Linked Open Data. In *Extended Semantic Web Conference (ESWC)*, 2015.

[287] J. Robie, D. Chamberlin, and et al., editors. *XQuery Update Facility 1.0" W3C Rec.* 2011. `www.w3.org/TR/xquery-update-10`.

[288] T. Rodrigues, P. Rosa, and J. Cardoso. Mapping XML to Exiting OWL ontologies. In *Intl. Conference WWW/Internet*, 2006.

[289] T. Rodrigues, P. Rosa, and J. Cardoso. Moving from syntactic to semantic organizations using JXML2OWL. *Computers in Industry*, 59:8, 2008.

[290] M. Rodriguez-Muro, J. Hardi, and D. Calvanese. Quest: Effcient SPARQL-to-SQL for RDF and OWL. In *Intl. Semantic Web Conference (ISWC)*, 2012.

[291] S. B. Roy, S. Amer-Yahia, A. Chawla, G. Das, and C. Yu. Space efficiency in group recommendation. *VLDB J.*, 19(6), 2010.

[292] B. S. Gloze: XML to RDF and back again. In *Jena User Conference*, 2006.

[293] D. Sacharidis, S. Papadopoulos, and D. Papadias. Topologically Sorted Skylines for Partially Ordered Domains. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2009.

[294] P. E. R. Salas, F. M. D. Mota, K. K. Breitman, M. A. Casanova, M. Martin, and S. Auer. Publishing Statistical Data on the Web. *Intl. J. Semantic Computing*, 6(4), 2012.

[295] A. D. Sarma, A. Lall, D. Nanongkai, and J. Xu. Randomized Multi-pass Streaming Skyline Algorithms. *VLDB Endowment*, 2(1), 2009.

[296] C. Sayers. Node-centric RDF Graph Visualization, 2004. Technical Report HP Laboratories.

[297] S. Schenk, P. Gearon, and P. A., editors. *SPARQL 1.1 Update*. W3C Rec., 2013. `www.w3.org/TR/sparql11-update`.

[298] K. Schlegel, T. Weißgerber, F. Stegmaier, C. Seifert, M. Granitzer, and H. Kosch. Balloon Synopsis: A Modern Node-Centric RDF Viewer and Browser for the Web. In *Extended Semantic Web Conference (ESWC)*, 2014.

[299] J. Sequeda, R. Depena, and D. Miranker. Ultrawrap: Using SQL Views for RDB2RDF. In *Intl. Semantic Web Conference (ISWC)*, 2009.

[300] H. Shang and M. Kitsuregawa. Skyline Operator on Anti-correlated Distributions. *VLDB Endowment*, 6(9), 2013.

[301] C. Shen and Y. Chen. A dynamic-programming algorithm for hierarchical discretization of continuous attributes. *European Journal of Operational Research*, 184(2), 2008.

[302] B. Shneiderman. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *IEEE Symposium on Visual Languages*, 1996.

[303] B. Shneiderman. Extreme visualization: squeezing a billion records into a million pixels. In *ACM Conference on Management of Data (SIGMOD)*, 2008.

[304] A. K. Smith, K.-H. Cheung, K. Y. Yip, M. H. Schultz, and M. Gerstein. Linkhub: a semantic web system that facilitates cross-database queries and information retrieval in proteomics. *BMC Bioinformatics*, 8(S-3), 2007.

[305] D. W. Sprague, F. Wu, and M. Tory. Music selection using the PartyVote democratic jukebox. In *Working Conference on Advanced Visual Interfaces*, 2008.

[306] C. Stadler, J. Lehmann, K. Hoffner, and S. Auer. LinkedGeoData: A core for a web of spatial open data. *Semantic Web*, 3(4), 2012.

[307] C. Stadler, M. Martin, and S. Auer. Exploring the web of spatial data with facete. In *World Wide Web Conference (WWW)*, 2014.

[308] I. Stavrakantonakis, C. Tsinaraki, N. Bikakis, N. Gioldasis, and S. Christodoulakis. SPARQL2XQuery 2.0: Supporting Semantic-based queries over XML data. In *Intl. Workshop on Semantic Media Adaptation and Personalization (SMAP)*, 2010.

[309] I. Stavrakantonakis, C. Tsinaraki, N. Bikakis, N. Gioldasis, and S. Christodoulakis. SPARQL2XQuery 2.0: Supporting Semantic-based Queries over XML Data. In *Workshop on Semantic Media Adaptation and Personalization (SMAP)*, 2010.

[310] K. Stefanidis, G. Koutrika, and E. Pitoura. A survey on representation, composition and application of preferences in database systems. *ACM Transactions on Database Systems (TODS)*, 36(3), 2011.

[311] C. Stolte, D. Tang, and P. Hanrahan. Query, analysis, and visualization of hierarchically structured data using Polaris. In *ACM Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2002.

[312] M. D. Storey, N. F. Noy, M. A. Musen, C. Best, R. W. Fergerson, and N. A. Ernst. Jambalaya: an interactive environment for exploring ontologies. In *IUI*, 2002.

[313] M. Stuhr, D. Roman, and D. Norheim. LODWheel - JavaScript-based Visualization of RDF Data. In *Workshop on Consuming Linked Data*, 2011.

[314] S. Sundara, M. Atre, V. Kolovski, S. Das, Z. Wu, E. I. Chong, and J. Srinivasan. Visualizing large-scale RDF data using Subsets, Summaries, and Sampling in Oracle. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2010.

[315] M. Tallis. SemanticWord processing for content authors. In *Knowledge Markup and Semantic Annotation Workshop*, 2003.

[316] K.-L. Tan, P.-K. Eng, and B. C. Ooi. Efficient Progressive Skyline Computation. In *Intl. Conference on Very Large Databases (VLDB)*, 2001.

[317] Y. Tao, L. Ding, X. Lin, and J. Pei. Distance-Based Representative Skyline. In *IEEE Intl. Conference on Data Engineering (ICDE)*, 2009.

[318] I. Tatarinov and A. Halevy. Efficient Query Reformulation in Peer Data Management Systems. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2004.

[319] F. Tauheed, T. Heinis, F. Schürmann, H. Markram, and A. Ailamaki. SCOUT: Prefetching for Latent Feature Following Queries. *Proc. of the VLDB Endowment (PVLDB)*, 5(11), 2012.

[320] A. D. Taylor. *Social choice and the mathematics of manipulation*. Cambridge University Press, 2005.

[321] K. Techapichetvanich and A. Datta. Interactive Visualization for OLAP. In *Computational Science and Its Applications (ICCSA)*, 2005.

[322] K. Thellmann, M. Galkin, F. Orlandi, and S. Auer. Linkdaviz - automatic binding of linked data to visualizations. In *Intl. Semantic Web Conference (ISWC)*, 2015.

[323] P. Thiran, F. Estievenart, J. L. Hainaut, and G. J. Houben. A Generic Framework for Extracting XML Data from Legacy Databases. *J. Web Eng. (JWE)*, 4:3, 2005.

[324] P. T. T. Thuy, Y. K. Lee, and S. Lee. DTD2OWL: automatic transforming XML documents into OWL ontology. In *Interaction Sciences Conference*, 2009.

[325] P. T. T. Thuy, Y. K. Lee, S. Lee, and B. S. Jeong. Transforming Valid XML Documents into RDF via RDF Schema. In *Intl. Conference on Next Generation Web Services Practices*, 2007.

[326] P. T. T. Thuy, Y. K. Lee, S. Lee, and B. S. Jeong. Exploiting XML Schema for Interpreting XML Documents as RDF. In *Intl. Conference on Services Computing*, 2008.

[327] C. Tominski, J. Abello, and H. Schumann. CGV - An interactive graph visualization system. *Computers & Graphics*, 33(6), 2009.

[328] G. Tschinkel, E. E. Veas, B. Mutlu, and V. Sabol. Using Semantics for Interactive Visual Analysis of Linked Open Data. In *Intl. Semantic Web Conference (ISWC)*, 2014.

[329] C. Tsinaraki and S. Christodoulakis. Interoperability of XML Schema Applications with OWL Domain Knowledge and Semantic Web Tools. In *Ontologies, DataBases, and Applications of Semantics (ODBASE)*, 2007.

[330] J. Umbrich, M. Hausenblas, A. Hogan, A. Polleres, and S. Decker. Towards dataset dynamics: Change frequency of linked open data sources. In *Intl. Workshop on Linked Data on the Web, (LDOW)*, 2010.

[331] J. Umbrich, B. Villazo'n-Terrazas, and M. Hausenblas. Dataset dynamics compendium: A comparative study. In *Intl. Workshop on Consuming Linked Data (COLD)*, 2010.

[332] J. Unbehauen, C. Stadler, and S. Auer. *Accessing Relational Data on the Web with SparqlMap*. 2012.

[333] V. S. Uren, P. Cimiano, J. Iria, S. Handschuh, M. Vargas-Vera, E. Motta, and F. Ciravegna. Semantic annotation for knowledge management: Requirements and a survey of the state of the art. *Journal of Web Semantics*, 4, 2006.

[334] F. Valsecchi, M. Abrate, C. Bacciu, M. Tesconi, and A. Marchetti. Dbpedia atlas: Mapping the uncharted lands of linked data. In *Workshop on Linked Data on the Web, LDOW*, 2015.

[335] F. Valsecchi and M. Ronchetti. Spacetime: a two dimensions search and visualisation engine based on linked data. In *Conference on Advances in Semantic Processing (SEMAPRO)*, 2014.

[336] M. Vargas-Vera, E. Motta, J. Domingue, M. Lanzoni, A. Stutt, and F. Ciravegna. MnM: Ontology Driven Semi-automatic and Automatic Support for Semantic Markup. In *Conference on Knowledge Engineering and Knowledge Management (EKAW)*, 2002.

[337] M. Vartak, S. Madden, A. G. Parameswaran, and N. Polyzotis. SEEDB: Automatically Generating Query Visualizations. *VLDB Endowment (PVLDB)*, 7(13), 2014.

[338] E. Vildjiounaite, V. Kyllönen, T. Hannula, and P. Alahuhta. Unobtrusive dynamic modelling of TV programme preferences in a Finnish household. *Multimedia Syst.*, 15(3), 2009.

[339] M. Voigt, S. Pietschmann, L. Grammel, and K. Meißner. Context-aware recommendation of visualization components. In *Conference on Information, Process, and Knowledge Management (eKNOW)*, 2012.

[340] M. Voigt, S. Pietschmann, and K. Meißner. A Semantics-Based, End-User-Centered Information Visualization Process for Semantic Web Data. In *Semantic Models for Adaptive Interactive Systems*. 2013.

[341] J. Volz, C. Bizer, M. Gaedke, and G. Kobilarov. Discovering and maintaining links on the web of data. In *Intl. Semantic Web Conference (ISWC)*, 2009.

[342] T. D. Wang and B. Parsia. CropCircles: Topology Sensitive Visualization of OWL Class Hierarchies. In *Intl. Semantic Web Conference (ISWC)*, 2006.

[343] M. O. Ward. Xmdvtool: Integrating multiple methods for visualizing multivariate data. In *IEEE Visualization*, 1994.

[344] H. Wickham. Bin-summarise-smooth: a framework for visualising large data. Technical report, 2013.

[345] R. C.-W. Wong, A. W.-C. Fu, J. Pei, Y. S. Ho, T. Wong, and Y. Liu. Efficient skyline querying with variable user preferences on nominal attributes. *VLDB Endowment*, 1(1), 2008.

[346] E. Wu, L. Battle, and S. R. Madden. The Case for Data Visualization Management Systems. *VLDB Endowment (PVLDB)*, 7(10), 2014.

[347] H. Xiao and I. Cruz. RDF-based metadata management in peer-to- peer systems. In *IST MMGPS Workshop*, 2004.

[348] H. Xiao and I. Cruz. Integrating and Exchanging XML Data Using Ontologies. *Journal on Data Semantics VI*, 2006.

[349] M. L. Yiu and N. Mamoulis. Efficient Processing of Top-k Dominating Queries on Multi-Dimensional Data. In *Intl. Conference on Very Large Databases (VLDB)*, 2007.

[350] C. Yu and L. Popa. Constraint-based XML Query Rewriting for Data Integration. In *ACM Intl. Conference on Management of Data (SIGMOD)*, 2004.

[351] H. Yu, C. Hsieh, S. Si, and I. S. Dhillon. Parallel matrix factorization for recommender systems. *Knowl. Inf. Syst.*, 41(3), 2014.

[352] Z. Yu, X. Zhou, Y. Hao, and J. Gu. TV Program Recommendation for Multiple Viewers Based on user Profile Merging. *User Model. User-Adapt. Interact.*, 16(1), 2006.

[353] K. Zhang, H. Wang, D. T. Tran, and Y. Yu. ZoomRDF: semantic fisheye zooming on RDF data. In *World Wide Web Conference (WWW)*, 2010.

[354] S. Zhang, N. Mamoulis, B. Kao, and D. W.-L. Cheung. Efficient Skyline Evaluation over Partially Ordered Domains. *VLDB Endowment*, 3(1), 2010.

[355] Y. Zhiwen, Z. Xingshe, and Z. Daqing. An adaptive in-vehicle multimedia recommender for group users. In *IEEE Vehicular Technology Conference*, 2005.

[356] M. Zinsmaier, U. Brandes, O. Deussen, and H. Strobelt. Interactive Level-of-Detail Rendering of Large Graphs. *IEEE Trans. Vis. Comput. Graph.*, 18(12), 2012.

[357] K. Zoumpatianos, S. Idreos, and T. Palpanas. Indexing for interactive exploration of big data series. In *ACM Conference on Management of Data (SIGMOD)*, 2014.