Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας, Πληροφορικής & Υπολογιστών

Ανάπτυξη ποιοτικού συνόλου δεδομένων εκπαίδευσης για το πρόβλημα απάντησης σε ερώτημα σε βάσεις γνώσης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΜΥΡΣΙΝΗ ΒΑΚΑΛΟΠΟΥΛΟΥ

Επιβλέπων : Γεώργιος Στάμου
Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2019

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας, Πληροφορικής & Υπολογιστών

Ανάπτυξη ποιοτικού συνόλου δεδομένων εκπαίδευσης για το πρόβλημα απάντησης σε ερώτημα σε βάσεις γνώσης

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΜΥΡΣΙΝΗ ΒΑΚΑΛΟΠΟΥΛΟΥ

Επιβλέπων : Γεώργιος Στάμου
Αν. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 25η Νοεμβρίου 2019.

Γεώργιος Στάμου Αν. Καθηγητής Ε.Μ.Π.	Ανδρέας-Γεώργιος Σταφυλοπάτης Καθηγητής Ε.Μ.Π.	Νικόλαος Παπασπύρου Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2019

Μυρσίνη Βακαλοπούλου

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

Περίληψη

Η εξαγωγή απάντησης ενός ερωτήματος πάνω σε μία βάση γνώσης αποτελεί πρόβλημα υψηλής πολυπλοκότητας, η οποία πηγάζει από την ανάγκη να ελεγχθούν εξαντλητικά όλοι οι πιθανοί συμπερασμοί που μπορούν να εξαχθούν με βάση τη ”λογική” του συστήματος, ώστε να δοθεί πλήρης απάντηση. Το μεγαλύτερο μέρος της έρευνας πάνω στο πρόβλημα αυτό, έχει ως αντικείμενο την αντιμετώπιση αυτής της πολυπλοκότητας, είτε θυσιάζοντας την εκφραστικότητα της γλώσσας που περιγράφει τη γνώση, είτε θυσιάζοντας την πληρότητα των απαντήσεων, είτε βελτιστοποιώντας τις ήδη υπάρχουσες μεθόδους.

Στο πλαίσιο αυτό, προτείνουμε μία νέα προσέγγιση, αυτή της εκμάθησης ενός συνόλου λογικών αξιωμάτων από ένα σύστημα μηχανικής μάθησης, το οποίο θα μπορεί να απαντά αμέσως σε οποιαδήποτε ερώτημα και πάνω σε οποιαδήποτε βάση, σύμφωνα με τα αξιώματα του συνόλου αυτού. Η περιγραφική λογική με την οποία καταπιανόμαστε είναι η DL-Lite, η οποία χαρακτηρίζεται από εκφραστική απλότητα αλλά ταυτόχρονα σχετίζεται άμεσα με το πρόβλημα απάντησης σε ερώτημα σε βάσεις γνώσεις. Το σύνολο εκπαίδευσης που αναπτύσσουμε περιέχει όλα τα ελάχιστα χρήσιμα ερωτήματα που μπορούν να γίνουν σε σχέση με τα συγκεκριμένα λογικά αξιώματα, και στη συνέχεια, για κάθε ένα τέτοιο ερώτημα, παράγεται ένα σύνολο από ABoxes, την πλήρη απάντησή του στο ερώτημα αυτό αλλά και μία μετρική δυσκολίας για την εξαγωγή της απάντησης. Τα στοιχεία αυτά συνδυάζονται και παράγουν μεγαλύτερους κόσμους, εμπλουτίζοντας το σύνολο δεδομένων εκμάθησης.

Η μετρική δυσκολίας που παράγεται, διατάσσει τα ABoxes σε σχέση με ένα σύνολο λογικών αξιωμάτων κι ένα ερώτημα, με βάση την πολυπλοκότητα των συμπερασμών που πρέπει να εξαχθούν για να δοθεί πλήρης απάντηση. Η διάταξη αυτή δίνει τη δυνατότητα για ελεγχόμενη και όχι τυχαία ή εξαντλητική παραγωγή δεδομένων εκπαίδευσης του συστήματος μηχανικής μάθησης.

Λέξεις κλειδιά

Βάση Γνώσης, Οντολογία, ABox, TBox, Δεδομένα Εκπαίδευσης Μηχανικής Μάθησης, Πρόβλημα Απάντησης σε Ερώτημα, Επαναγραφή Ερωτημάτων, DL-Lite, Δυσκολία Επαναγραφής, Δυσκολία Εύρεσης Απάντησης

Abstract

Query-Answering over a knowledge base is a high complexity problem. The reason is that all the implicite entailments have to be extracted, according to system's logic, in order to give a complete answer. Most of the research on this problem is about reducing its complexity by sacrificing completeness either by using non-expressive description logics or by optimizing the existing methods of reasoning.

We propose a new approach on Query-Answering problem. That is, training a machine learning system to learn a set of logic rules and then answer any question on any ABox that is related with this set of rules. The description logic undertaken is DL-Lite, which is non-expressive but completely relevant with Query-Answering problem on knowledge bases. The training data we produce includes a query set of all the minimum queries that enable these logic rules in order to get a complete answer. For each of these queries, a base of ABoxes is produced, together with the certain answer and also a difficulty metric which describes the reasoning-cost to get this specific answer. We extend the data by combining its values and compose the dataset.

This difficulty metric defines an order between ABoxes, based on the specific query and the set of logic rules. This order, describes the complexity that is necessary to extract all the implicit knowledge in order to get a complete answer. Also, the machine learning system is able to be trained by specified and not random neither exhaustive data.

Key words

Knowledge Base, Ontology, ABox, TBox, Training Data for Machine Learning, Query-Answering Problem, Query Rewriting, DL-Lite, Rewriting Difficulty, Answer Difficulty

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Γιώργο Στάμου, για τη δημιουργική ελευθερία που μου προσέφερε αλλά και τη δυνατότητα να εκπονήσω μία ερευνητική εργασία. Επίσης, θα ήθελα να ευχαριστήσω τον κ. Αλέξανδρο Χορταρά για την απλόχερη βοήθεια που μου προσέφερε, ώστε να εξοικιωθώ με προχωρημένες έννοιες του κόσμου των Βάσεων Γνώσης. Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου, για την αμέτρητη στήριξή της κατά τη διάρκεια των σπουδών μου, αλλά και τους φίλους που έδωσαν στα χρόνια αυτά παραπάνω ενδιαφέρον.

Μυρσίνη Βακαλοπούλου,

Αθήνα, 25η Νοεμβρίου 2019

Περιεχόμενα

Περίληψη . . . 5

Abstract . . . 7

Ευχαριστίες . . . 9

Περιεχόμενα . . . 11

Κατάλογος πινάκων . . . 13

Κατάλογος σχημάτων . . . 15

Κατάλογος αλγορίθμων . . . 17

1. Εισαγωγή . . . 19
- 1.1 Ορισμοί . . . 19
- 1.2 Προκλήσεις . . . 21
- 1.3 Εύρος της εργασίας . . . 23
- 1.4 Συνεισφορά της εργασίας . . . 23
- 1.5 Δομή κεφαλαίων . . . 24

2. Θεωρητικό Υπόβαθρο . . . 25
- 2.1 Πρόβλημα απάντησης σε ερώτημα . . . 25
  - 2.1.1 Υλοποίηση . . . 25
  - 2.1.2 Επαναγραφή ερωτημάτων . . . 26
  - 2.1.3 Συσχετιστικά νευρωνικά δίκτυα . . . 28
- 2.2 Δημιουργία συνόλου δεδομένων εκπαίδευσης στη μηχανική μάθηση . . . 28
- 2.3 Περιγραφική λογική συστήματος . . . 29
  - 2.3.1 Περιγραφική λογική DL-Lite . . . 29
  - 2.3.2 Σύνταξη . . . 30
    - 2.3.2.1 DL-Lite$_{core}$ . . . 30
    - 2.3.2.2 DL-Lite$_{R}$ . . . 31
- 2.4 Σύστημα SyGENiA . . . 32
  - 2.4.1 Γενικά . . . 32
  - 2.4.2 Σύνολο δεδομένων ελέγχου πληρότητας . . . 32
  - 2.4.3 Παραγωγή των ερωτημάτων . . . 33
- 2.5 Αλγόριθμος Rapid . . . 38
  - 2.5.1 Γενικά . . . 38
  - 2.5.2 Διαισθητική επεξήγηση αλγορίθμου . . . 39
    - 2.5.2.1 Πρώτος αλγόριθμος ξετυλίγματος . . . 41
    - 2.5.2.2 Δεύτερος αλγόριθμος ξετυλίγματος . . . 42

3. Σχεδιασμός . . . 45
3.1 Ορισμοί . . . 45
3.1.1 Γενικοί χαρακτηρισμοί . . . 45
3.1.2 Δυσκολία ABox . . . 46
3.1.3 Δυσκολία επαναγραφής . . . 48
3.2 Βάση συνόλου εκπαίδευσης . . . 49
3.2.1 Βάση των ελάχιστων ABoxes . . . 49
3.2.2 ABox επαναγραφής . . . 49
3.2.3 Απλή βάση των ABoxes . . . 50
3.2.4 Εκτεταμένη απλή βάση των ABoxes . . . 52
3.2.5 Σχηματισμός εκτεκατμένης απλής βάσης των ABoxes . . . 52
3.3 Ερωτήματα εκπαίδευσης . . . 53
3.4 Εξαγωγή δυσκολίας . . . 54
3.4.1 Γράφος ξετυλιγμάτων . . . 54
3.4.2 Υπολογισμός δυσκολίας δύο όρων . . . 55
3.4.3 Υπολογισμός δυσκολίας δύο ερωτημάτων . . . 56
3.4.3.1 Υπολογισμός δυσκολίας από ξετυλίγματα . . . 58
3.4.3.2 Υπολογισμός δυσκολίας από συρρικνώσεις . . . 60
3.5 Σχηματισμός συνόλου δεδομένων εκπαίδευσης . . . 61

4. Υλοποίηση-Πειράματα . . . 65
4.1 Δημιουργία συνόλου για πειράματα . . . 65
4.1.1 Υπολογισμός δυσκολίας ABox . . . 65
4.2 Java-OWL . . . 67

5. Επίλογος . . . 71
5.1 Σύνοψη . . . 71
5.2 Συμπεράσματα - Συνεισφορά . . . 72
5.3 Μελλοντικές κατευθύνσεις . . . 73

Βιβλιογραφία . . . 75

Κατάλογος πινάκων

2.1 Chase rules για $\mathcal{L} \in \{\text{DL-Lite}, \mathcal{EL}\}$ 36
2.2 Κανόνες αντικατάστασης DL-Lite_R λογικών αξιωμάτων σε προτάσεις λογικής πρώτης τάξης 40

Κατάλογος σχημάτων

1.1 Σχηματική αναπαράσταση νευρωνικού δικτύου 23

2.1 Σχηματική αναπαράσταση συστήματος Rapid 40

3.1 Παράδειγμα 3.4.1. 55
3.2 Παράδειγμα 3.4.4. 59
3.3 Σχηματική αναπαράσταση αλγορίθμου εξαγωγής δυσκολίας σε συρρίκνωσης . . . 60
3.4 Σχηματική αναπαράσταση συστήματος DatAbox 63

4.1 Σχηματική αναπαράσταση υλοποιημένου συστήματος και DifficAbox 68

Κατάλογος αλγορίθμων

1 CalcRewABoxAns 51
2 CalcExtSimpleAboxBase 53
3 CalcDifficultyFromGraph 57
4 CalcUnfoldingDifficulty 59
5 CalcShrinkingDifficulty 61
6 CalcDatAbox 62
7 SimpleJoin 62
8 CalcDatAboxK 63
9 CalcUnfoldingDifficultyABox 66

Κεφάλαιο 1

Εισαγωγή

Σε αυτό το κεφάλαιο γίνεται μία σύντομη εισαγωγή του αντικειμένου της παρούσας διπλωματικής εργασίας. Αρχικά, στην ενότητα 1.1, δίνονται κάποιοι βασικοί ορισμοί που είναι χρήσιμοι για να κατανοήσει κανείς βασικές έννοιες στον τομέα των βάσεων γνώσης. Στη συνέχεια, στην ενότητα 1.2, παρουσιάζεται το γενικότερο πρόβλημα και οι προκλήσεις τις οποίο πραγματεύεται η εργασία αυτή και τέλος, στην ενότητα 1.3 σχολιάζεται η συνεισφορά της.

1.1 Ορισμοί

Αρχικά, καταγράφονται ορισμοί που εισάγουν τον αναγνώστη στον κόσμο των οντολογιών ή αλλιώς βάσεων γνώσης, και οι οποίοι χρησιμοποιούνται καθόλη τη διάρκεια της εργασίας.

Ορισμός 1. Καλούμε *σώμα ορολογίας (terminology box - TBox)* ένα σύνολο από αξιώματα υπαγωγής και ισοδυναμίας εννοιών και ρόλων.

Ορισμός 2. Καλούμε *σώμα ισχυρισμών (assertion box - ABox)* ένα σύνολο από ισχυρισμούς ισότητας ή ανισότητας στιγμιοτύπων, ισχυρισμούς εννοιών και ισχυρισμούς ρόλων.

Ορισμός 3. Καλούμε *οντολογία (ontology)* ή *βάση γνώσης (knowledge base - KB)* την ένωση ενός TBox, που περιέχει τα λογικά αξιώματα, κι ενός ABox, που περιέχει τα αξιώματα ιδιότητας μέλους.

Ορισμός 4. Καλούμε *άτομο (atom)* μίας οντολογίας, ένα στιγμιότυπο μίας έννοιας ή ενός ρόλου, όπως αναγράφεται στο ABox. Ένα *ανώνυμο (unamed)* άτομο είναι ένα άγνωστο άτομο για την οντολογία, δηλαδή δεν προσδιορίζεται ακριβώς και μπορεί να είναι κάποιο από τα ονομασμένα. Ένα *ονομασμένο (named)* άτομο αποτελεί ένα διακεκριμένο άτομο στην οντολογία.

Ορισμός 5. Καλούμε *περιγραφική λογική* μίας οντολογίας *(description logic - DL)* ένα τυπικό φορμαλισμό αναπαράστασης της οντολογίας, δηλαδή ένα σύνολο από τελεστές, πράξεις και σχέσεις μεταξύ των εννοιών και ρόλων, με σκοπό να εκφραστεί η γνώση που περιέχει. Ανάλογα με το σύνολο αυτό, καθορίζεται η εκφραστικότητα μίας περιγραφικής λογικής, δηλαδή η δυνατότητα να εκφράζονται σύνθετες έννοιες.

Ορισμός 6. Καλούμε *εξαγωγή συμπερασμών (reasoning)* τη διαδικασία ανάλυσης ισχυρισμών μίας οντολογίας, με βάση τα λογικά αξιώματα του TBox, που οδηγεί στην παραγωγή νέων ισχυρισμών. Το προγραμματιστικό εργαλείο που υλοποιεί εξαγωγές συμπερασμών ονομάζεται εξαγωγέας συμπερασμών *(reasoner)*.

Ορισμός 7. Καλούμε *συζευκτικό ερώτημα (conjunctive query - CQ)* μία έκφραση της μορφής

$$q(\vec{x}) = \{q_1, q_2, ..., q_n\},$$

όπου $q_i, i \in N_n$, είναι ένα ατομικό ερώτημα της μορφής

$$q(x) = C(x) \text{ ή } q(x, y) = R(x, y)$$

όπου C, R ατομικά. Το διάνυσμα $\vec{x}$ ονομάζεται *διάνυσμα μεταβλητών (variable vector)* του q και κάθε στοιχείο του, δηλαδή κάθε μεταβλητή του q, απαιτείται να περιλαμβάνεται τουλάχιστον σε ένα ατομικό ερώτημα q_i. Στην παρούσα εργασία αναφερόμαστε μόνο σε συζευτκικά ερωτήματα.

Ορισμός 8. Έστω το συζευκτικό ερώτημα q. Μία μεταβλητή που δεν περιέχεται στο διάνυσμα μεταβλητών του q αλλά εμφανίζεται στο σώμα του q θα ονομάζεται *ελεύθερη μεταβλητή (free/unbound variable)* ενώ μία μεταβλητή που ανήκει στο διάνυσμα μεταβλητών του q θα ονομάζεται *μεταβλητή απάντησης (answer/bound variable).*

Ορισμός 9. Καλούμε *βέβαιη απάντηση (certain answer)* ενός ερωτήματος q πάνω σε μία οντολογία $\mathcal{O}$, και συμβολίζουμε με $\text{cert}(q, \mathcal{O})$ ή $\text{cert}(q, \mathcal{T}, \mathcal{A})$, ένα διάνυσμα ατόμων c, ίδιας διάστασης με το διάνυσμα απάντησης του q, αν και μόνο αν σε κάθε μοντέλο της $\mathcal{O}$, το c είναι απάντηση του q. Με άλλα λόγια, η βέβαιη απάντηση είναι η ορθή και πλήρης απάντηση ενός του q πάνω στην $\mathcal{O}$.

Ορισμός 10. Ορίζουμε τον συμβολισμό $\mathbf{A} \leftarrow \{\mathbf{B}_i\}_{i=1}^n$ για συζευκτικά ερωτήματα, όπου $\mathbf{A}$ είναι η *κεφαλή (head)* του ερωτήματος και τα άτομα $\mathbf{B}_i$ το *σώμα (body).* Ένα άτομο που ανήκει στο σώμα μπορεί να είναι της μορφής $A(t)$ ή $A(t, s)$, ανάλογα με το αν είναι άτομο ρόλου ή έννοιας.

Ορισμός 11. Ένας *αλγόριθμος απάντησης* συζευκτικών ερωτημάτων για μία περιγραφική γλώσσα $\mathcal{L}$ είναι μία διαδικασία όπου για κάθε $\mathcal{L}$-οντολογία $\mathcal{O}$ και κάθε συζευκτικό ερώτημα q υπολογίζει σε πεπερασμένο αριθμό βημάτων ένα σύνολο από διανύσματα απάντησης $\text{ans}(q, \mathcal{O})$ ίδιας πληθηκότητας με τη βέβαιη απάντηση. Ο αλγόριθμος αυτός είναι:

- *Ορθός (sound)*, αν για κάθε $\mathcal{O}$ και q ισχύει ότι $\text{ans}(q, \mathcal{O}) \subseteq \text{cert}(q, \mathcal{O})$.
- *Πλήρης (complete)*, αν για κάθε $\mathcal{O}$ και q ισχύει ότι $\text{cert}(q, \mathcal{O}) \subseteq \text{ans}(q, \mathcal{O})$.
- *Μονοτονικός (monotonic)*, αν για κάθε $\mathcal{O}, \mathcal{O}'$ και q με $\mathcal{O} \subseteq \mathcal{O}'$, ισχύει ότι $\text{ans}(q, \mathcal{O}) \subseteq \text{ans}(q, \mathcal{O}')$.
- *Αμετάβλητος υπό μετονομασίες (invariant under renamings)*, αν για κάθε q, $\mathcal{O} = \mathcal{T} \cup \mathcal{A}$ και $\mathcal{O}' = \mathcal{T} \cup \mathcal{A}'$ με $\mathcal{A}, \mathcal{A}'$ ισομορφικά, ισχύει ότι $\text{ans}(q, \mathcal{O})$, $\text{ans}(q, \mathcal{O}')$ ταυτίζονται modulo τον ίδιο ισομορφισμό.

Λέμε ότι ο ans είναι καλώς ορισμένος *(well-behaved)* αν είναι ορθός, μονοτονικός και αμετάβλητος υπό μετονομασίες.

Ορισμός 12. Καλούμε *πρόβλημα απάντησης ερωτημάτων (query answering)* πάνω σε μία οντολογία, το πρόβλημα εξαγωγής διανυσμάτων απαντήσεων με βάση ένα ερώτημα q και μία οντολογία $\mathcal{O}$. Πιο διασθητικά, είναι το πρόβλημα εξαγωγής ατόμων της οντολογίας που έχουν συγκεκριμένες ιδιότητες, όσον αφορά τόσο τα χαρακτηριστικά τους όσο και τις συσχετίσεις τους με άλλα άτομα.

Ορισμός 13. Καλούμε *επαναγραφή ερωτημάτων (query rewriting)* ενός ερωτήματος q, με βάση ένα TBox $\mathcal{T}$, τη δημιουργία ενός εξαντλητικού συνόλου S από ερωτήματα που προκύπτουν από το q με επίλυση των λογικών αξιωμάτων του $\mathcal{T}$. Η διαδικασία αυτή, αν μπορεί να είναι εξαντλητική και δεν απειρίζεται, ή δεν παράγει απείρως μεγάλα ερωτήματα, μπορεί να αντικαταστήσει το πρόβλημα απάντησης στο q υπό το $\mathcal{T}$, με το πρόβλημα απάντησης κάθε ερωτήματος του S με κενό TBox. Φορμαλιστικά, $\text{cert}(q, \mathcal{T}, \mathcal{A}) = \bigcup_{q_i \in S} \text{cert}(q_i, \emptyset, \mathcal{A})$.

Εν συνεχεία, καταγράφονται ακόμα κάποιοι πολύ βασικοί ορισμοί που αφορούν τη μηχανική μάθηση.

Ορισμός 14. Η *Μηχανική Μάθηση (Machine Learning)* ορίζεται ως το πεδίο μελέτης που δίνει σε ένα σύστημα την ικανότητα να μαθαίνει, χωρίς να έχει ρητά προγραμματιστεί. Στο πεδίο αυτό, βασική μελέτη αποτελεί η κατασκευή αλγορίθμων που μπορούν αρχικά να εκπαιδεύονται από δεδομένα και στη συνέχεια να κάνουν προβλέψεις σχετικά με αυτά. Για την εκπαίδευση χρησιμοποιούνται πειραματικά δεδομένα, προκειμένου να γίνουν προβλέψεις βασιζόμενες στα δεδομένα ή να εξαχθούν αποφάσεις που εκφράζονται ως το αποτέλεσμα.

Ορισμός 15. Το *Νευρωνικό Δίκτυο (Neural Network)* είναι η δομική μονάδα ενός συστήματος μηχανικής μάθησης και αποτελείται από απλούς υπολογιστικούς κόμβους, τους νευρώνες. Ο νευρώνας μοντελοποιεί ένα νευρώνα του ανθρώπινου εγκεφάλου, με την έννοια ότι δέχεται ένα σύνολο εισόδων, επιτελεί έναν υπολογισμό υπό κάποιες παραμέτρους και τελικά εξάγει μία έξοδο.

1.2 Προκλήσεις

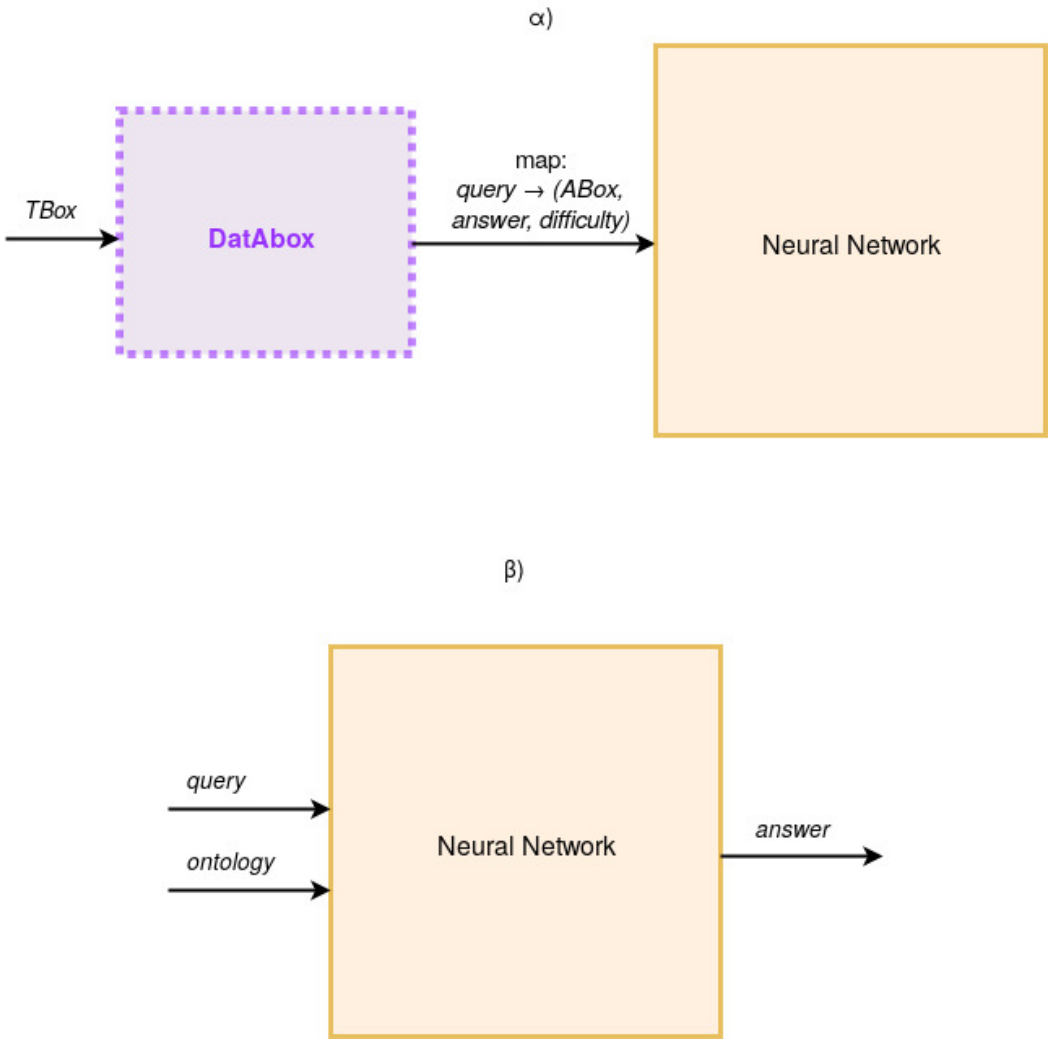
Οι βάσεις γνώσης, κυρίως σε συνδυασμό με το ζήτημα απαντήσεων σε ερωτήματα, αποτελούν πολύ βασική εφαρμογή της τεχνητής νοημοσύνης. Ουσιαστικά, οι βάσεις γνώσεις εξελίσσουν τις απλές σχεσιακές βάσεις, οι οποίες απλά καταγράφουν πληροφορία, σε βάσεις με νοημοσύνη, εισάγοντας λογικούς κανόνες που αφορούν τις εγγραφές και που δημουργούν σχέσεις μεταξύ τους. Το πρόβλημα ερώτησης - απάντησης βασίζεται ακριβώς στο παραπάνω απαιτώντας από το σύστημα να "σκεφτεί", δηλαδή να εξάγει συμπερασμούς, ώστε να δώσει μία ορθή και πλήρη απάντηση στο ερώτημα που του τίθεται. Εκτός από το παραπάνω πρόβλημα, πολύ συνηθισμένα στον τομέα των βάσεων γνώσης είναι το πρόβλημα ικανοποιησιμότητας ή υπαγωγής ενός ισχυρισμού.

Το σοβαρότερο ζήτημα με τις βάσεις γνώσης είναι η κλιμάκωση. Η εξαγωγή συμπερασμών, που απαιτείται για να συμπεριφέρεται με ευφυΐα το σύστημα, χαρακτηρίζεται από πολύ υψηλή πολυπλοκότητα. Συγκεκριμένα, όσο πιο εκφραστική είναι η περιγραφική λογική της οντολογίας, τόσο αυξάνεται η πολυπλοκότητα. Αυτό διαφαίνεται από την πρώτη στιγμή μελέτης της περιοχής, από τον πιο απλό αλγόριθμο ανάλυσης. Μέχρι τώρα, βασικός σκοπός της έρευνας πάνω στην περιοχή είναι η μελέτη περιγραφικών λογικών με περιορισμένη εκφραστικότητα, σχεδιασμός συστημάτων που κλιμακώνουν αλλά θυσιάζουν την πληρότητά τους και, βέβαια, τεχνικές βελτιστοποίησης της αποδόσης της εξαγωγής συμπερασμών.

Ένα άλλο ζήτημα, που αφορά τα νευρωνικά δίκτυα και συγκεκριμένα την εκπαίδευσή τους, είναι οι υπολογιστικοί πόροι (resources) που απαιτούνται για να λάβει χώρα αυτή η διεργασία, κυρίως όταν αναφερόμαστε σε συνελικτικά νευρωνικά δίκτυα και όταν η γνώση που θέλουμε να ενσωματωθεί είναι περίπλοκη. Τα συνελικτικά νευρωνικά δίκτυα είναι πολύ πιο απαιτητικά σε

πόρους από τα απλά νευρωνικά δίκτυα λόγω της μεγαλύτερης πολυπλοκότητάς τους, η οποία όμως ευθύνεται και για τα καλύτερα αποτελέσματα που παρέχουν. Επίσης, όσο πιο σύνθετη είναι η γνώση που θέλουμε να "μάθει" ένα νευρωνικό, τόσο πιο σύνθετο και συνήθως μεγαλύτερο είναι το σύνολο δεδομένων εκπαίδευσης. Η έλλειψη υπολογιστικών πόρων μπορεί να περιορίσει την εκπαίδευση και να οδηγήσει είτε σε ατελείωτο χρόνο παραμονής για να ολοκληρωθεί η διαδικασία, είτε, συνήθως, σε εκπαίδευση με περιορισμένα δεδομένα και άρα σε χειρότερα αποτελέσματα.

Στην παρούσα εργασία, περιοριζόμαστε στη βελτιστοποίηση του προβλήματος απάντησης σε ερωτήματα μίας DL-Lite-οντολογίας και εισάγουμε μία νέα ιδέα αντιμετώπισης της τόσο υψηλής πολυπλοκότητας της εξαγωγής συμπερασμών. Συγκεκριμένα, δημιουργούμε ένα σύνολο δεδομένων εκπαίδευσης με σκοπό την εκμάθηση των λογικών αξιωμάτων της οντολογίας από ένα νευρωνικό δίκτυο, ώστε να απαντάει σε κάθε ερώτημα που μπορεί να προκύψει. Εκπαιδεύουμε, δηλαδή, ένα νευρωνκό δίκτυο να εξάγει αυτόματα τις απαντήσεις σε ένα ερώτημα, έχοντας ενσωματώσει τη διαδικασία εξαγωγής συμπερασμών. Τα ερωτήματα που χρησιμοποιούνται στην εκπαίδευση είναι πεπερασμένου πλήθους και ενεργοποιούν κάθε πιθανό συμπερασμό της οντολογίας. Για κάθε ερώτημα, δημιουργείται ένα σύνολο δεδομένων από τριπλέτες $(\mathcal{A}_q, \text{cert}(q, \emptyset, \mathcal{A}_q), \text{δυσκολία}(\mathcal{A}_q))$, όπου q είναι το ερώτημα και $\mathcal{A}_q$ ένα ABox ειδικά δημιουργημένο για το q. Το τρίτο πεδίο της τριπλέτας είναι μία μετρική δυσκολίας που αφορά το συγκεκριμένο ABox και σχετίζεται με την πολυπλοκότητα των συμπερασμών που πρέπει να πραγματοποιηθούν ώστε να εξαχθεί η βέβαιη απάντηση στο q. Περισσότερα για το σύστημα, το οποίο ονομάστηκε *DatAbox*, βρίσκονται στο 3.



Σχήμα 1.1: a) Αναπαράσταση συστήματος εκπαίδευσης. b) Αναπαράσταση λειτουργίας νευρωνικού δικτύου μετά την εκπαίδευση.

1.3 Εύρος της εργασίας

Το εύρος της συγκεκριμένης εργασίας είναι αρκετά μεγάλο, αφού πραγματεύεται το πρόβλημα ερώτησης - απάντησης σε οποιαδήποτε βάση γνώσης που περγράφεται από DL-Lite. Ουσιαστικά, ο μόνος περιορισμός αφορά την περιγραφική λογική. Όμως, τονίζεται εδώ, ότι η συγκεκριμένη περιγραφική λογική προορίζεται συνήθως για οντολογίες που βασική διεργασία αυτών είναι η επίλυση του συγκεκριμένου προβλήματος, επομένως δεν καλύπτουμε απλά την περίπτωση μίας τυχαίας περιγραφικής λογικής. Σημειώνεται ότι ακόμα και αυτός ο περιορισμός μπορεί να αφαιρεθεί σε μελλονική έρευνα για την επέκταση του συστήματος πάνω σε πιο περιγραφικές λογικές, όπως η $\mathcal{EL}$.

1.4 Συνεισφορά της εργασίας

Το αξιοσημείωτο σημείο αυτής της εργασίας είναι η αυτοματοποιημένη δημιουργία συνόλου εκπαίδευσης για το πρόβλημα απάντησης ενός ερωτήματος σε οντολογίες, το οποίο όμως αναπτύ-

σεται στοχευμένα και όχι τυχαία ή εξαντλητικά. Αυτό σημαίνει, ότι ένα νευρωνικό δίκτυο που δε διαθέτει αρκετούς πόρους για να εκπαιδευτεί από ένα τεράστιο σύνολο δεδομένων, το οποίο διαθέτει πιθανώς και άχρηστα στοιχεία, μπορεί να χρησιμοποιήσει το σύστημά μας το οποίο εγγυάται δημιουργία ποιοτικού κι εν μέρη ελεγχόμενου σύνολου δεοδομένων, ανεξάρτητα από το μέγεθός του. Δηλαδή, ακόμα και μικρό να είναι το παραγόμενο σύνολο δεδομένων, θα είναι σίγουρα εύστοχο. Η ελεγχόμενη ανάπτυξη του συνόλου που αναφέρθηκε, αφορά την επίγνωση του μεγέθους των ABox στα δεδομένων, το οποίο κατά κάποιο τρόπο μπορεί να καθοριστεί.

Επίσης, η βάση για το σχηματισμό του συνόλου αυτού είναι ένα πεπερασμένο πλήθος από ελάχιστα ερωτήματα που σχετίζονται με το TBox της ονοτλογίας, το οποίο έχει εύστοχα επιλεχθεί, όπως αναλύεται στο 2. Επίσης, το σύνολο αυτό είναι πλήρως αντιπροσωπευτικό του TBox και δεν περιέχει περιττά στοιχεία. Κάτι τέτοιο, περιορίζει το πλήθος στοιχείων της βάση στο ελάχιστο, και άρα εισάγει την ελάχιστη πολυπλοκότητα, ενώ παράλληλα παραμένει πλήρες ως προς την κάλυψη των κανόνων του TBox.

Ακόμα, τονίζεται ότι καθώς το σύνολο σχηματίζεται με βάση το σύνολο αξιωμάτων της μελετούμενης οντολογίας, δεν υπάρχει εξάρτηση από το εκάστοτε ABox το οποίο σημαίνει ότι δε στοχεύουμε στην εκμάθηση μίας συγκεκριμένης οντολογίας αλλά στην "κατανόηση" και ενσωμάτωση συγκεκριμένων λογικών κανόνων. Κάτι τέτοιο είναι πολύ χρήσιμο, καθώς δεν χρειάζεται να επανεκπαιδευτεί το νευρωνικό για δύο οντολογίες με ίδιους κανόνες, δηλαδή ίδια νοημοσύνη, που απλώς έχουν διαφορετικούς ισχυρισμούς στον κόσμο τους, καθώς η δυσκολία και η ουσία παραμένει στην εκμάθηση των λογικών κανόνων.

Τέλος, η εξαγωγή μίας μετρικής δυσκολίας κάποιου ABox σε σχέση με κάποιο ερώτημα, που προκύπτει από τη μέτρηση της δυσκολίας επαναγραφών του ερωτήματος, ομαδοποιεί τα ABoxes με βάση τη δυσκολία συμπερασμών που πρέπει να εξχθούν για να απαντηθεί το ερώτημα, κι επομένως, κατηγοριοποιεί τη διαδικασία εκμάθησης συμπερασμών. Αυτό είναι ένα ακόμα χαρακτηριστικό του συστήματος που το κάνει ελεγχόμενο. Επίσης, η μετρική αυτή μπορεί να φανεί χρήσιμη και σε άλλες μελέτες πάνω στις βάσεις γνώσης. Σημειώνεται επίσης ότι η μετρική αυτή δύναται να τροποποιηθεί σχετικά εύκολα ώστε να εξάγει και άλλα στοιχεία που αφορούν τις επαναγραφές και κατ' επέκταση και τα ABoxes.

1.5 Δομή κεφαλαίων

Στο Κεφάλαιο 2 αρχικά καλύπτεται το βασικό θεωρητικό υπόβαθρο του προβλήματος απάντησης ερωτημάτων σε οντολογίες, όπως και της αυτόματης δημιουργίας συνόλων δεδομένων εκπαίδευσης. Στη συνέχεια, παρουσιάζεται αναλυτικά η περιγραφική λογική DL-Lite, όπως επίσης κομβικές για την παρούσα εργασία μελέτες και εργαλεία, πάνω στα οποία στηρίχτηκε η ανάπτυξη του DatAbox. Στο Κεφάλαιο 3 αναλύεται το συστήματός μας, αιτιολογούνται οι επιλογές που κάναμε και παρουσιάζονται οι αλγόριθμοι που σχεδιάστηκαν. Στο Κεφάλαιο 4, παρουσιάζονται κάποιες αλλαγές των αλγορίθμων του συστήματός μας, ώστε να αξιολογηθεί πάνω σε ήδη υλοποιημένο σύστημα μηχανικής μάθησης, κι εν συνεχεία, σχολιάζεται η υλοποίηση των αλγορίθμων αυτών αλλά και του συστήματος γενικά σε προγραμματιστικό περιβάλλον. Τέλος, στο Κεφάλαιο 5 συνοψίζονται τα βασικά σημεία της παρούσας εργασίας και αναφέρονται μελλοντικές κατευθύνσης επιστημονικής μελέτης για την επέκτασή της.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

Στο παρόν κεφάλαιο καλύπτεται αρχικά το βασικό θεωρητικό υπόβαθρο του προβλήματος απάντησης σε ερωτήματα σε βάσεις γνώσης, για βαθύτερη κατανόηση του προβλήματος και παρουσίαση των μελετών που έχουν πραγματοποιηθεί στον τομέα αυτό. Επίσης, παρουσιάζεται πλήρως η περιγραφική λογική που χρησιμοποιείται. Στη συνέχεια, γίνεται ανάλυση δύο προγραμματιστικών εργαλείων που αποτελούν τη βάση του συστήματός μας: των SyGENiA και Rapid.

2.1 Πρόβλημα απάντησης σε ερώτημα

Το πρόβλημα απάντησης σε ερώτημα είναι ένα από τα σημαντικότερα προβλήματα των βάσεων γνώσης. Το πρόβλημα αυτό είναι πολύ δύσκολο να λυθεί, ειδικά σε περιγραφικές λογικές υψηλής εκφραστικότητας. Η διαφορά με τις απλές σχεσιακές βάσεις είναι ότι στις βάσεις γνώσης υπάρχει η υπονοούμενη γνώση η οποία δεν είναι ρητά δηλωμένη αλλά προκύπτει από την εφαρμογή των λογικών κανόνων του TBox πάνω στη ρητά δηλωμένη γνώση. Ουσιαστικά, είναι η γνώση που εξάγεται από λογικούς συμπερασμούς του συστήματος. Η υψηλή πολυπλοκότητα του πρόβληματος έγκειται ακριβώς σε αυτό, δηλαδή στην εξαγωγή όλων των συμπερασμών με βάση το TBox, το οποίο λύνεται σε εκθετικό χρόνο [Stamou and Chortaras (2017)].

Στην ενότητα αυτή παρουσιάζονται οι βασικές κατευθύνσεις επίλυσης του προβλήματος απάντησης σε ερώτημα σε βάσεις γνώσης. Αρχικά αναλύεται σύντομα η διαδικασία της *υλοποίησης (materialization)*, στη συνέχεια η μέθοδος επαναγραφής ερωτημάτων και τέλος το γενικό πλαίσιο χρήσης μηχανικής μάθησης ως μέθοδο επίλυσης, μέσα από τα λεγόμενα *συσχετιστικά νευρωνικά δίκτυα (relational neural networks)*.

2.1.1 Υλοποίηση

Η υλοποιήση είναι η διαδικασία επέκτασης του ABox μίας οντολογίας με εφαρμογή των λογικών αξιωμάτων του TBox. Ουσιαστικά, στους ισχυρισμούς του αρχικού ABox εφαρμόζονται οι λογικοί κανόνες του TBox με σκοπό να εξαχθεί όλη η υπονοούμενη γνώση που προκύπτει από αυτούς και να προστεθεί τελικά ως ρητή γνώση στο ABox. Με αυτό τον τρόπο, εφόσον είναι δυνατόν η διαδικασία της υλοποίησης να είναι πλήρης, επιτυγχάνεται απαλοιφή του TBox και άρα η μετατροπή της βάσης γνώσης σε μία απλή σχεσιακή βάση, στην οποία αρκεί να ερωτηθεί το αρχικό ερώτημα ως ένα SQL-ερώτημα ώστε να δοθεί πλήρης απάντηση. Η εφαρμογή των κανόνων για την εξαγωγή της υπονοούμενης γνώσης ονομάζεται *ανάλυση ή εξαγωγή συμπερασμών (reasoning)* και πραγματοποιείται από τους αλγορίθμους ανάλυσης, γνωστούς ως αλγορίθμους tableau. Η πο-

λυπλοκότητα των αλγορίθμων tableau είναι πολύ υψηλή, συγκεκριμένα εκθετική στο μέγεθος του ABox [Volz et al. (2005), Stamou and Chortaras (2017)].

Όπως αναφέρθηκε, για να έχει νόημα η υλοποίηση ώστε να μετατρέπει την οντολογία σε μία σχεσιακή βάση δεδομένων, θα πρέπει να πραγματοποιείται εξαντλητικά, ώστε να ενσωματώνει όλη την υπονοούμενη γνώση. Όμως, οι αλγόριθμοι tableau, όπως είναι γνωστό, υπάρχει περίπτωση να απειρίζονται σε συγκεκριμένες περιγραφικές λογικές. Συγκεκριμένα, απειρίζεται σε περιγραφικές λογικές που επιτρέπουν λογικά αξιώματα που η εφαρμογή τους σε ένα ABox προκαλεί επαναληπτικά αύξηση του μεγέθους του μέσω εισαγωγής νέων ατόμων. Σε τέτοιες περιπτώσεις, η διαδικασία της υλοποίησης δεν μπορεί να είναι εξαντλητική κι έτσι διακόπτεται, με αποτέλεσμα να μην λαμβάνεται σίγουρα πλήρης απάντηση στο SQL-ερώτημα. Μία περίπτωση που συμβαίνει απειρισμός είναι να υπάρχουν κύκλοι αξιωμάτων στο TBox στους οποίους προστίθονται συνεχώς νέα άτομα. Αυτό φαίνεται και στο παρακάτω παράδειγμα.

Παράδειγμα 2.1.1. Έστω ένα $\mathcal{EL}$-TBox $\mathcal{T}$:

$\mathcal{T} = \{\text{Parent} \sqsubseteq \exists\, \text{hasChild.Child},\ \text{Child} \sqsubseteq \exists\, \text{hasParent.Parent}\}$,

το οποίο εκφράζει ότι κάθε γονιός έχει ένα παιδί και ότι κάθε παιδί έχει ένα γονιό. Έστω επίσης ένα ABox $\mathcal{A}$ κι ένα συζευκτικό ερώτημα q ως εξής:

$\mathcal{A} = \{\text{Parent}(a)\}$

$q(x) = \{\text{Child}(x)\}$

Τα στάδια της διαδικασίας υλοποίησης, δηλαδή ενός αλγορίθμου tableau για την εξαγωγή όλων των συμπερασμών ακολουθεί τα εξής στάδια (με bold αναγράφεται ο όρος που έχει αναλυθεί σε κάθε βήμα):

$\mathcal{A}_1 = \{\textbf{Parent}(a), \text{hasChild}(a, b), \text{Child}(b)\}$

$\mathcal{A}_2 = \{\text{Parent}(a), \text{hasChild}(a, b), \textbf{Child}(b)\ , \text{hasParent}(b, c), \text{Parent}(c)\}$

$\mathcal{A}_3 = \{\text{Parent}(a), \text{hasChild}(a, b), \text{Child}(b)\ , \text{hasParent}(b, c),\ \textbf{Parent}(c)\ , \text{hasChild}(c, d), \text{Child}(d)\}$

...

Παρατηρούμε ότι η διαδικασία απειρίζεται και άρα πρέπει να να τη διακόψουμε σε συγκεκριμένο σημείο το οποίο έχει ως αποτέλεσμα να μη μπορεί να δοθεί πλήρης απάντηση στο ερώτημα.

Γενικά το πρόβλημα απειρισμών είναι ένα βασικό πρόβλημα στις βάσεις γνώσης για την εξαγωγή της υπονοούμενης γνώσης κι εμφανίζεται κάθε φορά που ζητείται η γνώση αυτή. Δηλαδή, το πρόβλημα δεν ανήκει μόνο στη διαδικασία της υλοποίησης. Όμως, σημειώνεται ότι υπάρχουν λιγότερο εκφραστικές περιγραφικές λογικές οι οποίες αποκλείουν τέτοιους κύκλους και γενικότερα τον απειρισμό της διαδικασίας. Οι λογικές αυτές είναι εκφραστικά πολύ περιορισμένες, και γι' αυτό η υλοποίηση είναι χρήσιμη και πλήρης.

2.1.2 Επαναγραφή ερωτημάτων

Στη συνέχεια, παρουσιάζεται η μέθοδος επαναγραφής ερωτημάτων, η οποία βασίζεται στη δημιουργίας ενός συνόλου από ερωτήματα που έχουν ενσωματώσει τους λογικούς κανόνες του TBox της οντολογίας, το οποίο τελικά είναι ισοσύναμο του αρχικού ερωτήματος. Όπως γίνεται αντιληπτό, η μέθοδος αυτή διαχωρίζει τη διαδικασία της ανάλυσης από το ABox, μεταφέροντάς την εξολοκλήρου στο σχηματισμό κατάλληλου συνόλου από ερωτήματα. Έτσι, θεωρώντας το ABox ως μία σχεσιακή βάση δεδομένων, εκμεταλλευόμαστε τις βελτιστοποιήσεις του πολυμελετημένου

προβλήματος απάντησης σε ερώτηση σε σχεσιακές βάσεις, κάτι πολύ σημαντικό για την απόδοση. Επίσης, η εκθετική πολυπλοκότητα της εξαγωγής συμπερασμών αφορά πλέον το μέγεθος του TBox και του ερωτήματος και όχι το ABox όπως στη διαδικασία υλοποίησης, που συνήθως είναι πολύ μεγαλύτερο. Για όλους τους παραπάνω λόγους η επαναγραφή ερωτημάτων ως μέθοδος επίλυσης του προβλήματος απάντησης σε ερώτημα σε βάσεις γνώσης χρησιμοποιείται ευρύτερα από τη μέθοδος υλοποίησης κι έχει γίνει έντονη μελέτη πάνω σε αυτή [Stamou and Chortaras (2017), Trivela et al. (2015)].

Συγκεκριμένα, κατά τη διαδικασία επαναγραφής το αρχικό ερώτημα επαναγράφεται με βάση τα λογικά αξιώματα του TBox, δηλαδή κάθε επαναγραφή προκύπτει από εφαρμογή κάποιου κανόνα στο ερώτημα. Η διαδικασία είναι εξαντλητική, δηλαδή παράγονται όλες οι δυνατές επαναγραφές, εφόσον επιτρέπεται από το TBox. Εν συνεχεία, κάθε στοιχείο του συνόλου επαναγραφών, ερωτάται ως SQL-ερώτημα πάνω στο ABox (το οποίο θεωρείται απλή σχεσιακή βάση). Η ένωση των απαντήσεων στα ερωτήματα αυτά αποτελεί την απάντηση του αρχικού ερωτήματος. Αν το σύνολο ερωτημάτων είναι πλήρες, τότε η απάντηση που δίνει η διαδικασία είναι βέβαιη. Δυστυχώς όμως, αυτό δεν είναι πάντα δυνατό. Σε κάποιες περιγραφικές λογικές με υψηλή εκφραστικότητα, υπάρχει ο κίνδυνος να απειρίζεται το σύνολο επαναγραφών (με τη μορφή συζευτκικών ερωτημάτων), όπως φαίνεται στο παρακάτω παράδειγμα.

Παράδειγμα 2.1.2. Έστω ένα $\mathcal{EL}$-TBox $\mathcal{T}$:

$$\mathcal{T} = \{\exists \text{ hasParent.Human} \sqsubseteq \text{Human}\},$$

το οποίο εκφράζει ότι όποιος έχει γονιό άνθρωπο είναι άνθρωπος. Έστω επίσης το συζευκτικό ερώτημα q:

$$q(x) = \{\text{Human}(x)\}$$

Τα στάδια της επαναγραφής ερωτημάτων είναι τα εξής:

$$q_1(x) = \{\text{hasParent}(x, y), \text{Human}(y)\}$$

$$q_2(x) = \{\text{hasParent}(x, y), \text{hasParent}(y, z), \text{Human}(z)\}$$

...

Παρατηρούμε ότι η διαδικασία απειρίζεται διότι παράγονται συνεχώς νέα ερωτήματα.

Για το λόγο, γενικεύουμε τη μορφή των επαναγραφών στη μορφή κανόνων datalog [Bienvenu et al. (2014)], οι οποίοι μπορούν να αναπαραστήσουν και αναδρομές. Στην περιγραφική λογική $\mathcal{EL}$, που είναι η ευρύτερη περιγραφική λογική, κάθε ερώτημα να μπορεί να επαναγραφεί σε ένα πεπερασμένο σύνολο datalog επαναγραφών [Hansen et al. (2015), Trivela et al. (2015)]. Datalog ερωτήματα δεν υποστηρίζονται πάντα σε σχεσιακές βάσεις δεδομένων. Για παράδειγμα, αναδρομικά datalog ερωτήματα όπως αυτό που θα προέκυπτε στην περίπτωση του παραδείγματος, απαιτεί κάποια επαγωγική βάση δεδομένων (inductive database) ώστε να απαντηθεί πλήρως [Lutz et al. (2009), Trivela et al. (2015)].

Τέλος, σημειώνεται ότι σε λιγότερο εκφραστικές περιγραφικές λογικές η διαδικασία επαναγραφής δίνει πεπερασμένο σύνολο επαναγραφών σε μορφή συζευτκικών ερωτημάτων κι επομένως βέβαιη απάντηση, ενώ ταυτόχρονα είναι ιδιαίτερα αποτελεσματική. Η περιγραφική λογική που σχεδιάστηκε κατεξοχήν για να έχει τις παραπάνω ιδιότητες είναι η DL-Lite η οποία θα παρουσιαστεί εκτενώς στη συνέχεια.

2.1.3 Συσχετιστικά νευρωνικά δίκτυα

Ένα συσχετιστικό νευρωνικό δίκτυο είναι η δομική μονάδα ενός συστήματος μηχανικής μάθησης για την αυτόματη εξαγωγή συμπερασμών πάνω σε μία οντολογία. Τα συστήματα αυτά θεωρούν το ABox ως ένα γράφο στον οποίο οι κόμβοι είναι τα άτομα του κόσμου τα οποία έχουν διάφορες ετικέτες για τις έννοιες που ανήκουν καθώς επίσης συνδέονται με ακμές που δηλώνουν τη συμμετοχή τους σε κάποιο ρόλο. Με βάση λοιπόν αυτο το γράφο, τα συσχετιστικά νευρωνικά δίκτυα προσπαθούν να προβλέψουν νέα γνώση, δηλαδή νέες ακμές κι ετικέτες στο γράφο. Υπάρχουν διάφορες μέθοδοι που που μελετώνται για το σκοπό αυτό, με τις δύο βασικότερες να βασίζονται στην εξαγωγή προτύπων (mining observable patterns) στο γράφο και στα μοντέλα κρυφών χαρακτηριστικών (latent feature model). Για μία ολοκληρωμένη εικόνα των μεθόδων που χρησιμοποιούνται μπορεί κανείς να παραπεμφθεί στην ανασκόπηση Nickel et al. (2015).

Ουσιαστικά, τα συσχετιστιά νευρωνικά δίκτυα εκπαιδεύονται πάνω σε ένα σγκεκριμένο ABox και προσπαθούν να εξάγουν νέα γνώση βασιζόμενα στην ήδη υπάρχουσα. Δεν λαμβάνουν υπόψιν το σύνολο λογικών κανόνων της οντολογίας επομένως δεν εισάγουν ”νέου είδους” γνώση, η οποία δεν εμφανίζεται προηγουμένως στο γράφο. Το γεγονός αυτό είναι κάπως περιοριστικό αλλά από την άλλη κατευθύνει σε εξαγωγή συμπερασμών με μεγαλύτερη συχνότητα εμφάνισης και άρα κατά κάποιο τρόπο σημασία.

2.2 Δημιουργία συνόλου δεδομένων εκπαίδευσης στη μηχανική μάθηση

Το πρόβλημα προς επίλυση, δηλαδή η δημιουργία ενός συνόλου δεδομένων από έξυπνα επιλεγμένα ABoxes σε σχέση με κάποιο ερώτημα, μπορεί να θεωρηθεί ως ”ποιοτική” δειγματοληψία από τον κόσμο όλων των πιθανών στιγμιοτύπων (ABoxes) που δίνουν κάποια απάντηση στο ερώτημα. Διαφορετικά, θα μπορούσαμε να πούμε ότι αναζητούμε μία κατανομή δειγματοληψίας, αντιπροσωπευτική των στιγμιοτύπων του κόσμου που μπορεί να προκύψουν στην πραγματικότητα, ώστε να καλύψουμε όσο το δυνατόν καλύτερα τις περιπτώσεις που προκύπτουν και να εκπαιδεύσουμε ευρύτατα το νευρωνικό δίκτυο. Ουσιαστικά, το πλαίσιο στο οποίο θέτουμε με αυτό τον τρόπο το πρόβλημα, είναι το πλαίσιο της Πιθανώς Προσσεγγιστικά Σωστής μάθησης (Probably Approximately Correct (PAC) learning) [Denis (1998)], κατά το οποίο σκοπός είναι η μείωση του λάθους γενίκευσης. Η εξαγωγή της δυσκολίας για ένα ABox, όπως εισάγεται στην παρούσα εργασία, χαρακτηρίζει τα στοιχεία του κόσμου ως προς ένα κόστος αλλά και ένα σύνολο από συμπερασμούς, έτσι ώστε επιλογή ενός συνόλου βάση αυτής, να είναι περισσότερο αντιπροσωπευτικό του κόσμου από ότι μία τυχαία επιλογή, κι έτσι να μαθαίνει καλύτερα και γρηγορότερα το σύστημα μηχανικής μάθησης. Για τη δυσκολία ενός ABox, μπορούν να λυθφούν υπόψιν και άλλες μετρικές, όπως το πλήθος των ατόμων, που να την καθιστούν ακόμα καταλληλότερη μετρική για το σκοπό αυτό.

Σημειώνεται επίσης, ότι στην παρούσα εργασία, ασχοληθήκαμε κυρίως με δημιουργία επιλεγμένων δειγμάτων (ABoxes) τα οποία χρησιμοποιήθηκαν στην εκπαίδευση, με δεδομένο μόνο ένα TBox. Το πλαίσιο αυτό είναι ευκολότερο από το διαφορετικό αλλά και πιο ρεαλιστικό πλαίσιο όπου δίνονται κάποιες οντολογίες μαζί με ένα TBox, και πρέπει η γνώση των οντολογιών αυτών

να συμπεριλαμβάνεται σε όλα τα δείγματα. Δηλαδή, στην πιο ρεαλιστική περίπτωση, τίθεται και το πρόβλημα εκμάθησης γνώσης, υπό τον περιορισμό κάποιων ABoxes.

2.3 Περιγραφική λογική συστήματος

Η περιγραφική λογική που επιλέχθηκε για την ανάπτυξη του συστήματός μας είναι η DL-Lite, η οποία συνδιάζει απλότητα και αποδοτικότητα. Γενικά, υπάρχει ένα trade-off μεταξύ περιγραφικότητας και αποδοτικότητας στις περιγραφικές λογικές, όπως έχει ήδη αναφερθεί. Σε αυτό το πλαίσιο, η DL-Lite δεν είναι από τις πιο εκφραστικές, αλλά είναι σχεδιασμένη έτσι ώστε να καλύπτει τις βασικές γλώσσες οντολογιών και παράλληλα να διατηρεί χαμηλή πολυπλοκότητα. Για την ακρίβεια, είναι σχεδιασμένη με βάση το πρόβλημα απάντησης σε ερώτημα, εξού και ορισμένα από τα χαρακτηριστικά της που θα αναλυθούν παρακάτω. Για το λόγο αυτό, επιλέχθηκε ως η περιγραφική λογική του συστήματός μας.

Πέρα από τη DL-Lite, μελετήθηκε και η $\mathcal{EL}$ ως επιλογή για το σύστημά μας. Η $\mathcal{EL}$ είναι η περιγραφική λογική στην οποία βασίζεται το OWL 2 EL προφίλ και παρέχει υψηλή περιγραφική ισχύ [Baader et al. (2005), Baader et al. (2008)]. Η υψηλή αυτή εκφραστικότητα εισήγαγε παραπάνω σημεία που απαιτούσαν έντονη μελέτη, όπως ο περιορισμός των κύκλων ή γενικότερα η διαχείριση συνεχώς αυξανόμενου μεγέθους ενός ερωτήματος στην επαναγραφή ερωτημάτων. Παρόλαυτα, σαν μελλοντική εργασία, η παρούσα μελέτη μπορεί να επεκταθεί στην $\mathcal{EL}$ ή σε κάποια άλλη πιο περιγραφική λογική από τη DL-Lite.

2.3.1 Περιγραφική λογική DL-Lite

Στην υποενότητα γίνεται μία παρουσίαση και σχολιασμλός της περιγαφικής λογικής DL-Lite με βάση το άρθρο Calvanese et al. (2007). Αποδεικνύεται ότι στη DL-Lite τα συνήθη προβλήματα που απαιτούν εξαγωγή συμπερασμών, όπως ο έλεγχος ικανοποιησιμότητας *(satisfiability)* και υπαγωγής *(subsumption)* λύνονται πολυωνυμικά ως προς το μέγεθος του TBox, όπως επίσης το πρόβλημα απάντησης σε ερώτημα λύνεται πολυωνυμικά ως προς το μέγεθος του ABox, ή αλλιώς το πλήθος των ατόμων. Ακόμα, το πρόβλημα απάντησης σε συζευκτικά ερωτήματα λύνεται στη χειρότερη περίπτωση εκθετικά ως προς το μέγεθος του TBox και του ερωτήματος, δηλαδή όχι χειρότερα από ότι στις σχεσιακές βάσεις. Κάτι τέτοιο είναι πολύ σημαντικό στον κόσμο των περιγραφικών λογικών, όπου οι εκθετικές πολυπολοκότητες είναι οι πιο συνηθισμένες. Αυτός είναι κι ο βασικός λόγος που εμποδίζει τις βάσεις γνώσης να εξαπλωθούν και να αντικαταστήσουν τις σχεσιακές βάσεις.

Όμως, το σημαντικότερο χαρακτηριστικό της DL-Lite είναι ότι η ανάλυση των κανόνων του TBox, κατά την εξαγωγή συμπερασμών, ανεξαρτητοποιείται από το ABox και αρκεί η ανάλυση να εφαρμοστεί μόνο στο TBox. Έτσι, μπορεί κανείς να εκμεταλλευτεί τις βελτιστοποιήσεις των σχεσιακών βάσεων για την εξαγωγή απαντήσεων. Ουσιαστικά, αναφερόμαστε στη διαδικασία της επαναγραφής ερωτημάτων κατά την οποία γίνεται επέκταση του αρχικού ερωτήματος σε ένα σύνολο από ερωτήματα που προκύπτουν από το αρχικό, με εξαντλητική εφαρμογή των λογικών κανόνων του TBox. Μετά από αυτή τη διαδικασία, αρκεί να απαντηθεί το σύνολο αυτό από ερωτήματα σε περιβάλλον σχεσιακών βάσεων. Το παρακάτω παράδειγμα δείχνει μία πολύ απλή περίπτωση.

Παράδειγμα 2.3.1. Έστω το TBox $\mathcal{T} = \{A \sqsubseteq B, \exists R \sqsubseteq B, C \sqsubseteq E\}$, το ABox $\mathcal{A} = \{A(a), B(b), C(c), R(d, e)\}$ και το ερώτημα $Q(x) \leftarrow B(x)$. Επαναγράφοντας το Q, δηλαδή χρησιμοποιώντας τους δύο πρώτους λογικούς κανόνες από το $\mathcal{T}$, προκύπτουν τα ερωτήματα $\{Q(x) \leftarrow B(x), Q_1(x) \leftarrow A(x), Q_2(x) \leftarrow R(x, y)\}$ τα οποία ουσιαστικά έχουν συμπεριλάβει όλο το σχετικό με το Q reasoning. Κάνοντας τα παραπάνω ερωτήματα ως SQL ερωτήματα στο , προκύπτει η πλήρης και ορθή απάντηση $\{a, b, d\}$.

2.3.2 Σύνταξη

Ακολουθεί παρουσίαση της σύνταξης των κανόνων, των εννοιών και των ρόλων που επιτρέπονται στη DL-Lite και που καθορίζουν την εκφραστικότητά της. Για την ακρίβεια, η DL-Lite είναι μία οικογένεια περιγραφικών λογικών και συγκεκριμένα των DL-Lite$_{core}$, DL-Lite$_R$, DL-Lite$_F$. Οι δύο τελευταίες αποτελούν μικρές επεκτάσεις της πρώτης. Επομένως, αρχικά επικεντρωνόμαστε στη DL-Lite$_{core}$.

2.3.2.1 DL-Lite$_{core}$

Οι έννοιες στη DL-Lite$_{core}$ ορίζονται ως εξής:

$$R ::= P \mid P^- \qquad B ::= A \mid \exists R$$
$$C ::= B \mid \neg B \qquad E ::= R \mid \neg R$$

όπου το A δηλώνει μία ατομική έννοια (atomic concept) και το P ένα (ατομικό) ρόλο (atomic role). Το R δηλώνει ένα βασικό ρόλο (basic role), δηλαδή είτε ένα ατομικό ρόλο, είτε έναν ανάστροφο (inverse role). Το δηλώνει μία βασική έννοια (basic concept), η οποία μπορεί να είναι είτε μία ατομική έννοια είτε μία έννοια της μορφής $\exists R$, δηλαδή είτε δήλωση υπαρξιακού ρόλου χωρίς περιορισμό είτε που εμπλέκει κάποιο ανάστροφο ρόλο. Το C δηλώνει μία (γενικευμένη) έννοια (general concept). Επιπλέον εισάγωνται οι έννοιες $\bot$ και $\top$ που δηλώνουν το κενό *(bottom)* και το ολόκληρο *(top)* σύνολο ατόμων του κόσμου αντίστοιχα.

Στη συνέχεια, ορίζεται η μορφή αξιωμάτων του TBox στη DL-Lite$_{core}$:

$$B \sqsubseteq C \qquad \textit{ισχυρισμός υπαγωγής}$$

Σχολιάζοντας τα παραπάνω, το αξίωμα $C_3 \sqsubseteq C_1 \sqcap C_2$ μπορεί να γραφεί και ως $\{C_3 \sqsubseteq C_1, C_3 \sqsubseteq C_2\}$ οπότε αυτή η περίπτωση καλύπτεται από την παραπάνω σύνταξη. Τονίζεται, επίσης, ότι άρνηση επιτρέπεται μόνο στις βασικές έννοιες και ότι δεν εμφανίζεται η ύπαρξη ένωσης στα αριστερά αξιώματος. Αξώματα της μορφής $B_1 \sqcup B_2 \sqsubseteq C$ μπορούν να εκφραστούν ισοδύναμα ως $\{B_1 \sqsubseteq C, B_2 \sqsubseteq C\}$. Επίσης, η λογική πράξη της ένωσης αποκλείεται ουσιαστικά από τη δεξιά πλευρά των αξιωμάτων διότι αν σε ένα TBox υπάρχουν τέτοια αξιώματα, η εφαρμογή αυτών σε ένα ABox μπορεί να προσθέτει συνεχώς νέους ισχυρισμούς και να αυξάνει συνεχώς το μέγεθός του.

Τέλος, όσο για τους ισχυρισμούς σε ένα DL-Lite ABox, θα έχουν τη μορφή:

$$A(a), P(a, b) \qquad \textit{ισχυρισμός ιδιότητας μέλους}$$

όπου a, b είναι σταθερές και ουσιαστικά συμβολίζουν τα άτομα της οντολογίας. Οι παραπάνω ισχυρισμοί ιδιότητας μέλους δηλώνουν ότι το a είναι ένα στιγμιότυπο της βασικής έννοιας A και ότι το ζεύγος ατόμων που δηλώνεται από (a, b) είναι στιγμιότυπο του ρόλου P.

Όσο για τη σύνταξη των ερωτημάτων που επιτρέπει η DL-Lite, είναι πολύ πιο ισχυρή από τη σύνταξη ερωτημάτων σε πιο εκφραστικές περιγραφικές λογικές, όπου συνήθως ερωτάται μόνο η συμμετοχή ενός ατόμου σε μία έννοια ή σε ένα ρόλο. Συγκεκριμένα, στη DL-Lite επιτρέπεται οποιοδήποτε συζευκτικό ερωτήμα, ανεξάρτητα από την πολυπλοκότητά του. Η DL-Lite φροντίζει να διατηρεί ή να μειώνει την πολυπλοκότητα των ερωτημάτων αυτών κατά την επαναγραφή τους, διαδικασία με την οποία σχετίζεται απολύτως, όπως εξηγήθηκε και παραπάνω. Συγκεκριμένα, αξιώματα της μορφής $\exists R.A \sqsubseteq B$ αποκλείονται στη DL-Lite για το λόγο ότι κατά την επαναγραφή, με χρήση τέτοιου αξιώματος, αυξάνεται το μέγεθος του ερωτήματος. Αυτό μπορεί να δημιουργεί άπειρως μεγάλα ερωτήματα, κάτι μη συμβατό με τους σκοπούς της DL-Lite. Αποκλείοντας τα λοιπόν, ένα ερώτημα με επαναγραφή μπορεί μόνο να διατηρήσει (αν εφαρμόζονται μόνο κανόνες ξετυλίγματος (unfolding)) ή να μειώσει (αν εφαρμοστεί κάποιος κανόνας συρρίκνωσης (shrinking)) το μέγεθός του. Επιπλέον, αξιώματα της μορφής $A \sqsubseteq \exists R.B$ επιτρέπονται στη DL-Lite καθώς υπάρχει ισοδύναμο σύνολο αξιωμάτων σε DL-Lite που μπορούν να τα αντικαταστήσουν, όπως αναφέρεται στο [efficient query answering for owl2]:

$$\{A \sqsubseteq \exists R.B\} \iff \{A \sqsubseteq \exists P_1, \exists P_1^- \sqsubseteq B, P_1 \sqsubseteq R\}$$

Το γεγονός ότι αυτά τα αξιώματα είναι αποδεκτά στη DL-Lite μπορεί και διαισθητικά να ερμηνευτεί, καθώς με βάση το σκοπό δημιουργίας της DL-Lite, αξιώματα που δεν δημιουργούν μη διαχειρίσιμες επαναγραφές (για παράδειγμα άπειρες στο μέγεθος), δεν έχουν λόγο να μην είναι αποδεκτά.

Παράδειγμα 2.3.2. Έστω το TBox $\mathcal{T} = \{A \sqsubseteq B, \exists R \sqsubseteq B, C \sqsubseteq \exists R.D\}$ και το ερώτημα $Q(x) \leftarrow B(x) \wedge R(x, y) \wedge D(y)$. Επαναγράφοντας το Q, χρησιμοποιώντας με τη σειρά έναν από τους λογικούς κανόνες κάθε φορά από το $\mathcal{T}$, προκύπτουν αντίστοιχα τα ερωτήματα:

$$\begin{gathered} Q_1(x) \leftarrow A(x) \wedge R(x, y) \wedge D(y) \\ Q_2(x) \leftarrow R(x, z) \wedge R(x, y) \wedge D(y) \xrightarrow{condense} Q_2(x) \leftarrow R(x, y) \wedge D(y) \\ Q_3(x) \leftarrow B(x) \wedge C(x) \end{gathered}$$

Γίνεται φανερό ότι, έχοντας όλων των ειδών κανόνες στο $\mathcal{T}$, σε καμία περίπτωση δεν επεκτείνεται το ερώτημα κατά την επαναγραφή. Αρνήσεις ή αναστροφοι ρόλου δεν επηρεάζουν την επέκταση του ερωτήματος.

2.3.2.2 DL-Lite$_R$

Στην παρούσα εργασία χρησιμοποιήθηκε η DL-Lite$_R$, γι' αυτό και την παρουσιάζουμε. Επεκτείνει ελαφρώς τη DL-Lite$_{core}$ και συγκεκριμένα, συμπεριλαμβάνει στο TBox επιπλέον αξιώματα της μορφής:

$$R \sqsubseteq E$$

όπου R, E ορίστηκαν παραπάνω. Ουσιαστικά, εισάγει αξιώματα ξετυλίγματος μεταξύ ρόλων. Στη DL-Lite$_R$ βασίζεται το προφίλ της OWL 2 QL, το οποίο χρησιμοποιείται για εξαγωγή πλήρων και ορθών απαντήσεων σε ερωτήματα σε λογαριθμικό χώρο (LOGSPACE) και πολυωνυμικό χρόνο σε σχέση με τα άτομα στην οντολογία.

2.4 Σύστημα SyGENiA

Το προγραμματιστικό εργαλείο SyGENiA χρησιμοποιήθηκε για την εξαγωγή χρήσιμων ερωτημάτων μίας οντολογίας, τα οποία αντιπροσωπεύουν την οντολογία αλλά και αποτελούν μία βάση για κάθε πιθανό ερώτημα. Το θεωρητικό υπόβαθρο του SyGENiA παρουσιάζεται συνοπτικά παρακάτω [Grau and Stoilos (2011)].

2.4.1 Γενικά

Το αντικείμενο της μελέτης αυτής είναι να χαρακτηρίσει ένα Semantic Web Reasoner ως προς την πληρότητα των απαντήσεων που δίνει σε διάφορα ερωτήματα σε περιγραφική λογική $\mathcal{EL}$ ή λιγότερο εκφραστικές λογικές, όπως η DL-Lite. Κάτι τέτοιο είναι πολύ χρήσιμο καθώς πολλοί reasoners "θυσιάζουν" την πληρότητα των απαντήσεών τους με σκοπό τη χρονική κλιμάκωση, δηλαδή επιλέγουν να απαντήσουν ταχύτερα ένα ερώτημα χωρίς να δώσουν ολόκληρο το σύνολο απαντήσεων. Επίσης, το να ελεχθεί η πληρότητα εξαντλητικά πάνω σε μία οντολογία, για κάθε τυχόν ερώτημα, ισοδυναμεί με το να εξαχθούν όλα τα συμπεράσματα, οπότε δεν κερδίζεται κάτι από άποψη χρόνου. Ακόμα και ο εξαντλητικός έλεγχος, είναι εφικτό να πραγματοποιηθεί σε συγκεκριμένες μόνο οντολογίες κατά το testing οι οποίες μπορεί να μην είναι καθόλου αντιπροσωπευτικές μίας τυχαίας οντολογίας που θα προκύψει στη συνέχεια στην πράξη. Θα ήταν εύλογο λοιπόν να γνωρίζει κανείς κατά πόσο πλήρεις είναι οι απαντήσεις που δίνει ένα σύστημα, εξ' αρχής, για μία συγκεκριμένη οντολογία και ανεξάρτητα από το ερώτημα που δίνεται στη συνέχεια.

2.4.2 Σύνολο δεδομένων ελέγχου πληρότητας

Παρουσιάζεται μία μέθοδος παραγωγής test-data πάνω στα οποία μπορεί να ελεχθεί κάτι τέτοιο, ανεξαρτήτως από τα ερωτήματα αλλά και το ABox που θα τεθούν κάθε φορά. Test-data παράγονται για διάφορες οντολογίες πάνω στα οποία δοκιμάζεται ο reasoner. Συγκεκριμένα, για κάθε οντολογία, λαμβάνοντας υπόψιν μόνο το TBox, παράγεται ένα σύνολο ερωτημάτων και στη συνεχεία, για καθένα ερώτημα του συνόλου αυτού παράγεται ένα testing-base από ABoxes. Το testing-base είναι ελάχιστο και εξαντλητικό και αποτελείται από ελάχιστα ABoxes τα οποία μπορούν να δώσουν απάντηση στο αντίστοιχο ερώτημα. Είναι ελάχιστο γιατί δεν περιέχει ισομορφικά ABoxes κι εξαντλητικό γιατί περιέχει ένα ABox για κάθε στοιχείο του συνόλου των justifications που αντιστοιχεί σε κάθε ένα διάνυσμα απάντησης ($\vec{a} \in \text{cert}(q, \mathcal{O})$, όπου q το ερώτημα και $\mathcal{O}$ η οντολογία). Επίσης λέγοντας ελάχιστα ABoxes εννοούμε ABoxes που δεν έχουν περιττούς ισχυρισμούς. Τέλος, αναφέρεται ότι σε περιγραφικές λογικές, όπως η $\mathcal{EL}$, που επιτρέπουν μεταβατικά αξιώματα, είναι πιθανόν να μην υπάρχει εξαντλητικό testing-base και τότε, για να διατηρήσουμε την εξαντλητικότητα, θα πρέπει να περιορίσουμε τα άτομα της οντολογίας *(n-exhaustive testing-base)*. Τέτοια προβλήματα δεν εμφανίζονται στη DL-Lite η οποία χρησιμοποιείται στην παρούσα εργασία. Ακολουθεί ένα πολύ απλό παραδείγμα, που δίνει καλύτερη εικόνα για τους παραπάνω όρους:

Παράδειγμα 2.4.1. Έστω $\mathcal{T} = \{\exists R.\top \sqsubseteq A, S \sqsubseteq R\}$, και $Q(x) \leftarrow A(x) \wedge R(x, y)$. Τα ακόλουθα ABoxes αποτελούν testing-units, δηλαδή τα συστατικά του testing-base B $= \{\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_8\}$, το οποίο είναι ελάχιστο κι εξαντλητικό:

$$\mathcal{A}_1 = \{R(a, b)\} \qquad \mathcal{A}_2 = \{A(a), R(a, b)\}$$

$\mathcal{A}_3 = \{R(a,a)\}$ $\mathcal{A}_4 = \{A(a), R(a,a)\}$
$\mathcal{A}_5 = \{S(a,b)\}$ $\mathcal{A}_6 = \{A(a), S(a,b)\}$
$\mathcal{A}_7 = \{S(a,a)\}$ $\mathcal{A}_8 = \{A(a), S(a,a)\}$

Γίνεται φανερό και από το παραπάνω παράδειγμα, ότι δεδομένου ABox $\mathcal{A}$, οποιοδήποτε διάνυσμα απάντησης $\vec{a} \in \text{cert}(Q, \mathcal{T}, \mathcal{A})$, θα προκύπτει από κάποιο υποσύνολο του $\mathcal{A}$ το οποίο είναι ισομορφικό με κάποιο από τα ABoxes στο B. Έτσι, ένας αλγόριθμος συζευκτικών ερωτημάτων που απαντά σωστά το $\text{cert}(q, \mathcal{T}, \mathcal{A}_i)$ για κάθε $\mathcal{A}_i \in$ B, θα απαντά $\text{cert}(q, \mathcal{T}, \mathcal{A})$ για οποιοδήποτε ABox $\mathcal{A}$. Επίσης, αν δώσει πλήρη απάντηση σε συγκεκριμένο σύνολο ερωτημάτων, το καθένα πάνω στο αντίστοιχο testing-base, τότε είναι πλήρης για οποιοδήποτε ερώτημα. Επομένως, τα test-data αποτελούνται από ερωτήματα, καθένα από τα οποία συνδέεται με ένα testing-base B. Η ισχύς του θεωρήματος καθώς και ειδικές περιπτώσεις αποδεικνύονται στο άρθρα της έρευνας αυτής. Το SyGENiA λαμβάνει ως είσοδο μία οντολογία (κατά βάση ένα TBox) και παράγει τα test-data σε δύο φάσεις:

- Παραγωγή του συνόλου ερωτημάτων (*Query Testing Base - QTB*)
- Παραγωγή του testing-base από ABoxes για κάθε ερώτημα στο QTB

Τα ερωτήματα του πρώτου βήματος αποτελούν το σύνολο ερωτημάτων εισόδου στον αλγόριθμο παραγωγής του DatAbox για μία οντολογία, όπως θα επεξηγηθεί αναλυτικά παρακάτω.

2.4.3 Παραγωγή των ερωτημάτων

Στο σημείο αυτό θα παρουσιαστεί η ανάλυση που γίνεται για την επιλογή του συνόλου ερωτημάτων που αναφέρθηκε προηγουμένως. Γενικά, η περιοχή της αυτόματης παραγωγής ερωτημάτων automated query generation αποτελεί ενεργό ερευνητικό πεδίο, κυρίως όμως με σκοπό τον έλεγχο της απόδοσης εξαγωγής απαντήσεων και όχι σε σχέση με την πληρότητα. Η αυτόματη επιλογή του συνόλου ερωτημάτων για έλεγχο της πληρότητας σε σχέση με ένα TBox και ανεξάρητα από το εκάστοτε ABox, αποτελεί πρωτεύον σημείο στον έλεγχο πληρότητας ενός συστήματος. Το σύνολο αυτό αποτελείται από συζευκτικά ερωτήματα έτσι ώστε αν ένα σύστημα είναι (q, T)-πλήρες για κάθε ερώτημα q στο σύνολο αυτό, τότε είναι (q', T)-πλήρες για κάθε τυχαίο ερώτημα q'. Κάτι τέτοιο είναι πολύ ισχυρό, καθώς περιορίζει τον έλεγχο ενός συστήματος για πληρότητα σε ένα μόνο σύνολο από ερωτήματα, από τα άπειρα ερωτήματα που μπορούν να γίνουν σε μία οντολογία.

Ουσιαστικά, τα ερωτήματα αυτά θα πρέπει να επιλέγονται ώστε να εντοπίζουν μη πληρότητα, δηλαδή παράβλεψη κάποιου λογικού κανόνα από το TBox. Γιατί, όταν ένας reasoner είναι μη πλήρης, θα δώσει λιγότερες απαντήσεις από ότι αν ήταν πλήρης το οποίο σημαίνει ότι δεν έχει "διαβάσει" κάποιον κανόνα που εισάγει και άλλες οντότητες του κόσμου στην απάντηση.

Παράδειγμα 2.4.2. Δεδομένου TBox $\mathcal{T} = \{A \sqsubseteq B, B \sqsubseteq C\}$, query $q(x) = A(x)$ και ABox $\mathcal{A} = \{A(a), B(b), C(c)\}$, ένας πλήρης reasoner r_1 απαντά $ans_{r_1}(q, \mathcal{T}, \mathcal{A}) \supseteq \text{cert}(q, \mathcal{T}, \mathcal{A}) = \{a, b, c\}$ ενώ ένας μη πλήρης r_2 απαντά $ans_{r_2}(q, \mathcal{T}, \mathcal{A}) \subseteq \text{cert}(q, \mathcal{T}, \mathcal{A}) = \{a, b, c\}$. Μία πιθανή απάντηση για τον r_2 θα ήταν $ans_{r_2}(q, \mathcal{T}, \mathcal{A}) = \{a, b\}$ όπου ουσιαστικά παραλείπει το τρίτο αξίωμα του $\mathcal{T}$.

Επομένως, είναι λογικό το σύνολο ερωτημάτων αυτό να απαιτεί ενεργοποίηση όλων των κανόνων του TBox, ώστε αν υπάρχει παράβλεψη κάποιου κανόνα από έναν reasoner, τότε να μη μπορεί

να απαντήσει πλήρως το αντίστοιχο ερώτημα, κι έτσι να φανερώνει ότι δεν είναι πλήρης. Αυτό φροντίζει να πετυχαίνει και το σύνολο των ερωτημάτων που παράγει το SyGENiA. Στο [στοιλοσ κουιρι] αναφέρεται αναλυτικά ο τρόπος παραγωγής τους και η θεωρία πίσω από αυτόν. Στη συνέχεια γίνεται παρουσίαση και σχολιασμός της ανάλυσης αυτής.

Η ανάλυση περιλαμβάνει οποιοδήποτε TBox σε DL-Lite και $\mathcal{EL}$ που τηρεί κάποιες προϋποθέσεις ώστε να υπάρχει πάντα πεπερασμένο QTB, εκτός του κενού TBox οπότε δεν υπάρχει QTB. Με βάση την παραπάνω εισαγωγή για το σκοπό δημιουργίας του QTB είναι προφανές ότι δεν υπάρχει νόημα σχηματισμού του συνόλου για κενό TBox. Αρχικά, όταν προκύπτουν κύκλοι από τα αξιώματα του TBox, ορίζονται αλγόριθμοι που ξετυλίγουν τον κύκλο μέχρι συγκεκριμένο βάθος. Γενικά αποφεύγονται Tboxes που περιέχουν κύκλους και για το λόγο αυτό εισάγονται έννοιες όπως η *ασθενής ακυκλικότητα (weak acyclicity).* Η ιδιότητα αυτή εξασφαλίζει πολυωνυμικό όριο σε πλήθος και μέγεθος των ερωτημάτων στο QTB. Για να οριστεί η ασθενής ακυκλικότητα πρέπει να εισαχθεί η έννοια του *γράφου συσχετίσεων (dependency graph).*

Ορισμός 2.4.1. Έστω $\mathcal{T}$ ένα $\mathcal{L}$-TBox, με $\mathcal{L} \in$ {DL-Lite, $\mathcal{EL}$}. Ο *γράφος συσχετίσεων* για το $\mathcal{T}$ είναι ο μικρότερος γράφος με τα εξής χαρακτηριστικά:

- *Κόμβοι:* Ένας κόμβος u_A για κάθε ατομική έννοια *(atomic concept)* A και κόμβοι $u_{R/1}$ και $u_{R/2}$ για κάθε ατομικό ρόλο *(atomic role)* στο $\mathcal{T}$.
- *Ακμές:* Για κάθε ατομικό ρόλο R και A, B ατομικές έννοιες ή $\top$, έστω $[A] = A$, $[\exists R.B] = R/1$ και $[\exists R^-.B] = R/2$.
  - Αν $\mathcal{T} \in$ DL-Lite, τότε ο γράφος περιέχει τις ακόλουθες ακμές για κάθε θετικό GCI $(B \sqsubseteq C) \in \mathcal{T}$:
    - Μία ακμή $u_{[B]} \rightarrow u_{[C]}$
    - Μία ειδική ακμή *(special edge)* $u_{[B]} \xrightarrow{*} u_{R/2}$ αν $C = \exists R.\top$.
    - Μία ειδική ακμή $u_{[B]} \xrightarrow{*} u_{R/1}$ αν $C = \exists R^-.\top$.
  - Αν $\mathcal{T} \in \mathcal{EL}$, έχει τις ακμές $u_{[A]} \rightarrow u_\top$ και $u_{R/1} \rightarrow u_\top$ για κάθε ατομικά A και R, όπως επίσης και τις ακόλουθες ακμές για κάθε $(\prod_{1 \leq i \leq n} B_i \sqsubseteq C) \in \mathcal{T}$ με $n \leq 2$:
    - Τις ακμές $u_{[B_i]} \rightarrow u_{[C]}$
    - Τις ειδικές ακμές $u_{[B_i]} \xrightarrow{*} u_{[D]}$ και $u_{[B_i]} \xrightarrow{*} u_{[R/2]}$ αν $C = \exists R.D$.

Ο γράφος συσχετίσεων είναι *ασθενώς ακυκλικός (weakly acyclic graph)* αν δεν περιέχει κύκλο που περιλαμβάνει μία ειδική ακμή.

Ουσιαστικά, οι ειδικές ακμές αντιπροσωπεύουν υπαρξιακά αξιώματα και σηματοδοτούν πιθανή εισαγωγή νέου ατόμου στον κόσμο της απάντησης. Όπως γίνεται αντιληπτό, κύκλος μεταξύ τέτοιων αξιωμάτων σημαίνει συνεχή παραγωγή νέων ατόμων όσο εφαρμόζονται κανόνες, δηλαδή κατά το ξετύλιγμα.

Παράδειγμα 2.4.3. Έστω το παρακάτω $\mathcal{EL}$-TBox:

$\mathcal{T} =$ {Student $\equiv \exists$takesCourse.Course,

GradCourse $\sqsubseteq$ Course,
GradStudent $\sqsubseteq \exists$takesCourse.GradCourse}

Εδώ, υπάρχει κύκλος στο γράφο συσχετίσεων λόγω της κυκλικής συσχέτισης από το πρώτο αξίωμα το οποίο αντιστοιχεί στο ζεύγος κανονικοποιημένων αξιωμάτων:
{Student $\sqsubseteq \exists$takesCourse.Course, $\exists$takesCourse.Course $\sqsubseteq$ Student}. Παρόλο που υπάρχει κυκλική συσχέτιση (Student $\rightarrow \exists$takesCourse.Course $\rightarrow$ Student), δεν εμπλέκεται ειδική ακμή και άρα το $\mathcal{T}$ είναι ασθενώς κυκλικό. Η χρησιμότητα της ασθενούς ακυκλικότητας θα φανεί στο τέλος της ανάλυσης.

Εν συνεχεία επεξηγείται η έννοια *chase* που είναι μία τροποποιημένη έκδοση της ίδιας έννοιας στις σχεσιακές βάσεις. Η έννοια αυτή είναι πρωτεύουσας σημασίας για την παραγωγή του QTB. Με δεδομένη μία συνεπή $\mathcal{L}$-οντολογία $\mathcal{O} = \mathcal{T} \cup \mathcal{A}$, το chase$_\mathcal{L}(\mathcal{O})$ είναι ένα πιθανά άπειρο δενδρικής μορφής ABox το οποίο κατασκευάζεται βήμα-βήμα από το $\mathcal{A}$ και αντιπροσωπεύει όλα τα μοντέλα της για τους σκοπούς του CQ answering. Δηλαδή ισχύει cert$(q, \mathcal{O})$ = cert$(q,$ chase$_\mathcal{L}(\mathcal{O}))$, όπου με κατάχρηση της ορολογίας, ο όρος cert$(q,$ chase$_\mathcal{L}(\mathcal{O}))$ χρησιμοποιείται για την αναπαράσταση των απαντήσεων στο q που περιλαμβάνουν άτομα μόνο από το $\mathcal{A}$.

Στην τεχνική αναφορά της μελέτης αυτής, αναλύεται η διαδικασία για την κατασκευή του chase. Ο ακριβής τρόπος κατασκευής του chase και συνεπώς του QTB αποτελεί σημαντικό κομμάτι στην παρούσα εργασία καθώς δικαιολογεί την επιλογή του συγκεκριμένου QTB ως σύνολο ερωτημάτων εισόδου στο σύστημα DatAbox. Οι κανόνες κατασκευής *(chase rules)* [Calvanese et al. (2007), Rosati (2007)] δίνονται συνοπτικά στον Πίνακα 2.1.. Σημειώνεται ότι ο κανόνας chr4 εφαρμόζεται μόνο σε DL-Lite ενώ ο cr5 μόνο σε $\mathcal{EL}$. Επίσης, οι κανόνες cr2, cr3 εφαρμόζονται στη DL-Lite μόνο για $A_2 = \top$ και $A_1 \neq \top$ και στην $\mathcal{EL}$ για R ατομικό ρόλο. Γίνεται αντιληπτό ότι με την ύπαρξη υπαρξιακών αξιωμάτων είναι πιθανή η εισαγωγή νέων ανώνυμων ατόμων καθώς και ότι ο cr2 στη DL-Lite δεν προσθέτει νέο ισχυρισμό στο chase.

Με βάση τα παραπάνω, για ασθενώς ακυκλικό TBox μίας συνεπούς οντολογίας $\mathcal{O} = \mathcal{T} \cup \mathcal{A}$, η chase ακολουθία που παράγεται φράσσεται πολυωνυμικά από το μέγεθος του $\mathcal{A}$. Με άλλα λόγια, η ασθενής ακυκλικότητα φράσει το μέγεθος του chase. Επίσης, όπως προαναφέρθηκε, το chase είναι δενδρικής μορφής που σημαίνει ότι κάθε ανώνυμο άτομο είναι συνδεδεμένο με κάποιο ονομασμένο μέσω ενός μονοπατιού από ισχυρισμούς ρόλων. Οι παρατηρήσεις αυτές οδηγούν στους παρακάτω ορισμούς, του άρθρου με σκοπό να καταλήξουμε στο σχηματιστμό του QTB.

Ορισμός 2.4.2. Έστω $\mathcal{O} = \mathcal{T} \cup \mathcal{A}$ μία $\mathcal{L}$-οντολογία για $\mathcal{L} \in$ {DL-Lite, $\mathcal{EL}$}. Ένα μονοπάτι *(path)* του chase$_\mathcal{L}(\mathcal{O})$ είναι ένα υποσύνολο του chase$_\mathcal{L}(\mathcal{O})$ της μορφής $\{A(a)\}$ με ατομικό, ονομασμένο άτομο και $\{A(a)\} \notin \mathcal{A}$ ή της μορφής $\{R_1(a, b_1), ..., R_n(b_{n-1}, b_n), B(b_n)\}$ όπου B ατομικό ή $\top$, a ονομασμένο άτομο και όλα τα b_i ανώνυμα άτομα.

Ορισμός 2.4.3. Έστω $\mathcal{L} \in$ {DL-Lite, $\mathcal{EL}$}, $\mathcal{T}$ ένα $\mathcal{L}$-TBox με $\mathcal{T}_p \subseteq \mathcal{T}$ το υποσύνολο του $\mathcal{T}$ που περιέχει μη αρνητικά GCIs και LHS = $\{C \mid C \sqsubseteq D \in \mathcal{T}_p\}$. Για κάθε $C \in$ LHS, έστω c_1, c_2 άτομα που σχετίζονται μοναδικά με το C και έστω

$$\mathcal{A}_C = \begin{cases} \{A(c_1)\}, & C = A \\ \{A_1(c_1), A_2(c_2)\}, & C = A_1 \sqcap A_2 \\ \{R(c_1, c_2), A(c_2\}, & C = \exists R.A \end{cases}$$

cr1	$A_1 \sqsubseteq A_2 \in \mathcal{T}$, $A_1(a) \in \text{chase}_\mathcal{L}(\mathcal{O})$ και $A_2(a) \notin \text{chase}_\mathcal{L}(\mathcal{O})$, πρόσθεσε το $A_2(a)$ στο $\text{chase}_\mathcal{L}(\mathcal{O})$.
cr2	$A_1 \sqsubseteq \exists R.A_2 \in \mathcal{T}$, $A_1(a) \in \text{chase}_\mathcal{L}(\mathcal{O})$ και δεν υπάρχει b έτσι ώστε $\{R(a,b), A_2(b)\} \in \text{chase}_\mathcal{L}(\mathcal{O})$, πρόσθεσε τα $R(a,a'), A_2(a')$ στο $\text{chase}_\mathcal{L}(\mathcal{O})$.
cr3	$\exists R.A_2 \sqsubseteq A_1 \in \mathcal{T}$, $R(a,b) \in \text{chase}_\mathcal{L}(\mathcal{O})$, $A_2(b) \in \text{chase}_\mathcal{L}(\mathcal{O})$, και $A_1(a) \notin \text{chase}_\mathcal{L}(\mathcal{O})$, πρόσθεσε το $A_1(a)$ στο $\text{chase}_\mathcal{L}(\mathcal{O})$.
cr4	$\exists R_1.\top \sqsubseteq \exists R_2.\top \in \mathcal{T}$, $R_1(a,b) \in \text{chase}_\mathcal{L}(\mathcal{O})$ και δεν υπάρχει c έτσι ώστε $R_2(a,c) \in \text{chase}_\mathcal{L}(\mathcal{O})$, πρόσθεσε το $R_2(a,a')$ στο $\text{chase}_\mathcal{L}(\mathcal{O})$.
cr5	$A_1 \sqcap A_2 \sqsubseteq A \in \mathcal{T}$, $A_1(a) \in \text{chase}_\mathcal{L}(\mathcal{O})$, $A_2(a) \in \text{chase}_\mathcal{L}(\mathcal{O})$ και $A(a) \notin \text{chase}_\mathcal{L}(\mathcal{O})$, πρόσθεσε το $A(a)$ στο $\text{chase}_\mathcal{L}(\mathcal{O})$.

Πίνακας 2.1: Chase rules για $\mathcal{L} \in \{\text{DL-Lite}, \mathcal{EL}\}$. Το άτομο a' είναι νέο και ανώνυμο και οι έννοιες είναι ατομικές ή $\top$. Θεωρείται ότι ο ισχυρισμός $\top(c)$ προστίθεται στο $\text{chase}_\mathcal{L}(\mathcal{O})$ για κάθε (ονομασμένο ή ανώνυμο) άτομο c που προκύπτει στο $\text{chase}_\mathcal{L}(\mathcal{O})$.

όπου είναι ατομικό ή $\top$. Έστω $\mathcal{A}_\mathcal{T} = \bigcup_{C \in LHS} \mathcal{A}_C$. Ένα συζευκτικό ερώτημα q είναι *chase ερώτημα (chase query)* με βάση το $\mathcal{T}$ αν μπορεί να προκύψει από κάποιο μονοπάτι του $\text{chase}_\mathcal{L}(\mathcal{T} \cup \mathcal{A}_\mathcal{T})$ αντικαθιστώντας κάθε άτομο c με μία μεταβλητή x_c, έτσι ώστε x_c είναι διακεκριμένη αν c είναι ονομασμένο.

Τελικά, αποδεικνύεται το παρακάτω κομβικό θεώρημα.

Θεώρημα 2.4.1. Έστω $\mathcal{L} \in \{\text{DL-Lite}, \mathcal{EL}\}$ και έστω $\mathcal{T}$ ένα ασθενώς ακυκλικό $\mathcal{L}$-TBox. Τότε, το σύνολο όλων των chase ερωτημάτων (μοναδικά υπό ισομορφισμούς) είναι ένα QTB για το $\mathcal{T}$ για όλους τους καλώς ορισμένους αλγορίθμους ανάλυσης για την $\mathcal{L}$. Επίσης, το μέγεθος του QTB είναι πολυωνυμικό ως προς το μέγεθος του $\mathcal{T}$.

Το παρακάτω παράδειγμα αναλύει κάθε βήμα της διαδικασίας παραγωγής των chase ερωτημάτων.

Παράδειγμα 2.4.4. Τα ακόλουθα ερωτήματα είναι όλα chase ερωτήματα (μοναδικά υπό ισομορφισμούς) με βάση το TBox $\mathcal{T}$ του Παραδείγματος 1.3.2:

$q_1(x) \leftarrow \{\text{Course}(x)\}$
$q_2(x) \leftarrow \{\text{Student}(x)\}$
$q_3(x) \leftarrow \{\text{takesCourse}(x,y)\}$
$q_4(x) \leftarrow \{\text{takesCourse}(x,y), \text{Course}(y)\}$
$q_5(x) \leftarrow \{\text{takesCourse}(x,y), \text{GradCourse}(y)\}$

Αρχικά το TBox $\mathcal{T}$ σε κανονική μορφή έχει ως εξής:

$\mathcal{T} =$ {Student $\sqsubseteq \exists$takesCourse.Course,
$\exists$takesCourse.Course $\sqsubseteq$ Student,
GradCourse $\sqsubseteq$ Course,
GradStudent $\sqsubseteq \exists$takesCourse.GradCourse}

Επομένως, ισχύει ότι

LHS = {Student, $\exists$takesCourse, GradCourse, GradStudent}

και

$\mathcal{A}_{Student} =$ {Student(a_1)},
$\mathcal{A}_{\exists takesCourse.Course} =$ {takesCourse(b_1, b_2), Course(b_2)}
$\mathcal{A}_{GradCourse} =$ {GradCourse(c_1)}
$\mathcal{A}_{GradStudent} =$ {GradStudent(d_1)}

Τελικά προκύπτει ότι

$\mathcal{A}_{\mathcal{T}} =$ {Student(a_1),
takesCourse(b_1, b_2), Course(b_2),
GradCourse(c_1),
GradStudent(d_1)}

από το οποίο, με εφαρμογή των κανόνων του Πίνακα 2.1, υπολογίζεται το chase$_{\mathcal{L}}(\mathcal{T} \cup \mathcal{A}_{\mathcal{T}})$. Το παρακάτω σύνολο προκύπτει μετά τον πρώτο κύκλο εφαρμογής των chase κανόνων, όπου με u_i συμβολίζονται τα ανώνυμα άτομα:

chase$_{\mathcal{L}}(\mathcal{T} \cup \mathcal{A}_{\mathcal{T}})_1 =$ {
Student(a_1), takesCourse(a_1, u_1), Course(u_1), //*cr2*
takesCourse(b_1, b_2), Course(b_2), Student(b_1), //*cr3*
GradCourse(c_1), Course(c_1), //*cr1*
GradStudent(d_1), takesCourse(d_1, u_2), GradCourse(u_2) //*cr2*
}

Παρατηρούμε ότι αν συνεχίζαμε με εφαρμογή των κανόνων, στον επόμενο κύκλο θα εισάγονταν νέοι ισχυρισμοί, όπως για παράδειγμα οι ισχυρισμοί της πρώτης γραμμής για το άτομο b_1, που προκύπτουν από το Student(b_1). Οι νεοϊσαχθέντες ισχυρισμοί δεν θα άλλαζαν την QTB γιατί εισάγουν ισχυρισμούς ισομορφικούς με ήδη υπάρχοντες. Ούτως ή άλλως όμως, δεν εισάγονται εξαρχής στο chase διότι δεν παράγεται κάποιο νέο μοντέλο απάντησης. Εδώ φαίνεται και η χρησιμότητα της ασθενούς ακυκλικότητας, καθώς ακριβώς τέτοιου είδους "άκακους" κύκλους επιτρέπει. Λαμβάνοντας τους ισχυρισμούς από το chase$_{\mathcal{L}}(\mathcal{T} \cup \mathcal{A}_{\mathcal{T}})$ που ανήκουν στο chase-μονοπάτι, δηλαδή το ακόλουθο σύνολο:

$P =$ { takesCourse(a_1, u_1), Course(u_1),
Student(b_1),
Course(c_1),
takesCourse(d_1, u_2), GradCourse(u_2)}

και θεωρώντας διακριτές μεταβλητές απάντησης κάθε ονομασμένο άτομο, προκύπτει το παραπάνω QTB.

Ουσιαστικά, το σύνολο $\mathcal{A}_C$, για κάθε $C \in$ LHS, είναι οι ισχυρισμοί που προκύπτουν αν υπολίζαμε το rewriting του ατομικού ερωτήματος το οποίο ισούται με τη δεξιά πλευρά κάθε κανόνα του . Δηλαδή, αν σε κάθε ένα από τα $\mathcal{A}_C$ εφαρμοστεί materialization, ο κανόνας που θα ενεργοποιηθεί θα είναι ο C. Επίσης, τα chase-μονοπάτια αντιπροσωπεύουν τις δεξιές πλευρές κάθενός από τους κανόνες του $\mathcal{T}$ με αποτέλεσμα, κάνοντας ένα από αυτά τα ερωτήματα στην οντολογία, θα χρειαστεί να γίνει απαραίτητα reasoning, δηλαδή να ενεργοποιηθεί κάποιος κανόνας από το $\mathcal{T}$. Για παράδειγμα, αν στην οντολογία του Παραδείγματος 1.3.3. ερωτηθεί το q_4, για να δοθεί σωστά η απάντηση από το σύστημα, θα πρέπει να χρησιμοποιηθεί ο πρώτος κανόνα του $\mathcal{T}$ ώστε να συμπεριλάβει και τα άτομα που είναι απλά εγγεγραμμένα ως Student.

Μετά την παραπάνω ανάλυση, αντιλαμβάνεται κανείς γιατί επιλέχθηκε το QTB ως η βάση των ερωτημάτων στο δικό μας σχεδιασμό. Σκοπός του συστήματος που σχεδιάσαμε είναι να εκπαιδεύσει ένα νευρωνικό να κάνει reasoning. Επομένως, θέλουμε να εξασφαλίσουμε ότι κάθε δεδομένο εκπαίδευσης έχει κάποιο νόημα σε σχέση με την οντολογία, δηλαδή το TBox, που ουσιαστικά σημαίνει ότι ενεργοποιεί κάποιο λογικό κανόνα. Το πώς ακριβώς χρησιμοποιείται η βάση αυτή από ερωτήματα θα παρουσιαστεί παρακάτω.

2.5 Αλγόριθμος Rapid

Ο αλγόριθμος Rapid [Chortaras et al. (2011)] έχει κομβική σημασία στην παρούσα εργασία. Λαμβάνοντας ένα TBox, σε DL-Lite$_R$, κι ένα σύνολο από συζευκτικά ερωτήματα, παράγει όλες τις επαναγραφές (rewritings) κάθε ερωτήματος από το σύνολο ερωτημάτων, με βελτιωμένο τρόπο σε σχέση με τους συνήθεις αλγορίθμους επαναγραφής. Το μεγαλύτερο και πιο σύνθετο μέρος του συτήματος αυτής της εργασίας, το οποίο υλοποιήθηκε στην πράξη, αφορά την τροποποίηση του αλγορίθμου Rapid για να εξάγει κι ένα βαθμό δυσκολίας που χαρακτηρίζει την κάθε επαναγραφή, όπως αναφέρθηκε στην εισαγωγή. Αυτό θα αναλυθεί σε επόμενο κεφάλαιο (3), αφού πρώτα παρουσιαστούν τα βασικά σημεία της θεωρίας και της υλοποίησης του αλγορίθμου αυτού.

2.5.1 Γενικά

Ο Rapid αποτελεί ένα βελιτωμένο αλγόριθμο επαναγραφής ερωτημάτων, ο οποίος οφείλει την αποδοτικότητά του στην ελεγχόμενη και οργανωμένη εφαρμογή των κανόνων επίλυσης (resolution rules). Αντί να εφαρμόζεται αδιακρίτως κανόνες επίλυσης στο εισαχθέν ερώτημα, λαμβάνεται υπόψιν η δομή και η σύνταξη του ερωτήματος και πραγματοποιείται μία ελαχιστοποιημένη ακολουθία από επιλύσεις που οδηγεί στο σχηματισμό του συνόλου των επαναγραφών. Με αυτό τον τρόπο, αποφεύγονται περιττοί συμπερασμοί ή συμπερασμοί που έχουν ήδη υπολογιστεί στο παρελθόν, και οι οποίοι μπορεί να είναι η αιτία σε προβήματα κλιμάκωσης. Η βελτιστοποιημένη απόδοση του Rapid φαίνεται πιο ξεκάθαρα σε περιπτώσεις μεγάλου ερωτήματος ή μετάλου TBox. Ακόμα, ο Rapid, μετατρέπει κάθε ερώτημα που προκύπτει πλεονάζον (redundant query) σε μη πλεονάζον, καθόλη τη διάρκεια της λειτουργίας του, κάτι που θα υποννοείται παρακάτω. Πλεονάζοντα είναι τα ερωτήματα που η απάντησή τους δίνεται και από κάποιο μικρότερο ερώτημα. Ακόμα, και αυτή είναι μία λειτουργία που ο Rapid εκτελεί βελτιωμένα, καθώς φροντίζει να παράγει πολύ λιγότερα πλεονάζοντα ερωτήματα λόγω βελτιωμένου τρόπου ξετυλίγματος. Επομένως, μη πλεονάζουσες είναι και οι επαναγραφές εξόδου.

Παράδειγμα 2.5.1. Έστω το λογικό αξίωμα και τα πλεονάζοντα ερωτήματα:

$A(x) \leftarrow B(x)$

$Q(x) \leftarrow A(x), B(x),$
$Q(x) \leftarrow A(x), A(y),$
$Q(x) \leftarrow R(x, y), R(x, z)$

Αυτά τα ερωτήματα δεν εμφανίζονται σε κανένα σημείο στο Rapid γιατί θα έχουν αντικατασταθεί από τα εξής μη πλεονάζοντα ερωτήματα:

$Q(x) \leftarrow B(x),$
$Q(x) \leftarrow A(x),$
$Q(x) \leftarrow R(x, y)$

Για τις δύο τελευταίες περιπτώσεις έχει χρησιμοποιηθεί η διαδικασία της συμπύκνωσης *(condensation)* για τις μεταβλητές y και z αντίστοιχα, οι οποίες συμπυκνώθηκαν με τις x και y αντίστοιχα.

Επιπλέον δεν παράγονται αντίγραφα.

Παράδειγμα 2.5.2. Έστω το σύνολο λογικών αξιωμάτων και το ερώτημα:

$A(x) \leftarrow B(x),$
$B(x) \leftarrow C(x),$
$A(x) \leftarrow C(x)$

$Q(x) \leftarrow A(x)$

Παρατηρούμε ότι η επαναγραφή $Q'(x) \leftarrow C(x)$ προκύπτει με δύο διαφορετικούς τρόπους με ξετυλίγματα. Όμως, θα υπολογιστεί μία φορά, λόγω του τρόπου λειτουργίας του αλγορίθμου.

Λαμβάνοντας μία οντολογία στην είσοδό του, ο αλγόριθμος μεταφράζει τα λογικά αξιώματά της σε ένα σύνολο από προτάσεις *(clauses)* πρώτης τάξης οι οποίες χρησιμοποιούνται για τη διαδικασία επαναγραφής ερωτημάτων. Η οργάνωση αυτή των αξιωμάτων είναι πολύ επιβοηθητική για τη δημιουργία του **γράφου ξετυλίγματος** του συστήματός μας, όπως θα φανεί στο κεφάλαιο 3. Ουσιαστικά, οι προτάσεις αυτές είναι οι κανόνες αντικατάστασης όρων σε μία ερώτημα ώστε να προκύψει μία επαναγραφή. Οι αντιστοιχήσεις για τη μετατροπή αυτή φαίνονται στον Πίνακα 2.1. Κάθε υπαρξιακό αξίωμα, αντικαθίσταται από μία συναρτησιακή πρόταση. Η συνάρτηση αυτή είναι μοναδική για το κάθε αξίωμα. Επίσης γίνεται αντιληπτό, ότι αν πρόκειται για αξίωμα συρρίκνωσης, δηλαδή $A \sqsubseteq \exists P.B$ ή $A \sqsubseteq \exists P^{-}.B$, αντικαθίσταται με δύο συναρτησιακές προτάσεις, οι οποίες συνδέονται από την κοινή συνάρτηση στην κεφαλή τους.

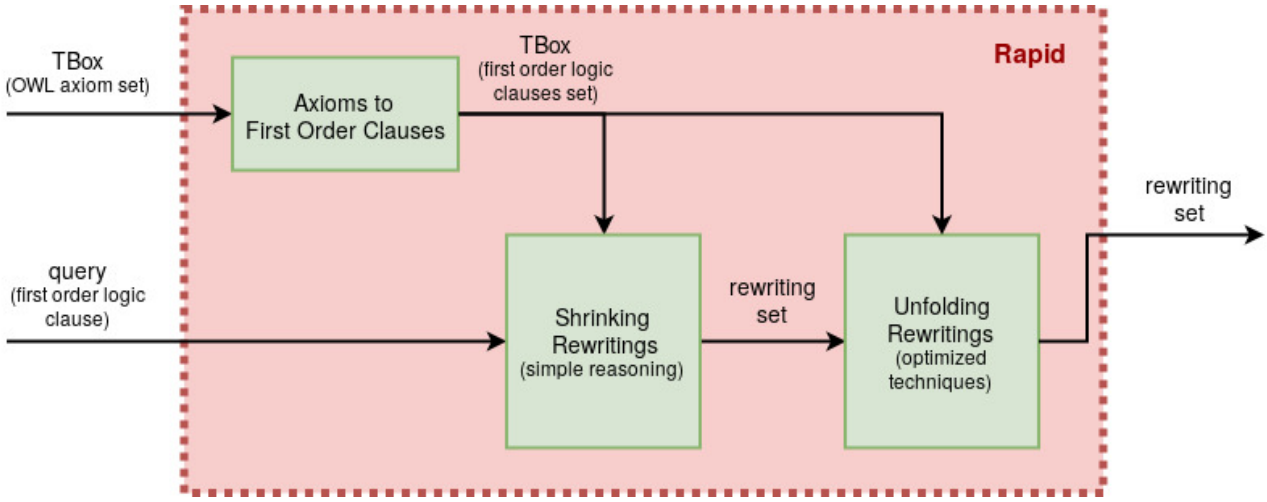
Όσο για τα ερωτήματα εισόδου, αυτά πρέπει να δίνονται στη σύνηθη μορφή που δίνεται ένα συζευκτικό ερώτημα, όπως παρουσιάζεται στους ορισμούς. Στην εργασία αυτή, υποθέτεται για απλότητα ότι τα συζευκτικά ερωτήματα δεν περιέχουν διακεκριμένες σταθερές και ασύνδετους όρους και ότι όλες οι διακεκριμένες μεταβλητές εμφανίζονται στο σώμα.

2.5.2 Διαισθητική επεξήγηση αλγορίθμου

Αρχικά, η διαδικασία παραγωγής των επαναγραφών, που αναλύεται στη συνέχεια, εκτελείται για κάθε ένα από τα ερωτήματα εισόδου. Συγκεκριμένα, αφού προσθέσει το αρχικό ερώτημα σε ένα ενδιάμεσο σύνολο, εκτελείται επαναληπτικά, για κάθε πρόταση του συνόλου, το εξής: Για κάθε

Αξίωμα	Πρόταση	Αξίωμα	Πρόταση
$A \sqsubseteq B$	$B(x) \leftarrow A(x)$		
$P \sqsubseteq S$ $P^- \sqsubseteq S^-$	$S(x,y) \leftarrow P(x,y)$	$P \sqsubseteq S^-$ $P^- \sqsubseteq S$	$S(x,y) \leftarrow P(y,x)$
$\exists P \sqsubseteq A$	$A(x) \leftarrow P(x,y)$	$\exists P^- \sqsubseteq A$	$A(x) \leftarrow P(y,x)$
$A \sqsubseteq \exists P$	$P(x, f^A_P(x)) \leftarrow A(x)$	$A \sqsubseteq \exists P^-$	$P(f^A_{P^-}(x), x) \leftarrow A(x)$
$A \sqsubseteq \exists P.B$	$P(x, f^A_{P.B}(x)) \leftarrow A(x)$ $B(f^A_{P.B}(x)) \leftarrow A(x)$	$A \sqsubseteq \exists P^-.B$	$P(f^A_{P^-.B}(x), x) \leftarrow A(x)$ $B(f^A_{P^-.B}(x)) \leftarrow A(x)$

Πίνακας 2.2: Κανόνες αντικατάστασης. Προκύπτουν από μετατροπή των αξιωμάτων της DL-Lite$_R$ σε προτάσεις λογικής πρώτης τάξης.



Σχήμα 2.1: Σχηματική αναπαράσταση συστήματος Rapid. Στο πρώτο στάδιο (Shrinking Rewriting) εφαρμόζεται ο πρώτος αλγόριθμος ξετυλίγματος και στο δεύτερο στάδιο (Unfolding Rewriting) εφαρμόζεται ο δεύτερος, βελτιστοποιημένος αλγόριθμος ξετυλίγματος.

υποψήφια μεταβλητή στην πρόταση για συρρίκνωση, δηλαδή κάθε μεταβλητή που υπάρχει στο σώμα αλλά όχι στην κεφαλή της πρότασης, γίνεται προσπάθεια να εξαλλειφθεί με τη διαδικασία συρρίκνωσης. Αν είναι δυνατό να πραγματοποιηθεί συρρίκνωση, με βάση τα λογικά αξιώματα, αντικαθίστανται οι όροι που περιέχουν την υποψήφια μεταβλητή με κάποιον άλλο που δεν την περιέχει και επιστρέφεται το σύνολο των νέων παραγόμενων προτάσεων, που αποτελούν επαναγραφές. Οι επαναγραφές αυτές προστίθενται σε αυτό το ενδιάμεσο σύνολο και η διαδικασία αυτή επαναμβάνεται για την επόμενη υποψήφια μεταβλητή.

Με το τέλος της επαναληπτικής διαδικασίας, έχουν πραγματοποιηθεί όλες οι επιτρεπτές συρρικνώσεις και το ενδιάμεσο σύνολο περιέχει όλα τα αποτελέσματα συρρικνώσεων, μαζί με τις αρχικές προτάσεις. Ουσιαστικά, μένει να εφαρμοστούν οι κανόνες ξετυλίγματος στις προτάσεις του συνόλου ώστε να παραχθεί όλόκληρο το σύνολο επαναγραφών.

Στην παραπάνω διαδικασία χρησιμοποιούνται δύο διαφορετικοί αλγόριθμοι ξετυλίγματος, οι οποίοι παρουσιάζονται παρακάτω.

2.5.2.1 Πρώτος αλγόριθμος ξετυλίγματος

Ο πρώτος χρησιμοποιείται εσωτερικά του αλγορίθμου συρρίκνωσης και δεν υλοποιεί όλες τις ιδέες βελτιστοποίησης του Rapid. Πιο συγκεκριμένα, τροποποποιεί πλεονάζοντα ερωτήματα αλλά επιλύει τα αξιώματα στην αρχική πρόταση στα "τυφλά". Αυτό που κάνει σε μία πρόταση, είναι να ξετυλίγει τους όρους που περιέχουν την υποψήφια προς συρρίκνωση μεταβλητή μέχρι να προκύψουν όροι που να επιδέχονται συρρίκνωση, δηλαδή μέχρι να εμφανιστεί κάποιο ζεύγος ισχυρισμών στο σώμα της πρότασης που είναι στιγμιότυπο ζεύγους αξιωμάτων συρρίκνωσης. Ένα ζεύγος αξιωμάτων συρρίκνωσης είναι δύο προτάσεις με συναρτησιακές μεταβλητές της ίδιας συνάρτησης f στην κεφαλή τους. Ακολουθούν μερικά παραδείγματα για να γίνει κατανοητό πώς λειτουργεί ο πρώτος αλγόριθμος ξετυλίγματος.

Παράδειγμα 2.5.3. Έστω το σύνολο λογικών αξιωμάτων και το ερώτημα:

$$B(x) \leftarrow A(x),$$
$$R(x,y) \leftarrow S(x,y),$$
$$S(x, f_0(x)) \leftarrow C(x),$$
$$D(f_0(x)) \leftarrow C(x)$$

$$Q(x) \leftarrow B(x), R(x,y), D(y)$$

Πρώτη πρόταση στη διαδικασία συρρίκνωσης είναι το Q. Για τη μεταβλητή y, που είναι η μόνη υποψήφια, θα γίνει προσπάθεια συρρίκνωσης των όρων που περιέχουν την y. Οι όροι αυτοί κάθ' αυτοί δενμ μπορούν να συρρικνωθούν. Επομένως, ξετυλίγονται με βάση το δεύτερο αξίωμα οπότε παράγεται η επαναγραφή $Q_1 : Q(x) \leftarrow B(x), S(x,y), D(y)$. Πλέον, στην πρόταση αυτή και με βάση τα δύο τελευταία αξιώματα μπορεί να γίνει συρρίκνωση της μεταβλητής y. Το νέο παραγόμενο ερώτημα $Q_2 : Q(x) \leftarrow B(x), C(x)$ προστίθεται στο σύνολο αποτελέσματος. Γίνεται αντιληπτό ότι το πρώτο αξίωμα δεν χρησιμοποιήθηκε κάπου στη διαδικασία γιατί δεν σχετίζεται με τη μεταβλητή συρρίκνωσης y. Στη συνέχεια, το Q_2 είναι η νέα πρόταση στη διαδικασία που όμως δεν επιδέχεται συρρίκνωση, αφού δεν υπάρχει υποψήφια μεταβλητή.

Παράδειγμα 2.5.4. Έστω το σύνολο λογικών αξιωμάτων και το ερώτημα:

$$R(x, f_0(x)) \leftarrow C(x),$$
$$A(f_0(x)) \leftarrow C(x),$$
$$R(x, f_1(x)) \leftarrow D(x),$$
$$A(f_1(x)) \leftarrow D(x),$$
$$D(x) \leftarrow E(x)$$

$$Q(x) \leftarrow R(x,z), R(z,y), A(y)$$

Για το αρχικό ερώτημα, έστω ότι η πρώτη υποψήφια μεταβλητή είναι η z. Όπως φαίνεται δεν μπορεί να συρρικνωθεί γιατί δεν υπαρχουν τα κατάλληλα αξιώματα. Η επόμενη υποψήφια είναι η y, η οποία συρρικνώνεται είτε με το f_0-ζεύγος είτε με το f_1-ζεύγος αξιωμάτων. Για το λόγο αυτό θα επιστραφούν οι επαναγραφές $\{Q_1 : Q(x) \leftarrow R(x,z), C(z), Q_2 : Q(x) \leftarrow R(x,z), D(z)\}$. Στη συνέχεια, το Q_1 είναι η νέα πρόταση στη διαδικασία που όμως δεν επιδέχεται συρρίκνωση, λόγω έλλειψης κατάλληλων αξιωμάτων. Ομοίως για το Q_2.

$$Q_3 : Q(x) \leftarrow R(x,z), E(z)$$

Παράδειγμα 2.5.5. Έστω το σύνολο λογικών αξιωμάτων και το ερώτημα:

$$R(x, f_0(x)) \leftarrow C(x),$$
$$A(f_0(x)) \leftarrow C(x),$$
$$R(x, f_1(x)) \leftarrow D(x),$$
$$C(f_1(x)) \leftarrow D(x),$$
$$D(x) \leftarrow E(x)$$
$$Q(x) \leftarrow R(x, z), R(z, y), A(y)$$

Για το Q, ομοίως με πριν, η μεταβλητή z δεν συρρικνώνεται. Εν συνεχεία, η μεταβλητή y συρρικνώνεται με το f_0-ζεύγος παράγεται η επαναγραφή $Q_1 : Q(x) \leftarrow R(x, z), C(z)$, η οποία και επιστρέφεται. Σε δεύτερη φάση, το Q_1 είναι η νέα πρόταση στη διαδικασία η οποία επιδέχεται επιπλέον συρρίκνωση λόγω του f_1-ζεύγους αξιωμάτων και οπότε προκύπτει το $Q_2 : Q(x) \leftarrow D(x)$. Το Q_2 επιστρέφεται και προστίθεται στο σύνολο αποτελέσματος μαζί με τα Q και Q_1. Το Q_2 είναι η νέα πρόταση στη διαδικασία, όμως δεν επιδέχεται συρρίκνωση, αφού δεν υπάρχει υποψήφια μεταβλητή.

Παράδειγμα 2.5.6. Έστω το σύνολο λογικών αξιωμάτων και το ερώτημα:

$$R(x, f_0(x)) \leftarrow B(x),$$
$$A(f_0(x)) \leftarrow B(x)$$
$$Q(x, y) \leftarrow R(x, y), A(y)$$

Τονίζεται ότι εδώ δεν υπάρχει καμία υποψήφια μεταβλητή για συρρίκνωση και άρα δεν εκτελείται καμία φορά ο αλγόριθμος συρρίκνωσης.

2.5.2.2 Δεύτερος αλγόριθμος ξετυλίγματος

Ο δέυτερος αλγόριθμος ξετυλίγματος εφαρμόζεται στο ενδιάμεσο σύνολο αποτελεσμάτων που περιέχει όλα τα αρχικά ερωτήματα και τις συρρικνώσεις αυτών. Στη φάση αυτή, δεν εξαλείφεται κάποια bound μεταβλητή αλλά μόνο ίσως κάποια unbound μέσω συμπύκνωσης. Επίσης είναι δυνατό να εισαχθεί κάποια unbound μεταβλητή μέσω κάποιου υπαρξιακού αξιώματος ξετυλίγματος, όπως το $A(x) \leftarrow R(x, y)$. Ο αλγόριθμος αυτός δεν επιλύει τους όρους των ερωτημάτων στα "τυφλά", όπως ο προηγούμενος αλλά οργανωμένα και συστηματικά, ώστε να μην κάνει άσκοπο reasoning. Συγκεκριμένα, σε ένα πίνακα δύο διαστάσεων, όπου σε στήλες και γραμμές έχει όλους τους όρους της οντολογίας, διατηρεί την πληροφορία των επιλύσεων, δηλαδή ποιοι όροι προκύπτουν από ποιους, και, κάθε φορά που υπολογίζει μία νέα επίλυση, δηλαδή εξάγει μία νέα συσχέτιση μεταξύ όρων, ανανεώνει όλα τα κελιά του πίνακα που σχετίζονται με την επίλυση αυτή. Με αυτόν τον τρόπο, γλιτώνει το να επιλύσει αλυσίδες κανόνων για ένα ερώτημα τις οποίες έχει ήδη επιλύσει για ένα άλλο ερώτημα, κάτι πολύ χρήσιμο όταν προκύπτυν ίδιες, πολύπλοκες αλυσίδες αξιωμάτων ή μικρότερες που είναι υποσύνολα μεγαλύτερων.

Τα αποτελέσματα αυτού του αλγορίθμου είναι όλες οι δυνατές επαναγραφές του ενδιάμεσου συνόλου που προκύπτουν με αξιώματα ξετυλίγματος και δεν υπολογίζονται σειριακά σε σχέση με τα λογικά βήματα ξετυλίγματος που υποννοούνται, λόγω της λογικής της προηγούμενης παραγράφου. Στη συνέχεια επεκτείνεται ένα από τα προηγούμενα παραδείγματα για να γίνει πλήρως κατανοητή η διαδικασία.

Παράδειγμα 2.5.7. Σε συνέχεια του Παραδείγματος 1.4.3., το ενδιάμεσο σύνολο επαναγραφών έχει ως εξής:

$Q(x) \leftarrow B(x), R(x, y), D(y),$
$Q(x) \leftarrow B(x), C(x)$

Με εφαρμογή του δεύτερου αλγορίθμου ξετυλίγματος προκύπτουν οι επαναγραφές:

$Q(x) \leftarrow A(x), R(x, y), D(y),$ //*εφ. 1ου κανόνα στο 1ο ερώτημα*
$Q(x) \leftarrow A(x), S(x, y), D(y),$ //*εφ. 1ου και 2ου κανόνα στο 1ο ερώτημα*
$Q(x) \leftarrow B(x), S(x, y), D(y),$ //*εφ. 2ου κανόνα στο 1ο ερώτημα*
$Q(x) \leftarrow A(x), C(x)$ //*εφ. 1ου κανόνα στο 2ο ερώτημα*

Τελικά το σύνολο επαναγραφών εξόδου είναι η ένωση των δύο παραπάνω συνόλων.

Κεφάλαιο 3

Σχεδιασμός

Στο κεφάλαιο αυτό παρουσιάζεται ο θεωρητικός σχεδιασμός και οι προϋποθέσεις του συστήματος παραγωγής του συνόλου δεδομένων εκπαίδευσης, DatAbox. Αρχικά, αναλύεται ο σχηματισμός της βάσης του συνόλου δεδομένων, η οποία επεκτείνεται με σκοπό τη δημιουργία του συνόλου DatAbox. Παρουσιάζονται ορισμοί νέων εννοιών και κάποιες συνθήκες οι οποίες είναι απαραίτητο να ισχύουν για τα στοιχεία της βάσης. Τέλος, παρουσιάζεται ο αλγόριθμος σχηματισμού του DatAbox.

3.1 Ορισμοί

Όπως αναφέρθηκε ήδη από την εισαγωγή, σκοπός του συστήματος που αναπτύχθηκε είναι η παραγωγή ποιοτικών δεδομένων για εκπαίδευση ενός νευρωνικού, σε σχέση με μία οντολογία. Με τον όρο "ποιοτικά δεδομένα" για μία οντολογία, εννοούμε ABoxes, που δημιουργούνται για ένα συγκεκριμένο ερώτημα, πάνω στα οποία απαιτείται να γίνει εξαγωγή συμπερασμών ώστε να απαντηθεί σωστά και πλήρως το ερώτημα. Στο πλαίσιο αυτό εισάγουμε κάποιες νέες έννοιες.

3.1.1 Γενικοί χαρακτηρισμοί

Αρχικά, παρουσιάζονται διάφοροι χρήσιμοι χαρακτηρισμοί για ABoxes. Εισάγεται η έννοια του *ελάχιστου (mini)* ABox σε σχέση με ένα ερώτημα, στο οποίο αν αφαιρεθεί ένας ισχυρισμός τότε η απάντησή του στο ερώτημα αυτό γίνεται κενή. Ουσιαστικά, ένα ελάχιστο ABox περιέχει ένα ελάχιστο σύνολο ισχυρισμών ώστε να μπορεί να εξαχθεί ένα μόνο άτομο του κόσμου ως απάντηση. Στη συνέχεια, ορίζεται το *βαρετό (boring)* ABox το οποίο ουσιαστικά δεν απαιτεί καμία εξαγωγή συμπερασμού για να δώσει πλήρη απάντηση όπως και το *ενδιαφέρον (interesting)* ABox το οποίο, αντίθετα με το βαρετό, απαιτεί την εξαγωγή τουλάχιστον ενός συμπερασμού για να απαντήσει πλήρως στο ερώτημα.

Ορισμός 3.1.1. Καλούμε ένα ABox $\mathcal{A}$ ελάχιστο σε σχέση με ένα TBox $\mathcal{T}$ κι ένα ερώτημα q, αν και μόνο αν $\text{ans}(q, \mathcal{T}, \mathcal{A}) \neq \emptyset$ και για κάθε $\mathcal{A}' \subset \mathcal{A}$, ισχύει ότι $\text{ans}(q, \mathcal{T}, \mathcal{A}') = \emptyset$.

Ορισμός 3.1.2. Καλούμε ένα ABox $\mathcal{A}$ βαρετό σε σχέση με ένα TBox $\mathcal{T}$ κι ένα ερώτημα q, αν και μόνο αν $\text{ans}(q, \mathcal{T}, \mathcal{A}) \neq \emptyset$ και ταυτόχρονα ισχύει ότι $\text{ans}(q, \mathcal{T}, \mathcal{A}) = \text{ans}(q, \emptyset, \mathcal{A})$, δηλαδή το $\mathcal{A}$ έχει υλοποιηθεί (materialized).

Ορισμός 3.1.3. Καλούμε ένα ABox $\mathcal{A}$ ενδιαφέρον σε σχέση με ένα TBox $\mathcal{T}$ κι ένα ερώτημα q, αν και μόνο αν $\{\text{ans}(q, \mathcal{T}, \mathcal{A}) \setminus \text{ans}(q, \emptyset, \mathcal{A})\} \neq \emptyset$.

Τα ελάχιστα Aboxes θα ήταν πολύ χρήσιμα για τη δημιουργία ενδιαφέροντων αρνητικών δειγμάτων στο σύνολο εκπαίδευσης, δηλαδή ABoxes που παρά λίγο δεν δίνουν απάντηση, γι' αυτό και εισάγεται ως έννοια. Τα αρνητικά δείγματα θα προέκυπταν πολύ εύκολα, εξάγοντας έναν οποιοδήποτε ισχυρισμό από κάποιο ελάχιστο ABox. Επίσης είναι ελαχιστοτικές δομικές μονάδες μεγαλύτερων ABoxes.

Εισάγεται, επίσης, η έννοια του ενδιαφέροντος ερωτήματος, δηλαδή αυτού που για να απαντηθεί πλήρως πρέπει να γίνει κάποια εξαγωγή συμπερασμού. Συγκεκριμένα, ένα ενδιαφέρον ερώτημα πρέπει να περιέχει κάποιον όρο που να αντιστοιχεί στη δεξιά πλευρά κάποιου κανόνα του κανονικοποιημένου $\mathcal{T}$, ώστε να πρέπει να αναλυθεί ο συγκεκριμένος κανόνας για να δοθούν ως απαντήσεις και τα άτομα που αντιστοιχούν στην αριστερή του πλευρά. Δηλαδή, η απάντηση σε ένα ενδιαφέρον ερώτημα χωρίς το $\mathcal{T}$, είναι μικρότερη σε σχέση με την απάντηση υπό το $\mathcal{T}$.

Ορισμός 3.1.4. Καλούμε ένα ερώτημα q ενδιαφέρον σε σχέση με ένα TBox $\mathcal{T}$, αν και μόνο αν υπάρχει κάποιο ABox $\mathcal{A}$ έτσι ώστε $\{\text{ans}(q, \mathcal{T}, \mathcal{A}) \setminus \text{ans}(q, \emptyset, \mathcal{A})\} \neq \emptyset$.

Με βάση τους παραπάνω ορισμούς, ορίζονται και οι έννοιες *ελάχιστο βαρετό* ABox, *ελάχιστο ενδιαφέρον* ABox, *ελάχιστο ενδιαφέρον* ερώτημα. Στη συνέχεια παρουσιάζονται κάποια πολύ απλά παραδείγματα για την κατανόηση των παραπάνω όρων.

Παράδειγμα 3.1.1. Έστω το TBox $\mathcal{T} = \{A \sqsubseteq B\}$. Ένα ενδιαφέρον ερώτημα σε σχέση με το $\mathcal{T}$ είναι το $q : Q(x) \leftarrow B(x)$. Με βάση τα $q, \mathcal{T}$:

Βαρετό ABox:	$\mathcal{A}_1 = \{B(a), B(b)\} \xrightarrow{ans} \{a, b\}$
Ενδιαφέροντα ABoxes:	$\mathcal{A}_2 = \{A(a), B(b)\} \xrightarrow{ans} \{a, b\}$,
	$\mathcal{A}_3 = \{A(a), A(b)\} \xrightarrow{ans} \{a, b\}$
Ελάχιστο βαρετό ABox:	$\mathcal{A}_4 = \{B(a)\} \xrightarrow{ans} \{a\}$
Ελάχιστο ενδιαφέρον ABox:	$\mathcal{A}_5 = \{A(a)\} \xrightarrow{ans} \{a\}$

Σημειώνεται ότι το $Q(x) \leftarrow A(x)$ δεν είναι ενδιαφέρον ερώτημα. Επίσης, αν στην υπογραφή της οντολογίας, δηλαδή στο "λεξιλόγειό" της, υπήρχε η έννοια C, ένας περιττός ισχυρισμός για όλα τα παραπάνω ABoxes θα ήταν ο $C(a)$. Αν ο ισχυρισμός αυτός προστίθετο σε κάποιο ελάχιστο ABox, το κάνει μη ελάχιστο.

3.1.2 Δυσκολία ABox

Στο σημείο αυτό, ρίζεται μία κομβική έννοια στην παρούσα εργασία, η μετρική δυσκολίας *(difficulty)* ενός ABox, σε σχέση με ένα ερώτημα κι ένα TBox. Η μετρική αυτή αντικατοπτρίζει την ελάχιστη δυσκολία που απαιτείται για να εξαχθεί το διάνυσμα απάντησης σε αυτό το ερώτημα, το οποίο σχετίζεται με το πόσες εξαγωγές συμπερασμών πρέπει να λάβουν χώρα και πόσα λογικά αξιώματα πρέπει να χρησιμοποιηθούν.

Ορισμός 3.1.5. Καλούμε κόστος απάντησης *(answer cost)* ενός ατόμου απάντησης $a \in \text{ans}(q, \mathcal{T}, \mathcal{A})$, όπου q ένα ερώτημα, $\mathcal{A}$ ένα ABox και $\mathcal{T}$ ένα TBox την ποσότητα

$$\text{cost}_{q,\mathcal{T},\mathcal{A}}(a) = w_{unf} \cdot n_{unf} + w_{shr} \cdot n_{shr} + w_{exist} \cdot n_{exist},$$

όπου w_i δηλώνει το βάρος της διαδικασίας i, n_i δηλώνει το *ελάχιστο* πλήθος των i-διαδικασιών που λαμβάνουν χώρα για την εξαγωγή του a ως άτομο απάντησης και $i \in \{unf, shr, exist\}$, που δηλώνουν τις διαδικασίες ξετυλίγματος, συρρίκνωσης και ξετυλίγματος υπαρξιακού αξιώματος αντίστοιχα.

Ορισμός 3.1.6. Καλούμε μονοπάτι απάντησης *(answer path)* ενός ατόμου απάντησης $a \in \text{ans}(q, \mathcal{T}, \mathcal{A})$, όπου q ένα ερώτημα, $\mathcal{A}$ ένα ABox και $\mathcal{T}$ ένα TBox το διάνυσμα

$$\text{path}_{q,\mathcal{T}}(a) = (p_1, p_2, ..., p_n),$$

όπου p_i είναι το όνομα του ρόλου ή της έννοιας που προέκυψε από το p_{i-1} με χρήση κάποιου λογικού αξιώματος του TBox. Η διαδοχή των p_i αντιστοιχεί στο μονοπάτι εξαγωγής συμπερασμών που απαιτούνται για να χαρακτηριστεί το a ως άτομο απάντησης, με το *ελάχιστο* κόστος.

Ορισμός 3.1.7. Καλούμε δυσκολία ενός ABox $\mathcal{A}$ σε σχέση με ένα ερώτημα q κι ένα TBox $\mathcal{T}$, ένα διάνυσμα $\text{difficulty}_{q,\mathcal{T}}(\mathcal{A}) = (\text{cost}_{q,\mathcal{T}}(\mathcal{A}), \text{path}_{q,\mathcal{T}}(\mathcal{A}))$, όπου

$$\text{cost}_{q,\mathcal{T}}(\mathcal{A}) = \max_{\forall a \in ans(q,\mathcal{T},\mathcal{A})} \{\text{cost}_{q,\mathcal{T},\mathcal{A}}(a)\},$$

είναι το κόστος απάντησης του $\mathcal{A}$ και

$$\text{path}_{q,\mathcal{T}}(\mathcal{A}) = \text{path}_{q,\mathcal{T},\mathcal{A}}(a_{max}), \text{όπου } a_{max} \text{ είναι τ.ώ. } \text{cost}_{q,\mathcal{T}}(\mathcal{A}) = \text{cost}_{q,\mathcal{T},\mathcal{A}}(a_{max}).$$

είναι το μονοπάτι απάντησης του $\mathcal{A}$. Ουσιαστικά, η δυσκολία του $\mathcal{A}$ παίρνει την τιμή της από το άτομο απάντησης με το μέγιστο κόστος απάντησης.

Η εξαγωγή κάθε ατόμου απάντησης έχει ένα κόστος, το οποίο είναι το ελάχιστο δυνατό με βάση τα λογικά αξιώματα του TBox, όπως φαίνεται και από το πολύ απλό παράδειγμα:

Παράδειγμα 3.1.2. Έστω TBox $\mathcal{T} = \{A \sqsubseteq B\}$, ABox $\mathcal{A} = \{A(a), B(a)\}$ και $q : Q(x) \leftarrow B(x)$. Τα βάρη w_i για το κόστος απάντησης είναι μοναδιαία. Για την απάντηση ισχύει ότι $\text{ans}(q, \mathcal{T}, \mathcal{A}) = \{a\}$, το οποίο μπορεί να εξαχθεί και από τους δύο ισχυρισμούς. Όμως, επιλέγεται αυτός που δίνει το ελάχιστο κόστος απάντησης, δηλαδή τον $B(a)$ και τελικά το διάνυσμα δυσκολίας είναι $(0, \{B\})$.

Έτσι, όταν αναφερόμαστε στη δυσκολία εξαγωγής ολόκληρης της απάντησης από ένα ABox, τότε θα πρέπει να λάβουμε υπόψην τη δυσκολία για κάθε άτομο απάντησης. Επομένως, η ελάχιστη δυσκολία που απαιτείται για να δοθεί η απάντηση αυτή είναι η δυσκολία του πιο κοστοβόρου ατόμου απάντησης. Το παρακάτω θεώρημα συνοψίζει αυτές τις σκέψεις.

Θεώρημα 3.1.1. Έστω ένα ABox $\mathcal{A}$, ένα ερώτημα q κι ένα TBox $\mathcal{T}$, κι έστω ότι η δυσκολία εξαγωγής της απάντησης $\text{ans}(q, \mathcal{T}, \mathcal{A})$ είναι το διάνυσμα $\text{difficulty}_{q,\mathcal{T}}(\mathcal{A}) = (\text{cost}_{q,\mathcal{T}}(\mathcal{A}), \text{path}_{q,\mathcal{T}}(\mathcal{A}))$. Οι δύο μετρικές του διανύσματος περιγράφουν το *ελάχιστο* κόστος απάντησης και το αντίστοιχο μονοπάτι, που απαιτείται για να εξαχθεί η απάντηση $\text{ans}(q, \mathcal{T}, \mathcal{A})$.

Απόδειξη: Το διάνυσμα δυσκολίας βασίζεται στο κόστος απάντησης, επομένως η απόδειξη θα γίνει σε σχέση με αυτό. Έστω $\text{cost}_{q,\mathcal{T},\mathcal{A}}(a_i)$ το κόστος απάντησης του ατόμου a_i υπό τα $q, \mathcal{T}, \mathcal{A}$. Έστω $\text{cost}_{q,\mathcal{T}}(\mathcal{A})$ το κόστος απάντησης του $\mathcal{A}$ σε σχέση με τα $q, \mathcal{T}$ το οποίο προκύπτει από το άτομο απάντησης a, δηλαδή $\text{cost}_{q,\mathcal{T}}(\mathcal{A}) = \text{cost}_{q,\mathcal{T},\mathcal{A}}(a)$, και το οποίο δεν είναι ελάχιστο. Τότε, θα υπάρχει μία τιμή $\text{cost'}_{q,\mathcal{T}}(\mathcal{A})$ για την οποία θα ισχύει $\text{cost}_{q,\mathcal{T}}(\mathcal{A}) < \text{cost'}_{q,\mathcal{T}}(\mathcal{A})$ και άρα θα υπάρχει ένα άτομο απάντησης a' για το οποίο θα ισχύει $\text{cost'}_{q,\mathcal{T}}(\mathcal{A}) = \text{cost}_{q,\mathcal{T},\mathcal{A}}(a')$.

Αν ισχύει ότι $a = a'$, τότε $\text{cost}_{q,\mathcal{T}}(\mathcal{A}) < \text{cost'}_{q,\mathcal{T}}(\mathcal{A})$ $\text{cost}_{q,\mathcal{T},\mathcal{A}}(a) < \text{cost}_{q,\mathcal{T},\mathcal{A}}(a)$, άτοπο, καθώς υπάρχει μία μοναδική τιμή $\text{cost}_{q,\mathcal{T},\mathcal{A}}(a)$.

Αν ισχύει ότι $a \neq a'$, τότε για να εξαχθεί το a απαιτείται κόστος $\text{cost}_{q,\mathcal{T},\mathcal{A}}(a) > \text{cost'}_{q,\mathcal{T}}(\mathcal{A})$ και άρα το άτομο a δεν ανήκει στην απάντηση, καθώς το $\text{cost'}_{q,\mathcal{T}}(\mathcal{A})$ συμπεριλαμβάνει το κόστος για την εξαγωγή όλων των ατόμων απάντησης, άτοπο γιατί το a ανήκει στην απάντηση.

Ακολουθεί ένα απλό παράδειγμα για την κατανόηση όλων των παραπάνω.

Παράδειγμα 3.1.3. Έστω το TBox $\mathcal{T} = \{A \sqsubseteq B, B \sqsubseteq C, \exists R.\top \sqsubseteq C\}$. Ένα ενδιαφέρον ερώτημα σε σχέση με το $\mathcal{T}$ είναι το $q : Q(x) \leftarrow C(x)$. Με βάση τα $q, \mathcal{T}$:

Βαρετό ABox:	$\mathcal{A}_1 = \{C(a), C(b)\} \xrightarrow{ans} \{a, b\}$
Ενδιαφέροντα ABoxes:	$\mathcal{A}_2 = \{A(a), B(b)\} \xrightarrow{ans} \{a, b\}$,
	$\mathcal{A}_3 = \{B(a), R(b, c)\} \xrightarrow{ans} \{a, b\}$,
	$\mathcal{A}_4 = \{A(a), A(b)\} \xrightarrow{ans} \{a, b\}$,
	$\mathcal{A}_5 = \{A(a), C(a), B(b)\} \xrightarrow{ans} \{a, b\}$
Ελάχιστο βαρετό ABox:	$\mathcal{A}_6 = \{C(a)\} \xrightarrow{ans} \{a\}$
Ελάχιστα ενδιαφέροντα ABoxes:	$\mathcal{A}_7 = \{A(a)\} \xrightarrow{ans} \{a\}$,
	$\mathcal{A}_8 = \{B(a)\} \xrightarrow{ans} \{a\}$,
	$\mathcal{A}_9 = \{R(a, b)\} \xrightarrow{ans} \{a\}$

Οι δυσκολίες των παραπάνω ABoxes, για μοναδιαία βάρη w_i, έχουν ως εξής:

$\mathcal{A}_1 \longrightarrow (0, \{C\})$	*//άτομο a ή b*
$\mathcal{A}_2 \longrightarrow (2, \{C, B, A\})$	*//άτομο a*
$\mathcal{A}_3 \longrightarrow (2, \{C, R\})$	*//άτομο b*
$\mathcal{A}_4 \longrightarrow (2, \{C, B, A\})$	*//άτομο a ή b*
$\mathcal{A}_5 \longrightarrow (1, \{C, B\})$	*//άτομο b*
$\mathcal{A}_6 \longrightarrow (0, \{C\})$	*//άτομο a*
$\mathcal{A}_7 \longrightarrow (2, \{C, B, A\})$	*//άτομο a*
$\mathcal{A}_8 \longrightarrow (1, \{C, B\})$	*//άτομο a*
$\mathcal{A}_9 \longrightarrow (2, \{C, R\})$	*//άτομο a*

Στο σύνολο δεδομένων που δημιουργούμε, επιθυμούμε να περιέχονται μόνο ενδιαφέροντα ABoxes, ώστε να υπηρετείται η εκμάθηση εξαγωγών συμπερασμών και όχι "αυτονόητων" απαντήσεων που δίνονται από βαρετά ABoxes. Με άλλα λόγια, ζητούμε ABoxes με μη μηδενικό κόστος απάντησης. Επίσης, όσο πιο σύνθετο θέλουμε να γίνει το σύνολο δεδομένων εκπαίδευσης, τόσο περισσότερο πρέπει να εμφανίζονται ABoxes με υψηλή τιμή κόστους απάντησης.

3.1.3 Δυσκολία επαναγραφής

Ακολουθούν χρήσιμοι ορισμοί για το σχηματισμό της βάσης από ABoxes που θα χρησιμοποιηθεί για το σχηματισμό του συνόλου δεδομένων εκπαίδευσης. Η μετρική δυσκολίας ABox, όπως ορίστηκε μέχρι τώρα, αφορά μόνο τα ονόματα εννοιών και ρόλων που εμφανίζονται σε ένα ABox. Έτσι, εισάγουμε τη δυσκολία ενός όρου του ερωτήματος επαναγραφής και στη συνέχεια τη δυσκολία του ίδιου του ερωτήματος επαναγραφής, σε σχέση με ένα αρχικό ερώτημα.

Ορισμός 3.1.8. Έστω ένα ερώτημα q κι ένα DL-Lite$_R$-TBox $\mathcal{T}$. Έστω επίσης το *κόστος μεταβλητής απάντησης (answer variable cost)* και το *μονοπάτι μεταβλητής απάντησης (answer variable path)* τα οποία ορίζονται όπως το κόστος απάντησης και το μονοπάτι απάντησης αντίστοιχα, μόνο που αφορούν μία οποιαδήποτε μεταβλητή x, αντί ενός ατόμου απάντησης a. Καλούμε *δυσκολία επαναγραφής (rewriting difficulty)* μίας επαναγραφής q_i ενός ερωτήματος q, σε σχέση με το $\mathcal{T}$, το διάνυσμα $\text{rewdifficulty}_{q,\mathcal{T}}(q_i) = (\text{cost}_{q,\mathcal{T}}(q_i), \text{path}_{q,\mathcal{T}}(q_i))$, τα συστατικά του οποίου ορίζονται αντίστοιχα με αυτά στον ορισμό δυσκολίας ενός ABox.

Ορισμός 3.1.9. Καλούμε δυσκολία επαναγραφής ενός όρου Q_i του αρχικού ερωτήματος σε ένα όρο Q_j της επαναγραφής, τη δυσκολία $\text{rewdifficulty}_{Q_i,\mathcal{T}}(Q_j)$, δηλαδή θεωρούμε τα Q_i, Q_j ατομικά ερωτήματα και υπολογίζουμε τη δυσκολία επαναγραφής τους.

3.2 Βάση συνόλου εκπαίδευσης

Για κάθε ένα ερώτημα από το σύνολο ερωτημάτων εκπαίδευσης, χρησιμοποιείται μία βάση από "μικρά" ABoxes, τα οποία εν συνεχεία συνδυάζονται ώστε να παράξουν μεγαλύτερα και πιο σύνθετα. Η βάση αυτή θέλουμε να είναι ελάχιστη, δηλαδή να μην περιέχει ισομορφικά ABoxes, αλλά ούτε και στοιχεία που μπορούν να προκύψουν από άλλα στοιχεία της βάσης.

3.2.1 Βάση των ελάχιστων ABoxes

Η αρχική σκέψη ήταν η βάση αυτή να περιέχει τα ελάχιστα ABoxes, σε σχέση με ένα TBox κι ένα ερώτημα, κάτι που θα ήταν πολύ χρήσιμο, γιατί θα αποτελούσε όλους τους ελαχιστοτικούς κόσμους στους οποίους δίνεται μη κενή απάντηση στο ερώτημα. Μία τέτοια βάση θα ήταν η εξής:

Ορισμός 3.2.1. Καλούμε *βάση των ελάχιστων ABoxes (mini Abox-base)*, σε σχέση με ένα TBox $\mathcal{T}$ κι ένα ερώτημα q, το ελάχιστο σύνολο $\{\mathcal{A} \xrightarrow{ans} \text{ans}(q, \mathcal{T}, \mathcal{A})\}_{\forall \mathcal{A}:\mathcal{A}\in\mathcal{M}}$, όπου $\mathcal{M}$ είναι το σύνολο ελάχιστων ABoxes $\mathcal{A}$ σε σχέση με τα $\mathcal{T}, q$. Δηλαδή, η βάση είναι εξαντλητική και ελάχιστη ως προς τα ελάχιστα ABoxes.

Παράδειγμα 3.2.1. Στο Παράδειγμα 3.1.3, για το q, για τη βάση $\mathcal{B}$ των ABoxes ισχύει ότι $\mathcal{B} = \{\mathcal{A}_6, \mathcal{A}_7, \mathcal{A}_8, \mathcal{A}_9\}$ καθώς είναι όλα τα δυνατά ελάχιστα Aboxes σε σχέση με τα $\mathcal{T}, q$. Στο ίδιο παράδειγμα, για το $q' : Q(x) \leftarrow B(x)$, η βάση $\mathcal{B}'$ των ABoxes θα αποτελείτο από τα: $\mathcal{B}' = \{\mathcal{A}'_1 \xrightarrow{ans} \{a\}, \mathcal{A}'_2 \xrightarrow{ans} \{a\}\}$, όπου $\mathcal{A}'_1 = \{A(a)\}$ και $\mathcal{A}'_2 = \{B(a)\}$.

Η εξαγωγή της βάσης των ελάχιστων ABoxes, είναι ένα μεγάλο πρόβλημα από μόνο του και θεωρήθηκε πέρα από τα πλαίσια αυτής της εργασίας. Επίσης, είναι ζήτημα το ότι πρέπει να υπολογισθούν οι απαντήσεις των ερωτημάτων πάνω στα ελάχιστα ABoxes, κάτι που μπορεί να αυξήσει την πολυπλοκότητα του συστήματος.

Για το σκοπό μας, δηλαδή για την παραγωγή ενδιαφέροντος συνόλου δεδομένων, αρκεί μία λιγότερο περιοριστική κι επομένως απλούστερη βάση, η λεγόμενη *απλή βάση των Aboxes (simple ABox-base)*, στην οποία, ακόμα, η απάντηση προκύπτει άμεσα.

3.2.2 ABox επαναγραφής

Πριν ορίσουμε τη βάση αυτή, εισάγουμε την εξής έννοια:

Ορισμός 3.2.2. Έστω ένα DL-Lite$_R$-TBox $\mathcal{T}$, συζευκτικό ερώτημα q κι ένα σύστημα επαναγραφής ερωτημάτων $\mathcal{S}$, το οποίο παράγει ένα εξαντλητικό κι ελάχιστο σύνολο από μη περιττά ερωτήματα, έστω το UCQ. Καλούμε *ABox επαναγραφής (rewriting Abox)*, κάθε ABox που παράγεται από μία επαναγραφή q_i του UCQ, με τον εξής τρόπο: γίνεται αντιστοίχιση κάθε διακεκριμένης μεταβλητής απάντησης με ένα διακεκριμένο άτομο απάντησης και κάθε ελεύθερη μεταβλητή με κάποιο διακεκριμένο άτομο, κ έτσι, δημιουργούνται οι παρακάτω ισχυρισμοί:

- Για κάθε όρο $A(x)$ του σώματος του q_i, όπου A ατομικό, παράγεται ο ισχυρισμός $A(a_j^i)$, όπου a_j^i το άτομο που αντιστοιχήθηκε η μεταβλητή x.
- Για κάθε όρο $R(x, y)$ του σώματος του q_i, όπου R ατομικό, παράγεται ο ισχυρισμός $R(a_j^i, a_k^i)$, όπου a_j^i, a_k^i τα άτομα που αντιστοιχήθηκαν οι μεταβλητές x, y.

Με βάση τον παραπάνω ορισμό και τον τρόπο σχηματισμού ενός ABox επαναγραφής $\mathcal{A}$, γίνεται προφανές το επόμενο θεώρημα.

Θεώρημα 3.2.1. Έστω ένα TBox $\mathcal{T}$ κι ερώτημα επαναγαφής q_i ενός αρχικού ερωτήματος q, που προκύπτει από ένα σύστημα επαναγαφής $\mathcal{S}$ ε τις γνωστές επιθυμητές ιδιότητες. Έστω ένα ABox επαναγραφής $\mathcal{A}$ που προκύπτει από το q_i. Ισχύει ότι $\text{cert}(q, \mathcal{T}, \mathcal{A}) = \text{cert}(q_i, \emptyset, \mathcal{A}) = \text{SQLAns}(q_i, \mathcal{A})$, όπου SQLAns είναι η απάντηση στο q_i ως SQL ερώτημα.

Παράδειγμα 3.2.2. Έστω ένα συζευκτικό ερώτημα $q : Q(x, y, z) \leftarrow A(x), R(x, y), R(x, z)$ που με TBox $\mathcal{T} = \{B \sqsubseteq A\}$ έχει ως επαναγραφή $q_1 : Q(x, y, z) \leftarrow B(x), R(x, y), R(x, z)$, όπως προκύπτει από ένα σύστημα $\mathcal{S}$ με τις γνωστές ιδιότητες. Το ABox επαναγραφής όπως ορίστηκε παραπάνω είναι το $\mathcal{A}_1 = \{B(a_x^1), R(a_x^1, a_y^1), R(a_x^1, a_z^1)\}$ και η απάντηση που προκύπτει με την παραπάνω διαδικασία είναι η $\text{cert}(q, \mathcal{T}, \mathcal{A}_1)) = \text{cert}(q_1, \emptyset, \mathcal{A}_1) = \text{SQLAns}(q_1, \mathcal{A}_1) =$
$\{(a_x^1, a_y^1, a_z^1), (a_x^1, a_z^1, a_y^1), (a_x^1, a_y^1, a_y^1), (a_x^1, a_z^1, a_z^1)\}$

Τονίζεται επίσης, ότι δεν κατασκευάζουμε ABoxes επαναγραφής όπου δύο μεταβλητές x, y του ερωτήματος αντιστοιχίζονται στο ίδιο άτομο a, διότι τότε, αν διατηρήσουμε την απλή διαδικασία μετατροπής του ερωτήματος επαναγραφής σε ABox που περιγράφηκε από πάνω, υπάρχει ο κίνδυνος να προκύψουν περιττοί ισχυρισμοί.

Παράδειγμα 3.2.3. Έστω μία επαναγραφή $q_1 : Q(x) \leftarrow A(x), R(x, y), A(y)$ ενός συζευκτικού ερωτήματος q, όπως προκύπτει από ένα σύστημα $\mathcal{S}$ με τις γνωστές ιδιότητες. Το ABox επαναγραφής όπως ορίστηκε παραπάνω είναι το $\mathcal{A}_1 = \{A(a_x^1), R(a_x^1, a_y^1), A(a_y^1)\}$. Ένα ακόμα ABox που προκύπτει από το q_1, αντιστοιχίζοντας τις μεταβλητές x, y το ίδιο άτομο a^1, είναι το $\mathcal{A}_1' = \{A(a^1), R(a^1, a^1), A(a^1)\}$, στο οποίο όμως ο τελευταίος ισχυρισμός είναι περιττός.

Παρ' όλ' αυτά, μία πιθανή επέκταση του συστήματός μας θα αποτελούσε το να δημιουργούμε και τέτοιου είδους ABoxes, στα οποία θα γίνονταν κάποιοι έλεγχοι για αποφυγή προσθήκης περιττών ισχυρισμών. Ακόμα όμως και αφήνοντας τους περιττούς ισχυρισμούς, το πρόβλημα δεν είναι σοβαρό, αφού τελικά δεν είναι και τόσο κακό να υπάρχουν περιττά πράγματα σε ένα σύνολο εκπαίδευσης ενός νευρωνικού. Από τα παραπάνω προκύπτει ο αλγόριθμος CalcABoxAns, ο οποίος χρησιμοποιεί τον calcSQLAns για να εκτελέσει ένα SQL ερώτημα πάνω στο ABox.

(ABox $\xrightarrow{ans}$ ans, difficulty)

3.2.3 Απλή βάση των ABoxes

Πλέον είμαστε σε θέση να ορίσουμε την εξής έννοια:

Ορισμός 3.2.3. Έστω ένα DL-Lite$_R$-TBox $\mathcal{T}$, ένα συζευκτικό ερώτημα q κι ένα σύστημα επαναγραφής ερωτημάτων $\mathcal{S}$, το οποίο παράγει ένα εξαντλητικό κι ελάχιστο σύνολο από μη περιττά ερωτήματα, έστω το UCQ. Καλούμε *απλή βάση των Aboxes* το πεπερασμένο σύνολο $\mathcal{B} = \{\mathcal{A} \xrightarrow{ans} \text{ans}(q, \mathcal{A})\}_{\forall \mathcal{A}: \mathcal{A} \in \mathcal{R}}$, όπου $\mathcal{R}$ είναι το σύνολο των ABoxes επαναγραφών του q και ans η βέβαιη απάντηση του q στο εκάστοτε $\mathcal{A}$.

Αλγόριθμος 1 Σχηματισμός ABox επαναγραφής και υπολογισμός απάντησης

Input: · CQ as first order logic clasue: q
Output: · (ABox $\xrightarrow{ans}$ ans): *aboxAns*

function CalcRewABoxAns(q)
  $qa \leftarrow q$.Body
  for each variable v of qa **do**
    Replace in qa: $v \leftarrow$ **new** atom a_v^i
  end for
  $ans \leftarrow$ calcSQLAns(q, qa) ▷ certain answer of q over qa
  return $(qa \xrightarrow{ans} ans)$
end function

Παράδειγμα 3.2.4. Έστω το DL-Lite$_R$-TBox $\mathcal{T} = \{\exists R \sqsubseteq A, S \sqsubseteq R\}$ και συζευκτικό ερώτημα $q : Q(x) \leftarrow A(x), R(x, y)$. Οι επαναγραφές του q είναι τα ερωτήματα $q_1 : Q(x) \leftarrow R(x, y)$, καθώς ο όρος $A(x)$ στο q είναι περιττός, και $q_2 : Q(x) \leftarrow S(x, y)$. Με βάση την παραπάνω διαδικασία, στο q_1, οι μεταβλητές x, y αντιστοιχίζονται στα άτομα a_x^1, a_y^1 αντίστοιχα και παράγεται το $\mathcal{A}_1 = \{R(a_x^1, a_y^1)\}$. Αντίστοιχα για το q_2, παράγεται το $\mathcal{A}_2 = \{S(a_x^2, a_y^2)\}$. Έτσι, η απλή βάση των Aboxes $\mathcal{B}$ είναι η εξής:

$$\mathcal{B} = \{\mathcal{A}_1 \xrightarrow{ans} \{a_x^1\},\ \mathcal{A}_2 \xrightarrow{ans} \{a_x^2\}\}$$

Στο παραπάνω παράδειγμα τυχαίνει η απλή βάση των Aboxes $\mathcal{B}$ να συμπίπτει με την βάση των ελάχιστων Aboxes. Κάτι τέτοιο όμως δεν ισχύει στη γενική περίπτωση, όπως στο επόμενο παράδειγμα.

Παράδειγμα 3.2.5. Έστω το DL-Lite$_R$-TBox $\mathcal{T} = \{B \sqsubseteq C\}$ και συζευκτικό ερώτημα $q : Q(x, y, z) \leftarrow A(x), R(x, y), B(y), R(x, z), C(z)$. Οι επαναγραφές του q, όπως δίνονται από τον Rapid, είναι τα ερωτήματα:

$$q_1 : Q(x, y, z) \leftarrow A(x), R(x, y), B(y), R(x, z), C(z),$$
$$q_2 : Q(x, y, z) \leftarrow A(x), R(x, y), B(y), R(x, z), B(z).$$

Η απλή βάση των Aboxes $\mathcal{B}$ είναι η εξής:

$$\mathcal{B} = \{\mathcal{A}_1 = \{A(a_x^1), R(a_x^1, a_y^1), B(a_y^1), R(a_x^1, a_z^1), C(a_z^1)\} \xrightarrow{ans} \{(a_x^1, a_y^1, a_z^1)\},$$
$$\mathcal{A}_2 = \{A(a_x^2), R(a_x^2, a_y^2), B(a_y^2), R(a_x^2, a_z^2), B(a_z^2)\} \xrightarrow{ans} \{(a_x^2, a_y^2, a_z^2)\}\}$$

Όμως, η βάση των ελάχιστων ABoxes στο συγκεκριμένο παράδειγμα δεν ταυτίζεται με την $\mathcal{B}$ διότι ans$(q, \mathcal{T}, \mathcal{A}_2 \setminus \{B(a_z^2)\}) = \{(a_x^2, a_y^2, a_y^2)\}$, διότι το a_y^2 απαντά και στη μεταβλητή z, εκτός από τη μεταβλητή y στο q.

Διαισθητικά, το σύνολο των Aboxes επαναγραφών, είναι οι πιθανοί κόσμοι, modulo ισομορφισμών, όπου ένα ερώτημα έχει απάντηση. Τα στοιχεία του είναι πιθανό να μην είναι ελάχιστα με την έννοια ότι δίνουν κενή απάντηση αν εξαλλειφθεί ένας ισχυρισμός, όπως φάνηκε από το Παράδειγμα 3.2.3, όπου υπάρχει απάντηση με λιγότερους ισχυρισμούς αν ένα άτομο απαντά σε περισσότερες από μία μεταβλητές και οι απαντήσεις strans που παράγονται δεν είναι πλήρεις. Καλύπτει όμως τις ανάγκες μίας βάσης η οποία χρησιμοποιείται για σχηματισμό μεγαλύτερων ABoxes, διότι εξασφαλίζει ότι:

- δεν περιέχει ισομορφικά Aboxes, αφού το σύστημα επαναγραφών S δεν παράγει δύο ισομορφικές επαναγραφές

- δεν περιέχει Aboxes, με περιττούς ισχυρισμούς, αφού αντιστοιχίζει διακριτές μεταβλητές σε διακριτά άτομα
- όλα τα ABoxes που περιέχει είναι ενδιαφέροντα, εκτός από ένα που αντιστοιχεί στο αρχικό ερώτημα
- έχει γραμμικό κόστος κατασκευής σε σχέση με το πλήθος των επαναγραφών

Ως σύστημα $\mathcal{S}$ στην παρούσα εργασία, χρησιμοποιήθηκε το σύστημα Rapid, το οποίο είναι ένα βελτιωμένο σύστημα επαναγραφών σε περιγραφική λογική DL-Lite$_R$ και το οποίο παράγει ένα εξαντλητικό κι ελάχιστο σύνολο από μη περιττές επαναγραφές ενός συζευκτικού ερωτήματος, όπως παρουσιάστηκε στο 2.

3.2.4 Εκτεταμένη απλή βάση των ABoxes

Τελικά, χρησιμοποιούμε μία εκτεταμένη βάση, η οποία περιέχει κάθε στοιχείο της απλής βάσης των ABoxes που περιγράψαμε παραπάνω, συνοδευόμενο από τη αντίστοιχη δυσκολία του. Όπως έγινε φανερό, τα Aboxes που ανήκουν στη βάση, δηλαδή τα Aboxes επαναγραφών, αποτελούνται επακριβώς από τους όρους των επαναγραφών. Έτσι, με βάση τους ορισμούς για δυσκολία ενός ABox κι ενός ερωτήματος επαναγραφής, το επόμενο θεώρημα είναι προφανές.

Θεώρημα 3.2.2. Έστω ένα ABox επαναγραφής $\mathcal{A}_i$ που προκύπτει με βάση μία επαναγραφή q_i ενός συζευκτικού ερωτήματος q, σε σχέση με ένα DL-Lite$_R$-TBox $\mathcal{T}$. Ισχύει ότι difficulty$_{q,\mathcal{T}}(\mathcal{A}_i) =$ rewdifficulty$_{q,\mathcal{T}}(q_i)$.

Επομένως, αρκεί να βρεθεί η δυσκολία των επαναγραφών ενός ερωτήματος ώστε εξάγουμε άμεσα τη δυσκολία ενός Abox επαναγραφής. Η εξαγωγή της μετρικής αυτής παρουσιάζεται αναλυτικά στην επόμενη παράγραφο, απλά εισάγουμε εδώ τον όρο για να ορίσουμε τον αλγόριθμο παραγωγής της εκτεταμένης απλής βάσης των Aboxes. Αρχικά ας ορίσουμε την εκτεταμένη απλή βάση των Aboxes.

Ορισμός 3.2.4. Έστω μία απλή βάση των ABoxes $\mathcal{B}$, σε σχέση με ένα TBox $\mathcal{T}$ κι ένα ερώτημα q. Ορίζουμε ως *εκτεταμένη απλή βάση των ABoxes (extended Abox-base)*, σε σχέση με το $\mathcal{T}$ και το q, το σύνολο $\{(B_i, \text{difficuty}_{q,\mathcal{T}}(B_i))\}_{B_i \in \mathcal{B}}$.

Άμεσα προκείπτει και ο ορισμός της εκτεταμένης βάσης των επαναγραφών, η οποία χρησιμοποιείται στον αλγόριθμο παραγωγής της εκτεταμένης απλής βάσης των ABoxes.

Ορισμός 3.2.5. Έστω ένα σύστημα επαναγραφών $\mathcal{S}$ με τις συνήθεις ιδιότητες, ένα TBox $\mathcal{T}$ κι ένα ερώτημα q. Ορίζουμε ως *εκτεταμένη απλή βάση των επαναγραφών (extended rewriting-base)*, σε σχέση με το $\mathcal{T}$ και το q, το σύνολο $\{(q_i, \text{rewdifficuty}_{q,\mathcal{T}}(q_i))\}_{q_i \in \mathcal{R}}$, όπου $\mathcal{R}$ το σύνολο επαναγραφών του q από το σύστημα $\mathcal{S}$.

3.2.5 Σχηματισμός εκτεκατμένης απλής βάσης των ABoxes

Με την εισαγωγή όλων των παραπάνω όρων, καταλήγουμε στον αλγόριθμο παραγωγής της εκτεταμένης απλής βάσης των ABoxes, όπως αυτός υλοποιήθηκε. Αρχικά, όπως έχει ειπωθεί, ο αλγόριθμος αυτός υλοποιήθηκε με βάση το σύστημα Rapid, κι επομένως σχεδιάστηκε ώστε να

συμφωνεί με αυτό. Για το λόγο αυτό, το ερώτημα εισόδου q και το TBox εισόδου $\mathcal{T}$ είναι εκφρασμένα σε προτασιακή λογική πρώτης τάξης, με τη μορφή δηλαδή που τα επεξεργάζεται ο Rapid, όπως επεξηγήκε στο 2. Επίσης, θεωρούμε προς το παρόν "μαύριο κουτί" τη δημιουργία του γράφου ξετυλιγμάτων και τον υπολογισμό των δυσκολιών-επαναγραφών (συναρτήσεις CalcRewDifficulty, createUnfoldingGraph στον αλγόριθμο), καθώς οι διαδικασίες αυτές απαιτούν ιδιαίτερη επεξήγηση που θα γίνει στην επόμενη παράγραφο.

Όσο για τη διαδικασία δημιουργίας της βάσης, δεν είναι άλλη από αυτή που περιγράφηκε παραπάνω για τον ορισμό των ABoxes επαναγραφής: για κάθε επαναγραφή qc του αρχικού ερωτήματος q, αντικαθιστούμε κάθε μεταβλητή με ένα διακριτό άτομο στο qc. Το σώμα του ερωτήματος που πρκύπτει αποτελεί το Abox επαναγραφής. Η δυσκολία αυτού, ταυτίζεται με τη δυσκολία επαναγραφής του q_i, όπως ειπώθηκε στο Θεώρημα 3.2.1. Επίσης, αν η μεταβλητή είναι μεταβλητή απάντησης, τότε το άτομο που της αντιστοιχείται προστίθεται στο σύνολο ans που είναι η προφανής απάντηση.

Αλγόριθμος 2 Σχηματισμός εκτεταμένης απλής βάσης των ABoxes

Input: · CQ as first order logic clasue: q
· DL-Lite$_R$-TBox as a set of first order logic clauses: $\mathcal{T}$
Output: · Extended simple ABox-base: $\mathcal{B}_A$

function CalcExtSimpleAboxBase(q, $\mathcal{T}$)
  $\mathcal{B}_A \leftarrow \{\}$
  $\mathcal{G} \leftarrow$ createUnfoldingGraph($\mathcal{T}$)
  $\mathcal{B}_R \leftarrow$ CalcRewDifficulty($q, \mathcal{G}$)
  for each $\mathcal{B}_{R,i}$ in $\mathcal{B}_R$ **do**
    diff $\leftarrow \mathcal{B}_{R,i}$.Difficulty
    $qc \leftarrow \mathcal{B}_{R,i}$.Rewriting
    $\{\mathcal{A} \xrightarrow{ans} ans\} \leftarrow$ CalcRewAboxAns(qc)
    $\mathcal{B}_A \leftarrow \mathcal{B}_A \cup \{(\mathcal{A} \xrightarrow{ans} ans, diff)\}$
  end for
  return $\mathcal{B}_A$
end function

3.3 Ερωτήματα εκπαίδευσης

Η επιλογή των ερωτημάτων για την εκπαίδευση, σε σχέση με ένα TBox $\mathcal{T}$, αποτελεί κομβικό σημείο της εργασίας. Τα ερωτήματα αυτά, είναι τα μόνα ερωτήματα πάνω στα οποία θα εκπαιδευτεί το νευρωνικό δίκτυο, επομένως χρήζει μεγάλης προσοχής η επιλογή τους. Σύμφωνα με τους παραπάνω ορισμούς, υποψιαζόμαστε ότι ζητάμε ενδιαφέροντα ερωτήματα τα οποία απαιτούν εξαγωγές συμπερασμών για να απαντησθούν σωστά. Η εκτενής ανάλυση που έγινε στο 2 για το SyGENiA, προϊδεάζει για την επιλογή μας αυτή. Η βάση ερωτημάτων QTB που παράγει το SyGENiA είναι πλήρως αντιπροσωπευτική του TBox για το οποίο την παράγει και εξασφαλίζει ότι ενεργοποιούνται όλες οι πιθανές εξαγωγές συμπερασμών που μπορούν να προκύψουν από οποιοδήποτε ερώτημα. Επίσης, η QTB είναι πεπερασμένη. Για τους λόγους αυτούς, ως σύνολο ερωτημάτων εκπαίδευσης, σε σχέση με το $\mathcal{T}$, επιλέγεται το QTB που παράγει το SyGENiA με είσοδο το $\mathcal{T}$.

3.4 Εξαγωγή δυσκολίας

3.4.1 Γράφος ξετυλιγμάτων

Για την εξαγωγή της μετρικής δυσκολίας που αναφέρθηκε παραπάνω, ουσιαστικά αναζητούμε τη διαδοχή των λογικών αξιωμάτων του TBox που έχουν χρησιμοποιηθεί για τις εξαγωγές συμπερασμών, μέχρι να θεωρηθεί ένα άτομο ως άτομο απάντησης. Για το σκοπό αυτό, δημιουργούμε το *γράφο λογικών ξετυλιγμάτων (unfolding graph)* του TBox, αφού πρώτα φέρουμε τα αξιώματα σε μορφή λογικών προτάσεων πρώτης τάξης, και συγκεκριμένα στη μορφή του Πίνακα 2.2. Η μετατροπή αυτή γίνεται σε γραμμικό χρόνο ως προς το μέγεθος του TBox και μπορεί να παραχθεί με χρήση του εργαλείου Rapid.

Ο γράφος αυτός είναι κατευθυνόμενος και, ουσιαστικά, αναπαριστά όλες τις σχέσεις ξετυλίγματος των εννοιών και των ρόλων που ορίζονται από τα λογικά αξιώματα. Συγκεκριμένα, αν δύο έννοιες συνδέονται με μία ακμή, τότε υπάρχει αξίωμα υπαγωγής που στο δεξί του μέρος έχει την αφετηρία και στο αριστερό το τέλος της ακμής, και αντίστροφα, για κάθε αξίωμα υπαγωγής, υπάρχει ζεύγος των αντίστοιχων κόμβων που συνδέονται με μία ακμή. Επίσης, η κάθε ακμή έχει μία ετικέτα "True" ή "False" που δηλώνει ανάστροφη σχέση ή όχι αντίστοιχα. Ο γράφος αυτός ονομάζεται γράφος ξετυλιγμάτων, διότι οι κατευθύνσεις των ακμών σχετίζονται με τη διαδικασία επαναγραφής, καθώς, για παράδειγμα, για τον κανόνα $A \sqsubseteq B$ υπάρχει ακμή από το u_B στο u_A, το οποίο μοιάζει με την πρόταση *"Το B επαναγράφεται ως A."*

Ορισμός 3.4.1. Έστω ένα DL-Lite-TBox $\mathcal{T}$. Ορίζουμε ως γράφο ξετυλιγμάτων του $\mathcal{T}$, ένα κατευθυνόμενο γράφο ο οποίος περιγράφεται ως εξής:

- *Κόμβοι*: Ένας κόμβος u_A για κάθε ατομική έννοια ή ατομικό ρόλο A που εμφανίζεται στο $\mathcal{T}$, εκτός αν εμφανίζεται μόνο σε αξίωμα $B \sqsubseteq \exists R.A$.
- *Ακμές*: Μία προσανατολισμένη ακμή $u_A \xrightarrow{False} u_B$, για κάθε πρόταση της μορφής:
  - $A(x) \leftarrow B(x)$, που είναι ισοδύναμο με το αξίωμα $B \sqsubseteq A$
  - $A(x) \leftarrow B(x, y)$, που είναι ισοδύναμο με το αξίωμα $\exists B \sqsubseteq A$
  - $A(x, f_A^B(x)) \leftarrow B(x)$, που είναι ισοδύναμο με το αξίωμα $B \sqsubseteq \exists A$
  - $A(x, f_{A,C}^B(x)) \leftarrow B(x)$, που είναι μέρος κάποιου αξιώματος $B \sqsubseteq \exists A.C$
  - $A(x, y) \leftarrow B(x, y)$, που είναι ισοδύναμο με το αξίωμα $B \sqsubseteq A$ ή το $B^- \sqsubseteq A^-$

  και μία προσανατολισμένη ακμή $u_A \xrightarrow{True} u_B$ για κάθε πρόταση της μορφής:
  - $A(x) \leftarrow B(y, x)$, που είναι ισοδύναμο με το αξίωμα $\exists B^- \sqsubseteq A$
  - $A(f_{A^-}^B(x), x) \leftarrow B(x)$, που είναι ισοδύναμο με το αξίωμα $B \sqsubseteq \exists A^-$
  - $A(f_{A^-,C}^B(x), x) \leftarrow B(x)$, που είναι μέρος κάποιου αξιώματος $B \sqsubseteq \exists A^-.C$
  - $A(x, y) \leftarrow B(y, x)$, που είναι ισοδύναμο με το αξίωμα $B \sqsubseteq A^-$ ή το $B^- \sqsubseteq A$

  όπου A, B ατομικές έννοιες ή ατομικοί ρόλοι, ανάλογα με τις παραπάνω περιπτώσεις.

Όπως γίνεται αντιληπτό, τα αξιώματα της μορφής $A(f(x)) \leftarrow B(x)$ παραλείπονται. Αυτό συμβαίνει γιατί περιέχουν μόνο συναρτησιακή μεταβλητή στην κεφαλή τους, που σημαίνει ότι εξαρτώνται εξ' ολοκλήρου από κάποιο άλλο αξίωμα, και ότι μόνα τους δεν έχουν ισχύ. Ο λόγος που εισάγονται είναι η διαδικασία της συρρίκνωσης, καθώς κάθε ζεύγος με κοινή f μπορεί να συρρικνωθεί κατά την παραγωγή επαναγραφών στον αριστερό όρο των f-αξιωμάτων, όπως έχει ήδη αναφερθεί στο 2.

Η συνάρτηση createUnfoldingGraph που χρησιμοποιήθηκε στον αλγόριθμο σχηματισμού της απλής εκτεταμένης βάσης των ABoxes, πραγματοποιεί ακριβώς αυτά που περιγράφει ο ορισμός του ίδιου του γράφου, με βάση το TBox εισόδου.

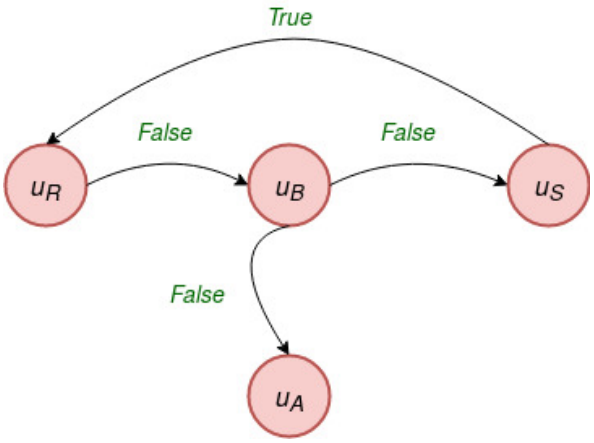
Παράδειγμα 3.4.1. Έστω το TBox $\mathcal{T}$:

$$\mathcal{T} = \{A \sqsubseteq B, B \sqsubseteq \exists R.C, R \sqsubseteq S^{-}, \exists S \sqsubseteq B\},$$

το οποίο, με βάση τον Πίνακα 2.2, μετατρέπεται στο σύνολο $\mathcal{P}$ προτάσεων λογικής πρώτης τάξης:

$$\begin{aligned}\mathcal{P} = \{&B(x) \leftarrow A(x),\\ &R(x, f^{B}_{R,C}(x)) \leftarrow B(x),\\ &S(x, y) \leftarrow R(y, x),\\ &B(x) \leftarrow S(x, y)\}\end{aligned}$$

Ο αντίστοιχος γράφος ξετυλιγμάτων φαίνεται στο ακόλουθο σχήμα:



Σχήμα 3.1

Πρατηρούμε ότι είναι δυνατό ο γράφος να έχει κύκλους.

3.4.2 Υπολογισμός δυσκολίας δύο όρων

Με βάση το γράφο ξετυλιγμάτων, μπορούμε να βρούμε σε γραμμικό χρόνο ως προς το σύνολο εννοιών και ρόλων του TBox, την ελάχιστη δυσκολία επαναγραφής ενός όρου σε κάποιον άλλο. Η διαδικασία αυτή αποτυπώνεται στον αλγόριθμο CalcDifficultyFromGraph και βασίζεται στην κατά πλάτος αναζήτηση πάνω στο γράφο ξετυλιγμάτων, με αρχικό κόμβο τον αρχικό όρο και τελικό κόμβο τον όρο-επαναγραφή. Επίσης, αν η σχέση των όρων είναι ανάστροφη, δηλαδή για παράδειγμα αν ο αρχικός κόμβος αντιστοιχεί στον όρο $A(x, y)$ και ο τελικός στον $B(y, x)$ ή στον $B(y)$, ή αντίστροφα, αναζητούμε μονοπάτι που ξεκινώντας από "True" καταλήγουμε σε "False", με βάση τις ετικέτες αληθείας των ακμών. Η ανάστροφη σχέση υπολογίζεται από τον αλγόριθμο areInverseRelated και δίνεται ως είσοδος στον αλγόριθμο CalcDifficultyFromGraph. Τέλος, αν δεν

υπάρχει η ζητούμενη διαδρομή στο γράφο, επιστρέφεται δυσκολία με κενό μονοπάτι και κόστος -1. Ο αλγόριθμος αυτής της διαδικασίας ακολουθεί.

Ο αλγόριθμος calcDifficultyFromGraph είναι μία BFS διαδικασία και γι' αυτό εξασφαλίζει ελάχιστα μονοπάτια, και άρα ελάχιστη δυσκολία από ένα όρο σε ένα άλλο. Η συνάρτηση calcPath υπολογίζει το μονοπάτι από το πρώτο όρισμα στο δεύτερο, μέσω της αντιστοίχισης $parents$. Μέσα από το μονοπάτι, μπορούμε να εξάγουμε τον αριθμό των ξετυλιγμάτων αλλά και πόσα από αυτά αντιστοιχούν σε υπαρξιακά ή ανάστροφα αξιώματα και να εξάγουμε το αντίστοιχο κόστος. Παρατηρούμε ότι ουσιαστικά κάθε κόμβος κατά την αναζήτησης BFS, δεν είναι απλά ένα όνομα μίας έννοιας ή ενός ρόλου, αλλά ρόλο παίζει και η ετικέτα που έχει εκείνη τη στιγμή με βάση τη διαδρομή που έχει ακολουθηθεί. Ακολουθεί ένα παράδειγμα για να γίνει κατανοητή η λειτουργία του αλγορίθμου.

Παράδειγμα 3.4.2. Έστω το TBox $\mathcal{T}$ και ο γράφος ξετυλιγμάτων $\mathcal{G}$ του Παραδείγματος 3.4.1. Αν ζητώ την απόσταση των όρων:

- $R(x,y) \rightarrow S(y,x)$,
  τότε invRelated$(R(x,y), S(x,y))$ = "False", επομένως η ουρά ξεκινάει με το στοιχείο $(u_R$, "False") και αναζητούμε το στοιχείο $(u_S$, "False"). Σε πρώτο βήμα, εισάγεται στην $queue$ το $(u_B$, "False") και καθώς αυτό δεν αποτελεί το στόχο συνεχίζουμε στο επόμενο στοιχείο της ουράς. Τότε, εισάγεται στην $queue$ το $(u_A$, "False") και αφού δεν αποτελεί το στόχο, συνεχίζουμε και εισάγεται το $(u_S$, "False"). Αυτό ισούται με το στόχο άρα βγαίνουμε από το βρόχο. Υπολογίζεται το μονοπάτι $\{R, B, S\}$ και τελικά, με μοναδιαία βάρη, επιστρέφεται η δυσκολία difficulty$_{R(x,y) \leftarrow S(x,y)} = (2+2+0, \{R, B, S\})$.

- $R(x,y) \rightarrow R(y,x)$,
  τότε invRelated$(R(x,y), R(y,x))$ = "True", επομένως η ουρά ξεκινάει με το στοιχείο $(u_R$, "False") και αναζητούμε το στοιχείο $(u_R$, "True"). Εδώ γίνεται αντιληπτό γιατί δεν μας αφορά απλά το όνομα αλλά και η ετικέτα, γιατί διαφορετικά η απόσταση αυτή στο γράφο θα ήταν μηδενική. Ομοίως με προηγουμένως φτάνουμε στο σημείο η $queue$ να περιέχει τα $(u_A$, "False") και $(u_S$, "False"). Το πρώτο δεν έχει παιδιά, οπότε αφαιρείται από την $queue$ και συνεχίζουμε με το δεύτερο. Αυτό δίνει το παιδί $(u_R$, "True") το οποίο είναι ο στόχος. Υπολογίζεται το μονοπάτι $\{R, B, S, R\}$ και τελικά, με μοναδιαία βάρη, επιστρέφεται η δυσκολία difficulty$_{R(x,y) \leftarrow R(y,x)} = (3+2+1, \{R, B, S, R\})$.

3.4.3 Υπολογισμός δυσκολίας δύο ερωτημάτων

Στη συνέχεια παρουσιάζονται οι αλγόριθμοι εξαγωγής των δυσκολιών - επαναγραφών CalcUnfoldingDifficulty και CalcShrinkingDifficulty, οι οποίοι εκτελελούνται κατά την παραγωγή των επαναγραφών, δηλαδή παρεμβαίνουν στον αλγόριθμο Rapid. Συγκεκριμένα, ένας από αυτούς τους αλγορίθμους, εκτελείται κάθε φορά που παράγεται κάποια επαναγραφή και ακριβώς πριν προστεθεί στο σύνολο αποτελεσμάτων του Rapid, ώστε η επαναγραφή να είναι στην επιθυμητή μορφή, δηλαδή χωρίς περιττούς όρους. Όπως αναφέρθηκε στο [2], ο Rapid πρώτα παράγει όλες τις επαναγραφές που προκύπτουν από όλες τις πιθανές συρρικνώσεις και στη συνέχεια, πραγματοποιεί σε αυτές και στο αρχικό ερώτημα παράγει όλες τις επαναγραφές που προκύπτουν από

Αλγόριθμος 3 Υπολογισμός δυσκολίας μεταξύ δύο ονομάτων μέσω του γράφου ξετυλιγμάτων

Input: · Source atomic concept or role: A
· Destination atomic concept or role: B
· Unfolding graph: $\mathcal{G}$
· Inverse-relation boolean flag A-B: *inv*
Output: · Rewriting difficulty of atomic: *diff*

1: **function** CalcDifficultyFromGraph(A, B, *inv*, $\mathcal{G}$)
2: **if** u_A or u_B not in $\mathcal{G}$ **then**
3: **return** $(-1, \emptyset)$
4: **else if** A, B are indent and *inv* is "False" **then**
5: **return** $(0, \{A\})$
6: **end if**
7: *found* $\leftarrow$ "False"
8: *parents*[(A, "False")] $\leftarrow$ (A, "False") ▷ map *parents* to specify source atomic
9: *queue* $\leftarrow$ (u_A, "False")
10: **while** *queue* is not empty **do**
11: (u_X, label) $\leftarrow$ *queue*.top
12: Remove top from *queue*
13: **for each** child $u_{X'}$ of u_X from $\mathcal{G}$ that are connected with edge of label$(u_X, u_{X'})$ **do**
14: label' $\leftarrow$ label $\oplus$ label$(u_X, u_{X'})$
15: **if** (X', label') is not already visited **then**
16: *parents*[(X', label')] $\leftarrow$ (X, label)
17: **if** X' is B and label' equals *inv* **then**
18: *found* $\leftarrow$ "True"
19: **break**
20: **else**
21: *queue* $\leftarrow$ *queue* $\cup$($u_{X'}$, label')
22: **end if**
23: **end if**
24: **end for**
25: **if** *found* **then**
26: **break**
27: **end if**
28: **end while**
29: **if** *found* **then**
30: *path* $\leftarrow$ calcPath((u_A, "False"), (u_B, *inv*), *parents*)
31: (*unfNum, existNum, invNum*) $\leftarrow$ takeParametersFromPath(*path*)
32: **return** (*unfNum* $\cdot w_{unf}$ + *existNum* $\cdot\ w_{exist}$ + *invNum* $\cdot\ w_{inv}$, *path*)
33: **else**
34: **return** $(-1, \emptyset)$
35: **end if**
36: **end function**

αξιώματα ξετυλίγματος. Επομένως, υπάρχουν δύο σημεία που πρέπει να υπολογιστεί η δυσκολία μίας επαναγραφής:

- κατά την παραγωγή επαναγραφής από αξιώματα συρρίκνωσης
- κατά την παραγωγή επαναγραφής από αξίωμα ξετυλίγματος

Έτσι, οι αλγόριθμοι που εφαρμόζονται σε κάθε περίπτωση διαφέρουν. Συγκεκριμένα, ο αλγόριθμος για την εξαγωγή δυσκολίας επαναγραφής στην πρώτη περίπτωση (CalcShrinkingDifficulty), πραγματοποιεί μια προεργασία στο συρρικνωμένο ερώτημα πριν εφαρμόσει τον αλγόριθμο εξαγωγής δυσκολίας απλού ξετυλίγματος (CalcUnfoldingDifficulty). Οι αλγόριθμοι αναγράφονται και αναλύονται παρακάτω.

3.4.3.1 Υπολογισμός δυσκολίας από ξετυλίγματα

Αρχικά, αυτό που κάνει ο αλγόριθμος CalcUnfoldingDifficulty είναι να παίρνει ένα ερώτημα και μία επαναγραφή του και για κάθε όρο του ερωτήματος να βρίσκει τους όρους στο ερώτημα επαναγραφής, στους οποίους είναι πιθανό να έχει επαναγραφεί, και στη συνέχεια να επιλέγει τη μικρότερη, ως προς το κόστος, δυσκολία επαναγραφής προς κάθε τέτοιο όρο. Η διαδικασία επαναλαμβάνεται για κάθε όρο του ερωτήματος και τελικά, σαν δυσκολία επαναγραφής ολόκληρου του ερωτήματος, επιλέγεται η μέγιστη δυσκολία αυτών των ελαχίστων.

Παράδειγμα 3.4.3. Έστω το TBox $\mathcal{T}$ και ο γράφος ξετυλιγμάτων $\mathcal{G}$ του Παραδείγματος 3.4.1. Έστω το ερώτημα $q : Q(x) \leftarrow R(x, y), S(y, z), A(z)$ και η επαναγραφή $q_i : Q(x) \leftarrow B(x), S(y, z), A(z)$. Ο πρώτος όρος του q, $R(x, y)$, θα αντιστοιχηθεί στον $B(x)$ επειδή είναι ο κοντινότερος αφού έχει κόστος $1 + 1$, ενώ η δυσκολία από τον όρο $S(y, x)$ έχει κόστος $5 + 4 + 1$. Η δυσκολία από τον όρο $A(z)$ δεν λαμβάνεται υπόψιν γιατί δεν υπάρχουν κοινές μεταβλητές. Οι άλλοι δύο όροι του q, υπάρχουν στο q_i. Επομένως η δυσκολία έχει κόστος 2 και μονοπάτι $\{R, B\}$.

Σημειώνεται, ο αλγόριθμος CalcUnfoldingDifficulty μπορεί να φαίνεται μη ακριβής, από την άποψη ότι μπορεί να αντιστοιχίζει κάποιον όρο του αρχικού ερωτήματος όχι σε αυτόν που έχει πραγματικά μετατραπεί μέσα από συμπερασμούς, αλλά σε κάποιον άλλο με κοινή μεταβλητή, στον οποίο αντιστοιχίζεται κάποιος άλλος όρος. Ένα παράδειγμα που μπορεί να σκεφτεί κάποιος, στο οποίο φαίνεται αυτή η ανακρίβεια, είναι το παρακάτω.

Παράδειγμα 3.4.4. Έστω ο γράφος ξετυλιγμάτων του Σχήματος 3.2, που αντιστοιχεί σε ένα DL-Lite$_R$-TBox, και τα εξής ερωτήματα:

$q : Q(x) \leftarrow A(x), B(x)$
$q_1 : Q(x) \leftarrow E(x), C(x)$

όπου το q_1 είναι μία επναγραφή του q. ο αλγόριθμος CalcUnfoldingDifficulty θα ψάξει να αντιστοιχίσει κάθε όρο του q σε όρους του q_i με κοινές μεταβλητές και τη μικρότερη δυσκολία, επομένως θα κάνει τις εξής αντιστοιχίσεις:

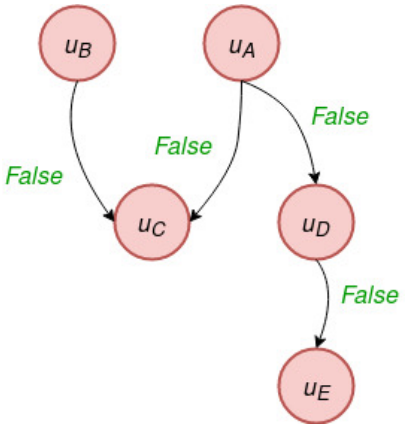
$A(x) \rightarrow C(x)$
$B(x) \rightarrow C(x)$

και τελικά η δυσκολία επαναγαφής θα είναι $\mathit{diff} = (1, \{A, C\})$, ενώ στην πραγματικότητα είναι $\mathit{diff} = (2, \{A, D, E\})$ γιατί αν και οι δύο όροι είχαν αναλυθεί στον όρο $C(x)$, τότε θα είχαμε έναν όρο στο ερώτημα επαναγραφής, επομένως ο όρος $A(x)$ έχει αντιστοιχηθεί στο $E(x)$

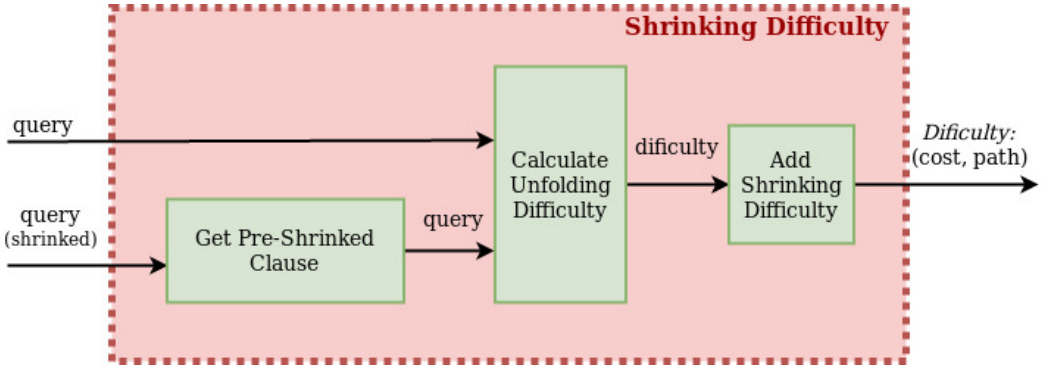
Αλγόριθμος 4 Υπολογισμός δυσκολίας από αξιώματα ξετυλίγματος, με βάση το γράφο ξετυλιγμάτων

Input: · CQ as a first order logic clause: q
· CQ as a first order logic clause (rewriting): q_i
· Unfolding graph: $\mathcal{G}$
Output: · Rewriting difficulty: *diff*

function CalcUnfoldingDifficulty(q, q_i, $\mathcal{G}$)
  if q, q_i are indent **then**
    return $(0, \emptyset)$
  end if
  min $\leftarrow$ (MAX_VALUE, $\emptyset$)
  max $\leftarrow (-1, \emptyset)$
  for each atomic a in body of q **do**
    if a is contained in q_i **then**
      min $\leftarrow (0, \emptyset)$ ▷ Same atomic found, so zero difficulty
    else
      min $\leftarrow$ (MAX_VALUE, $\emptyset$)
      for each atomic a_i in body of q_i **do**
        if a, a_i have variables in common **then**
          cur $\leftarrow$ CalcDifficultyFromGraph$(a, a_i,$ areInverseRelated$(a, a_i))$
          if *cur*.Cost≥ 0 και *cur*.Cost<m*min*.Cost **then**
            min $\leftarrow$ *cur* ▷ If *cur* is reasonable and less than *min*
          end if
        end if
      end for
    end if
    if *min*.Cost $<$ MAX_VALUE και *min*.Cost $>$ *max*.Cost **then**
      max $\leftarrow$ *min*
    end if
  end for
  return *max*
end function



Σχήμα 3.2



Σχήμα 3.3: Σχηματική αναπαράσταση αλγορίθμου εξαγωγής δυσκολίας σε συρρίκνωσης

Όμως, το σύστημα Rapid εξασφαλίζει ότι δεν παράγονται επαναγραφές που μπορούν να προκαλέσουν αυτή τη "σύγχιση" στα αποτελέσματα του αλγορίθμου. Πιο συγκεκριμένα, επαναγραφές που περιέχουν πάνω από δύο όρους στους οποίους μπορεί να καταλήξει με εξαγωγές συμπερασμών ένας όρος του αρχικού ερωτήματος, είναι περριτές κι επομένως δεν παράγονται από τον Rapid.

Θεώρημα 3.4.1. Έστω ένα DL-Lite$_R$-TBox $\mathcal{T}$, ένα ερώτημα q και μία επαναγραφή αυτού q_i, που προκύπτει από ένα σύστημα επαναγραφών $\mathcal{S}$ με τις γνωστές επιθυμητές ιδιότητες. Ισχύει ότι κάθε όρος του q μπορεί να αναλυθεί σε ένα μόνο όρο του q_i, με βάση τα αξιώματα του q.

Απόδειξη: Έστω ότι οι όροι Q_k, Q_l είναι δύο όροι του q που αναλύονται στους όρους Q_m, Q_n του q_i, αντίστοιχα, και ο όρος Q_k μπορεί να αναλυθεί και στον Q_n. Επομένως, υπάρχει ακόμα μία επαναγραφή στο σύνολο επαναγραφών του q, έστω $q_j = q_i \setminus Q_m$, στο οποίο ο όρος Q_k έχει αναλυθεί στον όρο Q_n. Τότε, οι απαντήσεις στο q_i είναι πιο ειδικές από τις απαντήσεις του q_j πάνω σε μία βάση, επομένως για να παράγουμε όλες τις απαντήσεις του q δε χρειάζεται να ρωτήσουμε και το q_i, εφόσον ρωτώντας το q_j παίρνω τις απαντήσεις του q_i. Αυτό σημαίνει ότι το q_i είναι περιττό ερώτημα στο σύνολο επαναγραφών που έχει παράγει το $\mathcal{S}$, άτοπο διότι το $\mathcal{S}$ δεν παράγει περιττές επαναγραφές.

Μέσα από αυτό το θεώρημα φαίνεται η χρησιμότητα των βελτιστοποιήσεων του Rapid, ώστε να παράγει αποδοτικά ένα μη περιττό σύνολο επαναγραφών. Με αυτόν την παραπάνω απόδειξη αποδεικνύεται η ορθότητητα του αλγορίθμου CalcUnfoldingDifficulty, δηλαδή ότι όντως υπολογίζει την ελάχιστη δυσκολία εξαγωγής του συγκεκριμένου ερωτήματος.

3.4.3.2 Υπολογισμός δυσκολίας από συρρικνώσεις

Έτσι, καταλήγουμε στον τελευταίο αλγόριθμο εξαγωγής δυσκολίας, δηλαδή σε αυτόν που αφορά την επαναγραφή μέσω της διαδικασίας συρρίκνωσης. Αρχικά, βρίσκουμε το ερώτημα q_i' στο οποίο οποίο καταλήγει το q με κανόνες ξετυλίγματος και μετά εφαρμόζεται ο κανόνας συρρίκνωσης. Δηλαδή το q_i' είναι το ερώτημα ακριβώς πριν γίνει η συρρίνωση με βάση μία δυάδα αξιωμάτων. Ο τρόπος με τον οποίο γίνεται αυτό είναι, είναι να αντικαθιστούμε στο q_i τον όρο συρρίκνωσης με το ζεύγος όρων που τον εξάγουν, με τις κατάλληλες μεταβλητές. Αυτό το κάνει η συνάρτηση calcPreShrinkClause στον αλγόριθμο CalcShrinkingDifficulty. Στη συνέχεια, όντας σίγουροι πλέον ότι έχουν γίνει μόνο ξετυλίγματα από το q στο q_i', υπολογίζουμε τη δυσκολία επαναγραφής από ξετυλίγματα με βάση τον αλγόριθμο CalcUnfoldingDifficulty και, ύστερα, για την

εξαγωγή της τελικής δυσκολίας επαναγραφής του q_i, προσθέτουμε τη δυσκολία που απαιτεί αυτή η μία συρρίκνωση που εξήγαγε το q_i από το q_i'.

Αλγόριθμος 5 Υπολογισμός δυσκολίας σε περίπτωση συρρίκνωσης στην επαναγραφή

Input: · CQ as a first order logic clause: q
· CQ as a first order logic clause (rewriting): q_i
· Unfolding graph: $\mathcal{G}$
Output: · Rewriting difficulty: *diff*

1: **function** CalcShrinkingDifficulty(q, q_i, $\mathcal{G}$)
2: $q_i' \leftarrow$ calcPreShrinkClause(q_i)
3: (*cost*, *path*) $\leftarrow$ CalcUnfoldingDifficulty($q, q_i', \mathcal{G}$)
4: **return** (*cost* + w_{shr}, *path* $\cup$ {*shrinking atomic*})
5: **end function**

Σημειώνεται ακόμα, ότι κατά τη διαδικασία επαναγραφής, το q δεν είναι πάντα το αρχικό ερώτημα εισόδου στο σύστημα, άλλα μπορεί να είναι και κάποια επαναγραφή που έχει προκύψει εσωτερικά και που ξετυλίγεται περαιτέρω. Η τελική δυσκολία του q_i θα συμπεριλάβει τη δυσκολία του q σε αυτή την περίπτωση.

3.5 Σχηματισμός συνόλου δεδομένων εκπαίδευσης

Το τελικό σύνολο δεδομένων εκπαίδευσης, για κάθε ερώτημα, ζητούμε να έχει τα εξής χαρακτηριστικά:

- να μην περιέχει ισομορφικά Aboxes
- τα ABoxes που περιέχει, να μην είναι υλοποιήσεις του ερωτήματος
- τα ABoxes που περιέχει, να μην περιέχουν περιττούς όρους

Τα χαρακτηριστικά αυτά ισχύουν για την απλή βάση των ABoxes, όπως αναλύσαμε στην προηγούμενη παράγραφο. Επομένως, ένας απλός συνδυασμός των ABoxes της απλή βάσης όπου δεν υπάρχουν κοινά άτομα μεταξύ των συνδυαζόμενων στοιχείων, παράγει ABoxes με τα παραπάνω χαρακτηριστικά. Αυτό, βέβαια, δε σημαίνει ότι η ταυτοποίησης κάποιων ατόμων από τα ABoxes που συνδυάζονται αναιρεί απαραίτητα κάποιο από τα τρία επιθυμητά χαρακτηριστικά. Για να μην συμβαίνει αυτό, απαιτείται αρκετή μελέτη για την εξαγωγή κάποιων γενικών ποϋποθέσεων που πρέπει να ισχύουν ώστε να μην αναιρούνται τα τρία χαρακτηριστικά, ή έστω κάποια από αυτά. Από την άλλη, μπορούμε να χαλαρώσουμε τις επιθημητές συνθήκες και να αρκεστούμε στη δεύτερη, καθώς οι άλλες δύο απλώς προσθέτουν περιττή πληροφορία, η οποία σε ένα σύνολο δεδομένων εκπαίδευσης δεν έχει αρνητικές επιδράσεις, πέρα από την χωρίς λόγο αύξηση του όγκου δεδομένων. Η δεύτερη συνθήκη από την άλλη, κάνε προφανή την απάντηση σε ένα ABox κι έτσι χάνει αυτό την ποιότητά του σε σχέση με τις εξαγωγές συμπερασμών που κρύβει. Η μελέτη ως προς αυτές τις κατευθύνσεις αφήνεται ως μελλονικό έργο, και στην παρούσα εργασία αρκούμαστε στην παραγωγή ενός απλού συνόλου των ABoxes της απλής βάσης, χωρίς κοινά άτομα μεταξύ τους, το οποίο ονομάσαμε DatAbox. Ο χρήστης καθορίζει πόσες φορές συνδυάζονται τα στοιχεία του DatAbox μεταξύ τους, μέσω της μεταβλητής εισόδου n.

Αλγόριθμος 6 Δημιουργία απλού συνόλου δεδομένων εκπαίδευσης

Input: · CQ as a first order logic clause: q
· DL-Lite$_R$-TBox as a set of first order logic clauses: $\mathcal{T}$
· Times of combination: n
Output: · Set of (ABox, difficulty): $\mathcal{D}$

function CalcDatAbox(q, n, $\mathcal{T}$)
  Initialize $\mathcal{D} \leftarrow$ calcExtSimpleABoxBase$(q, \mathcal{T})$
  for ($comb \leftarrow 0$, $comb < n$, $comb \leftarrow comb$+1) **do**
    $size \leftarrow \mathcal{D}$.Size
    for ($i \leftarrow 0$, $i <$*size*, $i \leftarrow i + 1$) **do**
      for ($j \leftarrow 0$, $j < i$, $j \leftarrow j + 1$) **do**
        $\mathcal{B}_{i+j} \leftarrow$ SimpleJoin$(\mathcal{D}[i], \mathcal{D}[j])$
        $\mathcal{D} \leftarrow \mathcal{D} \cup \{\mathcal{B}_{i+j}\}$
      end for
    end for
  end for
  return $\mathcal{D}$
end function

Αλγόριθμος 7 Αλγόριθμος συνδυασμού δύο στοιχείων του DatAbox

Input: · tuple of type ($\mathcal{A} \xrightarrow{ans}$ *ans, diff*): $\mathcal{B}_i$
· tuple of type ($\mathcal{A} \xrightarrow{ans}$ *ans, diff*): $\mathcal{B}_j$
Output: · tuple of type ($\mathcal{A} \xrightarrow{ans}$ *ans, diff*): $\mathcal{B}_{i+j}$

function SimpleJoin($\mathcal{B}_i$, $\mathcal{B}_j$)
  $\mathcal{A}_{i+j} \leftarrow \mathcal{B}_i$.ABox $\cup$ $\mathcal{B}_j$.ABox
  $ans_{i+j} \leftarrow \mathcal{B}_i$.Ans $\cup$ $\mathcal{B}_j$.Ans
  $diff_{i+j} \leftarrow$ max{$\mathcal{B}_i$.Diff, $\mathcal{B}_j$.Diff}
  $\mathcal{B}_{i+j} \leftarrow (\mathcal{A}_{i+j} \xrightarrow{ans} ans_{i+j}, diff_{i+j})$
  return $\mathcal{B}_{i+j}$
end function

Όπως γίνεται αντιληπτό, τα άτομα απάντησης του παραγόμενου ABox είναι η ένωση των ατόμων απάντησης των ABoxes που συνδυάζονται. Επίσης, ως δυσκολία του παραγόμενου ABox, διατηρείται η μέγιστη δυσκολία των επιμέρους, κάτι αναμενόμενο, διότι η δυσκολία αντικατοπτρίζει την ελάχιστη δυσκολία για να δοθούν τα συγκεκριμένα διάνυσματα ατόμων ως απάντηση.

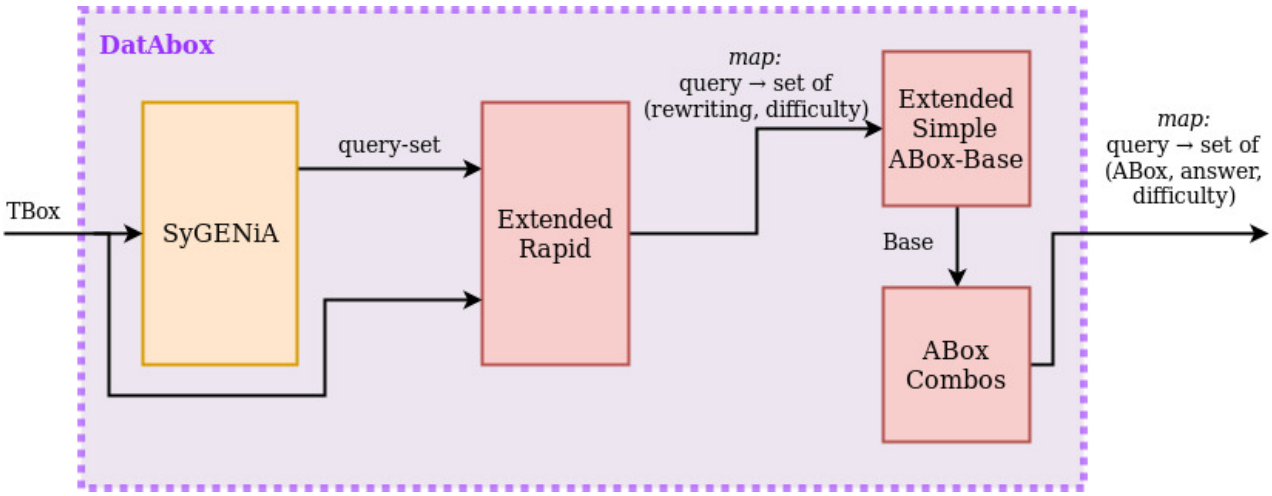
Μία άλλη σκέψη πάνω στον παραπάνω αλγόριθμο είναι να δίνεται ως είσοδος από το χρήστη το μέγιστο πλήθος των ατόμων k που θα έχουν τα ABoxes του συνόλου δεδομένων. Πάλι, στην περίπτωση που διατηρούμε ξεχωριστά άτομα μεταξύ των ατόμων, κάτι τέτοιο προκύπτει εύκολα, συνδυάζοντας κάθε φορά ABoxes που έχουν πλήθος ατόμων που αθροίζουν το πολύ στο k. Για την εύρεση των ABoxes που υπάρχουν στο DatAbox και που συν δυαζόμενα δίνουν ABoxes επιθυμητού πλήθους ατόμων, ουσιαστικά ζητάμε να λύσουμε το πρόβλημα εύρεσης υποσυνόλων που αθροίζουν στο k, το οποίο λύνεται ψευδοπολυωνυμικά με δυναμικό προγραμματισμό. Ουσιαστικά, βάζοντας τα ABoxes του DatAbox σε μία απεικόνιση (map), όπου κλειδί είναι το πλήθος των ατόμων του, ζητάμε τα υποσύνολα της απεικόνισης που αθροίζουν το πολύ στο k. Αφού τα βρούμε, πραγματοποιούμε ένα join μεταξύ τους, για κάθε υποσύνολο. Η join που χρησιμοποιείται εδώ, είναι η προφανής επέκταση της SimpleJoin, για παραπάνω στοιχεία εισόδου.

Αλγόριθμος 8 Σχηματισμός συνόλου δεδομένων με ABoxes μεγέθους το πολύ k

Input: · CQ as a first order logic clause: q
· DL-Lite$_R$-TBox as a set of first order logic clauses: $\mathcal{T}$
· Maximum ABox size: k
Output: · Set of (ABox, difficulty): $\mathcal{D}$

function CalcDatAboxK(q, k, $\mathcal{T}$)
  Initialize $\mathcal{B}_A \leftarrow$ calcExtSimpleABoxBase$(q, \mathcal{T})$
  map $\leftarrow$ **new** Map with key type: int (num of atoms in ABox), and value: set of extended simple ABox-Base values
  Insert the elements of $\mathcal{B}_A$ to *map*
  Using dynamic programming, initialize the set *Set* of all the subsets s of *map*-keys, which sum up to k
  for each $s \in$ *Set* **do**
    Join between the discrete values of $\{map[i]\}_{i \in s}$
    Add results in $\mathcal{D}$
  end for
  return D
end function

Όλοι οι παραπάνω αλγόριθμοι συνδυασμών και παραγωγής συνόλου δεδομένων είναι εξαντλητικοί ως προς τα στοιχεία της βάσης, με τον δεύτερο να είναι εξαντλητικός με ένα άνω όριο στο πλήθος ατόμων. Οι αλγόριθμοι αυτοί είναι ενδεικτικοί, καθώς μπορεί κανείς να κάνει επιλεκτικούς συνδυασμούς που θα παράγουν για παράδειγμα μόνο "πολύ ενδιαφέροντα" Aboxes. Τέτοια Aboxes θα μπορούσαν να είναι αυτά που παράγονται από μικρότερα τα οποία έχουν διαφορετικά κόστη δυσκολιών και ταυτόχρονα διαφέρουν ποσοστιαία πάνω από 70% ως προς τα μονοπάτια δυσκολιών τους. Αντιλαμβανόμαστε ότι αν είχαμε τη δυνατότητα να πειραματιστούμε πάνω σε τέτοιους σχηματισμούς δεδομένων θα μπορούσαμε να εξάγουμε και κάποιον "ενδιαφέρον" αλγόριθμο, του οποίου τα δεδομένα θα εκπαίδευαν καλύτερα ή γρηγορότερα από άλλα. Τέτοια κριτήρια και παραλλαγές είναι σχετικά εύκολο να πραγματοποιηθούν, εφόσον έχει σχηματισθεί μία σωστή βάση.



Σχήμα 3.4: Σχηματική αναπαράσταση του συστήματος DatAbox, με τα δομικά στοιχεία που παρουσιάστηκαν παραπάνω. Ως "Extended Rapid", αναφέρεται το σύστημα Rapid με τις ενσωματωμένες διεργασίες που αφορούν την εξαγωγή δυσκολίας επαναγραφής. Ως "ABox Combos", αναφέρεται η διαδικασία συνδυασμών των στοιχείων της βάσης.

Κεφάλαιο 4

Υλοποίηση-Πειράματα

Στον παρόν κεφάλαιο παρουσιάζεται μία τροποποίηση του συστήματός μας ώστε να αξιολογηθεί πάνω σε ένα ήδη υλοποιημένο συστημα παραγωγής ABoxes σε σχέση με ένα ερώτημα για την εκπαίδευση ενός νευρωνικού δικτύου. Στη συνέχεια παρουσιάζονται κάποια τεχνικά θέματα υλοποίησης του συστήματος.

4.1 Δημιουργία συνόλου για πειράματα

Με σκοπό μία μερική αξιολόγηση του συστήματός μας, μεταποιήσαμε λίγο κάποιους αλγορίθμους ώστε να συμβιβαστούν με ένα ήδη υλοποιμένο σύστημα. Το νέο, λίγο μεταποιημένο σύστημα το ονομάζουμε DifficAbox και η ενσωμάτωσή του στο ήδη υλοποιημένο σύστημα φαίνεται σχηματικά στο Σχήμα 4.1.

4.1.1 Υπολογισμός δυσκολίας ABox

Αυτό που συμβαίνει στην ουσία, είναι ότι το σύστημα αυτό εκπαιδεύει ένα νευρωνικό δίκτυο, σε περιγραφική λογική $\mathcal{EL}$, με ατομικά ερωτήματα μίας μεταβλητής απάντησης και με τυχαία ABoxes για κάθε ένα από αυτά. Η συνεισφορά της εργασίας μας και των αλγορίθμων που αναπτύξαμε ήταν να εξάγεται η μετρική δυσκολίας για κάθε ένα από αυτά τα ABoxes που παράγονταν σε σχέση με το ερώτημα, στην λιγότερο περιγραφική λογική DL-Lite$_\mathcal{R}$. Τα ABoxes αυτά, δεν έχουν τις ιδιότητες των ABoxes του DatAbox που επεξηγήσαμε πριν, επομένως περιέχουν πιθανόν περιττούς ή μη σχετικούς όρους. Επίσης, σε κάθε τέτοιο ABox είναι δυνατό να υπάρχουν παραπάνω από μία απάντηση στο ερώτημα, επομένως μας αφορά η εξαγωγή της ελάχιστης δυσκολίας του κάθε ατόμου απάντησης, ώστε να συμπεριληφθεί ως άτομο απάντησης. Συγκεκριμένα, οι διαδικασίες που χρησιμοποιήθηκαν ήταν η κατασκευή του γράφου ξετυλιγμάτων και η εξαγωγή δυσκολίας ξετυλίγματος, ολίγον τροποποιημένη, όπως παρουσιάζεται παρακάτω. Μας ενδιαφέρουν μόνο κανόνες ξετυλίγματος διότι μόνο τέτοιοι μπορούν να εφαρμοστούν κατά την επαναγραφή ατομικών ερωτημάτων.

Ο αλγόριθμος CalcUnfoldingDifficultyABox αρχικά ομαδοποιεί στην απεικόνιση map τους ισχυρισμούς στα άτομα που περιέχουν. Σε κάθε τέτοια αντιστοίχιση, το άτομο-κλειδί θεωρείται το άτομο απάντησης του οποίο θα αναζητήσουμε τη δυσκολία. Για το λόγο αυτό, στο σύνολο τιμών του ατόμου-κλειδί αντικαθίσταται, στη συνέχεια, το άτομο αυτό με τη μεταβλητή απάντησης του ερωτήματος. Έτσι, μπορούμε απλά να αναζητήσουμε τη δυσκολία επαναγραφής στο σύνολο τιμών, σε σχέση με το αρχικό ερώτημα, με τον αλγόριθμο CalcUnfoldingDifficulty. Αυτό ισχύει, καθώς όλοι οι ισχυρισμοί του ABox που αφορούν το άτομο βρίσκονται στο σύνολο τιμών του

Αλγόριθμος 9 Υπολογισμός δυσκολίας ατόμων απάντησης από ABox

Input: · Atomic query as first order logic clause: q
· ABox: $\mathcal{A}$
· Rewriting graph: $\mathcal{G}$
Output: · Atom difficulty map: *diff*

```
1:  function CalcUnfoldingDifficultyABox(q, A, G)
2:      map ← new Map with atoms as keys and assertions as value
3:      for each assertion in A do
4:          for each atom a in assertion do
5:              add assertion to map[a]
6:          end for
7:      end for
8:      for each atom-key a in map do
9:          q' ← new query with head ← q.head and body ← map[a]
10:         diff[a] ← CalcUnfoldingDifficulty(q, q', G)
11:     end for
12:     return diff
13: end function
```

κι επομένως από αυτό και μόνο προκύπτει η δυσκολία για αυτό. Η έξοδος του αλγορίθμου είναι μία απεικόνιση των ατόμων του ABox σε μία δυσκολία, η οποία αφορά την εξαγωγή του ατόμου-κλειδιού ως άτομο απάντησης. Αν ένα άτομο δεν ανήκει στην απάντηση, τότε το κόστος δυσκολίας είναι αρνητικό και το μονοπάτι δυσκολίας κενό. Ακολουθεί ένα απλό παράδειγμα που δείχνει τη λειτουργία του παραπάνω αλγορίθμου.

Ακολουθούν δύο παραδείγματα που δείχνουν τη λειτουργία των παραπάνω αλγορίθμων.

Παράδειγμα 4.1.1. Έστω ένα DL-Lite$_\mathcal{R}$-TBox $\mathcal{T}$, ένα ατομικό ερώτημα q κι ένα ABox $\mathcal{A}$ ως εξής:

$\mathcal{T} = \{C(x) \leftarrow B(x), B(x) \leftarrow A(x)\}$
$q : Q(x) \leftarrow C(x)$
$\mathcal{A} = \{A(a), B(a), B(b), C(b), A(c)\}$

Τα διαφορετικά σύνολα που δημιουργούνται για το κάθε άτομο έχουν ως εξής:

$a \longrightarrow \{A(a), B(a)\}$
$b \longrightarrow \{B(b), C(b)\}$
$c \longrightarrow \{A(c)\}$

τα οποία με αντικατάσταση ατόμου με τη μεταβλητή απάντησης οδηγούν στα ερωτήματα:

$q_1 : Q(x) \rightarrow A(x), B(x)$
$q_2 : Q(x) \rightarrow B(x), C(x)$
$q_3 : Q(x) \rightarrow A(x)$

Τελικά, εκτελείται ο αλγόριθμος CalcUnfoldingDifficulty για κάθε q_i σε σχέση με το q και προκύτει:

$diff[a] \longrightarrow (1, \{C, B\})$
$diff[b] \longrightarrow (0, \{C\})$
$diff[c] \longrightarrow (2, \{C, B, A\})$

Παράδειγμα 4.1.2. Έστω ένα DL-Lite$_\mathcal{R}$-TBox $\mathcal{T}$, ένα ατομικό ερώτημα q κι ένα ABox $\mathcal{A}$ ως εξής:

$\mathcal{T} = \{R(x, y) \leftarrow A(x), R(x, y) \leftarrow S(x, y)\}$ $q : Q(x) \leftarrow R(x, y)$ $\mathcal{A} = \{A(a), S(a, b)\}$

[1pt] Τα διαφορετικά σύνολα που δημιουργούνται για το κάθε άτομο έχουν ως εξής:

$a \longrightarrow \{A(a), S(a,b)\}$

$b \longrightarrow \{S(a,b)\}$

τα οποία με αντικατάσταση ατόμου με τη μεταβλητή απάντησης οδηγούν στα ερωτήματα:

$q_1 : Q(x) \rightarrow A(x), S(x,b)$

$q_2 : Q(x) \rightarrow S(a,x)$

Τελικά, εκτελείται ο αλγόριθμος CalcUnfoldingDifficulty για κάθε q_i σε σχέση με το q και προκύτει:

$diff[a] \longrightarrow (1, \{R, S\})$

$diff[b] \longrightarrow (-1, \{\})$

Η διαδικασία που παρουσιάστηκε εφαρμόζεται για κάθε ABox του συνόλου που συνδέονται με ένα ερώτημα στην εκμάθηση. Με αυτόν τον τρόπο, κατηγοριοποιούμε τα ABoxes σε σχέση με την ελάχιστη δυσκολία εξαγωγής συμπερασμών που απαιτείται, ώστε να εξαχθεί μία συγκεκριμένη απάντηση. Αυτό εν συνεχεία μπορεί να χρησιμοποιηθεί από το ήδη υπάρχον σύστημα για περαιτέρω συνδυσμούς ”δύσκολων” ή ”εύκολων” ABoxes και την μελέτη της συμπεριφοράς εκμάθησης του νευρωνικού δικτύου ανάλογα. Γενικά, είναι αναμενόμενο ότι όσο μεγαλύτερο είναι το εύρος δυσκολιών, όχι μόνο ως προς το κόστος αλλά και ως προς τα διαφορετικά μονοπάτια, τόσο ποιοτικότερη είναι και η εκπαίδεσυη του νερυωνικού δικτύου, καθώς καλύπτεται ένα μεγαλύτερο εύρος των πιθανών κόσμων που δημιουργεί.

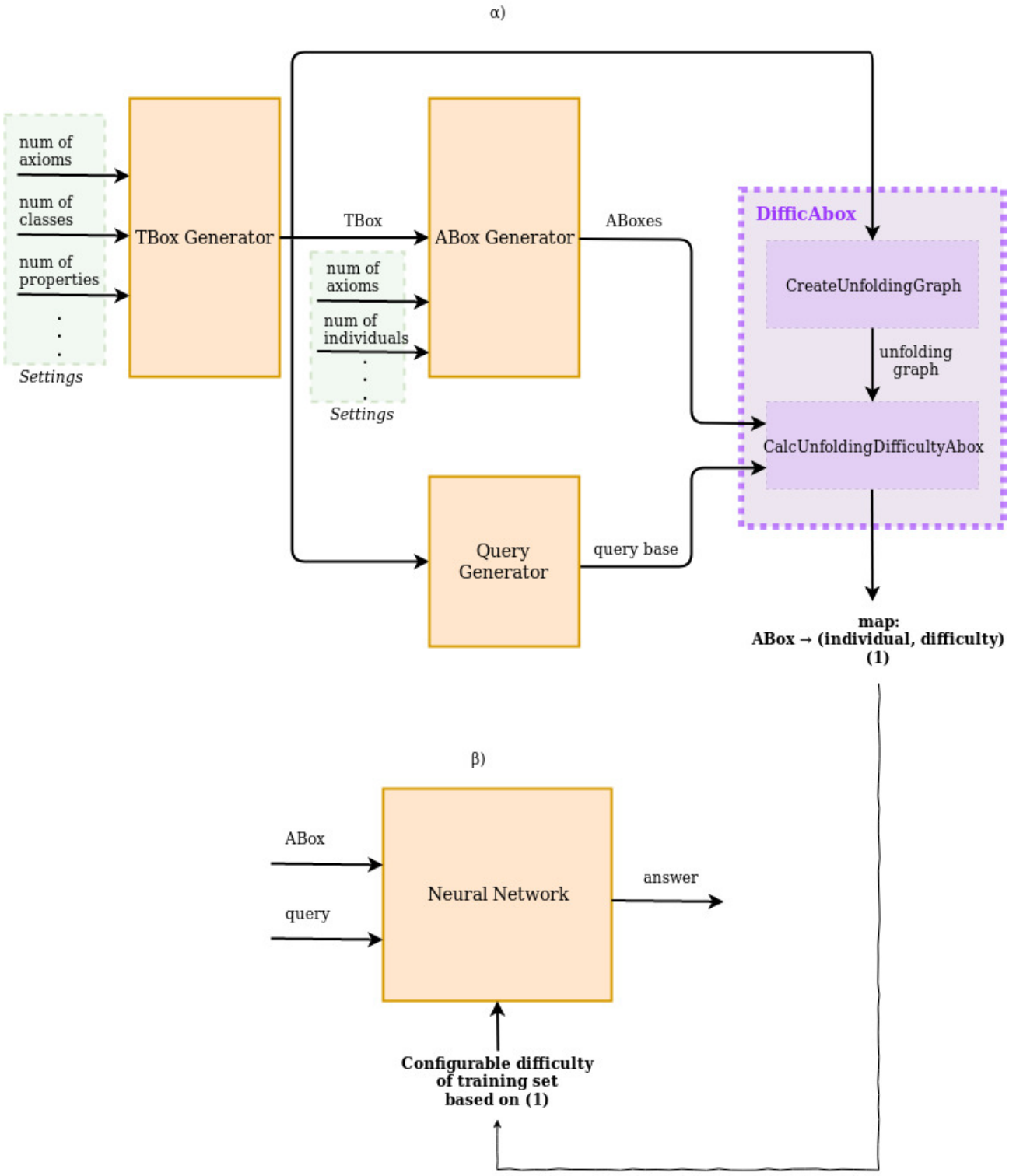
Εϊναι φυσιολογικό να αναρωτηθεί κανείς γιατί δεν σχεδιάζουμε διαδικασίες για εξαγωγή δυσκολίας ενός ABox σε ένα συζευτκικό ερώτημα γενικά. Η διαδικασία για ένα τέτοιο ερώτημα θα ήταν πολύ πιο δύσκολη διότι θα πρέπει ουσιαστικά να κάνουμε εξολοκλήρου τη διαδικασία εξαγωγής συμπερασμών, για να βρούμε όλα τα πιθανά υποσύνολα του ABox στο οποίο μπορεί να αναπτυχθεί το ερώτημα με βάσειτα αξιώματα. Αυτό μοιάζει με το να βρούμε όλες τις επαναγραφές και στη συνέχεια να ψάξουμε στο ABox για ταυτόσημους ισχυρισμούς με τους όρους, κάτω υπό αντιστοιχίσεις μεταβλητών με άτομα. Κάτι τέτοιο φαίνεται να έχει πολύ υψηλή πολυπλοκότητα, κυρίως λόγω της αναζήτησης απεικόνισης μεταβλητών σε άτομα και στη συνέχεια εύρεση της απεικόνισης που δίνει ελάχιστη δυσκολία για ένα άτομο.

4.2 Java-OWL

Οι αλγόριθμοι που παρουσιάστηκαν στην εργασία αυτή αναπτύχθηαν σε Java για δύο λόγους κυρίως:

- Το σύστημα Rapid, στο οποίο ενσωματώθηκαν οι αλγόριθμοί μας είναι αναπτυγμένο σε Java
- Γίνεται χρήση του πολύ χρήσιμου API της Java για βάσεις γνώσης, δηλαδή του OWLAPI.

Οι μικροί κόσμου, δηλαδή τα ABoxes, που δημιουργούνται μέσα από τους αλγορίθμους, αποθηκεύονται ως οντολογίες, δηλαδή ”.owl” αρχεία. Ως προς την είσοδο στο σύστημα, είναι όπως ακριβώς τη δέχεται ο Rapid, δηλαδή λαμβάνεται ένα TBox είτε μέσω ενός ”.owl” αρχείου το οποίο μετατρέπεται σε ένα σύνολο προτάσεων λογικής πρώτης τάξης, όπως παρουσιάστηκε στο 2, είτε ως απλό ”.txt” αρχείο με το σύνολο των προτάσεων λογικής πρώτης τάξης κατευθείαν. Το ερώ-



Σχήμα 4.1: Σχηματική αναπαράσταση υλοποιημένου συστήματος μηχανικής μάθησης, πάνω στο οποίο ενσωματώθηκε το DifficAbox για εξαγωγή δυσκολίας των ABoxes που παράγονται. Με αυτόν τον τρόπο, γίνεται διαμόρφωση του συνόλου εκπαίδευσης ανάλογα με τη δυσκολία του κάθε ABox.

τημα εισόδου δίνεται επίσης σύμφωνα με το Rapid, δηλαδή είναι ένα συζευκτικό ερώτημα σε μορφή πρότασης λογικής πρώτης τάξης που αριστερά έχει την κεφαλή και δεξιά το σώμα.

Τα ερωτήματα όπως είπαμε λαμβάνονται από το εργαλείο SyGENiA, το οποίο τα δίνει σε απλό ".txt" αρχείο στην επιθυμητή μορφή, επομένως δεν χρειάζεται κάποια ιδιαίτερη τροποποίηση.

Όσο για το σύνολο δεδομένων που παράγεται, είναι μία αντιστοιχίση (Java Map) με κλειδιά στιγμιότυπα κλάσης ABox, η οποία περιέχει και την απάντηση σε αυτό, και τιμή για κάθε κλειδί ένα στιγμιότυπο της κλάσης δυσκολίας. Τα ABoxes αυτά κάθε αυτά αποθηκεύονται ως οντολογίες OWLOntology του OWLAPI. Μία εναλλακτική θα ήταν να αποθηκεύονταν ως μία λίστα από RDF-Triples, αφού αποτελούνται μόνο από ισχυρισμούς και όχι κάποιο TBox. Κάτι τέτοιο θα ήταν "ελαφρύτερο" για το σύστημα, όμως επιλέχθηκε η δομή OWLOntology με σκοπό να μπορεί να χρησιμοποιηθεί το OWLAPI για αξιολόγηση των ABoxes και των απαντήσεων, για παράδειγμα να τρέξουμε κάποιο reasoner υπό το αντίστοιχο TBox και να συγκρίνουμε τις απαντήσεις cert και strans. Γενικά, είναι δυνατό πάντως, αν απλά θέλουμε να διατηρήσουμε τους ισχυρισμούς χωρίς κάποια ειδική λειτουργικότητα που παρέχεται άμεσα από το OWLAPI, να αποθηκεύουμε τα ABoxes ως σύνολο RDF-Triples.

Κεφάλαιο 5

Επίλογος

Στο παρόν κεφάλαιο συνοψίζεται το σύστημα που αναπτύχθηκε στην εργασία αυτή και παρουσιάζονται επιγραμματικά τα συμπεράσματα που προέκυψαν. Τέλος παρουσιάζονται προτάσεις για τις μελλοντικές κατευθύνσεις της επιστημονικής μελέτης.

5.1 Σύνοψη

Η παρούσα διπλωματική εργασία πραγματεύθηκε μία νέα προσέγγιση του προβλήματος απάντησης σε ερώτημα σε βάσεις γνώσης. Εκμεταλλευόμενοι εργαλεία που έχουν ήδη μελετηθεί και υλοποιηθεί για λίγο διαφορετικά προβλήματα, και συνδυάζοντάς τα, καταλήγουμε σε κάποιες πολύ ενδιαφέρουσες και πρωτοποριακές ιδέες.

Το σύστημά μας, με είσοδο μόνο ένα σύνολο λογικών αξιωμάτων (TBox), παράγει αρχικά τα ερωτήματα που είναι χρήσιμο να μελετηθούν σε σχέση με αυτό και που απαιτούν εξαγωγές συμπερασμών για να απαντηθούν. Στη συνέχεια, για κάθε τέτοιο ενδιαφέρον ερώτημα, παράγει ένα σύνολο "κόσμων", δηλαδή ABoxes, μαζί με τις απαντήσεις των κόσμων αυτών στο ερώτημα αλλά και μαζί με μία μετρική δυσκολίας εξαγωγής της απάντησης αυτής. Για την εξαγωγή των ABoxes, χρησιμοποιείται το σύνολο επαναγαφών που παράγει το σύστημα Rapid, το οποίο είναι ελάχιστο και εξαντλητικό για DL-Lite. Παράλληλα με τον υπολογισμό των επαναγραφών, υπολογίζεται η δυσκολία τους. Οι επαναγραφές αυτές μετατρέπονται απευθείας στα ABoxes που αναφέρουμε κι έτσι η δυσκολία κάθε ABox είναι η δυσκολία της επαναγραφής από την οποία προέρχεται. Τέλος, οι απαντήσεις του κάθε ABox παράγεται με το SQL ερώτημα που προκύπτει από την επαναγραφή.

Η μετρική δυσκολίας δημιουργεί ένα μέτρο σύγκρισης μεταξύ των ABoxes, και συνεπώς μία διάταξη, που γενικά λείπει από τη βιβλιογραφία. Η διάταξη αυτή μπορεί να χρησιμοποιηθεί για την ελεγχόμενη εκπαίδευση συστημάτων μηχανικής μάθησης, από την άποψη δημιουργίας συνόλου δεδομένων εκμάθησης με συγκεκριμένα χαρακτηριστικά και όχι τυχαία. Δηλαδή, μπορεί να επιλεγεί σύνολο δεδομένων εκπαίδευσης με συγκεκριμένη ελάχιστη δυσκολία, κάτι που εξασφαλίζει εκμάθηση συμπερασμών συγκεκριμένου βάθους (με βάση το κόστος δυσκολίας) ή με συγκεκριμένα αξιώματα (με βάση το μονοπάτι δυσκολίας). Η παραπάνω λογική δημιουργίας δεδομένων εκμάθησης παραπέμπει σε επιλογή στοιχείων του κόσμου που δίνουν απάντηση στο ερώτημα, το οποίο ανήκει στο πεδίο του PAC Learning και γενικότερα της εκμάθησης από θετικά παραδείγματα.

5.2 Συμπεράσματα - Συνεισφορά

Ο βασικός σκοπός της εργασίας αυτής ήταν η εξαγωγή μίας ικανοποιητικής βάσης για τη δημιουργία ποιοτικού συνόλου δεδομένων εκπαίδευσης, μίας μετρικής με βάση την οποία διατάσονται τα στοιχεία και τελικά ο σχηματισμός μεγαλύτερων στοιχείων με βάση συνδυασμούς των στοιχείων της βάσης ώστε να προκύπτουν στοιχεία με συγκεκριμένα χαρακτηριστικά. Επιγραμματικά, τα συμπεράσματα και η συνεισφορά αυτής της εργασίας είναι:

- Μέχρι τώρα, στο ήδη υλοποιημένο σύστημα μηχανικής μάθησης με σκοπό τη λύση του προβλήματος απάντησης σε ερώτημα, και το οποίο αναφέρθηκε στο κεφάλαιο υλοποίησης και αξιολόγησης 4, τα δεδομένα εκπαίδευσης, δηλαδή ABoxes με απάντηση, προέκυπταν ως εξής: Δίνονταν κάποια χαρακτηριστικά στο σύστημα παραγωγής οντολογιών OTAGen, όπως πλήθος ατόμων, έννοιες και ρόλοι, και αυτό παρήγαγε τυχαία κόσμους. Με την ενσωμάτωση του συστήματός μας, ξεφεύγουμε από την τυχαία παραγωγή ABoxes για την εκπαίδευση συστήματος μηχανικής μάθησης, και παράγουμε κόσμους με νόημα κι ενδιαφέρον, αλλά κι ελεγχόμενα, με την έννοια της δυνατότητας παραγωγής ABoxes συγκεκριμένης δυσκολίας.

- Επίσης, στο υλοποιημένο σύστημα μηχανικής μάθησης, μέχρι τώρα, οι απαντήσεις στο εκάστοτε ερώτημα υπολογίζονταν εκ των υστέρων σε σχέση με την παραγωγή τους, μέσω κάποιου reasoner. Στο σύστημά μας, οι απαντήσεις στο ερώτημα πάνω στα ABoxes που υπολογίζουμε, προκύπτουν χωρίς reasoning, κάτι πολύ σημαντικό για την απόδοση. Επίσης, κατά το σχηματισμό μεγαλύτερων ABoxes, λόγω του ότι δεν δημιουργούμε κοινά άτομα μεταξύ των συνδυαζόμενων ABoxes, οι απαντήσεις προκύπτουν εντελώς άμεσα, αφού είναι απλά η ένωση των απαντήσεων των συνδυαζόμενων ABoxes. Σημειώνεται, ότι αν προχωρούσαμε σε πιο πολύπλοκους συνδυασμούς όπου θα υπήρχαν κοινά άτομα μεταξύ των συνδυαζόμενων στοιχείων, οι απαντήσεις δεν θα προέκυπταν τόσο άμεσα και χρειάζεται περαιτέρω μελέτη.

- Ίσως η πιο αξιοσημείωτη μελέτη αυτής της εργασίας, είναι η δημιουργία και η εξαγωγή της δυσκολίας μίας επαναγαφής κι ενός ABox. Η μετρική δυσκολίας δημιουργεί μία διάταξη μεταξύ των ερωτημάτων ή των ABoxes, σε σχέση με την πολυπλοκότητα των συμπερασμών που πρέπει να εξαχθούν για να βρεθεί μία επαναγραφή ή να απαντηθεί το ερώτημα στο ABox, αντίστοιχα. Η μετρική αυτή μπορεί να επεκταθεί σχετικά εύκολα αλλά και να χρησιμοποιηθεί για άλλα προβλήματα εξαγωγής συμεπρασμών.

- Τέλος, σημαντικό σημείο της παρούσας εργασίας αποτελεί η δημιουργία του γράφου ξετυλιγμάτων, ο οποίος επιτρέπει γραμμική αναζήτηση συσχέτισης δύο εννοιών/ρόλων στο μέγεθος του TBox. Αυτό σημαίνει ότι σε γραμμικό χρόνο μπορούμε να δούμε αν μία έννοια ή ρόλος μπορεί να αναχθεί σε κάποια άλλη έννοια ή ρόλο, και συνεπώς, επιτρέπει να υπολογίσουμε τη δυσκολία μίας επαναγραφής σε σχέση με μία άλλη σε πολυωνυμικό χρόνο, και συγκεκριμένα $O(n_1 \cdot n_2 \cdot k)$, όπου n_1, n_2 τα μεγέθη των δύο ερωτημάτων και k το μέγεθος του γράφου. Ο γράφος αυτός μπορεί επίσης να χρησιμοποιηθεί και σε άλλα προβλήματα συμπερασμών.

5.3 Μελλοντικές κατευθύνσεις

Η παρούσα διπλωματική εργασία ανοίγει ένα ενδιαφέρον πεδίο μελέτης, αφήνοντας πολλά ανοιχτά ζητήματα που μπορούν να μελετηθούν μελλοντικά. Τα σημεία αυτά έχουν αναφερθεί κατά τη διάρκεια της εργασίας αλλά παρουσιάζονται επιγραμματικά κι εδώ:

- Η περιγραφική λογική DL-Lite είναι κατεξοχήν η "γλώσσα" που χρησιμοποιείται για επίλυση προβλημάτων απάντησης σε ερώτημα σε βάσεις γνώσης και βασικός λόγος γι' αυτό είναι η αποδοτικότητά της λόγων περιορισμένης εκφραστικότητας. Η ιδέα για εμπλακή της μηχανικής μάθησης, αναιρεί τον παράγοντα αποδοτικότητα κατά τη λειτουργία του συστήματος (όχι για την εκπαίδευση), αφού, άπαξ κι εκπαιδευτεί το σύστημα μηχανικής μάθησης, απαντά άμεσα σε ερωτήματα. Επομένως είναι λογικό να στραφούμε σε πιο εκφραστικές περιγραφικές λογικές, όπως είναι η ευρέως μελετούμενη $\mathcal{EL}$. Ο σχηματισμός συνόλου δεδομένων εκπαίδευσης θα είναι συνθετότερος και πιθανότατα αδύνατο να είναι εξαντλητικός, παρά μόνο υπό ορισμένες συνθήκες. Όμως, ακόμα και χωρίς πλήρη κάλυψη του πιθανού κόσμου απαντήσεων, έχει ενδιαφέρον η μελέτη και η αξιολόγηση αποτελεσμάτων αυτής της κατεύθυνσης.

- Όπως αναφέρθηκε, τα ABoxes της βάσης που χρησιμοποιείται, δηλαδή της εκτεταμένης απλής βάσης των ABoxes, δεν είναι ελάχιστα σε σχέση με το ερώτημα, από την άποψη ότι η εξάλλειψη ενός οποιοδήποτε ισχυρισμού δεν οδηγεί *πάντα* σε κενή απάντηση. Παρ' όλ' αυτά, έχουν βρεθεί κάποια μοτίβα στις επαναγραφές από τα οποία προκύπτουν μη ελάχιστα ABoxes. Τέτοια μοτίβα είναι αρχικά η ύπαρξη περισσοτέρων από δύο μεταβλητές απάντησης, ή η ύπαρξη κάποιας συμμετρίας μεταξύ κάποιων μεταβλητών ως προς τους όρους στους οποίους συμμετέχουν, για παράδειγμα ύπαρξη όρων $R(x, y), R(x, z)$ στο ερώτημα $q : Q(x, y, z) \leftarrow A(x), R(x, y), R(x, z)$. Όμως, αυτά είναι απλά παρατηρήσεις, επομένως θα είχε ενδιαφέρον να μελετηθεί και αποδειχτεί η ισχύς αυτών αλλά και άλλων χαρακτηριστικών που οδηγούν σε μη ελάχιστα ABoxes. Η κατεύθυνση αυτή είναι αρκετά σημαντική, καθώς οδηγεί άμεσα σε δημιουργία *αρνητικών* παραδειγμάτων του κόσμου για την εκμάθηση. Δηλαδή, έχοντας τα ελάχιστα ABoxes, σε σταθερό χρόνο με αφαίρεση απλά ενός οποιουδήποτε ισχυρισμού, θα λαμβάνεται αρνητικό ενδιαφέρον ABox, και θα μαθαίνει το σύστημα μηχανικλης μάθησης ότι η έλλειψη αυτού του ισχυρισμού είναι απαραίτητη για να δοθεί μία απάντηση.

- Ο αλγόριθμος μετατροπής ενός ερωτήματος επαναγραφής σε ABox είναι κάπως περιορισμένος, καθώς μετατρέπει απλά κάθε διακεκριμένη μεταβλητή σε ένα διακεκριμένο άτομο του κόσμου, διατηρώντας τους όρους του ερωτήματος ως ισχυρισμούς του ABox. Η διαδικασία αυτή είναι δυνατό να επεκταθεί, σε σχέση μάλιστα και με την προηγούμενη παράγραφο. Συγκεκριμένα, θα μπορούσαμε υπό συγκεκριμένες συνθήκες να αντιστοιχούμε διακεκριμένες μεταβλητές σε ίδια άτομα του κόσμου με σκοπό, για παράδειγμα, να παράγονται ελάχιστα ABoxes. Όμως, αν θέλουμε να διατηρήσουμε τη βάση χωρίς διπλότυπα και περιττούς ισχυρισμούς, απαιτείται μελέτη για το ποια άτομα και πότε μπορούν να ταυτιστούν. Επίσης, πρέπει να προσεχθεί να μην δημιουργούνται ασυνέπειες στο ABox. Στην παρούσα εργασία επιλέχθηκε να μην ταυτίζονται άτομα του κόσμου, με σκοπό να διατηρήσουμε μία βάση από ABoxes χωρίς περιττούς ισχυρισμούς και χωρίς ασυνέπειες.

- Η παραπάνω επέκταση, μπορεί να εφαρμοστεί και κατά τον συνδυασμό των ABoxes (join), με σκοπό το σχηματισμό πιο σύνθετων ABoxes, με κοινά άτομα. Ομοίως με πριν, πρέπει να μελετηθούν οι συνθήκες για να είναι δυνατή η ταυτοποίηση δύο ατόμων του κόσμου χωρίς να προκείπτουν ασυνέπειες. Ακόμα, ως προς την εξέλιξη του αλγορίθμου δημιουργίας συνδυασμών, είναι εύλογο να κατευθυνθεί προς δημιουργία πιο ελεγχόμενων συνόλων, από τους εξαντλητικούς συνδυασμούς που γίνονται τώρα. Για παράδειγμα, θα μπορούσαμε να ζητάμε να παράγονται ABoxes από μικρότερα ABoxes τα οποία θα έχουν κάποιο ελάχιστο κόστος δυσκολίας και θα διαφέρουν μεταξύ τους ποσοστιαία ως προς τα μονοπάτια δυσκολίας. Κάτι τέτοιο θα παρήγαγε στοιχεία του κόσμου που περιέχουν πολλά διαφορετικά στιγμιότυπα και απαιτούν πολλούς και διαφορετικούς συμπερασμούς για να δώσουν απάντηση.

- Μία ακόμα ενδιαφέρουσα επέκταση της παρούσας εργασίας είναι η επέκταση της οντότητας δυσκολίας, Προς το παρόν, στο μονοπάτι δυσκολίας μίας επαναγραφής (και κατ' επέκταση και του αντίστοιχου ABox), αναγράφονται τα ονόματα εννοιών και ρόλων που εμπλέκονται στα αξιώματα που εφαμρόζονται για την παραγωγή μιας επαναγραφής, και μέσα από αυτό, εξάγεται το κόστος δυσκολίας το οποίο προκύπτει από το πλήθος αξιωμάτων ξετυλίγματος, αυτών που είναι υπαρξιακά και αξιωμάτων συρρίκνωσης, καθένα πολλαπλασιασμένο με κάποιο βάρος. Επιλέξαμε τα παραπάνω για την εξαγωγή δυσκολίας γιατί θεωρήσαμε ότι είναι αρκετά αντιπροσωπευτικά για τον υπολογισμό μίας τέτοιας μετρικής. Όμως, η δυσκολία θα μπορούσε να επεκταθεί και να συνυπολογίζει, για παράδειγμα, το πλήθος των ατόμων του εκάαστοτε ABox, ή οτιδήποτε άλλο θεωρείται χρήσιμο για ληφθεί υπόψιν με σκοπό τη διάταξη των ABoxes.

- Τέλος, μία ακόμα κατεύθυνση για τον εμπλουτισμό του συνόλου εκπαίδευσης, είναι η επέκταση των ερωτημάτων που δίνει το SyGENiA και τα οποία λαμβάνουμε υπόψιν σε αυτή την εργασία. Δηλαδή, πέρα από την εκμάθηση απαντήσεων σε αυτά τα ερωτήματα, θα μπορούσε να το σύστημα να εκπαιδεύεται και πάνω σε άλλα ερωτήματα που ίσως περιέχουν άσχετους όρους, με σκοπό να το μπερδέψουν, ή σε ερωτήματα που είναι συνδυασμοί αυτών που προκείπτουν από το SyGENiA.

Βιβλιογραφία

Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the el envelope. In *IJCAI*, volume 5, pages 364–369, 2005.

Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the el envelope further. 2008.

David GT Barrett, Felix Hill, Adam Santoro, Ari S Morcos, and Timothy Lillicrap. Measuring abstract reasoning in neural networks. *arXiv preprint arXiv:1807.04225*, 2018.

Meghyn Bienvenu, Balder Ten Cate, Carsten Lutz, and Frank Wolter. Ontology-based data access: A study through disjunctive datalog, csp, and mmsnp. *ACM Transactions on Database Systems (TODS)*, 39(4):33, 2014.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Dl-lite: Tractable description logics for ontologies. In *AAAI*, volume 5, pages 602–607, 2005.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *Journal of Automated reasoning*, 39(3):385–429, 2007.

Alexandros Chortaras, Despoina Trivela, and Giorgos B. Stamou. Optimized query rewriting for OWL 2 QL. In *Automated Deduction - CADE-23 - 23rd International Conference on Automated Deduction, Wroclaw, Poland, July 31 - August 5, 2011. Proceedings*, pages 192–206, 2011. doi: 10.1007/978-3-642-22438-6_16. URL https://doi.org/10.1007/978-3-642-22438-6_16.

François Denis. Pac learning from positive statistical queries. In *International Conference on Algorithmic Learning Theory*, pages 112–126. Springer, 1998.

B Cuenca Grau, Boris Motik, Giorgos Stoilos, and Ian Horrocks. Completeness guarantees for incomplete ontology reasoners: Theory and practice. *Journal of Artificial Intelligence Research*, 43:419–476, 2012.

Bernardo Cuenca Grau and Giorgos Stoilos. What to ask to an incomplete semantic web reasoner? In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

Peter Hansen, Carsten Lutz, Inanç Seylan, and Frank Wolter. Efficient query rewriting in the description logic el and beyond. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all justifications of owl dl entailments. In *The Semantic Web*, pages 267–280. Springer, 2007.

Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic el using a relational database system. In *Twenty-first international joint conference on artificial intelligence*, 2009.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015.

Femke Ongenae, Stijn Verstichel, Filip De Turck, Tom Dhaene, Bart Dhoedt, and Piet Demeester. Otagen: A tunable ontology generator for benchmarking ontology-based agent collaboration. In *2008 32nd Annual IEEE International Computer Software and Applications Conference*, pages 529–530. IEEE, 2008.

Heiko Paulheim and Heiner Stuckenschmidt. Fast approximate a-box consistency checking using machine learning. In *European Semantic Web Conference*, pages 135–150. Springer, 2016.

Héctor Pérez-Urbina, Ian Horrocks, and Boris Motik. Efficient query answering for owl 2. In *International Semantic Web Conference*, pages 489–504. Springer, 2009.

Riccardo Rosati. On conjunctive query answering in el. 01 2007.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

Giorgos Stamou and Alexandros Chortaras. Ontological query answering over semantic data. In *Reasoning Web International Summer School*, pages 29–63. Springer, 2017.

Giorgos Stamou, Despoina Trivela, and Alexandros Chortaras. Progressive semantic query answering. In *The 6th International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS2010)*, page 112, 2010.

Giorgos Stoilos, Bernardo Cuenca Grau, and Ian Horrocks. Completeness guarantees for incomplete reasoners. In *International Semantic Web Conference*, pages 747–763. Springer, 2010.

Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos Stamou. Optimising resolution-based rewriting algorithms for owl ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1014, 02 2015. doi: 10.1016/j.websem.2015.02.001.

Eleni Tsalapati, Giorgos Stoilos, Giorgos B Stamou, and George Koletsos. Efficient query answering over expressive inconsistent description logics. In *IJCAI*, pages 1279–1285, 2016.

Stijn Verstichel, Femke Ongenae, Bruno Volckaert, Filip De Turck, Bart Dhoedt, Tom Dhaene, and Piet Demeester. An autonomous service-platform to support distributed ontology-based context-aware agents. *Expert Systems*, 28(5):437–460, 2011.

Raphael Volz, Steffen Staab, and Boris Motik. Incrementally maintaining materializations of ontologies stored in logic databases. In *Journal on Data Semantics II*, pages 1–34. Springer, 2005.

Wikipedia contributors. Probably approximately correct learning — Wikipedia, the free encyclopedia, 2019. URL https://en.wikipedia.org/w/index.php?title=Probably_approximately_correct_learning&oldid=926733521. [Online; accessed 12-December-2019].

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. 2015.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. Neural generative question answering. *arXiv preprint arXiv:1512.01337*, 2015.

Βιβλιογραφία

Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the el envelope. In *IJCAI*, volume 5, pages 364–369, 2005.

Franz Baader, Sebastian Brandt, and Carsten Lutz. Pushing the el envelope further. 2008.

David GT Barrett, Felix Hill, Adam Santoro, Ari S Morcos, and Timothy Lillicrap. Measuring abstract reasoning in neural networks. *arXiv preprint arXiv:1807.04225*, 2018.

Meghyn Bienvenu, Balder Ten Cate, Carsten Lutz, and Frank Wolter. Ontology-based data access: A study through disjunctive datalog, csp, and mmsnp. *ACM Transactions on Database Systems (TODS)*, 39(4):33, 2014.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Dl-lite: Tractable description logics for ontologies. In *AAAI*, volume 5, pages 602–607, 2005.

Diego Calvanese, Giuseppe De Giacomo, Domenico Lembo, Maurizio Lenzerini, and Riccardo Rosati. Tractable reasoning and efficient query answering in description logics: The dl-lite family. *Journal of Automated reasoning*, 39(3):385–429, 2007.

Alexandros Chortaras, Despoina Trivela, and Giorgos B. Stamou. Optimized query rewriting for OWL 2 QL. In *Automated Deduction - CADE-23 - 23rd International Conference on Automated Deduction, Wroclaw, Poland, July 31 - August 5, 2011. Proceedings*, pages 192–206, 2011. doi: 10.1007/978-3-642-22438-6_16. URL https://doi.org/10.1007/978-3-642-22438-6_16.

François Denis. Pac learning from positive statistical queries. In *International Conference on Algorithmic Learning Theory*, pages 112–126. Springer, 1998.

B Cuenca Grau, Boris Motik, Giorgos Stoilos, and Ian Horrocks. Completeness guarantees for incomplete ontology reasoners: Theory and practice. *Journal of Artificial Intelligence Research*, 43:419–476, 2012.

Bernardo Cuenca Grau and Giorgos Stoilos. What to ask to an incomplete semantic web reasoner? In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

Peter Hansen, Carsten Lutz, Inanç Seylan, and Frank Wolter. Efficient query rewriting in the description logic el and beyond. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

Aditya Kalyanpur, Bijan Parsia, Matthew Horridge, and Evren Sirin. Finding all justifications of owl dl entailments. In *The Semantic Web*, pages 267–280. Springer, 2007.

Carsten Lutz, David Toman, and Frank Wolter. Conjunctive query answering in the description logic el using a relational database system. In *Twenty-first international joint conference on artificial intelligence*, 2009.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2015.

Femke Ongenae, Stijn Verstichel, Filip De Turck, Tom Dhaene, Bart Dhoedt, and Piet Demeester. Otagen: A tunable ontology generator for benchmarking ontology-based agent collaboration. In *2008 32nd Annual IEEE International Computer Software and Applications Conference*, pages 529–530. IEEE, 2008.

Heiko Paulheim and Heiner Stuckenschmidt. Fast approximate a-box consistency checking using machine learning. In *European Semantic Web Conference*, pages 135–150. Springer, 2016.

Héctor Pérez-Urbina, Ian Horrocks, and Boris Motik. Efficient query answering for owl 2. In *International Semantic Web Conference*, pages 489–504. Springer, 2009.

Riccardo Rosati. On conjunctive query answering in el. 01 2007.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

Giorgos Stamou and Alexandros Chortaras. Ontological query answering over semantic data. In *Reasoning Web International Summer School*, pages 29–63. Springer, 2017.

Giorgos Stamou, Despoina Trivela, and Alexandros Chortaras. Progressive semantic query answering. In *The 6th International Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS2010)*, page 112, 2010.

Giorgos Stoilos, Bernardo Cuenca Grau, and Ian Horrocks. Completeness guarantees for incomplete reasoners. In *International Semantic Web Conference*, pages 747–763. Springer, 2010.

Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos Stamou. Optimising resolution-based rewriting algorithms for owl ontologies. *Web Semantics: Science, Services and Agents on the World Wide Web*, 1014, 02 2015. doi: 10.1016/j.websem.2015.02.001.

Eleni Tsalapati, Giorgos Stoilos, Giorgos B Stamou, and George Koletsos. Efficient query answering over expressive inconsistent description logics. In *IJCAI*, pages 1279–1285, 2016.

Stijn Verstichel, Femke Ongenae, Bruno Volckaert, Filip De Turck, Bart Dhoedt, Tom Dhaene, and Piet Demeester. An autonomous service-platform to support distributed ontology-based context-aware agents. *Expert Systems*, 28(5):437–460, 2011.

Raphael Volz, Steffen Staab, and Boris Motik. Incrementally maintaining materializations of ontologies stored in logic databases. In *Journal on Data Semantics II*, pages 1–34. Springer, 2005.

Wikipedia contributors. Probably approximately correct learning — Wikipedia, the free encyclopedia, 2019. URL https://en.wikipedia.org/w/index.php?title=Probably_approximately_correct_learning&oldid=926733521. [Online; accessed 12-December-2019].

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. 2015.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. Neural generative question answering. *arXiv preprint arXiv:1512.01337*, 2015.