ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Υλοποίηση Μηχανισμού Κατασκευής και Βελτίωσης Αντιστοιχιών για Δίκτυα Ομότιμων Βάσεων Δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΔΗΜΟΣΘΕΝΗ Ν. ΜΠΟΥΣΟΥΝΗ

Επιβλέπων : Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2008

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

Υλοποίηση Μηχανισμού Κατασκευής και Βελτίωσης Αντιστοιχιών για Δίκτυα Ομότιμων Βάσεων Δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΔΗΜΟΣΘΕΝΗ Ν. ΜΠΟΥΣΟΥΝΗ

Επιβλέπων : Τιμολέων Σελλής
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 29η Οκτωβρίου 2008.

.....	.....	.....
Τιμολέων Σελλής	Ιωάννης Βασιλείου	Νεκτάριος Κοζύρης
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Οκτώβριος 2008

.....

ΔΗΜΟΣΘΕΝΗΣ ΜΠΟΥΣΟΥΝΗΣ

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

Περίληψη

Ένα από τα βασικότερα προβλήματα σε εφαρμογές ομότιμων βάσεων δεδομένων είναι η γρήγορη και ακριβής αντιστοίχιση δεδομένων που είναι εκφρασμένα σε εν γένει διαφορετικές μορφές, ο τρόπος δηλαδή που μπορεί να ξεπερασθεί το πρόβλημα της ετερογένειας δεδομένων. Στις περισσότερες περιπτώσεις, τα συστήματα ομότιμων ΒΔ χρησιμοποιούν δηλωτικά ερωτήματα – όψεις, που φανερώνουν τις αντιστοιχίες πινάκων των δύο σχημάτων σε conjunctive μορφή. Παρόλα αυτά κρίνεται απαραίτητη η ανάπτυξη μιας διαδικασίας (ημι)αυτόματης παραγωγής των αντιστοιχιών αυτών, ώστε η αντιστοίχιση δύο σχημάτων να γίνεται με τη μικρότερη δυνατή παρέμβαση του χρήστη.

Ο σκοπός της διπλωματικής αυτής εργασίας είναι η παρουσίαση και υλοποίηση ενός αλγορίθμου για την παραγωγή και σταδιακή βελτίωση αντιστοιχιών πινάκων (table mappings) μεταξύ δύο σχεσιακών σχημάτων. Υποβοηθούμενη από ένα automatic schema matching tool και από την ανάδραση του χρήστη, η διαδικασία αυτή, συνδυάζει με ένα σύστημα σταδιακής μάθησης απλές ισοδυναμίες ιδιοτήτων (element correspondences) και παράγει SQL ερωτήματα που αποτελούν αντιστοιχίες εκφρασμένες σε GAV/LAV μορφή.

Ο τρόπος που γίνεται αυτός ο συνδυασμός ισοδυναμιών, έγκειται στην εύρεση των επιπλέον περιορισμών που πρέπει να ενσωματωθούν στις δηλωτικές όψεις, ώστε η αντιστοιχία που θα παράγεται να είναι σημασιολογικά σωστή. Ιδιαίτερη έμφαση δόθηκε στους περιορισμούς συνδέσμων, η αναζήτηση των οποίων υλοποιήθηκε και ελέγχθηκε με δύο διαφορετικά ευριστικά κριτήρια.

Βασικό προτέρημα του μηχανισμού που σχεδιάσθηκε είναι η δυνατότητα λειτουργίας του, τόσο μεταξύ πλήρων σχημάτων εισόδου, όσο και μεταξύ σχημάτων που αποκαλύπτονται σταδιακά από εισερχόμενα ερωτήματα. Για το λόγο αυτό η διπλωματική αυτή εργασία μπορεί να αποτελέσει αφενός, ένα σύστημα παραγωγής και βελτίωσης αντιστοιχιών μεταξύ κόμβων ενός p2p συστήματος, όπου η δομή των σχημάτων γνωστοποιείται σταδιακά, αφετέρου ένα ξεχωριστό εργαλείο εύρεσης GAV/LAV αντιστοιχιών μεταξύ δύο σχεσιακών σχημάτων πάνω στο ίδιο αντικείμενο (domain specific schemas).

Έτσι λοιπόν, τα αποτελέσματα της διαδικασίας που αναπτύχθηκε, ελέγχθηκαν σε πλήρη σχήματα, προσομοιώνοντας, τη λειτουργία ενός ανεξάρτητου εργαλείου, αλλά παράλληλα σχεδιάσθηκε και υλοποιήθηκε η ενσωμάτωση του μηχανισμού, στο σύστημα GrouPeer, ένα σύστημα που συνδυάζει τεχνικές μετάφρασης ερωτημάτων πάνω από GAV/LAV αντιστοιχίες και μεθόδους βελτίωσης αντιστοιχιών, ώστε να ξεπεράσει το πρόβλημα της σταδιακής απαξίωσης ερωτημάτων σε δίκτυα ομότιμων βάσεων δεδομένων.

Λέξεις Κλειδιά: <<Δίκτυα ομότιμων βάσεων δεδομένων, Παραγωγή Αντιστοιχιών, Ισοδυναμίες ιδιοτήτων, Ετερογένεια Δεδομένων>>

Abstract

At the heart of many data-intensive applications is the problem of quickly and accurately transforming data into a new form, in other words the problem of finding a way to overcome the data heterogeneity problem. In most of these systems, researchers use conjunctive SQL queries that form the GAV/LAV mappings between the two different relational schemas. However, tools that are able to derive these mappings in an automatic or semi-automatic manner are of great importance, since this approach would alleviate the burden of manually constructing and maintaining semantically correct mappings between two domain-specific schemas.

The scope of this thesis is the presentation and the implementation of an algorithm that would produce and progressively upgrade the mappings between two relational schemas. Given a set of possible value correspondences and accumulating user feedback, the proposed mechanism, would be able to construct from scratch new SQL queries that would form the GAV/LAV mappings between the two schemas.

The effective combination of value correspondences in order to produce the mappings, is based upon a set of additional constraints, that should be incorporated in the conjunctive SQL view, thus revealing the correct semantic of each mapping. In this thesis, special emphasis was attributed to the discovery of the right join constraints, a complex task, which was performed based upon two specific heuristic criteria.

The most important advantage of the proposed mechanism is its suitability, not only for use as an independent mapping tool, between two completely known schemas, but also for use under peer to peer database environments, where the involved schemas are being revealed progressively by new user queries. Therefore, the results of the of the developed algorithm were evaluated from a quantitative and a qualitative perspective between known schemas, but on the other side, we designed and implemented the incorporation of the mechanism in the system GrouPeer, a system which combines automatic matching and traditional query rewriting techniques , in order to overcome the query degradation problem.

Keywords: <<Peer to Peer Database Systems, Schema Mapping Creation, Value Correspondences, Data Heterogeneity>>

Πίνακας περιεχομένων

Πίνακας Αλγορίθμων

Πίνακας Σχημάτων

1 *Εισαγωγή*

Ο συνεχώς αυξανόμενος όγκος της πληροφορίας στο σημερινό διαδίκτυο σε συνδυασμό με την ραγδαία εξάπλωση της χρήσης του Παγκόσμιου ιστού (World Wide Web) για την αναζήτηση δεδομένων, έχει φθάσει πλέον σε τέτοιο επίπεδο, ώστε το παραδοσιακό αρχιτεκτονικό μοντέλο Client Server να έχει ήδη αρχίσει να δείχνει την ηλικία του. Το γεγονός ότι ένα μεγάλο πλήθος αποδεκτών πληροφορίας (Clients) εξυπηρετείται από ένα αρκετά μικρότερο σύνολο κόμβων (Servers) οδηγεί ολοένα και περισσότερο σε προβλήματα συμφόρησης δικτύου, ώστε οι κατανεμημένες αρχιτεκτονικές σχεδίασης να θεωρούνται πλέον απαραίτητες. Σε αυτό το πρόβλημα έρχονται να δώσουν μια πολλά υποσχόμενη λύση τα Δίκτυα Ομότιμων Κόμβων (Peer to Peer Networks), τα οποία στέκονται στον αντίποδα του μοντέλου Πελάτης-Εξυπηρετητής. Σε ένα P2P δίκτυο, κάθε κόμβος συμπεριφέρεται ταυτόχρονα τόσο ως πελάτης, όσο και ως εξυπηρετητής, με αποτέλεσμα η κίνηση των δεδομένων να κατανέμεται ομοιόμορφα σε όλο το δίκτυο.

Σε αυτήν ακριβώς τη λογική κινούνται και τα P2P δίκτυα βάσεων δεδομένων. Κάθε κόμβος διατηρεί ένα δικό του σχήμα βάσης και δύναται να στέλνει ερωτήματα σε οποιονδήποτε άλλο κόμβο και όχι σε έναν συγκεκριμένο, που διατηρεί το σύνολο των πληροφοριών.

Δεδομένου όμως ότι, η αναπαράσταση και αποθήκευση της πληροφορίας γίνεται ανεξάρτητα από κόμβο σε κόμβο και δεν έχει παντού την ίδια μορφή, η απόλυτα κατανεμημένη αρχιτεκτονική των P2P δικτύων βάσεων δεδομένων δημιουργεί το σοβαρό πρόβλημα της έλλειψης μιας κοινής μορφής αναπαράστασης της πληροφορίας, με άλλα λόγια το πρόβλημα της ετερογένειας δεδομένων. Εφόσον ο τοπικός κόμβος δε γνωρίζει, τον τρόπο με τον οποίο έχουν οργανώσει την πληροφορία οι απομακρυσμένοι κόμβοι, τα ερωτήματα που στέλνει στο δίκτυο, και βασίζονται στη δική του βάση δεδομένων, ενδέχεται σταδιακά να απαξιώνονται, ή ακόμη να καθίσταται αδύνατη η απάντηση τους. Συνεπώς, κρίνεται αναγκαία η ανάπτυξη ορισμένων κανόνων που θα αντιστοιχίζουν τη μορφή δεδομένων του τοπικού κόμβου με τη

μορφή των δεδομένων απομακρυσμένων κόμβων, ώστε τα ερωτήματα να μπορούν να μεταφράζονται στο αντίστοιχο σχήμα.

Γνωρίζοντας όμως, ότι σε ένα P2P δίκτυο, ο αριθμός των κόμβων είναι εν γένει πολύ μεγάλος, κρίνεται εξαιρετικά χρονοβόρο και ασύμφορο, οι κανόνες αυτοί να τίθενται χειροκίνητα από τους χρήστες. Σε αυτό ακριβώς το πρόβλημα, η συγκεκριμένη διπλωματική εργασία προτείνει μια αποδοτική λύση, που θα αυτοματοποιεί την δημιουργία και σταδιακή βελτίωση των κανόνων αντιστοίχισης.

1.1 Αντικείμενο διπλωματικής εργασίας

Κύριο αντικείμενο, λοιπόν, της διπλωματικής αυτής εργασίας είναι ο σχεδιασμός και η υλοποίηση ενός μηχανισμού που θα παράγει αυτόματα και θα βελτιώνει σταδιακά (υποβοηθούμενη από την ανάδραση του χρήστη) κανόνες αντιστοίχισης μεταξύ δυο εν γένει διαφορετικών σχημάτων σχεσιακών βάσεων, που θα αναφέρονται όμως στον ίδιο θεματικό χώρο (domain specific schemas). Ο μηχανισμός αυτός, θα είναι αφενός σε θέση να ενσωματωθεί σε συστήματα ομότιμων βάσεων δεδομένων, αφετέρου δε, θα μπορεί να αποτελέσει ένα ξεχωριστό εργαλείο για την εύρεση αντιστοιχιών μεταξύ δύο σχημάτων.

Όπως αναφέρθηκε στην εισαγωγή, η ετερογένεια δεδομένων σε κατανεμημένα συστήματα, δύναται να ξεπερασθεί ορίζοντας κανόνες αντιστοίχισης μεταξύ των δεδομένων, οι οποίοι θα επιτρέπουν τη μετάφραση ερωτημάτων από ένα σχήμα σε κάποιο άλλο. Οι κανόνες αυτοί ονομάζονται αντιστοιχίες δεδομένων (data mappings) και τις περισσότερες φορές εκφράζονται σε μορφή όψεων μεταξύ των δύο πηγών δεδομένων (στην περίπτωση μας σχήματα σχεσιακών βάσεων). Κάθε τέτοια όψη αποτελείται σε γενικές γραμμές από δύο κύριες συνιστώσες:

- Ένα σύνολο από απλές ισοδυναμίες ιδιοτήτων μεταξύ των δύο σχημάτων
- Ένα σύνολο με επιπλέον περιορισμούς, που συνδυάζουν τις απλές ισοδυναμίες με τέτοιο τρόπο, ώστε η αντιστοιχία να έχει το σωστό σημασιολογικό νόημα

Οι περιορισμοί κάθε αντιστοιχίας ενδέχεται να είναι περιορισμοί συνδέσμων (join constraints), περιορισμοί τιμών ιδιοτήτων (value constraints) , ή περιορισμοί συγκρίσεων (comparison constraints).

Ενώ, ερευνητικά, έχουν προταθεί διάφορες αξιόλογες λύσεις για την αυτόματη εύρεση απλών ισοδυναμιών ιδιοτήτων, πρόβλημα γνωστό και ως Schema Matching Problem, ο συνδυασμός των ισοδυναμιών αυτών και η αυτόματη εύρεση όλων των απαραίτητων επιπλέον περιορισμών, ώστε να προκύψουν οι αντιστοιχίες, αποτελεί ακόμη ένα από τα βασικότερα ανοιχτά προβλήματα στον χώρο της Ετερογένειας Δεδομένων. Με άλλα λόγια, μας ενδιαφέρει η μετάβαση από το πρόβλημα Schema Matching στο πρόβλημα Schema Mapping.

Η σημαντικότερη δυσκολία της μετάβασης αυτής, έγκειται στο γεγονός ότι η σημασιολογία κάθε ισοδυναμίας ιδιοτήτων απαιτεί εν γένει διαφορετικούς περιορισμούς, ανάλογα με την οργάνωση και δομή των δεδομένων στα δύο σχήματα. Κατ΄επέκταση, οποιαδήποτε προσέγγιση στο πρόβλημα, πρέπει να βασίζεται στην εύρεση των κατάλληλων περιορισμών, προκειμένου οι ισοδυναμίες ιδιοτήτων να συνδυάζονται με τέτοιο τρόπο ώστε να δίνουν την σημασιολογία της αντιστοιχίας που περιμένει ο χρήστης. Η αυτόματη ανακάλυψη περιορισμών τιμών και συγκρίσεων, απαιτεί προχωρημένες τεχνικές που εμπλέκουν χρήση οντολογιών και ανάλυσης δεδομένων και δε θα μας απασχολήσει σε αυτήν την εργασία. Από την άλλη μεριά όμως, η εύρεση των κατάλληλων περιορισμών συνδέσμων μπορεί να πραγματοποιηθεί, βάσει των πληροφοριών που κρύβουν τα metadata κάθε σχήματος (πχ, περιορισμοί εξωτερικών και κύριων κλειδιών) και για το λόγο αυτό, αποτελεί το σημαντικότερο χαρακτηριστικό του μηχανισμού που αναπτύχθηκε. Δεδομένου ότι σε κάθε σχήμα, τα metadata του, δημιουργούν πολλούς πιθανούς περιορισμούς συνδέσμων για ένα συγκεκριμένο σύνολο ισοδυναμιών ιδιοτήτων, το πρόβλημα ανάγεται σε πρόβλημα αναζήτησης ενός χώρου, ο οποίος μάλιστα αυξάνει εκθετικά με την πολυπλοκότητα του σχήματος. Στα πλαίσια λοιπόν αυτής της αναζήτησης, καθορίζουμε δύο ευριστικά κριτήρια για την επιλογή των κατάλληλων περιορισμών, βάσει των οποίων δημιουργούνται και βελτιώνονται σταδιακά οι αντιστοιχίες.

Ένα δεύτερο βασικό σημείο της εργασίας συνίσταται στην ανάπτυξη μιας μορφής επικοινωνίας του μηχανισμού με τον χρήστη, με την οποία ο τελευταίος θα μπορεί να προσφέρει ανάδραση στο σύστημα, συνεισφέροντας έτσι τόσο στην αποφυγή αστοχιών, όσο και στην ταχύτερη σύγκλιση προς την σωστή κατεύθυνση αναζήτησης. Το σύστημα θα πρέπει να ζητά πληροφορίες από τον χρήστη, που να βασίζονται αποκλειστικά στο αποτέλεσμα του μηχανισμού, και όχι στην διαδικασία βάσει της οποίας, προκύπτει αυτό το αποτέλεσμα. Έτσι λοιπόν, καθορίστηκαν οι χαρακτηρισμοί εκείνοι, που ο χρήστης θα μπορεί να προσδώσει σε τμήματα της αντιστοιχίας, αλλά και ο τρόπος που αντιδρά το σύστημα σε κάθε τέτοιο χαρακτηρισμό.

Τέλος, σχεδιάσθηκε η ενσωμάτωση του μηχανισμού παραγωγής και βελτίωσης αντιστοιχιών στο σύστημα GrouPeer, ώστε να επιτυγχάνεται η ανακάλυψη κόμβων, που δύνανται να αναπτύξουν ισχυρές σημασιολογικά αντιστοιχίες μεταξύ τους, και κατά συνέπεια να γειτονεύσουν, δίνοντας έτσι μια πολλά υποσχόμενη λύση στο πρόβλημα της σταδιακής απαξίωσης ερωτημάτων, ένα σύνηθες φαινόμενο σε δίκτυα ομότιμων βάσεων δεδομένων. Το βασικότερο ίσως σημείο της ενσωμάτωσης αυτής, είναι η τροποποίηση του μηχανισμού, έτσι ώστε να παράγει αντιστοιχίες μεταξύ σχημάτων που χτίζονται σταδιακά και δεν είναι γνωστά από την αρχή, μιας και σε πλήρως κατανεμημένες αρχιτεκτονικές, τα σχήματα των

απομακρυσμένων κόμβων γνωστοποιούνται μόνο από τα εισερχόμενα ερωτήματα των χρηστών.

1.1.1 Συνεισφορά

Η συνεισφορά της εργασίας αυτής συνοψίζεται στα παρακάτω σημεία:

1. Μελετήθηκε ο τρόπο κατασκευής και βελτίωσης των αντιστοιχιών από τρία άλλα ερευνητικά συστήματα (Clio, Tomas, Muse).
2. Σχεδιάσθηκε και υλοποιήθηκε ένα νέο σύστημα εύρεσης και βελτίωσης αντιστοιχιών για δύο εν γένει διαφορετικά σχήματα, το οποίο άρει την προϋπόθεση τα δύο σχήματα να είναι πλήρως γνωστά από την αρχή, και για το λόγο αυτό καθίσταται ιδανικό για συστήματα ομότιμων βάσεων δεδομένων.
3. Υλοποιήθηκαν δυο διαφορετικοί αλγόριθμοι για την κατασκευή και βελτίωση των αντιστοιχιών, των οποίων η επίδοση αξιολογήθηκε ποσοτικά.
4. Κατασκευάστηκε μια πλήρης σουίτα πειραμάτων του μηχανισμού, η οποία επιτρέπει τον αναλυτικό έλεγχο της διαδικασίας σε διάφορες περιπτώσεις (test cases).
5. Σχεδιάσθηκε πλήρως και υλοποιήθηκε στο μεγαλύτερο μέρος της, η ενσωμάτωση του όλου μηχανισμού στο σύστημα ερωταποκρίσεων σε P2P βάσεις δεδομένων GrouPeer,
6. Υλοποιήθηκε το Peer to Peer layer του συστήματος GrouPeer.

1.2 Οργάνωση τόμου

Η οργάνωση του τόμου έχει ως εξής:

Στο κεφάλαιο 2 παρουσιάζουμε διάφορες σχετικές εργασίες που κινούνται στο ίδιο ερευνητικό πεδίο με την παρούσα εργασία. Το κεφάλαιο 3 συνιστά το θεωρητικό υπόβαθρο αυτής της εργασίας. Ξεκινά με μια γενική περιγραφή για το δίκτυα ομότιμων βάσεων δεδομένων και στη συνέχεια αναλύει τις βασικότερες έννοιες, που θα συναντήσει ο αναγνώστης στη συνέχεια. Στο κεφάλαιο 4 δίνεται η λεπτομερής ανάλυση του συστήματος που αναπτύχθηκε, αρχικά παρέχοντας μια γενική εικόνα για την αρχιτεκτονική του συστήματος και ύστερα εστιάζοντας στις ξεχωριστές οντότητες που αποτελούν τον μηχανισμό. Στο κεφάλαιο 5 παρουσιάζεται ο τρόπος που σχεδιάσθηκε η υλοποίηση του συστήματος, μαζί με την περιγραφή των σημαντικότερων κλάσεων, ενώ στο κεφάλαιο 6, μελετούμε την πειραματική αξιολόγηση και τον έλεγχο της αποδοτικότητας του μηχανισμού. Το έβδομο κεφάλαιο αποτελεί τον επίλογο της εργασίας, όπου συνοψίζονται τα όσα επιτεύχθηκαν και δίνονται γενικές κατευθύνσεις και ιδέες για περαιτέρω επεκτάσεις. Στο κεφάλαιο 8 αναφέρουμε την βιβλιογραφία, πάνω στην οποία βασίστηκε η εργασία.

2 *Σχετικές εργασίες*

Η αυτοματοποίηση της δημιουργίας αντιστοιχιών μεταξύ δύο σχημάτων (Schema Mapping Problem) αποτελεί ένα σημαντικό πρόβλημα στον γενικότερο ερευνητικό χώρο της ετερογένειας δεδομένων. Διάφορες εργασίες, έχουν ήδη προτείνει μεθόδους που προσεγγίζουν το πρόβλημα, και πολλές από αυτές χρησιμοποιούνται ήδη σε εμπορικές και ερευνητικές εφαρμογές. Τα βασικότερα συστήματα που έχουν γίνει γνωστά πάνω σε αυτόν τον τομέα είναι τα εξής:

2.1 Το σύστημα Clio

Το εργαλείο Clio κατασκευάστηκε από την εταιρεία IBM Research Almaden και αποσκοπεί σε μια ημιαυτόματη (υποβοηθούμενη από το χρήστη) δημιουργία αντιστοιχιών μεταξύ ενός σχήματος στόχου (target schema) και ενός νέου σχήματος πηγής δεδομένων (source schema). Τα σχήματα εισόδου δύνανται να είναι εκφρασμένα τόσο σε σχεσιακή όσο και σε XML μορφή. Σε γενικές γραμμές, το εργαλείο αποτελείται από ένα σύνολο Αναγνωστών Σχημάτων (Schema Readers), που διαβάζουν ένα σχήμα και το μεταφράζουν σε μια εσωτερική αναπαράσταση, μια Μηχανή Ανταπόκρισης (Correspondence Engine – CE), που χρησιμοποιείται για να ανιχνευθούν συσχετιζόμενα τμήματα των σχημάτων ή των βάσεων δεδομένων, και μια Γεννήτρια Αντιστοιχιών (Mapping Generator), που παράγει περιγραφές όψεων για να συσχετιστούν τα δεδομένα του σχήματος πηγής στα δεδομένα του σχήματος στόχου. Η μηχανή ανταπόκρισης χρησιμοποιεί σύνθετες ισοδυναμίες ιδιοτήτων (high level matchings), οι οποίες παρέχονται από τον χρήστη σύμφωνα με το σημασιολογικό ταίριασμα των δύο σχημάτων και δημιουργεί ένα σύνολο από όλες τις πιθανές αντιστοιχίες πινάκων, όπως αυτές προκύπτουν, από τους επιπλέον περιορισμούς των δύο σχημάτων (low level mappings δηλαδή queries). Μέσω της κατάλληλης γραφικής διαπροσωπείας, ο χρήστης

μπορεί να απορρίψει μερικές από αυτές τις αντιστοιχίες, βοηθώντας έτσι στο να δημιουργηθούν σταδιακά οι ισχυρότερες σημασιολογικά αντιστοιχίες.

2.2 Το σύστημα Tomas

Ως συνέχεια πάνω στην εργασία του συστήματος Clio, το πανεπιστήμιο του Toronto, ανέπτυξε το σύστημα Tomas, το οποίο δεν έχει σκοπό πλέον να παράγει νέες αντιστοιχίες εκ του μηδενός, αλλά να διαχειρίζεται ήδη ανακαλυφθείσες αντιστοιχίες μεταξύ δύο σχημάτων, καθώς τα σχήματα εισόδου αλλάζουν. Έτσι λοιπόν, το Tomas παρουσιάζει έναν αλγόριθμο που προσαρμόζει αυτόματα τις αντιστοιχίες, επιφέροντας τις απαραίτητες αλλαγές, καθώς στα σχήματα προσθέτονται/αφαιρούνται νέοι πίνακες, ιδιότητες, περιορισμοί εξωτερικών και κύριων κλειδιών κ.λ.π. Και εδώ, οι παραπάνω λειτουργίες προσφέρονται τόσο για σχεσιακά όσο και για XML σχήματα, ενώ ο χρήστης συμμετέχει πάλι ενεργά, κατευθύνοντας, με την ανάδραση του, το σύστημα προς σημασιολογικά σωστές τροποποιήσεις αντιστοιχιών.

2.3 Το σύστημα Muse

Μια διαφορετική προσέγγιση σχετικά με την αυτόματη ανακάλυψη και σταδιακή βελτίωση αντιστοιχιών, προτείνει το σύστημα Muse. Στην εργασία αυτή, το σύστημα βοηθά τον χρήστη να κατανοήσει και να βελτιώσει τη σημασιολογία των αντιστοιχιών μεταξύ δύο σχημάτων προς τη σωστή κατεύθυνση, αυτή τη φορά με αντιπροσωπευτικά δείγματα δεδομένων, που φανερώνουν την αντιστοιχία. Ο χρήστης, δηλαδή, δεν κρίνει απευθείας τις μεταφράσεις στοιχείων σχήματος (πινάκων, ή ιδιοτήτων) αλλά παρέχει έμμεσα μια βοήθεια, κρίνοντας τη σημασιολογία των δεδομένων που του παρουσιάστηκαν. Η συγκεκριμένη προσέγγιση, βελτιώνοντας δηλαδή τις αντιστοιχίες, μέσω από παραδείγματα δεδομένων, και στη συνέχεια ανακαλύπτοντας μοτίβα στα δεδομένα, δύναται να οδηγήσει σε πολύ αποδοτικές μεθόδους ανακάλυψης σύνθετων και πολύπλοκών αντιστοιχιών, χωρίς το εκάστοτε σύστημα να απαιτεί από τον χρήστη να γνωρίζει τον ακριβή τρόπο αναπαράσταση των αντιστοιχιών αυτών.

3 *Θεωρητικό υπόβαθρο*

Στο συγκεκριμένο κεφάλαιο, αναλύονται λεπτομερώς όλα τα βασικά θεωρητικά σημεία, πάνω στα οποία στηρίζεται η διπλωματική εργασία. Αρχικά γίνεται μια γενική παρουσίαση για τη σημασία και λειτουργία των P2P βάσεων δεδομένων, ενώ στη συνέχεια αναφέρονται και εξηγούνται και με παραδείγματα όροι και έννοιες, που είναι απαραίτητες για την κατανόηση της λειτουργίας του μηχανισμού που αναπτύχθηκε.

3.1 Peer To Peer Βάσεις Δεδομένων

Όπως αναφέρθηκε και στην εισαγωγή τα peer to peer δίκτυα εμφανίστηκαν ως απάντηση στο πρόβλημα της συμφόρησης δικτύου που δημιουργούσε το παραδοσιακό μοντέλο αρχιτεκτονικής Client – Server. Εν γένει, ένα p2p δίκτυο αποτελεί ένα δίκτυο Η/Υ, στο οποίο οι κόμβοι είναι πλήρως ισότιμοι και συμπεριφέρονται τόσο ως αποστολείς όσο και ως αποδέκτες πληροφορίας, με άλλα λόγια εμφανίζουν ταυτόχρονα και εξίσου, ιδιότητες και πελάτη και εξυπηρετητή. Εκτός από την αποσυμφόρηση της κίνησης του δικτύου, ένα δεύτερο πλεονέκτημα που εμφανίζουν τα p2p συστήματα, είναι η τρομακτικά μεγάλη αποθηκευτική και υπολογιστική τους δύναμη, μιας και νέοι κόμβοι μπορούν να εισέλθουν στο δίκτυο οποιαδήποτε στιγμή και να συνεισφέρουν στις συνολικές δυνατότητες του συστήματος. Για τους παραπάνω λόγους, η P2P αρχιτεκτονική έχει αρχίσει να παίζει πρωτεύοντα ρόλο σε πολλές εφαρμογές του διαδικτύου (File Sharing, Instant Messaging, Telephony, Media Streaming, Discussion Forums), μια εκ των οποίων είναι η ανάπτυξη Peer to Peer βάσεων δεδομένων (Data Sharing).

Σε ένα σύστημα Ομότιμων βάσεων δεδομένων, κάθε κόμβος διατηρεί ένα δικό του τοπικό σχήμα βάσης(local schema) και συνδέεται σε ένα p2p δίκτυο από κόμβους, που και αυτοί αποθηκεύουν στις βάσεις τους πληροφορίες για το ίδιο αντικείμενο με αυτόν. Κάθε κόμβος πλέον δύναται να στέλνει ερωτήματα όχι μόνο στη δική του τοπική βάση, αλλά σε

ολόκληρο το δίκτυο. Τα ερωτήματα λαμβάνονται από τους απομακρυσμένους κόμβους, μεταφράζονται στο δικό τους τοπικό σχήμα, και τα αποτελέσματα στέλνονται πάλι πίσω στον χρήστη που εκκίνησε την ερώτηση. Έτσι λοιπόν, κάθε κόμβος έχει πρόσβαση σε ένα τεράστιο όγκο δεδομένων, πράγμα που είναι φανερώνει ξεκάθαρα την αποθηκευτική δύναμη των p2p δικτύων.

Το σημαντικότερο πρόβλημα που δημιουργείται στο παραπάνω σενάριο εντοπίζεται στην μετάφραση των ερωτημάτων που στέλνει ένας κόμβος, προκειμένου να απαντηθούν από τους απομακρυσμένους κόμβους. Δεδομένου ότι ο αποστολέας της ερώτησης έχει πλήρη άγνοια για τον τρόπο που είναι σχεδιασμένα τα σχήματα των απομακρυσμένων κόμβων, στέλνει το ερώτημα του εκφρασμένο πάνω στο δικό του σχήμα. Έτσι κάθε κόμβος που λαμβάνει αυτό το ερώτημα, οφείλει με κάποιο τρόπο να το μεταφράσει, ώστε να μπορέσει να εξάγει τα αντίστοιχα αποτελέσματα.

Ένας κλασσικός τρόπος για να γίνει αυτή η μετάφραση του ερωτήματος προέρχεται από τα συστήματα data integration. Η ιδέα είναι η εξής:

Προτού ένας κόμβος στείλει ένα ερώτημα στο δίκτυο, αυτό μεταφράζεται από το δικό του τοπικό σχήμα, σε ένα ενδιάμεσο σχήμα (mediated schema) γνωστό σε όλους τους κόμβους. Ύστερα κάθε απομακρυσμένος κόμβος που λαμβάνει αυτό το μεταφρασμένο ερώτημα, το ανάγει στο δικό του σχήμα, μεταφράζοντας το για δεύτερη φορά, και πλέον είναι σε θέση να το απαντήσει. Η μετάφραση από και προς το ενδιάμεσο σχήμα, γίνεται με τη χρήση αντιστοιχιών (mappings), διαφόρων κανόνων δηλαδή, που αντιστοιχούν έννοιες του ενός σχήματος σε έννοιες κάποιου άλλου.

Ο τρόπος αυτής της μετάφρασης, αν και ευρέως διαδεδομένος, έρχεται σε αντίθεση με το βασικό πλεονέκτημα των p2p δικτύων, μιας και με την εισαγωγή του ενδιάμεσου σχήματος, δε μπορούμε πλέον να μιλάμε για πλήρως αποκεντρωμένη αρχιτεκτονική και συνεπώς λιγότερη συμφόρηση.

Για το λόγο αυτό σε καθαρά p2p δίκτυα βάσεων δεδομένων η μετάφραση των ερωτημάτων γίνεται διαφορετικά. Αντί να υπάρχει ένα κεντρικό ενδιάμεσο σχήμα με το οποίο όλοι οι κόμβοι διατηρούν αντιστοιχίες, κάθε κόμβος διατηρεί αντιστοιχίες απευθείας με τα σχήματα ορισμένων άλλων κόμβων, τους γείτονές του. Έτσι λοιπόν, όταν ο αποστολέας θέλει να στείλει ένα ερώτημα στο δίκτυο, το στέλνει μόνο στους γείτονες του, οι οποίοι μπορούν να το μεταφράσουν, εφόσον διαθέτουν τις κατάλληλες αντιστοιχίες. Στη συνέχεια, οι γείτονες στέλνουν το μεταφρασμένο ερώτημα στους δικούς τους γείτονες, οι οποίοι τώρα θα μεταφράσουν εκ νέου, το ήδη μεταφρασμένο ερώτημα. Η όλη διαδικασία συνεχίζεται, με αποτέλεσμα το αρχικό ερώτημα να φθάνει σε όλους τους κόμβους του δικτύου. Όταν κάποιος νέος κόμβος θέλει να εισέλθει στο δίκτυο, δημιουργεί αντιστοιχίες με ορισμένους τυχαίους

κόμβους που στο εξής θα αποτελούν τους γείτονές του και εφεξής, λαμβάνει και αυτός μέρος στην όλη διαδικασία αποστολής και απάντησης ερωτημάτων.

3.2 GAV, LAV & GLAV αντιστοιχίες

3.2.1. Η έννοια της αντιστοιχίας

Όπως είδαμε παραπάνω, η μετάφραση ερωτημάτων σε p2p συστήματα βάσεων δεδομένων, τόσο σε καθαρά p2p δίκτυα, όσο και σε data integration συστήματα, βασίζεται σε αντιστοιχίες (mappings) μεταξύ των σχημάτων. Γενικά ο ρόλος μιας αντιστοιχίας ανάμεσα σε δύο σχήματα είναι να αντιστοιχεί ορισμένα γνωρίσματα του ενός σχήματος σε εκείνα τα γνωρίσματα του άλλου σχήματος, που έχουν την ίδια σημασιολογική σημασία, αλλά πιθανώς να είναι εκφρασμένα διαφορετικά.

Έστω για παράδειγμα ότι ένα σχήμα στον τομέα της εκπαίδευσης αποθηκεύει τους καθηγητές σε έναν πίνακα Professor(id, name, address), ενώ ένα δεύτερο σχήμα (πιθανώς κάποιου άλλου πανεπιστημίου) αποθηκεύει όλους τους ακαδημαϊκούς του πανεπιστημίου στον πίνακα Personnel(id,surname,contact,position).

Προφανώς στο δεύτερο σχήμα ένα υποσύνολο των πλειάδων του πίνακα Personnel είναι οι καθηγητές του πανεπιστημίου (συγκεκριμένα οι πλειάδες εκείνες για τις οποίες ισχύει Personnel.position = “Professor”). Θέλουμε τώρα να κατασκευάσουμε μια αντιστοιχία, με την οποία θα αντιστοιχίζονται οι καθηγητές του πρώτου σχήματος, στους καθηγητές του δεύτερου σχήματος, με άλλα λόγια θέλουμε να βρούμε εκείνα τα δεδομένα των δύο σχημάτων που αποτελούν την οντότητα καθηγητές, και να τα συνδέσουμε μεταξύ τους. Στο πρώτο σχήμα οι καθηγητές είναι όλα τα δεδομένα της σχέσης Professor, ενώ στο δεύτερο είναι εκείνο το υποσύνολο της σχέσης Personnel, για το οποίο ισχύει ο περιορισμός Personnel.position = “Professor”. Η σύνδεση των δεδομένων σε αυτή την περίπτωση μπορεί να γίνει ορίζοντας μια όψη στο δεύτερο σχήμα, η οποία θα είναι καθόλα ισοδύναμη με τον πίνακα Professor. Πράγματι, η όψη του σχήματος 3 a είναι σημασιολογικά ίδια με τον πίνακα Professor του άλλου σχήματος, και συνεπώς αποτελεί μια αντιστοιχία.

create view Professor (id, name, address) as

Select *id as id, surname as name, contact as address*

From *Personnel*

Where *position = “Professor”*

Σχήμα 3 a: Παράδειγμα μιας απλής αντιστοιχίας

3.2.2. Αναπαράσταση όψεων με τη γλώσσα Datalog

Προτού αναλύσουμε τις διάφορες κατηγορίες αντιστοιχιών σχημάτων που χρησιμοποιούνται στα κατανεμημένα συστήματα βάσεων δεδομένων, κρίνεται αναγκαία μια μικρή εισαγωγή στη γλώσσα Datalog, που θα αποτελεί στο εξής το μέσο με το οποίο θα αναπαριστούμε της αντιστοιχίες. Η γλώσσα Datalog είναι μια γλώσσα ερωτημάτων σε βάσεις δεδομένων, βασισμένη σε κανόνες , (συντακτικά) υποσύνολο της γλώσσας Prolog, και θεωρείται πλέον κατάλληλη για αναπαράσταση συζευκτικών ερωτημάτων(conjunctive queries).

Όπως είδαμε στο προηγούμενο κεφάλαιο, οι αντιστοιχίες στις σχεσιακές βάσεις, δεν είναι παρά όψεις, δηλαδή ερωτήματα. Στη συγκεκριμένη εργασία θεωρούμε ότι οι όψεις αυτές περιορίζονται μόνο σε μορφή SPJ(select project join).

Μία όψη, λοιπόν, στη γλώσσα της Datalog έχει την παρακάτω μορφή:

$M(X)$:- $T_1(X_1), T_2(X_2), \dots , T_n(X_n)$.

Το αριστερό τμήμα του ερωτήματος λέγεται κεφαλή (head) ενώ το δεξιό λέγεται σώμα (body). Κάθε στοιχείο (atom) του ερωτήματος λέγεται και κατηγόρημα(predicate) και αποτελείται από ένα όνομα πίνακα και από ένα σύνολο ιδιοτήτων. Το όνομα του κατηγορήματος της κεφαλής είναι το όνομα της όψης, και επειδή ενδιαφερόμαστε για τις όψεις που θα αποτελούν αντιστοιχίες, το όνομα της κεφαλής στο εξής θα είναι και το όνομα της αντιστοιχίας και θα συμβολίζεται γενικά με M(Mapping). Τα ονόματα των κατηγορημάτων του σώματος είναι τα ονόματα των πινάκων που συμμετέχουν σε μία αντιστοιχία. Τα συμβολίζουμε με $T_1, T_2, \dots , T_n$. Αντίστοιχα, το X είναι το σύνολο των ιδιοτήτων της αντιστοιχίας, και τα $X_1, X_2, \dots , X_n$ τα σύνολα ιδιοτήτων των πινάκων. Σημαντικό στο συμβολισμό της Datalog είναι το γεγονός ότι τα ονόματα των ιδιοτήτων κάθε πίνακα θεωρούνται γνωστά και δεν εμφανίζονται στο Datalog ερώτημα. Οι ιδιότητες ενός πίνακα διακρίνονται μεταξύ τους από τη θέση στην οποία βρίσκονται μέσα στο κατηγόρημα που αντιστοιχεί στον πίνακα. Η τιμή κάθε ιδιότητας αναπαρίσταται από μία μεταβλητή, ή μία σταθερά, εάν θέλουμε αυτή να έχει μία συγκεκριμένη τιμή. Με τον τρόπο αυτό, οι σύνδεσμοι(joins) ανάμεσα στους πίνακες εκφράζονται σαν πολλαπλές εμφανίσεις της ίδιας μεταβλητής στις θέσεις των αντίστοιχων ιδιοτήτων. Οι ιδιότητες της αντιστοιχίας παίρνουν τιμές από μεταβλητές που εμφανίζονται σαν ιδιότητες πινάκων στο σώμα του ερωτήματος. Έτσι, για να έχει η αντιστοιχία νόημα πρέπει το σύνολο X να είναι υποσύνολο (όχι γνήσιο) της ένωσης των συνόλων $X_1, X_2, \dots , X_n$. Εκτός από του παραπάνω τύπου κατηγορήματα, το σώμα ενός ερωτήματος μπορεί να περιέχει κατηγορήματα με αριθμητικές συγκρίσεις. Τα κατηγορήματα αυτά έχουν τη μορφή : $x_i <$ const, όπου στη θέση του '<' μπορεί να υπάρχει οποιοσδήποτε τελεστής σύγκρισης, και στη θέση του 'const' μία αριθμητική ή άλλου τύπου

σταθερά, ανάλογα με τον τύπο της μεταβλητής x_i. Απαιτούμε βέβαια η x_i να ανήκει σε κάποιο(α) από τα X_1, X_2, … , X_n. Τα κατηγορήματα κάθε τύπου του σώματος ενός ερωτήματος Datalog λέγονται subgoals. Τα κατηγορήματα των αριθμητικών συγκρίσεων λέγονται comparison subgoals.

Ορίζοντας λοιπόν την Datalog μορφή της αντιστοιχίας του προηγούμενου παραδείγματος, θα έχουμε:

Professor (a, b, c):-Personnel (a, b, c, d), d = "Professor".

3.2.3. Κατηγορίες Αντιστοιχιών

Έχοντας τώρα στη διάθεση μας, τον τρόπο αναπαράστασης των αντιστοιχιών-όψεων, μπορούμε να αναφερθούμε λεπτομερώς, στα τρία είδη αντιστοιχιών που συναντώνται στα συστήματα data integration, και θα τα δανειστούμε εδώ για αντιστοιχίες σε καθαρά P2P δίκτυα βάσεων δεδομένων.

Οι τρεις κατηγορίες αντιστοιχιών είναι οι εξής:

- Global as View Mappings (GAV)
- Local as View Mappings (LAV)
- Global & Local as View Mappings (GLAV)

Σε συστήματα data integration, των οποίων η διαδικασία μετάφρασης παρουσιάσθηκε προηγουμένως, global θεωρείται το ενδιάμεσο σχήμα (mediated schema), πάνω στο οποίο μεταφράζονται όλα τα ερωτήματα πριν σταλούν στο δίκτυο και local το τοπικό σχήμα του εκάστοτε κόμβου που στέλνει ή λαμβάνει ένα ερώτημα προς απάντηση.

Σε καθαρά p2p συστήματα βάσεων δεδομένων, όπου οι αντιστοιχίες κρατούνται μεταξύ των κόμβων, global θεωρείται το σχήμα του κόμβου που στέλνει μια ερώτηση στο δίκτυο (αποστολέας) και local το σχήμα του κόμβου που δέχεται την ερώτηση προς απάντηση (παραλήπτης).

Έστω για παράδειγμα το παρακάτω **global schema S_G**

Professor (id, name, address)

Department (dept_id, dept_name, domain)

Teaches (profID, deptID)

Lab (labID, name, numOfPhds, managerID)

Με τους εξής περιορισμούς εξωτερικών κλειδιών

Teaches.profID →Professor

Teaches.deptID →Department

Lab.managerID →Professor

Και το παρακάτω **local schema S_L**

Personnel (id, surname, contact, position, teachSchool)

School (schoolID, name, domain)

Laboratory (id, name, numOfResearchers)

IsHeadOf (persID, labID)

Με τους εξής περιορισμούς εξωτερικών κλειδιών

Personnel.teachSchool →School

IsHeadOf.persID →Personnel

IsHeadOf.labID →Laboratory

A) GAV αντιστοιχίες

GAV αντιστοιχίες είναι οι όψεις της μορφής

M(X):- $A_1(X_1),A_2(X_2),\ldots,A_n(X_n)$. όπου n≥1, M το όνομα ενός πίνακα του global schema και A_i i=1..n, ονόματα πινάκων του local schema

Δηλαδή αντιστοιχούν έναν πίνακα του global schema σε ένα αριθμό πινάκων του local schema. Παραδείγματος χάριν

Lab(a,b,c,d):-Laboratory(a,b,c),Personnel(d,-,-,-,-),IsHeadOf(d,a).

B) LAV αντιστοιχίες

Οι LAV αντιστοιχίες είναι το ακριβώς αντίστροφο από τις GAV αντιστοιχίες

Δηλαδή:

M(X):- $A_1(X_1),A_2(X_2),\ldots,A_n(X_n)$. όπου n≥1, M το όνομα ενός πίνακα του local schema και A_i i=1..n, ονόματα πινάκων του global schema

Δηλαδή αντιστοιχούν έναν πίνακα του local schema σε ένα αριθμό πινάκων του global schema. Παραδείγματος χάριν

Personnel (a, b, c, d,e):-Professor(a,b,c),Department(e,_,_),Teaches(a,e).

Γ) GLAV αντιστοιχίες

Η κατηγορία αυτή αποτελεί συνδυασμό των δύο παραπάνω κατηγοριών.

Γενικά έχει τη μορφή

$A_1(Y_1),A_2(Y_2),\ldots,A_n(Y_n).B_1(X_1),B_2(X_2),\ldots,B_m(X_m)$. όπου $m,n\geq 1$, A_i i=1..n, ονόματα πινάκων του global schema, και B_i i=1..m, ονόματα πινάκων του local schema A_i i=1..n, ονόματα πινάκων του global schema.

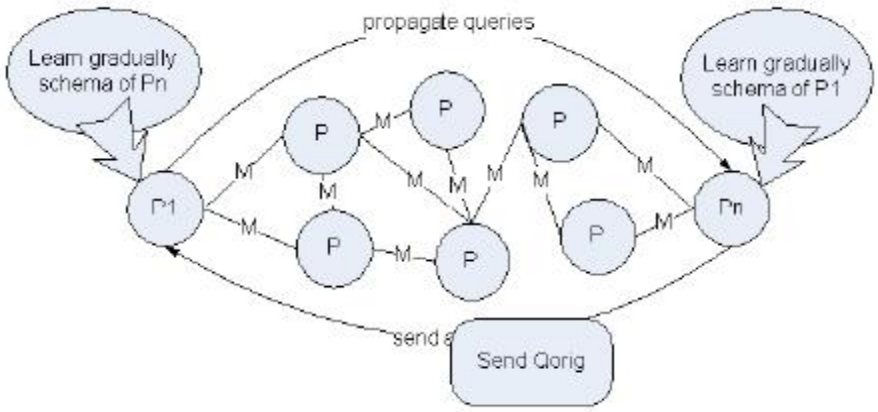
Στην εργασία αυτή δε θα ασχοληθούμε με την ανακάλυψη αυτού του είδους αντιστοιχιών.

3.3 Το σύστημα GrouPeer

Όπως είδαμε στο υποκεφάλαιο 3.1, η διάδοση ερωτημάτων ενός κόμβου, σε ένα καθαρό p2p δίκτυο βάσεων δεδομένων γίνεται μέσω της διαδοχικής μετάφρασης των ερωτημάτων στα σχήματα των γειτόνων του κάθε κόμβου που λαμβάνει το ερώτημα. Με τον τρόπο αυτό, τα ερωτήματα καταφθάνουν σταδιακά σε όλους τους κόμβους του δικτύου και απαντώνται. Αν και όπως είδαμε, με την τεχνική αυτή, μπορούμε να μιλάμε για ένα τελείως αποκεντρωμένο σύστημα, εμφανίζεται ένα σημαντικό μειονέκτημα σε σχέση με τα κλασσικά data integration σενάρια. Το πρόβλημα εντοπίζεται στο γεγονός, ότι οι αντιστοιχίες που υπάρχουν μεταξύ των κόμβων ενδέχεται να μην είναι πλήρεις και σημασιολογικά σωστές και συνεπώς η μετάφραση ενός ερωτήματος από το ένα σχήμα στο επόμενο, να απαξιώνει το ερώτημα, ώστε τελικά να απαντάται κάτι διαφορετικό από αυτό που ρώτησε ο αποστολέας. Σε περιπτώσεις μάλιστα όπου η τελική μετάφραση αποτελείται από πολλές διαδοχικές μεταφράσεις ερωτημάτων (όταν δηλαδή ανάμεσα στον κόμβο που απαντάει, και στον αποστολέα υπάρχουν αρκετοί κόμβοι δικτύου), ενδέχεται να υπάρχει πολύ μεγαλύτερος βαθμός απαξίωσης.

Ας μην ξεχνάμε άλλωστε ότι κατά την εισαγωγή νέων κόμβων στο δίκτυο, οι γείτονες επιλέγονται τυχαία και όχι βάση της ποιότητας των αντιστοιχιών τους με τον νέο κόμβο. Κατά συνέπεια, ένας νέος κόμβος πιθανώς να γειτονεύσει με κόμβους που, τα σχήματα τους, παρουσιάζουν σημαντικές διαφορές, και όχι με κόμβους, των οποίων τα σχήματα είναι σχετικά όμοια με το δικό του. Είναι λογικό λοιπόν, ότι και οι αντιστοιχίες που θα

δημιουργήσει με τους γείτονές του σε αυτή την περίπτωση να μην είναι πλήρεις και ενδεχομένως λανθασμένες.



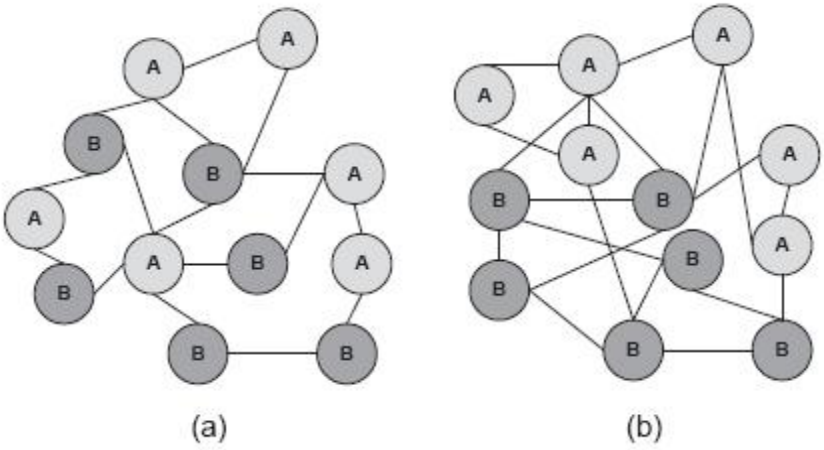
Σχήμα 3 b: Η διάδοση ερωτημάτων στο σύστημα GrouPeer

Το GrouPeer είναι ένα σύστημα, το οποίο έχει σκοπό να ξεπεράσει αυτό το πρόβλημα της σταδιακής απαξίωσης ερωτημάτων, διατηρώντας παράλληλα τον αποκεντρωτικό χαρακτήρα των καθαρών p2p δικτύων. Ο βασικός μηχανισμός που χρησιμοποιεί το GrouPeer, σχετίζεται με την ανακάλυψη και ομαδοποίηση (clustering) κόμβων στο δίκτυο, που έχουν παρόμοια σχήματα και κατ΄επέκταση δύνανται να δημιουργήσουν ισχυρές αντιστοιχίες μεταξύ τους. Ο ανακάλυψη των νέων κόμβων, γίνεται με την αποστολή σε κάθε κόμβο όχι μόνο του μεταφρασμένου ερωτήματος, αλλά και του αρχικού ερωτήματος του αποστολέα(σχήμα 3 b). Το αρχικό ερώτημα, αποκαλύπτει στον κόμβο που απαντά, πληροφορίες για το σχήμα του αποστολέα, ο οποίος χτίζει σταδιακά το άγνωστο σχήμα, κατασκευάζει μαζί του αντιστοιχίες και ελέγχει αν οι αντιστοιχίες αυτές μπορούν να θεωρηθούν αρκετά καλές, ώστε οι δύο κόμβοι να γειτονεύσουν. Στο σημείο αυτό, υπεισέρχεται και ο στόχος της διπλωματικής αυτής εργασίας: **να δημιουργηθεί μια τεχνική, που θα κατασκευάζει αυτόματα αντιστοιχίες μεταξύ δύο σταδιακά αποκαλυπτόμενων σχημάτων.**

Ο έλεγχος, κατά πόσο οι νέες αντιστοιχίες είναι αρκετά ισχυρές, ώστε να θεωρηθεί ότι οι δύο κόμβοι που επικοινωνούν έχουν παρόμοια σχήματα, γίνεται ως εξής:

Το αρχικό ερώτημα του αποστολέα που καταφθάνει σε έναν άγνωστο κόμβο (που δεν είναι γείτονάς του), μεταφράζεται όχι μόνο μέσω των διαδοχικών αντιστοιχιών κατά τη διαδρομή του ερωτήματος στο δίκτυο, αλλά και βάσει των αυτόματα παραγόμενων αντιστοιχιών μεταξύ των δύο κόμβων, και τελικά απαντάται το μεταφρασμένο ερώτημα που είναι περισσότερο σημασιολογικά όμοιο με το αρχικό. Ο χρήστης που λαμβάνει τις απαντήσεις, επιστρέφει ανάδραση στον άγνωστο σχετικά με την ποιότητα μετάφρασης του ερωτήματος.

Αν για πολλά ερωτήματα, ο αποστολέας μείνει ευχαριστημένος από την μετάφραση, αυτό σημαίνει ότι οι δύο κόμβοι έχουν παρόμοιο σχήμα, και ζητά στον απομακρυσμένο κόμβο να γίνει γείτονας του. Έτσι λοιπόν, σταδιακά αναδιατάσσονται οι γειτνιάσεις κόμβων στο δίκτυο, ώστε να δημιουργούνται ομάδες γειτόνων με παρόμοια σχήματα και ισχυρές αντιστοιχίες μεταξύ τους, πράγμα που σημαίνει ότι πλέον οι διαδοχική μετάφραση ερωτημάτων θα απαξιώνει πολύ λιγότερο τα ερωτήματα που διαχέονται στο δίκτυο. Μια τέτοια αναδιάταξη φαίνεται στο σχήμα 3 c.



Σχήμα 3 c: (α) Αρχικές γειτνιάσεις κόμβων (b) Γειτνιάσεις κόμβων ύστερα από την ομαδοποίηση

3.4 Schema Correspondences

Βάσει του προηγούμενου κεφαλαίου παρατηρούμε ότι, για την εύρεση κόμβων παρόμοιου σχήματος, απαιτείται η αυτόματη δημιουργία αντιστοιχιών μεταξύ δύο σχημάτων, κάτι που αποτελεί και τον απώτερο στόχο αυτής της διπλωματικής. Με τον όρο αυτόματη εννοούμε, ότι το σύστημα θα πρέπει να βρίσκει μόνο του, με την ελάχιστη δυνατή παρέμβαση του χρήστη, τη σημασιολογία των στοιχείων του κάθε σχήματος, να δημιουργεί ταιριάσματα μεταξύ στοιχείων που έχουν την ίδια σημασιολογία και τέλος να συνδυάζει αυτά τα ταιριάσματα μεταξύ τους με τους επιπλέον απαραίτητους περιορισμούς ώστε να προκύψουν αντιστοιχίες. Συνεπώς ο αυτοματισμός της παραγωγής αντιστοιχιών προϋποθέτει σε πρώτο στάδιο την αυτόματη παραγωγή ορισμένων ταιριασμάτων μεταξύ των στοιχείων δύο σχημάτων, βάσει της σημασιολογικής έννοιας που αυτά υποκρύπτουν.

Παραδείγματος χάριν, αν σε ένα σχήμα υπάρχει μια ιδιότητα Professor.contact που αναφέρεται στον τρόπο επικοινωνίας με έναν καθηγητή και σε κάποιο άλλο σχήμα υπάρχει η ιδιότητα Professor.email, που αναφέρεται στην ηλεκτρονική διεύθυνση ενός καθηγητή, θέλουμε το σύστημα να αναγνωρίζει αυτόματα ότι και τα δύο στοιχεία σχημάτων έχουν την ίδια σημασιολογική έννοια και για το λόγο αυτό μπορούν να θεωρηθούν ισοδύναμα.

Ορισμός 3.1

Ορίζουμε ως στοιχείο σχήματος (schema element) E ενός σχεσιακού σχήματος S, μια σχέση R (αλλιώς S.R) ή μια ιδιότητα A μιας σχέσης R στη μορφή S.R.A είτε στη μορφή R.A.

Η τεχνική ταιριάσματος στοιχείων δύο διαφορετικών σχημάτων, βάση της ομοιότητας της σημασιολογίας τους ονομάζεται Schema Matching και είναι κατά κοινή ομολογία ένα από τα δυσκολότερα προβλήματα στον τομέα της ετερογένειας δεδομένων. Πολλά συστήματα έχουν αναπτυχθεί (Matchers) , το οποία προσπαθούν με ευριστικές μεθόδους να λύσουν αυτό το πρόβλημα, δηλαδή δεδομένων δύο σχημάτων, να παράγουν ένα σύνολο από ισοδυναμίες στοιχείων, οι οποίες θα φανερώνουν την σημασιολογική αντιστοίχιση του ενός σχήματος στο άλλο.

Οι πιθανές ισοδυναμίες που μπορούν να εξαχθούν ανάμεσα σε δύο σχήματα, εν γένει έχουν διάφορες μορφές και για το λόγο αυτό μπορούν να κατηγοριοποιηθούν με διαφορετικά κριτήρια.

Ας δούμε μερικές από αυτές τις κατηγορίες:

<u>Κατηγορίες με βάση την κατεύθυνση της ισοδυναμίας:</u>

<u>Α) Ισοδυναμία δύο στοιχείων χωρίς κατεύθυνση (Undirected Correspondence)</u>

Ορισμός 3.2

Μια ισοδυναμία δύο στοιχείων διαφορετικού σχήματος E1 και E2, χωρίς κατεύθυνση ορίζεται το ταίριασμα της σημασιολογικής έννοιας των στοιχείων E1, E2 και γράφεται ως $C_U(E1,E2)$. Κάθε ισοδυναμία δύο στοιχείων σχήματος συνοδεύεται από μια τιμή $0 \leq |C_U(E1,E2)| \leq 1$ που εκφράζει το βαθμό βεβαιότητας, στον οποίο αυτή η ισοδυναμία ισχύει.

Παράδειγμα μιας τέτοιας ισοδυναμίας είναι το παράδειγμα που αναφέρθηκε στην αρχή του υποκεφαλαίου

Professor.contact ← → Professor.email ή C_U *(Professor.contact, Professor.email)*

<u>Β) Ισοδυναμίες δύο στοιχείων με κατεύθυνση (Directed Correspondence)</u>

Ορισμός 3.3

Ορίζουμε ως ισοδυναμία με κατεύθυνση $C_D(E1,E2)$(directed correspondence)μεταξύ δυο στοιχείων σχήματος E1,E2 το ταίριασμα της σημασιολογικής έννοιας των στοιχείων E1 και

E2, κατά το οποίο δηλώνεται ότι η έννοια του στοιχείου E1 περικλείεται στην έννοια του στοιχείου E2(Σχέση E1 IS-A E2). Κάθε ισοδυναμία με κατεύθυνση δύο στοιχείων σχήματος συνοδεύεται από μια τιμή $0 \leq |C_D(E1,E2)| \leq 1$ *που εκφράζει το βαθμό βεβαιότητας, στον οποίο αυτή η ισοδυναμία ισχύει.*

Για να δούμε ένα παράδειγμα μιας τέτοιας ισοδυναμίας, ας θυμηθούμε τα δύο σχήματα που χρησιμοποιήθηκαν για τον ορισμό των αντιστοιχιών.

Σε αυτά τα σχήματα, μια ισοδυναμία με κατεύθυνση είναι η

Professor.id → Personnel.id ή C_D (Professor.id, Personnel.id)

Πράγματι το αναγνωριστικό ενός καθηγητή **είναι ένα** αναγνωριστικό προσωπικού, το αντίθετο όμως δεν ισχύει, διότι δεν είναι όλα τα μέλη του προσωπικού καθηγητές.

Στο σημείο αυτό να παρατηρήσουμε ότι κάθε ισοδυναμία χωρίς κατεύθυνση μπορεί να δημιουργήσει δύο ισοδυναμίες με κατεύθυνση. Παραδείγματος χάριν, η ισοδυναμία

C_U (Professor.contact, Professor.email) γεννά τις ισοδυναμίες C_D (Professor.contact, Professor.email) και C_D (Professor.email, Professor.contact)

<u>Κατηγορίες με βάση την πληθικότητα (cardinality) της ισοδυναμίας:</u>

Με βάση αυτό το κριτήριο έχουμε 3 κατηγορίες ισοδυναμιών:

<u>Α) Ισοδυναμίες ένα προς ένα (1:1)</u>

Στην κατηγορία αυτή εμπίπτουν οι ισοδυναμίες που αντιστοιχίζουν ένα στοιχείο του ενός σχήματος με ένα στοιχείο του άλλου σχήματος. Παραδείγματα τέτοιων ισοδυναμιών είναι οι ισοδυναμίες που αναφέρθηκαν προηγουμένως.

<u>Β) Ισοδυναμίες ένα προς πολλά (1:Μ)</u>

Στην κατηγορία αυτή ένα στοιχείο ενός σχήματος αντιστοιχίζεται με ένα σύνολο στοιχείων του άλλου σχήματος. Πιο συγκεκριμένα:

Ορισμός 3.4

Ορίζουμε ως ισοδυναμία ενός συνόλου στοιχείων σχήματος $V = Set\{E_i\}$ *με ένα στοιχείο A1* $C_{SET}(V,A1)$ *το ταίριασμα της σημασιολογικής έννοιας f(V) με τη σημασιολογική έννοια του στοιχείου A1.. Κάθε ισοδυναμία αυτής της μορφής συνοδεύεται από μια τιμή* $0 \leq |C_{SET}(V,A1)| \leq 1$ *που εκφράζει το βαθμό βεβαιότητας, στον οποίο αυτή η ισοδυναμία ισχύει.*

Η συνάρτηση f, είναι κάποια συνάρτηση που παίρνει ως είσοδο τα n στοιχεία του συνόλου V, και δίνει ως έξοδο τη σημασιολογική έννοια, που εκπροσωπεί ο συνδυασμός της συνάρτησης με τα στοιχεία αυτά.

Έστω ο παρακάτω πίνακας του σχήματος S1

Employee (empID, name, payRate, HoursWork)

Και ένας δεύτερος πίνακας ενός άλλου σχήματος S2

Personnel (id, first_name, surname, salary)

Παρατηρούμε ότι ενώ το *id* της σχέσης Personnel μπορεί να αντιστοιχηθεί μονοσήμαντα με την ιδιότητα *empID* της σχέσης Employee, δε συμβαίνει το ίδιο για την ιδιότητα *salary* (μισθός). Παρόλα αυτά η σημασιολογική έννοια του εισοδήματος ενός εργαζομένου κρύβεται έμμεσα στον πίνακα *Employee* από τις ιδιότητες *PayRate* και *HoursWork* . Έτσι μπορούμε να ορίσουμε την παρακάτω ισοδυναμία 1 προς πολλά:

Personnel.salary ←→ *Employee.payRate*Employee.HoursWork* (πράγματι το εισόδημα ενός υπαλλήλου είναι το γινόμενο του ωρομισθίου του με τις ώρες που δουλεύει). Η συνάρτηση f που χρησιμοποιούμε εδώ είναι ο συντελεστής πολλαπλασιασμού, αλλά γενικά για κάθε ισοδυναμία ένα προς πολλά, η συνάρτηση αυτή μπορεί να διαφέρει.

Γ) Ισοδυναμίες πολλά προς πολλά (M:N)

Παρόμοια με την παραπάνω κατηγορία μπορούμε να ορίσουμε:

Ορισμός 3.5

Ορίζουμε ως ισοδυναμία ενός συνόλου στοιχείων σχήματος V1= Set{E_i} με ένα σύνολο στοιχείων V2 = Set{A_i} και συμβολίζουμε με $C_{SET\text{-}SET}(V1,V2)$ το ταίριασμα της σημασιολογικής έννοιας f1(V1) με τη σημασιολογική f2(V2). Κάθε ισοδυναμία αυτής της μορφής συνοδεύεται από μια τιμή $0 \leq |C_{SET\text{-}SET}(V,A1)| \leq 1$ που εκφράζει το βαθμό βεβαιότητας, στον οποίο αυτή η ισοδυναμία ισχύει.

Η συνάρτηση fi είναι μια συνάρτηση που παίρνει ως είσοδο τα n στοιχεία του συνόλου Vi, και δίνει ως έξοδο τη σημασιολογική έννοια, που εκπροσωπεί ο συνδυασμός της συνάρτησης με τα στοιχεία αυτά.

Στην παρούσα διπλωματική εργασία θα μας απασχολήσουν μόνο ισοδυναμίες ένα προς ένα (1:1) και για το λόγο αυτό θα αναφερθούμε μόνο στις τεχνικές για αυτόματη εξαγωγή αυτής της κατηγορίας των ισοδυναμιών.

Στη βιβλιογραφία υπάρχει μια πληθώρα κριτηρίων για το κατά πόσο ένα στοιχείο ενός σχήματος ισοδυναμεί με ένα άλλο στοιχείο κάποιου άλλου σχήματος και σε ποιο βαθμό. Πολλά από αυτά βασίζονται στη λεξικογραφική ομοιότητα των ονομάτων των δύο στοιχείων (πχ *Professor.id*←→*Professor.profID*), αλλά σε λίστες συνωνύμων και συντομογραφιών και

άλλα σε περιορισμούς των εκάστοτε σχημάτων. Ο αλγόριθμος που δέχεται ως είσοδο τα δύο σχήματα και παράγει ένα σύνολο από ισοδυναμίες στοιχείων, βασισμένος σε κάποιο συγκεκριμένο κριτήριο ονομάζεται συσχετιστής (Matcher). Δεδομένης λοιπόν της ύπαρξης πολλών διαφορετικών κριτηρίων για τη συσχέτιση δύο στοιχείων σχήματος, τα περισσότερα εργαλείο που έχουν αναπτυχθεί για Automatic Schema Matching συνδυάζουν τα αποτελέσματα από πολλούς διαφορετικούς συσχετιστές, και τελικά δίνουν ως έξοδο τις ισοδυναμίες εκείνες, των οποίων ο βαθμός βεβαιότητας είναι αθροιστικά υψηλότερος.

Παρακάτω παραθέτουμε μερικές από τις πιο γνωστές προσεγγίσεις συσχέτισης δύο στοιχείων σχήματος.

A) **Γλωσσολογικές Προσεγγίσεις (Linguistic approaches)**

Οι συσχετιστές αυτής της προσέγγισης βασίζονται στη γλώσσα (γλωσσολογικοί συσχετιστές) και χρησιμοποιούν ονόματα και κείμενο (π.χ. λέξεις ή προτάσεις) για να βρουν σημασιολογικά παρόμοια στοιχεία σχημάτων. Παρακάτω περιγράφουμε δύο προσεγγίσεις επιπέδου σχήματος, τον ονοματικό συσχετισμό (name matching) και τον περιγραφικό συσχετισμό (description matching).

i) Συσχέτιση βασισμένη στο όνομα

Ο συσχετισμός που βασίζεται στα ονόματα συσχετίζει στοιχεία σχήματος με όμοια ή παρόμοια ονόματα. Η ομοιότητα των ονομάτων μπορεί να καθοριστεί και να μετρηθεί με διάφορους τρόπους, συμπεριλαμβάνοντας:

- Ισότητα ονομάτων
- Ισότητα αναπαραστάσεων κανονικών (canonical) ονομάτων κατόπιν αποκοπής ή και άλλης προεπεξεργασίας. Μια τέτοια προεπεξεργασία είναι σημαντική στο χειρισμό ειδικών συμβόλων προθεμάτων/επιθεμάτων (π.χ. CName←→ customer name, και EmpNO ←→employee number)
- Ισότητα συνωνύμων (π.χ. car ←→automobile, και make ←→brand)
- Ισότητα υπερωνύμων (π.χ. book is-a publication και article is-a publication δηλώνει ότι book →publication, article →publication, και book ←→article)
- Ομοιότητα ονομάτων βασισμένα σε κοινά υπό-αλφαριθμητικά, προφορά, soundex (μια κωδικοποίηση των ονομάτων βασισμένη περισσότερο στο πως ακούγονται παρά στο πως γράφονται), κτλ. (π.χ. representedBy ←→representative, ShipTo ←→Ship2)
- Συσχετισμοί ονομάτων που δίνονται από το χρήστη (π.χ. reportsTo ←→manager, issue ←→bug)

Η εκμετάλλευση των συνωνύμων ή των υπερωνύμων απαιτεί τη χρήση θησαυρών λέξεων ή κοινών λεξικών. Τα γενικά λεξικά της φυσικής γλώσσας μπορεί να είναι επίσης χρήσιμα, ή

και τα πολυγλωσσικά λεξικά (π.χ. αγγλο-γερμανικό) για το χειρισμό σχημάτων εισόδου διαφορετικών γλωσσών. Επιπλέον, ο συσχετισμός ονομάτων μπορεί να χρησιμοποιεί λεξικά καθορισμένα για πεδία ή επιχειρήσεις και is-a ταξινομίες που περιλαμβάνουν κοινά ονόματα, συνώνυμα και περιγραφές στοιχείων σχημάτων, συντμήσεις, κτλ. Αυτά τα συγκεκριμένα λεξικά απαιτούν μια αξιόλογη προσπάθεια για να κατασκευαστούν σε ένα συνεπή τρόπο. Η προσπάθεια αυτή αξίζει την επένδυση, ειδικά για σχήματα με σχετικά επίπεδη δομή όπου τα λεξικά παρέχουν τα πιο σημαντικά στοιχεία για συσχέτιση. Επιπροσθέτως, χρειάζονται διάφορα εργαλεία για να επιτρέψουν στα ονόματα να προσπελαστούν και να επαναχρησιμοποιηθούν, όπως χρειάζονται και σε ένα συντάκτη σχημάτων όταν ορίζει νέα σχήματα.

Τα ομώνυμα είναι ίδιες ή παρόμοιες λέξεις που αναφέρονται σε διαφορετικά πράγματα. Είναι, λοιπόν, ξεκάθαρο ότι τα ομώνυμα μπορούν να μπερδέψουν ένα αλγόριθμο συσχετισμού. Τα ομώνυμα είναι μέρος της φυσικής γλώσσας, για παράδειγμα «κόμμα» ως πολιτικός σχηματισμός και σημείο στίξης, ή μπορεί να είναι συγκεκριμένα για ένα πεδίο εφαρμογής, όπως η "γραμμή" που μπορεί να σημαίνει την κατευθυντήρια γραμμή μιας επιχείρησης ή την τηλεφωνική γραμμή. Ένας ονοματικός συσχετιστής μπορεί να μειώσει τους λανθασμένους υποψήφιους συσχετισμούς εκμεταλλευόμενος αταίριαστη πληροφορία που παρέχεται από τους χρήστες ή λεξικά. Τουλάχιστον, ο συσχετιστής μπορεί να προσφέρει μια προειδοποίηση για ενδεχόμενη αμφισημία εξαιτίας πολλαπλών ερμηνειών ενός ονόματος. Μια πιο αυτοματοποιημένη χρήση της αταίριαστης πληροφορίας είναι πιθανή με τη χρήση πληροφορίας από τα συμφραζόμενα, για παράδειγμα, για να ξεχωρίσει το Order.Line από το Business.Line. Μια τέτοια τεχνική κάνει πιο ασαφή τη διάκριση μεταξύ σημασιολογικών και δομικών τεχνικών.

ii) Συσχέτιση βασισμένη στην περιγραφή

Συχνά, τα σχήματα περιέχουν σχόλια σε φυσική γλώσσα που εκφράζουν την αναμενόμενη σημασιολογία των στοιχείων του σχήματος. Αυτά τα σχόλια μπορούν να εκτιμηθούν γλωσσολογικά για να καθορίσουν την ομοιότητα ανάμεσα σε στοιχεία σχημάτων. Για παράδειγμα, αυτό θα βοηθούσε να βρεθεί ότι τα παρακάτω στοιχεία συσχετίζονται, μέσω μιας γλωσσολογικής ανάλυσης των σχολίων που υπάρχουν σε κάθε στοιχείο σχήματος:

S1: empn // employee name

S2: name // name of employee

Αυτή η γλωσσολογική ανάλυση μπορεί να είναι απλή εξάγοντας λέξεις – κλειδιά από την περιγραφή, οι οποίες χρησιμοποιούνται για σύγκριση συνωνύμων, και άρα ονομάτων. Ή μπορεί να είναι πιο πολύπλοκη χρησιμοποιώντας την τεχνολογία για κατανόηση φυσικής γλώσσας προκειμένου να βρούμε σημασιολογικά ισοδύναμες εκφράσεις.

Β) **Προσεγγίσεις Βασισμένες στους Περιορισμούς (Constraint-based approaches)**

Τα σχήματα πολύ συχνά περιέχουν περιορισμούς για τον ορισμό τύπων δεδομένων και ευρών τιμών, μοναδικότητας, προαιρετικότητας, τύπων σχέσεων και βαθμών, κτλ. Αν τα δύο σχήματα εισόδου περιέχουν τέτοια πληροφορία, αυτή μπορεί να χρησιμοποιηθεί από ένα συσχετιστή για να καθορίσει την ομοιότητα των στοιχείων των σχημάτων. Για παράδειγμα, η ομοιότητα μπορεί να βασιστεί στην ισοδυναμία τύπων δεδομένων και πεδίων, των χαρακτηριστικών των κλειδιών (π.χ. μοναδικό, πρωτεύον, ξένο), των βαθμών σχέσεων (π.χ. 1:1 σχέσεις), ή των is-a σχέσεων.

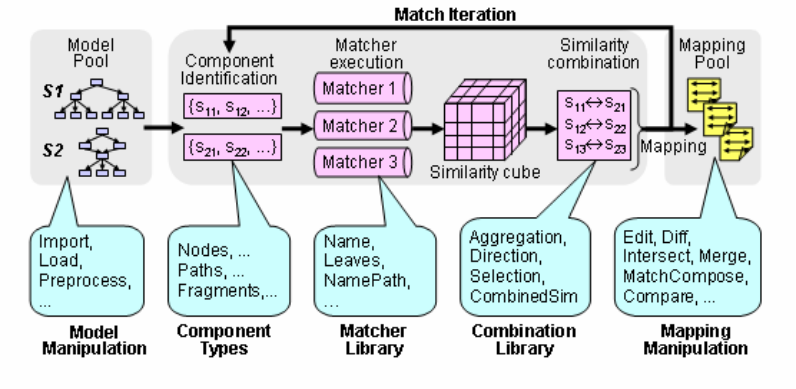
Η υλοποίηση μπορεί συχνά να πραγματοποιηθεί με ένα συσχετισμό επιπέδου στοιχείου όπως της συσχέτισης, χρησιμοποιώντας τώρα τους τύπους δεδομένων, τις δομές, και τους περιορισμούς στις συγκρίσεις. Ισοδύναμοι τύποι δεδομένων και ονόματα περιορισμών (π.χ., string ≅ varchar, primary key ≅ unique) μπορούν να δοθούν από ένα ειδικό πίνακα συνωνύμων.

Σχήμα S1	**Σχήμα S2**
Employee	Personnel
EmpNo – int, primary key	Pno – int, unique
EmpName – varchar (50)	Pname – string
DeptNo – int, references Department	Dept – string
Salary – dec (15,2)	Born – date
Birthdate – date	
Department	
DeptNo – int, primary key	
DeptName – varchar (40)	

Πίνακας 3 a: Δύο σχήματα για εξαγωγή ισοδυναμιών βασισμένη στους περιορισμούς

Στο παράδειγμα του Πίνακα 3 a, η πληροφορία του τύπου και του κλειδιού υποδεικνύει ότι το Born ισοδυναμεί με το Birthdate και το Pno ισοδυναμεί είτε με το EmpNo ή το DeptNo. Τα υπόλοιπα στοιχεία του S2 Pname και Dept είναι αλφαριθμητικά και άρα μάλλον συσχετίζονται με το EmpName ή το DeptName.

3.5 Automatic Schema Matcher , Coma/ Coma++

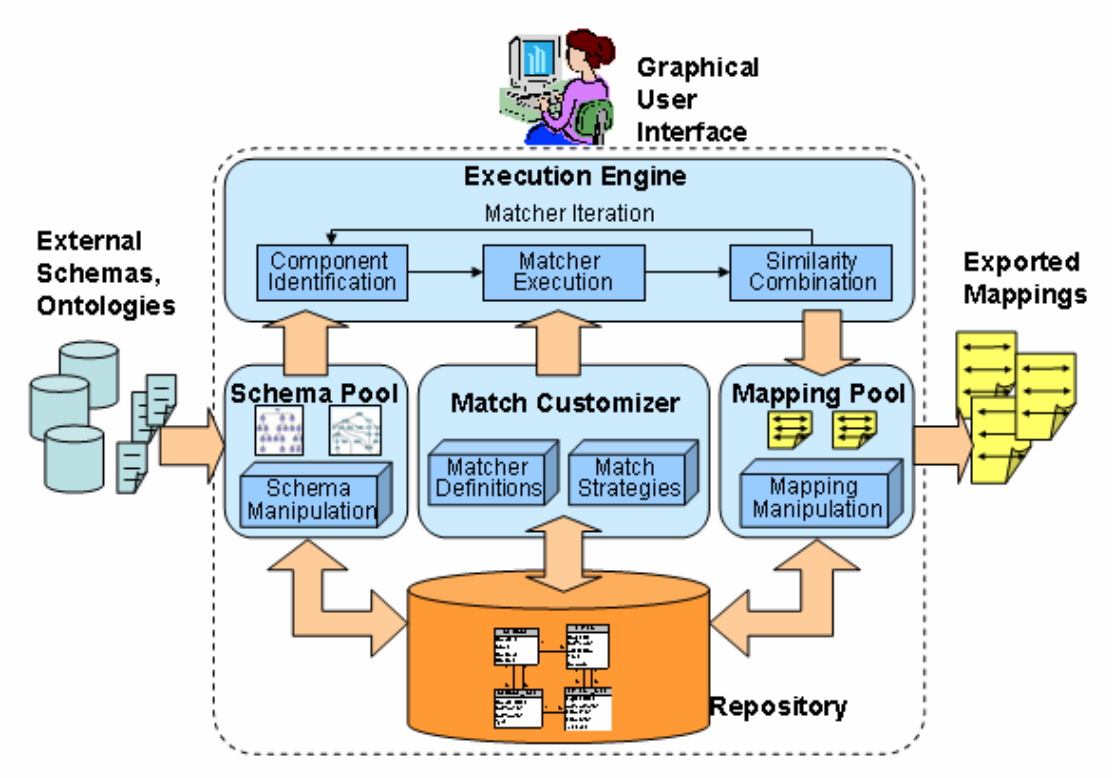


Σχήμα 3 d: Η λειτουργία του Matcher Coma++

Το Coma αποτελεί μια από τις πρώτες ολοκληρωμένες προσπάθειες δημιουργίας ενός εργαλείου που επιτρέπει στο χρήστη να συνδυάζει διαφορετικούς συσχετιστές. Εξέλιξη του εργαλείου αυτού αποτελεί το COMA++ το οποίο δόθηκε στην διάθεση των ερευνητών τον Οκτώβριο του 2005 και το οποίο χρησιμοποιήθηκε και στην εργασία αυτή.

Το εργαλείο αυτό παρέχει μια μεγάλη βιβλιοθήκη από διαφορετικούς συσχετιστές και υποστηρίζει διάφορους τρόπους συνδυασμού των αποτελεσμάτων από διαφορετικούς συσχετιστές. Οι συσχετιστές υποστηρίζουν εύρεση ισοδυναμιών βασισμένοι και σε επίπεδο δομής αλλά και σε επίπεδο στοιχείου. Ιδιαίτερο ενδιαφέρον παρουσιάζουν οι συσχετίσεις οι οποίες προκύπτουν από τους συσχετιστές επαναχρησιμοποίησης αποτελεσμάτων. Τα σχήματα μετατρέπονται σε κατευθυνόμενο ακυκλικό γράφο και αποθηκεύονται σε κατάλληλη βάση για να τα χρησιμοποιήσουν οι συσχετιστές που θα επιλέξει ο χρήστης μέσω της διεπαφής. Κάθε στοιχείο ενός σχήματος αναγνωρίζεται από το μονοπάτι που χρειάζεται να ακολουθηθεί από τον αρχικό κόμβο (root) μέχρι το κόμβο που βρίσκεται αυτό στον γράφο. Η διαδικασία συσχέτισης φαίνεται αναλυτικά στο σχήμα 3 d.

Η προσέγγιση που αναλύουμε χειρίζεται μεγάλη γκάμα σχημάτων. Οι συσχετιστές του εργαλείου μπορούν να πάρουν ως είσοδο σχεσιακές βάσεις, Οντολογίες (rdf, owl) καθώς και XML σχήματα (XSD, XDR). Ιδιαίτερο ενδιαφέρον παρουσιάζουν οι συσχετιστές τμημάτων (fragment) καθώς και οι συσχετιστές επαναχρησιμοποίησης αποτελεσμάτων. Τα αποτελέσματα που προκύπτουν από το εργαλείο είναι ιδιαιτέρως καλά λόγω της χρήσης κατάλληλων λεξικών συνωνύμων και συντμήσεων. Στο σχήμα 3 e φαίνεται η αρχιτεκτονική του εργαλείου διαχείρισης ισοδυναμιών COMA++.



Σχήμα 3 e: Η αρχιτεκτονική του εργαλείου Coma++

4 *Ανάλυση Συστήματος*

Έχοντας τώρα το κατάλληλο θεωρητικό υπόβαθρο, είμαστε σε θέση να αναλύσουμε λεπτομερώς το σύστημα που αναπτύχθηκε. Υπενθυμίζουμε ότι στόχος της εργασίας είναι να δημιουργηθεί ένας μηχανισμός, ο οποίος συνδυάζοντας απλές 1:1 ισοδυναμίες στοιχείων μεταξύ δύο σχημάτων και εκμεταλλευόμενος, την ανάδραση του χρήστη, θα παράγει αντιστοιχίες τύπου GAV και LAV μεταξύ των δύο σχημάτων. Μια τέτοια λειτουργία, όπως αναφέραμε είναι απαραίτητη για το σύστημα GrouPeer, προκειμένου δύο άγνωστοι κόμβοι να μπορούν να γειτονεύσουν, αν τα σχήματα τους είναι παραπλήσια και οι αντιστοιχίες μεταξύ τους ισχυρές. Παρόλα αυτά, ο μηχανισμός που αναπτύχθηκε δεν περιορίζεται στα πλαίσιο των p2p συστημάτων βάσεων δεδομένων και μπορεί να χρησιμοποιηθεί ως ένα ανεξάρτητο εργαλείο για την ημιαυτόματη παραγωγή και σταδιακή βελτίωση GAV/LAV αντιστοιχιών μεταξύ δύο σχημάτων. Η παραγωγή GLAV αντιστοιχιών δε θα μας απασχολήσει σε αυτή την εργασία.

Στο συγκεκριμένο κεφάλαιο, αρχικά δίνουμε μια γενική εικόνα για την αρχιτεκτονική του συστήματος, παρουσιάζουμε τις βασικές υπορουτίνες του όλου μηχανισμού και τον τρόπο που αυτές επικοινωνούν μεταξύ τους και ύστερα αναλύουμε την κάθε υπορουτίνα ξεχωριστά με περισσότερες λεπτομέρειες και παραδείγματα.

Να αναφερθεί ότι σε όλη την περαιτέρω ανάλυση του μηχανισμού, θεωρούμε ότι τα δύο σχήματα είναι γνωστά και πλήρη από την αρχή, κάτι το οποίο συμβαίνει όταν ο μηχανισμός χρησιμοποιείται ως ανεξάρτητο εργαλείο. Στην περίπτωση των p2p συστημάτων βάσεων δεδομένων όμως, το σχήμα του κόμβου που στέλνει ένα ερώτημα, δεν είναι γνωστό αρχικά. Αντίθετα χτίζεται σταδιακά από τα εισερχόμενα ερωτήματα που αποκαλύπτουν κάποιο μέρος του σχήματος. Οι αλλαγές που απαιτούνται στη λειτουργία της διαδικασίας παραγωγής αντιστοιχιών, όταν το ένα από τα δύο σχήματα είναι μη πλήρες δεν επηρεάζουν την βασική της λειτουργία και συμπεριφορά και για το λόγο αυτό θα αναφερθούν ανεξάρτητα

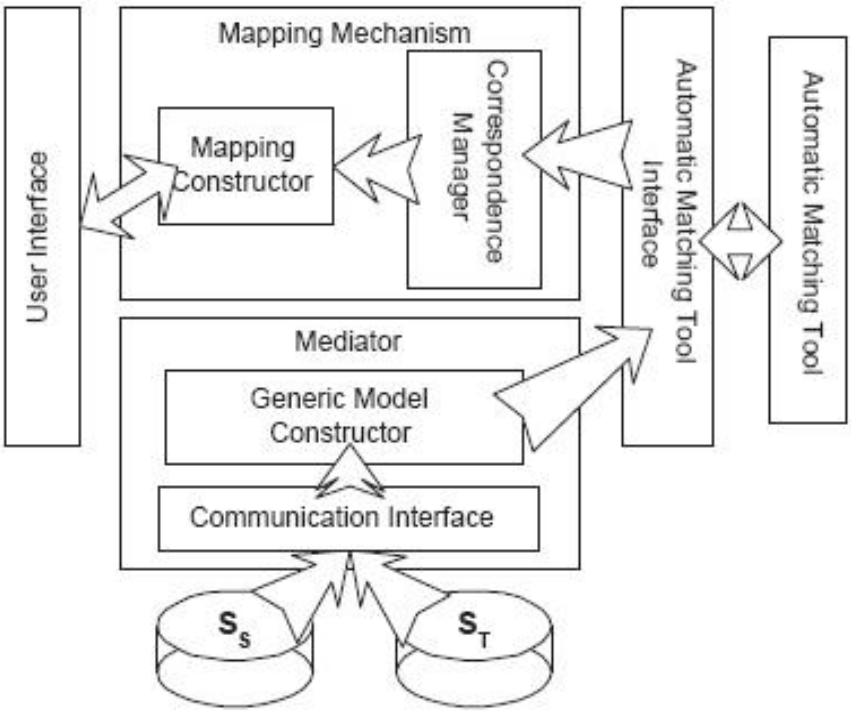
από την βασική ανάλυση του συστήματος, στην παράγραφο «Ενσωμάτωση του μηχανισμού στο GrouPeer».

4.1 Γενική δομή του συστήματος

Ας θεωρήσουμε ένα σύστημα διαχείρισης ομότιμων βάσεων δεδομένων (Peer to Peer Database Management System). Το σχήμα του εκάστοτε κόμβου στο δίκτυο περιλαμβάνει το σύνολο από τους πίνακες της βάσης του κόμβου, περιορισμούς κύριων κλειδιών και περιορισμούς εξωτερικών κλειδιών, οι οποίοι δημιουργούν τρόπους συνένωσης μεταξύ των πινάκων του σχήματος. Κάθε κόμβος κρατά για κάθε γειτονικό κόμβο του ένα σύνολο από αντιστοιχίες σχήματος μορφής GAV(Global As View) και LAV(Local As View), μεταξύ του δικού του σχήματος και του σχήματος του απομακρυσμένου κόμβου. Οι αντιστοιχίες αυτές βρίσκονται σε μορφή όψεων conjunctive μορφής (ισοδύναμα με ερωτήματα SPJ-Select, Project,Join). Υπενθυμίζουμε ότι σε συστήματα ομότιμων βάσεων δεδομένων, global σχήμα θεωρείται το σχήμα του κόμβου που στέλνει ένα ερώτημα στο δίκτυο και local, το σχήμα του κόμβου που λαμβάνει το ερώτημα και προσπαθεί να το μεταφράσει στη δική του βάση δεδομένων. Αρχικά το ερώτημα ενός κόμβου είναι εκφρασμένο πάνω στο δικό του σχήμα (global) και εξαπλώνεται στο δίκτυο πάνω σε μονοπάτια κόμβων, οι οποίοι προσπαθούν να το απαντήσουν με διαδοχικές μεταφράσεις. Συνεπώς η μετάφραση του ερωτήματος γίνεται στους απομακρυσμένους κόμβους.

Έστω ότι ο κόμβος Α στέλνει ερωτήματα στον κόμβο Β. Μέσω των ερωτημάτων αυτών, το σχήμα του κόμβου Α, αποκαλύπτεται σταδιακά στον κόμβο Β, και ο κόμβος Β θέλει να δημιουργήσει αντιστοιχίες πινάκων (GAV/LAV mappings) μεταξύ των δύο σχημάτων, ώστε να είναι σε θέση να μεταφράσει το ερώτημα που έλαβε.

Αν ονομάσουμε το ήδη ανακαλυφθέν και πιθανώς ημιτελές σχήμα του κόμβου Α Target Schema και το σχήμα του κόμβου Β Source Schema, ενδιαφερόμαστε για την κατασκευή και βελτίωση των GAV/LAV αντιστοιχιών (mapping) μεταξύ των σχημάτων S_T (Target Schema) και Ss (Source Schema).



Σχήμα 4 a: Γενική δομή του συστήματος

Στο σχήμα 4 a παρουσιάζεται η βασική αρχιτεκτονική του συστήματος για την παραγωγή των αντιστοιχιών. Παρατηρούμε ότι η βασική ρουτίνα κατασκευής των αντιστοιχιών (mapping constructor) επικοινωνεί τόσο με την διεπαφή του χρήστη (user interface), όσο και με την υπορουτίνα Correspondence Manager,που διαχειρίζεται τις ισοδυναμίες στοιχείων μεταξύ των δύο σχημάτων. Κύριος και βασικότερος ρόλος δηλαδή του συστήματος είναι να συνδυάζει αποτελεσματικά και ευέλικτα τις κατάλληλες ισοδυναμίες στοιχείων, ώστε να προκύπτουν αντιστοιχίες πινάκων. Με άλλα λόγια ένα σύνολο από αντιστοιχίσεις ιδιοτήτων μεταλλάσσεται σε αντιστοιχίσεις πινάκων, ώστε τα ερωτήματα να μπορούν να μεταφραστούν αυτόματα.

Ας δούμε συνοπτικά τα στάδια αυτής της διαδικασίας:

Σε πρώτη φάση, ο μηχανισμός επικοινωνεί με κάθε ένα από τα δυο σχήματα εισόδου μέσω της ρουτίνας Generic Model Constructor και συλλέγει όλα τα απαραίτητα χαρακτηριστικά που εμφανίζει το εκάστοτε σχήμα. Τα χαρακτηριστικά αυτά είναι τα παρακάτω γνωρίσματα:

- Τα ονόματα όλων των πινάκων του σχήματος
- Τα ονόματα όλων των ιδιοτήτων του σχήματος
- Τους περιορισμούς εξωτερικών κλειδιών
- Τους περιορισμούς κύριων κλειδιών
- Το σύνολο με τους πιθανούς συνδέσμους για κάθε ζεύγος πινάκων του σχήματος

- Γενικούς περιορισμούς σχήματος
- Το σύνολο με τα πιθανά μονοπάτια συνένωσης μέχρι κάποιο συγκεκριμένο μήκος για κάθε ζεύγος πινάκων του σχήματος

Ορισμός 4.1

Ορίζουμε ως σύνδεσμο (join) ενός πίνακα Ta με έναν πίνακα Tb, τη σχέση A1.a1 = A2.a2 όπου A1 = Ta και A2 = Tb, ή A1 = Tb και A2 = Ta, που δηλώνει ότι με αυτόν τον τρόπο, τα δεδομένα των δύο πινάκων δύνανται να συνδυαστούν

Οι πιθανοί σύνδεσμοι ενός σχήματος μπορούν να προέρχονται είτε από περιορισμούς εξωτερικών κλειδιών του σχήματος, είτε από τους συνδέσμους των ερωτημάτων που εκτελεί ο χρήστης σε μια βάση δεδομένων.

Ορισμός 4.2

Ορίζουμε ως μονοπάτι συνένωσης μήκους n (joinpath(n)) ενός πίνακα Ta με έναν πίνακα Tb ενός σχήματος, μια συγκεκριμένη σειρά από n συνδέσμους (joins) της μορφής JOINi (Ai1.ai1 = Ai2.ai2) με i = 1..n, όπου A11 = Ta, An2 = Tb και Ai2 = A(i+1)1 για i = 1..n-1

Ένα μονοπάτι συνένωσης δύο πινάκων δείχνει και αυτό έναν πιθανό τρόπο που συνδέονται δύο πίνακες στο σχήμα μεταξύ τους, αλλά αυτή τη φορά με τη βοήθεια ενδιάμεσων σχέσεων. Παραδείγματος χάριν το μονοπάτι συνένωσης μήκους 2 (*Employee.id = EmpWorks.empid and EmpWorks.deptID = Department.id*) στο παρακάτω SQL ερώτημα, μας δίνει πληροφορίες για τον υπάλληλο και το τμήμα που αυτός δουλεύει:

Select ALL

From Employee, Department

Where Employee.id = EmpWorks.empid and EmpWorks.deptID = Department.id

Ο πίνακας *EmpWorks* είναι ένας ενδιάμεσος πίνακας που συμμετέχει στη σύνδεση των σχέσεων *Employee* και *Department*. Παρατηρούμε επίσης ότι κάθε σύνδεσμος (join) ορίζει αυτόματα και ένα μονοπάτι συνένωσης μήκους 1.

Εύκολα συμπεραίνει κανείς ότι για n πίνακες έχουμε $n+\binom{n}{2}=\frac{n(n-1)}{2}+n$ ζεύγη πινάκων, και συνεπώς ισάριθμα σύνολα με συνδέσμους και πιθανά μονοπάτια συνένωσης.

Όλα τα παραπάνω στοιχεία είναι απολύτως απαραίτητα για τη δημιουργία των αντιστοιχιών και πρέπει να κρατούνται τόσο για το source schema όσο και για το local. Έτσι, ο Generic Model Constructor δημιουργεί ένα γενικό πλαίσιο για κάθε σχήμα, το οποίο στο

εξής θα ονομάζεται γενικό μοντέλο σχήματος (Schema Generic Model), και θα ενσωματώνει αυτές τις πληροφορίες.

Αφού κατασκευαστούν λοιπόν τα δύο γενικά μοντέλα για τα δύο σχήματα εισόδου, θέλουμε να βρούμε το σύνολο των απλών ισοδυναμιών που ισχύουν ανάμεσα σε αυτά. Εδώ λοιπόν υπεισέρχεται ο ρόλος της ρουτίνας Correspondence Manager, η οποία επικοινωνεί εξωτερικά με τον Matcher Coma++, και διαχειρίζεται τις ισοδυναμίες που αυτός ανακάλυψε.

Πιο συγκεκριμένα, η ρουτίνα Correspondence Manager εξάγει δύο σύνολα ($C_{\mathcal{G}}(S_T,S_S)$ και $C_{\mathcal{L}}(S_T,S_S)$) από directed 1:1 ισοδυναμίες στοιχείων μεταξύ των δύο σχημάτων. Οι ισοδυναμίες αυτές προέρχονται ύστερα από την επεξεργασία των undirected 1:1 ισοδυναμιών που παρήγαγε το Coma++ . Στο σημείο αυτό να παρατηρήσουμε, ότι η παραγωγή των undirected ισοδυναμιών είναι ένα καθαρά ανεξάρτητο στοιχείο του μηχανισμού. Κατά συνέπεια το εργαλείο αυτό μπορεί κάλλιστα να αντικατασταθεί με κάποιο παρόμοιο σύστημα, ή ακόμη και με την εισαγωγή των ισοδυναμιών απευθείας από το χρήστη.

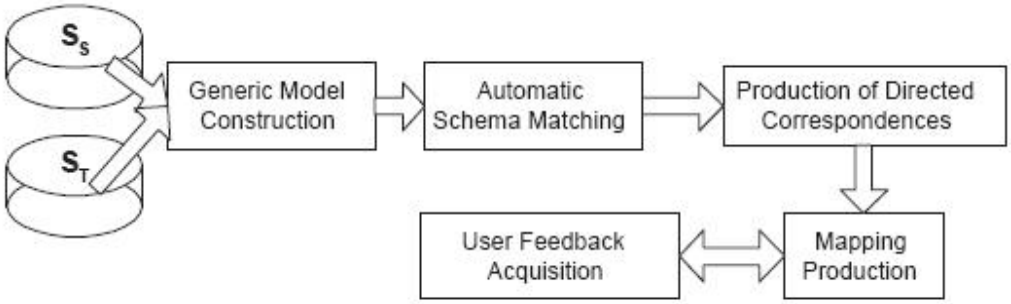
Το σύνολο $C_{\mathcal{L}}(S_T,S_S)$ περιλαμβάνει της directed ισοδυναμίες από το σχήμα S_T στο σχήμα S_S και θα χρησιμοποιηθεί για την παραγωγή των LAV αντιστοιχιών, ενώ το σύνολο $C_{\mathcal{G}}(S_T,S_S)$ περιλαμβάνει τις directed ισοδυναμίες από το σχήμα S_S στο σχήμα S_T και θα χρησιμοποιηθεί για την παραγωγή των GAV αντιστοιχιών.

Σε επόμενο στάδιο, η διαδικασία Mapping Construction αρχικοποιεί για κάθε πίνακα των σχημάτων S_T και Ss μια αντιστοιχία, συνδυάζοντας τις κατάλληλες ισοδυναμίες στοιχείων, και τα κατάλληλα μονοπάτια συνένωσης από τα μοντέλα των σχημάτων, παράγοντας έτσι δυο σύνολα από GAV και LAV mappings, M_G και M_L αντίστοιχα. Έτσι παράγονται οι αρχικές αντιστοιχίες των δύο σχημάτων και τώρα πλέον ο τοπικός κόμβος Β, μπορεί να μεταφράσει τα εισερχόμενα ερωτήματα του κόμβου Α, πάνω στο δικό του σχήμα.

Παρόλα αυτά, μια αντιστοιχία ενδέχεται να είναι λανθασμένη. Λάθη κατά τη δημιουργία της αντιστοιχίας δύνανται να συμβούν λόγω των παρακάτω:

- Κάποιες από τις ισοδυναμίες που χρησιμοποιήθηκαν ήταν σημασιολογικά λανθασμένες, παρόλο που ο βαθμός βεβαιότητας τους ήταν σχετικά υψηλός
- Κάποιο από τα μονοπάτια συνένωσης που χρησιμοποιήθηκαν στην αντιστοιχία ήταν σημασιολογικά λανθασμένο και πρέπει να αντικατασταθεί με κάποιο άλλο μονοπάτι συνένωσης
- Για ορισμένες ιδιότητες του πίνακα της αντιστοιχίας, δε βρέθηκε κάποια ισοδυναμία
- Για ορισμένες ιδιότητες του πίνακα της αντιστοιχίας δε βρέθηκε κάποιο μονοπάτι συνένωσης.

Παρατηρούμε λοιπόν, ότι για να βελτιώνονται σταδιακά οι αντιστοιχίες που κατασκευάστηκαν αρχικά, το σύστημα πρέπει να παρέχει στο χρήστη, τη δυνατότητα να κρίνει τις αντιστοιχίες αυτές, σημειώνοντας τις ισοδυναμίες και μονοπάτια συνένωσης που θεωρεί σωστά, αλλά και τα στοιχεία της αντιστοιχίας που έκρινε ο ίδιος λανθασμένα. Εδώ λοιπόν υπεισέρχεται ο ρόλος της διεπαφής χρήστη (user interface) με τη ρουτίνα κατασκευής αντιστοιχιών (mapping constructor),η αλληλεπίδραση των οποίων αποτελεί και το τελευταίο στάδιο του μηχανισμού. Τα στοιχεία ανάδρασης που έδωσε ο χρήστης μέσω του κατάλληλου interface, λαμβάνονται υπόψη από τον κατασκευαστή των mappings, και οι αντιστοιχίες βελτιώνονται.



Σχήμα 4 b: Πορεία Κατασκευής των Αντιστοιχιών

Στο σχήμα 4 b, φαίνεται η πορεία που ακολουθείται από τον μηχανισμό για την αρχική εύρεση και μετέπειτα βελτίωση των αντιστοιχιών. Στη συνέχεια περιγράφεται διεξοδικά ο τρόπος με τον οποίο σχεδιάσθηκε κάθε επιμέρους ρουτίνα του σχήματος 4 b και αναλύεται διεξοδικά η λειτουργία και ο σκοπός της.

4.2 Κατασκευή Γενικού Μοντέλου ενός σχεσιακού σχήματος

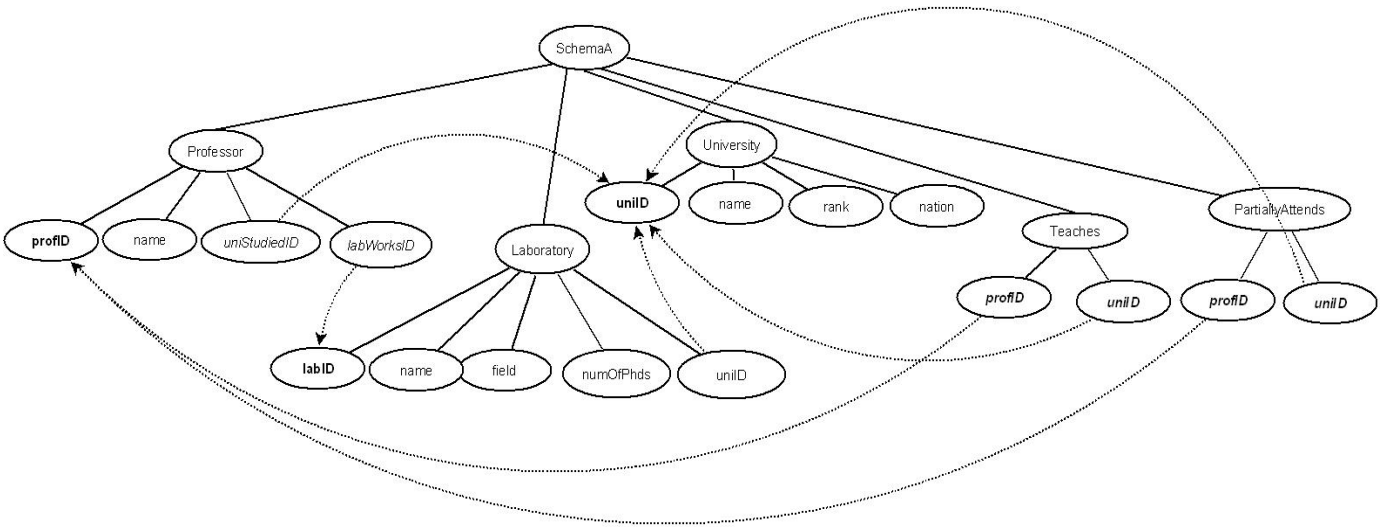
Όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο, ο ρόλος της διαδικασίας αυτής είναι να κατασκευάζει το πλαίσιο που θα εμπεριέχει όλες τις πληροφορίες για το σχεσιακό σχήματα του τοπικού κόμβου (local schema) και του απομακρυσμένου κόμβου (global schema). Δεδομένου ότι σε συστήματα ομότιμων βάσεων δεδομένων, ο κάθε κόμβος έχει πλήρη άγνοια για τον τρόπο που αναπαριστώνται τα δεδομένα σε μια απομακρυσμένη βάση (απόλυτα κατανεμημένη αρχιτεκτονική), το μοντέλο που θα αντιπροσωπεύει το απομακρυσμένο σχήμα οφείλει να είναι δυναμικό και αναπροσαρμόσιμο, ώστε νέες πληροφορίες να ενσωματώνονται στο μοντέλο κάθε φορά που ένα ερώτημα του απομακρυσμένου κόμβου καταφθάνει.

Υπενθυμίζουμε ότι τα χαρακτηριστικά ενός γενικού μοντέλου σχήματος είναι τα εξής:

- Τα ονόματα όλων των πινάκων του σχήματος
- Τα ονόματα όλων των ιδιοτήτων του σχήματος
- Τους περιορισμούς εξωτερικών κλειδιών
- Τους περιορισμούς κύριων κλειδιών
- Γενικούς περιορισμούς σχήματος
- Το σύνολο με τους συνδέσμους που προκύπτουν από τα εξωτερικά κλειδιά, για κάθε συνδυασμό δυο πινάκων του σχήματος
- Το σύνολο με τα πιθανά μονοπάτια συνένωσης μέχρι κάποιο συγκεκριμένο μήκος για κάθε συνδυασμό δυο πινάκων του σχήματος

Όπως παρατηρούμε όλα τα χαρακτηριστικά του μοντέλου εκτός του τελευταίου μπορούν να εξαχθούν πολύ εύκολα από τα metadata της τοπικής βάσης, ή από τα εισερχόμενα ερωτήματα μιας απομακρυσμένης βάσης. Ενδιαφέρον παρουσιάζει όμως η κατασκευή του συνόλου με τα πιθανά μονοπάτια συνένωσης για ένα ζεύγος πινάκων του σχήματος.

Ας θεωρήσουμε για παράδειγμα το ακόλουθο σχήμα που φαίνεται σε μορφή δένδρου.



Σχήμα 4 c: Ένα πιθανό σχήμα για κάποιο πανεπιστήμιο (Σχήμα Α)

Στο παραπάνω σχήμα Α έχουμε 4 πίνακες και συνεπώς προκύπτουν 4+(4*3)/2 = 10 ζεύγη πινάκων (συμπεριλαμβανομένων και των ζευγών πινάκων με τον εαυτό τους). Οι περιορισμοί εξωτερικών κλειδιών φαίνονται στο σχήμα 4 c με διακεκομμένες γραμμές και συνιστούν τους έγκυρους συνδέσμους που γίνονται μεταξύ των πινάκων του σχήματος. Για κάθε ένα από τα ζεύγη αυτά, ενδιαφερόμαστε να βρούμε τους τρόπους με τους οποίους οι πίνακες του ζεύγους συνενώνονται μεταξύ τους, με άλλα λόγια να βρούμε τα διαφορετικά μονοπάτια συνένωσης που ενδέχεται να υπάρχουν για τους δύο πίνακες. Τα μονοπάτια συνένωσης μήκους 1, κατασκευάζονται πολύ απλά, μιας και αποτελούνται μόνο από ένα σύνδεσμο. Συνεπώς για ν συνδέσμους στο σχήμα (ν περιορισμούς εξωτερικών κλειδιών), κατασκευάζουμε τα ν αντίστοιχα join paths μήκους 1. Έτσι για τα παρακάτω 7 επιλεγμένα ζεύγη πινάκων, έχουμε τα εξής join paths μήκους 1:

Table Pair	**Join Paths of Length 1**
Professor-Laboratory	*Professor.labWorksID = Laboratory.labID*
Professor-University	*Professor.uniStudiedID = University.uniID*
Laboratory-University	*Laboratory.uniID = University.uniID*
Teaches-University	*Teaches.uniID = University.uniID*
PartiallyAttends-Professor	*PartiallyAttends.profID = Professor.profID*
Teaches-Professor	*Teaches.profID = Professor.profID*
PartiallyAttends-University	*PartiallyAttends.uniID = University.uniID*

Πίνακας 4 a: Μονοπάτια μοναδιαίου μήκους για το σχήμα Α

Τα παραπάνω μονοπάτια μπορούσαν να έχουν προέλθει (εκτός από περιορισμούς εξωτερικών κλειδιών) και από τα joins των ερωτημάτων του χρήστη.

Όμως υπάρχουν παραπάνω τρόποι που δύνανται δύο πίνακες να συνδέονται μεταξύ τους, και κατά συνέπεια να φανερώνουν σημασιολογικές σχέσεις μεταξύ των δυο πινάκων. Παραδείγματος χάριν, ο πίνακας Professor συνδέεται με τον πίνακα University με τρία ακόμη μονοπάτια, τα οποία προκύπτουν έμμεσα από τις πληροφορίες του σχήματος. Αυτά είναι τα εξής.

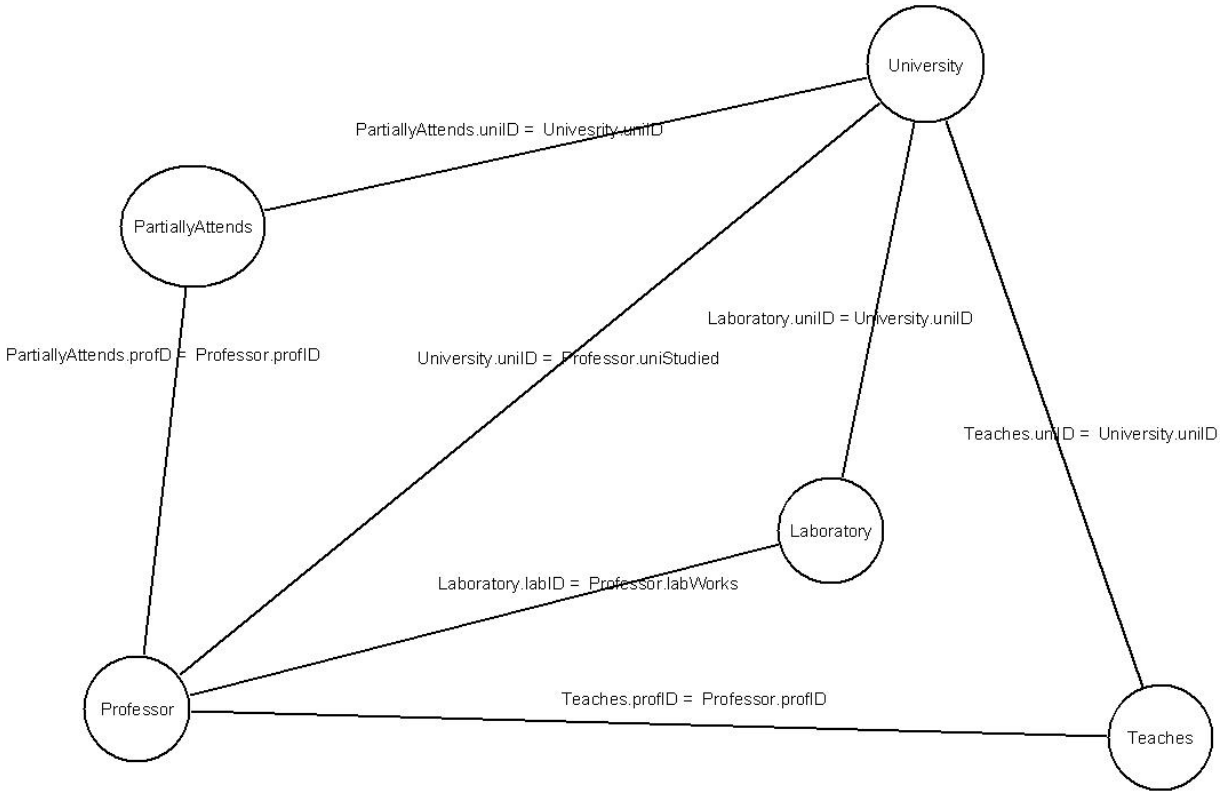
- Professor.profId = Teaches.profID and Teaches.uniID = University.uniID (το πανεπιστήμιο στο οποίο διδάσκει ένας καθηγητής)
- Professor.profID = PartiallyAttends.profID and PartiallyAttends.uniID = University.uniID (το πανεπιστήμιο το οποίο επισκέπτεται ο καθηγητής)
- Professor.labworksID = Laboratory.labID and Laboratory.uniID = University.uniID (το πανεπιστήμιο που ανήκει το εργαστήριο που εργάζεται ο καθηγητής)

Παρατηρούμε ότι κάθε μονοπάτι συνένωσης φανερώνει και μια διαφορετική σημασιολογική σύνδεση των δύο πινάκων και για το λόγο αυτό οφείλουμε να βρούμε όλες τις πιθανές συνδέσεις για κάθε ζεύγος πινάκων του σχήματος, μιας και αργότερα κατά την αρχικοποίηση και βελτίωση των αντιστοιχιών, ο μηχανισμός θα πρέπει να είναι σε θέση να προσφέρει διαφορετικές σημασιολογικές προσεγγίσεις στην αντιστοιχία, μέχρι να ανακαλύψει τη σωστή.

Ο τρόπος με τον οποίο κατασκευάζονται τα μονοπάτια συνένωσης μήκους μεγαλύτερου του 1 μπορεί να κατανοηθεί καλύτερα, ανάγοντας το εκάστοτε σχήμα σε έναν γράφο.

Ορισμός 4.5

Ορίζουμε ως γράφο ενός σχήματος S, το γράφο εκείνο με σύνολο κόμβων το σύνολο των σχέσεων του σχήματος S, στον οποίο υπάρχει μια ακμή Eij, μεταξύ των κόμβων Ri και Rj, για κάθε περιορισμό εξωτερικού κλειδιού του σχήματος S της μορφής Ri.ai → Rj ή Rj.aj→Ri και για κάθε σύνδεσμο Ri.ak = Rj.al που εμπεριέχεται στα ερωτήματα του απομακρυσμένου κόμβου, σε περίπτωση που το σχήμα S είναι το απομακρυσμένο σχήμα.



Σχήμα 4 d: Γράφος του Schema A

Το σύνολο των join paths (n), n>1 μεταξύ δύο πινάκων T1 και T2, κατασκευάζεται αναδρομικά, και ισοδυναμεί με την εύρεση όλων των μονοπατιών μήκους n από τον κόμβο T1 στον κόμβο T2. Για να βρούμε ένα μονοπάτι συνένωσης μεταξύ του πίνακα T1 και T2, με μήκος n, αρκεί να βρούμε κάποιο μονοπάτι συνένωσης του πίνακα T1 με κάποιον πίνακα A μήκους 1 και μετά να ψάξουμε αναδρομικά για μονοπάτι συνένωσης μήκους n-1 μεταξύ του πίνακα A και του πίνακα T2. Το τελικό μονοπάτι προκύπτει απλά συμπτύσσοντας τα δύο μονοπάτια μήκους 1 και n-1 αντίστοιχα. Επαναλαμβάνοντας την παραπάνω διαδικασία θέτοντας κάθε φορά ως πίνακα A έναν διαφορετικό πίνακα του σχήματος, και ανακαλύπτοντας όλα τα μονοπάτια μήκους 1 μεταξύ του T1 και του A, βρίσκουμε όλα τα join paths (n) για τους πίνακες T1, T2. Ας δούμε παραδείγματος χάριν, την εύρεση των μονοπατιών συνένωσης μήκους 2, μεταξύ των πινάκων *Professor* και *University* του σχήματος Schema A, του οποίου ο γράφος φαίνεται στο σχήμα 4 d. Αρχικά θέτουμε ως πίνακα A τον πίνακα *PartiallyAttends*. Μεταξύ του πίνακα Professor και *PartiallyAttends* υπάρχει ένα μόνο μονοπάτι μήκους 1 (*PartiallyAttends.profID = Professor.profID*). Κρατάμε αυτό το μονοπάτι και τώρα ψάχνουμε για κάποιο μονοπάτι συνένωσης μεταξύ του πίνακα *PartiallyAttends* και του πίνακα *University* μήκους 2-1 = 1.Και πάλι, υπάρχει ένα μονοπάτι μήκους 1 (*PartiallyAttends.uniID = University.uniID*). Οπότε το τελικό μονοπάτι προκύπτει από την σύμπτυξη των δύο επιμέρους join paths και είναι το εξής *Professor.profID* =

PartiallyAttends.profID and PartiallyAttends.uniID = University.uniID. επαναλαμβάνοντας την παραπάνω διαδικασία θέτοντας ως πίνακα Α τους πίνακες *Laboratory* και *Teaches*, ανακαλύπτουμε δύο επιπλέον μονοπάτια συνένωσης μήκους 2, και τα αποθηκεύουμε στο σύνολο μονοπατιών συνένωσης του ζεύγους *Professor-University*.

Να σημειωθεί ότι ο αριθμός των πιθανών μονοπατιών συνένωσης αυξάνεται εκθετικά με το μήκος των join paths που ψάχνουμε. Για παράδειγμα αν έχουμε μια αλυσίδα από N κόμβους και κάθε κόμβος συνδέεται με X ακμές με τον επόμενο, σε μήκος m, θα έχουμε X^m μονοπάτια συνένωσης. Για το λόγο αυτό, ψάχνουμε για μονοπάτια συνένωσης μεγίστου μήκους συνήθως 4 ή 5. Άλλωστε, ας μην ξεχνάμε ότι η εύρεση των join paths αποσκοπεί στην ανακάλυψη σημασιολογικών συνδέσεων μεταξύ των δύο πινάκων. Επειδή η σχεδίαση των βάσεων δεδομένων, διέπεται από ορισμένες κοινές αρχές και πολιτικές, περιμένουμε ότι οι εννοιολογικές συνδέσεις δύο πινάκων σε ένα σχήμα εκφράζονται με μονοπάτια συνένωσης μικρού μήκους.

Παραγωγή Μονοπατιών συνένωσης μήκους n μεταξύ δύο πινάκων T1, T2 ενός σχήματος S

Input: Το γενικό μοντέλο του σχήματος S, $Model_S$

Output: Το σύνολο όλων των μονοπατιών συνένωσης μεταξύ των δύο πινάκων με μήκος n

AllJps = NULL

If (n == 1)

Για κάθε join του συνόλου των συνδέσμων του ζεύγους T1,T2, εκτέλεσε τα παρακάτω:

Αρχικοποίησε ένα join path, JP με μόνο αυτό το join

Πρόσθεσε το JP στο σύνολο AllJps

Επέστρεψε το AllJps

Else

Για κάθε πίνακα A, στο $Model_S$, εκτέλεσε τα παρακάτω:

Βρες τα join paths μήκους 1 μεταξύ του πίνακα T1 και A

Αποθήκευσε τα join paths που βρέθηκαν στο σύνολο Set1

Βρες τα join paths μήκους n-1 μεταξύ του πίνακα A και T2

Αποθήκευσε τα join paths που βρέθηκαν στο σύνολο Set2

Για κάθε join path JP1, του συνόλου Set1

Για κάθε join path JP2 του συνόλου Set2

Αν ο πρώτος πίνακας του JP2 είναι ίδιος με τον τελευταίο πίνακα του JP1

Σύμπτυξε τα JP1 & JP2 στο νέο join path JPfinal

Πρόσθεσε το JPfinal στο σύνολο AllJps

Επέστρεψε το AllJps

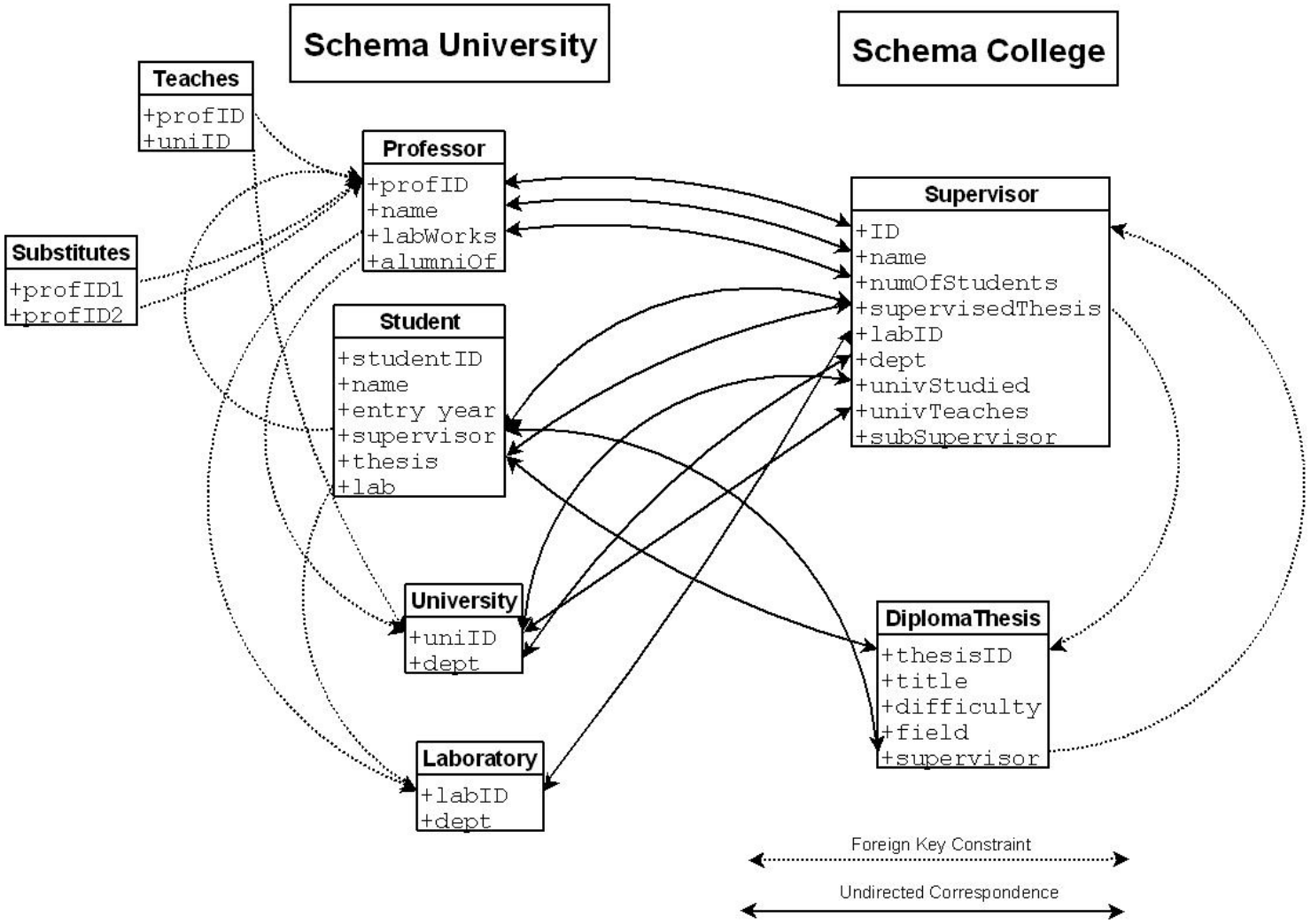
Αλγόριθμος 4 a: Αλγόριθμος Παραγωγής Μονοπατιών συνένωσης μεταξύ πινάκων T1,T2 μήκους n

4.3 Εξαγωγή κατευθυνόμενων ισοδυναμιών μεταξύ των δύο σχημάτων

Ύστερα από την κατασκευή των γενικών μοντέλων και για τα δύο σχήματα, ακολουθεί η διαδικασία που θα κατασκευάζει τα τελικά σύνολα ισοδυναμιών μεταξύ των δύο σχημάτων, που θα χρησιμοποιηθούν για την παραγωγή των αντιστοιχιών. Για τη λειτουργία αυτή, κρίθηκε αναγκαία, όπως αναφέρθηκε προηγουμένως η βοήθεια ενός automatic schema matching tool (Coma++). Το εργαλείο αυτό λαμβάνει ως είσοδο δύο σχεσιακά (στην περίπτωσή μας) σχήματα S_T, S_S και εξάγει ένα σύνολο από πιθανές undirected ισοδυναμίες (στο εξής $C_{\mathcal{U}}(S_T,S_S)$). Το τελευταίο σύνολο διασπάται σε δύο σύνολα από directed ισοδυναμίες $C_{\mathcal{G}}(S_T,S_S)$ και $C_{\mathcal{L}}(S_T,S_S)$. Το πρώτο από αυτά τα δύο σύνολα ισοδυναμιών θα χρησιμοποιηθεί για την παραγωγή των GAV αντιστοιχιών, ενώ το δεύτερο θα χρησιμοποιηθεί για την παραγωγή των LAV αντιστοιχιών ανάμεσα στα δύο σχήματα.

Κάθε undirected ισοδυναμία $C_U(A_T,A_S) \in C_{\mathcal{U}}$ παράγει δύο directed ισοδυναμίες $C_G(A_T,A_S) \in C_{\mathcal{G}}$, $C_L(A_T,A_S) \in C_{\mathcal{L}}$, οι οποίες διατηρούν τον βαθμό βεβαιότητας της αρχικής. Θα μπορούσαμε να χρησιμοποιήσουμε κατευθείαν τα δύο αυτά σύνολα για την αρχικοποίησης των GAV/LAV αντιστοιχιών. Παρόλα αυτά, η γνώση για τους περιορισμούς εξωτερικών κλειδιών των δύο σχημάτων μπορεί να αξιοποιηθεί, ώστε τα δύο σύνολα $C_{\mathcal{L}}$ και $C_{\mathcal{G}}$ να εμπλουτιστούν με επιπλέον 1:1 ισοδυναμίες. Ας δούμε ένα παράδειγμα:

Στο σχήμα 4 e βλέπουμε δύο σχεσιακά σχήματα με τους περιορισμούς εξωτερικών κλειδιών και τα undirected correspondences που παρήγαγε ο Automatic Matcher Coma++ για τα δύο αυτά σχήματα.



Σχήμα 4 e: Ένα σενάριο παραγωγής αντιστοιχιών για δύο σχεσιακά σχήματα $S_{university}$ και $S_{college}$

Προς χάριν απλότητας στο παραπάνω παράδειγμα παραλείψαμε τις βαθμολογίες των αντιστοιχιών.

Ας θεωρήσουμε ότι ο Automatic Schema Matcher δεν είχε καταφέρει να ανακαλύψει την σωστή undirected ισοδυναμία C_U(Supervisor.ID, Professor.profID) μιας και οι λέξεις Professor και Supervisor δεν μοιάζουν λεξικογραφικά. Αντί αυτού, θεώρησε σωστή την ισοδυναμία C_U(Supervisor.ID, Student.supervisor), λόγω της ομοιότητας των λέξεων. Η undirected ισοδυναμία, στη συνέχεια αναλύεται σε δύο directed ισοδυναμίες, C_G(Supervisor.ID, Student.supervisor) και C_L(Student.supervisor, Supervisor.ID), όπου παρατηρούμε ότι η πρώτη ισοδυναμία δεν είναι και η καλύτερη δυνατή μιας και το ID ενός Supervisor ισοδυναμεί καλύτερα με το ID ενός Professor και όχι μόνο με το ID των καθηγητών που επιβλέπουν κάποιον φοιτητή (εν γένει ένας Supervisor μπορεί να μην

επιτηρεί κανέναν φοιτητή). Γνωρίζουμε όμως ότι η ιδιότητα Student.supervisor είναι εξωτερικό κλειδί στην ιδιότητα Professor.profID, οπότε στην προκειμένη περίπτωση το σύνολο C_G, μπορεί να εμπλουτιστεί με την directed ισοδυναμία, C_G(Supervisor.ID,Professor.profID).

Γενικά λοιπόν μπορούμε να πούμε ότι για μια directed correspondence C_D(A.a,B.b) όπου B.b → T.key είναι περιορισμός εξωτερικού κλειδιού στο αντίστοιχο σχήμα, προσθέτουμε στο σύνολο κατευθυνόμενων ισοδυναμιών την ισοδυναμία C_D(A.a,T.key) και θέτουμε την τιμή της $|C_D(A.a,T.key)| = |C_D(A.a,B.b)|$. Παρακάτω φαίνεται ο αλγόριθμος κατασκευής των δύο συνόλων κατευθυνόμενων αντιστοιχιών $C_{\mathcal{G}}$ και $C_{\mathcal{I}}$.

Παραγωγή κατευθυνόμενων ισοδυναμιών μεταξύ δύο σχημάτων S1, S2

Input: Το γενικό μοντέλο του σχήματος S1, $Model_{S1}$

Το γενικό μοντέλο του σχήματος S2, $Model_{S2}$

Το σύνολο $C_{\mathcal{U}}(S_1, S2)$ των ισοδυναμιών χωρίς κατεύθυνση μεταξύ των δύο σχημάτων, που παρήγαγε ο Matcher (Coma++)

Output: Το σύνολο $C_{\mathcal{G}}$, $(S_1, S2)$ με τις κατευθυνόμενες ισοδυναμίες από το σχήμα S1 στο σχήμα S2.

Το σύνολο $C_{\mathcal{I}}$, (S_1, S_2) με τις κατευθυνόμενες ισοδυναμίες από το σχήμα S2 στο σχήμα S1.

$C_{\mathcal{I}}$, $(S_1, S2) = C_{\mathcal{G}}$, $(S_1, S2) = NULL$

Για κάθε ισοδυναμία $C_U(A1,A2)$ του συνόλου $C_{\mathcal{U}}(S_1,S_2)$:

Πρόσθεσε την ισοδυναμία $C_G(A1,A2)$ στο σύνολο $C_{\mathcal{G}}$, $(S_1, S2)$

Θέσε την τιμή της ισοδυναμίας σε: $|C_G(A1,A2)| = |C_U(A1,A2)|$

Πρόσθεσε την ισοδυναμία $C_L(A2,A1)$ στο σύνολο $C_{\mathcal{I}}$, $(S_1, S2)$

Θέσε την τιμή της ισοδυναμίας σε: $|C_L(A2,A1)| = |C_U(A1,A2)|$

Για κάθε κατευθυνόμενη ισοδυναμία $C_G(A1,A2)$ στο σύνολο $C_{\mathcal{G}}$, $(S_1, S2)$

Αν η ιδιότητα A2 είναι εξωτερικό κλειδί στο σχήμα S2

Βρες τον περιορισμό A2→X από το $Model_{S2}$

Πρόσθεσε την ισοδυναμία $C_G(A1,X)$ στο σύνολο $C_{\mathcal{G}}$, $(S_1, S2)$

Θέσε την τιμή της ισοδυναμίας σε: $|C_G(A1,X)| = |C_G(A1,A2)|$

Για κάθε κατευθυνόμενη ισοδυναμία $C_L(A2,A1)$ στο σύνολο $C_{\mathcal{I}}$, $(S_1, S2)$

Αν η ιδιότητα A1 είναι εξωτερικό κλειδί στο σχήμα S1

Βρες τον περιορισμό A1→X από το $Model_{S1}$

Πρόσθεσε την ισοδυναμία $C_G(A2,X)$ στο σύνολο $C_{\mathcal{I}}$, $(S_1, S2)$

Θέσε την τιμή της ισοδυναμίας σε: $|C_G(A2,X)| = |C_G(A2,A1)|$

Επέστρεψε τα σύνολα $C_{\mathcal{I}}$, $(S_1, S2)$ και $C_{\mathcal{G}}$, $(S_1, S2)$

Αλγόριθμος 4 b: Κατασκευή των δυο συνόλων directed ισοδυναμιών

4.4 Αρχική κατασκευή των GAV και LAV αντιστοιχιών

Βάσει των προηγούμενων υποκεφαλαίων, μπορούμε τώρα να περιγράψουμε την κύρια ρουτίνα του μηχανισμού εύρεσης αντιστοιχιών, η οποία στο σχήμα 4 b φαίνεται με το όνομα Mapping Construction. Όπως εξηγήσαμε προηγουμένως, στόχος μας είναι η παραγωγή δύο συνόλων αντιστοιχιών M_G και M_L για τις GAV αντιστοιχίες και LAV αντιστοιχίες αντίστοιχα. Μια αντιστοιχία M του συνόλου M_G, ή του συνόλου M_L αφορά μια σχέση R_S του σχήματος S_S που αντιστοιχίζεται σε ένα σύνολο σχέσεων του σχήματος S_T, συνδυάζοντας κατευθυνόμενες ισοδυναμίες από το σύνολο $C_{\mathcal{G}}$, (S_S, S_T) ή από το σύνολο $C_{\mathcal{L}}$, (S_S, S_T) αντίστοιχα. Επειδή η μόνη διαφορά της εύρεσης μιας αντιστοιχίας GAV από μια αντιστοιχία LAV έγκειται μόνο στην εναλλαγή των ρόλων των δύο σχημάτων ως Target και Source Schema (για GAV αντιστοιχίες έχουμε Source Schema → Global Schema και Target Schema→ Local Schema ενώ για LAV αντιστοιχίες αντίστροφα) και στην χρησιμοποίηση ενός διαφορετικού συνόλου από κατευθυνόμενες ισοδυναμίες, θα επικεντρωθούμε μόνο στη διαδικασία κατασκευής μιας αντιστοιχίας M ενός πίνακα του Source Schema,που αναφέρεται σε στοιχεία του target schema και συνδυάζει ισοδυναμίες από ένα σύνολο C(είτε το σύνολο $C_{\mathcal{G}}$, (S_S, S_T) ή το σύνολο $C_{\mathcal{L}}$, (S_S, S_T)), χωρίς πλέον να μας απασχολεί ο τύπος αυτής.

4.4.1 Τρόπος αναπαράστασης των αντιστοιχών

Κάθε αντιστοιχία M που αφορά μια σχέση $S_S.R_S$ δεν παραμένει στατική, αντίθετα καθώς ο χρήστης δίνει ανάδραση για την ποιότητα της αντιστοιχίας, αυτή θα πρέπει να μετεξελίσσεται και σταδιακά να βελτιώνεται μέχρι το σημείο που ο χρήστης θα κρίνει ότι μπορεί να θεωρηθεί μια ισχυρή/πλήρης αντιστοιχία. Ακόμη, ας μην ξεχνάμε ότι παρόλο που στην ανάλυση μας, προϋποθέτουμε την πληρότητα των δύο σχημάτων, ο μηχανισμός αυτός δύναται να χρησιμοποιηθεί και σε περιβάλλοντα p2p βάσεων δεδομένων, όπου τα σχήματα ενδέχεται να είναι σε ημιτελή μορφή (πχ ορισμένες ιδιότητες μιας σχέσης να μην είναι ακόμη γνωστές). Για τους παραπάνω λόγους, οφείλουμε να αναπαραστήσουμε την εκάστοτε αντιστοιχία M με μια δυναμική δομή, η οποία θα μπορεί να αλλάζει και κατά επέκταση, οι αλλαγές αυτές να φαίνονται και στην αντιστοιχία. Έτσι λοιπόν ορίζουμε μια δυναμική δομή, η οποία θα κρατά την τελευταία μορφή της αντιστοιχίας, αλλά και την ιστορία των αλλαγών της, και θα μπορεί οποιαδήποτε στιγμή να μετατρέψει την αντιστοιχία σε SQL όψη. Ονομάζουμε αυτή τη δομή αναπαράστασης της αντιστοιχίας M, Matrix.

Ορισμός 4.1

Η δομή Matrix(R_S,S_T) αποτελείται από το παρακάτω σύνολο στοιχείων. Για κάθε ιδιότητα $R_S.A$:

- ✓ *Την επιλεγμένη κατευθυνόμενη ισοδυναμία C_{D_A}(Rs.A,S_T.R_T.X)*
- ✓ *Ένα ψευδώνυμο ($Alias_A$) για την R_T αναφορικά με την ισοδυναμία C_{D_A} που επιλέχθηκε*
- ✓ *Ένα σύνολο με πιθανές ισοδυναμίες για την ιδιότητα $R_S.A$, $C_{D\_A\_possible}$*
- ✓ *Ένα σύνολο με λανθασμένες ισοδυναμίες για την ιδιότητα $R_S.A$, $C_{D\_A\_bad}$*
- ✓ *Το joinpath που επιλέχθηκε για την επιλεγμένη ισοδυναμία, JP_A*
- ✓ *Ένα σύνολο με λανθασμένα joinpaths για την επιλεγμένη ισοδυναμία JP_{BAD_A}*

Οποιαδήποτε στοιχείο του παραπάνω συνόλου μπορεί να μην έχει τιμή.

Συνοπτικά για κάθε ιδιότητα A της σχέσης R_S το σύνολο C περιλαμβάνει έναν αριθμό ισοδυναμιών, που δείχνουν με ποιες ιδιότητες του σχήματος S_T, μπορεί να αντιστοιχηθεί η A (Υπενθυμίζουμε ότι το Coma++ ενδέχεται να παράγει πολλές ισοδυναμίες για μια ιδιότητα, με διαφορετικό πιθανώς βαθμό βεβαιότητας, προσφέροντας έτσι διάφορες επιλογές για το πώς μπορεί να αντιστοιχηθεί η σημασιολογική έννοια της ιδιότητας A στο σχήμα S_T). Όλες αυτές οι ισοδυναμίες αποτελούν το σύνολο $C_{D\_A\_possible}$ της ιδιότητας A. Η ισοδυναμία που επιλέγεται από τον μηχανισμό για να αντιστοιχίσει την ιδιότητα A, είναι μια ισοδυναμία αυτού του συνόλου. Ενδέχεται όμως η επιλεγμένη ισοδυναμία να κριθεί λανθασμένη από το χρήστη, οπότε αυτή τοποθετείται στο σύνολο $C_{D\_A\_bad}$ ώστε έτσι να μη γίνει για δεύτερη φορά το ίδιο λάθος. Το στοιχείο JP_A μιας ιδιότητας αποτελεί το μονοπάτι συνένωσης που επιλέχθηκε από τον μηχανισμό, αν κριθεί απαραίτητο ότι η επιλεγμένη ισοδυναμία απαιτεί την προσθήκη ορισμένων συνδέσμων στην αντιστοιχία, προκειμένου να είναι σημασιολογικά ορθή. Τέλος το σύνολο JP_{BAD_A} περιέχει όλα εκείνα τα μονοπάτια συνένωσης που χρησιμοποιήθηκαν στην αντιστοιχία για την αντιστοίχηση της ιδιότητας A, αλλά κρίθηκαν λανθασμένα από το χρήστη. Περισσότερα για τον τρόπο που επιλέγεται η ισοδυναμία κάθε η ιδιότητας, για την απόφαση αν απαιτείται κάποιο επιπλέον μονοπάτι συνένωσης μεταξύ δύο πινάκων, ώστε η επιλεγμένη ισοδυναμία να είναι σωστή, και για τον τρόπο εύρεσης του κατάλληλου μονοπατιού συνένωσης, θα δούμε στις επόμενες υποπαραγράφους. Στον πίνακα 4 α φαίνεται ο τρόπος που θα οπτικοποιούμε στο εξής τη δομή Matrix μιας αντιστοιχίας M, μέσω ενός παραδείγματος για την αντιστοιχία του πίνακα Supervisor από το σχήμα 4 e. Στο εξής, θα αναφερόμαστε σε κάθε γραμμή του πίνακα Matrix, που αφορά μια ιδιότητα $R_S.A$ ως στοιχείο αντιστοιχίας της ιδιότητας $R_S.A$(Mapping Element ($R_S.A$)).

Ιδιότητα(A)	C_{D_A}	$Alias_A$	$C_{D\_A\_possible}$	$C_{D\_A\_BAD}$	JP_A	JP_{BAD_A}
ID	Professor.profID	-	Student.supervisor	-	-	-
name	Professor.name	-	-	-	-	-
numOfStudents	Professor.labWorks	-	-	-	-	-
supervisedThesis	Student.supervisor	-	Student.thesis	-	Professor.profid = Student.supervisor	-
labID	Laboratory.labID	-	Student.lab	-	Professor.labworks = Laboratory.labID	-
dept	University.dept	-	Laboratory.dept	-	Professor.alumniOf = Universit.uniID	-
univStudied	University.uniID	-	-	-	Professor.alumniOf = University.uniID	-
univTeaches	University.uniID	Univ$1	-	-	Professor.profID = Teaches.profID and Teaches.uniID = Univ$1.uniID	-
subSupervisor	Professor.profID	Prof$1	Student.supervisor	-	Professor.profID = Substitutes.profID1 and Substitutes.profID2 = Prof$1.profID	-

Πίνακας 4 b: Η δομή Matrix για την αντιστοιχία του πίνακα Supervisor κατά το πρώτο στάδιο δημιουργίας της

Το αντίστοιχο SQL ερώτημα-όψη της παραπάνω αντιστοιχίας φαίνεται παρακάτω:

Create View Supervisor (ID, name, numOfStudents, supervisedThesis, labID,dept, uniStudied, uniTeaches, subSupervisor) as

Select

Professor.profID as *ID*, Professor.name as *name*, Professor.labWorks as *numOfStudents*, Student.supervisor as *supervisedThesis*, Laboratory.labID as *labID*, University.dept as *dept*, University.uniID as *uniStudied*, Univ$1.uniID as *uniTeaches*, Prof$1.profID as *subSupervisor*

From

Professor, Student, University, Laboratory, Teaches, Substitutes, Univ$1 as University, Prof$1 as Professor

Where

Student.supervisor = Professor.profID and University.uniID = Professor.alumniOf and Professor.labWorks = Laboratory.labID and Professor.profId = Teaches.profID and Teaches.uniID = Univ$1.uniID and Professor.profID = Substitutes.profID1 and Substitutes.profID2 = Prof$1.profID

Όπως παρατηρούμε στο παραπάνω παράδειγμα, οι ισοδυναμίες που επιλέχθηκαν για ορισμένες ιδιότητες ήταν σωστές (πχ Supervisor.ID → Professor.profID, Supervisor.name→Professor.name κλπ), ενώ για άλλες ήταν λανθασμένες και χρειάζονται αντικατάσταση (πχ Supervisor.numOfStudent → Professor.labWorks, Supervisor.supervisedThesis → Student.supervisor). Ακόμη, για κάποιες ιδιότητες η ισοδυναμία που επιλέχθηκε, κρίθηκε ότι απαιτούσε την εισαγωγή κάποιου μονοπατιού συνένωσης στην αντιστοιχία (οι ισοδυναμίες των έξι τελευταίων ιδιοτήτων του πίνακα). Από αυτές τις ισοδυναμίες, για κάποιες το join path που βρέθηκε είναι ικανοποιητικό και εκφράζει της σημασιολογία της ισοδυναμίας σωστά, ενώ για άλλες πρέπει να αντικατασταθεί με κάποιο άλλο μονοπάτι συνένωσης. Παραδείγματος χάριν, το μονοπάτι συνένωσης που χρησιμοποιήθηκε για την ισοδυναμία Supervisor.univteaches →University.uniID μπορεί να θεωρηθεί από έναν χρήστη σωστό, ενώ το μονοπάτι συνένωσης για την ισοδυναμία Supervisor.dept → University.dept είναι λανθασμένο. Το τμήμα που ανήκει ένας Supervisor δεν είναι το τμήμα του πανεπιστημίου που σπούδασε αλλά το τμήμα του πανεπιστημίου που εργάζεται.

4.4.2 Μεθοδολογία αρχικοποίησης των αντιστοιχών

Έχοντας τώρα στη διάθεση μας, τον τρόπο που θα αναπαριστούμε μια δυναμική αντιστοιχία (μια αντιστοιχία που μεταλλάσσεται με το χρόνο), θα δούμε αναλυτικά, τον τρόπο που ο πίνακας Matrix μιας αντιστοιχίας M αρχικοποιείται. Στα παρακάτω υποκεφάλαια θα δούμε αναλυτικά πως επιλέγεται η κατάλληλη ισοδυναμία για μια ιδιότητα της σχέσης της αντιστοιχίας, πως κρίνεται αν για μια ισοδυναμία απαιτείται κάποιο μονοπάτι συνένωσης, και τον τρόπο που επιλέγουμε ποιο μονοπάτι συνένωσης είναι το πλέον κατάλληλο για να χρησιμοποιηθεί.

A) <u>Επιλογή των κατάλληλων ισοδυναμιών</u>

Η πρώτη επιλογή που πρέπει να γίνει από τον μηχανισμό κατά την αρχικοποίηση της αντιστοιχίας M μιας σχέσης R_S του σχήματος S_S είναι να βρεθεί η κατάλληλη ισοδυναμία για κάθε ιδιότητα R_S.A από το σύνολο $C_{D\_A\_POSSIBLE}$ που θα αντιστοιχίζει την ιδιότητα αυτή στην σωστή σημασιολογικά ιδιότητα του σχήματος S_T. Δεδομένου ότι κατά την αρχικοποίηση της αντιστοιχίας, δεν διατίθεται η ανάδραση του χρήστη για το αν μια ισοδυναμία είναι κατάλληλη ή όχι, το μόνο κριτήριο με το οποίο μπορούμε να κρίνουμε την ορθότητα μιας ισοδυναμίας $C_{D\_A}(A,S_T.X)$, είναι ο βαθμός βεβαιότητας της $|C_{D\_A}(A,S_T.X)|$, που μας έδωσε ο Matcher που χρησιμοποιήθηκε (Coma++). Συνεπώς επιλέγουμε την ισοδυναμία εκείνη από το σύνολο $C_{D\_A\_POSSIBLE}$ της ιδιότητας A, που εμφανίζει τον υψηλότερο βαθμό βεβαιότητας.

Ένα βασικό πρόβλημα που μπορεί να προκύψει σε αυτό το στάδιο είναι να επιλεχθούν για δύο διαφορετικές ιδιότητες της σχέσης R_S, A και B, εκείνες οι ισοδυναμίες

που αναφέρονται στην ίδια ιδιότητα X του Target Schema, δηλαδή για R_S.A και R_S.B με A≠B οι ισοδυναμίες $C_{D\_A}(A,S_T.X)$ και $C_{D\_B}(B,S_T.X)$ να εμφανίζουν τον υψηλότερο βαθμό βεβαιότητας στο σύνολο $C_{POSSIBLE}$ της ιδιότητας A και B αντίστοιχα. Σε μια τέτοια περίπτωση, θα ήταν λάθος να χρησιμοποιήσουμε την ίδια ιδιότητα S_T.X ως ισοδύναμη ιδιότητα τόσο της A όσο και της B, καθώς θα είχαμε στον ίδιο πίνακα R_S δύο ιδιότητες οι οποίες θα αντιστοιχούσαν ακριβώς στην ίδια πληροφορία. Κάτι τέτοιο θα αποτελούσε βασικό σχεδιαστικό λάθος, μιας και σχεδόν σε όλες τις περιπτώσεις σχεσιακών βάσεων δεδομένων, θεωρούμε ότι οι ιδιότητες ενός πίνακα αντιστοιχούν σε διαφορετικής φύσεως πληροφορία. Παραδείγματος χάριν δε νοείται δε έναν πίνακα Professor να υπάρχουν δύο ιδιότητες name1 και name2 που και οι δύο θα αποθηκεύουν το όνομα του καθηγητή. Περιπτώσεις όπου ένας πίνακας αποθηκεύει την ίδια πληροφορία σε δύο διαφορετικές ιδιότητες αλλά με διαφορετικό τύπο (πχ μια ιδιότητα να αποθηκεύει την ημερομηνία γέννησης ενός καθηγητή σε String μορφή, και μια δεύτερη να αποθηκεύει την ημερομηνία αυτή σε date μορφή), θεωρούνται εξαιρέσεις και είναι εκτός του σκοπού της παρούσας εργασίας.

Η λύση που δίνεται από τον μηχανισμό σε αυτές τις περιπτώσεις είναι η εξής. Σε κάθε ιδιότητα του πίνακα για την οποία επιλέχθηκε ως ισοδύναμη μια ιδιότητα του target schema που χρησιμοποιείται ήδη από κάποια άλλη γραμμή της δομής Matrix, κρατείται αυτή η ισοδυναμία, αλλά ο πίνακας της ισοδύναμης ιδιότητας μετονομάζεται με ένα νέο ψευδώνυμο(Alias). Η λύση αυτή έχει δύο πλεονεκτήματα συγκριτικά με το να απορρίπταμε την ισοδυναμία και να διαλέγαμε την αμέσως επόμενη σε βαθμό βεβαιότητας από το σύνολο $C_{POSSIBLE}$. Πρώτον εξοικονομείται υπολογιστικός χρόνος για την εύρεση της δεύτερης καλύτερης ισοδυναμίας από το σύνολο $C_{POSSIBLE}$, και δεύτερον και βασικότερο επιτρέπουμε την ύπαρξη self join paths στην αντιστοιχία, αλλά και την ύπαρξη πολλαπλών join paths με τον ίδιο πίνακα. Ας δούμε ένα παράδειγμα για να γίνουν τα παραπάνω περισσότερο κατανοητά:

Στον πίνακα 4 a βλέπουμε δύο περιπτώσεις του σεναρίου αυτού. Πρώτον με τις ισοδυναμίες που επιλέχθηκαν για τις ιδιότητες *univStudied* και *univTeaches* και δεύτερον για τις ισοδυναμίες που επιλέχθηκαν για τις ιδιότητες ID και *subSupervisor*. Στην πρώτη περίπτωση, κρατώντας την ισοδυναμία *univTeached→University.uniID* παρόλο που η ιδιότητα *University.uniID* χρησιμοποιείται ήδη από την *univStudied* επιτρέπουμε στον πίνακα *University* να συμμετέχει στην αντιστοιχία με δύο διαφορετικά μονοπάτια συνένωσης: ένα για το πανεπιστήμιο που σπούδασε ο Supervisor και ένα για το πανεπιστήμιο που διδάσκει ο *Supervisor*. Στη δεύτερη περίπτωση, κρατώντας την ισοδυναμία *subSupervisor→Professor.profID* επιτρέπουμε στον μηχανισμό να ψάξει να βρει το κατάλληλο self join path (join path μεταξύ δύο πινάκων Τα, Τβ όπου Τα = Tb), για τον

πίνακα *Professor*, και να εκφραστεί έτσι η σημασιολογία του *Supervisor* που αναπληρώνει έναν άλλο.

Παρακάτω παρουσιάζεται με μορφή ψευδοκώδικα ο αλγόριθμος για την επιλογή των ισοδυναμιών των ιδιοτήτων μιας σχέσης R_S

Επιλογή της κατάλληλης ισοδυναμίας για μια ιδιότητα Α ενός πίνακα R_S

Input: Η αντιστοιχία Μ του πίνακα R_S, το σύνολο κατευθυνόμενων ισοδυναμιών C, το σύνολο $C_{D\_A\_BAD}$ με τις λανθασμένες ισοδυναμίες της ιδιότητας Α.

Output: Το σύνολο $C_{D\_A\_POSSIBLE}$ της ιδιότητας Α, η ισοδυναμία C_{D_A} που επιλέχθηκε για την ιδιότητα., το ψευδώνυμο της ιδιότητας Α, $Alias_A$

$Alias_A$, $C_{D\_A\_POSSIBLE}$, C_{D_A} = NULL

Για κάθε ισοδυναμία C_D (R.K,T.L) στο σύνολο C:

    Αν R.K = R_S.A πρόσθεσε την ισοδυναμία C_D στο σύνολο $C_{D\_A\_POSSIBLE}$

Ταξινόμησε το σύνολο $C_{D\_A\_POSSIBLE}$ κατά αύξοντα βαθμό βεβαιότητας των ισοδυναμιών που περιέχει

Όσο υπάρχει επόμενη ισοδυναμία στο σύνολο $C_{D\_A\_POSSIBLE}$

    Διάλεξε την επόμενη ισοδυναμία C_D

    Θέσε C_{D_A} = C_D

    Εάν C_{D_A} ανήκει στο σύνολο $C_{D\_A\_BAD}$

        Συνέχισε

    Αλλιώς

        Διέκοψε

Αν υπάρχει ιδιότητα Rs.B που χρησιμοποιεί ισοδυναμία C_{D_B}΄(Rs.B,T.L)

    Δημιούργησε ένα νέο μοναδικό ψευδώνυμο για την ισοδύναμη ιδιότητα της R_S.A, T$i.L όπου i>0

    Θέσε το ψευδώνυμο $Alias_A$ της R_S.A με το νέο αυτό ψευδώνυμο,

    ($Alias_A$ = T$i)

Επέστρεψε τα C_{D_A}, $C_{D\_A\_POSSIBLE}$, $Alias_A$

Αλγόριθμος 4 c: Επιλογή των κατάλληλων ισοδυναμιών

B) <u>Επιλογή των κατάλληλων μονοπατιών συνένωσης</u>

Έχοντας επιλέξει την ισοδυναμία για κάθε ιδιότητα στη σχέση R_S, είμαστε σε θέση να προχωρήσουμε με τον συνδυασμό των ισοδυναμιών αυτών, προκειμένου η συνολική αντιστοιχία της σχέσης R_S να είναι σημασιολογική σωστή. Με τον συνδυασμό ισοδυναμιών, εννοούμε την πρόσθεση των κατάλληλων συνδέσμων στην αντιστοιχία, ώστε όλοι οι πίνακες R_T του target schema που συμμετέχουν σε αυτήν να συνδέονται μεταξύ τους με κάποιο μονοπάτι συνένωσης. Ας δούμε το παράδειγμα της αντιστοιχίας Supervisor, που φαίνεται στον πίνακα 4 a. Όπως παρατηρούμε από τη δεύτερη στήλη του πίνακα, οι σχέσεις του target schema που συμμετέχουν στο select τμήμα της αντιστοιχίας, συμπεριλαμβανομένων και των διαφορετικών στιγμιότυπων ίδιων σχέσεων είναι οι εξής:

{Professor, Student, Laboratory, University, Univ$1 (στιγμιότυπο του University), Prof$1 (στιγμιότυπο του Professor)}

Σε περίπτωση που δεν προσθέσουμε κανέναν περιορισμό-σύνδεσμο στην αντιστοιχία μας, τότε η όψη της σχέσης Supervisor, δε θα είναι παρά το καρτεσιανό γινόμενο έξι διαφορετικών σχέσεων του target schema. Κάτι τέτοιο βέβαια, θα συνιστούσε ένα σίγουρο σημασιολογικό λάθος, μιας και θα συνδυάζαμε πλειάδες διαφορετικών πινάκων μεταξύ τους, χωρίς ο συνδυασμός αυτών να έχει κάποια σημασιολογική σημασία. Έτσι λοιπόν, θεωρούμε ότι όλες οι σχέσεις που συμμετέχουν σε μια αντιστοιχία πρέπει να συνδέονται μεταξύ τους με επιπλέον συνδέσμους. Αν φανταστούμε τους πίνακες που συμμετέχουν στην αντιστοιχία ως κόμβους ενός γράφου, και τους πιθανούς συνδέσμους μεταξύ δύο πινάκων ως ακμές του γράφου αυτού, το πρόβλημα ανάγεται στο να βρούμε έναν τρόπο να μετατρέψουμε το γράφο αυτό συνεκτικό, ώστε κάθε κόμβος να συνδέεται με τουλάχιστον κάποιον άλλο κόμβο στο γράφο.

Ορισμός 4.2

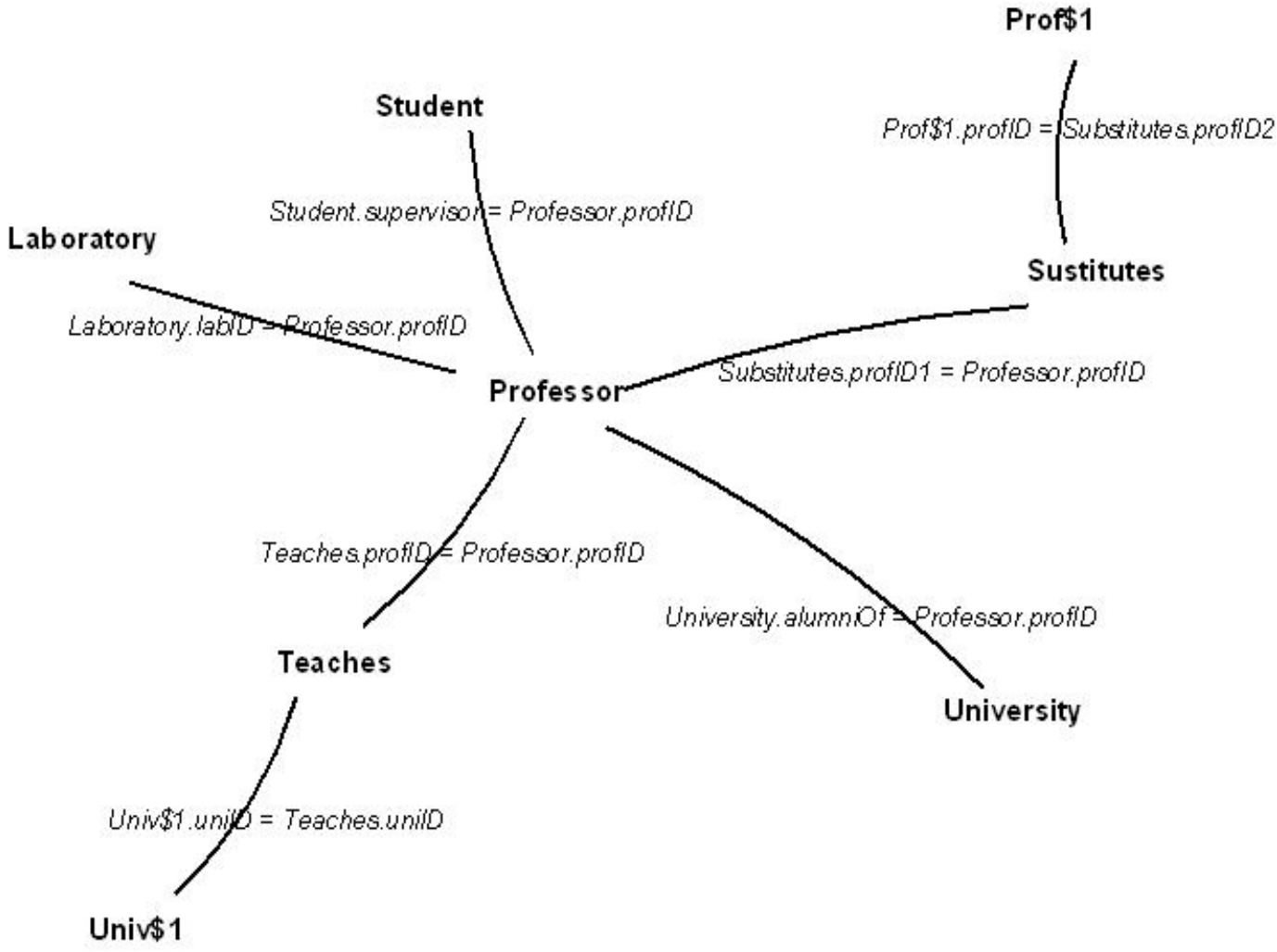
Ορίζουμε ως γράφο της αντιστοιχίας M, τον γράφο G(M) = {V,E}, όπου

$V = V1 \cup V2$ *και*

V1 = οι διαφορετικές σχέσεις του target schema που συμμετέχουν στο select τμήμα της M, με άλλα λόγια οι διαφορετικές σχέσεις που εμφανίζονται στις ισοδυναμίες που επιλέχθηκαν για τις ιδιότητες του πίνακα της M

V2 = οι διαφορετικές σχέσεις του target schema που συμμετέχουν στην αντιστοιχία M, αλλά δεν εμφανίζονται στο select τμήμα αυτής

E = οι σύνδεσμοι που εμπεριέχονται στην αντιστοιχία μεταξύ δύο πινάκων αυτής.



Σχήμα 4 f: Ο γράφος της αντιστοιχίας Supervisor του πίνακα 4 a

Στο σχήμα 4 f βλέπουμε τον γράφο G(Supervisor) της αντιστοιχίας του πίνακα 4 a.

Οι κόμβοι του συνόλου V1 είναι οι Univ$1, Prof$1, Professor, University, Student, Laboratory και οι κόμβοι του συνόλου V2 είναι οι κόμβοι Substitutes, Teaches.

Η μετατροπή του γράφου μιας αντιστοιχίας σε συνεκτικό γράφο ισοδυναμεί ουσιαστικά με την εύρεση του κατάλληλου μονοπατιού συνένωσης (join path) για κάθε μια από τις ισοδυναμίες που έχουν επιλεχθεί στην αντιστοιχία. Τα μονοπάτια αυτά, θα προέρχονται από το γενικό μοντέλο του target schema, και είχαν δημιουργηθεί ανά δύο πίνακες κατά την κατασκευή του γενικού μοντέλου (Παράγραφος 4.2). Υπενθυμίζουμε όμως ότι μεταξύ δύο πινάκων, εν γένει υπάρχουν παραπάνω του ενός μονοπάτια συνένωσης διαφορετικού ή και

ίδιου μήκους. Το join path που θα επιλεχθεί για μια συγκεκριμένη ισοδυναμία κατά την κατασκευή της αντιστοιχίας, πρέπει να είναι το πλέον κατάλληλο μεταξύ όλων των υπολοίπων διαθέσιμων μονοπατιών συνένωσης. Ιδιαίτερο ενδιαφέρον παρουσιάζει λοιπόν, να ορίσουμε την καταλληλότητα ενός join path για τη χρησιμοποίηση του από μια ισοδυναμία μιας αντιστοιχίας. Ας δούμε το παράδειγμα του γράφου στο σχήμα 4 f. Όπως παρατηρεί κανείς, ανάμεσα στους πίνακες University και Professor υπάρχουν δυο πιθανά μονοπάτια συνένωσης. Το πρώτο είναι το University.uniID = Professor.alumniOf και το δεύτερο το University.uniID = Teaches.uniID and Teaches.profID = Professor.profID.

Διαισθητικά καταλαβαίνει κανείς ότι το πρώτο μονοπάτι συνένωσης κρίνεται κατάλληλο για την ισοδυναμία της ιδιότητας Supervisor.univStudied ενώ το δεύτερο για την ιδιότητα Supervisor.univTeaches. Και στις δύο περιπτώσεις, ο μηχανισμός βρήκε το σωστό μονοπάτι συνένωσης όπως παρατηρούμε στον πίνακα 4 a. Αντίθετα για την ισοδυναμία της ιδιότητας Supervisor.dept χρησιμοποιήθηκε λανθασμένα το πρώτο από τα δύο μονοπάτια που αναφέρθηκαν.

Η ακριβής εύρεση εκείνου του μονοπατιού συνένωσης που εκφράζει πλήρως τη σημασιολογία της ισοδυναμίας μιας ιδιότητας είναι ένα δύσκολο πρόβλημα, μιας και το σύστημα, πρέπει να καταλαβαίνει πλήρως τις διάφορες σημασιολογικές έννοιες των join paths. Παρόλα αυτά, μπορούμε να θέσουμε μερικά ευριστικά κριτήρια κατά την επιλογή των join paths ώστε να μειώνουμε το χώρο αναζήτησης. Ας μην ξεχνάμε πως αν δύο πίνακες T1, T2 συνδέονται στον γράφο του σχήματος τους με μια αλυσίδα, N πινάκων και κάθε πίνακας έχει X πιθανούς συνδέσμους με τον επόμενο πίνακα της αλυσίδας, υπάρχουν X^N πιθανά join paths μήκους N ανάμεσα στους δύο πίνακες T1,T2, οπότε ο χώρος αναζήτησης αυξάνει εκθετικά με το μήκος του join path που ψάχνουμε.

Έτσι λοιπόν θέτουμε δύο προδιαγραφές που πρέπει να ικανοποιούνται ψάχνοντας για έναν τρόπο συνένωσης των κόμβων του γράφου G(M):

- **A1**:Τα μονοπάτια συνένωσης των πινάκων να έχουν το μικρότερο δυνατό μήκος
- **A2**:Η σύνδεση των κόμβων του συνόλου V1 να γίνεται με τέτοιο ώστε οι κόμβοι του συνόλου V2 να είναι οι ελάχιστοι δυνατοί.

Η πρώτη προδιαγραφή οδηγεί σε συνενώσεις μικρού μήκους μεταξύ των κόμβων του συνόλου V1, πράγμα το οποίο μπορεί να μεταφραστεί ως ισχυρές σημασιολογικές συσχετίσεις μεταξύ των σχέσεων που εμφανίζονται στο select τμήμα της αντιστοιχίας.

Η δεύτερη προδιαγραφή οδηγεί σε αντιστοιχίες που εμφανίζουν λίγους δευτερεύοντες πίνακες, δηλαδή πίνακες του συνόλου V2. Έτσι αποφεύγεται η εισαγωγή στην αντιστοιχία σχέσεων που δεν εμφανίζονται σε κάποια ισοδυναμία αυτής, και δυνητικά θα παρουσίαζαν σημασιολογικές έννοιες, ασυσχέτιστες με την σημασιολογία της αντιστοιχίας.

Εάν προσπαθήσουμε να επιλέξουμε ξεχωριστά για κάθε ισοδυναμία που κρίνουμε ότι χρειάζεται ένα μονοπάτι συνένωσης για να έχει νόημα, το μονοπάτι με το μικρότερο μήκος, δίνοντας έτσι προτεραιότητα στην πρώτη προδιαγραφή που θέσαμε, είναι πολύ πιθανό να εισάγουμε στην αντιστοιχία μας νέους κόμβους που θα ανήκουν στο σύνολο V2. Κατά συνέπεια ερχόμαστε σε αντίθεση με τη δεύτερη προδιαγραφή που θέσαμε.

Από την άλλη μεριά αν συνδέσουμε τους κόμβους του V1 μεταξύ τους με τέτοιο τρόπο ώστε οι να απαιτούνται οι ελάχιστοι δυνατοί δευτερεύοντες πίνακες, μπορεί να καλύψουμε την δεύτερη προδιαγραφή, αθετώντας όμως την πρώτη.

Ας δούμε ένα παράδειγμα. Έστω ότι για έναν γράφο μιας αντιστοιχίας M, το σύνολο V1 = {T1,T2,T3,T4}. Ας θεωρήσουμε ότι το σύστημα κρίνει πως οι τρεις ισοδυναμίες που αναφέρονται στους πίνακες T2, T3, T4 χρειάζονται ένα μονοπάτι συνένωσης με τον πίνακα T1. Σύμφωνα με την πρώτη προδιαγραφή πρέπει να επιλέξουμε το μικρότερο σε μήκος μονοπάτι συνένωσης για κάθε έναν από τους πίνακες στο σύνολο S = V1 – {T1} προς τον πίνακα T1. Για το λόγο αυτό επιλέγουμε για κάθε πίνακα του συνόλου S, το join path εκείνο με τον πίνακα T1 με το μικρότερο μήκος. Έστω ότι για κανέναν από αυτούς τους πίνακες δεν υπάρχει μονοπάτι μοναδιαίου μήκους με τον T1, και συνεπώς τα μονοπάτια που επιλέχθηκαν είναι όλα μήκους δυο:

- Για τον T2: *T2 – E2 – T1*
- Για τον T3: *T3 – E3 – T1*
- Για τον T4: *T4 – E4 – T1*

Βλέπουμε λοιπόν ότι πετυχαίνουμε την σύνδεση των κόμβων του V1 με έξι ακμές, αλλά το σύνολο V2 = {E2,E3,E4} δεν είναι το ελάχιστο δυνατό όπως θα θέλαμε, βάσει της δεύτερης προδιαγραφής.

Αντίθετα εάν επιλέγαμε τα εξής μονοπάτια (τα οποία θεωρούμε ότι υπάρχουν στο γενικό μοντέλο του target σχήματος):

- Για τον T2: *T2 – E2 – T1*
- Για τον T3: *T3 – T2 –E2- T1*
- Για τον T4: *T4 –T3— T2—E2 – T1*

Θα είχαμε μια σύνδεση του V1 με τέσσερις ακμές και το σύνολο τώρα θα αποτελείτο μόνο από τον κόμβο E2. Παρόλα αυτά, τα μονοπάτια συνένωσης για τους πίνακες T3 και T4 με τον πίνακα T1, δεν είναι τα συντομότερα και συνεπώς ερχόμαστε σε αντίθεση με την πρώτη προδιαγραφή.

Συμπεραίνουμε λοιπόν ότι οι δύο προδιαγραφές πολλές φορές δε γίνεται να καλυφθούν ταυτόχρονα, και για το λόγο αυτό οφείλουμε να επιλέγουμε κάθε φορά σε ποια από τις δυο προδιαγραφές θα δώσουμε προτεραιότητα.

Ένα σημαντικό θέμα που προκύπτει από το παραπάνω παράδειγμα είναι ο τρόπος που το σύστημα κρίνει ποιες ισοδυναμίες χρειάζονται κάποιο μονοπάτι συνένωσης και μεταξύ ποιων πινάκων πρέπει να αναζητηθεί αυτό το μονοπάτι. Στο σημείο αυτό υπεισέρχεται ο ρόλος της κύρια σχέσης μιας αντιστοιχίας M, η οποία εκπροσωπεί την σχέση εκείνη του target schema, που εμφανίζει τη μεγαλύτερη σημασιολογική ομοιότητα με την έννοια της σχέσης Rs της αντιστοιχίας M. πιο συγκεκριμένα μπορούμε να ορίσουμε:

Ορισμός 4.3

Κύρια σχέση (Main relation,R_{MAIN}) της αντιστοιχίας M μιας σχέσης R_S είναι εκείνη η σχέση του σχήματος S_T (target schema), η οποία ανήκει στο σύνολο V1 του γράφου G(M) και συγκριτικά με όλες τις άλλες σχέσεις που ανήκουν στο V1, ταιριάζει περισσότερο σημασιολογικά με την R_S.

Παραδείγματος χάριν η κύρια σχέση ενός πίνακα Doctor σε μια βάση δεδομένων ενός νοσοκομείου μπορεί να είναι η σχέση Physician της βάσης ενός άλλου νοσοκομείου, η κύρια σχέση του πίνακα Clerk ενδέχεται να είναι ένας πίνακας Employee κ.ο.κ. Στο παράδειγμα του πίνακα 4 a, ο κύριος σχέση του πίνακα Supervisor είναι ο πίνακας Professor.

Άπαξ και βρεθεί η κύρια σχέση μιας αντιστοιχίας, οι ισοδυναμίες που κρίνεται ότι χρειάζονται ένα μονοπάτι συνένωσης είναι οι ισοδυναμίες $C_{D\_A}(R_S.A,T.X)$ όπου $T \neq R_{MAIN}$. Για κάθε τέτοια ισοδυναμία αναζητείται ένα μονοπάτι συνένωσης μεταξύ των πινάκων T, R_{MAIN}. Το πως επιλέγεται η κύρια σχέση μιας αντιστοιχίας από το σύνολο V1 του γράφου της βασίζεται στα επόμενα δύο ευριστικά κριτήρια:

- **H1**: Εκείνη η σχέση R του συνόλου V1 που περιέχεται στις περισσότερες ισοδυναμίες που χρησιμοποιούνται από την αντιστοιχία
- **H2**: Εκείνη η σχέση R του συνόλου V1, που περιέχεται στις ισοδυναμίες των ιδιοτήτων που αποτελούν το κύριο κλειδί της σχέσης R_S.

Στο παράδειγμα του πίνακα 4 a , και τα δύο ευριστικά κριτήρια βρίσκουν τη σωστή κύρια σχέση του πίνακα Supervisor, R_{MAIN} = Professor. Πράγματι , σύμφωνα με το πρώτο κριτήριο τρεις από τις εννιά ισοδυναμίες περιλαμβάνουν ιδιότητες της σχέσης Professor, που είναι και ο μέγιστος αριθμός αντιστοιχιζόμενων ιδιοτήτων του target Schema στην ίδια σχέση.

Σύμφωνα με το δεύτερο κριτήριο, το κύριο κλειδί της σχέσης Supervisor, Supervisor.profID αντιστοιχίζεται στην ιδιότητα Professor.ID, οπότε και εδώ η κύρια σχέση ορίζεται ο πίνακας Professor.

Ένας επιπλέον περιορισμός που πρέπει να θέσουμε κατά την εύρεση των μονοπατιών συνένωσης σχετίζεται με τις διαφορετικές μετονομασίες ίδιων πινάκων. Ας θυμηθούμε τον τρόπο που προσθέτουμε ψευδώνυμα στις ισοδυναμίες των ιδιοτήτων του πίνακα RS. Δύο ιδιότητες του πίνακα R_S, A και B, ενδέχεται να αντιστοιχίζονται στην ίδια ιδιότητα του target schema T.X, δηλαδή επιτρέπουμε στην αντιστοιχία μας την ταυτόχρονη ύπαρξη δύο(ή και περισσότερων) ισοδυναμιών $C_{D\_A}(R_S.A,T.X)$ και $C_{D\_B}(R_S.B,T.X)$ όπου B≠A μόνο υπό την προϋπόθεση ότι ο πίνακας T στις δύο ισοδυναμίες θα εμφανίζεται με διαφορετικά ψευδώνυμα. Για τον ίδιο ακριβώς λόγο που δεν επιτρέπουμε την αντιστοίχιση δύο διαφορετικών ιδιοτήτων A,B στο ίδιο στιγμιότυπο του πίνακα T (στην ίδια μετονομασία αυτού), δε θα πρέπει να επιτρέπουμε και την εμφάνιση του ίδιου μονοπατιού συνένωσης σε ισοδυναμίες που αντιστοιχίζουν ιδιότητες του R_S στον ίδια ιδιότητα T.X. Έτσι λοιπόν, ο περιορισμός που θέτουμε είναι πως δε θα πρέπει να υπάρχουν στην αντιστοιχία δύο ιδιότητες R_S.A και R_S.B με A≠B, ισοδυναμίες $C_D(R_S.A,T.X)$ και $C_D(R_S.B,T.X)$ και ψευδώνυμα $Alias_A \neq Alias_B$ αντίστοιχα, για τις οποίες θα ισχύει $JP_A = JP_B$. Ο περιορισμός αυτός μπορεί να τηρηθεί απλά, κρατώντας για κάθε μονοπάτι συνένωσης το σύνολο των μετονομασμένων ιδιοτήτων που το χρησιμοποιούν.

Ορισμός 4.4

Ορίζουμε ως σύνολο ιδιοτήτων ενός μονοπατιού συνένωσης JP, το σύνολο Attrs = {$Alias_i.X_j$} όπου ένα στοιχείο $Alias_i.X_j$ είναι μέλος αυτού, αν και μόνο αν υπάρχει ισοδυναμία $C_D(R_S.A,T.X_j)$ στην αντιστοιχία της σχέσης R_S με ψευδώνυμο $Alias_i$ που χρησιμοποιεί το join path JP.

Βάσει των παραπάνω ένα μονοπάτι συνένωσης JP μπορεί να χρησιμοποιηθεί από μια ισοδυναμία $C_{D\_A}(R_S.A,T.K)$ με ψευδώνυμο $Alias_A$ αν και μόνο αν

- Δεν υπάρχει στο σύνολο ιδιοτήτων του JP ιδιότητα Alias.K
- Το JP δεν ανήκει στο σύνολο JP_{BAD_A}

Έχοντας τώρα βρει την κύρια σχέση μιας αντιστοιχίας, και γνωρίζοντας πότε ένα μονοπάτι είναι έγκυρο για μια ισοδυναμία, πρέπει να επιλέξουμε ένα μονοπάτι προς τον πίνακα R_{MAIN}, για όλες τις ισοδυναμίες που δεν περιλαμβάνουν την κύρια σχέση. Η αναζήτηση του κατάλληλου μονοπατιού για κάθε ισοδυναμία, θα κινηθεί ανάμεσα στα δύο κριτήρια-προδιαγραφές (A1-A1) που θέσαμε νωρίτερα. Ανάλογα σε ποιο κριτήριο δίνουμε μεγαλύτερη προτεραιότητα, μπορούμε να διαμορφώσουμε δύο ειδών αλγορίθμους εύρεσης συνενώσεων.

Έστω λοιπόν ότι ψάχνουμε για ένα μονοπάτι συνένωσης μιας ισοδυναμίας $C_D(R_S.A,T.X)$ όπου $T \neq R_{MAIN}$, μεταξύ των πινάκων T,R_{MAIN}. Σε γενικές γραμμές η διαδικασία που ακολουθούμε είναι η εξής:

Αρχικά θέτουμε προτεραιότητα σε ήδη ανακαλυφθέντα μονοπάτια συνένωσης. Αν μια συγκεκριμένη ακολουθία συνδέσμων, που συμμετέχουν ήδη στην αντιστοιχία, συνιστούν ένα μονοπάτι μεταξύ των πινάκων T,R_{MAIN} και η ισοδυναμία C_D μπορεί να χρησιμοποιήσει αυτό το μονοπάτι συνένωσης, τότε επιλέγεται αυτό το μονοπάτι συνένωσης ανεξαρτήτου μήκους. Σε αυτή την περίπτωση, δεν προσθέτουμε κανένα νέο σύνδεσμο ούτε δευτερεύοντες πίνακες στην ισοδυναμία, μιας και το μονοπάτι που βρέθηκε χρησιμοποιείται ήδη στην αντιστοιχία. Η επιλογή αυτή είναι σύμφωνη και με τα δύο κριτήρια A1,A2, και για το λόγο αυτό τις δίνουμε πάντα την υψηλότερη προτεραιότητα.

Σε περίπτωση που η αντιστοιχία δεν περιέχει κάποιο μονοπάτι μεταξύ των πινάκων T, R_{MAIN}, μπορούμε όπως είπαμε, εν συνεχεία να δώσουμε προτεραιότητα σε οποιοδήποτε από τα δύο κριτήρια A1,ή A2. Δίνοντας προτεραιότητα στο A1 συνεχίζουμε ψάχνοντας στο γενικό μοντέλο του σχήματος target schema για το μονοπάτι εκείνο μεταξύ των δύο πινάκων που μπορεί να χρησιμοποιηθεί από την ισοδυναμία και έχει το μικρότερο δυνατό μήκος. Η αναζήτηση γίνεται μέχρι ένα συγκεκριμένο μήκος, πέραν του οποίου θεωρούμε εξαιρετικά απίθανο να υπάρχει κάποιο μονοπάτι με σωστή σημασιολογία. Μια ρεαλιστική τιμή του μέγιστου αυτού μήκους είναι πέντε με έξι.

Δίνοντας προτεραιότητα στο κριτήριο A2, εξετάζουμε τα μονοπάτια που έχουν ήδη βρεθεί στην αντιστοιχία και ψάχνουμε για το μονοπάτι με το μικρότερο δυνατό μήκος, το οποίο θα αποτελεί υπερ-μονοπάτι ενός εκ των ήδη ανακαλυφθέντων μονοπατιών.

Αν και πάλι δε βρεθεί κάποιο κατάλληλο μονοπάτι ξαναψάχνουμε από την αρχή, σα να εφαρμόζαμε το κριτήριο A1.

Ορισμός 4.4

Ορίζουμε ως υπερ-μονοπάτι ενός join path JP, και συμβολίζουμε ως $JP_{SUPER}(JP,n)$, ένα μονοπάτι μήκους n του οποίου η ακολουθία των συνδέσμων, περιέχει (γνήσια) την ακολουθία των συνδέσμων του join path JP. Κατά επέκταση, αν k το μήκος του μονοπατιού JP, θα ισχύει πάντα n>k.

Έστω για παράδειγμα το προηγούμενο σενάριο με τους πίνακες του συνόλου V1 = {T1,T2,T3,T4}, όπου T1 είναι ο κύριος πίνακας της αντιστοιχίας. Αν έχει ήδη ανακαλυφθεί το join path για τον πίνακα T2, T2 – E2 – T1 και σε επόμενη φάση πρέπει να επιλεχθεί ένα join path για τον πίνακα T3, δίνοντας προτεραιότητα στο κριτήριο A1 θα βρεθεί το μονοπάτι T3 – E3 – T1, ενώ δίνοντας προτεραιότητα στο κριτήριο A2 θα βρεθεί το μονοπάτι T3 –T2 – E2 – T1.

Η διαδικασία που περιγράφηκε παραπάνω για την μετατροπή του γράφου της αντιστοιχίας σε συνεκτικό, αποτελεί και το τελευταίο στάδιο για την αρχικοποίηση της. Όλα τα ευριστικά κριτήρια που χρησιμοποιήθηκαν, τόσο κατά την επιλογή των ισοδυναμιών όσο και κατά την επιλογή των μονοπατιών συνένωσης προσπαθούν να προσεγγίσουν την έννοια του πίνακα της αντιστοιχίας στο target schema. Παρόλα αυτά, όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, πιθανά λάθη μπορούν να οδηγήσουν στην κατασκευή ενός ερωτήματος, το οποίο να μην ανταποκρίνεται ακριβώς στη σημασιολογία της σχέσης που μας ενδιαφέρει. Για το λόγο αυτό, οι αντιστοιχίες ως δυναμικές δομές, βελτιώνονται με την ανάδραση που προσφέρει ο χρήστης. Για τον τρόπο που επιτυγχάνεται η βελτίωση αυτή θα μιλήσουμε στο επόμενο κεφάλαιο. Στα επόμενα σχήματα δίνονται οι δύο αλγόριθμοι επιλογής των μονοπατιών συνένωσης σε μορφή ψευδοκώδικα.

Επιλογή ενός μονοπατιού συνένωσης για μια ιδιότητα A, δίνοντας προτεραιότητα στο κριτήριο A1

Input: Η αντιστοιχία M του πίνακα Rs που ανήκει η ιδιότητα A, η κύρια σχέση της M, R_{MAIN} το σύνολο JP_{BAD} της ιδιότητας A, το γενικό μοντέλο του target schema, $Model_{TARGET}$.

Output: Το μονοπάτι JP, που επιλέγεται για την ιδιότητα A.

JP = NULL

Βρες την ισοδυναμία της ιδιότητας A, $C_D(R_S.A,T.X)$

Αν ($T = R_{MAIN}$) επέστρεψε NULL

Known_JPS = όλα τα ήδη ανακαλυφθέντα μονοπάτια στην αντιστοιχία προς τον πίνακα R_{MAIN}

Για κάθε μονοπάτι JP_temp του συνόλου Known_JPS, μεταξύ των πινάκων T και R_{MAIN}

Εάν το JP_temp δεν ανήκει στο σύνολο JP_{BAD} και δεν χρησιμοποιείται από κάποια άλλη ιδιότητα B στον πίνακα R_S

Σημείωσε ότι πλέον το JP_temp χρησιμοποιείται από την R_S.A

JP = JP_temp

Επέστρεψε το JP

C = όλα τα μονοπάτια ανάμεσα στους πίνακες T, R_{MAIN} του $Model_{TARGET}$ με μήκος length ≤ MAX_DISTANCE

dist = 0

Όσο (dist ≤ MAX_DISTANCE)

dist = dist +1

Για κάθε μονοπάτι JP_temp απόστασης dist του συνόλου C

Ένα το JP_temp δεν ανήκει στο σύνολο JP_{BAD} και δεν χρησιμοποιείται από κάποια άλλη ιδιότητα B στον πίνακα R_S

Σημείωσε ότι πλέον το JP_temp χρησιμοποιείται από την R_S.A

JP = JP_temp

Επέστρεψε το JP

Επέστρεψε NULL

Αλγόριθμος 4 b: Επιλογή JoinPath για μια ιδιότητα του πίνακα Rs, βάσει του κριτηρίου A1(μονοπάτια με το ελάχιστο μήκος)

Επιλογή ενός μονοπατιού συνένωσης για μια ιδιότητα A, δίνοντας προτεραιότητα στο κριτήριο A2

Input: Η αντιστοιχία M του πίνακα Rs που ανήκει η ιδιότητα A, η κύρια σχέση της M, R_{MAIN} το σύνολο JP_{BAD} της ιδιότητας A, το γενικό μοντέλο του target schema, $Model_{TARGET}$.

Output: Το μονοπάτι JP, που επιλέγεται για την ιδιότητα A.

JP = NULL

Βρες την ισοδυναμία της ιδιότητας A, $C_D(R_S.A,T.X)$

Αν ($T = R_{MAIN}$) επέστρεψε NULL

Known_JPS = όλα τα ήδη ανακαλυφθέντα μονοπάτια στην αντιστοιχία προς τον πίνακα R_{MAIN}

Για κάθε μονοπάτι JP_temp του συνόλου Known_JPS, μεταξύ των πινάκων T και R_{MAIN}

Εάν το JP_temp δεν ανήκει στο σύνολο JP_{BAD} και δεν χρησιμοποιείται από κάποια άλλη ιδιότητα B στον πίνακα R_S

Σημείωσε ότι πλέον το JP_temp χρησιμοποιείται από την R_S.A

JP = JP_temp

Επέστρεψε το JP

C1 = όλα τα μονοπάτια του συνόλου Known_JPS με μήκος length ≤ (MAX_DISTANCE – 1)

C2 = όλα τα μονοπάτια ανάμεσα στους πίνακες T, R_{MAIN} του $Model_{TARGET}$ με μήκος length ≤ MAX_DISTANCE

dist = 1

Εάν (το σύνολο C1 είναι άδειο)

επέλεξε ένα μονοπάτι συνένωσης εφαρμόζοντας τον αλγόριθμο του κριτηρίου A1

Όσο (dist ≤ MAX_DISTANCE)

dist = dist +1

Για κάθε μονοπάτι JP_temp απόστασης ≤ (dist – 1) του συνόλου C1

Για κάθε μονοπάτι JP_super απόστασης dist του συνόλου C2

Ένα JP_super είναι υπερμονοπάτι του JP_temp, δεν ανήκει στο σύνολο JP_{BAD} και δεν χρησιμοποιείται από κάποια άλλη ιδιότητα B στον πίνακα R_S

Σημείωσε ότι πλέον το JP_super χρησιμοποιείται από την R_S.A

JP = JP_super

Επέστρεψε το JP

Επέλεξε ένα μονοπάτι συνένωσης εφαρμόζοντας τον αλγόριθμο του κριτηρίου A1

Αλγόριθμος 4 c: Επιλογή JoinPath για μια ιδιότητα του πίνακα Rs, βάσει του κριτηρίου A2(ελάχιστοι δευτερεύοντες πίνακες)

4.5 Βελτίωση των αντιστοιχιών

Όπως είπαμε και προηγουμένως, η πρώτη εκδοχή μιας αντιστοιχίας,M που προέκυψε κατά την αρχικοποίηση της δομής Matrix(M), ενδέχεται να σφάλλει σε ορισμένα σημεία και για το λόγο αυτό να μην αντικατοπτρίζει επακριβώς τη σημασιολογία του πίνακα της αντιστοιχίας. Το πρόβλημα αυτό, εξαλείφεται με την ανάδραση που προσφέρει ο χρήστης, ο οποίος παρέχει στο μηχανισμό πληροφορίες σχετικά με ποια τμήματα του Matrix(M) (mapping Elements) μεταφράσθηκαν σωστά και ποια λανθασμένα. Έτσι λοιπόν, η ανάδραση δεν αφορά τη συνολική ποιότητα του αποτελέσματος, αλλά τμηματοποιείται ώστε ο μηχανισμός βελτίωσης να εστιάζει στα λανθασμένα σημεία και να παγιώνει τις αντιστοιχίσεις που κρίθηκαν σωστές. Κάτω από αυτό το πλαίσιο, ο χρήστης προσφέρει ανάδραση για κάθε γραμμή του πίνακα Matrix: Για κάθε ιδιότητα του πίνακα της αντιστοιχίας, ο χρήστης κρίνει τόσο την ισοδυναμία που χρησιμοποιήθηκε, όσο και το μονοπάτι συνένωσης που αυτή χρησιμοποιεί ώστε να ενσωματωθεί στην αντιστοιχία. Στο εξής θα αναφερόμαστε στην ανάδραση του χρήστη για ένα στοιχείο αντιστοιχίας MapEl ως Feedback(MapEl). Κατά συνέπεια η συνολική ανάδραση αποτελείται από ένα σύνολο στοιχείων Feedback(MapEl), πληθικότητας ίσης με τον αριθμό γραμμών του πίνακα Matrix.

Ο τρόπος με τον οποίο θα ζητείται από τον χρήστη του συστήματος, να κρίνει την αντιστοιχία, πρέπει σε κάθε περίπτωση να διέπεται από μια βασική αρχή: Το νόημα της ανάδρασης θα πρέπει να γίνεται εύκολα αντιληπτό στον χρήστη, ο οποίος δε γνωρίζει λεπτομέρειες για τον μηχανισμό παραγωγής αντιστοιχιών. Σε καμία περίπτωση, παραδείγματος χάριν δε θα μπορούσαμε να ζητήσουμε από τον εκάστοτε χρήστη μια υπόδειξη για ποιο από τα δύο κριτήρια εύρεσης join paths είναι περισσότερο κατάλληλο στη συγκεκριμένη αντιστοιχία, μιας και κάτι τέτοιο είναι τεχνική λεπτομέρεια και δε γίνεται εύκολα αντιληπτό. Για το λόγο αυτό, ο χρήστης καλείται να δώσει έναν απλό χαρακτηρισμό σε κάθε Mapping Element της αντιστοιχίας. Ο διάφοροι χαρακτηρισμοί με τους οποίους δύναται κάποιος να κρίνει την ποιότητα αντιστοίχισης ενός Mapping Element είναι προδιαγεγραμμένοι από το σύστημα, ώστε αυτό να γνωρίζει τις απαραίτητες ενέργειες που πρέπει να κάνει προκειμένου να βελτιώσει την αντιστοιχία. Ας δούμε αυτές τους χαρακτηρισμούς αναλυτικά:

- *Good Correspondence, No Join Path needed (Good/NoJoinPath)*

Ο χρήστης προσάπτει αυτόν τον χαρακτηρισμό στα Mapping Elements της αντιστοιχίας, των οποίων η ισοδυναμία που χρησιμοποιήθηκε είναι σωστή, και δεν απαιτούν κάποιο μονοπάτι συνένωσης για να ενσωματωθούν στην αντιστοιχία, με άλλα λόγια, η ισοδύναμη ιδιότητα του Target Schema ανήκει στον κύριο πίνακα της αντιστοιχίας

Παραδείγματα αυτής της κατηγορίας είναι οι δύο πρώτες γραμμές του πίνακα Matrix(Supervisor) στον πίνακα 4 a

(Supervisor.ID ←→Professor.profID, Supervisor.name←→Professor.name)

- *Good Correspondence, Good Join Path (Good/GoodJoinPath)*

Ο χαρακτηρισμός αυτός απευθύνεται στα Mapping Elements μιας αντιστοιχίας, των οποίων τόσο η ισοδυναμία όσο και το μονοπάτι συνένωσης που χρησιμοποιήθηκε κρίθηκαν σωστά.

Στον πίνακα 4 a παραδείγματα αυτής της κατηγορίας είναι τα Mapping Elements των ιδιοτήτων *Supervisor.univTeaches, Supervisor.univStudied* και *Supervisor.subsupervisor*

- *Good Correspondence, Bad Join Path*

Σε αυτή την κατηγορία, ο χρήστης θέτει εκείνα τα Mapping Elements , για τα οποία, ναι μεν η ισοδυναμία που χρησιμοποιήθηκε κρίθηκε σωστή, αλλά το μονοπάτι συνένωσης είναι λανθασμένο, με άλλα λόγια ο τρόπος που συνδέεται ο πίνακας της ισοδύναμης ιδιότητας με τον κύριο πίνακα της αντιστοιχίας, δεν αντικατοπτρίζει το σημασιολογικό νόημα που θα περίμενε ο χρήστης. Παράδειγμα αυτής της κατηγορίας αποτελεί το Mapping Element της ιδιότητας Supervisor.dept. Αν υποθέσουμε ότι η ιδιότητα αυτή αναφέρεται στο τμήμα του πανεπιστημίου που εργάζεται ένας Supervisor, το μονοπάτι συνένωσης *Professor.alumniOf = University.uniID* είναι λανθασμένο, μιας και με τον τρόπο αυτό η ιδιότητα αντιστοιχίζεται στο τμήμα του πανεπιστημίου που σπούδασε ο Supervisor

- *Good Correspondence, Self Join Path needed (Good/SelfJoinPath)*

Η κατηγορία αυτή απευθύνεται στα Mapping Elements της αντιστοιχίας, τα οποία αντιστοιχούν σε ιδιότητες του κύριου πίνακα, αλλά για να στέκουν σημασιολογικά χρειάζονται ένα self join path με αυτόν, κάτι το οποίο ο μηχανισμός δεν ανακάλυψε. Έστω για παράδειγμα ότι η ιδιότητα *Supervisor.subSupervisor* αρχικοποιείτο πριν από την ιδιότητα *Supervisor.ID*. Αν θυμηθούμε τον τρόπο που γίνεται η επιλογή των ισοδυναμιών και η προσθήκη ψευδωνύμων, θα δούμε ότι σε αυτήν την περίπτωση η ιδιότητα *subSupervisor* θα αντιστοιχηθεί στην *Professor.profID* χωρίς κάποιο ψευδώνυμο ενώ η ιδιότητα Supervisor.ID θα χρησιμοποιεί την ίδια ισοδυναμία με το ψευδώνυμο *Prof$1.profID*, υπονοώντας έτσι ότι η πρώτη ισοδυναμία δεν απαιτεί κάποιο μονοπάτι συνένωσης, ενώ η δεύτερη απαιτεί. Στην πραγματικότητα όμως, γνωρίζουμε ότι τα πράγματα έπρεπε να είναι αντίστροφα. Δηλαδή η ιδιότητα που αποθηκεύει το αναγνωριστικό του αναπληρωτή *Supervisor* να απαιτεί κάποιο

self join path με τη σχέση *Professor* και αυτή που αποθηκεύει το αναγνωριστικό του Supervisor να μην απαιτεί συνένωση με κάποιον πίνακα. Ο χρήστης αναγνωρίζει αυτό το λάθος και θέτει σε αυτή την περίπτωση το Mapping Element της ιδιότητας *Supervisor.subSupervisor* στην κατηγορία Good Correspondence, Self Join Path needed.

- *Good Correspondence, No Self Join Path needed(Good/NoSelfJoinPath)*

Η κατηγορία αυτή αποτελεί την αντίστροφή περίπτωση της προηγούμενης: Τίθεται δηλαδή για ένα Mapping Element όταν για αυτό έχει προστεθεί ένα Self Join Path, ενώ στην πραγματικότητα δε χρειάζεται. Έτσι στο προηγούμενο παράδειγμα ο χρήστης θα θέσει σε αυτή την κατηγορία την ιδιότητα *Supervisor.ID* που αντιστοιχίζεται στην *Prof$1.profID*.

- *Good Correspondence, JoinPath needed (Good/NeedJoinPath)*

Στην κατηγορία αυτή τίθενται τα Mapping Elements, για τα οποία έχει ανακαλυφθεί η σωστή ισοδυναμία, αλλά κρίθηκε ότι δε χρειάζονται μονοπάτι συνένωσης με κάποιον πίνακα. Αν θυμηθούμε από τον τρόπο επιλογής των join paths, ότι σε μια ιδιότητα δεν προσδίδεται μονοπάτι συνένωσης αν και μόνο αν ο πίνακας της αντίστοιχης ιδιότητας στο target Schema είναι ο κύριος πίνακας της αντιστοιχίας, συμπεραίνουμε ότι η κατηγορία αυτή τίθεται μόνο στην περίπτωση που δεν βρέθηκε ο σωστός κύριος πίνακας της αντιστοιχίας.

Ας δούμε ένα παράδειγμα. Έστω ότι στη σχέση Supervisor δεν υπάρχουν οι ιδιότητες *Supervisor.name* και *Supervisor.numOfstudents*. Σε αυτή την περίπτωση ψάχνοντας για την κύρια σχέση με το κριτήριο H1[1] ορίζεται λανθασμένα ο πίνακας University μιας και τώρα πλέον αυτός έχει τις περισσότερες εμφανίσεις στις ισοδυναμίες που επιλέχθηκαν. Κατά την επιλογή των join paths η ιδιότητα *Supervisor.univStudied* αντιστοιχεί στην ιδιότητα *University.uniID* και επειδή R_{MAIN} = *University* δε θα αναζητηθεί κάποιο μονοπάτι. Σε αυτή την περίπτωση ο χρήστης χαρακτηρίζει το Mapping Element της ιδιότητας *Supervisor.uniStudied* ως Good/NeedJoinPath.

- *Bad Correspondence*

Η τελευταία αυτή κατηγορία αφορά εκείνα τα Mapping Elements,των οποίων η ισοδυναμία κρίθηκε λανθασμένη, με άλλα λόγια το σημασιολογικό νόημα της ιδιότητας του πίνακα δεν ταιριάζει με το σημασιολογικό νόημα της ιδιότητας του target schema που εμπλέκεται στην ισοδυναμία. Κατά επέκταση, ο χαρακτηρισμός για το μονοπάτι συνένωσης, (αν χρησιμοποιήθηκε) είναι περιττός. Παράδειγμα τέτοιας περίπτωσης αποτελεί η $3^{η}$ και η $4^{η}$ ιδιότητα του Matrix(Supervisor):

[1] βλέπε κεφάλαιο 4.4.2 Επιλογή των μονοπατιών συνένωσης

Τόσο η ισοδυναμία *Supervisor.numOfStudents*←→*Professor.labWorks*[2], όσο και η *Supervisor.thesis_supervised*←→ *Student.supervisor* δεν εμφανίζουν κάποιο σημασιολογικό ταίριασμα και για το λόγο αυτό κρίνονται λανθασμένες.

Ας δούμε τώρα τι βήματα ακολουθεί ο μηχανισμός βελτίωσης λαμβάνοντας ένα στοιχείο FeedBack(MapEl) για κάθε μία από τις παραπάνω κατηγορίες.

- *Good Correspondence, No Join Path needed (Good/NoJoinPath)*

Λαμβάνοντας έναν τέτοιο θετικό χαρακτηρισμό, ο μηχανισμός διαπιστώνει ότι δεν χρειάζεται να αλλάξει τίποτα, στη συγκεκριμένη γραμμή του πίνακα Matrix. Η ισοδυναμία κρατείται και δεν αλλάζει.

- *Good Correspondence, Good Join Path (Good/GoodJoinPath)*

Όμοια με την προηγούμενη κατηγορία, ούτε σε αυτή την ανάδραση δεν αλλάζει τίποτα στη δομή Matrix.

- *Good Correspondence, Bad Join Path*

Σε μια τέτοια κατηγορία, τα βήματα που πρέπει να γίνουν είναι τα εξής:

1) Αποδέσμευσε το join path της συγκεκριμένης γραμμής του Matrix(Mapping Element), από την ιδιότητα που το χρησιμοποιούσε, ώστε να είναι διαθέσιμο σε άλλες ιδιότητες, αν αυτές το ζητήσουν (αφαίρεση το αντίστοιχο στοιχείο Alias.X από το σύνολο ιδιοτήτων του μονοπατιού συνένωσης)
2) Πρόσθεσε το join path αυτό στο σύνολο JP_{BAD} του MapEl
3) Εφάρμοσε τον αλγόριθμο για την εύρεση ενός νέου μονοπατιού(αλγόριθμος 4 b ή 4 c ανάλογα με το κριτήριο που θέτουμε σε προτεραιότητα)

Ας δούμε τα παραπάνω για την ιδιότητα *Supervisor.dept* που όπως είδαμε εμφανίζει ένα λανθασμένο μονοπάτι συνένωσης, παρόλο που η επιλεγμένη ισοδυναμία είναι σωστή.

Καταρχάς το λανθασμένο μονοπάτι *Professor.alumniOf = University.uniID* αποδεσμεύεται από την ιδιότητα που το χρησιμοποιούσε *University.dept*. Έτσι αν υπήρχε μια ιδιότητα *Supervisor.alumniOfDept* που δείχνει από ποιο τμήμα αποφοίτησε ένας καθηγητής, η οποία αρχικά είχε και αυτή ένα λανθασμένο μονοπάτι συνένωσης, τώρα θα μπορεί να χρησιμοποιήσει το σωστό μονοπάτι *Professor.alumniOf = University.uniID* .

---

[2] Η λανθασμένη ισοδυναμία αυτή τέθηκε εσκεμμένα ως παράδειγμα. Στην πραγματικότητα ο Matcher Coma++ δε δίνει μεγάλη βαθμολογία σε αυτήν την συσχέτιση, λόγω έλλειψης κάποιας σχέσης μεταξύ των ιδιοτήτων, βάσει των κριτηρίων που αναφέρθηκαν στο κεφάλαιο 2.4

Στη συνέχεια το μονοπάτι αυτό προστίθεται στο αντίστοιχο σύνολο JP_{BAD}, αποκλείοντας έτσι την περίπτωση να ξαναχρησιμοποιηθεί για δεύτερη φορά το ίδιο μονοπάτι, μιας και κρίθηκε λανθασμένο.

Τέλος αναζητείται ένα καινούριο μονοπάτι. Στη συγκεκριμένη αντιστοιχία υπάρχει ήδη ένα μονοπάτι μεταξύ των πινάκων *University* και *Professor* διαφορετικό από αυτό που μόλις προστέθηκε στο σύνολο JP_{BAD}, το join path *Professor.profID = Teaches.profID* and *Teaches.uniID = Univ$1.uniID*, και συνεπώς αυτό θα επιλεχθεί από οποιονδήποτε από τους δύο αλγόριθμους. Έτσι λοιπόν η μόνη αλλαγή που χρειάζεται να γίνει είναι η προσθήκη του ψευδωνύμου *Univ$1* στο Mapping Element της ιδιότητας *Supervisor.dept*, ώστε πλέον η ισοδύναμη ιδιότητα να είναι *Univ$1.dept* και να εμφανίζει ένα διαφορετικό μονοπάτι, το οποίο παρεμπιπτόντως είναι και το σωστό.

- *Good Correspondence, Self Join Path needed (Good/SelfJoinPath)*

Σε έναν τέτοιο χαρακτηρισμό, απλά κρατάμε την ίδια ισοδυναμία, δίνοντας όμως ένα νέο ψευδώνυμο στην αντιστοιχιζόμενη ιδιότητα, και ψάχνοντας για ένα self join path με τον αλγόριθμο εύρεσης μονοπατιών συνένωσης

- *Good Correspondence, No Self Join Path needed(Good/NoSelfJoinPath)*

Εδώ τα βήματα είναι τα αντίστροφα με αυτά της παραπάνω κατηγορίας:

Αποδεσμεύουμε το μονοπάτι συνένωσης από την ιδιότητα που το χρησιμοποιεί και σβήνουμε το ψευδώνυμο που χρησιμοποιείται. Πλέον η ιδιότητα θα αντιστοιχίζεται βάσει της ισοδυναμία της σε ιδιότητα του κύριου πίνακα και όχι σε κάποια μετονομασία αυτού.

- *Good Correspondence, JoinPath needed (Good/NeedJoinPath)*

Η ύπαρξη όπως είπαμε ενός τέτοιου στοιχείου ανάδρασης, αντιστοιχεί στην λανθασμένη εύρεση του κύρια πίνακα της αντιστοιχίας. Συνεπώς, ψάχνουμε απλά για ένα μονοπάτι συνένωσης προς τον νέο κύριο πίνακα της αντιστοιχίας

- *Bad Correspondence*

Με αυτόν τον χαρακτηρισμό, ο μηχανισμός διαπιστώνει ότι η ισοδυναμία που χρησιμοποιήθηκε για το συγκεκριμένο Mapping Element είναι λανθασμένη. Οπότε διαγράφει την ισοδυναμία αυτή, αποδεσμεύει το μονοπάτι συνένωσης που τυχόν να χρησιμοποιείτο από την αντίστοιχη ιδιότητα και επιλέγει εκείνη την ισοδυναμία $C_D(R_S.A,T.X)$από το σύνολο $C_{POSSIBLE}$ με τον μεγαλύτερο βαθμό βεβαιότητας. Αν $T \neq R_{MAIN}$ αναζητείται ένα μονοπάτι με τον αλγόριθμο εύρεσης join path.

Εκτελώντας τα παραπάνω βήματα για κάθε στοιχείο Feedback(MapEl) που ορίζει ο χρήστης, η αντιστοιχία που μας ενδιαφέρει σταδιακά μετεξελίσσεται ώστε στο τέλος το

αντίστοιχο SQL ερώτημα να αντικατοπτρίζει σε ικανοποιητικό βαθμό την σημασιολογική έννοια του πίνακα της αντιστοιχίας εκφρασμένη στο target schema.

Δεδομένου όμως, ότι ενδιαφερόμαστε όχι μόνο για το αποτέλεσμα, αλλά και για το χρόνο/βήματα ανάδρασης που απαιτούνται ώστε η αντιστοιχία να τελειοποιηθεί, οφείλουμε να παρατηρήσουμε ορισμένα σημεία βελτιστοποίησης κατά την επεξεργασία της ανάδρασης του χρήστη. Το πρώτο σημείο σχετίζεται με την εύρεση του κύριου πίνακα σε περίπτωση που η πρώτη απόπειρα κατά την αρχικοποίηση της αντιστοιχίας ήταν ανεπιτυχής. Το δεύτερο σημείο σχετίζεται με την ταχύτερη εύρεση σωστών ισοδυναμιών για ιδιότητες που κατατάχθηκαν στην κατηγορία Bad Correspondence και τέλος ένα τρίτο σημείο σχετικά με την αποδέσμευση και επιλογή νέων μονοπατιών συνένωσης.

Όπως είδαμε στο κεφάλαιο 4.4.2, χρησιμοποιούμε δύο ευριστικά κριτήρια για την επιλογή της κύριας σχέσης μιας αντιστοιχίας. Παρόλα αυτά, ενδέχεται και τα δύο κριτήρια να υποδείξουν μια σχέση ως κύρια, η οποία να είναι λανθασμένη. Ένα τέτοιο παράδειγμα είδαμε κατά την επεξήγηση της κατηγορίας Good Correspondence, Join Path needed. Η λανθασμένη εύρεση της κύριας σχέσης σε μια αντιστοιχία έχει σημαντικές επιπτώσεις στην ποιότητα της, μιας και βάσει αυτής, κρίνεται για ποιες ισοδυναμίες απαιτείται κάποιο μονοπάτι συνένωσης και για ποιες όχι. Σε περιπτώσεις όπου αρκετές ισοδυναμίες ιδιοτήτων είναι λανθασμένες, τα ευριστικά κριτήρια που χρησιμοποιήσαμε για τον καθορισμό του κύριου πίνακα είναι προφανώς αρκετά αναποτελεσματικά και έτσι ενδέχεται τα βήματα ανάδρασης που απαιτούνται για την βελτίωση της αντιστοιχίας να αυξάνονται σε μεγάλο βαθμό. Εκμεταλλευόμενοι όμως την ανάδραση του χρήστη μπορούμε να ανακαλύψουμε τον κύριο πίνακα χωρίς την χρήση ευριστικών κριτηρίων. Η πληροφορία αυτή έγκειται στα στοιχεία ανάδρασης όλων των κατηγοριών εκτός των Bad Correspondence και Good Correspondence, Join Path needed. Έστω για παράδειγμα ότι στην αντιστοιχία του πίνακα *Supervisor*, μόνο η ιδιότητα *Supervisor.name* αντιστοιχήθηκε σωστά στην ιδιότητα *Professor.name*, και ο χαρακτηρισμός της από το χρήστη είναι Good Correspondence/No Join Path needed, ενώ όλες οι άλλες οι ιδιότητες εμφανίζουν λανθασμένες ισοδυναμίες, ώστε ο χρήστης να τις κατατάξει στην κατηγορία Bad Correspondence. Σε αυτήν την περίπτωση η πρώτη θετική ανάδραση μας υποδεικνύει ότι το όνομα ενός *Supervisor* ισοδυναμεί με το όνομα ενός καθηγητή, χωρίς την απαίτηση για κάποια συνένωση. Με άλλα λόγια ο πίνακας *Professor* έχει την ίδια σημασιολογική έννοια με τον πίνακα *Supervisor* και έτσι μπορεί να θεωρηθεί η κύρια σχέση αυτού. Το ίδιο συμπέρασμα μπορεί να εξαχθεί και από άλλες κατηγορίες ανάδρασης. Πχ στην κατηγορία Good Correspondence/ Good Joinpath λαμβάνεται ως κύρια σχέση, η σχέση στην οποία καταλήγει το μονοπάτι συνένωσης που χρησιμοποιήθηκε. Βλέπουμε λοιπόν, ότι πλέον η χρήση των ευριστικών κριτηρίων για τον καθορισμό της

κύριας σχέσης θα γίνεται μόνο στην περίπτωση που αυτή δεν έχει καθοριστεί έμμεσα από την ανάδραση του χρήστη.

Η δεύτερη βελτιστοποίηση αφορά την ταχύτερη εύρεση των σωστών ισοδυναμιών για τα Mapping Elements της κατηγορίας Bad Correspondence. Όπως παρατηρήσαμε, στην κατηγορία αυτή, όπου κάποιο συγκεκριμένο στοιχείο της αντιστοιχίας κρίθηκε λανθασμένο για την ισοδυναμία που χρησιμοποιήθηκε, επιλέγεται η ισοδυναμία με την αμέσως μεγαλύτερη βαθμολογία που εξέδωσε το πρόγραμμα Coma++. Με άλλα λόγια δεν αλλάζουμε τις βαθμολογίες των ισοδυναμιών που βρέθηκαν και βασιζόμαστε εξ ολοκλήρου στα αποτελέσματα του Schema Matcher για την επιλογή της επόμενης ισοδυναμίας. Παρόλα αυτά υπάρχουν περιπτώσεις όπου μπορούμε να θεωρήσουμε ορισμένες ισοδυναμίες πιο πιθανές από κάποιες άλλες, ενώ το πρόγραμμα Coma++ τους έδωσε μικρότερη βαθμολογία.

Ας σκεφτούμε το παρακάτω παράδειγμα: έστω ότι στον πίνακα Supervisor υπήρχε μια απλή ιδιότητα *Supervisor.email* για την ηλεκτρονική διεύθυνση του Supervisor και στον πίνακα Professor, η αντίστοιχη ιδιότητα *Professor.contact*.

Ακόμη έστω ότι η ιδιότητα email υπήρχε και σε άλλους πίνακες του target σχήματος (*Student.email*, *Laboratory.email* κλπ). Σε μια τέτοια περίπτωση περιμένουμε το πρόγραμμα Coma++ δώσει υψηλότερη βαθμολογία στις ισοδυναμίες Supervisor.email ←→ *Student.email* , *Supervisor.email* ←→ *Laboratory.email* και μικρότερη βαθμολογία στην ισοδυναμία *Professor*.contact ←→ *Supervisor.email* που είναι και η σωστή. Αν όμως ο κύριος πίνακας της αντιστοιχίας Supervisor έχει καθοριστεί (πχ έπειτα από ανάδραση της ισοδυναμίας *Supervisor.name* ←→ *Professor.name*) τότε θα μπορούσαμε να υποθέσουμε ότι είναι πιο πιθανό (όχι όμως και σίγουρο) ότι η σωστή ισοδύναμη ιδιότητα της ιδιότητας *Supervisor.email* βρίσκεται στον πίνακα Professor. Έτσι λοιπόν μπορούμε να αυξήσουμε κατά ένα ποσοστό την βαθμολογία της ισοδυναμίας *Professor.contact* ←→ *Supervisor.email* και την επόμενη φορά να επιλεγεί αυτή παρόλο που αρχικά είχε μικρότερη τιμή. Απαραίτητη προϋπόθεση βέβαια είναι η ισοδύναμη ιδιότητα της οποίας αυξάνουμε την βαθμολογία να μην χρησιμοποιείται βέβαια σε ισοδυναμία άλλης ιδιότητας του ίδιου πίνακα. Π.χ. αν είχε καθοριστεί ότι το correspondence *Professor.profID* ←→ *Supervisor.ID* είναι σωστό, και η ισοδυναμία *Supervisor.email* ←→ *Professor.profID* άνηκε στο σύνολο των Possible Correspondences του στοιχείου, δε θα έπρεπε να αυξήσουμε την βαθμολογία της μιας και η ιδιότητα *Professor.profID* έχει ήδη αντιστοιχηθεί επιτυχώς. Με τον παραπάνω τρόπο μπορούμε να αποφύγουμε εν μέρει λάθη του προγράμματος Coma++ και να ευνοήσουμε την χρήση ισοδυναμιών που φαίνονται περισσότερο ρεαλιστικές.

Τέλος, μια τρίτη βελτιστοποίηση που οδηγεί σε ταχύτερες συγκλίσεις αντιστοιχιών αφορά την σειρά με την οποία γίνονται οι επεξεργασίες των ξεχωριστών στοιχείων ανάδρασης. Όπως είδαμε, η λανθασμένη εύρεση της κύριας σχέσης μιας αντιστοιχίας, οδηγεί

σίγουρα σε λανθασμένα μονοπάτια συνένωσης. Για το λόγο αυτό, πρέπει πρώτα να γίνεται η επεξεργασία των στοιχείων ανάδρασης, μέσω των οποίων ενδέχεται να ανακαλυφθεί ο κύριος πίνακας και ύστερα των υπολοίπων. Έστω για παράδειγμα, ότι αρχικά καθορίστηκε ως κύρια σχέση του πίνακα *Supervisor* η σχέση *University*, και η μόνες σωστές ισοδυναμίες είναι οι *Supervisor.name* ←→*Professor.name* και *Supervisor.labID* ←→ *Laboratory.labID*. Η πρώτη ισοδυναμία τίθεται από το χρήστη στην κατηγορία Good Correspondence/No Join Path needed ενώ η δεύτερη τίθεται στην κατηγορία Good Correspondence,Bad Join Path. Αν ο μηχανισμός επεξεργαστεί πρώτα το δεύτερο στοιχείο ανάδρασης, ο κύριος πίνακας δε θα έχει αλλάξει και συνεπώς το νέο μονοπάτι που θα αναζητηθεί, θα είναι μεταξύ των πινάκων *Laboratory* και *University*, και για το λόγο αυτό σίγουρα λανθασμένο και τη δεύτερη φορά. Αντίθετα, αν γίνει πρώτα η επεξεργασία του πρώτου στοιχείου ο κύριος πίνακας θα αλλάξει, και τώρα κατά την επιλογή μιας συνένωσης για την ισοδυναμία *Supervisor.labID* ←→ *Laboratory.labID* θα αναζητηθεί ένα μονοπάτι μεταξύ του σωστού ζεύγους πινάκων.

Έτσι λοιπόν ανανεώνουμε την αντιστοιχία, επεξεργαζόμενοι τα στοιχεία ανάδρασης με την παρακάτω σειρά προτεραιότητας:

1) Good Correspondence, No Join Path needed
2) Good Correspondence, Good Join Path
3) Good Correspondence, No Self Join Path needed
4) Good Correspondence, Self Join Path needed
5) Good Correspondence, Bad Join Path
6) Good Correspondence, Join Path needed
7) Bad Correspondence

Παρακάτω παρουσιάζουμε με μορφή ψευδοκώδικα την όλη διαδικασία βελτίωσης μιας αντιστοιχίας M.

Διαδικασία Βελτίωσης της αντιστοιχίας M ενός πίνακα RS

Input: η δομή Matrix(M), το σύνολο S = {Feedback(MapEl)} που έδωσε ο χρήστης

Output: η ανανεωμένη δομή Matrix(M)

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Good Correspondence, No Join Path needed

Βρες την ισοδυναμία $C_D(R_S.A,T.B)$ του MapEl

Εάν $R_{MAIN} \neq T$ $R_{MAIN} \rightarrow T$

Για κάθε MapEl΄ της δομής Matrix(M) που έχει χαρακτηρισθεί με την κατηγορία Bad Correspondence

Για κάθε ισοδυναμία $C_D(R_S.A,T.B)$ του συνόλου $C_{POSSIBLE\_A}$ του MapEl΄

Εάν $T = R_{MAIN}$ και δεν υπάρχει Mapping Element K στον Matrix(M) όπου $C_{D\_K}(R_S.K,T.B)$

$| C_D(R_S.A,T.B) | \rightarrow | C_D(R_S.A,T.B) | * X\%$, όπου X το ποσοστό βελτίωσης

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Good Correspondence, Good Join Path

Βρες τον τελευταίο πίνακα, T, του μονοπατιού JP που χρησιμοποιεί το MapEl

Εάν $R_{MAIN} \neq T$ $R_{MAIN} \rightarrow T$

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Good Correspondence, No Self Join Path needed

Βρες την ισοδυναμία $C_D(R_S.A,T.B)$, το ψευδώνυμο Alias και το joinpath JP του MapEl

Διέγραψε την ιδιόττηα Alias.B από το σύνολο ιδιοτήτων του JP

JP → NULL, Alias →NULL

Εάν $R_{MAIN} \neq T$ $R_{MAIN} \rightarrow T$

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Good Correspondence, Self Join Path needed

Βρες την ισοδυναμία $C_D(R_S.A,T.B)$ του MapEl

Βρες ένα ψευδώνυμο για το MapEl $Alias_A$

Εάν $R_{MAIN} \neq T$ $R_{MAIN} \rightarrow T$

Βρες ένα join path JP για το MapEl μεταξύ των πινάκων $Alias_A$ και R_{MAIN} (Self Join Path)

Πρόσθεσε στο σύνολο ιδιοτήτων του JP την ιδιότητα $Alias_A.B$

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Good Correspondence, Bad Join Path

Βρες τον τελευταίο πίνακα, K, του μονοπατιού JP που χρησιμοποιεί το MapEl, την ισοδυναμία $C_D(R_S.A,T.B)$ αυτού και το αντίστοιχο ψευδώνυμο Alias

Εάν $R_{MAIN} \neq K$ $R_{MAIN} \rightarrow K$

Πρόσθεσε το JP στο σύνολο JP_{BAD_A} του MapEl

Διέγραψε την ιδιόττηα Alias.B από το σύνολο ιδιοτήτων του JP

Βρες ένα join path JP_{NEW} για το MapEl μεταξύ των πινάκων T και R_{MAIN}

Πρόσθεσε στο σύνολο ιδιοτήτων του JP_{NEW} την ιδιότητα Alias.B

JP →JP_{NEW}

Αλγόριθμος 4 d : Διαδικασία βελτίωσης της αντιστοιχίας M από το feedback του χρήστη

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Good Correspondence, Join Path needed

Βρες την ισοδυναμία $C_D(R_S.A,T.B)$ του MapEl

Βρες ένα join path JP για το MapEl μεταξύ των πινάκων T και R_{MAIN}

Πρόσθεσε στο σύνολο ιδιοτήτων του JP την ιδιότητα T.B

Για κάθε στοιχείο Feedback(MapEl) του S με χαρακτηρισμό Bad Correspondence

Βρες την ισοδυναμία $C_D(R_S.A,T.B)$ του MapEl

Βάλε την ισοδυναμία αυτή στο σύνολο C_{BAD} του MapEL

$C_D(R_S.A,T.B) \rightarrow$ NULL

Εάν (JP του MapEl ≠ NULL)

Διέγραψε την ιδιόττηα Alias.B από το σύνολο ιδιοτήτων του JP

JP → NULL

Βρες το σύνολο $C_{D\_A\_POSSIBLE}$ του MapEl

Επέλεξε την ισοδυναμία $C_{D\_NEW}(R_S.A,T'.X)$ όπου $| C_{D\_NEW}(R_S.A,T'.X) | = \text{Max In } C_{D\_A\_POSSIBLE}$ και $C_{D\_NEW}(R_S.A,T'.X)$ δεν ανήκει στο σύνολο C_{BAD}

Εάν υπάρχει $C_D(R_S.C,T'.X)$ στη δομή Matrix(M)

Βρες ένα ψευδώνυμο για το MapEl $Alias_A$

Εάν $T' \neq R_{MAIN}$ ή ($T' = R_{MAIN}$ και AliasA ≠ NULL)

Βρες ένα join path JP_{NEW} για το MapEl μεταξύ των πινάκων T΄ και R_{MAIN}

Πρόσθεσε στο σύνολο ιδιοτήτων του JP_{NEW} την ιδιότητα Alias.B

$JP \rightarrow JP_{NEW}$

Επέστρεψε την ανανεωμένη δομή Matrix(M)

Αλγόριθμος 4 d : (Συνέχεια)

4.6 Μετρικές για την ποιότητα μιας αντιστοιχίας

Όπως είδαμε στην προηγούμενη παράγραφο, μια αντιστοιχία σε κάποια δεδομένη χρονική στιγμή ενδέχεται να εμφανίζει αποκλίσεις από αυτό που θα περίμενε ο χρήστης. Αποκλίσεις εμφανίζονται λόγω λανθασμένων ισοδυναμιών και λανθασμένων μονοπατιών συνένωσης. Ενδιαφέρον λοιπόν παρουσιάζει η ανάπτυξη μιας εκτιμήτριας συνάρτησης (Mapping Evaluator) η οποία θα δέχεται ως όρισμα την εκάστοτε αντιστοιχία και θα δίνει μια τιμή εκτίμησης μεταξύ 0 και 1, χαρακτηρίζοντας έτσι το κατά πόσο η αντιστοιχία προσεγγίζει την ιδανική σημασιολογικά αντιστοιχία, που θα μπορούσε να υπάρξει για αυτόν τον πίνακα. Η συνάρτηση αυτή, αποδεικνύεται πολύ χρήσιμη για το σύστημα GrouPeer, στο σημείο όπου δύο κόμβοι καλούνται να αποφασίσουν αν οι αντιστοιχίες που έχουν χτίσει μεταξύ τους μπορούν να θεωρηθούν αρκετά ισχυρές για να γειτονεύσουν.

Θα προσεγγίσουμε τη συνάρτηση αυτή χωρίζοντας τις αποκλίσεις που μπορούν να εμφανίζονται σε μια αντιστοιχία, M, σε δύο ομάδες:

A) Αποκλίσεις λόγω λανθασμένων ισοδυναμιών (Correspondence Failures)

B) Αποκλίσεις λόγω λανθασμένων μονοπατιών συνένωσης (JoinPath Failures)

Αν ορίσουμε ως συντελεστή ισοδυναμιών ($Factor_{CORR}$) μια τιμή μεταξύ 0 και 1, που θα αντιπροσωπεύει την ποιότητα της αντιστοιχίας ως προς τα λάθη ισοδυναμιών, και ομοίως ως συντελεστή συνενώσεων ($Factor_{JP}$) την τιμή που θα αντιπροσωπεύει την ποιότητα της αντιστοιχίας ως προς τα λάθη μονοπατιών συνένωσης, η τελική τιμή της συνάρτησης μπορεί εύκολα να προκύπτει συνδυάζοντας βεβαρημένα τους δύο παραπάνω συντελεστές. Δηλαδή

$\mathbf{Eval(M) = w_{JP} * Factor_{JP} + w_{CORR} * Factor_{CORR}}$**, όπου**

$0 \leq w_{JP} \leq 1,$

$0 \leq w_{CORR} \leq 1,$

$0 \leq Factor_{JP} \leq 1,$

$0 \leq Factor_{CORR} \leq 1,$

$w_{JP} + w_{CORR} = 1$

Με $\mathbf{w_{JP}}$ συμβολίζουμε το βάρος για τον συντελεστή μονοπατιών συνένωσης και με $\mathbf{w_{CORR}}$ το βάρος για τον συντελεστή ισοδυναμιών. Οι τιμές των δύο βαρών, μπορούν εν γένει να αλλάζουν για την αντιστοιχία κάθε πίνακα, ανάλογα με το ποια κατηγορία σφαλμάτων θεωρείται ποιο σημαντική. Αν παραδείγματος χάριν θεωρούμε πιο σημαντική την εύρεση των σωστών ισοδυναμιών από την εύρεση των σωστών μονοπατιών συνένωσης μπορούμε να θέσουμε $w_{JP} \leq w_{CORR}$. Δεδομένου όμως ότι αντιστοιχίες που απαιτούν κάποιο μονοπάτι συνένωσης δεν έχουν σημασιολογικό νόημα, αν η συνένωση που γίνεται είναι λανθασμένη,

μπορούμε να θεωρήσουμε ότι ο λόγος των δύο βαρών είναι ίσος με το ποσοστό των ισοδυναμιών που συμμετέχουν στην αντιστοιχία και δε χρειάζονται κάποια συνένωση για να στέκουν σημασιολογικά, με άλλα λόγια το ποσοστό των ισοδυναμιών της αντιστοιχίας που εμπλέκουν ιδιότητες της κύριας σχέσης της αντιστοιχίας. Συνεπώς θα έχουμε

$\frac{Wcorr}{Wjp} = \frac{Nmain}{N}$ όπου N ο αριθμός όλων των ιδιοτήτων του πίνακα της αντιστοιχίας και N_{MAIN} ο αριθμός των ιδιοτήτων του πίνακα που αντιστοιχίζονται σε ιδιότητες της κύριας σχέσης της αντιστοιχίας.

Έχοντας ορίσει τώρα τη γενική μορφή της εκτιμήτριας συνάρτησης Eval(M), απομένει η εύρεση μια γενικής έκφρασης για τους δύο συντελεστές αυτής ***$Factor_{JP}$ και $Factor_{CORR}$*** .

Ας δούμε πρώτα τον συντελεστή ***$Factor_{CORR}$.***

Είναι προφανές ότι η τελική τιμή του συντελεστή αυτού, προκύπτει από το ποσοστό των ιδιοτήτων του πίνακα που αντιστοιχίζονται σωστά, αλλά και από το πόσο πιθανό είναι να βρεθούν οι σωστές ισοδυναμίες για τις ιδιότητες του πίνακα, για τις οποίες επελέγη μια λανθασμένη ισοδυναμία C_D από το σύνολο $C_{POSSIBLE}$. Έτσι λοιπόν ορίζουμε ως f_{CORR_A}, ($0 \le f_{CORR\_A} \le 1$), τον βαθμό βεβαιότητας, ότι η ισοδυναμία που θα χρησιμοποιηθεί για την ιδιότητα A την επόμενη φορά στην αντιστοιχία θα είναι και η σωστή. Δεδομένων των τιμών f_{CORR_A} για όλες τις ιδιότητες του πίνακα της αντιστοιχίας, ο συντελεστής $Factor_{CORR}$ υπολογίζεται ως εξής:

$$Factor_{CORR} = \frac{\sum_{i=1}^{N} f_{corr\_Ai}}{N}$$

όπου N το πλήθος των ιδιοτήτων του πίνακα της αντιστοιχίας.

Μια εκτίμηση για τον βαθμό βεβαιότητας f_{CORR_Ai} της ιδιότητας Ai μπορεί να προκύψει συνδυάζοντας τις τιμές $|C_D(R_S.Ai,T.X)|$ που παρήγαγε το automatic schema matching tool (Coma++) για τις πιθανές ισοδυναμίες της ιδιότητας RS.Ai, με την ανάδραση που παρέχει ο χρήστης. Έτσι λοιπόν έχουμε:

- Για τις ιδιότητες εκείνες που ο χρήστης χαρακτήρισε κατά την ανάδραση με οποιαδήποτε κατηγορία διαφορετικής της Bad Correspondence θέτουμε $f_{CORR\_Ai} = 1$. Προφανώς σε αυτή την περίπτωση η ιδιότητα κρίθηκε ότι αντιστοιχήθηκε σωστά, οπότε την επόμενη φορά θα χρησιμοποιηθεί η ίδια ισοδυναμία με πιθανότητα 1.
- Για τις ιδιότητες εκείνες που ο χρήστης χαρακτήρισε με την κατηγορία Bad Correspondence ο βαθμός βεβαιότητας f_{CORR_Ai} προκύπτει ως η εκτίμηση της πιθανότητας κατά την επόμενη επιλογή ισοδυναμιών από το σύνολο $C_{POSSIBLE}$, να επιλεχθεί η σωστή ισοδυναμία. Έστω M η πληθικότητα του συνόλου $C_{POSSIBLE}$. Τότε

μια εκτίμηση για την πιθανότητα αυτή μπορεί να εκφραστεί ως ο λόγος της βαθμολογίας που επιλέχθηκε από το σύνολο $C_{POSSIBLE}$ (η ισοδυναμία με τη μεγαλύτερη βαθμολογία), προς το άθροισμα όλων των βαθμολογιών των M στοιχείων του συνόλου $C_{POSSIBLE.}$.

Συμπερασματικά, μπορούμε να πούμε το εξής:

$$\mathbf{f_{CORR\_Ai}} = \begin{cases} 1 \ if\ Category\ Ai \neq Bad\ Correspondence \\ \dfrac{|\, C_D(R_S.Ai, T.X)\,|}{\sum_{i-1}^{M} |\, C_{D\_POSS}(R_S.Ai, Ki.Yi)\,|} \quad otherwise \end{cases}$$

Με την παραπάνω εξίσωση, πρέπει να παρατηρήσουμε ότι κάνουμε μια σημαντική παραδοχή: Η σωστή αντιστοιχία βρίσκεται πάντα στο σύνολο $C_{POSSIBLE}$. κάτι τέτοιο όμως δεν ισχύει. Πολλές φορές ούτε ο Matcher κατορθώνει να ανακαλύψει τη σωστή ισοδυναμία και δίνει ως αποτέλεσμα ένα σύνολο λανθασμένων ισοδυναμιών. Ο παραπάνω τύπος επομένως ισχύει όταν είμαστε σίγουροι ότι μία από τις ισοδυναμίες στο σύνολο $C_{POSSIBLE}$, θα είναι και η σωστή ισοδυναμία. Για να τροποποιήσουμε τον παραπάνω τύπο ώστε, να ισχύει και για περιπτώσεις όπου ενδέχεται η σωστή ισοδυναμία να μην ανακαλύφθηκε από τον Matcher, πρέπει να βρούμε ένα μέτρο που να εκφράζει το βαθμό βεβαιότητας, κάτω από τον οποίο πιστεύουμε ότι η σημασιολογικά σωστή ισοδυναμία ανήκει στο σύνολο $C_{POSSIBLE}$. Το μέτρο αυτό μπορεί να προκύψει από τον μέσο όρο των βαθμολογιών του συνόλου $C_{POSSIBLE}$. Πράγματι, αν ο Matcher έχει προσδώσει στις πιθανές αντιστοιχίες υψηλές βαθμολογίες, είναι πιο πιθανό να ανακάλυψε τι σωστή, συγκριτικά με την περίπτωση που οι πιθανές ισοδυναμίες χαρακτηρίζονταν γενικά από χαμηλούς βαθμούς βεβαιότητας. Ορίζοντας ως **Expectance = Average($\mathbf{C_{POSSIBLE}}$)** τον μέσο όρο αυτό, η σχέση για την τιμέ f_{CORR_Ai} μπορεί να τροποποιηθεί ως εξής:

$$\mathbf{f_{CORR\_Ai}} = \begin{cases} 1 \ if\ Category\ Ai \neq Bad\ Correspondence \\ Expectance * \dfrac{|\, C_D(R_S.Ai, T.X)\,|}{\sum_{i-1}^{M} |\, C_{D\_POSS}(R_S.Ai, Ki.Yi)\,|} \\ otherwise \end{cases}$$

Όμοια με την εύρεση του συντελεστή ισοδυναμιών μπορούμε τώρα να βρούμε και την εύρεση του συντελεστή μονοπατιών συνένωσης. Η τελική τιμή του συντελεστή αυτού, εξαρτάται τώρα από τις ισοδυναμίες εκείνες της αντιστοιχίας που δεν ανήκουν στην κύρια σχέση, με άλλα λόγια προϋποθέτουν ένα μονοπάτι συνένωσης για να στέκουν σημασιολογικά. Αν ορίσουμε ως f_{JP_Ai} την πιθανότητα ότι το μονοπάτι συνένωσης που θα χρησιμοποιηθεί για την ιδιότητα Ai την επόμενη φορά στην αντιστοιχία θα είναι και σημασιολογικά σωστό, μπορούμε να πούμε ότι

$$\text{Factor}_{JP} = \frac{\sum_{i=1}^{N-Nmain} f_{JP\_Ai}}{N - Nmain}$$

Προφανώς αν ο χρήστης κατατάξει την ιδιότητα στην κατηγορία Good Correspondence, Good Join Path η πιθανότητα αυτή είναι ίση με 1.

Για τις ιδιότητες του πίνακα, των οποίων την αντιστοίχιση ο χρήστης χαρακτήρισε είτε ως Good Correspondence,Bad JoinPath είτε ως Good Correspondence, Self JoinPath needed η τιμή f_{JP_Ai} προκύπτει ως εξής:

Έστω ότι υπάρχουν M διαθέσιμα join paths ανάμεσα στον πίνακα R_{MAIN} και στον πίνακα της ιδιότητας που αντιστοιχίζεται η ιδιότητα Ai. Ακόμη έστω ότι X το πλήθος των join paths που ανήκουν ήδη στο σύνολο JP_{BAD_Ai} της ιδιότητας Ai. Τότε μπορούμε να πούμε ότι

$f_{JP\_Ai} = \frac{1}{M - X}$ αν M-X≠0 ή $f_{JP\_Ai} = 0$ αν M-X = 0.

Συμπερασματικά λοιπόν έχουμε:

$$\mathbf{f_{JP\_Ai}} = \begin{cases} 1 \textit{ if Category Ai = Good Correspondence, Good JoinPath} \\ \frac{1}{M - X} \text{ if M-X≠0 or 0 if M-X = 0} \end{cases}$$

if category Ai = Good Correspondence, Bad Join Path or

Good Correspondence, Self Join Path needed

Βάσει της παραπάνω ανάλυσης, η τελική έκφραση για την εκτιμήτρια συνάρτηση αντιστοιχιών είναι η εξής:

$$\textbf{Eval(M)} = \mathbf{w_{JP}}* \frac{\sum_{i=1}^{N-Nmain} f_{JP\_Ai}}{N - Nmain} + \mathbf{w_{CORR}}* \frac{\sum_{i=1}^{N} f_{corr\_Ai}}{N}$$ όπου

$0 \le w_{JP} \le 1,\ 0 \le w_{CORR} \le 1,\ w_{JP} + w_{CORR} = 1,$

$$\frac{w_{corr}}{w_{jp}} = \frac{Nmain}{N},$$

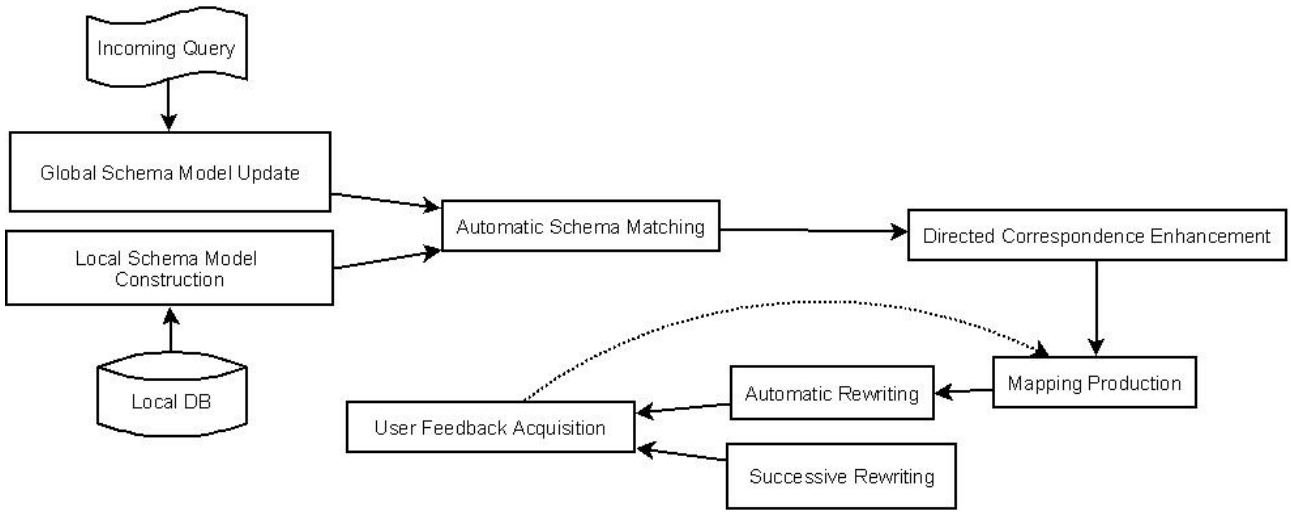
$$\mathbf{f_{JP\_Ai}} = \begin{cases} 1 \text{ } \textit{if Category Ai = Good Correspondence, Good JoinPath} \\ \frac{1}{M - X} \text{ if M-X}\neq 0 \text{ or 0 if M-X = 0} \\ \text{if category Ai = Good Correspondence, Bad Join Path or} \\ \text{Good Correspondence, Self Join Path needed} \end{cases}$$

Και

$$\mathbf{f_{CORR\_Ai}} = \begin{cases} 1 \text{ } \textit{if Category Ai} \neq \textit{Bad Correspondence} \\ \text{Expectance} * \frac{|C_D(Rs.Ai, T.X)|}{\sum_{i-1}^{M} |C_{D\_POSS}(Rs.Ai, Ki.Yi)|} \\ \textit{otherwise} \end{cases}$$

4.7 Ενσωμάτωση του μηχανισμού στο σύστημα GrouPeer

Ο μηχανισμός για την αυτόματη παραγωγή αντιστοιχιών που έχει παρουσιασθεί μέχρι στιγμής, προϋποθέτει ότι τα δύο σχήματα βάσεων (source και target schema) παραμένουν σταθερά και πλήρη κατά την αρχικοποίηση και βελτίωση των αντιστοιχιών. Παρόλα αυτά, όπως αναφέρθηκε και σε προηγούμενο κεφάλαιο, σε δίκτυα ομότιμων βάσεων δεδομένων, τα σχήμα του απομακρυσμένου κόμβου, είναι αρχικά άγνωστο στον τοπικό κόμβο που δημιουργεί τις αντιστοιχίες και χτίζεται σταδιακά από τις πληροφορίες που παρέχουν τα εισερχόμενα ερωτήματα. Έτσι λοιπόν, ο τοπικός κόμβος αρχικοποιεί ένα κενό σχήμα για τον απομακρυσμένο και νέοι πίνακες, ιδιότητες, περιορισμοί κύριων και εξωτερικών κλειδιών προστίθενται σε αυτό σταδιακά. Κάτω από αυτό το πλαίσιο, ο μηχανισμός που περιγράφηκε στα προηγούμενα κεφάλαια, οφείλει να λειτουργεί, όχι μόνο σε πλήρως γνωστά σχήματα εισόδου, αλλά και σταδιακά αποκαλυπτόμενα σχήματα (progressively revealing schemas). Στο κεφάλαιο αυτό, θα δούμε ότι μια τέτοια λειτουργία μπορεί πολύ εύκολα να επιτευχθεί, τροποποιώντας ορισμένα τμήματα της δομής του αλγορίθμου που παρουσιάσθηκε παραπάνω.



Σχήμα 4 g:Τροποποίηση του μηχανισμού για την ενσωμάτωση στο σύστημα GrouPeer

Στο σχήμα 4 g παρουσιάζεται η ακολουθιακή δομή του μηχανισμού σε περίπτωση που χρησιμοποιείται σε ένα περιβάλλον p2p βάσεων δεδομένων. Συγκρίνοντας το σχήμα αυτό με το σχήμα 4 b, παρατηρούμε ότι οι βασικές τροποποιήσεις που πρέπει να γίνουν στον μηχανισμό, προκειμένου να μπορεί να χρησιμοποιηθεί σε δίκτυο ομότιμων βάσεων, συνοψίζονται στα εξής σημεία:

Α) Ανανέωση του απομακρυσμένου σχήματος (Global Schema) από τα εισερχόμενα ερωτήματα

Β) Ενίσχυση των κατευθυνόμενων ισοδυναμιών (Directed Correspondences)

Γ) Βελτίωση των αντιστοιχιών μέσω των μεταφρασμένων ερωτημάτων.

Ας δούμε τα παραπάνω σημεία περισσότερο αναλυτικά.

Η πρώτη βασική διαφορά που εντοπίζεται κατά τη λειτουργία του αλγορίθμου σε p2p περιβάλλοντα, σχετίζεται με την έλλειψη γνώσης του τοπικού κόμβου για το απομακρυσμένο σχήμα. Συνεπώς, το γενικό μοντέλο που θα περιγράφει το απομακρυσμένο σχήμα δε μπορεί πλέον να κατασκευαστεί από τα μεταδεδομένα με απευθείας σύνδεση στη βάση, όπως στην περίπτωση γνωστών σχημάτων. Αντίθετα κατασκευάζεται σταδιακά, κάθε φορά που εισέρχονται ερωτήματα στον τοπικό κόμβο. Έτσι λοιπόν, ο τοπικός κόμβος δε διατηρεί ένα πλήρες γενικό μοντέλο για το global schema, αλλά κάθε φορά έχει στη διάθεσή του ένα ημιτελές στιγμιότυπο αυτού, το οποίο ανανεώνει μέσω της υπορουτίνας global schema model update, βάσει των ερωτημάτων. Η ανανέωση αυτή συνίσταται στα παρακάτω:

- Προσθήκη νέων σχέσεων που εμφανίζονται στο ερώτημα
- Προσθήκη νέων ιδιοτήτων που εμφανίζονται στο ερώτημα
- Προσθήκη νέων συνδέσμων που εμφανίζονται στο ερώτημα
- Δημιουργία νέων μονοπατιών συνένωσης για όλα τα ζεύγη πινάκων του σχήματος και προσθήκη των μονοπατιών αυτών στο γενικό μοντέλο σχήματος.

Στο σημείο αυτό να παρατηρήσουμε ότι ενώ κατά τη δημιουργία του γενικού μοντέλου ενός σχήματος όταν το σχήμα είναι πλήρες, προσθέτουμε τους συνδέσμους που υπονοούν οι περιορισμοί εξωτερικών κλειδιών. Στην περίπτωση ημιτελών σχημάτων όμως, κάτι τέτοιο δεν είναι γνωστό. Τα εισερχόμενα ερωτήματα δεν μας δίνουν πληροφορίες σχετικά με τα εξωτερικά κλειδιά του σχήματος τους. Για το λόγο αυτό, οι σύνδεσμοι που προσθέτουμε στο σχήμα είναι οι ξεχωριστοί σύνδεσμοι που συμπεριλαμβάνει ο χρήστης στο ερώτημα του.

Η δημιουργία των νέων μονοπατιών συνένωσης που προκύπτουν από τους νέους συνδέσμους του ερωτήματος γίνεται με τον τρόπο που περιγράφηκε για τη λειτουργία μεταξύ γνωστών σχημάτων.

Το δεύτερο σημείο που ο αλγόριθμος διαφοροποιείται είναι η παραγωγή των ισοδυναμιών που θα χρησιμοποιηθούν για την κατασκευή των αντιστοιχιών. Είναι προφανές, ότι σε περίπτωση που ένα εκ των δύο σχημάτων εμπλουτίζεται με νέες σχέσεις και ιδιότητες, ο Matcher, Coma++, ενδέχεται να ανακαλύψει νέες ισχυρότερες ισοδυναμίες ιδιοτήτων, οι οποίες προηγουμένως να μην ήταν γνωστές, δεδομένου ότι συγκεκριμένες ιδιότητες απουσίαζαν από το σχήμα. Έτσι λοιπόν, τα δύο σύνολο κατευθυνόμενων ισοδυναμιών,

$C_{\Sigma}(S_T,S_S)$ και $C_{\mathcal{I}}(S_T,S_S)$ οφείλονται να ενισχύονται κάθε φόρα που καταφθάνει ένα νέο ερώτημα, με τις νέες ισοδυναμίες που ανακάλυψε ο Schema Matcher που χρησιμοποιούμε. Με άλλα λόγια η διαδικασία του Schema Matching, τώρα δε γίνεται μια φορά κατά την αρχικοποίηση των ισοδυναμιών, αλλά καθίσταται αναγκαία κάθε φορά που το απομακρυσμένο σχήμα εμπλουτίζεται με νέα στοιχεία.

Το τρίτο σημείο διαφοροποίησης που παρατηρούμε συγκρίνοντας τα δύο σχήματα 4 g και 4 b είναι ο τρόπος που λαμβάνεται η ανάδραση του χρήστη.

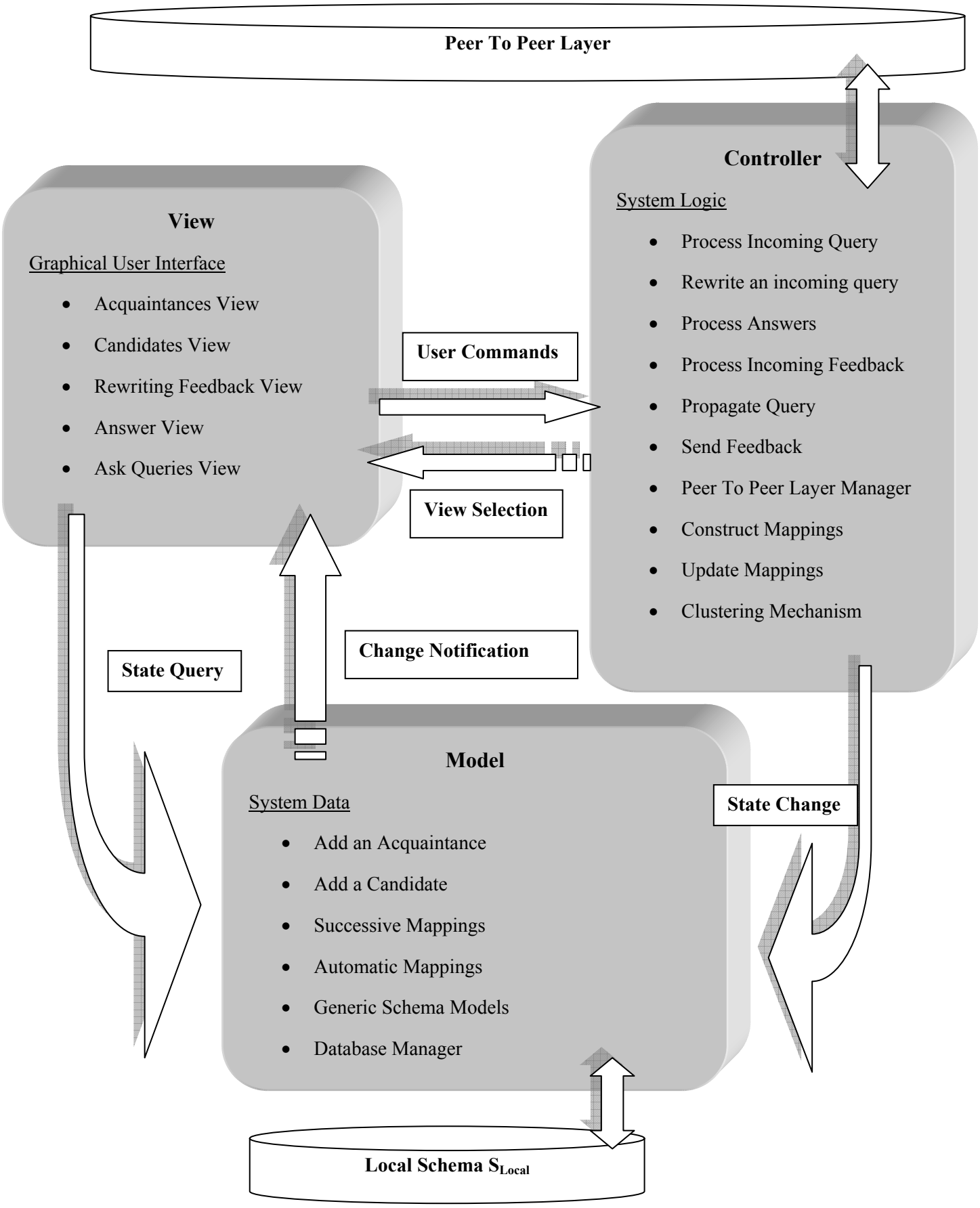
Με πλήρη σχήματα, σε περίπτωση δηλαδή που ο μηχανισμός χρησιμοποιείται ως ξεχωριστό εργαλείο εύρεσης αντιστοιχιών μεταξύ δύο σχημάτων, όπως είδαμε ο χρήστης χαρακτηρίζει τα στοιχεία των παραγόμενων αντιστοιχιών με τις κατηγορίες που αναφέραμε στην παράγραφο 4.5. Όταν όμως το ο μηχανισμός αποτελεί μέρος ενός γενικού συστήματος p2p βάσεων δεδομένων, ο χρήστης δεν κρίνει άμεσα τις αντιστοιχίες πινάκων, αλλά έμμεσα προσφέροντας ανάδραση για τη μετάφραση του ερωτήματος που έστειλε. Δεδομένου ότι οι αλγόριθμοι μετάφρασης ερωτημάτων που χρησιμοποιούνται σε συστήματα p2p βάσεων δεδομένων, προσφέρουν μια αναλυτική περιγραφή για την μετάφραση κάθε ιδιότητας του ερωτήματος (ποια ισοδυναμία και ποιο μονοπάτι συνένωσης χρησιμοποιείται), η ανάδραση μέσω χαρακτηρισμού των επιμέρους μεταφράσεων που εξηγήσαμε, μπορεί να εφαρμοσθεί και σε αυτή την περίπτωση. Στο σύστημα GrouPeer όμως, το εισερχόμενο ερώτημα Q_{orig} μεταφράζεται τόσο βάσει των νέων αντιστοιχιών, παράγοντας το ερώτημα $Q_{automatic}$ όσο και βάσει των διαδοχικών αντιστοιχιών που υπάρχουν ανάμεσα στους κόμβους της διαδρομής του ερωτήματος στο δίκτυο, παράγοντας έτσι το ερώτημα $Q_{successively}$. Η σημασιολογία των δύο εκδοχών του ερωτήματος Q_{orig} , $Q_{automatic}$, $Q_{successively}$ ενδέχεται να είναι διαφορετική μιας και πιθανώς να εμπλέκονται στη μετάφραση διαφορετικές ισοδυναμίες και μονοπάτια συνένωσης. Παρόλα αυτά, η επεξεργασία της ανάδραση του χρήστη θα πρέπει να βελτιώνει τις νέες αντιστοιχίες, ακόμη και αν το σύστημα επιλέξει να απαντήσει το ερώτημα $Q_{successively}$. Κάτι τέτοιο επιτυγχάνεται με ελάχιστη τροποποίηση του αλγορίθμου βελτίωσης αντιστοιχιών (αλγόριθμος 4 d), ώστε να συμπεριλάβει και την περίπτωση όπου το ερώτημα, μεταφράζεται μέσω διαδοχικών αντιστοιχιών κόμβων.

5 Σχεδίαση και Υλοποίηση Συστήματος

Στο συγκεκριμένο κεφάλαιο, θα δούμε τη γενικότερη αρχιτεκτονική του συστήματος GrouPeer και πως σε αυτό το πλαίσιο ενσωματώνεται ο μηχανισμός ημιαυτόματης παραγωγής των αντιστοιχιών μεταξύ των σχημάτων δύο κόμβων. Αρχικά παρατίθενται οι βασικότερες αρχές σχεδίασης πάνω στις οποίες στηρίχθηκε η δομή του συστήματος και μια γενική εικόνα για την αρχιτεκτονική του και ύστερα θα περιγράψουμε συνοπτικά κάθε επιμέρους τμήμα αυτού.

5.1 Αρχιτεκτονική

Η βασική σχεδίαση του συστήματος GrouPeer στηρίχθηκε κυρίως στο αρχιτεκτονικό μοτίβο Model-View-Controller (MVC). Όπως φαίνεται στο σχήμα 5 a, όπου παρουσιάζεται η σχεδίαση ενός κόμβου που συμμετέχει στο σύστημα, τα δεδομένα του συστήματος, η λογική των διαφόρων αλγορίθμων και ο τρόπος που το σύστημα παρουσιάζεται στο χρήστη, αποτελούν αυτοτελείς οντότητες που επικοινωνούν μεταξύ τους. Με τον διαχωρισμό αυτό, καθιστούμε το σύστημα εύκολα επεκτάσιμο, ώστε να μπορούμε να προσθέσουμε νέα δεδομένα, πιο προηγμένους, σύνθετους αλγορίθμους, και νέες διαπροσωπείες, χωρίς να επηρεάζουμε τη βασική αρχιτεκτονική δομή του συστήματος και τις υπόλοιπες αυτοτελείς οντότητες.



Σχήμα 5 a: Η αρχιτεκτονική του συστήματος GrouPeer

Η οντότητα View του παραπάνω σχήματος περιλαμβάνει όλες εκείνες τις λειτουργίες του συστήματος, που αφορούν την επικοινωνία του συστήματος με τον χρήστη. Πιο συγκεκριμένα, στην οντότητα αυτή περιλαμβάνεται η κύρια διεπαφή του προγράμματος, μέσω της οποίας ο χρήστης μπορεί να στείλει ερωτήματα στο δίκτυο, η εμφάνιση των απαντήσεων - δεδομένων που επέστρεψε το δίκτυο σε μία ερώτηση του χρήστη, η εμφάνιση των γειτονικών κόμβων αλλά και των υποψήφιων γειτόνων του συγκεκριμένου κόμβου. Μέσω της οντότητας αυτής, ο χρήστης μπορεί να προτείνει σε έναν υποψήφιο κόμβο να γίνει γείτονάς του, να στείλει ανάδραση για τις μεταφράσεις των ερωτημάτων του και να και να δει τις αντιστοιχίες που έχει κατασκευάσει με τους γειτονικούς του κόμβους. Οι λειτουργίες της οντότητας αυτής δεν υλοποιήθηκαν στα πλαίσιο αυτής της διπλωματικής εργασίας και παραμένουν ως μελλοντική επέκταση.

Η οντότητα Model περιλαμβάνει όλα τα δεδομένα της εφαρμογής, τα οποία παρουσιάζονται στον χρήστη μέσω της οντότητας View και τροποποιούνται καθώς συμβαίνουν διάφορες αλλαγές (ανανεώσεις σχημάτων, βελτιώσεις αντιστοιχιών κλπ) από την οντότητα Controller.

Τα δεδομένα που πρέπει να διατηρεί ένας κόμβος στο GrouPeer είναι το σύνολο των γειτόνων του, το σύνολο των υποψήφιων γειτόνων του, το σύνολο με τις GAV και LAV διαδοχικές αντιστοιχίες που έχει θεσπίσει με τους γειτονικούς κόμβους και το σύνολο με τις αυτόματα παραγόμενες αντιστοιχίες που βελτιώνει σταδιακά με τους υποψήφιους γείτονες. Ακόμη κάθε κόμβος κρατά πληροφορίες για τα γενικά μοντέλα σχημάτων των υποψήφιων γειτονικών κόμβων, καθώς και μια σύνδεση με τη δική του τοπική βάση. Οι βασικές κλάσεις της οντότητας αυτής είναι οι εξής: GenericModel, Mapping, Peer, MyPeer, DatabaseManager, RelationalToModel.

Τέλος η οντότητα Controller ενσωματώνει όλες τις λειτουργικές μονάδες του συστήματος, με άλλα λόγια τη λογική του. Έτσι σε αυτή την οντότητα, περιλαμβάνεται ο μηχανισμός μετάφρασης ερωτημάτων, βάσει GAV/LAV αντιστοιχιών, ο μηχανισμός αυτόματης παραγωγής αντιστοιχιών μεταξύ δύο κόμβων, η επεξεργασία της ανάδρασης για μεταφράσεις ερωτημάτων και ο τρόπος που επικοινωνεί ο κόμβος με το peer to peer layer, πάνω στο οποίο είναι συνδεδεμένος. Οι βασικές κλάσεις της οντότητας αυτής είναι οι εξής. PeerToPeerLayerManager, RewriteMechanism, MappingAlgorithm και MainControl. Η μόνη λειτουργία της οντότητας αυτής που απομένει προς υλοποίηση είναι ο τρόπος με τον οποίο οι κόμβοι δημιουργούν προοδευτικά ομάδες με παρόμοια σημασιολογικά σχήματα και ισχυρές αντιστοιχίες μεταξύ τους (Clustering Mechanism).

5.2 Περιγραφή Κλάσεων

Στη συνέχεια δίνουμε μια σύντομη συνοπτική περιγραφή των βασικότερων κλάσεων του συστήματος. Για κάθε κλάση περιγράφονται οι λειτουργίες που αυτή παρέχει και αναφέρεται η βασική οντότητα στην οποία ανήκει.

Κλάσεις που σχετίζονται με την οντότητα Model

5.2.1 Η κλάση Attribute Node

Η κλάση αυτή μοντελοποιεί μια ιδιότητα σε ένα σχήμα.

A) Αξιοσημείωτα πεδία

- **private** TableNode parent;

Το αντικείμενο που μοντελοποιεί τον πίνακα, στον οποίο ανήκει η συγκεκριμένη ιδιότητα

- **private** String name;

Το απλό όνομα της ιδιότητας

- **private** String treename;

Το πλήρες όνομα της ιδιότητας (Table.Attribute)

- **private boolean** FK = **false**;

Αληθές, αν η ιδιότητα είναι εξωτερικό κλειδί σε κάποιον πίνακα του σχήματος

- **private boolean** PK = **false**;

Αληθές αν η ιδιότητα συμμετέχει στο κύριο κλειδί του πίνακα

- **private** AttributeNode crosRef;

Το AttributeNode αντικείμενο που δείχνει η ιδιότητα, σε περίπτωση που αυτή είναι εξωτερικό κλειδί

- **private boolean** isNew;

Αληθές, αν η ιδιότητα ανακαλύφθηκε από τον κόμβο, με το τελευταίο εισερχόμενο ερώτημα

5.2.2 Η κλάση TableNode

Η κλάση αυτή μοντελοποιεί έναν πίνακα σε ένα σχήμα

A) Αξιοσημείωτα Πεδία

- **private** String name;

Το όνομα του πίνακα

- **private** List<AttributeNode> attributes;

Το σύνολο με τα AttributeNode αντικείμενα που εκπροσωπούν τις ιδιότητες του πίνακα αυτού

- **private** List<AttributeNode> keys;

Το σύνολο με τα AttributeNode αντικείμενα που απαρτίζουν το κύριο κλειδί του πίνακα

- **private** Vector<Join> crosses;

Το σύνολο με τους συνδέσμους, που προκύπτουν από τους περιορισμούς εξωτερικών κλειδιών του πίνακα αυτού προς άλλους πίνακες

- **private** Hashtable<TableNode, Vector<Join>> myhash;

Μια κατηγοριοποίηση των πιθανών συνδέσμων, στους οποίους εμπλέκεται ο πίνακας, με βάση τον άλλο πίνακα του συνδέσμου

- **private boolean** isNew;

Αληθές, αν ο πίνακας αυτός ανακαλύφθηκε από τον κόμβο με το τελευταίο εισερχόμενο ερώτημα

- **private** Vector<Join> joins;

Το σύνολο με όλους τους πιθανούς συνδέσμους του σχήματος, στους οποίους εμπλέκεται ο συγκεκριμένος πίνακας. Περιλαμβάνει τόσο τα crosses του πίνακα, όσο και συνδέσμους από εξωτερικά κλειδιά άλλων πινάκων που αναφέρονται στον πίνακα αυτόν.

B) Αξιοσημείωτες Μέθοδοι

- **public** AttributeNode findAttribute(String name)

Επιστρέφει το αντικείμενο AttributeNode που αντιστοιχεί στην ιδιότητα του πίνακα με όνομα name

- **public** void setCrosses()

Δημιουργεί τους συνδέσμους (Joins) που προκύπτουν από τους περιορισμούς εξωτερικών κλειδιών του πίνακα και τους αποθηκεύει στο σύνολο Crosses αυτού

- **public** void addAttribute(AttributeNode k)

Προσθέτει ένα αντικείμενο AttributeNode στο σύνολο των ιδιοτήτων του πίνακα

- **public** void removeAttribute(AttributeNode k)

Αφαιρεί μια ιδιότητα από τον πίνακα

- **public** void printJoins()

Τυπώνει στην οθόνη όλους τους πιθανούς συνδέσμους, στους οποίους εμπλέκεται ο πίνακας

5.2.3 Η κλάση Join

Η κλάση αυτή μοντελοποιεί ένα σύνδεσμο μεταξύ δύο πινάκων T1,T2 στη μορφή T1.a1 = T2.a2

A) Αξιοσημείωτα πεδία

- **private** String tab1;

Το όνομα του πρώτου πίνακα του συνδέσμου

- **private** String tab2;

Το όνομα του δεύτερου πίνακα του συνδέσμου

- **private** String att1;

Το όνομα της πρώτης ιδιότητας του συνδέσμου

- **private** String att2;

Το όνομα της δεύτερης ιδιότητας του συνδέσμου

- **private** String renamedtab1;

Το ψευδώνυμο (αν χρησιμοποιείται) για τον πρώτο πίνακα

- **private** String renamedtab2;
  Το ψευδώνυμο (αν χρησιμοποιείται) για τον πρώτο πίνακα

B) Αξιοσημείωτες μέθοδοι

- **public** Join(String aJoin)

Ο κύριος κατασκευαστής της κλάσης. Παίρνει ως είσοδο ένα σύνδεσμο σε string μορφή ("T1.a1 = T2.a1") και αρχικοποιεί τα πεδία με τις κατάλληλες τιμές

- **public** String toString()

Επιστρέφει την String αναπαράσταση του συνδέσμου

- **public** Join copy()

Επιστρέφει ένα νέο αντικείμενο Join, καθόλα όμοιο με το αρχικό

- **public** void swap()

Αντιστρέφει ένα σύνδεσμο. Πιο συγκεκριμένα ο σύνδεσμος T1.a1 = T2.a2 γίνεται T2.a2 = T1.a1

5.2.4 Η κλάση JoinPath

Με την κλάση αυτή μοντελοποιούμε ένα μονοπάτι συνένωσης.

A) Αξιοσημείωτα πεδία

- **private static final long** *serialVersionUID* ;

Το πεδίο αυτό, είναι απαραίτητο για το serialization του αντικειμένου

- **private int** ID;

Το αναγνωριστικό του Join path που το χαρακτηρίζει μοναδικά

- **private** JoinPath normal;

Σε περίπτωση που το μονοπάτι συνένωσης είναι ανεστραμμένο, το πεδίο αυτό αποθηκεύει το join path από το οποίο έχει προέλθει

- **private boolean** is_reversed;

Αληθές, όταν το μονοπάτι είναι ανεστραμμένο

- **private** Vector<Join> listofJoins;

Το διατεταγμένο σύνολο με τους συνδέσμους που απαρτίζουν το μονοπάτι

- **private** Vector<MappAttr> usedByMapAttrs;

Το σύνολο με τα Mapping Elements αντικείμενα που χρησιμοποιούν το μονοπάτι σε μια δεδομένη χρονική στιγμή

- **private** JoinPath formatted;

Το πεδίο αυτό αποθηκεύει το ίδιο μονοπάτι συνένωσης του αντικειμένου, εκφράζοντας όμως τους πίνακες που συμμετέχουν σε αυτό με τις απαραίτητες μετονομασίες, ώστε να μπορεί να χρησιμοποιηθεί σε μια αντιστοιχία

B) Αξιοσημείωτες μέθοδοι

- **public void** addJoin(Join join)

Προσθέτει ένα σύνδεσμο στο τέλος του συνόλου διατεταγμένων συνδέσμων του μονοπατιού

- **public** String getLastTable()

Επιστρέφει τον τελευταίο πίνακα του μονοπατιού

- **public** String getFirstTable()

Επιστρέφει τον πρώτο πίνακα του μονοπατιού

- **public** Vector<Join> getListOfJoin()

Επιστρέφει το διατεταγμένο σύνολο με τους συνδέμους, που απαρτίζουν το μονοπάτι

- **public** Vector<JoinPath> constructHiddenJoiPaths()

Κατασκευάζει ένα σύνολο με όλα τα υπομονοπάτια του συγκεκριμένου join path

- **public void** printHiddenJoinPaths()

Τυπώνει το σύνολο με τα υπομονοπάτια του join path

- **public** String toString()

Επιστρέφει την string αναπαράσταση του join path (T1.a1 = T2.a2 AND T2.a22 = T3.a3…AND Tn-1.an-1 = Tn.an)

- **public boolean** equals(JoinPath jp)

Αληθές, όταν το μονοπάτι jp είναι ίδιο με το joinpath (έχουν το ίδιο σύνολο διατεταγμένων συνδέσμων).

- **public** JoinPath getReversed()

Επιστρέφει ένα νέο αντικείμενο JoinPath που απαρτίζεται από το ίδιο σύνολο συνδέσμων του αντικειμένου, αλλά με ανεστραμμένη σειρά. Πχ το ανεστραμμένο μονοπάτι του join path T1.a1 = T2.a2 AND T2.a22 = T3.a3…AND Tn-1.an-1 = Tn.an είναι το

Tn.an = Tn-1.an-1 AND …T3.a3 = T2.a22 AND T2.a2 = T1.a1

- **public void** removeMappAttr(MappAttr map)

Αφαιρεί το αντικείμενο MappAttr (που μοντελοποιεί ένα στοιχείο Mapping Element του πίνακα Matrix) από το σύνολο usedByMapAttrs.

- **public int** getDistance()

Επιστρέφει το μήκος του μονοπατιού

- **public** JoinPath copy()

Επιστρέφει ένα νέο αντικείμενο JoinPath καθόλα όμοιο με το αρχικό

- **public boolean** canBeUsedBy(MappAttr map)

Ελέγχει αν ένα συγκεκριμένο Mapping Element map, δύναται να χρησιμοποιήσει το μονοπάτι αυτό

- **public boolean** containsJoinPath(JoinPath contained)

Ελέγχει αν το μονοπάτι contained αποτελεί υπομονοπάτι του συγκεκριμένου Join Path

- **public float** getValue()

Επιστρέφει την ευριστική τιμή του μονοπατιού, που χαρακτηρίζει την ισχύ της σημασιολογίας του.

5.2.5 Η κλάση TablePair

Η κλάση αυτή μοντελοποιεί ένα ζεύγος πινάκων στο γενικό μοντέλο ενός σχήματος.

Κάθε ζεύγος αποθηκεύει τους πιθανούς συνδέσμους που υφίστανται μεταξύ των δύο πινάκων, αλλά και τα πιθανά μονοπάτια συνένωσης των δύο πινάκων

A) Αξιοσημείωτα πεδία

- **private** String tableA;

Ο πρώτος πίνακας του ζεύγους

- **private** String tableB;

Ο δεύτερος πίνακας του ζεύγους

- **private** Hashtable<Integer,Vector<JoinPath>> tbpairHash;

Το σύνολο με όλα τα πιθανά μονοπάτια συνένωσης των δύο πινάκων, χωρισμένο σε επιμέρους σύνολα, ανάλογα με το μήκος του κάθε μονοπατιού

- **private** Vector<Join> joins;

Το σύνολο με όλους τους πιθανούς συνδέσμους που υφίστανται μεταξύ των δυο πινάκων

- **private int** nextID;

Το αναγνωριστικό που θα τεθεί στο επόμενο μονοπάτι συνένωσης που θα ανακαλυφθεί μεταξύ των δύο πινάκων

B) Αξιοσημείωτες μέθοδοι

- **public void** addJoin(Join join)

Προσθέτει το σύνδεσμο join στο σύνολο συνδέσμων του ζεύγους

- **public** String getTableA()

Επιστρέφει τον πρώτο πίνακα του ζεύγους

- **public** String getTableB()

Επιστρέφει τον δεύτερο πίνακα του ζεύγους

- **public** JoinPath findJoinPath(**int** id)

Επιστρέφει το μονοπάτι συνένωσης από το σύνολο του ζεύγους πινάκων που έχει αναγνωριστικό id

- **public void** addJoinPath(JoinPath a)

Προσθέτει το μονοπάτι συνένωσης a στο αντίστοιχο σύνολο, προσδίδοντας του ένα νέο αναγνωριστικό

- **public** JoinPath getJoinPath(**int** distance,MappAttr map,JoinPath contained)

Επιστρέφει ένα μονοπατι συνένωσης από το σύνολο όλων των join paths του ζεύγους, μήκους distance, το οποίο περιέχει το μονοπάτι contained και μπορεί να χρησιμοποιηθεί από το Mapping Element map. Αν δε βρει κάποιο μονοπάτι στο σύνολο, που πληρεί αυτές τις προϋποθέσεις, επιστρέφει NULL.

- **public** JoinPath getJoinPath(**int** distance,MappAttr map)

Όμοια με την προηγούμενη μέθοδο, χωρίς όμως τον περιορισμό το μονοπάτι που θα βρεθεί να περιέχει κάποιο άλλο μονοπάτι

- **public void** print()

Τυπώνει στην οθόνη του χρήστη την περιγραφή του ζεύγους (Τα ονόματα των δύο πινάκων και τα σύνολα συνδέσμων και μονοπατιών)

- **public** JoinPath findJoinPath(JoinPath tempj)

Βρίσκει το μονοπάτι εκείνο του συνόλου που είναι καθόλα όμοιο με το μονοπάτι tempj

5.2.6 Η κλάση JoinPathPool

Η κλάση αυτή μοντελοποιεί τη συλλογή όλων των μονοπατιών συνένωσης που έχουν ανακαλυφθεί για κάποιο σχήμα βάσης. Με άλλα λόγια περιλαμβάνει ένα σύνολο με όλα τα

ζεύγη πινάκων στο σχήμα, αλλά και τις λειτουργίες για την ανακάλυψη νέων μονοπατιών συνένωσης.

A) Αξιοσημείωτα πεδία

- **public static int** *MAX_JOIN_DISTANCE;*

Η μέγιστο μήκος μονοπατιών συνένωσης, τα οποία θα δημιουργηθούν ανάμεσα στους πίνακες του σχήματος

- **private** Vector<TablePair> tablePairs;

Το σύνολο με όλα τα ζεύγη των πινάκων του σχήματος

- **private** GenericModel gm;

Το αντικείμενο που μοντελοποιεί το γενικό μοντέλο του σχήματος, του οποίου θα κατασκευαστούν τα πιθανά μονοπάτια συνένωσης

B) Αξιοσημείωτες μέθοδοι

- **public void** updateTablePairs(String table)

Η μέθοδος αυτή χρησιμοποιείται όταν ένας νέος πίνακας με όνομα table προστίθεται στο σχήμα. Δημιουργεί όλα τα νέα ζεύγη πινάκων μεταξύ του πίνακα αυτού και όλων των άλλων πινάκων του σχήματος και τα προσθέτει στο σύνολο tablePairs

- **private void** addJoin(Join join)

Προσθέτει ένα σύνδεσμο μεταξύ δύο πινάκων στο αντίστοιχο σύνολο συνδέσμων του ζεύγους των δύο πινάκων

- **public** Vector<JoinPath> produceDirectedJoinPaths(String fromTable,String toTable)

Δημιουργεί όλα τα μονοπάτια συνένωσης μοναδιαίου μήκους, από τον πίνακα fromTable, προς τον πίνακα toTable.

- **public** Vector<JoinPath> produceJoinPaths(String fromTable, String toTable,int distance)

Δημιουργεί όλα τα μονοπάτια συνένωσης μεταξύ των πινάκων fromTable και toTable μήκους distance, καλώντας αναδρομικά την μέθοδο produceDirectedJoinPaths και τα αποθηκεύει στο αντίστοιχο σύνολο του ζεύγους πινάκων fromTable-toTable

- **public** TablePair findTablePair(String table1, String table2)

Επιστρέφει το ζεύγος πινάκων table1 – table2 από το σύνολο tablePairs

- **public** JoinPath mergeJoinPaths(JoinPath jp1, JoinPath jp2)

Συμπτύσσει τα δύο μονοπάτια συνένωσης jp1, jp2 σε ένα νέο μονοπάτι jp1—jp2

- **public void** makeJoinPaths()

Δημιουργεί για όλα τα ζεύγη πινάκων του σχήματος όλα τα μονοπάτια συνένωσης μέχρι μήκος MAX_JOIN_DISTANCE και τα αποθηκεύει στο αντίστοιχο σύνολο του εκάστοτε table pair

- **public void** printToFile(String filename)

Τυπώνει μια περιγραφή του αντικείμενου στο αρχείο με όνομα filename

5.2.7 Η κλάση GenericModel

Η κλάση αυτή μοντελοποιεί το γενικό μοντέλο ενός σχήματος, όπως αυτό περιγράφηκε στην αντίστοιχη παράγραφο του κεφαλαίου 4. Αντικείμενα της κλάσης αυτής είτε αρχικοποιούνται με σύνδεση σε μια σχεσιακή βάση δεδομένων, είτε ενημερώνονται σταδιακά από τα εισερχόμενα ερωτήματα του χρήστη. Ένα αντικείμενο της κλάσης αυτής, είναι ικανό να περιγράψει πλήρως ένα σχήμα βάσης, ενσωματώνοντας όλες τις πληροφορίες που απαιτούνται (ονόματα πινάκων, ιδιοτήτων, εμφάνιση συνδέσμων, δημιουργία μονοπατιών συνένωσης κλπ).

A) Αξιοσημείωτα πεδία

- **public static final int** *MINE* = 1;

Τον τύπο αυτό έχουν τα μοντέλα εκείνα που περιγράφουν το σχήμα βάσης του τοπικού κόμβου

- **public static final int** *NOTMINE* = 0;

Τον τύπο αυτό έχουν τα μοντέλα που περιγράφουν το σχήμα βάσης ενός απομακρυσμένου κόμβου

- **private int** ownertype;

Το πεδίο αυτό δείχνει αν το μοντέλο περιγράφει το σχήμα του τοπικού ή του απομακρυσμένου κόμβου (mine or not_mine)

- **private boolean** modelChanged;

Αληθές αν το γενικό μοντέλο σχήματος, τροποποιήθηκε από το τελευταίο εισερχόμενο ερώτημα του απομακρυσμένου κόμβου

- **private** String peer_name;

Το όνομα του κόμβου, που το γενικό μοντέλο αυτό περιγράφει το σχήμα της βάσης του

- **private** JoinpathPool jpPool;

Η συλλογή με όλα τα μονοπάτια συνένωσης και τους συνδέσμους για τους πίνακες αυτού του μοντέλου

- **private** Vector<String> allkeys;

Το σύνολο με τα ονόματα όλων των κύριων κλειδιών του σχήματος

- **private** Vector<String[]> allfkeys;

Το σύνολο με όλους τους περιορισμούς εξωτερικών κλειδιών του σχήματος

- **private** String schemaName;

Το όνομα του σχήματος που περιγράφει το μοντέλο

- **private** List<TableNode> tables;

Το σύνολο με όλα τα TableNode αντικείμενα που περιγράφουν τους πίνακες του σχήματος

B) Αξιοσημείωτες μέθοδοι

- **public void** addTable(TableNode k)

Προσθέτει το αντικείμενο TableNode k στο σύνολο πινάκων του μοντέλου

- **private boolean** addAttribute(String att)

Προσθέτει μια ιδιότητα με όνομα att στο μοντέλο κατασκευάζοντας το αντίστοιχο αντικέιμενο AttributeNode και προστίθοντας αυτό στον αντίστοιχο πίνακα

- **public void** addPkey(String att)

Προσθέτει ένα κύριο κλειδί στο μοντέλο

- **public void** addFkey(String from, String to)

Προσθέτει έναν περιορισμός εξωτερικού κλειδιού (from→to) στο μοντέλο

- **public** AttributeNode findAttribute(String treename)

Επιστρέφει το αντικείμενο AttributeNode που περιγράφει την ιδιότητα με πλήρες όνομα treename

- **public** TableNode findTable(String name)

Επιστρέφει το αντικείμενο TableNode που περιγράφει τον πίνακα με το όνομα name

- **public void** print()

Τυπώνει στην οθόνη του χρήστη την περιγραφή του γενικού αυτού μοντέλου

- **public void** extractToXML()

Εξάγει την περιγραφεί του μοντέλου σε ένα XML αρχείο

- **public** String getPeerName()

Επιστρέφει το όνομα του κόμβου, του οποίου το σχήμα περιγράφεται με αυτό το μοντέλο

- **public** JoinpathPool getJoinPathPool()

Επιστρέφει τη συλλογή μονοπατιών συνένωσης για τους πίνακες αυτού του μοντέλου

5.2.8 Η κλάση Peer

Η κλάση αυτή μοντελοποιεί το πλαίσιο που θα περιγράφονται οι απομακρυσμένοι κόμβοι, είτε αυτοί είναι γείτονες, είτε υποψήφιοι γείτονες. Στην κλάση αυτή κρατείται το όνομα του

κάθε κόμβου, η κατηγορία αυτού (γείτονας ή υποψήφιος), η κατάσταση του (ενεργός ή ανενεργός) και το γενικό μοντέλο που περιγράφει το σχήμα της βάσης του.

A) Αξιοσημείωτα πεδία

- **private final static int** *FOUND* = 1;

Με την τιμή αυτή χαρακτηρίσμο περιγράφουμε τους γειτονικούς κόμβους, που ανακαλύφθηκαν από τον εκάστοτε τοπικό κόμβο στο δίκτυο

- **private final static int** *NOT_FOUND* = 0;

Παρόμοια με το παραπάνω πεδίο, για τους κόμβους γείτονες που δεν έχουν ανακαλυφθεί στο δίκτυο

- **private final static int** *NEIGHBOR* = 1;

Χαρακτηρίζει τους κόμβους γείτονες

- **private final static int** *UNKNOWN* = 0;

Χαρακτηρίζει τους υποψήφιους κόμβους

- **private** GenericModel peerModel;

Το γενικό μοντέλο που περιγράφει το σχήμα το κάθε κόμβου (γείτονα ή υποψηφίου)

- **private** Vector<Mapping> gavMappings;

Το σύνολο με τις GAV αντιστοιχίες του τοπικού κόμβου προς τον συγκεκριμένο γείτονα

- **private** Vector<Mapping> lavMappings;

Το σύνολο με τις LAV αντιστοιχίες του κόμβου προς τον συγκεκριμένο υποψήφιο γείτονα

- **private** String peerName;

Το όνομα του κόμβου. Το πεδίο αυτό λειτουργεί και ως μοναδικό αναγνωριστικό για τον κάθε κόμβο

- **private** GrouPeerAdvertisement peerAdv;

Το αντικείμενο αυτό αποτελεί μια μορφή ταυτότητας του κόμβου στο p2p δίκτυο, ώστε ο κόμβος να είναι σε θέση να δηλώσει την παρουσία του σε απομακρυσμένους κόμβους

- **private** PipeAdvertisement pipeAdv;

Ομοίως με το προηγούμενο πεδίο, το αντικείμενο αυτό περιλαμβάνει όλα τα στοιχεία του κόμβου, που αφορούν τον τρόπο επικοινωνίας αυτού με τους απομακρυσμένους κόμβους.

- **private int** relation; // neigbor or unknown

Η σχέση που έχει ο κόμβος με τον τοπικό κόμβο (γείτονας ή υποψήφιος)

- **private int** status; // Found or notFound;

Η κατάσταση του κόμβου (ανακαλυφθείς ή όχι)

Β) Αξιοσημείωτες μέθοδοι

- **public** Vector<Mapping> getMappings(**int** type)

Επιστρέφει το σύνολο αντιστοιχιών τύπου type (GAV ή LAV) που κρατά ο τοπικός κόμβος με το συγκεκριμένο κόμβο

- **public** Mapping getMappingOf(String tbname,**int** type)

Επιστρέφει την αντιστοιχία τύπου type, του πίνακα με όνομα tbname

- **public** String getName()

Επιστρέφει το όνομα του κόμβου

- **public** GenericModel getModel()

Επιστρέφει το γενικό μοντέλο σχήματος του κόμβου

- **public boolean** isFound()

Αληθές αν ο κόμβος έχει ανακαλυφθεί στο δίκτυο από τον τοπικό κόμβο

- **public boolean** isNeighbor()

Αληθές αν ο κόμβος είναι γείτονας του τοπικού κόμβου

5.2.9 Η κλάση MyPeer

Με την κλάση αυτή περιγράφονται οι πληροφορίες που κρατά ο τοπικός κόμβος. Έτσι, τα συγκεκριμένα αντικείμενα περιλαμβάνουν το γενικό μοντέλο του δικού τους τοπικού σχήματος, το σύνολο των υποψήφιων και γειτονικών κόμβων, τα γενικά μοντέλα και τις αντιστοιχίες αυτών.

A) Αξιοσημείωτα πεδία

- **private** String myPeerName;

Το όνομα του τοπικού κόμβου

- **private** String myDbName;

Το όνομα της βάσης του τοπικού κόμβου

- **private** String myDbUser;

Το username για σύνδεση στη βάση του τοπικού κόμβου

- **private** String myDbPass;

Το password για σύνδεση στη βάση του τοπικού κόμβου

- **private** String mySchemaName;

Το όνομα του σχήματος του τοπικού κόμβου

- **private** String myIpAddress;

Η IP διεύθυνση του τοπικού κόμβου

- **private** String myPortNumber;

Το port number για σύνδεση στη βάση του τοπικού κόμβου

- **private** Hashtable<String,MappingAlgorithm> mapAlgols;

Οι αναφορές στους ξεχωριστούς μηχανισμούς παραγωγής και βελτίωσης αντιστοιχιών που κρατά ο τοπικός κόμβος για κάθε υποψήφιο γείτονα. Κλειδιά του πίνακα hash αυτού είναι τα ονόματα των υποψήφιων γειτόνων

- **private** GenericModel myModel;

Το γενικό μοντέλο που περιγράφει το σχήμα του τοπικού κόμβου

- **private** Vector<Peer> listOfCandidates;

Το σύνολο με όλους τους υποψήφιους γείτονες του τοπικού κόμβου

- **private** Vector<Peer> myNeighboors;

Το σύνολο με όλους τους γείτονες του τοπικού κόμβου

- **private** GrouPeerAdvertisement myGrouPeerAdvertisement;

Η ταυτότητα του τοπικού κόμβου στο p2p δίκτυο

- **private** PipeAdvertisement myPipeAdv;

Τα απαραίτητα στοιχεία για επικοινωνία με τον κόμβο αυτό.

- **private** RelationalToModel rm;

Μια αναφορά στον μηχανισμό δημιουργίας γενικού μοντέλου από μια σχεσιακή βάση δεδομένων, για την αρχικοποίηση των μοντέλων σχήματος

B) Αξιοσημείωτες μέθοδοι

- **public** GenericModel getMyModel()

Επιστρέφει το γενικό μοντέλο σχήματος του τοπικού κόμβου

- **public void** initializeFromConfigFile(String peername)

Αρχικοποιεί ορισμένα πεδία του αντικειμένου, σχετικά με τα σταθερά στοιχεία του τοπικού κόμβου (όνομα βάσης, username & password, όνομα κόμβου κλπ) από το αντίστοιχο configuration αρχείο του κόμβου

- **public void** initializeCandidates()

Αρχικοποιεί τους υποψήφιους γείτονες του εκάστοτε τοπικού κόμβου, από σχετικό αρχείο

- **public void** initializeNeighboors()

Αρχικοποιεί τους γείτονες του εκάστοτε τοπικού κόμβου, από σχετικό αρχείο

- **public** MappingAlgorithm getMappingAlgorithmOf(String peer_name)

Επιστρέφει τον μηχανισμό παραγωγής αντιστοιχιών για τον κόμβο με όνομα peer_name

- **public** Peer getPeerByName(String peer_name,String choice)

Επιστρέφει το αντικείμενο Peer για τον κόμβο με όνομα peer_name. Με την επιλογή choice καθορίζεται αν ο κόμβος είναι γείτονας ή υποψήφιος

- **public** GenericModel getModelFromCandidate(String peerName)

Επιστρέφει το γενικό μοντέλο σχήματος ενός υποψήφιου γείτονα

- **public** GenericModel getModelFromNeighbor(String peerName)

Επιστρέφει το γενικό μοντέλο σχήματος ενός γείτονα

- **public void** printPeers(String choice)

Τυπώνει στην οθόνη του χρήστη σχετικές πληροφορίες για τους γειτονικούς και υποψήφιους κόμβους.

5.2.10 Η κλάση MappAttr

Η κλάση αυτή μοντελοποιεί ένα Mapping Element μιας αντιστοιχίας, δηλαδή μοντελοποιεί μια γραμμή του δομής Matrix(M). Ο χρήστης χαρακτηρίζει ξεχωριστά την μετάφραση που αντιπροσωπεύει κάθε ένα τέτοιο αντικείμενο, και με τον τρόπο αυτό βελτιώνει τη συνολική αντιστοιχία

A) Αξιοσημείωτα πεδία

- **private** String orattr;

Το πλήρες όνομα της ιδιότητας του πίνακα της αντιστοιχίας, που μεταφράζεται μέσω αυτού του αντικειμένου (Table.Attribute)

- **private** String orAttSimple;

Το απλό όνομα της παραπάνω ιδιότητας (Attribute)

- **private** String orTableName;

Το όνομα του πίνακα της παραπάνω ιδιότητας (Table)

- **private** Mapping mapping;

Η αντιστοιχία, στην οποία ανήκει το συγκεκριμένο mapping element

- **private** String corrattr;

Το πλήρες όνομα της ιδιότητας που αντιστοιχίζεται, μέσω της ισοδυναμίας που επιλέχθηκε η ιδιότητα orattr

- **private** String corrAttSimple;

Το απλό όνομα της ιδιότητας corrattr

- **private** String corrTableName;

Το όνομα του πίνακα της ιδιότητας corrattr

- **private** String tableToJoin;

Το όνομα του πίνακας, προς τον οποίο θα αναζητηθεί ένα μονοπάτι συνένωσης για το συγκεκριμένο mapping element, αν αυτό θεωρηθεί αναγκαίο

- **private float** corrW;

Η πιθανότητα f_{CORR_Ai} του συγκεκριμένου mapping element, που θα χρησιμοποιηθεί για τον προσδιορισμό της εκτιμήτριας συνάρτησης

- **private float** joinPathW;

Η πιθανότητα f_{JP_Ai} του συγκεκριμένου mapping element, που θα χρησιμοποιηθεί για τον προσδιορισμό της εκτιμήτριας συνάρτησης

- **private** Vector<Correspondence> possibleCorrs;

Το σύνολο με τις πιθανές ισοδυναμίες για αυτό το mapping element

- **private** Vector<Integer> badJoins;

Το σύνολο με τα αναγνωριστικά των λανθασμένων μονοπατιών συνένωσης για αυτό το mapping element

- **private** JoinPath joinpath;

Το μονοπάτι συνένωσης που χρησιμοποιεί το αντικείμενο αυτό, στην αντιστοιχία

- **private float** value;

Η βαθμός βεβαιότητας της ισοδυναμίας που χρησιμοποιείται από το mapping element

- **private int** status;

Ο χαρακτηρισμός που έχει προσάψει ο χρήστης στο mapping element, μέσω της ανάδρασης

- **private** String alias;

Το ψευδώνυμο που χρησιμοποιείται για τον πίνακα corrTableName. Αν δε χρησιμοποιείται κανέναν ψευδώνυμο, το πεδίο αυτό παίρνει την ίδια τιμή με την τιμή του πεδίου corrTableName

- **private** Vector<String> badCorrs;

Το σύνολο με τις λανθασμένες ισοδύναμες ιδιότητες που διατηρεί το κάθε mapping element

B) Αξιοσημείωτες μέθοδοι

- **public boolean** isRenamed()

Αληθές, αν χρησιμοποιείται κάποιο ψευδώνυμο

- **public** Vector<String> getBadCorrs()

Επιστρέφει το σύνολο με τις λανθασμένες ισοδύναμες ιδιότητες

- **public** String getAlias()

Επιστρέφει το ψευδώνυμο του mapping element

- **public void** addInBadCorrs(String attr)

Προσθέτει μια λανθασμένη ιδιότητα στο αντίστοιχο σύνολο

- **public boolean** containedInBadCorrs(String attr)

Αληθές, αν η ιδιότητα ανήκει στο σύνολο λανθασμένων ιδιοτήτων

- **public float** getValue()

Επιστρέφει τον βαθμό βεβαιότητας της ισοδυναμίας που επιλέχθηκε

- **public** String getCorrAttr()

Επιστρέφει το πλήρες όνομα της ισοδύναμης ιδιότητας

- **public** String getAliasedAttr()

Επιστρέφει το πλήρες όνομα της ισοδύναμης ιδιότητας, μετονομασμένη, αν χρησιμοποιείται κάποιο ψευδώνυμο

- **public** String getTableToJoin()

Επιστρέφει τον πίνακα, προς τον οποίο θα αναζητηθεί κάποιο μονοπάτι συνένωσης

- **public** String getCorrTable()

Επιστρέφει το όνομα του πίνακα της ισοδύναμης ιδιότητας

- **public** String getCorrAttSimple()

Επιστρέφει το απλό όνομα της ισοδύναμης ιδιότητας

- **public** String getOrigAttr()

Επιστρέφει το πλήρες όνομα της ιδιότητας που αντιστοιχίζεται

- **public** String getOrigSimpleAtt()

Επιστρέφει το απλό όνομα της ιδιότητας που αντιστοιχίζεται

- **public** String getOrigSimpleTable()

Επιστρέφει το όνομα του πίνακα της αντιστοιχίας

- **public** Vector<Correspondence> getPossibleCorrs()

Επιστρέφει το σύνολο με τις πιθανές ισοδυναμίες για το συγκεκριμένο mapping element

- **public int** getStatus()

Επιστρέφει τον χαρακτηρισμό που έχει θέσει ο χρήστης, μέσω της ανάδρασης στο συγκεκριμένο mapping element

- **public** String toString()

Η string αναπαράσταση του αντικειμένου

- **public** JoinPath getJoinPath()

Επιστρέφει το μονοπάτι συνένωσης που χρησιμοποιεί το mapping element (null αν δε χρησιμοποιείται καμία συνένωση)

- **public** Vector<Integer> getBadJoins()

Επιστρέφει το σύνολο με τα αναγνωριστικά των λανθασμένων μονοπατιών συνένωσης

- **public void** removeCorrespondece()

Αφαιρεί την ισοδυναμία που χρησιμοποιείται από το στοιχείο αυτό, αλλάζοντας ταυτόχρονα τις τιμές των εμπλεκόμενων πεδίων (ονόματα αντιστοιχιζόμενων ιδιοτήτων)

5.2.11 Η κλάση Mapping

Αντικείμενα της κλάσης αυτής μοντελοποιούν την δυναμική δομή μιας αντιστοιχίας, όπως αυτή περιγράφηκε στο προηγούμενο κεφάλαιο με τη δομή Matrix(M). Κάθε τέτοιο αντικείμενο έχει τη δυνατότητα να ανανεώνεται από την σχετική ανάδραση του χρήστη, να παράγει το αντίστοιχο SQL ερώτημα που αντιπροσωπεύει αυτή η αντιστοιχία, να αρχικοποιείται από το αντίστοιχο σύνολο των κατευθυνόμενων ισοδυναμιών και το γενικό μοντέλο του target σχήματος.

A) Αξιοσημείωτα πεδία

- **public static final int** *SEARCH_HEURISTIC* = 1;
- **public static final int** *SEARCH_CHAINED* = 2;

Οι δύο διαφορετικοί τύποι για τους αλγόριθμους εύρεσης μονοπατιών συνένωσης

- **public int** search_type;

Ο τύπος αναζήτησης μονοπατιών συνένωσης που χρησιμοποιείται κατά τη δημιουργία και βελτίωση του συγκεκριμένου mapping

- **public static final int** *INIT* = 0;

Με τον χαρακτηρισμό αυτό περιγράφονται τα mapping elements για τα οποία δεν έχει ληφθεί καμία ανάδραση από το χρήστη.

- **public static final int** *GOOD_NOJOIN* = 1;

Κατηγορία Good Correspondence, No Join Path needed

- **public static final int** *GOOD_JOIN* = 2;

Κατηγορία Good Correspondence, Good Join Path

- **public static final int** *BAD_JOIN =3*;

Κατηγορία Good Correspondence, Bad Join Path

- **public static final int** *SELF_JOIN* = 5;

Κατηγορία Good Correspondence, Self Join Path needed

- **public static final int** *NOT_SELF_JOIN* = 6;

Κατηγορία Good Correspondence, No Self Join Path needed

- **public static final int** *BAD* = 7;

Κατηγορία Bad Correspondence

- **public static final int** *GOOD_NEEDJOIN* = 8;

Κατηγορία Good Correspondence, Join Path Needed

- **public static final int** *GAV* = 0;

Ένδειξη για αντιστοιχίες τύπου GAV

- **public static final int** *LAV* = 1;

Ένδειξη για αντιστοιχίες τύπου LAV

- **private int** type;

Ο τύπος της αντιστοιχίας (LAV ή GAV)

- **private** MappingAlgorithm mp;

Αναφορά στον μηχανισμό παραγωγής και βελτίωσης αντιστοιχιών, στον οποίο προσάγεται αυτή η αντιστοιχία

- **private** Hashtable<String, Vector<String>> allTables;

Σε αυτόν τον πίνακα hash αποθηκεύουμε όλους τους πίνακες που συμμετέχουν στην αντιστοιχία, μαζί με το σύνολο των ιδιοτήτων του καθενός

- **private** Hashtable<String, Integer> joins;

Σε αυτόν τον πίνακα Hash αποθηκεύουμε όλους τους συνδέσμους που συμμετέχουν στην αντιστοιχία. Για κάθε σύνδεσμο κρατάμε επίσης τον συνολικό αριθμό εμφανίσεων του στα join paths που χρησιμοποιούνται. Παραδείγματος χάριν αν χρησιμοποιούνται τα εξής δύο μονοπάτια σε μια αντιστοιχία, T1—T2—T3 και T2—T3 τότε ο πίνακας θα έχει την εξής μορφή
T1—T2 → 1
T2—T3 → 2
Διότι ο σύνδεσμος T1—T2 χρησιμοποιείται μόνο στο πρώτο μονοπάτι, ενώ ο δεύτερος χρησιμοποιείται και στα δύο μονοπάτια

- **private** Hashtable<String, JoinPath> hidden_joinPaths;

Στον πίνακα αυτόν καταχωρούμε όλα τα μονοπάτια συνένωσης που υπάρχουν στην αντιστοιχία, μαζί με τον πίνακα που συνενώνεται κάθε φορά με τον κύριο πίνακα της αντιστοιχίας. Στο προηγούμενο παράδειγμα, θα είχαμε:
T1 → T1 – T2 – T3
T2 → T2 – T3

- **private** Hashtable<String, Integer> hidden_joinPaths_Count;

Στον πίνακα αυτό καταχωρούμε τον αριθμό εμφανίσεων κάθε πίνακα στο mapping, ανάλογα με τον αριθμό των μονοπατιών που συνένωσης, στα οποία ο πίνακας συμμετέχει. Έτσι, στο προηγούμενο παράδειγμα, ο πίνακας T1 εμφανίζεται μια φορά, ενώ οι πίνακες T2, T3 εμφανίζονται δύο φορές, οπότε
T1→1
T2→2
T3→2

- **private** TableNode table;

Το tableNode αντικείμενο που μοντελοποιεί τον πίνακα της αντιστοιχίας

- **private** Vector<MappAttr> mapAttrs;

Το σύνολο με όλα τα mapping elements της αντιστοιχίας

- **private** String mainTable;

Το όνομα της κύριας σχέσης της αντιστοιχίας

- **private boolean** mainTbSet;

Αληθές, αν ο κύριος πίνακας έχει καθοριστεί επακριβώς, από την ανάδραση του χρήστη

- **private int** ID;

Το αναγνωριστικό της αντιστοιχίας

B) Αξιοσημείωτες μέθοδοι

- **public boolean** isMappingRight()

Αληθές, αν όλα τα mapping elements της αντιστοιχίας βρίσκονται είτε στην κατηγορία Good Correspondence, No Join Path Needed, είτε στην κατηγορία Good Correspondence,Good Join Path, με άλλα λόγια αληθές, αν η αντιστοιχία, δεν επιδέχεται περαιτέρω βελτίωση

- **public void** updateValues(String mainTable, **double** perc)

Αυξάνει τον βαθμό βεβαιότητας εκείνων των πιθανών ισοδυναμιών που εμπλέκουν ιδιότητες του κύριου πίνακα, κατά ένα ποσοστό perc

- **private void** formatTheCorrespondence(MappAttr map,String correspondence)

Ελέγχει αν μια ισοδυναμία μπορεί να χρησιμοποιηθεί χωρίς ψευδώνυμο από ένα mapping element. Αν όχι, παράγει ένα νέο ψευδώνυμο και το θέτει στο στοιχείο map

- **public void** initializeMappingAttr(MappAttr map)

Αρχικοποιεί το mapping element map, παράγοντας το σύνολο πιθανών ισοδυναμιών, και επιλέγοντας την ισχυρότερη ισοδυναμία. Αν είναι αναγκαίο αναζητείται ένα μονοπάτι συνένωσης και τίθεται ένα ψευδώνυμο

- **public static** String getOriginalTable(String renamedTable)

Επιστρέφει το όνομα του πίνακα που αντιστοιχεί το ψευδώνυμο renamedTable

- **private void** flushJoinPath(MappAttr map)

Αφαιρεί το μονοπάτι συνένωσης που χρησιμοποιείται από ένα mapping element, προσθέτοντας το αναγνωριστικό του μονοπατιού στο αντίστοιχο σύνολο λανθασμένων συνενώσεων του στοιχείου map

- **private** String requestAlias(String tablename)

Βρίσκει ένα νέο έγκυρο ψευδώνυμο για τον πίνακα με όνομα tablename

- **public void** findMainTable()

Βρίσκει την κύρια σχέση της αντιστοιχίας, με τον τρόπο που αναφέρθηκε στην αντίστοιχη παράγραφο

- **public** MappAttr findMappAttr(String attr)

Επιστρέφει το mapping element που μοντελοποιεί την μετάφραση της ιδιότητας attr του πίνακα της αντιστοιχίας

- **public** MappAttr findCorrMappAttr(String attrToFind)

Επιστρέφει το mapping element που μοντελοποιεί τη μετάφραση της ιδιότητας εκείνης του πίνακα της αντιστοιχίας, η οποία έχει αντιστοιχηθεί στην ιδιότητα attrToFind

- **public void** update(Vector<FeedBackElement> el, **int** ch)

Ανανεώνει την αντιστοιχία, με τον τρόπο που περιγράψαμε στο κεφάλαιο «Βελτίωση των αντιστοιχιών» για όλα τα Mapping Elements που περιέχονται στο σύνολο el. Η παράμετρος ch, αντιστοιχεί στον τρόπο που έγινε η μετάφραση του ερωτήματος, πάνω στο οποίο ο χρήστης έδωσε ανάδραση (Successively OR Automatic)

- **private void** removeAttribute(MappAttr map)

Αφαιρεί την αντιστοιχιζόμενη ιδιότητα από ένα mapping element

- **private** JoinPath requestJoinPath4(MappAttr map)

Με την μέθοδο αυτή, αναζητείται ένα έγκυρο μονοπάτι συνένωσης για το στοιχείο map της αντιστοιχίας, από το Join Path Pool του γενικού μοντέλου του target σχήματος. Ο τρόπος έυρεσης του μονοπατιού γίνεται με έναν από τους δύο αλγορίθμους που περιγράφηκαν στην παράγραφο «Επιλογή των μονοπατιών συνένωσης», και καθορίζεται από την τιμή του πεδίου search_type του αντικειμένου.

- **public** JoinPath findJoinPath(JoinPath jp, **int** type)

Επιστρέφει το αντικείμενο JoinPath που είναι καθόλα όμοιο με το join path jp και είναι αποθηκευμένο στο JoinPathPool του target σχήματος.

- **public void** findInitJoinPath(MappAttr map)

Αναζητά ένα μονοπάτι συνένωσης για το στοιχείο map. Αν υπάρχει κάποιο μονοπάτι στον πίνακα hidden Join Paths που είναι έγκυρο για το mapping element, χρησιμοποιείται αυτό, διαφορετικά αναζητείται ένα νέο μονοπάτι με τη μέθοδο requestJoinPath()

- **private void** formatHiddenJoinPath(JoinPath tempjp, String aliased,JoinPath lastToFormat)

Η μέθοδος αυτή, τροποποιεί το μονοπάτι tempjp ώστε στους πίνακες του μονοπατιού να πρσοδωθούν οι κατάλληλες μετονομασίες για να συμμετέχει στο εξής στην αντιστοιχία

- **private void** changeAlias(MappAttr map, String temp)

Αλλάζει την μετονομασία της ισοδύναμης ιδιότητας του στοιχείου map, στο όνομα temp

- **private boolean** canUseAlias(MappAttr map, String temp)

Ελέγχει αν η ισοδύναμη ιδιότητα του στοιχείου map μπορεί να χρησιμοποιήσει το ψευδώνυμο temp

- **public void** print()

Τυπώνει στην οθόνη του χρήστη, περιγραφικές πληροφορίες για την αντιστοιχία

- **public** String toSqlQuery()

Επιστρέφει την String αναπαράσταση της αντιστοιχίας, ως μια όψη – ερώτημα εκφρασμένο σε SQL

- **public double** evaluateMapping()

Επιστρέφει την τιμή της αντιστοιχίας, που προέκυψε από την εκτιμήτρια συνάρτηση

- **public void** writeToFile(String filename)

Γράφει την αντιστοιχία στο αρχείο filename σε μορφή SQL όψης, ώστε να μπορεί να χρησιμοποιηθεί από το μηχανισμό μετάφρασης ερωτημάτων

5.2.12 Η κλάση RelationalToModel

Η κλάση αυτή μεσολαβεί μεταξύ του γενικού μοντέλου σχήματος και της σύνδεσης στη βάση δεδομένων, προκειμένου να αρχικοποιηθεί το γενικό μοντέλο. Η λειτουργία της συνίσταται στην επεξεργασία των μεταδεδομένων μιας βάσης και στην ενσωμάτωση των πληροφοριών αυτών στο γενικό μοντέλο του εκάστοτε σχήματος.

Α) Αξιοσημείωτες μέθοδοι

- **public** GenericModel importRelationalSchema(String databaseName,String uName, String uPassword, String schemaName,String portNumber, String ipAddress, String peer)

Με τη μέθοδο αυτή, γίνεται σύνδεση σε μια σχεσιακή βάση, και παράγεται το αντίστοιχο γενικό μοντέλο που περιγράφει το σχήμα της βάσης αυτής.

- **public** GenericModel importRelationalSchema(String xmlFile)

Η μέθοδος αυτή χρησιμοποιείται και πάλι για αρχικοποίηση ενός γενικού μοντέλου σχήματος, αυτή τη φορά όμως, όχι μέσω απευθείας σύνδεσης στην βάση, αλλά μέσω με την προσπέλαση ενός XML αρχείου που περιγράφει το σχήμα.

Κλάσεις που σχετίζονται με την οντότητα Controller

5.2.13 Η κλάση RewriteMechanism

Η κλάση αυτή περιέχει όλους τους μηχανισμούς που χρησιμοποιεί το σύστημα GrouPeer προκειμένου να μεταφράσει ένα ερώτημα από ένα συγκεκριμένο σχήμα σε κάποιο άλλο, βάσει GAV και LAV αντιστοιχιών. Η μετάφραση γίνεται τόσο μέσω των διαδοχικών αντιστοιχιών των κόμβων, όσο και μέσω των αυτόματα παραγόμενων αντιστοιχιών.

Μια αναλυτική περιγραφή όλων των πεδίων, μεθόδων και αναφορικών κλάσεων αυτής της κλάσης, προσφέρεται στο [15].

5.2.14 Η κλάση MappingAlgorithm

Στην κλάση αυτή είναι υπεύθυνη για την αρχικοποίηση, διαχείριση, και παραγωγή των GAV/LAV αντιστοιχιών μεταξύ δύο σχημάτων. Επικοινωνεί εξωτερικά (μέσω αρχείου) με τις ισοδυναμίες που παρήγαγε ο Matcher, Coma++, προκειμένου να δημιουργήσει τα απαραίτητα σύνολο κατευθυνόμενων ισοδυναμιών.

A) Αξιοσημείωτα πεδία

- **private** GenericModel peermodel;

Το γενικό μοντέλο σχήματος του απομακρυσμένου κόμβου, με τον οποίο διατηρούνται οι αντιστοιχίες που θα παράγει – βελτιώσει ο μηχανισμός

- **private** GenericModel sourceModel;

Το γενικό μοντέλο του τοπικού κόμβου

- **private** Vector<Correspondence> usingCorrs;

Το σύνολο των ισοδυναμιών που έχουν βρεθεί μέχρι τώρα μεταξύ των δύο σχημάτων sourceModel και peerModel

- **private** Vector<Correspondence> newCorrs;

Το σύνολο των ισοδυναμιών που προέκυψαν από το τελευταίο εισερχόμενο ερώτημα του χρήστη

- **private** Vector<Mapping> LAVMappings;

Το σύνολο των LAV αντιστοιχιών μεταξύ των δύο σχημάτων

- **private** Vector<Mapping> GAVMappings;

Το σύνολο των GAV αντιστοιχιών μεταξύ των δύο σχημάτων

B) Αξιοσημείωτες μέθοδοι

- **public int** generateNextID()

Δημιουργεί ένα νέο μοναδικό αναγνωριστικό για μια νέα ισοδυναμία

- **public void** updateUsingCorrs(Vector<Correspondence> comaCorr)

Ενισχύει το σύνολο των μέχρι τώρα ανακαλυφθέντων ισοδυναμιών με νέες ισοδυναμίες που προέκυψαν από το νέο ερώτημα του χρήστη.

- **private boolean** isNew(Correspondence c)

Αληθές, αν η ισοδυναμία, ανακαλύφθηκε με το τελευταίο ερώτημα του χρήστη

- **public** Vector<Correspondence> getCorrsFromComa(String filename)

Επιστρέφει το σύνολο των ισοδυναμιών μεταξύ των δύο σχημάτων που ανακαλύφθηκαν από τον Matcher Coma++, διαβάζοντας το αρχείο εξόδου του Matcher, με όνομα filename

- **private boolean** containedInCorrs(Correspondence corr)

Ελέγχει αν μια ισοδυναμία υπάρχει ήδη στο σύνολο using Corrs

- **public void** printUsingCorrs()

Τυπώνει στην οθόνη του χρήστη, τις μέχρι τώρα ανακαλυφθέντες αντιστοιχίες

- **public** Vector<Correspondence> generateCoors(String attname, **int** type)

Επιστρέφει όλες τις πιθανές ισοδυναμίες για μια ιδιότητα attname, που υπάρχουν στο σύνολο using Corrs.

- **public void** updateMappingsFromFeedBack(**int** type,Vector<FeedBackElement> feeds)

Ανανεώνει τις αντιστοιχίες τύπου type από το σύνολο στοιχείων ανάδρασης feeds

- **public void** updateMappingsFromNewQuery(**int** type, **int** search)

Ανανεώνει τις αντιστοιχίες από ένα νέο εισερχόμενο ερώτημα του απομακρυσμένουν χρήστη

- **public** Mapping getMappingOfTable(TableNode table, **int** type)

Επιστρέφει την αντιστοιχία του πίνακα table, τύπου type

- **public void** updatePossibleCorrs(Mapping m)

Προθέτει στα mapping Elements της αντιστοιχίας m, νέες πιθανές ισοδυναμίες που ανακαλύφθηκαν

- **private** Correspondence findCorr(String corred, String mapped, **int** type)

Βρίσκει από το σύνολο using corrs την ισοδυναμία ιδιοτήτων coored←→mapped.

- **public** Correspondence findCorr(**int** id)

Βρίσκει από το σύνολο using corrs την ισοδυναμία με αναγνωριστικό id.

5.2.15 Η κλάση MainControl

Η κλάση αυτή ενσωματώνει όλες τις επιμέρους λειτουργίες του συστήματος σε μια λογική ροή, προκειμένου το σύστημα να συμπεριφέρεται σύμφωνα με τις προδιαγραφές του, όταν ένα εισερχόμενο ερώτημα καταφθάνει. Ουσιαστικά η κλάση αυτή είναι το βασικότερο σημείο της οντότητας Controller, μιας και συνδυάζει όλα τα χαρακτηριστικά του συστήματος σε γενικευμένες μεθόδους.

A) Αξιοσημείωτα πεδία

- **private static final int** *TTL*;

Το σταθερό αυτό πεδίο καθορίζει το μήκος του μονοπατιού που θα διαδίδεται το ερώτημα στο δίκτυο (time to live)

- **private** MainView mainView;

Η αναφορά στο κεντρικό παράθυρο της διαπροσωπείας του χρήστη (user interface)

- **private** MyPeer myPeer;

Η αναφορά στο αντικείμενο που περιγράφει τον τοπικό κόμβο

- **private** P2pLayerManager P2PManager;

Ο διαχειριστής του p2p layer του συστήματος. Μέσω αυτού, στέλνονται και λαμβάνονται ερωτήματα-δεδομένα από κόμβο σε κόμβο

- **private** DatabaseManager dbManager;

Ο διαχειριστής της τοπικής σχεσιακής βάσης κάθε κόμβου

- **private** RewriteMechanism rw;

Ο μηχανισμός μετάφρασης ερωτημάτων, βάσει GAV/LAV αντιστοιχιών

B) Αξιοσημείωτες μέθοδοι

- **public void** processMessage(Message message)

- **public boolean** isPeerCandidate(String peername)

Αληθές, αν ο κόμβος με το όνομα peername είναι υποψήφιος γείτονας του τοπικού κόμβου

- **public boolean** isPeerNeighbor(String peername)

Αληθές, αν ο κόμβος με όνομα peername είναι γείτονας του τοπικού κόμβου

- **public** Query produceOriginalQuery(String filename)

Δημιουργεί ένα αντικείμενο query από το ερώτημα που βρίσκεται στο αρχείο filename

- **public** void propagateQuery(Query main)

Στέλνει ένα ερώτημα main στο δίκτυο προς απάντηση

- **public** Vector<Log> processIncomingQuery(GrouPeerAdvertisement pipeAdv,Query initial,Vector<Log> incomingLogs)

Επεξεργάζεται ένα εισερχόμενο ζεύγος ερωτημάτων από κάποιον κόμβο: Το ζεύγος αποτελείται, σύμφωνα με τη λογική του συστήματος GrouPeer, τόσο από το αρχικό ερώτημα (initial), όσο και από το successively rewritten ερώτημα (incomingLogs)

- **private void** forwardAnswers(Query initial, Vector<Log> rewrittenLogs, GrouPeerAdvertisement myPipeAdv)

Προωθεί την μετάφραση του ερωτήματος initial, στον κόμβο που αρχικά έστειλε το ερώτημα

- **public void** requestFeedBack(Vector<Log> logs,GrouPeerAdvertisement pipeAdv)

Ζητά από τον χρήστη να προσφέρει ανάδραση στην μετάφραση που έγινε.

- **public void** processIncomingFeedBack(Vector<FeedBackElement> felGAV, Vector<FeedBackElement> felLAV,String owner,**int** typeOfAnswer)

Επεξεργάζεται τα εισερχόμενα στοιχεία ανάδρασης από έναν κόμβο, ώστε να βελτιωθούν οι αντιστοιχίες που ο τοπικός κόμβος έχει θεσπίσει με αυτόν

- **private void** forwardFeedback(Vector<FeedBackElement> felGAV, Vector<FeedBackElement> felLAV,String owner,**int** typeOfAnswer,GrouPeerAdvertisement pipeAdv)

Στέλνει πίσω στον κόμβο που εκτέλεσε την μετάφραση τα στοιχεία ανάδρασης του χρήστη

- **private void** forwardLogs(GrouPeerAdvertisement mypipeAdv,Query initialQuery,Vector<Log> logsToFwd, Peer nextpeer)

Στέλνει το ζεύγος initial & successively rewritten Query στους γείτονες του τοπικού κόμβου, ώστε να το απαντήσουν και αυτοί και το ερώτημα να διαδοθεί στο δίκτυο

- **public boolean** removePeerNeighbor(String peername)

Αφαιρεί έναν κόμβο από το σύνολο των γειτόνων

5.3 Κωδικοποίηση αρχείων

Το σύστημα χρησιμοποιεί τρεις διαφορετικές μορφές αρχείων, κατά τη λειτουργία του. Αυτές είναι οι εξής:

Α) Το αρχείο, στο οποίο εξάγει τις ισοδυναμίες ιδιοτήτων το πρόγραμμα Coma++

Πρόκειται για ένα αρχείο κειμένου (text file) της παρακάτω μορφής

Generation parameters: (Parameters)

-1[ss1.st1.sa1->st1.tt1.ta1:0.64844435]

-2[ss2.st2.sa2->st2.tt2.ta2:0.9714286]

-3[ss3.st3.sa3->st3.tt3.ta3:0.60764706]

...

-N[ssn.stn.san->stn.ttn.tan:1.0]

Στην πρώτη γραμμή του αρχείου αναφέρονται πάντα οι παράμετροι του Matcher Coma++, κάτω από τις οποίες έγινε η παραγωγή των ισοδυναμιών

Στις επόμενες γραμμές αναφέρονται οι ισοδυναμίες που προέκυψαν από το πρόγραμμα στην εξής μορφή:

-(ID)[sourceAttribute->targetAttribute:(degree of certainty)]

B) Το configuration αρχείο για την αρχικοποίηση ενός κόμβου

Πρόκειται για ένα XML αρχείο, με την εξής απλή κωδικοποίηση:

```
<?xml version="1.0"?>
<peer>
   <name>«όνομα κόμβου»</name>
   <dbusername>>«username για σύνδεση στην  βάση του κόμβου»</dbusername>
   <dbpass>>«password για σύνδεση στην  βάση του κόμβου »</dbpass>
   <dbname>>«το όνομα της βάσης του κόμβου»</dbname>
   <dbport>>«το port number για σύνδεση στη βάση»</dbport>
   <dbIP>>«η IP διεύθυνση του κόμβου»</dbIP>
   <schemaName>>«το όνομα του σχήματος του κόμβου»</schemaName>
</peer>
```

C) Τα αρχεία που περιέχουν την XML αναπαράσταση των γενικών μοντέλων σχήματος για τους γειτονικούς και υποψήφιους κόμβους.

Και πάλι πρόκειται για ένα XML αρχείο με την παρακάτω όμως δομή:

```
<model>
        <peername>PeerA</peername>
        <type>«ο τύπος του σχήματος (σχεσιακό, XML,κλπ)»</type>
        <tables>
                <table name="«Το όνομα του πίνακα»">
                        <attribute>«Το όνομα της ιδιότητας»</attribute>
                                        …
                        <attribute>>«Το όνομα της ιδιότητας»</attribute>
                </table>
                        …
                <table name="«Το όνομα του πίνακα»">
                        <attribute>«Το όνομα της ιδιότητας»</attribute>
                                        …
                        <attribute>>«Το όνομα της ιδιότητας»</attribute>
                </table>
        </tables>
        <keys>
                <key>«Το όνομα του κύριου κλειδιού»</key>
                                …
                <key>«Το όνομα του κύριου κλειδιού»</key>
        </keys>
        <fkeys>
                <fkey>
                        <attrkey>«Το όνομα του εξωτερικού κλειδιού»</attrkey>
                        <reference>«Η αναφορά του εξωτερικού κλειδιού»</reference>
                </fkey>
                        …
                <fkey>
                        <attrkey>«Το όνομα του εξωτερικού κλειδιού»</attrkey>
                        <reference>«Η αναφορά του εξωτερικού κλειδιού»</reference>
                </fkey>
        </fkeys>
</model>
```

Διατρέχοντας τους κόμβους των παραπάνω xml αρχείων, ο κάθε τοπικός κόμβος, μπορεί να αρχικοποιήσει τα αντίστοιχα πεδία στα αντικείμενα που χρειάζεται.

5.4 Πλατφόρμες και προγραμματιστικά εργαλεία

Η εργασία υλοποιήθηκε στην γλώσσα προγραμματισμού JAVA, χρησιμοποιώντας το περιβάλλον Eclipse (έκδοση 3.3.2). Το μηχάνημα που έτρεξαν τα πειράματα και ελέγχθηκε η σωστή λειτουργία του συστήματος είχε επεξεργαστεί Intel(R) Core(TM)2 CPU T5500 @1.66 GHz και μνήμη 1Gb, με λειτουργικό σύστημα Microsoft Windows XP Home Edition.

6 Έλεγχος

Στο κεφάλαιο αυτό θα παρουσιάσουμε τον έλεγχο του συστήματος. Δεδομένου ότι οι έγκυρες σημασιολογικά αντιστοιχίες εξαρτώνται από μια πληθώρα παραμέτρων, οι οποίες αλλάζουν ανάλογα με τα σχήματα που θέλουμε να αντιστοιχήσουμε, αλλά και με την αξιοπιστία των ισοδυναμιών που δημιουργεί ο Matcher, Coma++ ανάμεσα σε αυτά τα σχήματα, ο έλεγχος δεν πραγματοποιήθηκε με τη χρήση ενός σεναρίου αποκλειστικά. Κάτι τέτοιο θα παρουσίαζε τα χαρακτηριστικά του μηχανισμού μόνο για συγκεκριμένα σχήματα, ενώ κατά κύριο λόγο ενδιαφερόμαστε να τονίσουμε την αποδοτικότητα του συστήματος σε όλες τις πιθανές συνθήκες που ενδέχεται να προκύψουν. Έτσι λοιπόν, κατασκευάστηκε μια πλήρης πειραματική σουίτα, που δημιουργεί κάθε φορά διαφορετικές συνθήκες, ανάλογα με τις επιλογές του χρήστη. Στις παραγράφους που ακολουθούν, εξηγούμε αναλυτικά τις παραμέτρους που μπορούμε να αλλάζουμε κατά την πειραματική αξιολόγηση του συστήματος, τους λόγους που επιλέξαμε να τροποποιούμε αυτές τις παραμέτρους, αλλά και τα αποτελέσματα των χαρακτηριστικότερων πειραμάτων που έγιναν.

6.1 Μεθοδολογία ελέγχου

Σε γενικές γραμμές, ο έλεγχος του συστήματος βασίστηκε στην κατασκευή και σταδιακή βελτίωση μιας αντιστοιχίας M ενός πίνακα T προς ένα σχήμα S_T(target schema), μιας και η λειτουργία του μηχανισμού δεν επηρεάζεται ούτε από την κατεύθυνση των αντιστοιχιών (GAV ή LAV), ούτε από το πλήθος των αντιστοιχιών που κατασκευάζονται μεταξύ των δύο σχημάτων. Η αποδοτικότητα συστήματος βασίστηκε αποκλειστικά στον αριθμό των βημάτων ανάδρασης που απαιτούνται, ώστε το σύστημα να ανακαλύψει την αντιστοιχία M.

Κάτω από αυτό το πρίσμα, μπορούμε να διαχωρίσουμε τις μεταβλητές παραμέτρους της πειραματικής αξιολόγησης σε τρεις βασικές κατηγορίες:

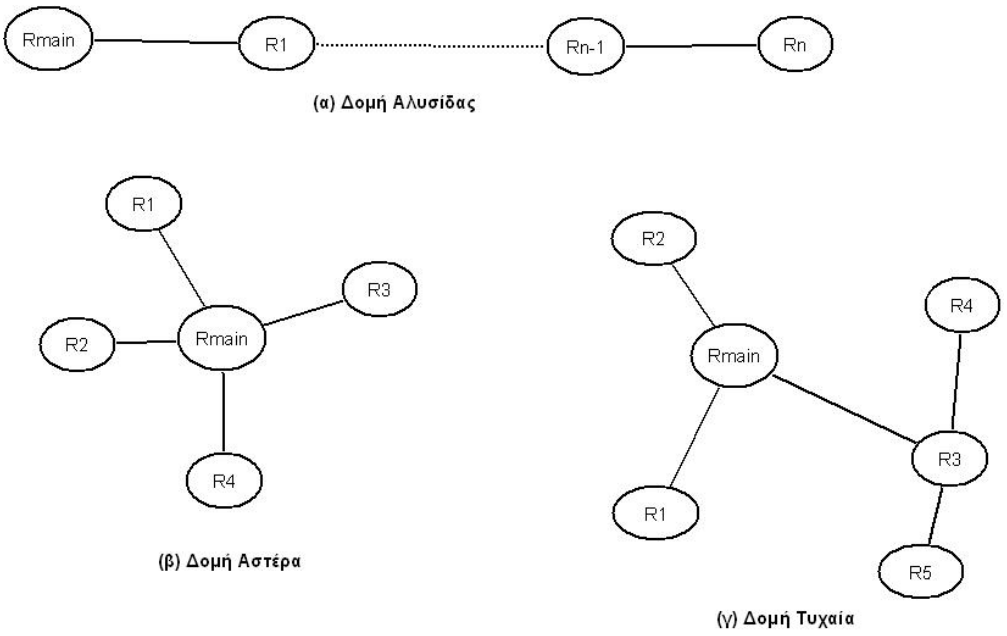
Η πρώτη κατηγορία αφορά παραμέτρους που τροποποιούν την πολυπλοκότητα του target σχήματος S_T. Όσο πιο πολύπλοκο είναι το target schema, τόσο περισσότερες εναλλακτικές λύσεις είναι διαθέσιμες στον μηχανισμό, και για το λόγο αυτό, τόσο δυσκολότερη είναι η εύρεση της σωστής αντιστοιχίας M. Έτσι λοιπόν, μπορούμε να τροποποιούμε κάθε φορά τον αριθμό των πινάκων που απαρτίζουν το σχήμα, τον αριθμό των ιδιοτήτων κάθε σχέσης, και κυρίως των αριθμό των περιορισμών εξωτερικών κλειδιών που υπάρχουν στο σχήμα. Ας μην ξεχνάμε, πως ο αριθμός των πιθανών συνενώσεων μεταξύ δύο πινάκων μεγαλώνει εκθετικά με την αύξηση των συνδέσμων που υπάρχουν ανάμεσα σε δύο σχέσεις του μονοπατιού συνένωσης. Στο target σχήμα επιλέγουμε μια σχέση, η οποία θα αποτελεί την κύρια σχέση της αντιστοιχίας M (R_{MAIN}) και βάσει αυτής καθορίζουμε το ποσοστό των συνδέσμων του σχήματος που θα εμπλέκουν την κύρια σχέση. Οι υπόλοιποι σύνδεσμοι θα αναφέρονται σε δύο πίνακες διαφορετικούς από την R_{MAIN}.

Η δεύτερη κατηγορία αφορά την μορφή της αντιστοιχίας M, την οποία θέλουμε το σύστημα να προσδιορίσει αυτόματα. Στην ανάλυση μας, μελετούμε κυρίως τρεις διαφορετικούς τύπους αντιστοιχιών που φαίνονται στο σχήμα 6 a. Η κατηγοριοποίηση γίνεται βάσει του τρόπου κατανομής και διασύνδεσης των σχέσεων του target schema που συμμετέχουν στην αντιστοιχία, με άλλα λόγια βάσει της μορφής του γράφου της αντιστοιχίας, G(M). Έτσι έχουμε αντιστοιχίες δομής αστέρα, όπου οι συμμετέχουσες σχέσεις συνδέονται με την κύρια σχέση με ξεχωριστά μονοπάτια συνένωσης μικρού μήκους (συνήθως μοναδιαίου), αντιστοιχίες δομής αλυσίδας, όπου οι συμμετέχουσες σχέσεις δημιουργούν μια αλυσίδα κόμβων στο γράφο, και αντιστοιχίες τυχαίας δομής, που αποτελούν έναν συνδυασμό των δύο προηγούμενων κατηγοριών. Σε κάθε δομή αντιστοιχίας, μπορούμε να καθορίσουμε το ποσοστό των ιδιοτήτων που θα περιλαμβάνονται στην κύρια σχέση της αντιστοιχίας, εισάγοντας έτσι περισσότερους ή λιγότερους δευτερεύοντες πίνακες, που θα απαιτήσουν μονοπάτια συνένωσης. Ειδικότερα στη δομή αλυσίδας, μπορούμε να καθορίσουμε το μήκος της, ενώ στη δομή αστέρα το μήκος των μονοπατιών συνένωσης των δευτερευόντων πινάκων με την κύρια σχέση της αντιστοιχίας (στο σχήμα 6 α επιλέχθηκε μοναδιαίο μήκος). Για την τυχαία δομή αντιστοιχίας, καθορίζουμε πόσοι από τους δευτερεύοντες πίνακες θα συνδέονται στον κύριο πίνακα με τη λογική του αστέρα, και πόσοι με τη λογική της αλυσιδωτής δομής.

Τέλος, αφού έχουμε παραμετροποιήσει επακριβώς την πολυπλοκότητα του target σχήματος, και την αντιστοιχία M, την οποία αναζητούμε, απομένει να καθορίσουμε την αξιοπιστία του προγράμματος Coma++, το οποίο δημιουργεί τις ισοδυναμίες μεταξύ των ιδιοτήτων των δύο σχημάτων. Δεδομένου ότι τα αποτελέσματα ενός Matcher, έχουν άρρηκτη σχέση με τα σχήματα που συνδέουν, η πραγματική απόδοση του Coma++, είναι αρκετά δύσκολο να προσομοιωθεί ρεαλιστικά. Ωστόσο μπορούμε να ορίσουμε ένα αποδεκτό μοντέλο

προσομοίωσης, καθορίζοντας τις εξής παραμέτρους. Πρώτον καθορίζουμε το ποσοστό των ισοδυναμιών κάθε αντιστοιχίας που θα ανακαλυφθούν επιτυχώς από το Coma++. Έτσι, για ποσοστό 100%, όλες οι ιδιότητες του πίνακα της αντιστοιχίας, θα αντιστοιχίζονται στην σωστή ιδιότητα του σχήματος S_T. Μια δεύτερη παράμετρος αποτελεί το πλήθος των πιθανών ισοδυναμιών που θα είναι διαθέσιμες για κάθε ιδιότητα, ενώ τέλος καθορίζουμε το βάρος των λανθασμένων ισοδυναμιών: μικρό βάρος ισοδυναμεί με γρήγορη εύρεση της σωστής ισοδυναμίας από το σύστημα, ενώ μεγάλο βάρος σημαίνει ότι το σύστημα θα επιλέξει διάφορες άλλες λανθασμένες ισοδυναμίες από το σύνολο $C_{POSSIBLE}$, προτού χρησιμοποιήσει την σωστή στην αντιστοιχία.

Παρατηρούμε λοιπόν, ότι η ανάλυση του αλγορίθμου βασίστηκε σε συνθετικά δεδομένα, τα οποία μπορούν να διαφοροποιούνται μέσω πολλών παραμέτρων. Δεδομένου ότι οι συνδυασμοί που μπορούν να ελεγχθούν, επιτρέπουν έναν πολύ μεγάλο αριθμό πειραμάτων, στην επόμενη παράγραφο θα παρουσιάσουμε μόνο τα πιο ενδιαφέροντα αποτελέσματα της ανάλυσης με μορφή διαγραμμάτων.



Σχήμα 6 a: Διάφορες πιθανές δομές αντιστοιχιών

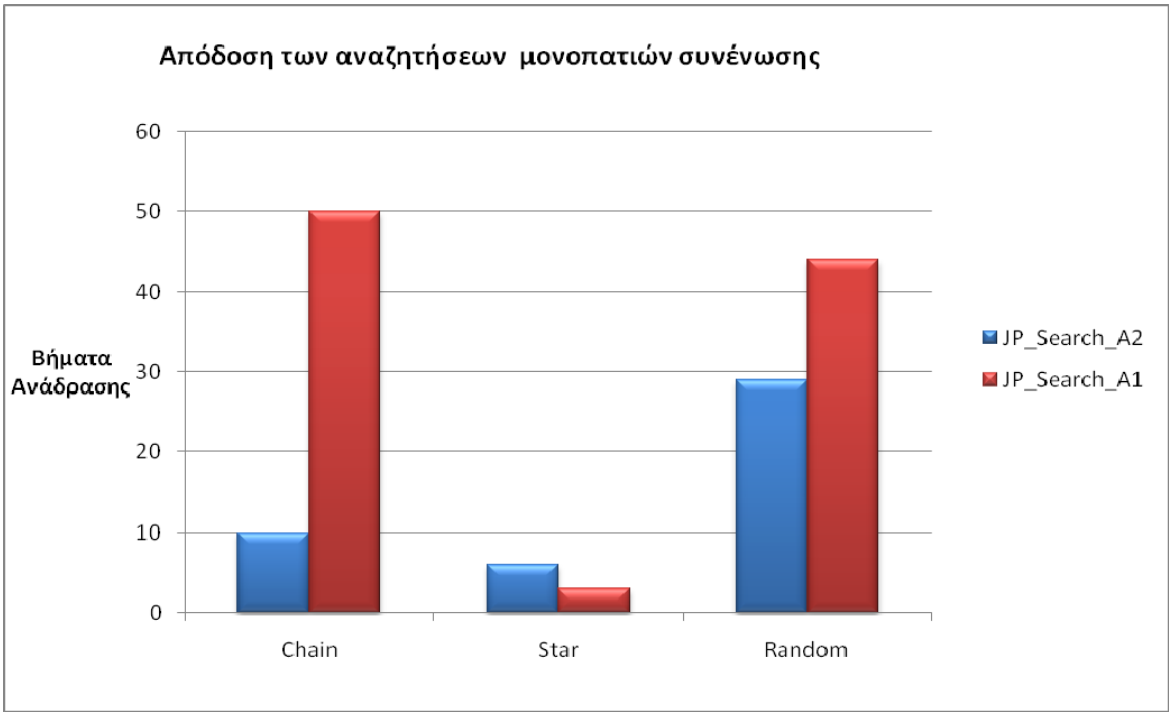
6.2 Αναλυτική παρουσίαση ελέγχου

Στο σχήμα 6 b παρουσιάζεται η απόδοση του μηχανισμού στις τρεις διαφορετικές δομές αντιστοιχιών. Η αναζήτηση των μονοπατιών συνένωσης σε κάθε περίπτωση έγινε δίνοντας προτεραιότητα τόσο στο ευριστικό κριτήριο A1 (JP_Search_A1) όσο και στο κριτήριο A2 (JP_search_A2), που παρουσιάστηκαν στην παράγραφο 4.4.1. Υπενθυμίζουμε τα δύο αυτά κριτήρια:

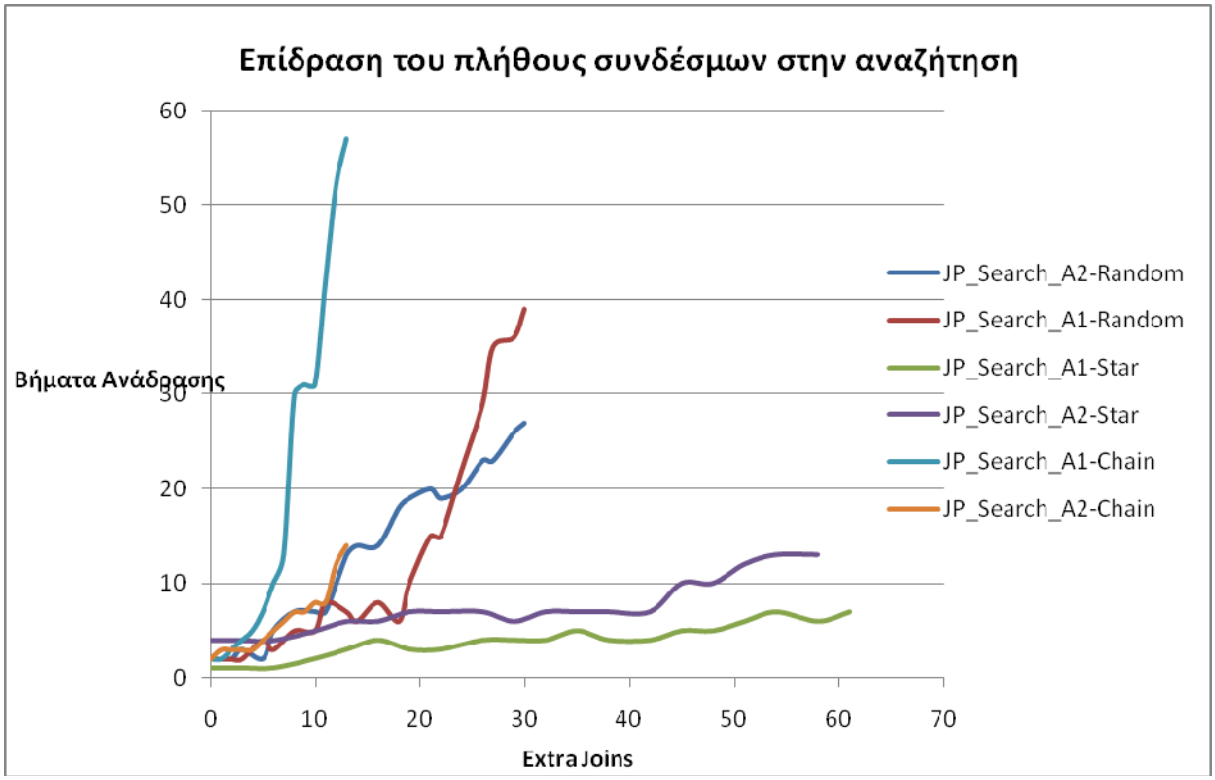
- **A1**:Τα μονοπάτια συνένωσης των πινάκων να έχουν το μικρότερο δυνατό μήκος
- **A2**:Η σύνδεση των κόμβων του συνόλου V1 να γίνεται με τέτοιο ώστε οι κόμβοι του συνόλου V2 να είναι οι ελάχιστοι δυνατοί.

Οι μετρήσεις του σχήματος 6 b έγιναν πάνω σε μεγάλες και πολύπλοκες αντιστοιχίες, στις οποίες οι σταθερές παράμετροι έχουν ως εξής: ο πίνακας της αντιστοιχίας έχει 37 με 38 ιδιότητες, και το 10-20% των ιδιοτήτων αυτών αντιστοιχίζεται σε ιδιότητες του κύριου πίνακα (δε χρειάζεται μονοπάτι συνένωσης για τις ιδιότητες αυτές). Ο αριθμός των δευτερευόντων πινάκων είναι 6 με 10, από τους οποίους περίπου οι μισοί ανήκουν στο σύνολο V2, δηλαδή δεν εμφανίζονται στις ισοδυναμίες της αντιστοιχίας. Μεταξύ των πινάκων της αντιστοιχίας υπάρχουν 8 – 22 περιορισμοί εξωτερικών κλειδιών.

Όπως παρατηρεί κανείς, στις αλυσιδωτές αντιστοιχίες, ο αλγόριθμος που δίνει προτεραιότητα στο κριτήριο A2 είναι πολύ πιο αποδοτικός από τον αλγόριθμο JP_Search_A1. Το αποτέλεσμα αυτό άλλωστε ήταν αναμενόμενο, μιας και γενικά οι αλυσιδωτές αντιστοιχίες περιλαμβάνουν μονοπάτια μεγάλου μήκους προς την κύρια σχέση R_{MAIN}. Έτσι, αναζητώντας συνενώσεις μικρού μήκους, ο αλγόριθμος JP_Search_A1, θα επιλέξει πολλά μικρά μονοπάτια μεταξύ των πινάκων που πρέπει να συνενωθούν, προτού φτάσει στο μήκος που ζητείται, ενώ ο αλγόριθμος JP_Search_A2 εκμεταλλεύεται τα ήδη ανακαλυφθέντα μονοπάτια και ψάχνει για μεγαλύτερα μήκη, αποφεύγοντας έτσι την άσκοπη αναζήτηση μικρών συνενώσεων. Στις αντιστοιχίες τύπου αστέρα, η αναζήτηση JP_Search_A1 φαίνεται να είναι καλύτερη, αλλά και πάλι η απόδοση του αλγορίθμου JP_Search_A2 είναι αρκετά ικανοποιητική. Τέλος, στις τυχαίες αντιστοιχίες η μέθοδος JP_Search_A1 είναι μεν υποδεέστερη από την JP_Search_A2, αλλά με υπολογίσιμη απόδοση.

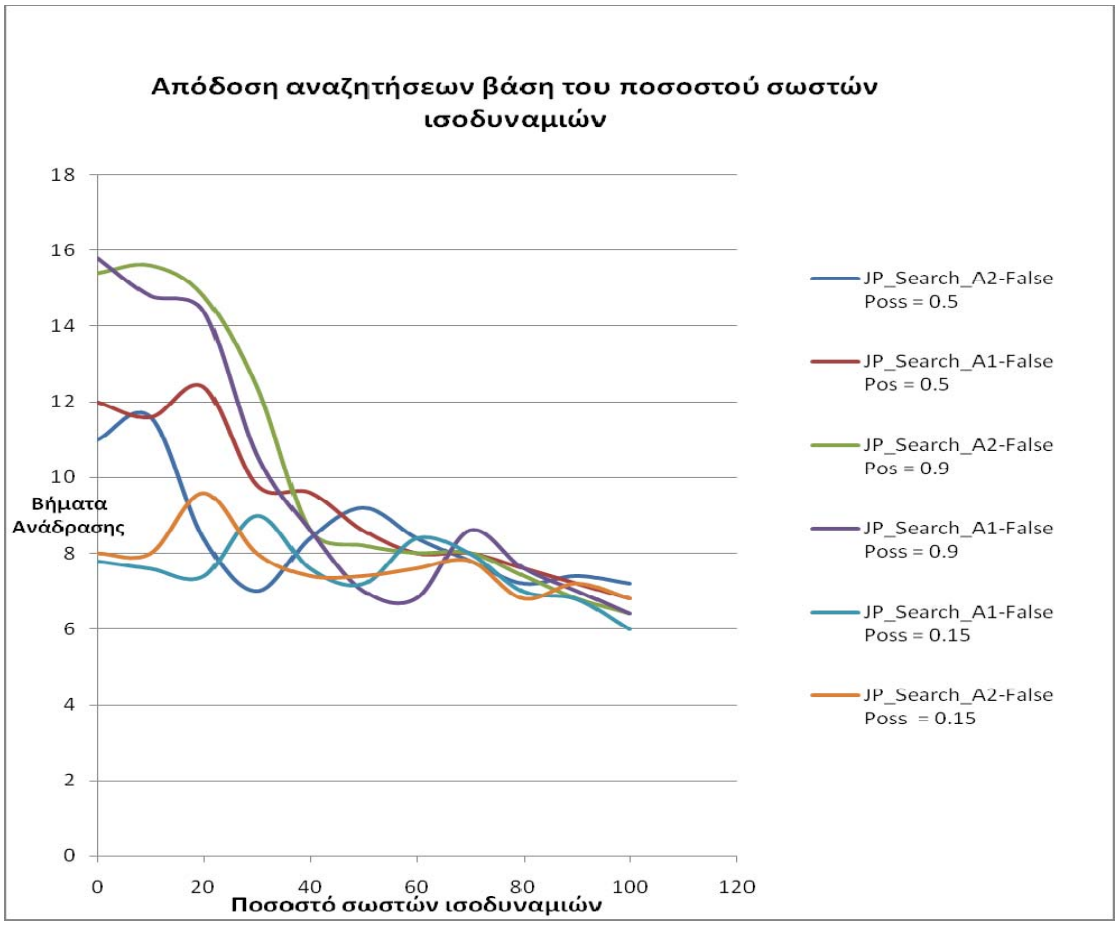


Σχήμα 6 b: Η απόδοση του μηχανισμού σε διαφορετικές δομές αντιστοιχιών



Σχήμα 6 c: Επίδραση του πλήθους συνδέσμων στην αναζήτηση συνενώσεων

Στο σχήμα 6 c παρουσιάζονται η αποδόσεις των δύο διαφορετικών μεθόδων αναζήτησης σε κάθε τύπο αντιστοιχίας, καθώς το σύνολο των συνδέσμων που εισάγουμε ανάμεσα στις σχέσεις του target schema αυξάνει. Το πολύ ενδιαφέρον στοιχείο του διαγράμματος, είναι, αφενός η σε γενικές γραμμές σταθερή απόδοση και των δύο μεθόδων σε αντιστοιχίες αστέρα (μιας και η αναζήτηση ξεκινά από μικρά μονοπάτια και δεν χάνεται σε συνενώσεις μεγάλου μήκος), αφετέρου δε, το πολύ μεγάλο συγκριτικό πλεονέκτημα της μεθόδου JP_Search_A2 έναντι της άλλης, σε αντιστοιχίες αλυσίδας. Δεδομένου ότι στις αλυσιδωτές αντιστοιχίες, εμπλέκονται μεγάλα μονοπάτια, των οποίων το πλήθος αυξάνει εκθετικά με την αύξηση των συνδέσμων, η αναζήτηση με βάση το κριτήριο A2 κατορθώνει ακόμα και σε σχήματα 15-20 συνδέσμων, να βρει τη σωστή σημασιολογία σε μόλις 12-13 βήματα, τη στιγμή που η JP_Search_A1 απαιτεί τόσο μεγάλο αριθμό βημάτων, που κρίνεται εξαιρετικά χρονοβόρα για τον χρήστη που προσφέρει την ανάδραση.



Σχήμα 6 d: Απόδοση αναζητήσεων βάση του ποσοστού σωστών ισοδυναμιών

Το τελευταίο πείραμα που παρουσιάζεται στο διάγραμμα του σχήματος 6 d δείχνει την απόδοση των μεθόδων αναζήτησης, καθώς το ποσοστό των σωστά ανακαλυφθέντων ισοδυναμιών αλλάζει για αντιστοιχίες τυχαίας δομής. Η μετρήσεις έγιναν για έναν σχήμα με 8 – 10 συνδέσμους μεταξύ πινάκων ενώ ο σταθερός όρος False Poss που παίρνει τιμές 0.15, 0.5 και 0.9 δηλώνει το βάρος των λανθασμένων ισοδυναμιών. Προφανώς, οι λανθασμένες ισοδυναμίες δυσχεραίνουν την εύρεση των αντιστοιχιών, μιας και έτσι απαιτείται ένα αρχικό στάδιο πριν την αναζήτηση των συνενώσεων, όπου ο χρήστης προσφέρει ανάδραση, βοηθώντας το σύστημα να ανακαλύψει τις σωστές ισοδυναμίες. Η απότομη αλλαγή της απόδοσης που παρατηρείται σε ποσοστό περίπου 35-40% οφείλεται στην εύρεση του κύριου πίνακα της αντιστοιχίας.

Συμπερασματικά, λοιπόν , βλέπουμε ότι ο μηχανισμός αυτόματης εύρεσης αντιστοιχιών που αναπτύχθηκε συμπεριφέρεται αρκετά ικανοποιητικά, ακόμη και σε συνθήκες υψηλής πολυπλοκότητας, ώστε να θεωρείται μια αποδοτική λύση στην αυτόματη κατασκευή αντιστοιχιών, πράγμα που αποτελεί και τον τελικό σκοπό αυτής της διπλωματικής εργασίας.

7 Επίλογος

7.1 Σύνοψη και συμπεράσματα

Συνοψίζοντας, στην παρούσα διπλωματική εργασία παρουσιάσθηκε η σχεδίαση και υλοποίηση ενός συστήματος, που προτείνει μια λύση για την ημιαυτόματη εύρεση GAV/LAV αντιστοιχιών μεταξύ δύο σχημάτων, που αναφέρονται στον ίδιο θεματικό χώρο δεδομένων. Η λύση αυτή βασίζεται στις απλές ισοδυναμίες που παράγει ένα ευρέως χρησιμοποιούμενο Automatic Schema Matching Tool (Coma++), στην ανάδραση του χρήστη που κρίνει τμηματικά τις αντιστοιχίσεις που έγιναν, και σε μια μέθοδο ευριστικής αναζήτησης για την ανακάλυψη των απαραίτητων περιορισμών συνδέσμων. Είδαμε, ότι οι ισοδυναμίες που παράγει το Coma++, δε χρησιμοποιούνται κατευθείαν από τον μηχανισμό του συστήματος, αλλά υφίστανται μια προεπεξεργασία που ενισχύει το σύνολο διαθέσιμων ισοδυναμιών, με επιπλέον στοιχεία ισχυρής σημασιολογικής σημασίας. Καθορίσαμε δύο ευριστικά κριτήρια για την αναζήτηση των απαραίτητων συνενώσεων μιας αντιστοιχίας, των οποίων την αποτελεσματικότητα ελέγξαμε πειραματικά σε διάφορες πιθανές μορφές αντιστοιχιών, και τέλος κατασκευάσαμε μια διαδικασία επικοινωνίας του συστήματος με το χρήστη, που επιτρέπει την ανανέωση και σταδιακή βελτίωση των αντιστοιχιών.

Το σύστημα που αναπτύχθηκε δύναται πλέον να ενσωματωθεί στο σύστημα GrouPeer, ώστε κόμβοι με παραπλήσια σημασιολογικά σχήματα, να είναι σε θέση να χτίσουν ισχυρές αντιστοιχίες μεταξύ τους και να γειτονεύσουν. Από την άλλη μεριά όμως, ο μηχανισμός μπορεί να αποτελέσει και τον πυρήνα ενός ανεξάρτητου εργαλείου για την ανάπτυξη αντιστοιχιών μεταξύ δύο γνωστών σχημάτων.

7.2 Μελλοντικές επεκτάσεις

Η παρούσα διπλωματική εργασία αποτελεί μια πρώτη προσπάθεια προσέγγισης του προβλήματος Schema Mapping, ένα από τα σημαντικότερα ανοιχτά προβλήματα στον ερευνητικό χώρο της ετερογένειας δεδομένων. Ο τρόπος που γίνεται αυτή η προσέγγιση, εστιάζει κυρίως στην προσθήκη των απαραίτητων περιορισμών συνδέσμων, προκειμένου η προκύπτουσα αντιστοιχία να είναι σημασιολογικά σωστή. Υπάρχουν περιπτώσεις όμως, που η εύρεση των απαραίτητων συνενώσεων δεν αρκεί για τη δημιουργία μιας πλήρους αντιστοιχίας. Ας δούμε μερικά παραδείγματα:

Έστω ότι ένα σχήμα ενός νοσοκομείου αποθηκεύει όλους τους εργαζόμενους του νοσοκομείου σε έναν πίνακα Employee (empID, name, surname, salary, role).

Έστω ακόμη, ότι το σχήμα ενός άλλου νοσοκομείου διαχωρίζει τους εργαζόμενους του νοσοκομείου σε ξεχωριστούς πίνακες (πχ Nurse, Maid, Doctor, Secretary κλπ) με τις ιδιότητες (id,name,surname,salary) για κάθε πίνακα. Η μεθοδολογία του μηχανισμού που παρουσιάσθηκε σε αυτήν την εργασία, όπως είδαμε εστιάζει κυρίως στην εύρεση των σωστών 1:1 ισοδυναμιών ιδιοτήτων και στον συνδυασμό αυτών των ισοδυναμιών μέσω των απαραίτητων συνενώσεων. Κάτω από αυτό το πλαίσιο, η καλύτερη αντιστοιχία που μπορεί να βρεθεί για τον πίνακα Nurse είναι η εξής:

Create View Nurse(id,name,surname,salary) as

Select Employee.empid as id, Employee.name as name, Employee.surname as surname, Employee.salary as salary

From Employee;

Παρατηρούμε ωστόσο ότι η παραπάνω όψη αντιστοιχίζει τις νοσοκόμες του ενός νοσοκομείου σε όλους τους εργαζόμενους του άλλου, κάτι το οποίο είναι προφανώς λάθος σημασιολογικά. Ο τρόπος που θα μπορούσε να ξεπερασθεί αυτό το πρόβλημα είναι η προσθήκη του περιορισμού τιμής (value constraint): where Employee.role = “Nurse”, κάτι το οποίο όμως το σύστημα θα πρέπει να ανακαλύπτει (ημι)αυτόματα. Μια τέτοια επέκταση του μηχανισμού, θα ήταν εξαιρετικά ενδιαφέρουσα, μιας και σε αυτήν την περίπτωση, το σύστημα θα μπορούσε να παράγει αυτόματα μια επιπλέον πολύ σημαντική κατηγορία αντιστοιχιών, που εμφανίζεται πολύ συχνά σε κατανεμημένα περιβάλλοντα. Προς αυτήν την κατεύθυνση, η χρήση οντολογιών, ή η επεξεργασία των δεδομένων της βάσης ενός σχήματος, θα μπορούσε να συνεισφέρει σε πολύ σημαντικό βαθμό.

Μια δεύτερη επέκταση της εργασίας αυτής μπορεί να προκύψει αν θεωρήσουμε το αντίστροφο του σεναρίου που περιγράφηκε παραπάνω: έστω για παράδειγμα ότι θέλουμε να βρούμε την αντιστοιχία του πίνακα Employee προς το άλλο σχήμα. Παρατηρούμε ότι σε

αυτήν την περίπτωση μια ισχυρή σημασιολογικά αντιστοιχία δεν θα έπρεπε να περιλαμβάνει μόνο έναν κύριο πίνακα του άλλου σχήματος. Πράγματι, εργαζόμενος θεωρείται και η νοσοκόμα και η καθαρίστρια, και ο γιατρός και η γραμματέας. Ο μηχανισμός που παρουσιάστηκε με αυτή την εργασία περιορίζεται μόνο σε αντιστοιχίες που ο κύριος πίνακας είναι ένας, και κατά συνέπεια αποτελούνται μόνο από ένα ερώτημα στην όψη. Έτσι λοιπόν, δεν είναι σε θέση να ανακύψει αντιστοιχίες, όπου οι σχετικές όψεις αποτελούνται από ενώσεις (UNIONs) ερωτημάτων σαν την παρακάτω αντιστοιχία του πίνακα Employee:

Create View Employee(id,name,surname,salary,role) as

Select Nurse.id as empid, Nurse.name as name, Nurse.surname as surname, Nurse.salary as salary, “Nurse” as role

From Nurse;

UNION

Select Maid.id as empid, Maid.name as name, Maid.surname as surname, Maid.salary as salary, “Maid” as role

From Maid;

UNION

Select Doctor.id as empid, Doctor.name as name, Doctor.surname as surname, Doctor.salary as salary, “Doctor” as role

From Doctor;

UNION

Select Secretary.id as empid, Secretary.name as name, Secretary.surname as surname, Secretary.salary as salary, “Secretary” as role

From Secretary;

Και σε αυτό το σημείο, η χρήση οντολογιών με τις σχέσεις υπερώνυμων που αυτές προσφέρουν, κρίνεται ιδιαίτερα σημαντική.

Συμπερασματικά λοιπόν, βλέπουμε, ότι η προσέγγιση του προβλήματος Schema Mapping είναι μια σύνθετη και δύσκολη διαδικασία. Ωστόσο, ο εμπλουτισμός του μηχανισμού που προτείνει αυτή η διπλωματική εργασία με τις επεκτάσεις που αναφέρθηκαν θα μπορούσε να οδηγήσει σε ένα πλήρως ολοκληρωμένο σύστημα και να συνεισφέρει σημαντικά στις εξαιρετικές δυνατότητες των p2p συστημάτων βάσεων δεδομένων.

8 *Βιβλιογραφία*

[1] B. Alexe, L. Chiticariu, R. J. Miller, D. Pepper, and W. C. Tan. Muse: a system for understanding and designing mappings. In SIGMOD Conference, pages 1281–1284, 2008.

[2] B. Alexe, L. Chiticariu, R. J. Miller, and W. C. Tan. Muse: Mapping understanding and design by example. In ICDE, pages 10–19, 2008.

[3] Y. An, A. Borgida, R. J. Miller, and J. Mylopoulos. A semantic approach to discovering schema mapping Expressions. In ICDE, pages 206–215, 2007.

[4] P. Andritsos, A. Fuxman, A. Kementsietsidis, R. J. Miller, and Y. Velegrakis. Kanata: Adaptation and evolution in data sharing systems. SIGMOD Record, 33(4):32–37, 2004.

[5] H. Do and E. Rahm. Coma - A System for Flexible Combination of Schema Matching Approaches. In VLDB, 2002.

[6] M. A. Hernandez, H. Ho, L. Popa, A. Fuxman, R. J. Miller, T. Fukuda, and P. Papotti. Creating nested mappings with clio. In ICDE, pages 1487–1488, 2007.

[7] M. A. Hernandez, R. J. Miller, and L. M. Haas. Clio: A semi-automatic tool for schema mapping. In SIGMOD Conference, page 607, 2001.

[8] V. Kantere, D. Bousounis, and T. Sellis. A Tool for Mapping Discovery over Revealing Schemas. Submitted for publication. http://www.dbnet.ece.ntua.gr/˜vkante/mapping.

[9] V. Kantere, D. Bousounis, and T. Sellis. Mapping Discovery over Revealing Peer Schemas. Submitted for publication. http://www.dbnet.ece.ntua.gr/˜vkante/mapping.

[10] V. Kantere, D. Tsoumakos, T. Sellis, and N. Roussopoulos. GrouPeer: Dynamic clusterinf of P2P Databases. In Information Systems, doi:10.1016/j.is.2008.04.002, 2008.

[11] R. J. Miller, L. M. Haas, and M. A. Hernandez. Schema mapping as query discovery. In VLDB, pages 77–88, 2000.

[12] E. Rahm and P.Bernstein. A Survey of Approaches to Automatic Schema Matching. In VLDB Journal, 2001.

[13] Y. Velegrakis, R. J. Miller, and L. Popa. Preserving mapping consistency under schema changes. VLDB J., 13(3):274–293, 2004.

[14] Y. Velegrakis, R. J. Miller, L. Popa, and J. Mylopoulos. Tomas: A system for adapting mappings while schemas evolve. In

ICDE, page 862, 2004.

[15] Γεώργιος Ι. Ορφανουδάκης: «Υλοποίηση Μηχανισμού Ερωταποκρίσεων για Δίκτυο Ομότιμων Βάσεων Δεδομένων» Διπλωματική εργασία Γεώργιου Ι. Ορφανουδάκη, ΕΜΠ 2007

[16] Δημήτρης Μανακανάτας: «Σχεδιασμός και Υλοποίηση Εργαλείου για την ημιαυτόματη σημασιολογική συσχέτιση σχημάτων»: Μεταπτυχιακή Εργασία Δημήτρη Μανακανάτα, Πανεπιστήμιο Κρήτης 2006