ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΑΓΡΟΝΟΜΩΝ ΚΑΙ ΤΟΠΟΓΡΑΦΩΝ ΜΗΧΑΝΙΚΩΝ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ

«ΓΕΩΠΛΗΡΟΦΟΡΙΚΗ»

ΕΡΩΤΗΜΑΤΑ ΑΚΡΙΒΟΥΣ ΤΡΟΧΙΑΣ ΣΕ ΒΑΣΕΙΣ ΔΕΔΟΜΕΝΩΝ ΚΙΝΟΥΜΕΝΩΝ ΑΝΤΙΚΕΙΜΕΝΩΝ

ΣΤΑΜΑΤΗΣ Ι. ΣΤΕΦΑΝΟΠΟΥΛΟΣ

ΕΠΙΒΛΕΠΩΝ: ΜΑΡΙΝΟΣ ΚΑΒΟΥΡΑΣ (ΚΑΘΗΓΗΤΗΣ)

ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αθήνα
Φεβρουάριος 2015

ΤΡΙΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

(υπογραφή)	*(υπογραφή)*	*(υπογραφή)*
Κάβουρας Μαρίνος	**Θεοδωρίδης Ιωάννης**	**Πελέκης Νικόλαος**
Καθηγητής	Καθηγητής	Επίκουρος Καθηγητής

Ευχαριστίες

Εν πρώτοις θα ήθελα να ευχαριστήσω θερμά τον κ. Νίκο Πελέκη, Επίκουρο Καθηγητή του Τμήματος Στατιστικής και Ασφαλιστικής Επιστήμης του Πανεπιστημίου Πειραιώς για την καθοδήγηση και υποστήριξη της παρούσας εργασίας σε οργανωτικό και τεχνικό επίπεδο και την κατανόηση που έδειξε στο βεβαρυμένο μου πρόγραμμα. Ακόμη, θα ήθελα να ευχαριστήσω τους κ. Κώστα Πατρούμπα και Μάριο Βόντα, για τις χρήσιμες τεχνικές συμβουλές που παρείχαν σε καίρια τεχνικά σημεία της υλοποίησης της εργασίας. Επίσης, όλο το διδακτικό προσωπικό του ΔΠΜΣ «Γεωπληροφορική» για τις γνώσεις που μου παρείχε κατά τη διάρκεια της φοίτησής μου σε αυτό, καθώς και το γραμματειακό προσωπικό, που με άριστη οργάνωση και υπευθυνότητα συνετέλεσε στην ομαλή ολοκλήρωσή του.

Ακόμη, θα ήθελα να ευχαριστήσω από καρδιάς όλους όσους με υποστήριξαν ανεχόμενοι και σεβόμενοι την απουσία μου ή την ενοχλητική παρουσία μου καθ' όλη τη διάρκεια της υλοποίησης της εργασίας, όσους με ενθάρρυναν με κάθε τρόπο όλο αυτό το διάστημα και ιδιαίτερα την οικογένειά μου, η οποία στήριζε και στηρίζει κάθε προσπάθειά μου.

Περίληψη

Η μελέτη των χωροχρονικών δεδομένων κίνησης οχημάτων εντός του αστικού οδικού δικτύου μπορεί να εξάγει χρήσιμα συμπεράσματα για ποικιλία εφαρμογών, όπως βελτιστοποίηση σηματοδότησης, παρακολούθηση συμφορήσεων και έλεγχο κυκλοφορίας. Για την εξαγωγή της πληροφορίας είναι απαραίτητη η ύπαρξη κατάλληλων, αποδοτικών ερωτημάτων προς τη Βάση Δεδομένων. Το ερώτημα ακριβούς τροχιάς (Strict Path Query – SPQ) χρησιμοποιείται για να ανακτήσει τις τροχιές οι οποίες ακολουθούν μία συγκεκριμένη διαδρομή χωρίς να παρεκκλίνουν καθόλου από αυτή. Για την υποστήριξη ερωτημάτων του συγκεκριμένου τύπου υλοποιείται η δομή NETTRA (προτείνεται στο [16]), ένα ευρετήριο ειδικά σχεδιασμένο για τέτοιου είδους ελέγχους το οποίο χρησιμοποιεί μια μέθοδο κωδικοποίησης για την αναγνώριση των ακμών του δικτύου από τις οποίες διέρχονται οι τροχιές. Πέραν του NETTRA, παρουσιάζονται και τυποποιούνται τα ερωτήματα ακριβούς τροχιάς, ενώ αναλύονται τεχνικές για τη βελτίωση της ταχύτητας εκτέλεσής τους και της ποιότητας των αποτελεσμάτων τους. Τέλος, αναπτύσσονται τα ερωτήματα που χρησιμοποιούνται για την ανάκτηση των τροχιών, ενώ υλοποιούνται και συναρτήσεις που αναλαμβάνουν τη δημιουργία και την εκτέλεση των ερωτημάτων για τη διευκόλυνση του εκάστοτε χρήστη.

Abstract

Studying network-constrained spatiotemporal traffic data is a useful activity in the terms of a variety of applications, including traffic signalling oprimization, traffic jam monitoring and traffic management. For the information extraction a set of suitable, efficient database queries is necessary. The Strict Path Query (SPQ) is used to extract from the database the trajectories that follow a specific route, without any detours. Supporting this type of database querying can be achieved by developing the NETTRA structure (proposed in [16]), an index specially designed for this kind of queries that uses an encoding technique for recognizing the network edges visited by the dataset's positions. Besides of NETTRA, the SPQ queries are presented and formalized, while techniques to enhance performance and results validation are mentioned and analyzed, some of them even developed. In the end, the SPQ queries for exact path extraction are developed, together with the wrapper functions that are used to construct the SPQ queries and hide their details from the end user for the sake of querying simplicity.

Περιεχόμενα

Κατάλογος Σχημάτων

Εικόνες

Διαγράμματα

Πίνακες

Ερωτήματα

Αλγόριθμοι

1. Εισαγωγή

Η ύπαρξη χωροχρονικών δεδομένων στον ψηφιακό κόσμο είναι τα τελευταία χρόνια εντονότερη από ποτέ. Η δημιουργία και διάθεση στην αγορά πληθώρας συσκευών και συστημάτων γεωγραφικού εντοπισμού έναντι ιδιαίτερα προσιτού πλέον αντιτίμου έχει συντελέσει καταλυτικά στην παραγωγή και αποθήκευση μεγάλου όγκου γεωγραφικών δεδομένων ανά τον κόσμο. Η τεράστια αυτή διαθεσιμότητα δεδομένων έχει δημιουργήσει τις κατάλληλες συνθήκες για την ανάπτυξη εφαρμογών και εργαλείων λογισμικού που, με χρήση εξειδικευμένων μεθόδων και τεχνικών επεξεργάζονται αυτά τα δεδομένα, εξάγοντας ιδιαίτερα χρήσιμα και ενδιαφέροντα συμπεράσματα από αυτά.

Στην πλειονότητα των περιπτώσεων, τα χωροχρονικά δεδομένα αναπαριστούν την θέση ενός αντικειμένου για μια δεδομένη χρονική στιγμή, αποτελώντας ουσιαστικά ένα σημείο σε ένα τρισδιάστατο (πολλές φορές τετραδιάστατο) σύστημα αξόνων. Η ιδιότητά τους αυτή τα καθιστά ιδανικά για την αναπαράσταση της κίνησης διαφόρων αντικειμένων, όπως οχημάτων, πεζών, ακόμη και λιθοσφαιρικών πλακών. Η ανάλυση των δεδομένων κίνησης μπορεί να εξάγει ενδιαφέροντα αποτελέσματα, ιδιαίτερα αν η κίνηση αυτή είναι μαζική και επηρεάζει τομείς της ανθρώπινης ή φυσικής δραστηριότητας.

Η παρακολούθηση της κίνησης οχημάτων εντός ενός αστικού οδικού δικτύου είναι αδιαμφισβήτητα ένα πεδίο που πληροί τις προδιαγραφές αυτές. Η κίνηση των οχημάτων στο δίκτυο δρόμων μιας πόλης επηρεάζει μεγάλη μερίδα της ανθρώπινης δραστηριότητας εντός της και αποτελεί παράγοντα που επιδέχεται εξέταση και βελτίωση.

Για την παρακολούθηση της κυκλοφορίας εντός του αστικού οδικού ιστού έχει κατά καιρούς προταθεί ποικιλία μεθόδων, μοντέλων και τεχνικών, πολλές εκ των οποίων έχουν ερείσματα σε διάφορους, εκ πρώτης όψεως ετερόκλητους επιστημονικούς τομείς. Ένας εκ των τομέων αυτών είναι και η Επιστήμη της Πληροφορικής, η οποία ασχολείται με την προτυποποίηση, διαχείριση, αποθήκευση και αποδοτική επεξεργασία των δεδομένων αυτών για την εξαγωγή χρήσιμων συμπερασμάτων.

-Παρουσίαση προβλήματος

Η παρακολούθηση της κίνησης των οχημάτων περιλαμβάνει μία σειρά από τεχνικές που επιτρέπουν στον αναλυτή να εξάγει συμπεράσματα για τη θέση των οχημάτων και τη μεταβολή της βάσει του χρόνου. Μπορούν να εξαχθούν συμπεράσματα για τις ιδιότητες της κίνησης ενός μεμονωμένου οχήματος, όπως την ταχύτητα και την επιτάχυνσή του, αλλά και τις τάσεις που διαμορφώνονται μαζικά από πλήθος οχημάτων, όπως την προτίμηση ή αποφυγή συγκεκριμένων κομματιών του δικτύου, τις τάσεις κίνησης προς συγκεκριμένες περιοχές ή τις τάσεις οδήγησης δια μέσου συγκεκριμένων περιοχών με σαφή χρονολογική σειρά.

Μία μέθοδος παρακολούθησης κίνησης οχημάτων που μπορεί να φανεί ιδιαίτερα χρήσιμη είναι η αναζήτηση *ακριβούς τροχιάς κίνησης* εντός του οδικού δικτύου. Ως αναζήτηση ακριβούς τροχιάς κίνησης ορίζεται η προσπάθεια εύρεσης τροχιών (συνήθως αποθηκευμένων σε μια Βάση Δεδομένων) που

ικανοποιούν κριτήρια διέλευσης από συγκεκριμένα κομμάτια του δικτύου οδών με καθορισμένη χρονολογική σειρά.

Η αναζήτηση ακριβούς τροχιάς μπορεί να εξάγει χρήσιμα συμπεράσματα σχετικά με [16]:

- Εντοπισμό ανωμαλιών κυκλοφορίας εντός του οδικού δικτύου, από την παρακολούθηση του χρόνου διέλευσης οχημάτων από συγκεκριμένες οδούς [20],
- Συντονισμό ελεγχόμενων διασταυρώσεων, όπου επιτυγχάνεται συγχρονισμός των φωτεινών σηματοδοτών για τη μεγιστοποίηση της ροής οχημάτων σε μια συγκεκριμένη διαδρομή [17],
- Ερευνητική αναζήτηση, κατά την οποία ζητώνται οχήματα που ακολουθούν συγκεκριμένη διαδρομή εντός συγκκεκριμένου χρονικού διαστήματος,
- Αναζήτηση της συχνότερα χρησιμοποιούμενης διαδρομής για προσπάθεια βελτίωσής της.

Οι παραπάνω εφαρμογές απαιτούν την εξαγωγή των τροχιών οχημάτων που ακολουθούν επακριβώς συγκεκριμένες διαδρομές εντός του δικτύου, χωρίς να παρεκκλίνουν από αυτές. Η εξαγωγή τροχιών που ακολουθούν ένα κομμάτι της διαδρομής, παρεκκλίνοντας από αυτή μπορούν να οδηγήσουν σε λανθασμένα συμπεράσματα ανάλυσης. Για τον σκοπό αυτό είναι απαραίτητη η ύπαρξη μεθόδου αναζήτησης ακριβούς τροχιάς για χωροχρονικά δεδομένα, που (ως συνηθίζεται) βρίσκονται αποθηκευμένα εντός μίας Βάσης Δεδομένων.

1.1 Χωροχρονικά Ευρετήρια

Ο ρόλος του ευρετηρίου στις Βάσεις Δεδομένων είναι ιδιαίτερα σημαντικός, διότι μέσω αυτού οι αναζητήσεις στα δεδομένα εκτελούνται σε χρόνους που είναι τάξεις μεγεθών μικρότεροι από τους χρόνους αναζήτησης χωρίς την ύπαρξη ευρετηρίων. Για το σκοπό αυτό έχει αναπτυχθεί και προταθεί κατά καιρούς πληθώρα δομών ευρετηρίων για την ευρετηριοδότηση πολλών τύπων δεδομένων. Πολλές από αυτές τις δομές είναι εξειδικευμένες στα χωρικά ή τα χωροχρονικά δεδομένα, διότι η περίπλοκή τους φύση (δεδομένα τριών ή τεσσάρων διαστάσεων) και οι ποικίλες εφαρμογές τους (που αποζητούν ιδιαίτερους τρόπους αναζήτησης) δεν βοηθούν στην εμφάνιση μιας ευρέως αποδεκτής λύσης που θα καλύπτει τις ανάγκες μεγάλης γκάμας εφαρμογών.

Ο χώρος των χωροχρονικών ευρετηρίων περιέχει διάφορες «ειδικότητες», με ευρετήρια να ειδικεύονται σε τοπολογικές σχέσεις αντικειμένων, όπως τα R-tree [12], R*-tree [2] ευρετήρια που χρησιμοποιούνται για αναζήτηση χωροχρονικών τροχιών κινούμενων αντικειμένων όπως τα TMN-tree [7], STR-tree [22], TB-tree [22], SETI [5], ευρετήρια που χρησιμοποιούνται για την ευρετηριοδότηση τροχιών δεσμευμένων σε τοπολογία δικτύου όπως τα MON-tree [25], FNR-tree [11], Network mapping [21], PARINET [23], Signature-based [6].

-Ευρετηριοδότηση δικτύων

Στον τομέα της ευρετηριοδότησης χωροχρονικών δεδομένων δεσμευμένων σε οδικό δίκτυο, πολλές προσεγγίσεις αντιμετωπίζουν αποδοτικά τις προκλήσεις της ευρετηριοδότησης και της υποβολής ερωτημάτων προς τη Βάση Δεδομένων. Ωστόσο, σε καμία προσέγγιση δεν δίνεται ιδιαίτερο βάρος στα ερωτήματα ακριβούς τροχιάς, καθιστώντας την εκτέλεσή τους μια δύσκολη και μη αποδοτική διαδικασία.

Στην σχετική βιβλιογραφία η τοπολογία οδικού δικτύου αναπαρίσταται είτε με βάση την ακμή του δικτύου (edge) είτε με βάση το στοιχειώδες τμήμα (segment) είτε με βάση τη διαδρομή (route). Ως ακμή ορίζεται η γραμμή που συνδέει δύο διασταυρώσεις, ως τμήμα ένα κομμάτι της ακμής μεταξύ δύο οποιονδήποτε διαδοχικών σημείων που την απαρτίζουν, χωρίς απαραίτητα αυτά να είναι διασταυρώσεις, ενώ ως διαδρομή ορίζεται ένα σύνολο ακμών του δικτύου.

-MON-tree

Η δομή ευρετηρίου MON-tree [25] είναι ένας συνδυασμός ενός δισδιάστατου R-tree (πρωτεύον) με έναν πίνακα κατακερματισμού (hash table) και ένα «δάσος» δισδιάστατων R-trees (δευτερεύοντα), κάθε ένα από τα οποία συνδέονται με τους κόμβους-φύλλα του πρωτεύοντος δένδρου. Κατά τις λειτουργίες αναζήτησης, εισαγωγής και ανανέωσης της δομής, το πρωτεύον R-tree υποβοηθούμενο από τη δομή του πίνακα κατακερματισμού, λειτουργεί σαν χωρικό ευρετήριο για την αναζήτηση των δευτερευόντων R-trees, τα οποία χρησιμοποιούνται για την ευρετηριοδότηση της πληροφορίας κίνησης των αντικειμένων. Η δομή MON-tree υποστηρίζει δικτυακές αναπαραστάσεις με βάση την ακμή, το τμήμα και τη διαδρομή.

Για την εύρεση τροχιών που ακολουθούν συγκεκριμένη διαδρομή, ανακτώνται οι ακμές του δικτύου που αντιστοιχούν στη διαδρομή και πραγματοποιείται αναζήτηση στα δευτερεύοντα R-trees που αντιστοιχούν στις ακμές. Οι τροχιές που ακολουθούν ολόκληρη τη διαδρομή χωρίς παρεκκλίσεις βρίσκονται εντός των αποτελεσμάτων για κάθε δευτερεύον R-tree.

-PARINET

Μία από τις αποδοτικότερες προσεγγίσεις για την ευρετηριοδότηση κινούμενων αντικειμένων υπό περιορισμούς δικτύου που επίσης υποστηρίζει δικτυακές αναπαραστάσεις με βάση την ακμή, το τμήμα και τη διαδρομή είναι η προσέγγιση PARINET [23]. Στην προσέγγιση αυτή, κάθε ακμή του δικτύου κωδικοποιείται με μια συγκεκριμένη τιμή βάρους, ανάλογη του αριθμού τροχιών που διέρχονται από αυτή. Μέσω ενός αλγορίθμου κατακερματισμού, το δίκτυο διαχωρίζεται σε περιοχές με περίπου ίσο συνολικό βάρος ακμών και χωρικό μέγεθος, προσαρμόζοντας το ευρετήριο στο εκάστοτε σύνολο δεδομένων.

Τα τμήματα του δικτύου που δημιουργούνται από τον αλγόριθμο κατακερματισμού είναι ανεξάρτητα μεταξύ τους και αποθηκεύονται σε διαφορετικά σημεία στο δίσκο, κατανέμοντας τις αναγνώσεις και εγγραφές στο δίσκο, συμβάλλοντας στη βελτίωση της απόδοσης του αλγορίθμου. Για την εισαγωγή μιας νέας τροχιάς στη δομή του PARINET, εισάγεται μια εγγραφή σε κάθε χωρίο του δίσκου που αντιστοιχεί σε περιοχή από την οποία διέρχεται η τροχιά.

Η αναζήτηση τροχιών που ακολουθούν συγκεκριμένη διαδρομή στο PARINET γίνεται με αντίστοιχο τρόπο, όπως και στο MON-tree. Αρχικά ανακτώνται οι ακμές της αρχικής διαδρομής και τα χωρία του δίσκου στα οποία βρίσκονται τα δεδομένα για κάθε συγκεκριμένη ακμή. Μετέπειτα εκτελείται αναζήτηση τροχιών που ακολουθούν κάθε ακμή ξεχωριστά και επιστρέφονται τα αντίστοιχα αποτελέσματα, τα οποία φιλτράρονται με ανάκτηση όλης της τροχιάς και έλεγχο ως προς τη συμφωνία της με την αρχική διαδρομή. Η απόδοση διατηρείται σε υψηλά επίπεδα λόγω της δυνατότητας αποθήκευσης πολλαπλών ακμών του δικτύου σε ένα χωρίο του δίσκου, γεγονός που διατηρεί το πλήθος αναγνώσεων από το δίσκο χαμηλό.

1.2 Πρόταση [16]

Η επίλυση του προβλήματος απαιτεί έναν τύπο χωρικού ευρετηρίου και μια σειρά ερωτημάτων Βάσης Δεδομένων, ειδικευμένων στην αναζήτηση τροχιών που ακολουθούν επακριβώς μια διαδρομή εντός ενός δικτύου οδών. Ένα ευρετήριο αυτού του τύπου, καθώς και τα αντίστοιχα ερωτήματα για την αποδοτική χρησιμοποίησή του έχουν προταθεί από τους Krogh κ.α. [16]. Το εν λόγω ευρετήριο ονομάζεται NETTRA (NETwork-constained TRAjectory) και στηρίζεται στην κωδικοποίηση των ακμών του δικτύου με έναν μοναδικό αριθμό για κάθε ακμή και την αναπαράσταση των διαδρομών εντός του δικτύου σαν μια ακολουθία αυτών των ακμών.

Για κάθε τροχιά του συνόλου δεδομένων, εντοπίζονται οι ακμές δικτύου από τις οποίες έχει διέλθει η τροχιά με την κατάλληλη χρονολογική σειρά. Για κάθε μία από αυτές τις ακμές, αποθηκεύεται μια εγγραφή σε έναν πίνακα δεδομένων η οποία περιέχει το αναγνωριστικό της τροχιάς, το αναγνωριστικό της ακμής, τον χρόνο εισόδου και εξόδου του κινούμενου αντικειμένου από την ακμή και το αναγνωριστικό που έχει αποδοθεί στην ακμή κατά τη διαδικασία κωδικοποίησης. Ακόμη, ακολουθείται συγκεκριμένη μέθοδος αθροιστικής παρουσίασης των αναγνωριστικών κωδικοποίησης, μέσω της οποίας γίνεται σαφής και η ακολουθία των ακμών από τις οποίες έχει διέλθει η εξεταζόμενη τροχιά.

Με χρήση του πίνακα δεδομένων που δημιουργείται από αυτή τη διαδικασία και πραγματοποιώντας αναζήτηση βάσει της αρχικής και της τελικής ακμής της τροχιάς ενός αντικειμένου χρησιμοποιώντας τα ερωτήματα που παρουσιάζονται [16], μπορεί να αναγνωριστεί αν η εξεταζζόμενη τροχιά ακολουθεί συγκεκριμένη διαδρομή χωρίς να παρεκκλίνει καθόλου από αυτή.

Η υλοποίηση του ευρετηρίου NETTRA γίνεται σε απλή SQL, δίνοντάς του πλήρη διαλειτουργικότητα με τα περισσότερα Συστήματα Διαχείρισης Βάσεων Δεδομένων.

Πέραν του NETTRA, στην παρούσα εργασία παρουσιάζονται:

- Οι διάφοροι τύποι (ακριβείς, προσεγγιστικοί) των ερωτημάτων αυστηρούς τροχιάς (SPQs),
- Μία βελτιωμένη ακριβής προσέγγιση ερωτήματος SPQ,
- Μία «πρακτική» υλοποίηση SPQ η οποία απαντάει στο ερώτημα με αναζήτηση μόνο για την πρώτη και την τελευταία ακμή της τροχιάς,
- Παραδείγματα χρήσης του NETTRA, μέσω της εκτέλεσης SPQs.

1.3 Δομή τόμου εργασίας

Η παρούσα εργασία αποτελείται από πέντε (5) κεφάλαια και ένα παράρτημα. Στο παρόν πρώτο κεφάλαιο διατυπώθηκε το πρόβλημα που αντιμετωπίζεται στην παρούσα εργασία και τον τρόπο που αντιμετωπίζεται ως τώρα από ήδη υπάρχουσες δομές ευρετηρίου. Στη συνέχεια έγινε σαφής ο σκοπός της εργασίας με τη διατύπωση της πρότασης επίλυσης του προβλήματος.

Στο δεύτερο κεφάλαιο του τόμου της εργασίας γίνεται θεωρητική προσέγγιση του προβλήματος και μαθηματική διατύπωσή του, ενώ γίνεται και μια πρώτη αναλυτική περιγραφή των μεθόδων που θα ακολουθηθούν για την επίλυση του προβλήματος.

Στο τρίτο κεφάλαιο γίνεται αναφορά στις πλατφόρμες και πακέτα λογισμικού που χρησιμοποιήθηκαν για την υλοποίηση της παρούσας λύσης και την εμφάνιση των αποτελεσμάτων της.

Στο τέταρτο κεφάλαιο παρουσιάζεται με τεχνικές λεπτομέρειες η υλοποίηση της λύσης που προτάθηκε, δίδονται παραδείγματα χρήσης της μέσω εκτέλεσης ερωτημάτων και αναλύεται η απόδοση των αλγορίθμων και των ερωτημάτων που την απαρτίζουν.

Στο πέμπτο κεφάλαιο παρουσιάζονται τα συμπεράσματα από την υλοποίηση της λύσης και ασκείται κριτική στα αδύναμα σημεία της, με παράλληλη παράθεση προτάσεων για βελτίωσή της.

2. Θεωρητικό υπόβαθρο εφαρμογής [16]

Όπως αναλύθηκε σε προηγούμενες παραγράφους της παρούσας εργασίας, το πρόβλημα που καλείται να αντιμετωπίσει η παρούσα εφαρμογή είναι η εύρεση των τροχιών που διέρχονται από συγκεκριμένο κομμάτι του δικτύου εντός συγκεκριμένου χρονικού διαστήματος.

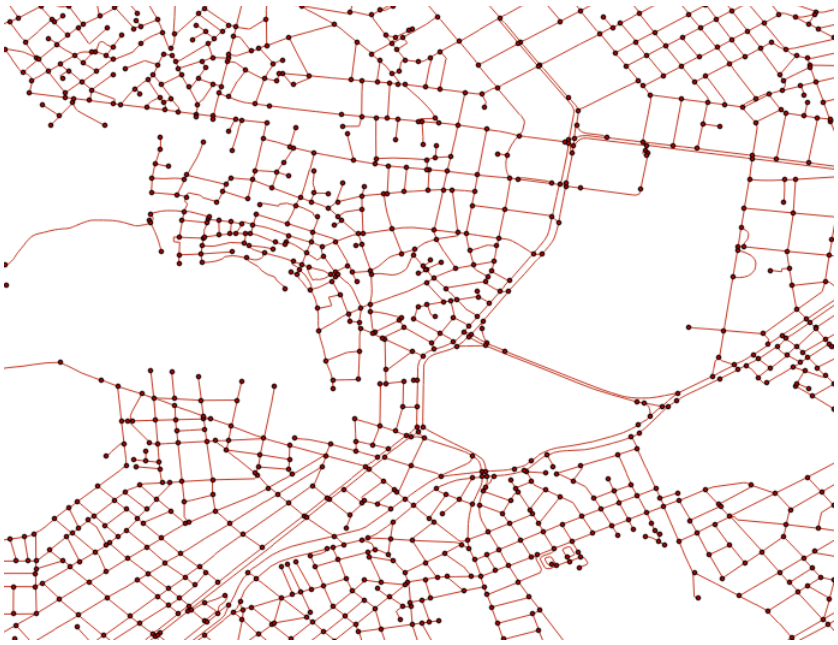
Για την επίλυση του προβλήματος και την εύρεση λύσης χρησιμοποιείται ένα σύνολο από μεθοδολογίες για την κατάλληλη μοντελοποίηση των δεδομένων, έχοντας ως απώτερο στόχο την δημιουργία ενός ευρετηρίου με όλη την απαραίτητη πληροφορία για την αποδοτική εκτέλεση ερωτημάτων ακριβούς τροχιάς. Οι μεθοδολογίες που παρουσιάζονται στο παρόν κεφάλαιο έχουν προταθεί [16] από τους Krogh B., Pelekis N., Theodoridis Y., Torp K.

Ακόμη, προτείνονται τεχνικές βελτίωσης του ευρετηρίου και της δεικτοδότησης των ακμών του χαρτογραφικού υποβάθρου, με σκοπό την βελτίωση της ακρίβειας και της αξιοπιστίας του αλγορίθμου επιλογής τροχιών και την εξάλειψη ψευδών αποτελεσμάτων.

2.1 Μοντέλο δεδομένων [16]

-Δεδομένα υποβάθρου

Τα δεδομένα του χαρτογραφικού υποβάθρου ουσιαστικά αποτελούν το δίκτυο της περιοχής εντός της οποίας κινούνται τα αντικείμενα του συνόλου δεδομένων που εξετάζεται. Το δίκτυο οδών αναπαρίσταται μέσω ενός κατευθυνόμενου γράφου $G=(V, E)$, με V το σύνολο κόμβων και E το σύνολο κατευθυνόμενων ακμών του γράφου, για τις οποίες ισχύει $E \subseteq VxV$.



Εικόνα 2.1: Δίκτυο οδών. Διακρίνονται οι κόμβοι και οι ακμές του δικτύου

-Δεδομένα τροχιών αντικειμένων

Τα δεδομένα κίνησης αντικειμένων (τροχιές αντικειμένων) είναι σύνολο από σημεία με σαφή θέση στον τρισδιάστατο χώρο *(x, y, t)*, όπου *x* το γεωγραφικό μήκος (longitude), *y* το γεωγραφικό πλάτος (latitude) και *t* ο χρόνος (time). Τα σημεία αυτά, προβαλλόμενα στο δισδιάστατο χώρο *(x, y)*, απεικονίζουν στο χάρτη τις διαδοχικές θέσεις από τις οποίες περνά το κινούμενο αντικείμενο με την πάροδο του χρόνου.

Εικόνα 2.2: Απεικόνιση θέσεων αντικειμένου στο χάρτη

Ένα κινούμενο αντικείμενο αναφέρει τη θέση του μέσω μιας συσκευής γεωγραφικού εντοπισμού (GPS) σε χρονικές στιγμές που συνήθως ακολουθούν σταθερή περιοδικότητα. Κάθε αναφορά θέσης αποτελεί μια πλειάδα παραμέτρων τύπου

$$loc=(moid, ts, pos),$$

όπου *moid* είναι το αναγνωριστικό του κινούμενου αντικειμένου, *ts* μία συγκεκριμένη χρονική στιγμή και *pos* η θέση του αντικειμένου την χρονική στιγμή *ts* με μορφή χωρικού προσδιορισμού (συντεταγμένες).

Κάθε τεχνική ευρετηριοδότησης κίνησης με περιορισμούς δικτύου απαιτεί την αντιστοίχιση των σημείων θέσης των κινούμενων αντικειμένων στο δίκτυο οδών εντός του οποίου κινούνται. Με τον τρόπο αυτό, κάθε σημείο θέσης αντιστοιχίζεται σε κατάλληλη ακμή του δικτύου οδών βάσει της θέσης του, των περιορισμών που ορίζει το δίκτυο και άλλων στοιχείων της κίνησής του, όπως η ταχύτητά του. Με κατάλληλη επεξεργασία τα αρχικά σημεία προσδιορισμού θέσης αντιστοιχίζονται σε μια ακολουθία ακμών του δικτύου, ορίζοντας με αυτό τον τρόπο μία τροχιά εντός του δικτύου. Κάθε εγγραφή που ανήκει σε αυτή την ακολουθία είναι μια πλειάδα παραμέτρων

$$loc_{mm}=(t_{id}, e_{id}, ts_{enter}, ts_{leave}),$$

όπου t_{id} αποτελεί ένα αναγνωριστικό τροχιάς, e_{id} ένα αναγνωριστικό της ακμής του δικτύου, ts_{enter} το χρόνο εισόδου του αντικειμένου στην ακμή του δικτύου, ts_{leave} το χρόνο εξόδου του αντικειμένου από την ακμή του δικτύου. Οι χρόνοι εισόδου και εξόδου από την ακμή του δικτύου υπολογίζονται με γραμμική παρεμβολή.

Η τροχιά $t=[loc_{mm1}, loc_{mm2}, ..., loc_{mmn}]$, όπως προαναφέρθηκε αποτελείται από μια ακολουθία σημείων αναφοράς θέσης αντιστοιχισμένων σε ακμές του δικτύου, που αντιστοιχούν σε ένα αντικείμενο και σε μία διαδρομή του. Συνεπώς, στην ίδια τροχιά δεν μπορούν να περιλαμβάνονται θέσεις άλλων αντικειμένων (διαφορετικό *moid*) και θέσεις μεταξύ των οποίων μεσολαβεί χρονικό διάστημα σημαντικά μεγαλύτερο από την περίοδο αναφοράς θέσης της συσκευής GPS. Στην περίπτωση αυτή τα δεδομένα μπορούν να δημιουργήσουν λανθασμένα αποτελέσματα κατά την αντιστοίχιση των θέσεων στις ακμές του δικτύου.

Για απλούστερο συμβολισμό, ορίζονται οι συναρτήσεις

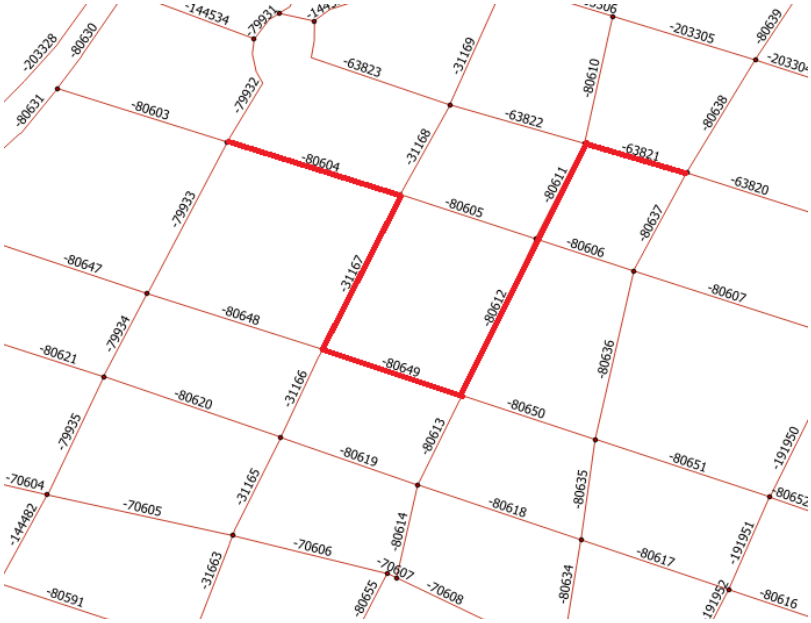
- *path(t):* Διαδρομή που ακολουθείται από την τροχιά t.
- *entertimes(t):* Ταξινομημένη λίστα χρονικών στιγμών εισόδου τροχιάς σε κάθε ακμή.
- *leavetimes(t):* Ταξινομημένη λίστα χρονικών στιγμών εξόδου τροχιάς από κάθε ακμή.

και το σύνολο των τροχιών του συνόλου δεδομένων συμβολίζεται ως $T=\{t_1, t_2, ..., t_n\}$.

2.2 Προσδιορισμός προβλήματος [16]

Ορίζουμε ως διαδρομή π μια λίστα ακμών δικτύου ταξινομημένη βάσει της σειράς προσπέλασης κάθε ακμής. Συγκεκριμένα, έχουμε

$$\pi=[e_1, e_2, ..., e_n],\ e_i \in E$$



Εικόνα 2.3: Η διαδρομή π=[-80604, -31167, -80649, -80612, -80611, -63821]

Η εύρεση των τροχιών που ακολουθούν συγκεκριμένες ακμές του δικτύου γίνεται μέσω της εκτέλεσης ειδικών ερωτημάτων προς τη Βάση Δεδομένων, που ονομάζονται Ερωτήματα Ακριβούς Διαδρομής (Strict Path Queries, SPQs).

Ένα SPQ για μια διαδρομή δικτύου π επιστρέφει σαν αποτέλεσμα όλες τις τροχιές αντικειμένων που ακολουθούν ακριβώς τη διαδρομή π, δηλαδή τις τροχιές που διέρχονται από τις ακμές του δικτύου που απαρτίζουν τη διαδρομή π, με την κατάλληλη σειρά προσπέλασης και εντός του χρονικού διαστήματος $[tm_{start}, tm_{end}]$.

Η συνάρτηση *SubList(path(t), i, j)* επιστρέφει τα τμήματα της τροχιάς t (η οποία ανήκει στο σύνολο τροχιών T) μεταξύ των θέσεων i και j $(i<j)$, περιλαμβάνοντας και την ακμή που αντιστοιχεί στο δείκτη i. Κατά συνέπεια, το αποτέλεσμα ενός SPQ περιλαμβάνει κάθε τροχιά t από το σύνολο T που έχει την διαδρομή π σαν υπο-διαδρομή της, εισέρχεται στην π τη χρονική στιγμή t_{start} ή αργότερα και εξέρχεται από την π τη χρονική στιγμή t_{end} ή νωρίτερα.

Τυπικά το SPQ ορίζεται ως εξής:

$$SPQ(\pi, tm_{start}, tm_{end}) = \left\{ t \middle| \begin{array}{c} t \in T, \exists i,j : \pi = SubList(path(t), i, j+1) \\ \cap\, tm_{start} \leq entertimes(t)[i] \\ \cap\, tm_{end} \geq leavetimes(t)[j] \end{array} \right\}$$

Συγκρινόμενο με το απλό ερώτημα διαδρομής (Path Query – PQ) [23] για το οποίο ισχύει

$$PQ(\pi, tm_{start}, tm_{end}) = \left\{ t \middle| \begin{array}{c} t \in T, \exists (e, ts_{enter}, ts_{leave}) \in t = e \ \in \pi \\ \cap\, (tm_{start} \leq ts_{enter} \leq tm_{end} \cup tm_{start} \leq ts_{leave} \leq tm_{end}) \end{array} \right\}$$

γίνεται εύκολα αντιληπτό ότι για το PQ αρκεί η διαδρομή να διέρχεται από μία ακμή της διαδρομής π εντός του χρονικού διαστήματος $[tm_{start}, tm_{end}]$, ενώ για το SPQ τα αποτελέσματα ακολουθούν όλη τη διαδρομή π εντός του χρονικού διαστήματος $[tm_{start}, tm_{end}]$. Λόγω των αυστηρότερων κριτηρίων που περιλαμβάνει ένα SPQ έναντι ενός PQ, τα αποτελέσματα του SPQ είναι πάντα υποσύνολο των αντίστοιχων αποτελεσμάτων ενός PQ για το ίδιο σύνολο δεδομένων, την ίδια διαδρομή π και το ίδιο χρονικό διάστημα αναζήτησης. Ισχύει δηλαδή

$$SPQ(\pi, tm_{start}, tm_{end}) \subseteq PQ(\pi, tm_{start}, tm_{end}).$$

Πράγματι, για τη διαδρομή $\pi=[e_{bc}]$ της Εικόνας 2.4 ισχύει

$$SPQ(e_{bc}) \subseteq PQ(e_{bc}) \text{ αφού } \{t_1, t_2, t_3, t_4\} \subseteq \{t_1, t_2, t_3, t_4\}$$

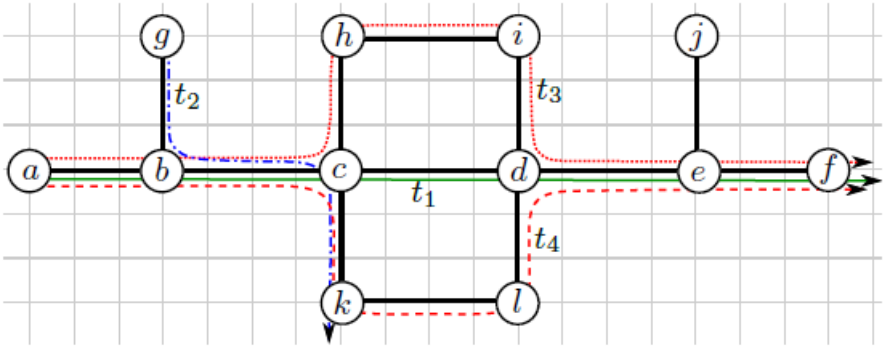
ενώ για τη διαδρομή $\pi=[e_{ab}, e_{bc}, e_{cd}, e_{de}, e_{ef}]$ ισχύει

$$SPQ(e_{ab}, e_{bc}, e_{cd}, e_{de}, e_{ef}) \subseteq PQ(e_{ab}, e_{bc}, e_{cd}, e_{de}, e_{ef}) \text{ αφού } \{t_1\} \subseteq \{t_1, t_2, t_3, t_4\}.$$

Σημείωση: Για την αποφυγή σύγχυσης, από εδώ και στο εξής τα δομικά κομμάτια της διαδρομής π θα ονομάζονται *ακμές της διαδρομής*, ενώ τα αντίστοιχα κομμάτια των τροχιών t των αντικειμένων θα ονομάζονται *τμήματα της τροχιάς t*.

2.3 Το ευρετήριο NETTRA [16]

Το ευρετήριο NETTRA, για να μπορέσει να επιτελέσει το σκοπό του, δηλαδή την ανάκτηση των τροχιών που ακολουθούν συγκεκριμένη διαδρομή του δικτύου, στηρίζεται στην απόδοση συγκεκριμένης κωδικοποίησης σε όλες τις ακμές του δικτύου. Η κωδικοποίηση αυτή καθώς και αριθμητικές πράξεις που γίνονται στις τιμές της, χρησιμοποιούνται σαν κριτήρια συνθηκών για την εξαγωγή συμπερασμάτων ως προς την ομοιότητα των τροχιών με την δεδομένη διαδρομή δικτύου.



Εικόνα 2.4: Δίκτυο και τροχιές [16]

Ακολούθως, εντός του παρόντος κεφαλαίου παρουσιάζονται τρεις προσεγγίσεις (δύο ακριβείς και μία προσεγγιστική) για την εύρεση των τροχιών που ακολουθούν μια συγκεκριμένη διαδρομή εντός δικτύου, οι οποίες χρησιμοποιούν την κωδικοποίηση ακμών και διαδρομής για την εξαγωγή συμπερασμάτων.

Η πρώτη ακριβής λύση εξάγει ακριβή αποτελέσματα εκτελώντας αναζητήσεις για κάθε ακμή της διαδρομής π, η δεύτερη ακριβής λύση πραγματοποιεί αναζήτηση για ορισμένες ακμές της διαδρομής π και η προσεγγιστική λύση πραγματοποιεί αναζητήσεις μόνο για την πρώτη και την τελευταία ακμή της διαδρομής π.

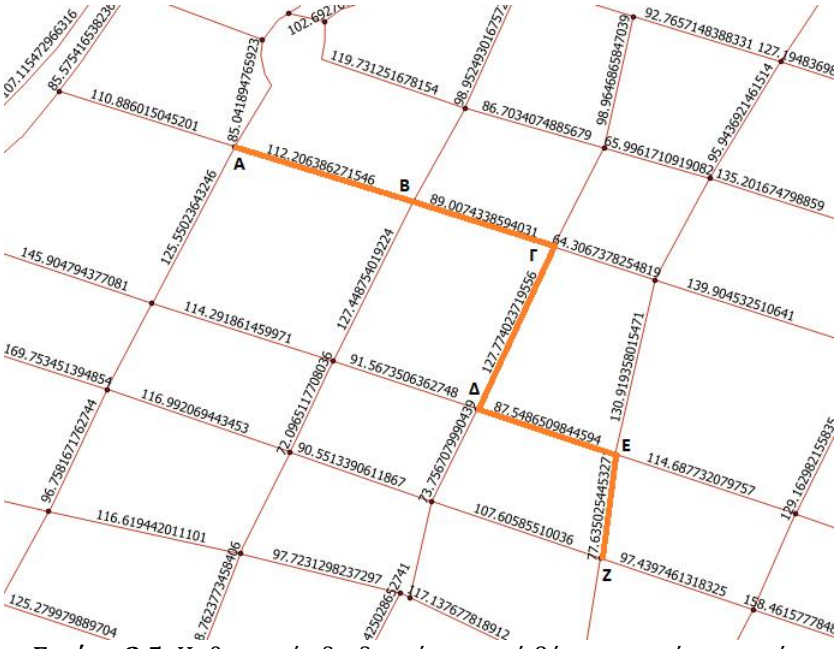
- Αναπαράσταση τροχιών και κωδικοποίηση

Για την εύρεση των αποτελεσμάτων ενός SPQ, η κίνηση των οχημάτων πρέπει να περιοριστεί βάσει της γεωμετρίας του δικτύου. Συνεπώς, όπως έχει αναφερθεί, οι θέσεις των οχημάτων πρέπει να αντιστοιχιστούν σε διαδρομές εντός του δικτύου που χρησιμοποιείται σαν οδικό υπόβαθρο για την περιοχή κίνησης των οχημάτων. Με αυτό τον τρόπο μπορεί να προσδιοριστεί επακριβώς η διαδρομή που έχει ακολουθήσει κάθε κινούμενο αντικείμενο και συνεπώς και οι ακμές του δικτύου από τις οποίες έχει διέλθει.

-Κωδικοποίηση

Για τον ακριβή προσδιορισμό των ακμών από τις οποίες έχει διέλθει μια τροχιά δεν επαρκεί η απλή αντιστοίχιση των δεδομένων θέσης σε ακμές του δικτύου και ο μετέπειτα έλεγχος αν η τροχιά διέρχεται από κάθε ακμή της π. Μία τέτοια προσέγγιση παρουσιάζει αδυναμία όταν η τροχιά επισκέπτεται όλες τις ακμές της διαδρομής π, αλλά με λανθασμένη σειρά. Σε αυτή την περίπτωση η πιθανότητα εξαγωγής λανθασμένου αποτελέσματος αυξάνεται δραματικά.

Μία προσέγγιση για την επίλυση του προβλήματος είναι η ανάθεση σε κάθε ακμή του δικτύου μιας τιμής-βάρους (μεγαλύτερης του μηδενός) και η εκχώρηση της τιμής αυτής σε κάθε τμήμα τροχιάς που διέρχεται από την εν λόγω ακμή. Αν η εκχώρηση της τιμής αυτής γίνεται αθροιστικά στην αντίστοιχη τιμή του προηγούμενου τμήματος τροχιάς, αναπαρίσταται και η σειρά διέλευσης της τροχιάς από τις ακμές του δικτύου.



Εικόνα 2.5: Καθορισμός διαδρομής με τιμή βάρους το μήκος ακμής

Για παράδειγμα, στην Εικόνα 2.5 αναπαρίσταται μια τροχιά για τα τμήματα της οποίας (κάθε ένα αντιστοιχεί σε μία ακμή) η τιμή βάρους θα υπολογιστεί στις τιμές του Πίνακα 2.1.

Τμήμα τροχιάς	Μήκος ακμής δικτύου	Βάρος τμήματος τροχιάς (hash)
e_{AB}	112,206386	112,206386
$e_{B\Gamma}$	89,007433	201,213819
$e_{\Gamma\Delta}$	127,774023	328,987842
$e_{\Delta E}$	87,548650	416,536492
e_{EZ}	77,635025	494,171517

Πίνακας 2.1: Υπολογισμός βάρους τμημάτων τροχιών

Συνεπώς , για τα τμήματα της διαδρομής $\pi = [e_1, e_2, \dots, e_n]$ θα ισχύει

$$hash(\pi, i) = \sum_{j=1}^{i} \pi[j].weight$$

Ένας απλός και αποδοτικός τρόπος να αναπαρασταθεί αυτή η αντιστοίχιση είναι με έναν απλό πίνακα, στον οποίο αποθηκεύονται πληροφορίες σχετικά με το αναγνωριστικό της τροχιάς, τον αριθμό τμήματος της τροχιάς, την ακμή στην οποία αντιστοιχεί το τμήμα της τροχιάς και τους χρόνους εισόδου και

εξόδου της τροχιάς στην ακμή. Στον πίνακα αυτόν μπορούν να προστεθούν και τα υπολογισμένα βάρη για κάθε τμήμα των τροχιών. Με αυτόν τον τρόπο για τις τροχιές της Εικόνας 2.4 (μήκος πλευράς κανάβου 1) φτάνουμε σε έναν πίνακα αντίστοιχο του Πίνακα 2.2. Όπως φαίνεται από τον πίνακα, ισχύει *hash(t_1, 2)=7*, *hash(t_2,3)=10*.

t_{id}	e_{id}	ts_{enter}	ts_{leave}	$hash$
t_1	e_{ab}	9	11	3
t_1	e_{bc}	11	13	7
t_1	e_{cd}	13	17	11
t_1	e_{de}	17	21	15
t_1	e_{ef}	21	23	18
t_2	e_{gb}	14	16	3
t_2	e_{bc}	16	19	7
t_2	e_{ck}	19	21	10
t_3	e_{ab}	14	16	3
t_3	e_{bc}	11	13	7
t_3	e_{ch}	13	15	10
t_3	e_{hi}	15	18	14
t_3	e_{id}	18	21	17
t_3	e_{de}	21	24	21
t_3	e_{ef}	24	27	24
t_4	e_{ab}	14	16	3
t_4	e_{bc}	16	20	7
t_4	e_{ck}	20	22	10
t_4	e_{kl}	22	25	14
t_4	e_{ld}	25	28	17
t_4	e_{de}	28	30	21
t_4	e_{ef}	30	33	24

Πίνακας 2.2: Πίνακας τμημάτων τροχιών με hash βάρη (visitedsegments)

Η προσέγγιση που περιγράφηκε είναι παρόμοια με αντίστοιχες προσεγγίσεις στα [25], [11], [23]. Ωστόσο, η προσθήκη της στήλης *hash* δεν παρατηρείται σε αυτές τις προσεγγίσεις.

2.4 Προσεγγιστικό SPQ με μέτρο απόστασης [16]

Δεδομένου ενός συνόλου υποψήφιων τροχιών, η στήλη αθροιστικού βάρους *(hash)* χρησιμοποιείται για το φιλτράρισμα των αποτελεσμάτων, ώστε να επιστραφούν όσες τροχιές πραγματικά ακολουθούν τη διαδρομή π που δίδεται σαν είσοδος στο SPQ. Συγκεκριμένα, ελέγχεται αν η διαφορά στις τιμές του *hash* για την ακμή εισόδου και την ακμή εξόδου από την διαδρομή π είναι ακριβώς ίδια με τη διαφορά μεταξύ των τιμών *hash* για την αρχική και τελική ακμή της διαδρομής π. Ελέγχεται δηλαδή αν ισχύει

$$hash_{end} - hash_{start} = deltahash(\pi)$$

όπου

$$deltahash(\pi) = \sum_{j=2}^{n} \pi[j].weight$$

Μία απλή, μη ακριβής προσέγγιση για την εκτέλεση ενός SPQ επιστρέφει ένα σύνολο υποψήφιων τροχιών και φιλτράρει το αποτέλεσμα με χρήση των τιμών *hash*. Θεωρώντας το δίκτυο και της τροχιές της Εικόνας 2.4, το SPQ για μια υποτιθέμενη διαδρομή $\pi = [e_{ab}, e_{bc}, e_{cd}, e_{de}, e_{ef}]$ θα πρέπει να επιστρέψει

μόνο την διαδρομή t_1, καθώς μόνο αυτή ακολουθεί ακριβώς τη διαδρομή π. Ελέγχοντας τις τροχιές μέσω του Πίνακα 2.1 παρατηρούμε ότι οι τροχιές t_1, t_3 και t_4 διέρχονται από την αρχική και την τελική ακμή της διαδρομής π, συνεπώς ανήκουν στο σύνολο υποψήφιων τροχιών για το SPQ(π). Για τις υποψήφιες τροχιές ισχύει

$$deltahash(t_1) = hash_{end}(t_1) - hash_{start}(t_1) = 18 - 3 = 15$$
$$deltahash(t_3) = hash_{end}(t_3) - hash_{start}(t_3) = 24 - 3 = 21$$
$$deltahash(t_4) = hash_{end}(t_4) - hash_{start}(t_4) = 24 - 3 = 21$$

ενώ για τη διαδρομή π ισχύει

$$deltahash(\pi) = e_{ab}.weight + e_{bc}.weight + e_{cd}.weight + e_{de}.weight + e_{ef}.weight = 4 + 4 + 4 + 3 = 15.$$

Άρα, εφόσον η τροχιά t_1 διέρχεται από την αρχική και τελική ακμή της διαδρομής π και ισχύει $deltahash(t_1) = deltahash(\pi)$ (κάτι που για τις τροχιές t_3 και t_4 δεν ισχύει), η τροχιά t_1 ακολουθεί πλήρως τη διαδρομή π χωρίς παρεκκλίσεις και είναι η μόνη που επιστρέφεται σαν αποτέλεσμα για το SPQ(π).

Συμπεραίνουμε λοιπόν ότι με αυτό τον τρόπο μπορούμε να εκτελέσουμε SPQ πραγματοποιώντας έλεγχο διέλευσης μόνο για την αρχική και την τελική ακμή της διαδρομής π και φιλτράροντας τα αποτελέσματα βάσει της διαφοράς τιμής βάρους *hash* μεταξύ των τμημάτων τροχιάς που διέρχονται από την πρώτη και την τελευταία ακμή της διαδρομής π αντίστοιχα. Η προσέγγιση αυτή μπορεί να διατυπωθεί σε απλή SQL ως εξής:

```
select e_start.tid
from visitedsegments as e_start
join visitedsegments as e_end using(tid)
where e_start.eid=first(π)
and e_end.eid=last(π)
and e_end.hash-e_start.hash=deltahash(π)
```

Ερώτημα 2.1. Πρώτη προσέγγιση SPQ

όπου *tid* το αναγνωριστικό της τροχιάς, e_{start} το τμήμα της τροχιάς *tid* που διέρχεται από την αρχική ακμή της π, e_{end} το τμήμα της τροχιάς *tid* που διέρχεται από την τελευταία ακμή της π, *first(π)* συνάρτηση που επιστρέφει την πρώτη ακμή της διαδρομής π, *last(π)* συνάρτηση που επιστρέφει την τελευταία ακμή της διαδρομής π, *eid* το αναγνωριστικό ακμής στο οποίο αντιστοιχεί το συγκεκριμένο τμήμα τροχιάς, *hash* η τιμή βάρους για το συγκεκριμένο τμήμα τροχιάς.

Το αδύναμο σημείο αυτής της προσέγγισης είναι ότι δεν εξαλείφει την πιθανότητα για ύπαρξη λανθασμένων αποτελεσμάτων σε περίπτωση που κάποια τροχιά δεν ακολουθεί επακριβώς την διαδρομή π αλλά διέρχεται από την αρχική και την τελική της ακμή, ενώ ισχύει $deltahash(t) = deltahash(\pi)$. Σε αυτές τις περιπτώσεις η τιμή του μήκους των ακμών δεν επαρκεί για τον επιτυχή διαχωρισμό μέσω κωδικοποίησης των ακμών του δικτύου και τον προσδιορισμό της διαδρομής που έχει ακολουθηθεί. Παράδειγμα από την Εικόνα 2.4 αποτελούν οι τροχιές t_3 και t_4, οι οποίες αν και έχουν ίδιο ακριβώς μήκος και ισχύει γι αυτές $deltahash(t_3) = deltahash(t_4)$ και $first(t_3) = first(t_4)$, $last(t_3) = last(t_4)$, ακολουθούν διαφορετικές διαδρομές εντός του δικτύου.

2.5 Ακριβές SPQ με μέτρο απόστασης [16]

Για την αντιμετώπιση του προβλήματος της ύπαρξης λανθασμένων αποτελεσμάτων του Ερωτήματος 2.1 μπορεί να υιοθετηθεί μια απλοϊκή λύση η οποία εγγυάται το σωστό αποτέλεσμα, χωρίς ωστόσο να είναι το ίδιο αποδοτική με την προηγούμενη.

Έστω μια διαδρομή π εντός του δικτύου η οποία αποτελείται μόνο από δύο ακμές. Στην περίπτωση αυτή, η πρώτη ακμή είναι η αρχική (first(π)) και η δεύτερη η τελική (*last(π)*), δηλαδή ισχύει *π=[first(π),last(π)]*. Η διαφορά βαρών μεταξύ τους είναι ίση με *deltahash(last(π),first(π))=deltahash(π)*. Για τη διαδρομή αυτή, και για κάθε διαδρομή εντός του δικτύου μήκους δύο ακμών, το *deltahash(π)* χαρακτηρίζει μοναδικά τη διαδρομή και το αποτέλεσμα του Ερωτήματος 2.1 είναι βέβαιο ότι θα επιστρέψει ακριβή αποτελέσματα χωρίς περαιτέρω φιλτράρισμα, αφού δεν μεσολαβεί καμία άλλη ακμή μεταξύ των *first(π) και last(π)*.

ΠΡΟΤΑΣΗ 1. Για κάθε διαδρομή π που απαρτίζεται από δύο ακμές $\pi=[e_{start}, e_{end}]$, το Ερώτημα 2.1 επιστρέφει μόνο τις τροχιές που ακολουθούν αυστηρά τη διαδρομή π.

ΑΠΟΔΕΙΞΗ. Για να ανήκει μία τροχιά στα αποτελέσματα του Ερωτήματος 2.1, θα πρέπει να ακολουθεί μία διαδρομή π_t μεταξύ της πρώτης και της τελευταίας ακμής της π, τέτοια ώστε

$$deltahash(\pi_t) = deltahash(\pi).$$

Έστω ότι $\pi_t=[e_{start}, e_2, \ldots, e_{n-1}, e_{end}]$. Τότε θα ισχύει

$$deltahash(\pi_t) = e_2.weight + \cdots + e_{n-1}.weight + e_{end}.weight.$$

Αφού

$$deltahash([e_{start}, e_{end}]) = e_{end}.weight,$$

ισχύει ότι

$$deltahash(\pi_t) > deltahash(\pi).$$

Αφού όμως $deltahash(\pi_t) > deltahash(\pi)$, η τροχιά t δεν μπορεί να ανήκει στα αποτελέσματα του SPQ(π).

-Επεξεργασία ερωτήματος

Χρησιμοποιώντας το συμπέρασμα της Πρότασης 1, μπορούν να αποφευχθούν λανθασμένα αποτελέσματα για κάθε διαδρομή π μήκους n ακμών, αν αποδειχθεί ότι οι υποψήφιες τροχιές ακολουθούν όλες τις υπο-διαδρομές στις οποίες διασπάται η π. Δεδομένου ότι κάθε διαδρομή π μήκους δύο ακμών χαρακτηρίζεται από το deltahash της και μια τροχιά μπορεί να ελεγχθεί αν την ακολουθεί με εκτέλεση ενός προσεγγιστικού ερωτήματος (Ερώτημα 2.1), τότε αν η π (μήκους n) διαχωριστεί σε n-1 τμήματα μήκους δύο ακμών και μια τροχιά αποδειχθεί ότι ακολουθεί κάθε ένα από αυτά τα τμήματα της π, τότε συνεπάγεται ότι η τροχιά ακολουθεί επακριβώς τη διαδρομή π.

Συνεπώς, για την εκτέλεση ενός SPQ ακριβούς προσέγγισης μπορεί να ακολουθηθεί μια διαδικασία φίλτρου-διύλισης (filter-refinement): Αρχικά, εκτελείται προσεγγιστικό ερώτημα με χρήση της αρχικής και τελικής ακμής της διαδρομής το οποίο επιστρέφει ένα σύνολο από υποψήφιες τροχιές. Κατόπιν, η αρχική διαδρομή μήκους n ακμών διασπάται σε n-1 υπο-διαδρομές μήκους δύο

ακμών, που οι υποψήφιες τροχιές θα ελεγχθούν αν τις ακολουθούν. Αν μια τροχιά δεν ακολουθεί κάποια από τις υπο-διαδρομές, απορρίπτεται από το τελικό αποτέλεσμα.

Παράδειγμα:

Για τη διαδρομή π=[e1, e2, e3, e4] εξάγεται ένα σύνολο υποψήφιων τροχιών με την εκτέλεση προσεγγιστικού ερωτήματος για την πλήρη διαδρομή. Στη συνέχεια, η διαδρομή π διασπάται στις υπο-διαδρομές π1=[e1, e2], π2=[e2, e3] και π3=[e3, e4]. Αν μία τροχιά t έχει τις τιμές h1, h2, h3, h4 στις ακμές e1, e2, e3, e4 αντίστοιχα, και h2-h1=deltahash(π1), τότε η τροχιά t ακολουθεί, σύμφωνα με την Πρόταση 1, την υπο-διαδρομή π1. Αν αυτό δεν ισχύει, η τροχιά t απορρίπτεται από το σύνολο υποψηφίων τροχιών. Η διαδικασία επαναλαμβάνεται, και για κάθε υπο-διαδρομή της π πρέπει να ισχύει h_(i+1)-h_i=deltahash(π_i), με κάθε τροχιά που δεν ακολουθεί την υπο-διαδρομή να απορρίπτεται από το σύνολο υποψηφίων.Ουσιαστικά ο αλγόριθμος λειτουργεί προσθετικά, απορρίπτοντας για κάθε υπο-διαδρομή π_i τις τροχιές που δεν διέρχονται από την ακμή *last(π_i).*

Οι τροχιές που απομένουν είναι αυτές οι οποίες πληρούν τα κριτήρια του Ερωτήματος 2.1 για όλη τη διαδρομή *π* και για κάθε υπο-διαδρομή της. με αυτό τον τρόπο, αποκλείονται οι τροχιές που δεν ακολουθούν επακριβώς ολόκληρη την διαδρομή *π*.

Το SQL ερώτημα που υλοποιεί τη δεύτερη προσέγγιση, αποτελεί μια τροποποίηση του πρώτου στην οποία ελέγχεται ο πίνακας visitedsegments για κάθε υπο-διαδρομή ξεχωριστά. Η προσέγγιση αυτή φυσικά έχει πολύ μεγαλύτερο κόστος από την πρώτη, ωστόσο είναι πιο αξιόπιστη ως προς τα αποτελέσματα που εξάγει.

```
select e_start.tid
from visitedsegments as e_start
join visitedsegments as e_end using(tid)
join visitedsegments as e_2 using(tid)
. . .
join visitedsegments as e_n-1 using(tid)
where e_start.eid=first(π)
and e_end.eid=last(π)
and e_2.eid=last(π_1)
. . .
and e_n-1.eid=last(π_n-1)
and e_end.hash=e_start.hash+deltahash(π)
and e_end.hash=e_start.hash+deltahash(π)
and e_2.hash=e_start.hash+deltahash(π_1)
and e_3.hash=e_2.hash+deltahash(π_2)
. . .
and e_end.hash=e_n-1.hash+deltahash(π_n-1)
```

Ερώτημα 2.2: Δεύτερη (ακριβής) προσέγγιση SPQ

Το ερώτημα διαδρομής (PQ - Path Query) της προσέγγισης PARINET [23], εκτελεί μεγάλο αριθμό αναζητήσεων, μιας και αναζητά για κάθε ακμή της διαδρομής δικτύου, όλες τις τροχιές που διέρχονται από αυτή, για να προχωρήσει μετά σε εκκαθάριση των τροχιών για εξαγωγή του τελικού αποτελέσματος. Το ερώτημα ακριβούς προσέγγισης που παρουσιάστηκε παραπάνω εκτελεί επίσης πολλές αναζητήσεις, αντισταθμίζοντας όμως τη μείωση της απόδοσης με την αύξηση της αξιοπιστίας των αποτελεσμάτων. Η τρίτη προσέγγιση SPQ που παρουσιάζεται παρακάτω περιορίζει σε μεγάλο βαθμό τις αναζητήσεις, αποφεύγοντας παράλληλα την απώλεια αξιοπιστίας των αποτελεσμάτων.

2.6 Βελτιωμένη ακριβής προσέγγιση SPQ [16]

2.6.1 Μειονεκτήματα της ακριβούς προσέγγισης SPQ

Η ακριβής προσέγγιση SPQ (Ερώτημα 2.2) μπορεί, από πλευράς απόδοσης, να παρομοιασθεί με πολλαπλές εκτελέσεις του SPQ Ερωτήματος 2.1 για μία διαδρομή π. Όπως αναφέρθηκε και στην προηγούμενη παράγραφο, μια διαδρομή π με n ακμές διαχωρίζεται σε $n-1$ υπο-διαδρομές που η καθεμία αποτελείται από δύο ακμές, για τις οποίες εκτελούνται οι απαραίτητοι έλεγχοι για διέλευση από τις υποψήφιες τροχιές και ισότητα της διαφοράς βαρών (*deltahash*) της τροχιάς με την διαφορά βαρών της διαδρομής π. Συνεπώς, ο χρόνος εκτέλεσης ενός ερωτήματος ακριβούς προσέγγισης πλησιάζει το χρόνο εκτέλεσης n προσεγγιστικών ερωτημάτων, με n να είναι το πλήθος των ακμών που ακολουθεί η διαδρομή π. Στην πραγματικότητα, όπως φαίνεται και στο Ερώτημα 2.2, εκτελείται ένα ερώτημα SQL εμπλουτισμένο με συνθήκες που αντιστοιχούν στα υπόλοιπα προσεγγιστικά ερωτήματα SPQ.

Η δημιουργία ενός τόσο πολύπλοκου ερωτήματος γίνεται διότι έως αυτό το σημείο θεωρούμε τις υπο-διαδρομές μήκους δύο ακμών τις μόνες αξιόπιστες υπο-διαδρομές για την εκτέλεση προσεγγιστικού SPQ, δηλαδή τις μόνες διαδρομές για τις οποίες δεν επιστρέφονται λανθασμένα στοιχεία από ένα προσεγγιστικό SPQ. Ωστόσο, η μορφολογία του δικτύου, οι περιορισμοί κίνησης που υπάρχουν σε αυτό και οι ιδιότητες των ακμών του επιτρέπουν τον εντοπισμό μοναδικών διαδρομών για τη μετάβαση από μία ακμή σε άλλη με μήκος μεγαλύτερο του 2. Αυτό σημαίνει ότι για την εκτέλεση ενός ακριβούς SPQ σε μία διαδρομή π, δεν απαιτείται σε κάθε περίπτωση ο κατακερματισμός της διαδρομής σε n υπο-διαδρομές των δύο ακμών, αλλά σε $1 \leq x \leq n$ υπο-διαδρομές των $2 \leq x \leq n$ ακμών. Σε τέτοια περίπτωση, δεν απαιτούνται $n-1$ διαδοχικά self-joins στον πίνακα visitedsegments για την εκτέλεση του Ερωτήματος 2.2, αλλά ένα self-join για κάθε ξεχωριστή υπο-διαδρομή που προκύπτει από την π, χωρίς να επηρεάζεται η ακρίβεια του αποτελέσματος.

2.6.2 Μοναδικές υπο-διαδρομές

Το Ερώτημα 2.1 επιστρέφει εσφαλμένα αποτελέσματα μόνο όταν υπάρχει μία διαδρομή π', διαφορετική από την π, για την οποία ισχύει $first(\pi) = first(\pi')$, $last(\pi) = last(\pi')$ και $deltahash(\pi) = deltahash(\pi')$. Αν δεν υπάρχει μια τέτοια διαδρομή, η διαδρομή π *είναι μοναδική*. Αν η π είναι μονα-

δική, το Ερώτημα 2.1 εγγυάται την ύπαρξη τροχιών στα αποτελέσματα που όντως θα ακολουθούν ακριβώς τη διαδρομή π. Αυτό το συμπέρασμα είναι πολύ σημαντικό, διότι το Ερώτημα 2.1 εξάγει αποτελέσματα για τη διαδρομή π αναζητώντας δεδομένα μόνο για την πρώτη και την τελευταία της ακμή, ανεξαρτήτως του πλήθους των ακμών της.

Η διαδρομή π είναι *πάντα μοναδική* όταν ταυτίζεται με τη συντομότερη διαδρομή μεταξύ της πρώτης και της τελευταίας ακμής της π, και υπάρχει μόνο μία συντομότερη διαδρομή μεταξύ της πρώτης και της τελευταίας ακμής. Σε αυτήν την περίπτωση, κάθε άλλη πιθανή διαδρομή θα είναι μεγαλύτερη από την π, συνεπώς, αφού ως *hash* χρησιμοποιείται το μήκος της ακμής, θα ισχύει

$$deltahash(\pi) < deltahash(\pi').$$

Όταν μια διαδρομή μεταξύ δύο ακμών της π δεν είναι μοναδική, ή υπάρχει διαδρομή συντομότερη από αυτή που ακολουθεί η π για τη σύνδεση των δύο ακμών, τότε η διαδρομή π διαιρείται σε μικρότερα τμήματα, με στόχος κάθε ένα από αυτά τα τμήματα να είναι μοναδικό. Τυπικότερα, ισχύει

$$\pi = [\pi_1, \pi_2, \dots, \pi_k]$$
$$\text{και } first(\pi_1) = first(\pi), last(\pi_k) = last(\pi), last(\pi_i) = first(\pi_{i+1}).$$

Στην ιδανική περίπτωση θα ισχύει *k=1*, δηλαδή ολόκληρη η διαδρομή π θα είναι μοναδική, οπότε το αποτέλεσμα από το Ερώτημα 2.1 θα είναι ακριβές. Στην χειρότερη περίπτωση, θα δημιουργηθούν *n-1* διαδρομές μήκους δύο ακμών (οι οποίες είναι πάντα μοναδικές, λόγω της Πρότασης 1), όπως στην περίπτωση του Ερωτήματος 2.2.

Συμπεραίνουμε ότι για την απόδειξη ότι μία τροχιά ακολουθεί επακριβώς τη διαδρομή π θα πρέπει να αποδειχθεί ότι η τροχιά ακολουθεί όλες τις μοναδικές υπο-διαδρομές της π, αποκλείοντας κατά τη διάρκεια του ελέγχου κάθε υπο-διαδρομής τα λανθασμένα αποτελέσματα από τις υποψήφιες τροχιές (ακολουθείται ίδια διαδικασία με την ακριβή επίλυση της προηγούμενης παραγράφου).

2.6.3 Αλγόριθμος Unique SubPaths

Στην παρούσα παράγραφο παρουσιάζεται ο αλγόριθμος που προτείνεται στο [16] (Αλγόριθμος 2.1) σχεδιασμένος κατάλληλα για τη διάσπαση μιας διαδρομής π σε ένα σύνολο από υπο-διαδρομές, κάθε μία εκ των οποίων είναι μοναδική.

Ο αλγόριθμος UniqueSubPaths μετασχηματίζει μια διαδρομή π σε ένα σύνολο από μοναδικές υπο-διαδρομές. Σαν είσοδος στον αλγόριθμο δίδεται η διαδρομή π, ενώ το αποτέλεσμα που εξάγει είναι οι μοναδικές υπο-διαδρομές της π.

```
1: function UniqueSubPaths(π)
2:     if size(π) < 3 then
3:         return π
4:     end if
5:     vi ← 2
6:     for i = 3 → size(π) do
7:         sppath ← Shortest(π[1], π[i])
8:         if sppath ≠ SubList(π, 2, i) then
9:             break
10:        end if
11:        vi ← i
12:    end for
13:    if vi = size(π) then
14:        return π
15:    else
16:        first ← SubList(π, 1, vi + 1)
17:        π′ ← SubList(π, vi, size(π) + 1)
18:        subpaths ← UniqueSubPaths(π′)
19:        return {first} ∪ subpaths
20:    end if
21: end function
```

Αλγόριθμος 2.1: Αλγόριθμος UniqueSubPaths

-Επεξήγηση αλγορίθμου

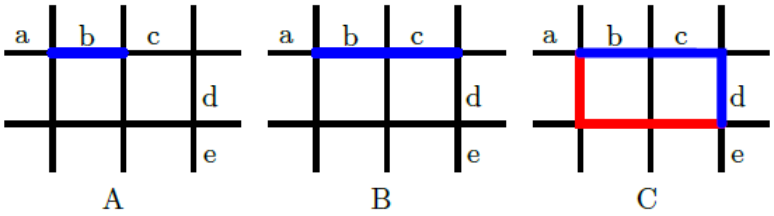
Στον αλγόριθμο UniqueSubPaths δίνεται σαν είσοδος μια διαδρομή π, η οποία ελέγχεται αναδρομικά για την πιθανή ύπαρξη εντός της μη μοναδικών υπο-διαδρομών. Ο αλγόριθμος έχει σκοπό την απαλοιφή των μη μοναδικών υπο-διαδρομών και την επιστροφή ενός συνόλου μοναδικών υπο-διαδρομών, οι οποίες αν συνδεθούν διαδοχικά ισοδυναμούν με την αρχική διαδρομή π. Για την τμηματοποίηση της διαδρομής χρησιμοποιείται η συνάρτηση SubList(π, i, j), η οποία επιστρέφει τη λίστα ακμών που απαρτίζουν την π από τη θέση i μέχρι τη θέση j, χωρίς να περιλαμβάνει την ακμή που αντιστοιχεί στη θέση j. Αρχικά ελέγχεται το μήκος της διαδρομής (γραμμές 2-4), το οποίο αν είναι μικρότερο των τριών ακμών η διαδρομή θεωρείται μοναδική και επιστρέφεται εξολοκλήρου σαν αποτέλεσμα από τον αλγόριθμο, ο οποίος τερματίζεται.

Σε διαφορετική περίπτωση, πραγματοποιείται επαναληπτικά διάσχιση της διαδρομής π (γραμμές 6-12), ξεκινώντας από την υπο-διαδρομή των τριών πρώτων ακμών SubList(π, 1, 3) και προσθέτοντας μία ακμή της διαδρομής σε κάθε επανάληψη . Η νέα υπο-διαδρομή ελέγχεται ως προς τη μοναδικότητά της μέσω αναδρομικής κλήσης του αλγορίθμου UniqueSubPaths με παράμετρο την ίδια την υπο-διαδρομή και χρήση της συνάρτησης Shortest(estart, eend), η οποία επιστρέφει τη συντομότερη διαδρομή μεταξύ του τελικού κόμβου της estart και του αρχικού κόμβου της eend.

Με χρήση της Shortest ελέγχεται αν η υπο-διαδρομή είναι και η συντομότερη δυνατή βάσει περιορισμών δικτύου για τις επιλεγμένες ακμές. Στην περίπτωση που είναι η συντομότερη, θεωρείται μοναδική διαδρομή και ο αλγόριθμος ενημερώνει κατάλληλα τη μεταβλητή vi (που αποτελεί το δείκτη της τελικής ακμής) για την επόμενη επανάληψη. Σε διαφορετική περίπτωση, η τελική ακμή (που προστέθηκε τελευταία στην υπο-διαδρομή) αφαιρείται από την υπο-διαδρομή και η προκύπτουσα υπο-διαδρομή προστίθεται στο σύνολο μοναδικών υπο-διαδρομών της μεταβλητής first (γραμμή 16). Το υπόλοιπο κομμάτι της διαδρομής θα δοθεί σαν όρισμα σε αναδρομική εκτέλεση του

UniqueSubPaths, συνεχίζοντας τη διαδικασία μέχρι το πέρας της διαδρομής π. Το αποτέλεσμα του αλγορίθμου είναι το σύνολο μοναδικών υπο-διαδρομών της π που έχει εκχωρηθεί στη μεταβλητή first.

-Παράδειγμα



Εικόνα 2.6: UniqueSubPaths για την τροχιά abcde

Στο σημείο αυτό παρουσιάζεται ένας έλεγχος μοναδικότητας διαδρομής με τον αλγόριθμο UniqueSubPaths. Η διαδρομή π=[a, b, c, d, e] του μικρού δικτύου που απεικονίζεται στην Εικόνα 2.6 αποτελείται από πέντε ακμές. Στο ξεκίνημα του αλγορίθμου, εξετάζεται η διαδρομή π1=[a, b, c], διότι κάθε διαδρομή με μήκος μικρότερο των τριών είναι και μοναδική, σύμφωνα με την Πρόταση 1. Η διαδρομή π1 με μήκος τριών ακμών είναι μοναδική, αφού πλην της ακμής b δεν υπάρχει άλλη συντομότερη διαδρομή εντός του δικτύου που να συνδέει τις ακμές a και c. Ακολούθως, προστίθεται μία ακμή στην εξεταζόμενη διαδρομή για την επόμενη επανάληψη (Εικόνα 2.6 μέρος Β), δημιουργώντας τη διαδρομή π1=[a, b, c, d] η οποία επίσης χαρακτηρίζεται ως μοναδική, αφού η διαδρομή [b, c] είναι η συντομότερη και μοναδική διαδρομή του δικτύου που συνδέει τις ακμές a και d. Προσθέτοντας άλλη μία ακμή της π για την επόμενη επανάληψη, έχουμε την διαδρομή π1=[a, b, c, d, e] (Εικόνα 2.6 μέρος C). Η διαδρομή αυτή δεν είναι μοναδική, μιας και η υποδιαδρομή [b, c, d] δεν είναι η μοναδική και συντομότερη διαδρομή μεταξύ των ακμών a και e (μια εναλλακτική ίσου μήκους σημειώνεται με κόκκινο χρώμα στο μέρος C της Εικόνας 2.6).

Συμπερασματικά, η μακρύτερη μοναδική υπο-διαδρομή της διαδρομής π=[a, b, c, d, e] είναι η π1=[a, b, c, d]. Στη συνέχεια πραγματοποιείται αναδρομική κλήση του UniqueSubPaths για το υπόλοιπο της διαδρομής π, δηλαδή την υπο-διαδρομή π2=[d, e] η οποία και χαρακτηρίζεται μοναδική από την Πρόταση 1 λόγω του μήκους της (το οποίο είναι ίσο με δύο) χωρίς να απαιτείται επιπλέον επεξεργασία. Άρα οι υπο-διαδρομές που επιστρέφονται από τον αλγόριθμο για την διαδρομή π=[a, b, c, d, e] είναι οι π1=[a, b, c, d] και π2=[d, e].

2.7 «Πρακτική» ακριβής προσέγγιση SPQ [16]

Η αξιοπιστία του ερωτήματος SPQ μέχρι τώρα επηρεάζεται σημαντικά από το βάρος που έχει επιλεχθεί για την κωδικοποίηση των ακμών. Σε περιπτώσεις όπου πολλές ακμές του δικτύου τυχαίνει να έχουν το ίδιο μήκος, υπάρχει σημαντική πιθανότητα για ύπαρξη λανθασμένων αποτελεσμάτων. Ωστόσο, αυτό είναι πιο πιθανόν να συμβεί στην περίπτωση όπου υπάρχουν πολλές ακμές του δικτύου με παρόμοιο μήκος σε κοντινή απόσταση μεταξύ τους, αυξάνοντας τις πιθανότητες για ίσο άθροισμα μηκών ακμών που προέρχεται από διαφορετική ακολουθία ακμών. Ακόμη, το αποτέλεσμα μπορεί να επηρεαστεί αν το μήκος των ακμών έχει στρογγυλοποιηθεί σημαντικά, επηρεάζοντας έτσι την ακρίβεια των αθροισμάτων. Συνεπώς, τα αποτελέσματα του SPQ μπορούν να παρουσιάσουν λάθη αν υπάρχουν περισσότερες από μία διαδρομές με συγκεκριμένο μήκος.

Για την αντιμετώπιση αυτού του προβλήματος προτείνεται ένας διαφορετικός τρόπος κωδικοποίησης των ακμών, τέτοιος ώστε η πιθανότητα για ψευδή αποτελέσματα να μειώνεται στο ελάχιστο. Για το σκοπό αυτό, θα πρέπει το βάρος κάθε ακμής του δικτύου να είναι μοναδικό σε όλο το δίκτυο.

Η νέα μέθοδος κωδικοποίησης βασίζεται στο θεμελιώδες θεώρημα της αριθμητικής [15], σύμφωνα με το οποίο το γινόμενο ενός μοναδικού συνδυασμού πρώτων παραγόντων είναι επίσης μοναδικό. Η βασική ιδέα της προσέγγισης είναι η ανάθεση ενός μοναδικού πρώτου αριθμού σε κάθε ακμή του δικτύου, ο οποίος χρησιμοποιείται μελλοντικά για τον προσδιορισμό της ακμής εντός του συνόλου ακμών του δικτύου. Για τον ακριβή προσδιορισμό της διέλευσης της διαδρομής από συγκεκριμένο σύνολο ακμών πρέπει να υπολογιστεί το "βάρος" της διαδρομής, που στις προηγούμενες προσεγγίσεις αντιστοιχούσε στο deltahash της διαδρομής.Στην παρούσα προσέγγιση, πρέπει να υπολογιστεί το γινόμενο των πρώτων αριθμών που έχουν ανατεθεί στις ακμές από τις οποίες διέρχεται η διαδρομή. Το γινόμενο αυτό θα είναι μοναδικό για κάθε διαδρομή, υπό την προϋπόθεση ότι η διαδρομή είναι μη κυκλική, διότι σε τέτοια περίπτωση το γινόμενο πρώτων παραγόντων δεν μπορεί να εγγυηθεί την ακεραιότητα των αποτελεσμάτων, αφού δεν μπορεί να προσδιορίσει την σειρά με την οποία η τροχιά επισκέφθηκε τις ακμές μιας διαδρομής. Στην περίπτωση κυκλικών διαδρομών, η διαδρομή πρέπει να μετασχηματιστεί σε σύνολο μη κυκλικών υπο-διαδρομών, οι οποίες θα τεθούν σε επεξεργασία ξεχωριστά.

Για την ευκολότερη διαχείριση των μεγάλων σε μέγεθος αριθμών που αποτελούν τα γινόμενα των επίσης μεγάλων σε μέγεθος πρώτων που χρησιμοποιούνται για την κωδικοποίηση των ακμών του δικτύου, υπολογίζεται ο αντίστοιχος λογάριθμος για κάθε πρώτο, μειώνοντας έτσι το μέγεθος του αριθμού και διευκολύνοντας τον υπολογισμό του deltahash, λόγω της ιδιότητας των λογαρίθμων $\log(xy) = \log(x) + \log(y)$. Με τη μέθοδο αυτή, δεν απαιτείται ο υπολογισμός γινομένου αλλά αθροίσματος για τον υπολογισμό του deltahash της διαδρομής, όπως ακριβώς γίνεται και με τις μεθόδους που έχουν ήδη περιγραφεί σε προηγούμενες παραγράφους.

-Πιθανότητα ύπαρξης λαθών

Η μοναδικότητα των αναγνωριστικών της κωδικοποίησης με πρώτους αριθμούς δεν επηρεάζεται άμεσα από την αντικατάσταση των πρώτων αριθμών από τους λογαρίθμους τους, διότι οι λογάριθμοι που προκύπτουν είναι επίσης μοναδικοί και οδηγούν σε μοναδικά αθροίσματα κατά τον υπολογισμό του deltahash τροχιάς ή διαδρομής. Ωστόσο, κατά την αποθήκευση στη βάση δεδομένων το πεδίο που θα φιλοξενήσει τους λογαρίθμους ενδέχεται να παίξει σημαντικό ρόλο στη διατήρηση της μοναδικότητας των τιμών. Ένα πεδίο με ικανότητα αποθήκευσης λίγων δεκαδικών ψηφίων μπορεί να οδηγήσει σε στρογγυλοποιήσεις και κατά συνέπεια στην ύπαρξη ίσων βαρών σε διαφορετικές ακμές. Η πιθανότητα όμως για την ύπαρξη μη μοναδικών βαρών είναι αντιστρόφως ανάλογη του πλήθους δεκαδικών ψηφίων που χρησιμοποιούνται για την αποθήκευση του βάρους ακμής.

Για παράδειγμα [16], αν θεωρηθεί ένα δίκτυο 50 εκατομμυρίων ακμών όπου οι τροχιές ακολουθούν έως ένα εκατομμύριο ακμές η καθεμιά, το μέγιστο βάρος ακμής θα είναι μικρότερο από 10^9, αφού οι 50 εκατομμύρια μικρότεροι πρώτοι φυσικοί αριθμοί είναι μικρότεροι από 10^9. Αν μια τροχιά διήλθε από αυτή την ακμή 10^6 φορές, τότε η τιμή βάρους για την τροχιά θα είναι το πολύ $\log(10^9) \cdot 10^6 = 9 \cdot 10^6$. Αν για την αποθήκευση αυτής της τιμής διαθέτουμε έναν τύπο δεδομένων 64 bits, απαιτούνται $\lceil \log_2(9 \cdot 10^6) \rceil = 24$ bits για την αναπαράσταση του ακέραιου μέρους, αφήνοντας 40 bits για την αποθήκευση του δεκαδικού μέρους, που μεταφράζονται σε $\lfloor \log_{10}(2^{40}) \rfloor = 12$ bits δεκαδικών ψηφίων. Άρα λόγω της στρογγυλοποίησης μπορούν να παρατηρηθούν ψευδή αποτελέσματα για ακρίβειες μεγαλύτερες του 10^{-12}. Άρα η πιθανότητα δύο διαφορετικές διαδρομές να παρουσιάσουν ίσα deltahash είναι 10^{-12}. Αν ωστόσο, χρησιμοποιηθεί τύπος δεδομένων που υποστηρίζει περισσότερα δεδκαδικά ψηφία, η πιθανότητα αυτή μπορεί να μειωθεί ακόμη περισσότερο.

Για την παρουσία λανθασμένου αποτελέσματος, πρέπει να ισχύουν ταυτόχρονα οι εξής υποθέσεις:

- $first(\pi_1) = first(\pi_2)$
- $last(\pi_1) = last(\pi_2)$
- $deltahash(\pi_1) = deltahash(\pi_2)$
- π_1, π_2 ακολουθούνται από τουλάχιστον μία τροχιά

Διαισθητικά, η πιθανότητα για εμφάνιση λανθασμένου αποτελέσματο αυξάνεται όσο αυξάνεται ο αριθμός πιθανών εναλλακτικών διαδρομών π μεταξύ $first(\pi_1)$ και $last(\pi_1)$. Για τον υπολογισμό της πιθανότητας ύπαρξης λανθασμένου αποτελέσματος υποθέτουμε ότι όλα τα δεκαδικά ψηφία της τιμής βάρους είναι τυχαία. Με αυτή την υπόθεση σε ισχύ, η πιθανότητα μπορεί να υπολογιστεί με τον τύπο

$$p_n = 1 - \left(\frac{B-1}{B}\right)^n$$

όπου p_n η πιθανότητα για λανθασμένο αποτέλεσμα, n ο αριθμός των ξεχωριστών διαδρομών μεταξύ $first(\pi_1)$ και $last(\pi_1)$ και B το πλήθος των πιθανών τιμών για το δεκαδικό μέρος, π.χ. 10^{12}. Η τιμή του n είναι συνήθως μικρή, αρκετά κάτω από 10^3, διότι τα οχήματα τείνουν να ακολουθούν παρόμοιες διαδρομές εντός του δικτύου της πόλης. Για αυτούς τους αριθμούς, η

πιθανότητα για λανθασμένα αποτελέσματα χρησιμοποιώντας την «πρακτική» προσέγγιση περιορίζεται περίπου στο 10^{-9}.

2.8 Επεξεργασία ερωτημάτων SPQ

Στην παρούσα παράγραφο παρουσιάζεται ο αλγόριθμος ProcessSPQ [16], ένας αναδρομικής φύσης αλγόριθμος ο οποίος χρησιμοποιείται για την επεξεργασία, εκτέλεση και αξιολόγηση των ερωτημάτων SPQ.

```
1: function ProcessSPQ(π, exact, (start, end))
2:     C ← LoadTrajectories(first(π), last(π),
                            deltahash(π), (start, end))
3:     if exact or π has cycles then
4:         Subpaths ← UniqueSubPaths(π)
5:         for all π_i ∈ Subpaths do
6:             C' ← ProcessSPQ(π_i, false, (start, end))
7:             if i = 1 then
8:                 C ← filter_init(C, C')
9:             else
10:                C ← filter(C, C')
11:            end if
12:        end for
13:    end if
14:    return C
15: end function
```

Αλγόριθμος 2.2: ProcessSPQ

Ο αλγόριθμος δέχεται σαν ορίσματα μια διαδρομή *π*, μία δυαδική τιμή *exact* (αληθής/ψευδής) και ένα χρονικό διάστημα (*start, end*). Η δυαδική τιμή *exact* καθορίζει αν τα ψευδή αποτελέσματα είναι αποδεκτά στο σύνολο αποτελεσμάτων.

Η συνάρτηση LoadTrajectories που καλείται στη δεύτερη γραμμή του αλγορίθμου εκτελεί ένα ερώτημα παρόμοιο με το 2.1, ανακτώντας παράλληλα τις τιμές βάρους εισόδου και εξόδου στη διαδρομή π για κάθε τροχιά t. Στην LoadTrajectories δίδεται ακόμη σαν είσοδος ένα χρονικό διάστημα, το οποίο και θα αποτελέσει το χρονικό κριτήριο για το ερώτημα ακριβούς τροχιάς που θα εκτελεστεί, στα πρότυπα του Ερωτήματος 2.3.

```
select e_start.tid, e_start.hash, e_end.hash
from visitedsegments as e_start
join visitedsegments as e_end using(tid)
where e_start.eid=first(π)
and e_end.eid=last(π)
and e_end.hash-e_start.hash=deltahash(π)
and e_start.ts_enter>=start
and e_end.ts_leave<=end
```

Ερώτημα 2.3: Ερώτημα SPQ συνάρτησης LoadTrajectories

Για την απαλοιφή ψευδών αποτελεσμάτων, σε περίπτωση κυκλικών διαδρομών ή όταν ζητηθεί λεπτομερής επεξεργασία από το χρήστη (παράμετρος exact αληθής), η διαδρομή μετασχηματίζεται με τη χρήση του αλγορίθμου

UniqueSubPaths σε ένα σύνολο υπο-διαδρομών για τη συνέχιση της επεξεργασίας (γραμμή 4). Στην περίπτωση αυτή μειώνονται κατά πολύ οι πιθανότητες για την εξαγωγή λανθασμένων αποτελεσμάτων, ειδικά στις περιπτώσεις κυκλικών διαδρομών, όπου η σειρά διέλευσης των τροχιών από τις ακμές του δικτύου δεν μπορεί να προσδιοριστεί επακριβώς.

Στις γραμμές 7-11 του αλγορίθμου οι συναρτήσεις filter και filter_init αναλαμβάνουν να προσδιορίσουν ποιες τροχιές ακολουθούν την διαδρομή π ακολουθώντας επακριβώς κάθε υπο-τροχιά της. Συγκεκριμένα, η συνάρτηση filter_init καλείται μόνο για την πρώτη μοναδική υπο-διαδρομή της π και επιλέγει από το σύνολο αποτελεσμάτων C' (υποψήφιες τροχιές για την πρώτη υπο-διαδρομή της π) τις τροχιές που βρίσκονται στο σύνολο αποτελεσμάτων C (περιλαμβάνει τις υποψήφιες τροχιές ολόκληρης της διαδρομής π, αποτέλεσμα της LoadTrajectories) για τις οποίες ισχύει

$$C\big(hash_{start}(tid)\big) = C'\big(hash_{start}(tid)\big).$$

Η συνάρτηση filter εκτελείται για τις υπόλοιπες υπο-διαδρομές της π. Η λειτουργία της είναι να επιλέγει από το C' τις τροχιές που ανήκουν και στο σύνολο C για τις οποίες ισχύει $C\big(hash_{end}(tid)\big) = C'\big(hash_{start}(tid)\big)$.

Όταν ολοκληρωθεί ο έλεγχος για κάθε υπο-διαδρομή, οι τροχιές που ακολουθούν επακριβώς τη διαδρομή π, χωρίς ψευδή αποτελέσματα, έχουν εκχωρηθεί στη μεταβλητή C, η οποία κι επιστρέφεται ως αποτέλεσμα από τον αλγόριθμο. Ο αλγόριθμος ProcessSPQ είναι αναδρομικός, ωστόσο δεν υπάρχει ανάγκη για την αναδρομική κλήση του αλγορίθμου σε βάθος μεγαλύτερο του 2, διότι ο αλγόριθμος καλεί τον εαυτό του μόνο όταν υπάρχουν κυκλικές διαδρομές ή πρέπει να αναζητηθούν μοναδικές υπο-διαδρομές. Ωστόσο, οι δύο αυτές περιπτώσεις εξαλείφονται όταν η παράμετρος exact τεθεί ως αληθής και κληθεί ο αλγόριθμος UniqueSubPaths, συνεπώς στην αναδρομική κλήση του αλγορίθμου η exact τίθεται ως ψευδής, γεγονός που αποκλείει την περαιτέρω αναδρομική κλήση του αλγορίθμου.

3. Τεχνολογίες και δεδομένα

Για την υλοποίηση της εφαρμογής που περιγράφηκε από θεωρητικής σκοπιάς στο κεφάλαιο 3 του παρόντος τόμου, χρησιμοποιήθηκε ένα σύνολο τεχνολογιών και υποδομών δεδομένων. Στο παρόν υποκεφάλαιο παρουσιάζονται συνοπτικά οι τεχνολογίες που χρησιμοποιήθηκαν για την υλοποίηση της εφαρμογής και την παρουσίαση των αποτελεσμάτων της.

3.1 Τεχνολογίες

3. 1.1 Σύστημα Διαχείρισης Βάσεων Δεδομένων (DBMS)

Το ΣΔΒΔ που επιλέχθηκε για την υποστήριξη των δεδομένων της εφαρμογής είναι η πλατφόρμα PostgreSQL, η οποία αποτελεί μια από τις ευρέως χρησιμοποιούμενες πλατφόρμες διαχείρισης δεδομένων ανοιχτού κώδικα η οποία αναπτύχθηκε και εξελίσσεται για περισσότερο από δεκαπέντε χρόνια, δημιουργώντας ιδιαίτερα θετική φήμη για την αξιοπιστία και την ακεραιότητα στη διαχείριση των δεδομένων.

-Γενικά χαρακτηριστικά

Η PostgreSQL (ή απλά Postgres), αποτελεί ένα αντικειμενοστρεφές DBMS (ORDBMS), συμβατό με τα περισσότερα κατοχυρωμένα πρότυπα διαχείρισης δεδομένων και εύκολα επεκτάσιμο μέσω διαφόρων πρόσθετων που έχουν αναπτυχθεί γι αυτό, πολλά εκ των οποίων είναι επίσης ανοιχτού κώδικα και διαθέσιμα δωρεάν μέσω του Διαδικτύου.

Η Postgres μπορεί να υποστηρίξει από απλές βάσεις δεδομένων με πρόσβαση μόνο από τον τοπικό υπολογιτή έως μεγάλες βάσεις με ταυτόχρονη πρόσβαση από πολλούς χρήστες μέσω του Διαδικτύου, εφαρμόζοντας παράλληλα τεχνικές διασφάλισης της ακεραιότητας των δεδομένων της κατά τη χρήση. Η Postgres είναι γραμμένη σε γλώσσα προγραμματισμού C και είναι συμβατή με πολλά ευρέως χρησιμοποιούμενα λειτουργικά συστήματα όπως το UNIX, τα Windows, το Solaris και το Apple OS.

Η λειτουργία της είναι πλήρως συμβατή με το πρότυπο ACID [14], ενώ υποστηρίζονται δομές ξένων κλειδιών (foreign keys), συνδέσεις (joins), προσωρινές απεικονίσεις (views), triggers και αποθηκευμένες διαδικασίες (stored procedures).

Στο σύστημα περιλαμβάνονται επίσης εξειδικευμένες λειτουργίες όπως μηχανισμός σχεδιασμού και βελτιστοποίησης ερωτημάτων (query planner and optimizer) έλεγχος συγχρονισμού (MVCC) [3], επαναφορά συστήματος σε συγκεκριμένη χρονική στιγμή, εμφωλευμένες συναλλαγές (savepoints), δυνατότητα αποθήκευσης αντιγράφου της βάσης δεδομένων ακόμα και όταν το σύστημα είναι σε λειτουργία. Για τον εξειδικευμένο χρήστη παρέχεται επίσης λεπτομερής τεκμηρίωση και προγραμματιστικές διεπαφές σε ποικιλία γλωσσών προγραμματισμού, μεταξύ άλλων C/C++, Java, .NET, Perl, Python, Ruby, Tcl, ODBC.

Ο πηγαίος κώδικας της Postgres (σε γλώσσα C) είναι διαθέσιμος στο ευρύ κοινό, υπό την άδεια PostgreSQL. Η άδεια αυτή δίνει την ελευθερία σε όποιον επιθυμεί να κάνει οποιεσδήποτε αλλαγές στο σύστημα και να το χρησιμοποιήσει

για προσωπική ή εμπορική χρήση. Συνεπώς, η Postgres δεν αποτελεί μόνο ένα ισχυρό ΣΔΒΔ, αλλά και μια πλατφόρμα όπου μπορούν να αναπτυχθούν εφαρμογές που απαιτούν ένα αξιόπιστο ΣΔΒΔ για τη σωστή λειτουργία τους.

-Εξειδικευμένα χαρακτηριστικά

Στην πλατφόρμα Postgres περιλαμβάνονται επίσης εξειδικευμένα χαρακτηριστικά που δίνουν ακόμη μεγαλύτερες δυνατότητες και ευελιξία στους χρήστες. Ορισμένα από αυτά είναι:

- Κληρονομικότητα σε επίπεδο πίνακα (table inheritance), η οποία επιτρέπει τη δημιουργία πινάκων βασισμένων σε πρότυπα ήδη υπαρχόντων πινάκων, χρησιμοποιώντας ένα είδος αντικειμενοστρεφούς προσέγγισης. Με αυτόν τον τρόπο η Postgres μπορεί να υποστηρίξει απλή και πολλαπλή κληρονομικότητα μεταξύ πινάκων.
- Σύστημα ρόλων (rules / query rewrite system). Δίνει τη δυνατότητα στο διαχειριστή της Βάσης Δεδομένων να δημιουργήσει χαρακτηριστικούς κανόνες για συγκεκριμένους πίνακες ή απεικονίσεις καθώς και να τους τροποποιήσει δυναμικά την στιγμή που επεξεργάζονται.
- Σύστημα συμβάντων (events system). Αποτελεί ένα σύστημα διασύνδεσης διαδικασιών μέσω του οποίου μπορούν να μεταβιβαστούν μηνύματα μεταξύ χρηστών του συστήματος, με χρήση των εντολών LISTEN και NOTIFY, επιτρέποντας τόσο την επικοινωνία μεταξύ των χρηστών όσο και τον επιτυχή συντονισμό των συμβάντων εντός της Βάσης Δεδομένων, ενώ οι χρήστες έχουν τη δυνατότητα να παρακολουθήσουν λειτουργίες που συμβαίνουν στους πίνακες όπως εισαγωγή, διαγραφή ή ανανέωση δεδομένων τη στιγμή που συμβαίνουν.

-Ευρετήρια

Η πλατφόρμα της Postgres παρέχει πολλών ειδών ευρετήρια, μεταξύ άλλων B-trees, πίνακες κατακερματισμού (Hash tables), αντεστραμμένα ευρετήρια (GIN) και γενικευμένα δένδρα αναζήτησης (GiST), ενώ δίδεται η δυνατότητα στον εξειδικευμένο χρήστη για τη δημιουργία και χρήση δικών του μεθόδων ευρετηριοδότησης. Μεταξύ άλλων, τα ευρετήρια του συστήματος Postgres υποστηρίζουν τα εξής χαρακτηριστικά:

- Ευρετήρια εκφράσεων (expression indexes): Δημιουργούνται για την ευρετηριοδότηση των αποτελεσμάτων ενός ερωτήματος ή μιας συνάρτησης.
- Τμηματικά ευρετήρια (partial indexes): Δημιουργούνται για την ευρετηριοδότηση ενός τμήματος των δεδομένων, με την προσθήκη μια συνθήκης WHERE στην εντολή δημιουργίας του ερωτήματος, επιτρέποντας την δημιουργία ενός μικρότερου και συνεπώς ταχύτερου ευρετηρίου.
- Ταυτόχρονη χρήση από τη μηχανή ερωτημάτων πολλών διαφορετικών ευρετηρίων για την ταχύτερη εκτέλεση πολύπλοκων ερωτημάτων.
- Από την έκδοση 9.1 και μεταγενέστερα, υποστηρίζεται η αναζήτηση k κοντινότερων γειτόνων (KNN) [1] για την αναζήτηση τιμών πλησιέστερων στο κριτήριο αναζήτησης, λειτουργία ιδιαίτερα χρήσιμη σε εφαρμογές αναζήτησης κειμένου ή εφαρμογές γεωχωρικής φύσεως.

- Από την έκδοση 9.2 και μεταγενέστερα, υποστηρίζεται η λειτουργία της αναζήτησης αποτελεσμάτων μόνο από το ευρετήριο, χωρίς πρόσβαση για ανάγνωση του πίνακα δεδομένων.

Ειδικότερα, τα GiST ευρετήρια αποτελούν μια εξειδικευμένη μορφή ευρετηρίου, η οποία συδυάζει δομές και αλγορίθμους των B-tree, B+-tree, R-tree, ranked B+-trees και άλλων ευρετηρίων. Συνδυαστικά, παρέχει κατάλληλη διεπαφή χρήστη για τον ορισμό νέων τύπων δεδομένων και μεθόδων για την αναζήτησή τους μέσω του ευρετηρίου, επιτρέποντας έτσι στον χρήστη να ορίσει επακριβώς τι αποθηκεύει, με ποιον τρόπο το αποθηκεύει και με ποιον τρόπο θέλει να κάνει αναζήτηση σε αυτό, υπερσκελίζοντας τα όρια των υπαρχουσών δομών ευρετηριοδότησης. Τα ευρετήρια GiST χρησιμοποιούνται πολύ συχνά σε πρόσθετες δομές διαχείρισης δεδομένων που έχουν αναπτυχθεί για την πλατφόρμα της Postgres, όπως το PostGIS και το OpenFTS [31].

-Τύποι δεδομένων

Υποστηρίζονται οι περισσότεροι τύποι δεδομένων του προτύπου ANSI-SQL: 2008, συμπεριλαμβανομένων των integer, numeric, boolean, varchar, date, interval, timestamp, ενώ σε βάση δεδομένων Postres μπορούν να αποθηκευτούν και δυαδικά αρχεία διαφόρων τύπων, όπως εικόνες, ήχοι ή video.

-Διαδικασιακές γλώσσες

Η πλατφόρμα Postgres υποστηρίζει την εκτέλεση συναρτήσεων γραμμένων σε πληθώρα γλωσσών προγραμματισμού, μερικές εκ των οποίων είναι οι Java, Perl, Python, Ruby, Tcl, C/C++, ενώ υποστηρίζεται και η PL/pgSQL, διαδικασιακή έκδοση της γλώσσας SQL, η οποία είναι παρόμοια με την PL/SQL της Oracle. Πέραν των συναρτήσεων που εγκαθίστανται μαζί με το ΣΔΒΔ, οι οποίες περιλαμβάνουν μεταξύ άλλων μαθηματικές, κρυπτογραφικές και συναρτήσεις διαχείρισης συμβολοσειρών, οι χρήστες μπορούν να δημιουργήσουν τους δικούς τους τύπους δεδομένων, συναρτήσεις και τελεστές που θα χρησιμοποιηθούν για την επεξεργασία των νέων τύπων δεδομένων. Η ιδιότητα αυτή έχει συμβάλλει στην ύπαρξη μεγάλης ποικιλίας τύπων δεδομένων στις πρόσφατες εκδόσεις της Postgres, που περιλαμβάνει μεταξύ άλλων τύπους γεωχωρικών δεδομένων, ακόμη και τύπους διεθνούς αρίθμησης βιβλίων (ISBN/ISSN), οι οποίοι μπορούν προαιρετικά να εγκατασταθούν στο σύστημα.

Οι διαδικασιακές γλώσσες προγραμματισμού που υποστηρίζονται από την Postgres μπορούν να χρησιμοποιηθούν για την επέκταση της λειτουργικότητας της Βάσης Δεδομένων με τη δημιουργία αποθηκευμένων διαδικασιών (stored procedures). Οι συναρτήσεις αυτές μπορούν να χρησιμοποιηθούν για τη δημιουργία triggers ή την εκτέλεση πολύπλοκων ερωτημάτων που υλοποιούνται δύσκολα με χρήση απλής SQL.

Για λόγους ασφαλείας οι γλώσσες προγραμματισμού που υποστηρίζονται από την Postgres διαχωρίζονται σε δύο ομάδες, τις ασφαλείς (safe) και τις μη ασφαλείς (unsafe). Ο διαχωρισμός τους γίνεται με κριτήρια που αφορούν την πρόσβασή τους σε ευαίσθητα σημεία του συστήματος, όπου λάθη στον κώδικα των συναρτήσεων μπορούν να προκαλέσουν αλλοίωση ή απώλεια δεδομένων. Ασφαλείς γλώσσες θεωρούνται οι SQL και PL/pgSQL, ενώ η C, αν και ταχύτερη όλων, θεωρείται μη ασφαλής.

-Διεπαφές

Η Postgres διαθέτει επίσης μεγάλη ποικιλία βιβλιοθηκών διεπαφής με γλώσσες προγραμματισμού (Java, ODBC, Perl, Python, Ruby, C, C++, PHP, Lisp, Scheme, Qt), γεγονός που επιτρέπει σε πολλές εφαρμογές να αλληλεπιδράσουν άμεσα με την Postgres.

3.1.2 PostGIS

Το σύστημα PostGIS [32] είναι ένα πακέτο λογισμικού ανοιχτού κώδικα το οποίο προσθέτει υποστήριξη για γεωχωρικά δεδομένα στο ΣΔΒΔ Postgres, προ-σθέτοντας έτσι λειτουργικότητα Χωρικής Βάσης Δεδομένων στο ΣΔΒΔ. Το PostGIS ακολουθεί τα πρότυπα που έχουν καθοριστεί από το Open Geospatial Consortium (OGC). Τα σημαντικότερά του χαρακτηριστικά είναι τα εξής:

- Κατάλληλοι τύποι δεδομένων για την αναπαράσταση σημείων, γραμμών, πολυγώνων καθώς και συλλογών σημείων, γραμμών, πολυγώνων και γεωμετριών.
- Υποδομή για συσχέτιση γεωμετριών βάσει του προτύπου αλληλεπιδρά-σεων 3x3 DE-9IM [9] (μοντέλο 9 αλληλεπιδράσεων).
- Χωρικοί τελεστές για πραγματοποίηση γεωχωρικών υπολογισμών όπως εμβαδόν, απόσταση, μήκος, περίμετρος.
- Χωρικοί τελεστές για πράξεις συνόλου μεταξύ γεωμετριών (ένωση, διαφορά συνόλων, συμμετρική διαφορά συνόλων, buffers.
- Χωρικά ευρετήρια (R-tree βασιζόμενα σε GiST).
- Επιλογή ευρετηρίου προς χρήση από τον μηχανισμό βελτιστοποίησης ερωτημάτων, για τη δημιουργία αποδοτικών πλάνων για χωρικά και μη χωρικά ερωτήματα.
- Εργαλεία επεξεργασίας δεδομένων κανάβου (raster) με το πακέτο PostGIS WKT Raster.

Ειδικότερα χαρακτηριστικά:

- Μετατροπή προβολής σε διαφορετικό Σύστημα Αναφοράς μέσω SQL για διανυσματικά και raster δεδομένα.
- Εισαγωγή/εξαγωγή δεδομένων σε ESRI shapefile και άλλα πρότυπα μέσω εργαλείων ανοιχτού κώδικα.
- Υποστήριξη πληθώρας τύπων raster δεδομένων (GeoTiff, NetCDF, PNG, JPG).
- Εισαγωγή και απεικόνιση διανυσματικών δεδομένων από πρότυπα κειμένου KML,GML, GeoJSON,GeoHash και WKT με χρήση SQL.
- Απεικόνιση raster δεδομένων σε πρότυπα GeoTIFF, PNG, JPG, NetCDF με χρήση SQL.
- Συναρτήσεις συρραφής raster δεδομένων βάσει γεωμετρίας, εξαγωγής διανυσματικών από raster με χρήση SQL.
- Υποστήριξη τρισδιάστατων τύπων δεδομένων με αντίστοιχα ευρετήρια και λειτουργίες.
- Υποστήριξη τοπολογίας δικτύου
- Υποστήριξη λειτουργιών γεωαναφοράς και αντίστροφης γεωαναφοράς.

Για τους σκοπούς της παρούσας εργασίας επιλέχθηκε η έκδοση PostGIS 2.1 για σύστημα Postgres 9.3.

3.1.3 pgRouting

Με το όνομα pgRouting [33] εννοείται το σύνολο συναρτήσεων που χρησι-μοποιείται σαν πρόσθετο μιας εγκατάστασης Postgres/PostGIS για την παροχή υποστήριξης σε διαδικασίες δρομολόγησης εντός τοπολογίας δικτύου. Τα κυριό-τερα χαρακτηριστικά λειτουργίας του pgRouting είναι τα εξής:

- Τα δεδομένα και τα χαρακτηριστικά τους μπορούν να τροποποιηθούν από πολλά προγράμματα-πελάτες, όπως τα QGIS και uDig, μέσω τεχνολογιών όπως το JDBC, το ODBC, ή απευθείας μέσω PL/pgSQL.
- Αλλαγές στα δεδομένα διαμορφώνουν απευθείας τον γράφο του δικτύου, χωρίς να υπάρχει ανάγκη για επανυπολογισμό της γεωμετρίας του.
- Η παράμετρος κόστους μετακίνησης μπορεί να υπολογιστεί δυναμικά με χρήση SQL και η τιμή της μπορεί να προέρχεται από πολλαπλά πεδία ή πίνακες.

Το pgRouting παρέχει συναρτήσεις που υλοποιούν τους εξής αλγορίθμους δρο-μολόγησης:

- All Pairs Shortest Path, αλγόριθμος Floyd-Warshall
- All Pairs Shortest Path, αλγόριθμος Johnson
- Shortest Path A*
- Bi-directional Dijkstra Shortest Path
- Bi-directional A* Shortest Path
- Shortest Path Dijkstra
- Driving Distance
- K-Shortest Path, Multiple Alternative Paths
- K-Dijkstra, One to Many Shortest Path
- Traveling Sales Person
- Turn Restriction Shortest Path (TRSP)

Για τους σκοπούς της παρούσας εργασίας επιλέχθηκε η έκδοση 2.0.0 του pgRouting για σύστημα Postgres 9.3.

3.1.4 Quantum GIS [39]

Το QGIS (παλαιότερα γνωστό ως Quantum GIS) είναι ένα Γεωγραφικό Πληροφοριακό Σύστημα (GIS) ανοιχτού κώδικα (υπό την άδεια GNU GPL), συμβατό με λειτουργικά συστήματα Windows, Linux, Mac OS X και Android (σε πειραματικό στάδιο).

Το QGIS παρέχει λειτουργικότητα εμφάνισης, επεξεργασίας και ανάλυσης γεωγραφικών δεδομένων, ενώ, όπως πολλά άλλα συστήματα GIS επιτρέπει στους χρήστες να δημιουργήσουν χάρτες με μεγάλο αριθμό επιπέδων ή σε διαφορετικές χαρτογραφικές προβολές. Οι δημιουργούμενοι χάρτες μπορούν να υποστηρίζουν διαφορετικά πρότυπα αρχείων, και τύπων δεδομένων, τόσο διανυσματικά (vector) όσο και κανάβου (raster). Τα διανυσματικά δεδομένα διαχωρίζονται, όπως στα περισσότερα GIS στις κατηγορίες σημείο, γραμμή και

πολύγωνο, ενώ πολλοί τύποι αρχείων raster δεδομένων μπορούν να γεω-αναφερθούν μέσω του προγράμματος. Το πρόγραμμα επίσης υποστηρίζει τη χρήση αρχείων Autocad dxf, ESRI shp, coverage, mdb (ArcGIS personal geodatabase) ενώ υποστηρίζει επίσης χαρτογραφικές υπηρεσίες Διαδικτύου WMS και WFS για την προσθήκη δεδομένων από εξωτερικές πηγές.

Το QGIS παρουσιάζει πολύ καλή διαλειτουργικότητα με άλλα πακέτα ελεύθερου λογισμικού γεωχωρικής φύσης, όπως τα PostGIS, GRASS και MapServer, παρέχοντας στους χρήστες ακόμα μεγαλύτερη λειτουργικότητα και ευελιξία. Ειδικά πρόσθετα, υλοποιημένα σε γλώσσα Python ή C++, επεκτείνουν ακόμη περισσότερο τις δυνατότητες του QGIS, με μερικά εξ αυτών να υλοποιούν γεωκωδικοποίηση με χρήση του Google Geocoding API, επεξεργασία γεωγραφικών δεδομένων παρόμοια με τα εργαλεία του ArcGIS ή άμεση διασύνδεση με ΣΔΒΔ PostgreSQL/PostGIS, SpatiaLite και MySQL.

Λόγω της διαθεσιμότητας του κώδικα στο ευρύ κοινό, οι λειτουργίες του μπορούν να τροποποιηθούν από τους χρήστες όπου κρίνεται αναγκαίο και να διανεμηθούν σε μετέπειτα εκδόσεις του QGIS ή ως πρόσθετα.

Για την απεικόνιση των δεδομένων τροχιών και του χαρτογραφικού υποβάθρου, καθώς και την απεικόνιση των αποτελεσμάτων από τα ερωτήματα ακριβούς τροχιάς, χρησιμοποιήθηκε το QuantumGIS 2.0.1 “Dufour”.

3.2 Δεδομένα

Στην παρούσα προσέγγιση, χρησιμοποιήθηκαν δύο τύπων δεδομένα. Τα δεδομένα υποβάθρου, που περιέχουν τα δεδομένα που αναπαριστούν το δίκτυο της περιοχής που εξετάζεται και τα δεδομένα τροχιών, που περιλαμβάνουν τη θέση των κινούμενων αντικειμένων ως προς τις διαστάσεις του χώρου και του χρόνου.

3.2.1 Δεδομένα Υποβάθρου

Όπως ήδη αναφέρθηκε, στην κατηγορία αυτή εμπίπτουν τα δεδομένα που αναπαριστούν το δίκτυο δρόμων της εξεταζόμενης περιοχής. Για το σκοπό αυτόν, χρησιμοποιήθηκαν δεδομένα από την κοινότητα OpenStreetMap, ενώ για την επεξεργασία και προσαρμογή τους στα χαρακτηριστικά της παρούσας εφαρμογής χρησιμοποιήθηκαν αλγόριθμοι του προγράμματος Routing/Travel Time Analysis στους οποίους παρακάτω θα γίνει συνοπτική αναφορά.

- Δεδομένα OpenStreetMap [35]

Το OpenStreetMap (OSM) είναι ένα συνεργατικό πρόγραμμα για τη δημιουργία ενός ελεύθερου, επεξεργάσιμου χάρτη για όλο τον κόσμο. Οι δύο κύριες κινητήριες δυνάμεις για την ανάπτυξη του προγράμματος υπήρξαν η ανάγκη για την ύπαρξη χαρτογραφικών δεδομένων σε περιοχές που η διαθεσιμότητά τους ήταν περιορισμένη και η έλευση των οικονομικά προσιτών συσκευών δορυφορικού εντοπισμού (GPS).

Το πρόγραμμα ιδρύθηκε από τον Steve Coast το 2004 στη Βρετανία, εμπνεόμενο από την επιτυχία που είχε ήδη σημειώσει η Wikipedia και την υπεροχή των ιδιόκτητων χαρτογραφικών δεδομένων στο Ηνωμένο Βασίλειο και

σε άλλες περιοχές. Από ιδρύσεώς του, πάνω από 1,6 εκατομμύρια χρήστες έχουν συμβάλλει στη χαρτογράφηση, συλλέγοντας δεδομένα μέσω τοπογραφικών μεθόδων, συσκευών GPS, αεροφωτογραφιών και άλλων δωρεάν ελεύθερων πηγών δεδομένων. Αυτά τα δεδομένα που έχουν ως πηγή όλους τους συμμετέχοντες στο πρόγραμμα, είναι διαθέσιμα μέσω της άδειας χρήσης Open Database.

Το βασικό προϊόν του OpenStreetMap δεν θεωρείται ο χάρτης αυτός καθαυτός, αλλά τα δεδομένα που τον απαρτίζουν, τα οποία με την πάροδο του χρόνου γίνονται όλο και πιο ακριβή και αξιόπιστα, γεγονός που έχει συντελέσει στην χρήση τους από πολλές δημοφιλείς ιστοσελίδες, συσκευές GPS και εφαρμογές. Ωστόσο, η ποιότητα των δεδομένων παρουσιάζει διακυμάνσεις ποιότητας αναλόγως των περιοχών που απεικονίζει.

-Μορφοποίηση δεδομένων

Το OpenStreetMap χρησιμοποιεί μια τοπολογική δομή δεδομένων, που αποτελείται από τέσσερα βασικά στοιχεία (πρωταρχικοί τύποι δεδομένων):

1. Κόμβοι (nodes): Σημεία με καθορισμένη γεωγραφική θέση, προσδιορισμένη με μορφή συντεταγμένων (γεωγραφικό μήκος, γεωγραφικό πλάτος), στο σύστημα αναφοράς WGS84. Εκτός από τη χρήση τους ως σημεία δρόμων, χρησιμοποιούνται και για την αναπαράσταση οντοτήτων του χάρτη χωρίς συγκεκριμένες διαστάσεις, όπως σημεία ενδιαφέροντος ή κορυφές βουνών.
2. Δρόμοι (ways): Αποτελούν ταξινομημένες λίστες κόμβων, οι οποίες αναπαριστούν μια πολυγραμμή (polyline), ή σπανιότερα ένα πολύγωνο (polygon) αν σχηματίζουν βρόγχο. Χρησιμοποιούνται για την αναπαράσταση γραμμικών οντοτήτων όπως δρόμοι και ποτάμια και περιοχών, όπως δάση, πάρκα και λίμνες.
3. Συσχετίσεις (relations): ταξινομημένες λίστες κόμβων, δρόμων και συσχετίσεων (μέλη), όπου κάθε μέλος μπορεί να έχει έναν «ρόλο», ο οποίος αναπαρίσταται με μια συμβολοσειρά. Χρησιμοποιούνται για να αναπαραστήσουν τη σχέση μεταξύ κόμβων και δρόμων. Μπορούν να χρησιμοποιηθούν για την εισαγωγή περιορισμών αλλαγής κατεύθυνσης σε δρόμους, ή για συσχέτιση διαδρομής με πολλούς δρόμους.
4. Ετικέτες (tags): Ζεύγη κλειδιού-τιμής (key-value). Χρησιμοποιούνται για την αποθήκευση μεταδεδομένων σχετικών με τις οντότητες που απεικονίζονται στο χάρτη (όπως ο τύπος, το όνομα ή φυσικές τους ιδιότητες). Οι ετικέτες είναι πάντα σχετιζόμενες με έναν κόμβο, ένα δρόμο ή μια συσχέτιση.

-Αποθήκευση δεδομένων

Οι πρωταρχικοί τύποι OSM δεδομένων αποθηκεύονται και επεξεργάζονται σε συγκεκριμένα πρότυπα. Τα δεδομένα για όλο τον πλανήτη βρίσκονται αποθηκευμένα σε μια Postgres Βάση Δεδομένων, η οποία διαθέτει έναν πίνακα για κάθε πρωταρχικό τύπο δεδομένων, με μία εγγραφή ανά αντικείμενο. Κάθε επεξεργασία των δεδομένων γίνεται εντός αυτής της Βάσης Δεδομένων και κάθε αρχείο δεδομένων δημιουργείται από αυτή, ακολουθώντας κάθε φορά συγκεκριμένα πρότυπα (XML, SHP, GeoJSON, PBF).

Στην παρούσα εργασία, η περιοχή κίνησης των κινούμενων αντικειμένων είναι ο Νομός Αττικής, συνεπώς χρησιμοποιήθηκε το OSM metro extract

δεδομένων για την περιοχή της Αθήνας, το οποίο παρέχεται από τον ιστότοπο Mapzen [36]. Το πρότυπο αρχείου που χρησιμοποιήθηκε είναι το XML (athens-greece.osm), λόγω της εύκολης επεξεργασίας του .osm αρχείου για την εξαγωγή δεδομένων που ήταν απαραίτητη σε διάφορα σημεία της εφαρμογής.

3.2.2 Routing/Travel Time Analysis

Ένα πολύ σημαντικό σημείο της εφαρμογής είναι η αντιστοίχιση των δεδομένων τροχιών κινουμένων αντικειμένων με τα δεδομένα υποβάθρου. Η σημαντικότητα αυτή έγκειται στο γεγονός ότι η αντιστοίχιση αυτή αποτελεί τη «γέφυρα» μεταξύ των πραγματικών δεδομένων και της αναπαράστασης εντός της ΒΔ. Μία από τις δυσκολίες της αντιστοίχισης είναι η διαφορά γεωμετριών μεταξύ δεδομένων και δικτύου, αφού τα δεδομένα τροχιών είναι σημεία (κόμβοι) σε χρονολογική σειρά, ενώ το δίκτυο του χαρτογραφικού υποβάθρου αποτελείται από γραμμές (δρόμους). Η ασυμβατότητα αυτή, σε συνδυασμό με την πολλές φορές αραιή κατανομή των θέσεων του οχήματος εντός δικτύου και τους περιορισμούς του δικτύου κάνουν την αντιστοίχιση αυτή περίπλοκη υπόθεση.

Για την εκτέλεση ερωτημάτων ακριβούς τροχιάς απαιτείται η ακριβής γνώση των τμημάτων που έχει επισκεφθεί κάθε τροχιά, με τη σειρά που αυτή έχει γίνει (λεπτομερέστερη περιγραφή πραγματοποιείται στο επόμενο κεφάλαιο). Για το σκοπό αυτό χρησιμοποιείται ένας αλγόριθμος χαρτογραφικής αντιστοίχισης (map-matching) ο οποίος περιγράφεται στο [18], έχει ενσωματωθεί στο OSM Wiki [37] και ο πηγαίος του κώδικας (έχει υλοποιηθεί σε C#) είναι ελεύθερα διαθέσιμος [38].

Για τους σκοπούς της εργασίας χρησιμοποιήθηκαν ως black boxes τα εκτελέσιμα OSM2Routing.exe και MatchGPX2OSM.exe, ενώ για την επεξεργασία των παραγόμενων αρχείων χρησιμοποιήθηκαν scripts γραμμένα σε γλώσσα PHP.

4. Υλοποίηση εφαρμογής

Στο παρόν κεφάλαιο παρουσιάζονται με χρονολογική σειρά υλοποίησης οι μέθοδοι και οι τεχνικές που χρησιμοποιήθηκαν για την υλοποίηση των ερωτημάτων SPQ και των αλγορίθμων που χρησιμοποιούνται υποστηρικτικά για τον περιορισμό των σφαλμάτων τους. Ακόμη περιγράφεται η υποδομή για τη φόρτωση και το μετασχηματισμό των δεδομένων ώστε να είναι εφικτή η λειτουργία των ερωτημάτων SPQ.

Για καλύτερη ροή του κειμένου, πολλές φορές παραλείπονται καθαρά τεχνικές λεπτομέρειες σχετικά με την υλοποίηση των συναρτήσεων, οι οποίες παρουσιάζονται αναλυτικότερα στο Παράρτημα του παρόντος τόμου. Ο πλήρης κώδικας των συναρτήσεων και των scripts που αναφέρονται είναι διαθέσιμος με τα αρχεία που συνοδεύουν την εργασία.

4.1 Χαρτογραφικό υπόβαθρο

Όπως ήδη αναφέρθηκε στο προηγούμενο κεφάλαιο, τα δεδομένα υποβάθρου που επιλέχθηκαν για τους σκοπούς της εργασίας είναι τα δεδομένα OSM (Open Street Map) της ευρύτερης περιοχής των Αθηνών. Πρόκειται για δεδομένα που καλύπτουν το σύνολο του Νομού Αττικής (πλην νήσων) και μικρό κομμάτι από τους Νομούς Κυκλάδων, Ευβοίας, Βοιωτίας, Κορινθίας και Αργολίδας (Εικόνα 4.1). Τα δεδομένα αποτελούν OSM Metro extract σε μορφή XML (αρχείο .osm) και είναι διαθέσιμα στον ιστότοπο Mapzen [36].

Εικόνα 4.1: Περιοχή δεδομένων OSM

Τα δεδομένα που επιλέχθηκαν περιέχουν σημεία, γραμμές, πολύγωνα και ετικέτες για την ευρύτερη περιοχή των Αθηνών, ωστόσο στην προκειμένη περίπτωση τα δεδομένα γραμμών αποτελούν τα σημαντικότερα δεδομένα του αρχείου, τα οποία πρέπει να εξαχθούν από αυτό για τη φόρτωση στη Βάση Δεδομένων.

Ωστόσο, η απλή ανάκτηση και φόρτωση των γραμμικών δεδομένων στη ΒΔ κατευθείαν μετά την εξαγωγή από το αρχείο δεν είναι επαρκής για τις

ανάγκες της συγκεκριμένης εφαρμογής. Οι λόγοι για τους οποίους απαιτείται περαιτέρω επεξεργασία είναι:

- Ύπαρξη ακατάλληλων κατηγοριών οδών για ένταξη στο δίκτυο. Στην εν λόγω εφαρμογή τα δεδομένα κίνησης που εξετάζουμε αναφέρονται σε οχήματα, συνεπώς πρέπει να αποκλειστούν από τις γραμμικές οντότητες
- Πρότυπο οργάνωσης γραμμικών οντοτήτων ακατάλληλο για δικτυακή τοπολογία.
- Πρότυπο οργάνωσης σημειακών οντοτήτων ακατάλληλο για δικτυακή τοπολογία.
- Ανάγκη για αντιστοίχηση δεδομένων θέσης οχημάτων στις ακμές του δικτυακού υποβάθρου.

Για την αντιμετώπιση των τριών πρώτων παραπάνω προβλημάτων μπορεί να χρησιμοποιηθεί απ' ευθείας το pgRouting, η ύπαρξη του τελευταίου προβλήματος δημιουργεί άλλα δεδομένα στην αντιμετώπιση των γραμμικών δεδομένων του αρχείου.

Η αντιστοίχιση των δεδομένων θέσης στις ακμές του δικτυακού υποβάθρου γίνεται με κώδικα από το πρόγραμμα Routing/Travel Time Analysis [37], που περιέχει τα εργαλεία OSM2Routing και MatchGPX2OSM τα οποία θα χρησιμοποιηθούν για τις ανάγκες της εφαρμογής.

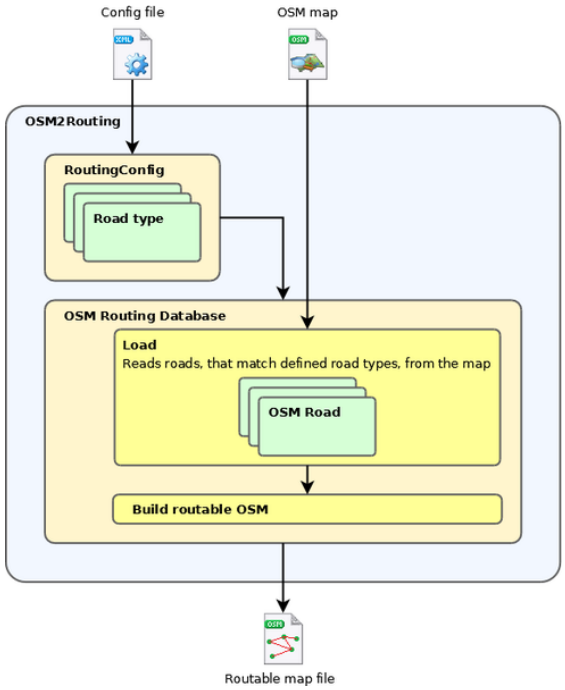
-OSM2Routing [37], [38]

Το OSM2Routing αποτελεί μια εφαρμογή που διαβάζει το αρχείο OSM των δεδομένων υποβάθρου και ένα αρχείο config στο οποίο καθορίζονται οι κατηγορίες οδών που θα περιέχονται στο δίκτυο και εξάγει ένα νέο αρχείο OSM με οδούς από τις επιλεγμένες κατηγορίες και με κατάλληλη διαμόρφωση ώστε να ακολουθούν τοπολογία δικτύου. Αυτό το βήμα είναι απαραίτητο, διότι τα απλά δεδομένα OSM περιέχουν ακριβείς πληροφορίες για τις οδούς της περιοχής, χωρίς ωστόσο αυτές να ανήκουν σε τοπολογία δικτύου. Οι ακμές που αναπαριστούν τις οδούς είναι ενιαίες, δηλαδή κάθε οδός αποτελείται συνήθως από μόνο μία ακμή, ενώ η τοπολογία δικτύου απαιτεί τουλάχιστον *n-1* ακμές για κάθε οδό, αν *n* είναι τα σημεία τομής της με άλλες οδούς του δικτύου (Εικόνα 4.2).

Εικόνα 4.2: Ακμή οδού πριν και μετά το OSM2Routing

Μετά την επεξεργασία από το OSM2Routing, κάθε δρόμος «σπάει» σε ακμές με άκρα δύο σημεία, κοινά με άλλες ακμές, τα οποία υποδηλώνουν τη

δυνατότητα μετάβασης από μία ακμή σε άλλη κατά τη δρομολόγηση. Τα σημεία αυτά είναι τα σημεία που μετέπειτα θα χρησιμοποιηθούν για τη δρομολόγηση εντός του δικτύου από τους αλγορίθμους του pgRouting. Η λειτουργία του OSM2Routing παρουσιάζεται διαγραμματικά στην Εικόνα 4.3. Το παραγόμενο αρχείο είναι ένα XML αρχείο (με κατάληξη .osm), το οποίο περιέχει τις ακμές του δικτύου και τους κόμβους του δικτύου που αποτελούν τα κοινά σημεία μετάβασης μεταξύ των ακμών.



Εικόνα 4.3: Λειτουργίες OSM2Routing [37]

4.1.1 Φόρτωση δεδομένων χάρτη

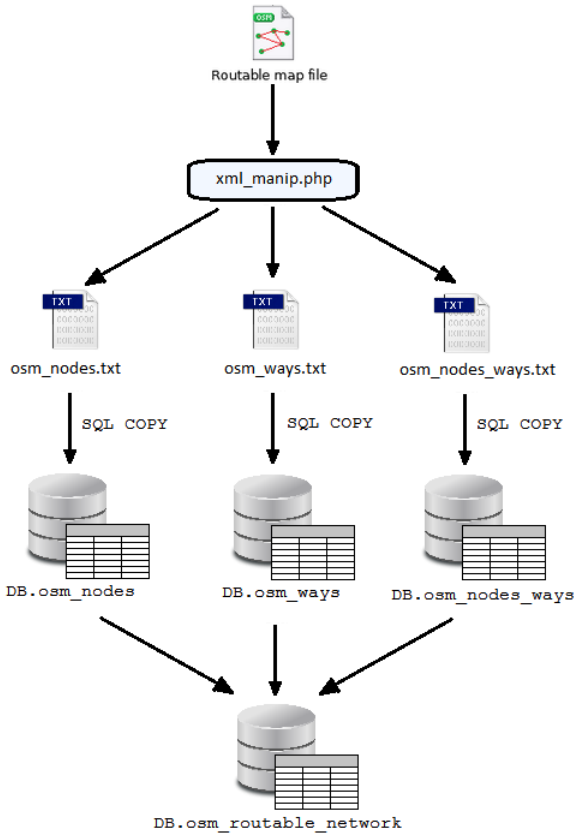
Το αρχείο xml_manip.php περιέχει τον κώδικα για την ανάγνωση του παραγόμενου αρχείου (routable map file). Το xml_manip.php διαβάζει το XML αρχείο του δρομολογημένου χάρτη και εξάγει τρία (3) txt αρχεία με τις πληροφορίες για:

1. Τους κόμβους του δικτύου (osm_nodes.txt)
2. Τους δρόμους του δικτύου (osm_ways.txt)
3. Τη συσχέτιση κόμβων-δρόμων (osm_nodes_ways.txt)

Τα αρχεία αυτά, παρ'όλο που είναι απλά αρχεία κειμένου (TXT) ακολουθούν σε μεγάλο βαθμό τη μορφοποίηση των CSV, καθώς περιέχουν πληροφορία διατεταγμένη σε πεδία, με κάθε γραμμή του αρχείου να περιέχει παραμέτρους που διαχωρίζονται μεταξύ τους με σύμβολο (στην προκειμένη περίπτωση το ελληνικό ερωτηματικό ';'). Τα περιεχόμενα αυτών των αρχείων μπορούν να φορτωθούν σε αντίστοιχους πίνακες της ΒΔ με απλές εντολές SQL COPY. Οι τεχνικές λεπτομέρειες της φόρτωσης στη ΒΔ αναφέρονται στο Παράρτημα, ενώ η διαδικασία παρουσιάζεται οπτικά στην Εικόνα 4.4.

4.1.2 Δημιουργία τοπολογίας δικτύου

Μετά τη δημιουργία των πινάκων osm_nodes, osm_ways και osm_nodes _ways και τη φόρτωση των δεδομένων σε αυτούς από τα αρχεία txt (Εικόνα 4.5) δημιουργείται ο πίνακας που θα φιλοξενήσει τα δεδομένα του δικτύου οδών (osm_routable_network) και εισάγονται σε αυτόν τα δεδομένα των ακμών του δικτύου. Τα δεδομένα αυτά εξάγονται με SQL από τους osm_nodes, osm_ways και osm_nodes _ways και περιέχουν το αναγνωριστικό της ακμής, που προέκυψε από το OSM2Routing και τη γεωμετρία της ακμής, η οποία θα χρησιμοποιηθεί για τους σκοπούς του pgRouting. Ακόμη, στον πίνακα δημιουργούνται πεδία για τη φιλοξενία των τιμών των βαρών που αντιστοιχούν σε κάθε ακμή του δικτύου και χρησιμοποιούνται κατά την επεξεργασία του SPQ.



Εικόνα 4.4: Πέρασμα δεδομένων χάρτη στη ΒΔ

Μετά τη δημιουργία του πίνακα osm_routable_network και τη φόρτωση των δεδομένων δικτύου σε αυτόν, καλείται η pl/pgsql συνάρτηση `assign_weights_to_routable_segments()` για την εισαγωγή των βαρών μήκους και πρώτων παραγόντων σε κάθε ακμή του δικτύου. Η συνάρτηση αυτή υπολογίζει το μήκος κάθε ακμής από το πεδίο γεωμετρίας της (geom) και το εισάγει στο πεδίο length του πίνακα, ενώ υπολογίζει και έναν μοναδικό για κάθε ακμή θετικό πρώτο αριθμό, του οποίου τον δεκαδικό λογάριθμο εισάγει στο πεδίο hash του πίνακα.

Τέλος, με εκτέλεση κατάλληλων εντολών SQL φορτώνονται στη ΒΔ οι συναρτήσεις του pgRouting και δημιουργούνται τα κατάλληλα πεδία και ευρετήρια στον πίνακα osm_routable_network για την εκτέλεση ερωτημάτων δρομολόγησης στο δίκτυο. Μετέπειτα δημιουργείται η τοπολογία δικτύου για τα

δεδομένα του osm_routable_network, η οποία έχει σαν συνέπεια και την ταυτόχρονη δημιουργία του πίνακα osm_routable_network_vertices_pgr, ο οποίος περιέχει τους κόμβους δικτύου στους οποίους μπορεί να παρατηρηθεί μετάβαση από μία ακμή δικτύου σε άλλη.

osm_nodes.txt

```
78695;37.5964742;23.0709818
78696;37.5961011;23.0711918
78697;37.5958087;23.071361
78698;37.5956932;23.0716514
78699;37.5957626;23.0729767
78700;37.595817;23.0737664
78701;37.5958555;23.0745927
78702;37.5958921;23.0752386
```

DB.osm_nodes

	id [PK] serial	osm_id bigint	lat double precision	lon double precision	geom geometry
1	1	78695	37.5964742	23.0709818	01010000007890
2	2	78696	37.5961011	23.0711918	01010000004EB
3	3	78697	37.5958087	23.071361	0101000000B93
4	4	78698	37.5956932	23.0716514	01010000006EB
5	5	78699	37.5957626	23.0729767	0101000000B9D
6	6	78700	37.595817	23.0737664	01010000002D8
7	7	78701	37.5958555	23.0745927	0101000000350
8	8	78702	37.5958921	23.0752386	01010000009665

osm_ways.txt

```
-1;4263049;30;yes;no
-2;4263049;30;yes;no
-3;4349655;50;yes;yes
-4;4349655;50;yes;yes
-5;4349655;50;yes;yes
-6;4349655;50;yes;yes
-7;4349655;50;yes;yes
-8;4349655;50;yes;yes
```

DB.osm_ways

	id [PK] serial	routable_osm_id bigint	osm_id bigint	speed integer	accessible character vary	accessible_reverse character varying(
1	1	-1	4263049	30	yes	no
2	2	-2	4263049	30	yes	no
3	3	-3	4349655	50	yes	yes
4	4	-4	4349655	50	yes	yes
5	5	-5	4349655	50	yes	yes
6	6	-6	4349655	50	yes	yes
7	7	-7	4349655	50	yes	yes
8	8	-8	4349655	50	yes	yes

osm_nodes_ways.txt

```
-3;4349655;1281800322;1
-3;4349655;1281800373;2
-3;4349655;1281800247;3
-4;4349655;1281800247;1
-4;4349655;26490736;2
-5;4349655;26490736;1
-5;4349655;26490724;2
-6;4349655;26490724;1
-6;4349655;26490632;2
```

DB.osm_nodes_ways

	id [PK] serial	routable_osm_id bigint	osm_way_id bigint	osm_node_id bigint	node_no integer
25	25	-3	4349655	1281800322	1
26	26	-3	4349655	1281800373	2
27	27	-3	4349655	1281800247	3
28	28	-4	4349655	1281800247	1
29	29	-4	4349655	26490736	2
30	30	-5	4349655	26490736	1
31	31	-5	4349655	26490724	2
32	32	-6	4349655	26490724	1
33	33	-6	4349655	26490632	2

Εικόνα 4.5: Εισαγωγή δεδομένων στους πίνακες osm_nodes, osm_ways, osm_nodes_ways

Μετά το πέρας της διαδικασίας, η υποδομή δικτύου στην οποία θα αντιστοιχισθούν οι θέσεις των οχημάτων και θα εκτελεστούν τα ερωτήματα SPQ έχει δημιουργηθεί επιτυχώς (Εικόνα 4.6).

DB.osm_routable_network

	id serial	routable_osm_id bigint	geom geometry	source integer	target integer	length double precision	hash double precision
1	568	-568	0102000020	91065	150071	64.106582280	3.7460890430
2	569	-569	0102000020	150071	150072	38.353367549	3.7467120225
3	570	-570	0102000020	150072	150067	4.6407799155	3.7474894922
4	571	-571	0102000020	135360	150064	11.476940237	3.7499680835
5	572	-572	0102000020	150064	24174	231.19950557	3.7512020945
6	573	-573	0102000020	24174	135395	164.19012118	3.7513560997
7	574	-574	0102000020	143139	150073	35.345916842	3.7518177877
8	575	-575	0102000020	150073	91126	46.161147257	3.7521253072
9	576	-576	0102000020	91126	91124	7.5106564676	3.7522789854
10	577	-577	0102000020	91124	150074	69.467165318	3.7525861787

Εικόνα 4.6: Πίνακας δρομολογήσιμου δικτύου

4.2 Αντιστοίχιση δεδομένων στο χάρτη

Η αντιστοίχιση δεδομένων στο χάρτη (map-matching) είναι ένα πολύ σημαντικό κομμάτι της υλοποίησης της εφαρμογής, που μπορεί να επηρεάσει σημαντικά τα αποτελέσματα των SPQ ερωτημάτων ως προς την ορθότητά τους, μιας και αποτελεί το συνδετικό κρίκο της πραγματικής κίνησης των οχημάτων με την κίνηση που αναπαριστάται εντός της ΒΔ. Η αντιστοίχιση γίνεται σε δύο στάδια,

1. την αντιστοίχιση των δεδομένων σε ακμές βάσει του δρομολογήσιμου αρχείου OSM που έχει δημιουργηθεί από το OSM2Routing.
2. την εισαγωγή και επεξεργασία της παραγόμενης αντιστοίχισης εντός της ΒΔ.

4.2.1 XML Map-matching

Σε αυτό το στάδιο αντιστοίχισης αρχικά δημιουργείται ένα αρχείο GPX που περιέχει όλα τα δεδομένα θέσης των κινούμενων αντικειμένων που εξετάζονται. Η δημιουργία του αρχείου GPX από το αρχικό txt αρχείο γίνεται και πάλι μέσω ενός php script.

Το αρχείο gpx των δεδομένων θέσης των κινουμένων αντικειμένων δίδεται μαζί με το αρχείο osm του δικτύου δρομολόγησης σαν είσοδος στο πρόγραμμα MatchGPX2OSM, το οποίο επιχειρεί να αντιστοιχήσει τα δεδομένα του gpx πάνω στις ακμές του δικτύου που περιγράφεται στο αρχείο osm. Η διαδικασία που ακολουθείται από το MatchGPX2OSM παρουσιάζεται στην Εικόνα 4.7.

Ο MatchGPX2OSM εξετάζει την αντιστοίχιση με το χάρτη για κάθε κομμάτι τροχιάς ξεχωριστά. Συγκεκριμένα, κατά την κατασκευή του gpx αρχείου, επιλέγονται διαδοχικά σημεία για σχηματισμό n-1 τμημάτων τροχιάς για κάθε τροχιά αντικειμένου που απαρτίζεται από n σημεία. Δύο διαδοχικά σημεία αποτελούν ένα τμήμα (segment) της τροχιάς.

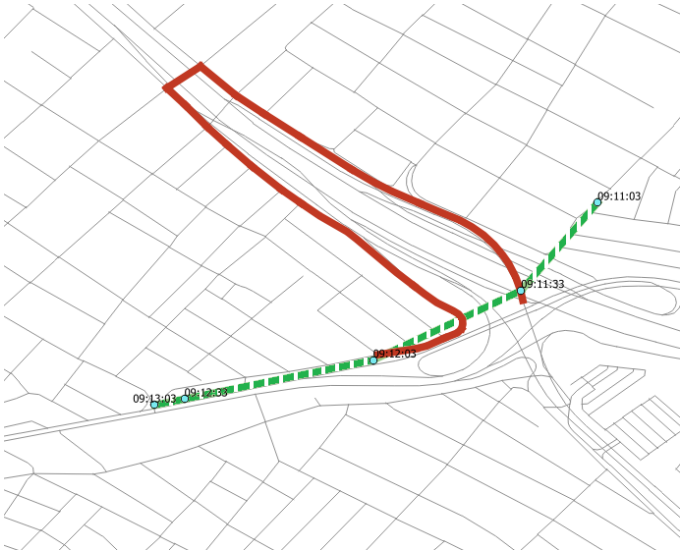
Ο MatchGPX2OSM παράγει ένα αρχείο για κάθε segment που βρίσκει εντός του gpx, στο οποίο αποτυπώνει, επίσης σε μορφή osm, το σημείο δικτύου (osm_node) στο οποίο αντιστοιχεί κάθε σημείο και κατ' επέκταση την ακμή ή τις ακμές του δικτύου από τις οποίες διέρχεται το segment τροχιάς που εξετάζεται.

-Περαιτέρω επεξεργασία

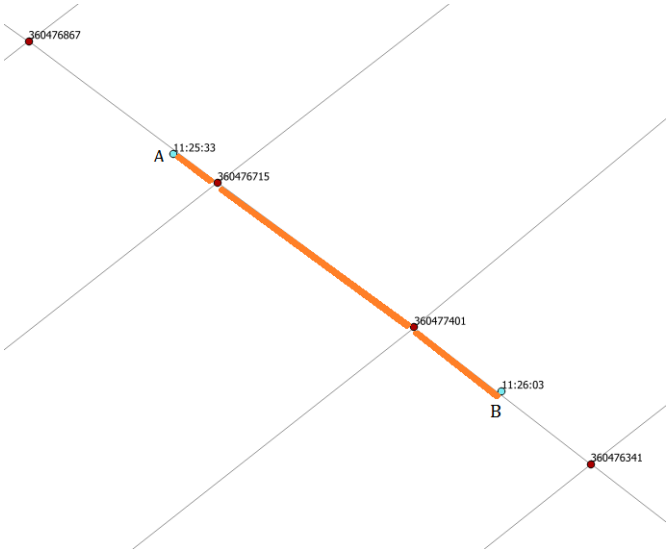
Η διαδικασία δεν εξάγει πάντα ξεκάθαρα αποτελέσματα: Είναι πολύ πιθανό το segment να αντιστοιχεί σε πολλές ακμές εντός του δικτύου, γεγονός που συμβαίνει πολύ συχνά όταν η δειγματοληψία είναι αραιή και οι δύο διαδοχικές θέσεις απέχουν μεταξύ τους μεγάλη απόσταση. Το ίδιο πρόβλημα μπορεί να προκύψει και όταν οι θέσεις των αντικειμένων φαίνονται να ακολουθούν τροχιά που δεν συμφωνεί με το υπάρχον δίκτυο. Στις περιπτώσεις αυτές το segment αντιστοιχείται σε ένα *μονοπάτι* ακμών εντός του δικτύου, συνήθως τη συντομότερη επιτρεπτή διαδρομή μεταξύ των σημείων(Εικόνα 4.8). Για την αντιπετώπιση του προβλήματος ενδέχεται η τροχιά να πρέπει να «σπάσει» σε μικρότερες κατά την αντιστοίχιση των θέσεων στο χάρτη, διατηρώντας όμως τα αρχικά της αναγνωριστικά ώστε να μπορεί να ανασυντεθεί μετά την αντιστοίχιση.

Επίσης, πολλές φορές, σε περιπτώσεις ιδιαίτερα πυκνής δειγματοληψίας, πολλά διαδοχικά segments μπορούν να αντιστοιχιστούν στην ίδια ακμή. Το πρόβλημα αυτό όμως μπορεί να λυθεί με μετέπειτα επεξεργασία εντός της ΒΔ, παραλείποντας τις διαδοχικές ίδιες ακμές βάσει του αναγνωριστικού τους, και διατηρώντας μόνο μία φορά την ακμή για το συγκεκριμένο τμήμα τροχιάς.

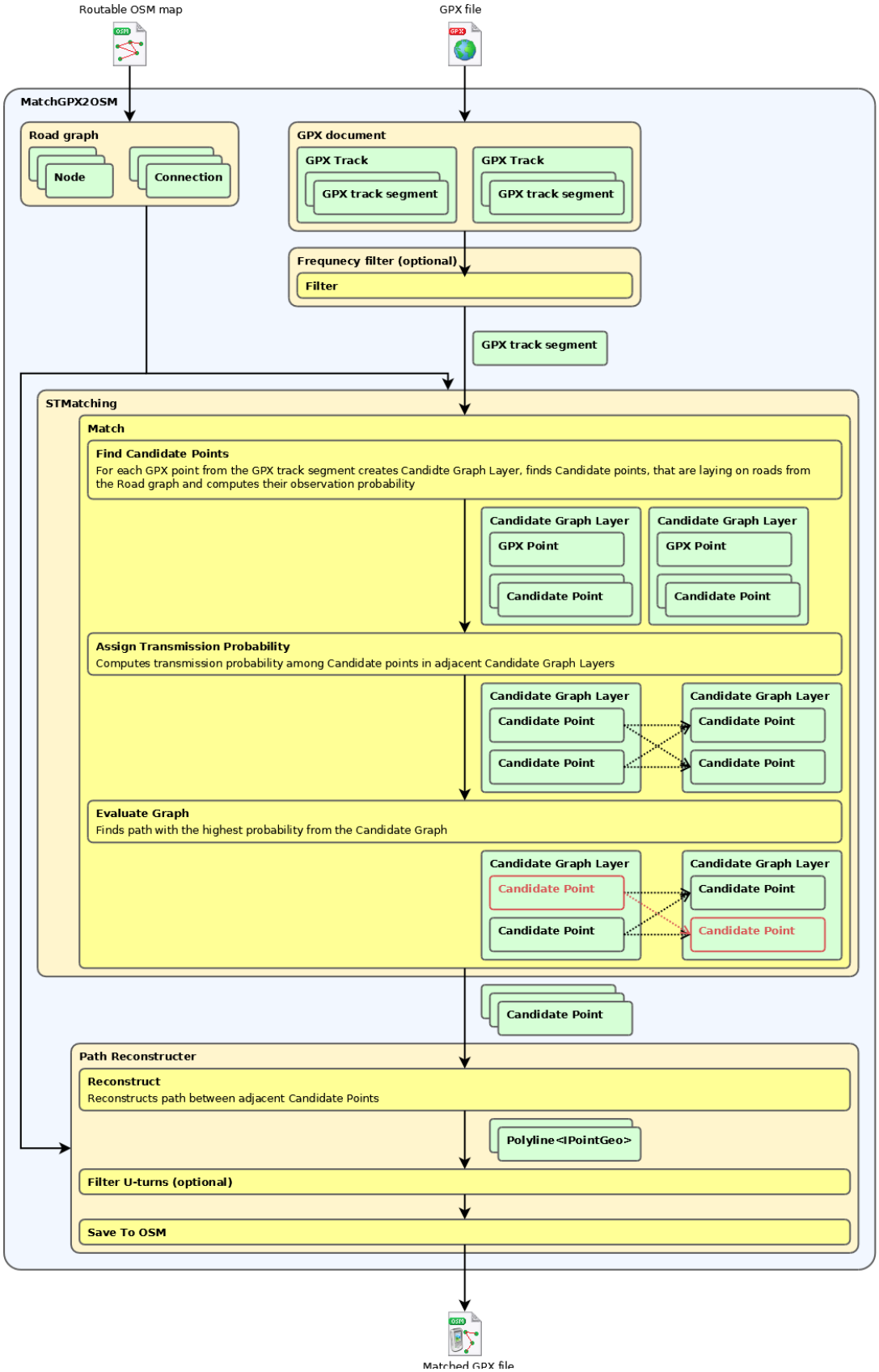
Τέλος, οι θέσεις που σηματοδοτούν την αρχή και το τέλος ενός segment σπάνια αντιστοιχίζονται σε κόμβο του υπάρχοντος δικτύου. Ωστόσο, αυτό συνήθως δεν αποτελεί πρόβλημα, διότι οι κόμβοι που ακολουθούν παρέχουν την πληροφορία για την αντιστοίχιση του segment σε συγκεκριμένη ακμή του δικτύου (Εικόνα 4.9).



Εικόνα 4.8: Επιλογή συντομότερης διαδρομής μεταξύ δύο σημείων. Η επιλογή της συντομότερης διαδρομής γίνεται υπό περιορισμούς δικτύου.



Εικόνα 4.9: Προσδιορισμός αντιστοιχούσας ακμής από τους ενδιάμεσους κόμβους του segment



Εικόνα 4.7: MatchGPX2OSM [37]

Στην περίπτωση της Εικόνας 4.9, το osm αρχείο για το segment που ορίζεται από τα σημεία A, B αναφέρει ότι το segment(A, B) ακολουθεί τα σημεία [A, 360476715, 360477401, B]. Από τα σημεία A, B δεν μπορεί να εξαχθεί συμπέρασμα, διότι δεν ανήκουν στους κόμβους δικτύου, ωστόσο από τους ενδιάμεσους κόμβους μπορεί να εξαχθεί η ακμή που ακολουθείται. Ομοίως για το επόμενο segment της τροχιάς, ο πρώτος ενδιάμεσος κόμβος θα είναι ο 360476341, ο οποίος και θα χρησιμοποιηθεί για τον προσδιορισμό της επόμενης ακμής που ακολουθείται από την τροχιά. Η διαδικασία επεξεργασίας παρουσιάζεται αναλυτικότερα στην επόμενη παράγραφο του παρόντος κεφαλαίου.

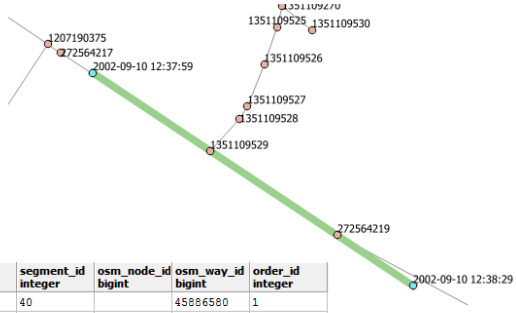
4.2.2 Εισαγωγή και επεξεργασία αντιστοίχισης στη ΒΔ

-Φόρτωση αντιστοιχίσεων στη ΒΔ

Όπως αναφέρθηκε στην προηγούμενη παράγραφο, ο αλγόριθμος MatchGPX2OSM δημιουργεί ένα osm αρχείο ανά segment τροχιάς, στο οποίο αναφέρεται η αντιστοίχιση της τροχιάς με τους δρόμους και τους κόμβους του δικτύου υποβάθρου. Η αντιστοίχιση με τις ακμές του δικτύου δρομολόγησης δεν είναι σαφής και προκύπτει μετά από περαιτέρω επεξεργασία.

Για την ανάκτηση των δεδομένων αντιστοίχισης για κάθε segment τροχιάς, χρησιμοποιείται ένα php script (gpx_reader.php), το οποίο ανατρέχει στο φάκελο με όλα τα osm αρχεία αντιστοίχισης και εξάγει σε ένα αρχείο txt με delimited format τα στοιχεία αντιστοίχισης για κάθε κόμβο που βρήκε εντός του αρχείου. Για παράδειγμα, στην Εικόνα 4.10 γίνεται αντιπαραβολή του xml κώδικα ενός osm αρχείου και των δεδομένων που εισάγει στον «ενδιάμεσο» πίνακα osm_visited_segments της ΒΔ.

```
<?xml version="1.0" encoding="utf-8"?>
<osm version="0.5" generator="OSMUtils">
  <node id="-2" visible="true" lat="37.9909585474383" lon="23.8769251090884">
    <tag k="time" v="10/9/2002 10:37:59 πμ" />
  </node>
  <node id="-3" visible="true" lat="37.9903137" lon="23.8779077">
    <tag k="node-id" v="1351109529" />
    <tag k="crossroad" v="yes" />
  </node>
  <node id="-5" visible="true" lat="37.9896171" lon="23.8789691">
    <tag k="node-id" v="272564219" />
  </node>
  <node id="-6" visible="true" lat="37.9892615801669" lon="23.8796350121984">
    <tag k="time" v="10/9/2002 10:38:29 πμ" />
  </node>
  <way id="-1" visible="true">
    <nd ref="-2" />
    <nd ref="-3" />
    <tag k="way-id" v="45886580" />
    <tag k="order" v="1" />
  </way>
  <way id="-4" visible="true">
    <nd ref="-3" />
    <nd ref="-5" />
    <nd ref="-6" />
    <tag k="way-id" v="45886580" />
    <tag k="order" v="2" />
  </way>
</osm>
```



	id [PK] serial	traj_id integer	trip_id integer	segment_id integer	osm_node_id bigint	osm_way_id bigint	order_id integer
3052	3052	862	1	40		45886580	1
3053	3053	862	1	40	1351109529	45886580	1
3054	3054	862	1	40	1351109529	45886580	2
3055	3055	862	1	40	272564219	45886580	2
3056	3056	862	1	40		45886580	2

Εικόνα 4.10: XML αρχείο και πίνακας osm_visited_segments

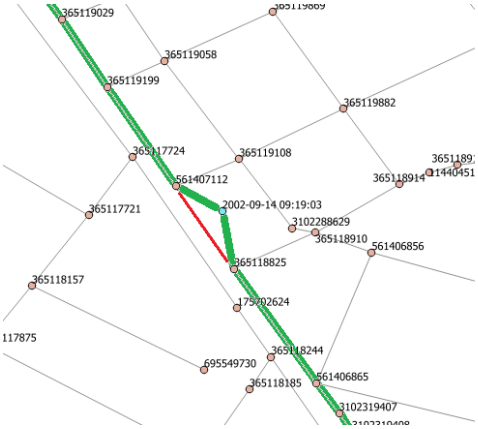
Από την Εικόνα 4.8 γίνεται εύκολα αντιληπτό ότι οι εγγραφές για τις οποίες το πεδίο osm_node_id μένει κενό είναι εγγραφές που αντιστοιχούν σε σημεία θέσης κινούμενων αντικειμένων από το αρχικό σύνολο δεδομένων, τα οποία ο συγκεκριμένος αλγόριθμος δεν αντιστοιχεί σε κάποιο κόμβο, όπως τα γαλάζια σημεία στην Εικόνα 4.10. Τα καφέ σημεία είναι κόμβοι που ανήκουν στο δίκτυο υποβάθρου. Αξίζει να σημειωθεί ότι ο αλγόριθμος εξήγαγε δύο ξεχωριστές ακμές στις οποίες ανήκει το segment, με σημείο διαχωρισμού τον κόμβο 135110959.

Πράγματι, στο σημείο αυτό οι ακμές του δρομολογίσιμου δικτύου διαχωρίζονται, αφού στον συγκεκριμένο κόμβο υπάρχει διακλάδωση του δικτύου, συνεπώς λειτουργεί σαν σημείο όπου μπορεί να πραγματοποιηθεί αλλαγή κατεύθυνσης της τροχιάς. Αν στο συγκεκριμένο σημείο δεν υπήρχε διακλάδωση του δρόμου, ο αλγόριθμος θα είχε επιστρέψει σαν αποτέλεσμα μόνο τον δρόμο με way_id=-1.

-Κενές τιμές κόμβων στον πίνακα osm_visited_segments

Όπως είδαμε στις προηγούμενες παραγράφους, δεν γίνεται ποτέ αντιστοίχιση των σημείων του αρχικού συνόλου δεδομένων επάνω στο χάρτη, αλλά πραγματοποιείται αντιστοίχιση του segment τροχιάς στους κόμβους του χάρτη, απ' όπου μπορούν να ανακληθούν οι ακμές στις οποίες αντιστοιχίζεται το segment μέσω των πινάκων του χαρτογραφικού υποβάθρου osm_nodes, osm_ways και osm_nodes_ways.

Η αντιμετώπιση αυτή οδηγεί σε μη αντιστοίχιση των θέσεων οχημάτων του συνόλου δεδομένων σε κόμβους του osm δικτύου, δημιουργώντας κενές τιμές στο πεδίο osm_node_id του πίνακα osm_visited_segments. Οι κενές αυτές τιμές εμφανίζονται σε δύο συνεχόμενες εγγραφές του osm_visited_segments σε κάθε αλλαγή segment (Εικόνα 4.11), αλλά μπορούν να εμφανιστούν σε πολλές συνεχόμενες θέσεις του osm_visited_segments όταν υπάρχει πολύ πυκνή δειγματοληψία[1], δηλαδή πολλαπλές θέσεις του οχήματος σε μία ακμή δικτύου (Εικόνα 4.12).



	id [PK] serial	traj_id integer	trip_id integer	segment_id integer	osm_node_id bigint	osm_way_id bigint	order_id integer
71	71	862	1	106	3102319410	305450529	3
72	72	862	1	106	3102287226	305450529	3
73	73	862	1	106	175702423	305450529	3
74	74	862	1	106	3102319408	305450529	3
75	75	862	1	106	3102319407	305450529	3
76	76	862	1	106	561406865	305450529	3
77	77	862	1	106	561406865	305450529	4
78	78	862	1	106	365118825	305450529	4
79	79	862	1	106	365118825	305450529	5
80	80	862	1	106		305450529	5
81	81	862	1	107		305450529	1
82	82	862	1	107	561407112	305450529	1
83	83	862	1	107	561407112	305450529	2
84	84	862	1	107	365119199	305450529	2
85	85	862	1	107	365119199	305450529	3
86	86	862	1	107	365119029	305450529	3
87	87	862	1	107	365119029	305450529	4
88	88	862	1	107	3102319409	305450529	4

Εικόνα 4.11: Κενές εγγραφές osm_node_id σε αλλαγή segment τροχιάς. Με κόκκινο σημειώνεται το segment τροχιάς που συντίθεται από την map_matching_strict()

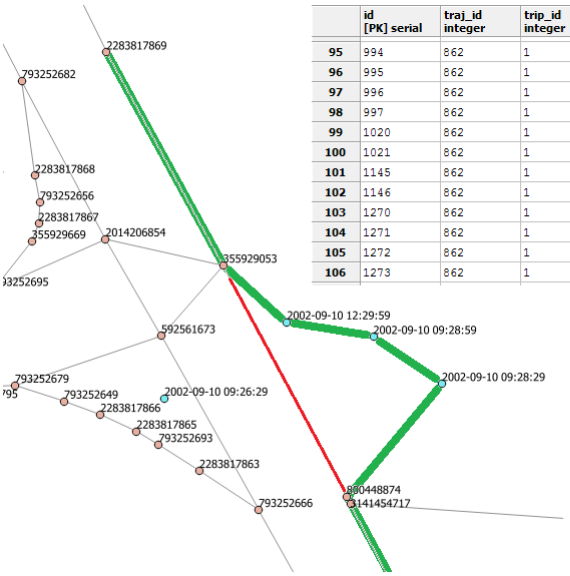
Το ερώτημα που προκύπτει είναι κατά πόσο οι κενές τιμές στο πεδίο osm_node_id του πίνακα osm_visited_segments επηρεάζουν τη διαδικασία αντιστοίχισης;

Η απάντηση σε αυτό το ερώτημα είναι ότι είναι προτιμότερο οι κενές τιμές να μην ληφθούν καθόλου υπόψη παρά να υποτεθεί μια εσφαλμένη αντιστοίχιση με κόμβο η οποία θα επηρεάσει αρνητικά τα αποτελέσματα του map matching. Η αντιμετώπιση αυτή δεν επηρεάζει την ακρίβεια των τροχιών, δεδομένου ότι και στις δύο περιπτώσεις (Εικόνα 4.11, Εικόνα 4.12) δεν υπάρχει

Σημείωση: Με τον αδόκιμο όρο «πυκνή δειγματοληψία» εννοούμε την ύπαρξη πολλών θέσεων του οχήματος σε μία ακμή δικτύου, δεδομένου ότι η συνήθης περίπτωση είναι η μεσολάβηση τουλάχιστον μίας ακμής δικτύου μεταξύ δύο θέσεων.

απώλεια πληροφορίας για την τροχιά διότι οι τιμές osm κόμβων που έπονται ή ακολουθούν τις κενές εγγραφές μπορούν να εξεταστούν ως προς τη σύνδεσή τους και η τροχιά να ανασυντεθεί βάσει αυτών, παραλείποντας εξ' ολοκλήρου τις κενές εγγραφές.

Η pl/pgsql συνάρτηση map_matching_strict() αναλαμβάνει να εντοπίσει την ακμή ή το συντομότερο μονοπάτι δρομολογήσιμου δικτύου μεταξύ δύο διαδοχικών κόμβων του πίνακα osm_visited_segments και να εισάγει τις σωστές αντιστοιχίσεις στον πίνακα map_matching. Σε περίπτωση που δεν βρεθεί σύνδεση μεταξύ των κόμβων η map_matching_strict() εισάγει μια μηδενική τιμή στον πίνακα map_matching, υποδεικνύοντας ότι δε βρέθηκε σύνδεση μεταξύ των κόμβων.

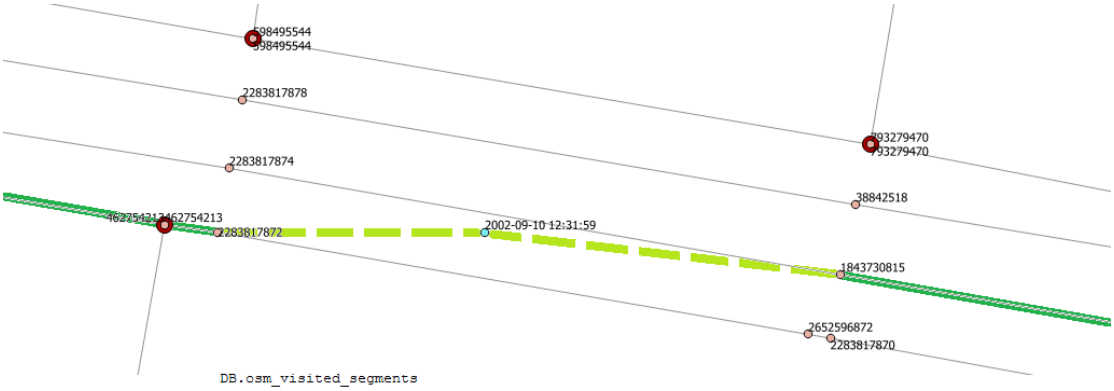


	id [PK] serial	traj_id integer	trip_id integer	segment_id integer	osm_node_id bigint	osm_way_id bigint	order_id integer
95	994	862	1	21	3141454717	46874467	1
96	995	862	1	21	3141454717	46874467	2
97	996	862	1	21	800448874	46874467	2
98	997	862	1	21		46874467	2
99	1020	862	1	22		46874467	1
100	1021	862	1	22		46874467	1
101	1145	862	1	23		46874467	1
102	1146	862	1	23		46874467	1
103	1270	862	1	24		46874467	1
104	1271	862	1	24	355929053	46874467	1
105	1272	862	1	24	355929053	46874467	2
106	1273	862	1	24	2283817869	46874467	2

Εικόνα 4.12: Δημιουργία συνεχόμενων κενών εγγραφών από πυκνή δειγματοληψία. Με κόκκινο σημειώνεται το segment τροχιάς που συντίθεται από την map_matching_strict()

Η μη ύπαρξη σύνδεσης μεταξύ των κόμβων μπορεί να προκληθεί από την αντιστοίχιση των σημείων που έπονται ή προηγούνται των κενών εγγραφών σε διαφορετικό osm δρόμο και σε κόμβους που δεν αντιστοιχούν σε κόμβους δρομολογήσιμου δικτύου. Αυτό συμβαίνει λόγω της αναντιστοιχίας μεταξύ των κόμβων του αρχικού (osm) δικτύου (οι οποίοι παρατηρούνται *και* εντός μιας ακμής δρομολογήσιμου δικτύου) και των κόμβων του δρομολογήσιμου δικτύου (οι οποίοι απαντώνται *μόνο* στην αρχή και το πέρας κάθε ακμής δικτύου). Στην περίπτωση αυτή, δεν βρίσκεται αντιστοίχιση κόμβου δρομολογήσιμου δικτύου και δεν μπορεί να συντεθεί το κομμάτι τροχιάς, οπότε η τρέχουσα τροχιά διακόπτεται και δημιουργείται μία νέα.

Η περίπτωση αδυναμίας σύνδεσης παρουσιάζεται στην Εικόνα 4.13. Οι «δρομολογήσιμοι» κόμβοι παρουσιάζονται με έντονο καφέ χρώμα, ενώ οι osm με ανοιχτό καφέ χρώμα. Αρχικά ελέγχεται αν οι osm κόμβοι 2283817872 και 1843730815 ανήκουν στο ίδιο osm_way. Αυτό δεν ισχύει, οπότε δοκιμάζεται η σύνδεσή τους με pgrouting. Η σύνδεση δεν είναι εφικτή, διότι οι κόμβοι αυτοί δεν αντιστοιχούν σε κόμβο του δικτύου pgrouting (δρομολογήσιμου). Συνεπώς δεν δύναται να βρεθεί σύνδεση μεταξύ τους, και αυτή η απώλεια σύνδεσης σημειώνεται στον πίνακα map_matching με 0.



DB.osm_visited_segments

	id [PK] serial	traj_id integer	trip_id integer	segment_id integer	osm_node_id bigint	osm_way_id bigint	order_id integer
123	1520	862	1	27	2283817875	217614901	2
124	1521	862	1	27	462754212	217614901	2
125	1522	862	1	27	462754212	217614901	3
126	1523	862	1	27	2283817873	217614901	3
127	1524	862	1	27	462754213	217614901	3
128	1525	862	1	27	462754213	217614901	4
129	1526	862	1	27	2283817872	217614901	4
130	1527	862	1	27		217614901	4
131	1663	862	1	28		46874129	1
132	1664	862	1	28	1843730815	46874129	1
133	1665	862	1	28	800448749	46874129	1
134	1666	862	1	28	462754220	46874129	1

map_matching_strict()

DB.map_matching

	id serial	traj_id integer	trip_id integer	segment_id integer	order_id integer	routable_osn integer
53	53	862	1	27	3	-194150
54	54	862	1	27	3	-194150
55	55	862	1	27	4	-194151
56	56	862	1	28	1	0
57	57	862	1	28	1	-64538
58	58	862	1	28	1	-64538

create_visited_segments()

DB.visited_segments

	id serial	traj_id integer	traj_trip_id integer	traj_segment integer	trip_id integer	segment_id integer	routable_osn integer	time_enter timestamp w	time_leave timestamp w	s_hash double precis	s_length double precis
10	33	862	1	27	5	17	-150942	2002-09-10	2002-09-10	104.9166446	1283.565475
11	34	862	1	27	5	18	-194149	2002-09-10	2002-09-10	111.3424000	1379.073836
12	35	862	1	27	5	19	-194150	2002-09-10	2002-09-10	117.7681563	1434.063738
13	36	862	1	27	5	20	-194151	2002-09-10	2002-09-10	124.1939134	1556.023771
14	37	862	1	28	6	1	-64538	2002-09-10	2002-09-10	5.908656022	591.5106716
15	38	862	1	29	6	2	-194164	2002-09-10	2002-09-10	12.33444236	622.6601321
16	39	862	1	29	6	3	-194165	2002-09-10	2002-09-10	18.76023555	664.8507117

Εικόνα 4.13: Περίπτωση διακοπής τροχιάς λόγω μη εφικτής σύνδεσης διαδοχικών κόμβων.

4.3 Κατασκευή ευρετηρίου NETTRA

Με την ολοκλήρωση εκτέλεσης της map_matching_strict(), ο πίνακας map_matching περιέχει όλες τις αντιστοιχίσεις segments σε ακμές του δικτύου. Μετέπειτα καλείται η συνάρτηση create_visited_segments(), η οποία συνδέει τα δεδομένα από τον πίνακα map_matching για να συνθέσει ακολουθίες ακμών οι οποίες θα αναπαριστούν τις αντιστοιχισμένες στο δίκτυο τροχιές των κινούμενων αντικειμένων.

Η συνάρτηση create_visited_segments() «σπάει» τις τροχιές των κινούμενων αντικειμένων όπου βρει μηδενική τιμή στο πεδίο routable_osm_id του πίνακα map_matching (Εικόνα 4.13). Αυτό συμβαίνει γιατί, όπως αναφέρθηκε στην προηγούμενη παράγραφο, ενδέχεται να μην υπάρχει σύνδεση μεταξύ δύο διαδοχικών ακμών ή ο κόμβος που έχει εντοπιστεί από το osm αρχείο αντιστοίχισης να μην αντιστοιχεί σε κόμβο του

osm_routable_network_vertices_ pgr, συνεπώς να μην μπορεί να εκτελεστεί δρομολόγηση μεταξύ των διαδοχικών κόμβων.

Με την ολοκλήρωση εκτέλεσης της create_visited_segments() καλούνται οι συναρτήσεις add_timestamps() και assign_weights_to_routable_segments() για την συμπλήρωση και των υπολοίπων στηλών του visited_segments, δηλαδή των χρονικών στιγμών εισόδου και εξόδου της τροχιάς από κάθε ακμή του δικτύου και των αθροιστικών βαρών μήκους και λογαρίθμου πρώτου φυσικού αριθμού. Με την ολοκλήρωση της εκτέλεσης των συναρτήσεων δημιουργείται ευρετήριο b-tree στις στήλες routable_osm_id, time_enter, traj_id, trip_id, time_leave, s_hash του πίνακα visited_segments, για ταχύτερη εκτέλεση των SPQ ερωτημάτων. Η δομή του πίνακα visited_segments, που υλοποιεί τη δομή NETTRA παρουσιάζεται στην Εικόνα 4.14.

	id serial	traj_id integer	traj_trip_id integer	traj_segment integer	trip_id integer	segment_id integer	routable_osm integer	time_enter timestamp w	time_leave timestamp w	s_hash double precis	s_length double precis
1	24	862	1	21	5	8	-228564	2002-09-10	2002-09-10	50.64192671	426.7623594
2	25	862	1	21	5	9	-228565	2002-09-10	2002-09-10	57.14410889	637.0823307
3	26	862	1	21	5	10	-80071	2002-09-10	2002-09-10	63.15421838	649.3596452
4	27	862	1	21	5	11	-64600	2002-09-10	2002-09-10	69.06330723	723.8552301
5	28	862	1	21	5	12	-64601	2002-09-10	2002-09-10	74.97240680	865.2280173
6	29	862	1	21	5	13	-64602	2002-09-10	2002-09-10	80.88151172	1032.168330
7	30	862	1	21	5	14	-64603	2002-09-10	2002-09-10	86.79061985	1135.927834
8	31	862	1	21	5	15	-64604	2002-09-10	2002-09-10	92.69973227	1173.170542
9	32	862	1	24	5	16	-64605	2002-09-10	2002-09-10	98.60885111	1251.545323
10	33	862	1	27	5	17	-150942	2002-09-10	2002-09-10	104.9166446	1283.565475
11	34	862	1	27	5	18	-194149	2002-09-10	2002-09-10	111.3424000	1379.073836
12	35	862	1	27	5	19	-194150	2002-09-10	2002-09-10	117.7681563	1434.063738
13	36	862	1	27	5	20	-194151	2002-09-10	2002-09-10	124.1939134	1556.023771
14	37	862	1	28	6	1	-64538	2002-09-10	2002-09-10	5.908656022	591.5106716
15	38	862	1	29	6	2	-194164	2002-09-10	2002-09-10	12.33444236	622.6601321
16	39	862	1	29	6	3	-194165	2002-09-10	2002-09-10	18.76023555	664.8507117
17	40	862	1	29	6	4	-194166	2002-09-10	2002-09-10	25.18603069	732.1095569
18	41	862	1	29	6	5	-194167	2002-09-10	2002-09-10	31.61183007	754.5369843
19	42	862	1	29	6	6	-194168	2002-09-10	2002-09-10	38.03763597	808.7047159
20	43	862	1	29	6	7	-194217	2002-09-10	2002-09-10	44.46356599	842.3920468
21	44	862	1	29	6	8	-230015	2002-09-10	2002-09-10	50.96866046	904.6395153

Εικόνα 4.14: Πίνακας visited_segments (δομή NETTRA)

4.4 Ερωτήματα SPQ

Στο παρόν υποκεφάλαιο παρουσιάζονται συνοπτικά οι τεχνικές που ακολουθήθηκαν για την υλοποίηση των ερωτημάτων SPQ και των συναρτήσεων που προτείνονται στο [16] για βελτίωση της απόδοσής τους. Ο πλήρης κώδικας των συναρτήσεων είναι διαθέσιμος με τα αρχεία που συνοδεύουν την εργασία.

4.4.1 Προσεγγιστικό SPQ

Το προσεγγιστικό SPQ, που περιγράφεται από το Ερώτημα 2.1 του κεφαλαίου 2, υλοποιείται από την pl/pgsql συνάρτηση spq1(bigint, bigint), η οποία λειτουργεί σαν wrapper για το Ερώτημα 2.1, δηλαδή δεχόμενη μόνο τα osm_node_ids για τον πρώτο και τον τελευταίο κόμβο αναζητά τη συντομότερη διαδρομή μεταξύ των δύο κόμβων, υπολογίζει το deltahash μεταξύ της αρχικής και της τελικής ακμής και εκτελεί το Ερώτημα 2.1 που αντιστοιχεί στη διαδρομή.

Αξίζει να σημειωθεί ότι λόγω στρογγυλοποιήσεων που γίνονται από το σύστημα κρίθηκε απαραίτητη η εισαγωγή μικρού ποσοστού ανοχής για την τιμή του deltahash, γίνεται δηλαδή αναζήτηση για

$$e_{end}.hash - e_{start}.hash = deltahash(\pi) \pm 0.00000001$$

Σημείωση: Με τον όρο wrapper εννοείται η συνάρτηση που δημιουργείται για να «καλύψει» τις λεπτομέρειες κατασκευής και εκτέλεσης ενός πολύπλοκου SQL ερωτήματος.

η οποία είναι αρκετά μικρή ανοχή ώστε να προκύψει σφάλμα σε μετρήσεις με χρήση του μήκους της τροχιάς ως βάρος.

Με την κλήση της συνάρτησης πραγματοποιείται αντιστοιχία των επιλεγμένων κόμβων με τους κόμβους που έχουν οριστεί από το pgRouting για τη λειτουργία της δρομολόγησης. Μετέπειτα καλείται ο αλγόριθμος pgrdijkstra του pgRouting, μια υλοποίηση αντίστοιχη του αλγορίθμου Dijkstra για την εύρεση του συντομότερου δικτυακού μονοπατιού μεταξύ δύο κόμβων δικτύου. Ο αλγόριθμος επιστρέφει τη συντομότερη διαδρομή μεταξύ των δύο κόμβων η οποία ορίζεται ως η διαδρομή π για το SPQ και υπολογίζεται η τιμή deltahash της. Βάσει των παραμέτρων deltahash και του αναγνωριστικού της αρχικής και της τελικής ακμής της π, κατασκευάζεται και εκτελείται το SPQ Ερωτήματος 2.1 για τη διαδρομή π.

Η συνάρτηση επιστρέφει ως αποτέλεσμα έναν πίνακα με τα αναγνωριστικά των τροχιών που ακολουθούν επακριβώς την διαδρομή π. Ένα παράδειγμα κλήσης της συγκεκριμένης συνάρτησης είναι το εξής:

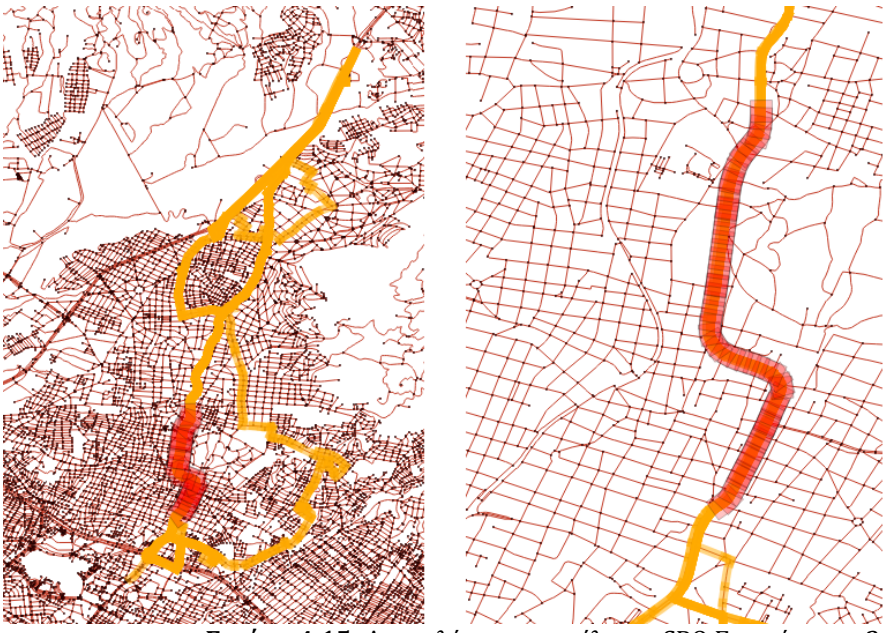
```
select * from spq1 (1753749971, 279591283);
```

το οποίο εκτελεί το ερώτημα που παρουσιάζεται στο ερώτημα 4.1 και επιστρέφει τα αποτελέσματα που απεικονίζονται στην Εικόνα 4.15.

Η διαδρομή του SPQ σημειώνεται με κόκκινη πλατιά γραμμή ενώ οι τροχιές που ακολουθούν τη διαδρομή του SPQ σημειώνονται με λεπτότερες κίτρινες γραμμές.

```
select e_start.traj_id, e_start.trip_id
from visited_segments e_start
join visited_segments e_end on e_start.traj_id=e_end.traj_id
                           and e_start.trip_id=e_end.trip_id
where e_start.routable_osm_id=-192930
and e_end.routable_osm_id=-92329
and e_end.s_length-e_start.s_length>2134.61046968405-0.00000001
and e_end.s_length-e_start.s_length<2134.61046968405+0.00000001
```

Ερώτημα 4.1: Παράδειγμα προσεγγιστικού SPQ



	traj_id_ integer	trip_id_ integer
1	869	11
2	876	6
3	876	14
4	882	75
5	896	15
6	897	83
7	899	30
8	911	9
9	912	84
10	913	61
11	913	113
12	914	51
13	914	58

Εικόνα 4.15: Αποτελέσματα εκτέλεσης SPQ Ερωτήματος 2.1

4.4.2 Ακριβές SPQ

Ομοίως, το ακριβές SPQ υλοποιείται μέσω μιας pl/pgsql συνάρτησης που λειτουργεί ως wrapper για τη δημιουργία και την εκτέλεση του ερωτήματος ακριβούς τροχιάς.

Στην συνάρτηση δίνονται επίσης ως παράμετροι δύο κόμβοι του OSM δικτύου υποβάθρου. Εντοπίζονται οι κόμβοι δρομολογήσιμου δικτύου που αντιστοιχούν στους δύο αυτούς κόμβους και εκτελείται δρομολόγηση για εύρεση διαδρομής μεταξύ των κόμβων εντός του δικτύου με τον αλγόριθμο pgr_dijkstra. Ο αλγόριθμος βρίσκει τη συντομότερη διαδρομή μεταξύ των δύο κόμβων και ορίζει την διαδρομή αυτή ως διαδρομή π του SPQ.

Για κάθε ακμή της διαδρομής π εισάγονται στο ερώτημα SPQ (στα πρότυπα του Ερωτήματος 2.2) οι αντίστοιχες συνθήκες για id ακμής και deltahash ενώ πραγματοποιείται και η κατάλληλη αυτό-σύνδεση με τον πίνακα visited_segments. Για την τελευταία ακμή της π πραγματοποιείται και έλεγχος ισότητας με το συνολικό deltahash της διαδρομής.

Η συνάρτηση επιστρέφει έναν πίνακα που περιέχει όλες τις τροχιές που ακολουθούν επακριβώς την διαδρομή π. Και σε αυτή την περίπτωση εισάγουμε μια ανοχή της τάξης του 0,000000001 η οποία σαν απόσταση είναι αμελητέα, αλλά σαν μέγεθος αρκετή για την αποφυγή της αποτροπής σωστών αποτελεσμάτων λόγω στρογγυλοποίησης του hash αθροίσματος.

Η εκτέλεση της εντολής

```
select * from spq2 (1753749971, 279591283)
```

συνεπάγεται την εκτέλεση ερωτήματος αρκετά μεγάλου σε μέγεθος για την αναφορά του εντός του κειμένου της εργασίας.

Τα αποτελέσματα που εξάγονται από το εν λόγω ερώτημα είναι ακριβώς τα ίδια με τα αποτελέσματα της Εικόνας 4.15, λόγω του ότι η διαδρομή π που ακολουθούν και τα δύο SPQs είναι μοναδική και συντομότερη, συνεπώς, όπως αναφέρθηκε αναλυτικά στην παράγραφο 2.6, τα δύο ερωτήματα είναι ισοδύναμα, αν και διαφορετικά σε ταχύτητα εκτέλεσης.

4.4.3 Unique SubPaths

Για την βελτίωση της απόδοσης του ακριβούς ερωτήματος SPQ, χρησιμοποιείται ο αλγόριθμος UniqueSubPaths, ο οποίος επίσης υλοποιείται από μια σειρά συναρτήσεων υλοποιημένες σε pl/pgsql.

Για την κλήση του αλγορίθμου καλείται η συνάρτηση call_unique_subpaths, η οποία δέχεται σαν παραμέτρους τα ids δύο osm κόμβων που ορίζουν την αρχή και το πέρας της διαδρομής π. Η call_unique_subpaths εντοπίζει τους κόμβους δρομολογήσιμου δικτύου που αντιστοιχούν στα osm_ids που δόθηκαν ως παράμετροι, δημιουργεί έναν προσωρινό πίνακα όπου θα αποθηκευτούν οι υπο-διαδρομές της π με το deltahash μεταξύ πρώτης και τελευταίας ακμής της καθεμιάς και καλεί την συνάρτηση unique_shortest_paths με παραμέτρους τους δύο κόμβους του δρομολογήσιμου δικτύου. Η unique_shortest_paths λειτουργεί αναδρομικά καλώντας τον εαυτό της κάθε φορά που βρίσκει σημείο όπου μπορεί να τμηματοποιηθεί η διαδρομή π, υλοποιώντας τη λειτουργία του UniqueSubPaths που περιγράφηκε στην ενότητα 2.6 του παρόντος τόμου.

Με το πέρας των αναδρομικών εκτελέσεων της unique_shortest_paths, ο πίνακας unique_subpaths περιέχει τις μοναδικές υπο-διαδρομές της π καθώς και

την τιμή deltahash μεταξύ των αρχικών και τελικών ακμών τους. Η συνάρτηση spq2_from_usp() αναλαμβάνει τότε τη δημιουργία και εκτέλεση ενός SPQ Ερωτήματος 2.2, όπου εκτελείται για 1≤x≤n-1 υπο-διαδρομές μήκους 2≤x≤n αντί για n-1 υποδιαδρομές μήκους 2.

Τα αποτελέσματα της Εικόνας 4.16 προέκυψαν από την κλήση της συνάρτησης με την εντολή

```
select * from call_unique_subpaths(801467980, 985946206).
```

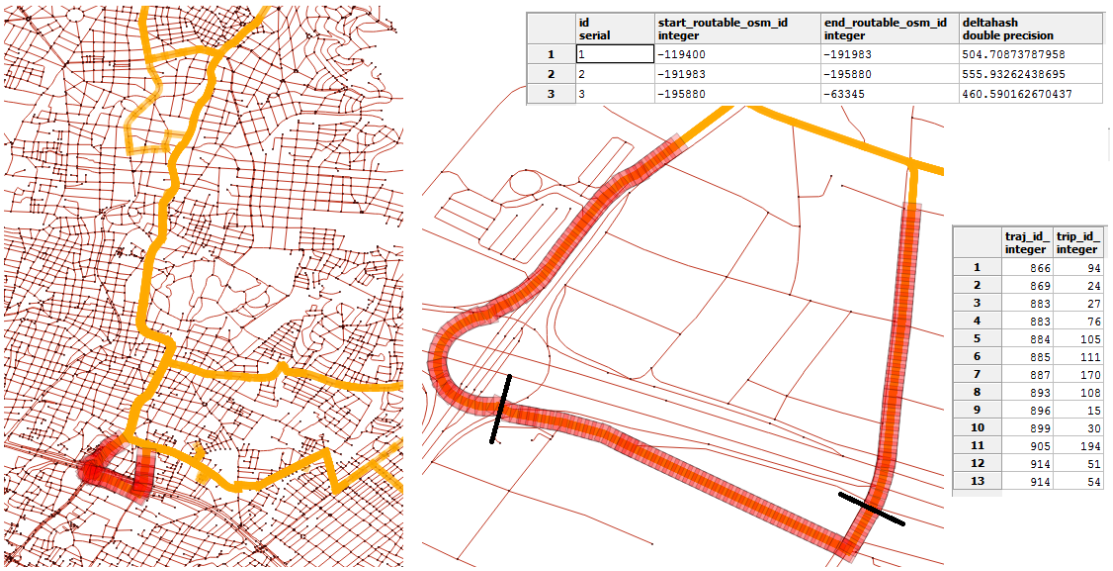
Λόγω της χρήσης αλγορίθμου συντομότερης διαδρομής (pgr_dijkstra) για τον καθορισμό της διαδρομής μεταξύ του αρχικού και του τελικού κόμβου (801467980, 985946206), η διαδρομή που δημιουργείται (σημειώνεται με κόκκινο χρώμα στο σχήμα 4.16) είναι η συντομότερη δυνατή, βάσει των περιορισμών του δικτύου. Συνεπώς, η κλήση του UniqueSubPaths με τους ίδιους περιορισμούς δικτύου, δεν θα «σπάσει» περαιτέρω την τροχιά σε μοναδικές τροχιές, αφού και ο ίδιος εσωτερικά χρησιμοποιεί την συνάρτηση pgr_dijkstra. Για τον έλεγχο της λειτουργίας του αλγορίθμου και την εξαγωγή των αποτελεσμάτων της Εικόνας 4.16 εκτελέστηκε ο UniqueSubPaths χωρίς περιορισμό κατεύθυνσης κίνησης εντός των ακμών του δικτύου, θεωρώντας δηλαδή όλες τις ακμές σαν δρόμους διπλής κυκλοφορίας.

Στα σημεία της διαδρομής όπου βρέθηκε εναλλακτική συντομότερη υπο-διαδρομή που δεν ακολουθούσε την αρχική (σημειωμένη με κόκκινο χρώμα στην Εικόνα 4.16), η αρχική διαδρομή «έσπασε» σε δύο υπο-διαδρομές, των οποίων ο αρχικός, ο τελικός κόμβος και το deltahash σημειώνονται σε αντίστοιχο πίνακα (πάνω μέρος της εικόνας), ο οποίος μετέπειτα θα χρησιμοποιηθεί ως αναφορά για την εκτέλεση των προσεγγιστικών ερωτημάτων ακριβούς διαδρομής για κάθε μία υπο-διαδρομή. Τα σημεία τμηματοποίησης της αρχικής διαδρομής σημειώνονται με μαύρες ενδείξεις.

Όπως σημειώνεται στην Εικόνα 4.10, η αρχική διαδρομή τμηματοποιείται σε τρεις υπο-διαδρομές:

1. Υπο-διαδρομή 1 με id αρχικής ακμής -119400, id τελικής ακμής -191983 και deltahash ίσο με 504,70873787958
2. Υπο-διαδρομή 2 με id αρχικής ακμής -191983, id τελικής ακμής -195880 και deltahash ίσο με 555,93262438675
3. Υπο-διαδρομή 1 με id αρχικής ακμής -195880, id τελικής ακμής -63345 και deltahash ίσο με 460,590162670437

Για τις υπο-διαδρομές αυτές θα εκτελεστεί ένα SPQ ερώτημα ακριβούς προσέγγισης, με τα στοιχεία κάθε μίας από τις υπο-διαδρομές, όπως υπολογίστηκαν από τον αλγόριθμο UniqueSubPaths. Με αυτόν τον τρόπο, εκτελείται ερώτημα ακριβούς προσέγγισης μόνο για τρεις υπο-διαδρομές, ενώ με την κλασσική ακριβή SPQ προσέγγιση ο αριθμός των υπο-διαδρομών θα ήταν σημαντικά μεγαλύτερος (στην προκειμένη περίπτωση θα είχαμε 15 υπο-διαδρομές μήκους δύο ακμών) αυξάνοντας κατά πολύ τον χρόνο εκτέλεσης των ερωτημάτων. Εσωτερικά, το SQL ερώτημα που εκτελείται είναι της μορφής του Ερωτήματος 4.2.



	id serial	start_routable_osm_id integer	end_routable_osm_id integer	deltahash double precision
1	1	-119400	-191983	504.70873787958
2	2	-191983	-195880	555.93262438695
3	3	-195880	-63345	460.590162670437

	traj_id_ integer	trip_id_ integer
1	866	94
2	869	24
3	883	27
4	883	76
5	884	105
6	885	111
7	887	170
8	893	108
9	896	15
10	899	30
11	905	194
12	914	51
13	914	54

Εικόνα 4.16: Παράδειγμα εκτέλεσης UniqueSubPaths. Στο επάνω μέρος της εικόνας διακρίνονται οι υπο-διαδρομές της διαδρομής π. Με μαύρο χρώμα σημειώνονται τα σημεία που «σπάει» η διαδρομή π.

Η απόδοση του ερωτήματος της βελτιωμένης προσέγγισης έναντι της ακριβούς είναι σημαντικά καλύτερη στις περιπτώσεις που η π δεν «σπάει» σε αριθμό υπο-διαδρομών κοντά στο n-1. Ωστόσο, η διαδικασία εντοπισμού των μοναδικών υπο-διαδρομών είναι αρκετά χρονοβόρα, δεδομένων των πολλών κλήσεων του pgr_dijkstra που πραγματοποιούνται για την εύρεση συντομότερης διαδρομής.

```
select e0.traj_id, e0.trip_id
from visited_segments e0
join visited_segments e1 on e1.traj_id=e0.traj_id
     and e1.trip_id=e0.trip_id
join visited_segments e2 on e2.traj_id=e1.traj_id
     and e2.trip_id=e1.trip_id
join visited_segments e3 on e3.traj_id=e2.traj_id
     and e3.trip_id=e2.trip_id
where e0.routable_osm_id=-119400
     and e1.routable_osm_id=-191983
     and e2.routable_osm_id=-195880
     and e3.routable_osm_id=-63345
     and e1.s_length>e0.s_length+504.70873786958
     and e1.s_length<e0.s_length+504.70873788958
     and e2.s_length>e1.s_length+555.93262437695
     and e2.s_length<e1.s_length+555.93262439695
     and e3.s_length>e2.s_length+460.590162660437
     and e3.s_length<e2.s_length+460.590162680437
     and e3.s_length>e0.s_length+1521.23152492697
     and e3.s_length<e0.s_length+1521.23152494697
```

Ερώτημα 4.2: Ακριβές SPQ για υπο-διαδρομές που υπολογίστηκαν μέσω του UniqueSubPaths.

4.4.4 «Πρακτική» λύση SPQ

Για την εξαγωγή ακριβούς αποτελέσματος αποδοτικά επιλέγεται η δημιουργία ενός ξεχωριστού βάρους για κάθε ακμή, διαδικασία που περιγράφεται θεωρητικά στην ενότητα 2.7 και ως προς την υλοποίηση στην παράγραφο 4.3 του παρόντος κεφαλαίου.

Για την εκτέλεση SPQ με χρήση των βαρών πρώτων αριθμών, έχει υλοποιηθεί η wrapper συνάρτηση spq3, η οποία είναι πανομοιότυπη με την spq1, με μόνη διαφορά τη χρήση διαφορετικού πεδίου βάρους (hash αντί του length) για τις ακμές του δρομολογήσιμου δικτύου και διαφορετικού πεδίου αθροιστικού βάρους (s_hash αντί του s_length). φυσικά και η ανοχή που δίδεται εντός του ερωτήματος για τις τιμές του deltahash είναι κατά πολύ μικρότερη, λόγω των μικρότερων μεγεθών των βαρών στα πεδία hash και s_hash. Συγκεκριμένα, βάσει των εξαγόμενων αποτελεσμάτων, βρέθηκε ότι ανοχή της τάξης του 0,000000000001 λίγο λιγότερα από τα αποτελέσματα με την ανοχή μεγέθους 0,00000001 για τα ερωτήματα που χρησιμοποιούν το μήκος ακμής ως πεδίο βάρους.

Στην περίπτωση της κλήσης ενός ερωτήματος αντίστοιχου της Εικόνας 4.15 με την εντολή `select * from spq3 (1753749971, 279591283)` εξάγο-νται τα αποτελέσματα που παρουσιάζονται στην Εικόνα 4.17. Με κόκκινο χρώμα σημειώνεται η διαδρομή και με πορτοκαλί οι διαδρομές που την ακολουθούν, τα αναγνωριστικά των οποίων φαίνονται στον πίνακα που βρίσκεται στα δεξιά της εικόνας. Η απόδοση της spq3 είναι αντίστοιχη της απόδοσης της spq1, μιας και η προσέγγιση διαφέρει μόνο ως προς το βάρος που χρησιμοποιείται για την κωδικοποίηση των ακμών του δικτύου. Το ερώτημα που εκτελείται για την εξαγωγή των αποτελεσμάτων βάσει αυτής της μεθόδου για την προκειμένη περίπτωση παρουσιάζεται στο Ερώτημα 4.3.

	traj_id_ integer	trip_id_ integer
1	869	11
2	876	6
3	876	14
4	882	75
5	896	15
6	897	83
7	899	30
8	911	9
9	912	84
10	913	113
11	914	58

Εικόνα 4.17: Αποτελέσματα SPQ με βάρος πρώτων αριθμών

```
select e_start.traj_id, e_start.trip_id
from visited_segments e_start
join visited_segments e_end on e_start.traj_id=e_end.traj_id
      and e_start.trip_id=e_end.trip_id
where e_start.routable_osm_id=-192930
      and e_end.routable_osm_id=-92329
      and e_end.s_hash-e_start.s_hash=108.897356466237
```

Ερώτημα 4.3: Εκτέλεση ερωτήματος «πρακτικής» προσέγγισης SPQ

4.4.5 Process SPQ

Για την υλοποίηση του ProcessSPQ δημιουργήθηκε η συνάρτηση process_spq, η οποία δέχεται σαν ορίσματα δύο ids OSM κόμβων του δικτύου, μία παράμετρο (αληθή/ψευδή) για την αναζήτηση ακριβούς τροχιάς και δύο χρονικές στιγμές, που μαζί ορίζουν ένα χρονικό διάστημα.

Παρ' όλο που ο αλγόριθμος ProcessSPQ παρουσιάζεται ως αναδρομικός στην ενότητα 2.8, η υλοποίηση επιλέχθηκε να μη γίνει αναδρομικά. Αυτό συνέβη για χάρη της απλότητας, μια και όπως αναφέρεται στην Ενότητα 2.8 (και στην ενότητα 5 του [16]), ο αλγόριθμος καλεί τον εαυτό του το πολύ μία φορά. Συνεπώς, η αναδρομή μπορεί να προσομοιωθεί με επανάληψη ορισμένων εντολών του αλγορίθμου στο σημείο όπου η συνάρτηση θα καλούσε τον εαυτό της. Στην περίπτωση που υπήρχε ενδεχόμενο η συνάρτηση να καλέσει τον εαυτό της περισσότερες από μία φορά, η υλοποίηση θα έπρεπε υποχρεωτικά να γίνει αναδρομικά.

Τα ενδεικτικά παραδείγματα εκτέλεσης του ProcessSPQ

```
select * from process_spq(1753749971, 279591283, true, '2000-01-01 00:00:00',
'2015-01-01 00:00:00')
```

και

```
select * from process_spq(1753749971, 279591283, false, '2000-01-01 00:00:00',
'2015-01-01 00:00:00')
```

εξάγουν 13 και 14 τροχιές αντίστοιχα σαν αποτέλεσμα (η πρώτη εκτέλεση είναι «αυστηρότερη», μιας κι εκτελεί ερώτημα ακριβούς προσέγγισης), ενώ οι χρόνοι εκτέλεσής τους διαφέρουν σημαντικά, μιας και στην πρώτη περίπτωση καλείται ο αλγόριθμος UniqueSubPaths για εύρεση των μοναδικών υπο-διαδρομών και εκτελείται ακριβές SPQ ερώτημα, ενώ στην δεύτερη περίπτωση εκτελείται απ' ευθείας προσεγγιστικό ερώτημα. Η διαφοροποίηση αυτή συμβαίνει επειδή η παράμετρος ακριβούς ερωτήματος είναι αληθής στην πρώτη περίπτωση, ενώ στη δεύτερη είναι ψευδής.

4.5 Επιδόσεις ερωτημάτων SPQ

Οι διορθώσεις για τον περιορισμό των λανθασμένων αποτελεσμάτων των SPQ σε μηδενικά επίπεδα καθιστούν το χρόνο κατασκευής και επεξεργασίας των ερωτημάτων SPQ τον κυρίαρχο παράγοντα καθορισμού απόδοσης των ερωτημάτων και κατ'επέκτασιν του ευρετηρίου NETTRA.

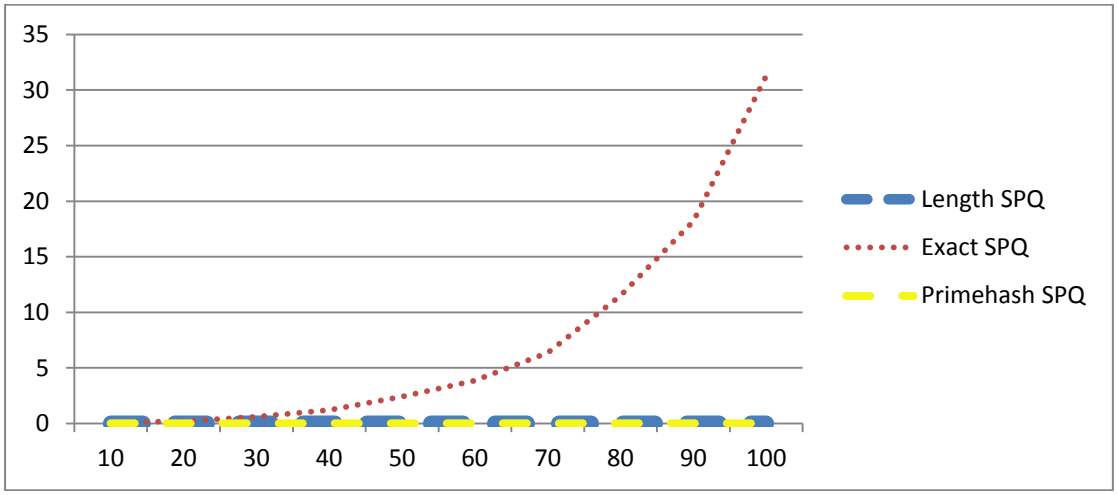
Ο χρόνος εκτέλεσης καθορίζεται σε μεγάλο βαθμό από τον αριθμό ακμών της διαδρομής π, τη μορφολογία του ερωτήματος και τον βαθμό κατακερματισμού της από τη συνάρτηση UniqueSubPaths. Στην παρούσα παράγραφο

παρουσιάζεται η επιρροή της μορφής του ερωτήματος και του αριθμού των ακμών στο χρόνο εκτέλεσης των ερωτημάτων κάθε κατηγορίας.

4.5.1 Απλά ερωτήματα SPQ

Ένα μεγάλο κομμάτι του χρόνου εκτέλεσης των wrapper συναρτήσεων καταλαμβάνει η προετοιμασία του ερωτήματος SPQ, η οποία πολλές φορές απαιτεί την διάσχιση μεγάλου κομματιού της διαδρομής π, στην χειρότερη περίπτωση με κλήση του αλγορίθμου pgr_dijkstra για κάθε ακμή της π, όπως γίνεται στον αλγόριθμο UniqueSubPaths (υπενθυμίζουμε ότι ο UniqueSubPaths χρησιμοποιείται και στη συνάρτηση exact ProcessSPQ).

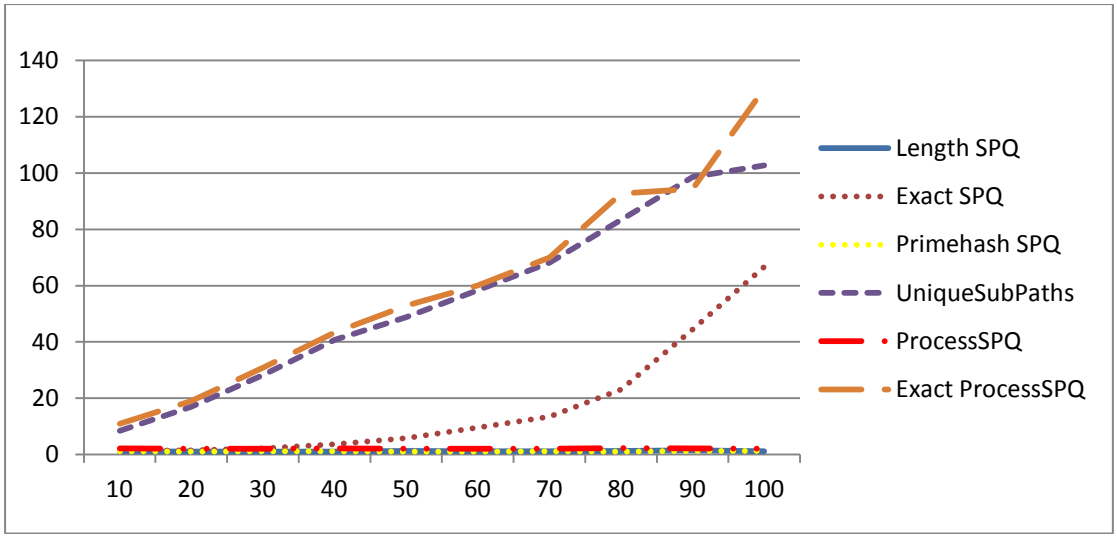
Στις περισσότερες περιπτώσεις, τα απλά ερωτήματα SQL που εκτελούνται δεν υπερβαίνουν τα 20 msec για τα προσεγγιστικά SPQ (Διάγραμμα 4.1) ανεξαρτήτως αριθμού ακμών της διαδρομής π. Ωστόσο, ο χρόνος εκτέλεσης του exact SPQ αυξάνεται εκθετικά όσο ο αριθμός των ακμών μεγαλώνει. Αυτή η εξέλιξη είναι αναμενόμενη, διότι σε ένα ερώτημα exact SPQ για διαδρομή n ακμών περιέχονται n-1 self joins του πίνακα visited_segments και 2n συνθήκες ισότητας (αναγνωριστικού ακμής και deltahash), ενώ στα προσεγγιστικά ερωτήματα ο αριθμός self joins στον visited_segments είναι ίσος με 1, και υπάρχουν μόνο τρεις συνθήκες ισότητας ανεξαρτήτως αριθμού ακμών.



Διάγραμμα 4.1: Χρόνοι (sec) εκτέλεσης SPQ ερωτημάτων βάσει προσέγγισης για 10-100 ακμές

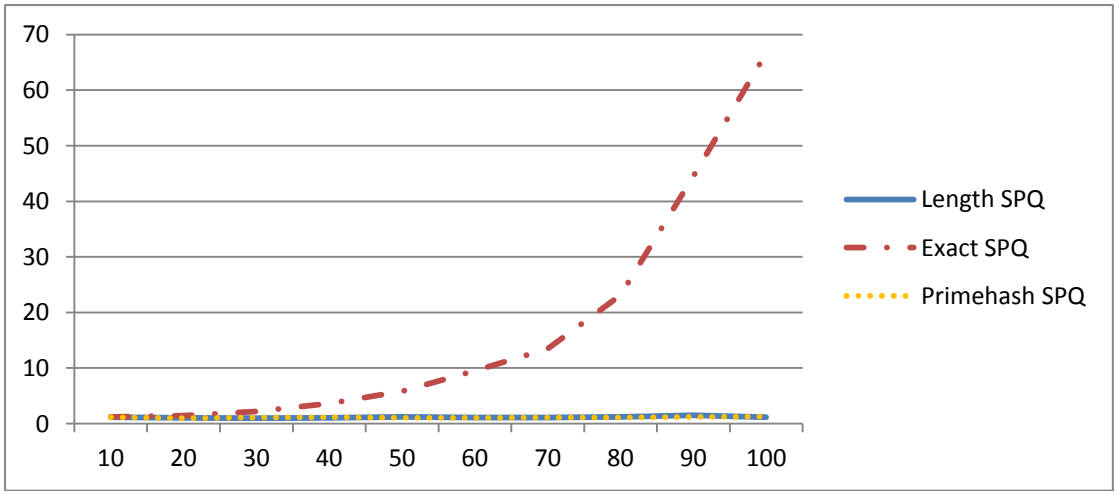
4.5.2 SPQ wrappers

Στο Διάγραμμα 4.2 παρουσιάζονται οι χρόνοι εκτέλεσης (σε sec) για τα SPQ wrappers που παρουσιάστηκαν στο παρόν κεφάλαιο. Όπως γίνεται εύκολα αντιληπτό, οι συναρτήσεις που χρησιμοποιούν προσεγγιστικά SPQ (Length SPQ, Primehash SPQ, ProcessSPQ) παραμένουν σταθερές στο χρόνο που χρειάζονται για να ολοκληρώσουν την εκτέλεσή τους για λόγους που εξηγήθηκαν στην παράγραφο 4.5.1, με τις Length SPQ και Primehash SPQ να είναι σταθερά ταχύτερες από την ProcessSPQ (Διάγραμμα 4.5).

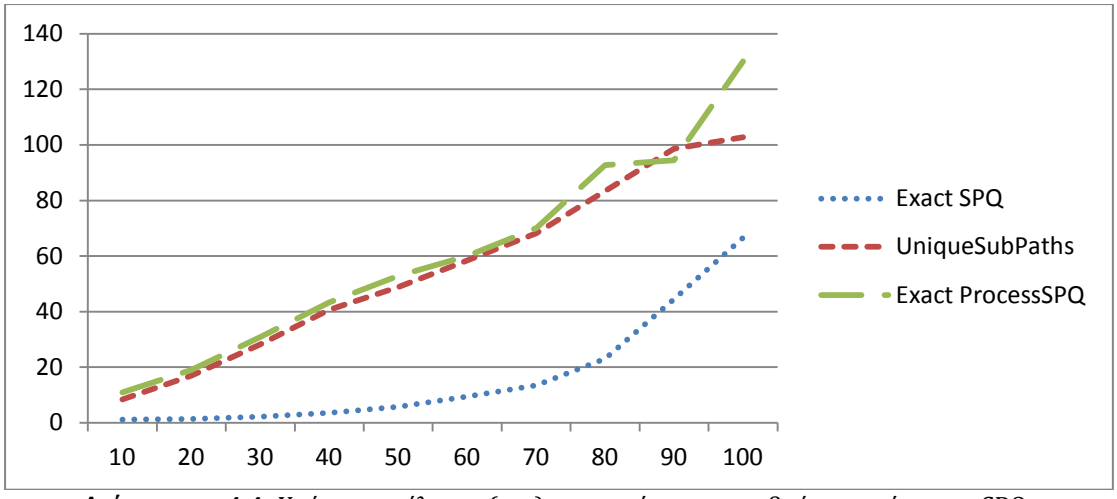


Διάγραμμα 4.2: Χρόνος εκτέλεσης (sec) wrapper συναρτήσεων για διαδρομή π 10-100 ακμών

Η συνάρτηση που υλοποιεί την προσέγγιση Exact SPQ (Ερώτημα 2.2) παρουσιάζει εκθετική αύξηση του χρόνου που απαιτείται για την εκτέλεσή της, όπως αναφέρθηκε και στην προηγούμενη παράγραφο, ενώ οι συναρτήσεις που χρησιμοποιούν τον αλγόριθμο UniqueSubPaths (τον χρησιμοποιεί και η Exact ProcessSPQ) παρουσιάζουν αύξηση του χρόνου εκτέλεσης που κινείται μεταξύ γραμμικής και εκθετικής τάξεως.

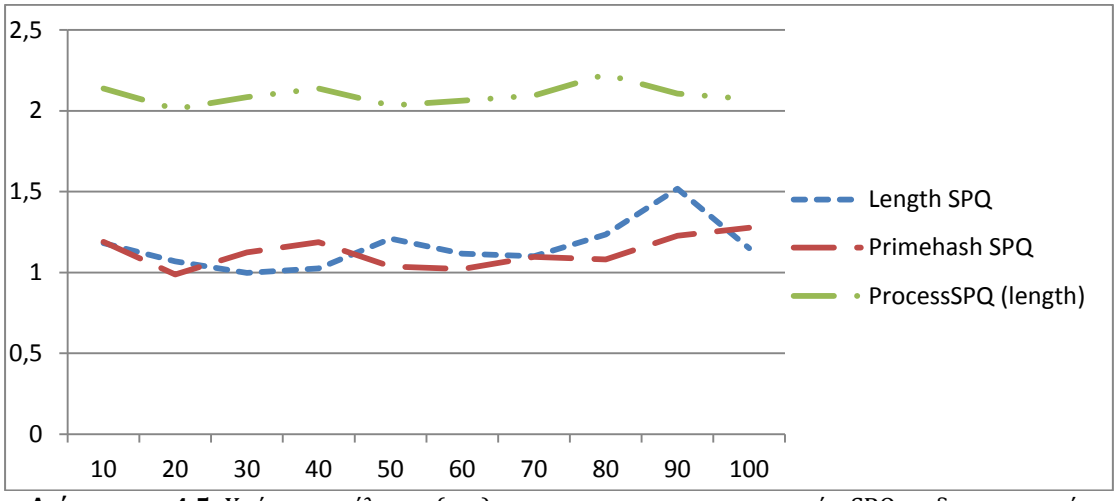


Διάγραμμα 4.3: Χρόνος εκτέλεσης (sec) προσεγγιστικών σε σχέση με την ακριβή προσέγγιση SPQ



Διάγραμμα 4.4: Χρόνος εκτέλεσης (sec) συναρτήσεων ακριβούς προσέγγισης SPQ

Η γραμμική αύξηση του χρόνου εκτέλεσης στον UniqueSubPaths μπορεί να εξηγηθεί από τον αριθμό κλήσεων της συνάρτησης pgr_dijkstra (η οποία αποτελεί και την κυριότερη τροχοπέδη στην απόδοση του αλγορίθμου), ο οποίος είναι ανάλογος των ακμών της διαδρομής π. Αν η διαδρομή δεν «σπάσει» σε περισσότερες από 30 υπο-διαδρομές κατά την εκτέλεση του UniqueSubPaths, δεν υπάρχει σημαντική αύξηση στον χρόνο εκτέλεσης του Exact SPQ ερωτήματος που ακολουθεί και συνεπώς όλου του wrapper. Αν ωστόσο προκύψουν πάνω από 30 υπο-τροχιές, αναμένεται σημαντική αύξηση του χρόνου εκτέλεσης του wrapper λόγω του χρόνου εκτέλεσης του Exact SPQ που έπεται του αλγορίθμου UniqueSubPaths.



Διάγραμμα 4.5: Χρόνος εκτέλεσης (sec) για wrappers προσεγγιστικών SPQ με διαφορετικά βάρη

5. Συμπεράσματα και προτάσεις βελτίωσης

5.1 Συμπεράσματα

Τα ερωτήματα αυστηρής τροχιάς (Strict Path Queries), μπορούν να αποτελέσουν ιδιαίτερα χρήσιμο εργαλείο σε αναλυτές, δεδομένης της ικανότητάς τους να εξάγουν ως αποτελέσματα τροχιές του συνόλου δεδομένων που ακολουθούν συγκεκριμένη διαδρομή χωρίς να παρεκκλίνουν καθόλου από αυτή. Η χρήση τους για παρακολούθηση της κίνησης σε μεγάλους οδικούς κόμβους και άξονες μπορεί να εξάγει χρήσιμες πληροφορίες για την μοντελοποίηση της κίνησης στην περιοχή, την παρακολούθηση του φόρτου των οδικών αξόνων και την εξαγωγή τάσεων στην κίνηση των οχημάτων εντός του αστικού οδικού δικτύου.

Με τις μεθόδους και τεχνικές που παρουσιάστηκαν πραγματοποιήθηκε φόρτωση χαρτογραφικού υποβάθρου και δημιουργία τοπολογίας δικτύου, ενώ πραγματοποιήθηκε επίσης αντιστοίχιση δεδομένων θέσης GPS σε ακμές του δημιουργηθέντος δικτύου για τη δημιουργία κατάλληλου ευρετηρίου υποστήριξης SPQ.

Οι συναρτήσεις–wrappers που υλοποιήθηκαν επιτρέπουν στον χρήστη την εκτέλεση ερωτημάτων αυστηρούς τροχιάς σε δίκτυο οδών με ιδιαίτερα απλό τρόπο (δίνοντας τα ids δύο σημείων). Οι προσεγγίσεις που παρουσιάστηκαν δεν δίνουν απλά την ευκαιρία στο χρήστη να επιλέξει ανάμεσα σε ακρίβεια (Exact SPQ, UniqueSubPaths) και ταχύτητα (Length SPQ), αλλά και να συνδυάσει ακρίβεια και ταχύτητα (Primehash SPQ). Ειδικά οι προσεγγίσεις Length SPQ και Primehash SPQ αποδείχθηκαν ταχύτατες ανεξαρτήτως αριθμού ακμών της διαδρομής π, ενώ η ProcessSPQ παρέχει στο χρήστη τη δυνατότητα εισαγωγής χρονικών κριτηρίων στο Ερώτημα Ακριβούς Τροχιάς που εκτελεί.

5.2 Προτάσεις βελτίωσης

Στην παρούσα προσέγγιση έγινε μια προσπάθεια για την υλοποίηση του προτεινόμενου [16] αλγορίθμου εξαγωγής τροχιών με χρήση συστημάτων λογισμικού ανοιχτού κώδικα. Η υλοποίηση κατέδειξε προτερήματα αλλά και αδυναμίες σε ορισμένα σημεία της υλοποίησης, οι οποίες αν διορθωθούν θα συμβάλλουν καταλυτικά στην απόδοσή της.

Ο αλγόριθμος αντιστοίχισης δεδομένων στο δίκτυο υποβάθρου (map-matching) παρ'ότι υπήρξε αξιόπιστος ως προς τα αποτελέσματα που εξήγε, εισήγαγε σημεία στα οποία απαιτούνταν σημαντική μετέπειτα επεξεργασία για την τελική αντιστοίχιση στο χάρτη και την σύνθεση των τροχιών. Η επεξεργασία αυτή είναι μάλλον αδύνατον να εξαλειφθεί, ωστόσο μπορεί να περιοριστεί σημαντικά αν ο αλγόριθμος σχεδιαστεί ώστε να συσχετίζει τις θέσεις των αντικειμένων που δίδονται από το GPX των δεδομένων στις ακμές του δρομολογήσιμου δικτύου και όχι στους δρόμους του αρχικού δικτύου, όπως συμβαίνει μέχρι τώρα. Με αυτόν τον τρόπο θα παρέχεται μια σημαντική πληροφορία για την μετέπειτα επεξεργασία, η οποία αυτή τη στιγμή στηρίζεται στην αντιστοίχιση των τροχιακών segments με κόμβους του αρχικού OSM δικτύου από

όπου γίνεται αντιστοίχιση με τους κόμβους του δρομολογήσιμου δικτύου και τελικά με τις ακμές.

Η υλοποίηση των συναρτήσεων σε γλώσσα pl/pgsql στερείται της ταχύτητας της αντίστοιχης υλοποίησης σε γλώσσα C, που επίσης υποστηρίζεται από το ΣΔΒΔ Postgres. Η υλοποίηση των συναρτήσεων σε γλώσσα C αναμένεται να αυξήσει σημαντικά τις επιδόσεις της κατασκευής των ερωτημάτων SPQ και κατ' επέκτασιν να μειώσει σημαντικά το χρόνο εκτέλεσης των SPQ wrappers.

Ο αλγόριθμος αναζήτησης συντομότερης διαδρομής μεταξύ δύο δικτυακών κόμβων που χρησιμοποιείται στην παρούσα υλοποίηση καθυστερεί σημαντικά την λειτουργία αλγορίθμων όπως ο UniqueSubPaths, ο οποίος πραγματοποιεί συνεχώς αναζητήσεις συντομότερης διαδρομής κατά την εκτέλεσή του, αλλά και ο exact ProcessSPQ, ο οποίος καλεί τον UniqueSubPaths όσο εκτελείται. Η χρήση ενός ιδιαίτερα γρήγορου αλγορίθμου συντομότερης διαδρομής αναμένεται να βελτιώσει δραματικά την απόδοση των εν λόγω αλγορίθμων, ενώ θα βελτιώσει τις επιδόσεις και των άλλων συναρτήσεων, που χρησιμοποιούν αλγόριθμο εύρεσης κοντινότερης διαδρομής για τη σύνθεση της διαδρομής π που απαιτείται για την εκτέλεση του SPQ.

6. Βιβλιογραφία

[1] Altman N. S. *An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician (1992). 46 (3): 175–185.*

[2] Beckmann N., Begel H.-P., Schneider R., Seeger B. *The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. Praktiche Informatik, Universität Bremen, West Germany, 1990.*

[3] Bernstein P. A.; Goodman N. *Concurrency Control in Distributed Database Systems. ACM Computing Surveys (1981).*

[4] Cha Chang Il, Kim Sang-Wook, Won Jung-Im, Lee Junghoon, Bae Duck-Ho, *Efficient Indexing in Trajectory Databases, International Journal of Database Theory and Application, Vol.1, No.1, pages 21-28, 2008*

[5] Chakka V. P., Everspaugh A. C., Patel J. M. *Indexing large trajectory data sets with SETI. Procceedings Int'l. Conference on Biennial Conference on Innovative Data Systems Research, 2003.*

[6] Chang Jae-Woo, Um Jung-Ho, Lee Wang-Chen. *A New Trajectory Indexing Scheme for Moving Objects on Road Networks. D. Bell and J. Hong Editions: BNCOD 2006, Lecture Notes in Computer Science, Vol. 4042, pp. 291-294, 2006.*

[7] Chang Jae-Woo, Song Myoung-Seon, Um Jung-Ho. *TMN-tree: New Trajectory Index Structure for Moving Objects in Spatial Networks. 10th IEEE International Conference on Computer and Information Technology, 2010.*

[8] Comer D. *The Ubiquitous B-tree. ACM Computing Surveys, 11(2): 121-137, June 1979.*

[9] Egenhofer M., Herring J. *Categorizing Binary Topological Relationships Between Regions, Lines and Points in Geographic Databases. Technical Report, Department of Surveying Engineering, University of Maine, Orono, ME*

[10] Finkel R., Bentley J.. *Quad Trees: A Data Structure for Retrieval on Composite Keys. Acta Informatica, Springer Verlag 4 (1974), 1-9.*

[11] Frentzos E. *Indexing Objects Moving on Fixed Networks. Advances in Spatial and Temporal Databases, pages 289–305, 2003.*

[12] Guttman A. *R-trees: A dynamic index structure for spatial searching. Proceedings of the 1984 ACM SIGMOD international conference on Management of data - SIGMOD '84. p. 47*

[13] Hadjieleftheriou M., Kollios G., Tsotras V. J., and Gunopulos D. *Indexing spatiotemporal archives. VLDB J., 15(2):143–164, 2006.*

[14] Haerder T.; Reuter A. *Principles of transaction-oriented database recovery*. *ACM Computing Surveys (1983). 15 (4): 287*

[15] Hardy G., Wright E. *An introduction to the theory of numbers. Oxford University Press, USA, 1980.*

[16] Krogh B., Pelekis N., Theodoridis Y., Torp K. *Path-based Queries on Trajectory Data. SIGSPATIAL '14, November 04-07 2014, Dallas/Fort Worth, TX, USA.*

[17] Krogh B., Andersen O., and Torp K. *Trajectories for Novel and Detailed Traffic Analysis. ACM-GIS IWGS, 2012.*

[18] Lou Yin, Xie Xing, Zhang Chengyang, Wang Wei, Zheng Yu, Huang Yan. *Map-Matching for Low-Sampling-Rate GPS Trajectories. Proceedings of ACM SIGSPATIAL GIS 2009.*

[19] Orenstein J. *Spatial Query Processing in an Object-Oriented Database System, Proceedings of ACM SIGMOD International Condference on Management of Data, May 1986.*

[20] Pan B., Zheng Y., Wilkie D., and Shahabi C.. *Crowd sensing of traffic anomalies based on human mobility and social media. ACM-GIS, 2013.*

[21] Pfoser D., Jensen C. S. *Indexing of Network Constrained Moving Objects. ACM-GIS, pages 25–32. ACM, 2003.*

[22] Pfoser D., Jensen C. S., Theodoridis Y. *Novel Approaches to the Indexing of Moving Object Trajectories. Proceedings of the 26th International Conference on Very Large Databases, Cairo, Egypt, 2000.*

[23] Popa I. S., Zeitouni K., Oria V., Barth D., and Vial S. *Indexing in-network trajectory flows. VLDB J., 20(5):643{669, 2011.*

[24] Tao Yufei, Papadias D. *The MV3R-tree: A Spatio-Temporal Access Method for Timestamp and Interval Queries. Proceedings of the 27th VLDB Conference, Roma, Italy, 2001.*

[25] Teixeira de Almeida V., Guting R.H. *Indexing the Trajectories of Moving Objects in Networks. Praktische Informatik IV Fernuniversität Hagen, D-58084 Hagen, Germany*

[26] Theodoridis Y., Vazirgiannis M., Sellis T. *Spatio-Temporal Indexing for Large Multimedia Applications. Proceedings of the IEEE International Conference on Multimedia Systems, Hiroshima,Japan, 1996.*

Διαδικτυακές αναφορές

[27] http://www.postgresql.org/about/

[28] http://en.wikipedia.org/wiki/PostgreSQL

[29] http://en.wikipedia.org/wiki/ACID

[30] http://en.wikipedia.org/wiki/MVCC

[31] http://openfts.sourceforge.net/

[32] http://en.wikipedia.org/wiki/PostGIS

[33] http://pgrouting.org/

[34] http://www.openstreetmap.org/

[35] http://en.wikipedia.org/wiki/OpenStreetMap

[36] https://mapzen.com/metro-extracts

[37] http://wiki.openstreetmap.org/wiki/Routing/Travel_Time_Analysis

[38] http://traveltimeanalysis.googlecode.com/svn/trunk/

[39] http://en.wikipedia.org/wiki/QGIS

7. Παράρτημα

7.1 Πίνακες Βάσης Δεδομένων

Στην παρούσα παράγραφο περιγράφεται η λειτουργία, κατασκευή και χρήση των πινάκων που δημιουργούνται στη Βάση Δεδομένων για την υποστήριξη των λειτουργιών του συστήματος. Ακόμη αναφέρονται συνοπτικά τα ευρετήρια και τα πεδία που περιέχονται σε αυτούς.

➢ `load_trajectories`

ΧΡΗΣΗ: Περιέχει τα αναγνωριστικά τροχιών που επιστρέφονται από τον αλγόριθμο ProcessSPQ, που υλοποιείται από τη συνάρτηση process_spq.

ΛΕΙΤΟΥΡΓΙΑ: Στον πίνακα αποθηκεύονται τα αποτελέσματα ενός προσεγγιστικού SPQ με συνάρτηση βάρους το μήκος ακμής βάσει των κριτηρίων που δίνονται στη συνάρτηση process_spq, υλοποιώντας το σύνολο C του αλγορίθμου ProcessSPQ. Τα αποτελέσματα αυτά ενδέχεται να φιλτραριστούν περαιτέρω στη συνέχεια της εκτέλεσης του ProcessSPQ.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας κατασκευάζεται από τη συνάρτηση process_spq(bigint, bigint, boolean, timestamp, timestamp).

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

traj_id integer	ID τροχιάς
trip_id integer	ID υποτροχιάς
start_length double precision	Hash αρχικής ακμής
end_length double precision	Hash τελικής ακμής
time_enter timestamp without time zone	Χρονική στιγμή εισόδου αντικειμένου στη διαδρομή
time_leave timestamp without time zone	Χρονική στιγμή εξόδου αντικειμένου από τη διαδρομή

➢ `load_trajectories2_temp`

ΧΡΗΣΗ: Φιλτράρισμα αποτελεσμάτων που έχουν αποθηκευτεί στον load_trajectories.

ΛΕΙΤΟΥΡΓΙΑ: Σε περίπτωση κυκλικής διαδρομής ή επιλογής εκτέλεσης ακριβούς αναζήτησης από τον χρήστη κατά την κλήση της process_spq, καλείται η συνάρτηση call_unique_subpaths, η οποία υλοποιεί τον αλγόριθμο UniqueSubPaths. Η συνάρτηση ενδέχεται να τμηματοποιήσει την αρχική διαδρομή, βάσει των μοναδικών υπο-διαδρομών που θα εντοπίσει σε αυτήν. Για το

φιλτράρισμα των αποτελεσμάτων της process_spq, για κάθε υπο-διαδρομή που θα βρεθεί για την αρχική διαδρομή πραγματοποιείται προσεγγιστικό SPQ και τα αποτελέσματά του αποθηκεύονται στον load_trajectories2_temp, υλοποιώντας το σύνολο C' του ProcessSPQ. Όσα αναγνωριστικά τροχιών δεν περιέχονται στον πίνακα load_trajectories2-_temp, αφαιρούνται από τον load_trajectories.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας κατασκευάζεται από τη συνάρτηση process-_spq(bigint, bigint, boolean, timestamp, timestamp).

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

traj_id integer	ID τροχιάς
trip_id integer	ID υποτροχιάς
start_length double precision	Hash αρχικής ακμής
end_length double precision	Hash τελικής ακμής
time_enter timestamp without time zone	Χρονική στιγμή εισόδου αντικειμένου στη διαδρομή
time_leave timestamp without time zone	Χρονική στιγμή εξόδου αντικειμένου από τη διαδρομή

- **map_matching**

ΧΡΗΣΗ: Αντιστοίχιση κινούμενων αντικειμένων στο χάρτη, προετοιμασία ευρετηρίου NETTRA.

ΛΕΙΤΟΥΡΓΙΑ: Στον πίνακα αποθηκεύονται οι πληροφορίες αντιστοίχισης θέσεων κινούμενων αντικειμένων με τις ακμές του δικτύου. Ο πίνακας συμπληρώνεται αυτόματα από την συνάρτηση map_matching_strict(), αποτελώντας ουσιαστικά τον χώρο αποθήκευσης των αποτελεσμάτων της συνάρτησης.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας κατασκευάζεται από τη συνάρτηση map_matching_strict().

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
traj_id integer	ID τροχιάς
trip_id integer	ID υποτροχιάς
segment_id integer	ID τμήματος τροχιάς
order_id integer	ID αύξοντος αριθμού ακμής για ένα τμήμα τροχιά
routable_osm_id integer	ID ακμής δρομολογήσιμου δικτύου

- **osm_nodes**

ΧΡΗΣΗ: Φόρτωση OSM δικτύου στη Βάση Δεδομένων.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση των κόμβων του OSM δικτύου και της γεωμετρίας τους για το χαρτογραφικό υπόβαθρο. Περιέχει τα δεδομένα του αρχείου osm_nodes.txt που παράγεται από το xml_manip.php και φορτώνονται στη ΒΔ με εντολή SQL COPY.

ΚΑΤΑΣΚΕΥΗ: Κατασκευάζεται από το SQL script δημιουργίας της ΒΔ.

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
osm_id bigint NOT NULL	OSM ID κόμβου
lat double precision	Γεωγραφικό πλάτος (WGS 84)
lon double precision	Γεωγραφικό μήκος (WGS 84)
geom geometry	Γεωμετρία

- **osm_nodes_ways**

ΧΡΗΣΗ: Φόρτωση OSM δικτύου στη Βάση Δεδομένων.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση των κόμβων του OSM δικτύου και της γεωμετρίας τους για το χαρτογραφικό υπόβαθρο. Περιέχει τα δεδομένα του αρχείου osm_nodes.txt που παράγεται από το xml_manip.php και φορτώνονται στη ΒΔ με εντολή SQL COPY.

ΚΑΤΑΣΚΕΥΗ: Κατασκευάζεται από το SQL script δημιουργίας της ΒΔ.

ΕΥΡΕΤΗΡΙΑ:

node_no_idx	B-tree στο πεδίο **node_no**
osm_node_id_idx	B-tree στο πεδίο **osm_node_id**
osm_routable_way_idx	B-tree στο πεδίο **routable_osm_id**
osm_way_id_idx	B-tree στο πεδίο **osm_way_id**

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
routable_osm_id bigint	ID ακμής δρομολογήσιμου δικτύου
osm_way_id bigint	ID OSM δρόμου
osm_node_id bigint	ID OSM κόμβου
node_no integer	ID κόμβου

- **`osm_routable_network`**

ΧΡΗΣΗ: Δρομολογήσιμο δίκτυο οδών.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση ακμών δρομολογήσιμου δικτύου. Περιέχει πληροφορία για τη γεωμετρία κάθε ακμής, το αναγνωριστικό της (που έχει προκύψει από το OSM2Routing) και τα βάρη κάθε ακμής (μήκους και πρώτου αριθμού) που θα χρησιμοποιηθούν μελλοντικά για την εκτέλεση των ερωτημάτων ακριβούς τροχιάς. Ακόμη, στον πίνακα εκτελείται η δημιουργία τοπολογίας του pgrouting, καθιστώντας τον πίνακα απαραίτητο για κάθε δρομολόγηση εντός του δικτύου.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας κατασκευάζεται από τη συνάρτηση create_routable_line-strings().

ΕΥΡΕΤΗΡΙΑ:

osm_routable_network_geom_gidx	B-tree στο πεδίο **geom**
osm_routable_network_idx	B-tree στο πεδίο **routable_osm_id**
osm_routable_network_source_idx	B-tree στο πεδίο **source**
osm_routable_network_target_idx	B-tree στο πεδίο **target**

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
routable_osm_id bigint	ID ακμής δρομολογήσιμου δικτύου
geom geometry	Γεωμετρία ακμής δικτύου
source integer	Πεδίο source για pgrouting
target integer	Πεδίο target για pgrouting
length double precision	Βάρος μήκους ακμής
hash double precision	Βάρος λογαρίθμου πρώτου αριθμού

- **`osm_routable_network_vertices_pgr`**

ΧΡΗΣΗ: Δρομολογήσιμο δίκτυο οδών.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση κόμβων δρομολογήσιμου δικτύου. Οι κόμβοι δρομολογήσιμου δικτύου αποτελούν υποσύνολο των osm_nodes, και, αν και χωρικά ταυτίζονται με τους αντίστοιχους OSM κόμβους, έχουν διαφορετικά IDs, πρόβλημα που διορθώνεται από συγκεκριμένη pl/pgsql συνάρτηση. Οι κόμβοι αυτοί δημιουργούνται κατά τη δημιουργία τοπολογίας δικτύου από το pgrouting και είναι απαραίτητοι για τη δρομολόγηση εντός του δικτύου.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας κατασκευάζεται από τη συνάρτηση pgr_create_topology() του pgrouting. Ωστόσο, για την αντιστοίχιση των OSM κόμβων με τους κόμβους pgrouting προστίθεται το πεδίο osm_id, στο οποίο γίνεται η αντιστοίχιση από ειδική συνάρτηση.

ΕΥΡΕΤΗΡΙΑ:

osm_routable_network_vertices_ _pgr_the_geom_idx	B-tree στο πεδίο **the_geom**
pgr_vertices_id_idx	B-tree στο πεδίο **id**
pgr_vertices_osm_id_idx	B-tree στο πεδίο **osm_id**

ΠΕΔΙΑ:

id bigserial NOT NULL	ID εγγραφής, δημιουργείται από το pgrouting
cnt integer	Δημιουργείται από το pgrouting
chk integer	Δημιουργείται από το pgrouting
ein integer	Δημιουργείται από το pgrouting
eout integer	Δημιουργείται από το pgrouting
the_geom geometry(Point,4326)	Γεωμετρία κόμβου
osm_id bigint	ID OSM κόμβου

- **`osm_visited_segments`**

ΧΡΗΣΗ: Αντιστοίχιση δεδομένων στο χάρτη, προετοιμασία ευρετηρίου NETTRA.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση δεδομένων αντιστοίχισης θέσεων αντικειμένων στις ακμές του δικτύου. Στον πίνακα αποθηκεύονται τα αποτελέσματα της πρώτης φάσης του map matching, όπου με χρήση του gpx_reader.php διαβάζονται τα αρχεία που παρήχθησαν από την MatchGPX2OSM και δημιουργείται αρχείο (osm_visited_segments.txt) που φορτώνεται στον πίνακα με εντολή SQL COPY.

ΚΑΤΑΣΚΕΥΗ: Δημιουργείται κατά την κατασκευή της ΒΔ με χρήση SQL script.

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
traj_id integer	ID τροχιάς
trip_id integer	ID υποτροχιάς
segment_id integer	ID τμήματος τροχιάς
osm_node_id bigint	ID κόμβου OSM
osm_way_id bigint	ID δρόμου OSM
order_id integer	Αύξων αριθμός ακμής για συγκεκριμένο τμήμα τροχιάς

- **`osm_ways`**

ΧΡΗΣΗ: Φόρτωση δικτύου OSM στη Βάση Δεδομένων.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση δεδομένων OSM δρόμων. Περιέχει τα δεδομένα δρομολογήσιμου δικτύου που δημιουργούνται από την εφαρμογή OSM2Routing και μετατρέπονται σε txt από το php script xml_manip.php.
ΚΑΤΑΣΚΕΥΗ: Δημιουργείται κατά την κατασκευή της ΒΔ με χρήση SQL script.

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
routable_osm_id bigint NOT NULL	ID ακμής δρομολογήσιμου δικτύου
osm_id bigint NOT NULL	OSM ID δρόμου
speed integer	Όριο ταχύτητας
accessible character varying(10)	Πρόσβαση στο δρόμο από οχήματα
accessible_reverse character varying(10)	Πρόσβαση με αντίθετη κατεύθυνση

- **`result_routes`**

ΧΡΗΣΗ: Λειτουργίες εμφάνισης αποτελεσμάτων στο QGIS.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση γεωμετριών τροχιών που έχουν προκύψει ως αποτελέσματα από wrappers επίδειξης SPQ ερωτημάτων. Οι συγκεκριμένες συναρτήσεις χρησιμοποιούνται για την εμφάνιση των αποτελεσμάτων των SPQ σε ένα χάρτη του QGIS.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας δημιουργείται από τις συναρτήσεις επίδειξης (περιέχουν το λεκτικό «_show» στην ονομασία τους π.χ. spq1_show(bigint, bigint)).

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
traj_id integer	ID τροχιάς
trip_id integer	ID υποτροχιάς
segment_id integer	ID τμήματος τροχιάς
geom geometry	Γεωμετρία τροχιάς

- **spatial_ref_sys**

ΧΡΗΣΗ: Πίνακας μη άμεσα απαραίτητος για τη λειτουργία του αλγορίθμου, απαραίτητος ωστόσο για την προσθήκη PostGIS και κατ' επέκτασιν όλο το σύστημα.

ΛΕΙΤΟΥΡΓΙΑ: Πληροφορίες για τα συστήματα αναφοράς που χρησιμοποιούνται από τη ΒΔ.

ΚΑΤΑΣΚΕΥΗ: Δημιουργείται αυτόματα με την εγκατάσταση των λειτουργιών του PostGIS στη ΒΔ.

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

srid integer NOT NULL,	SRID συστήματος αναφοράς
a**uth_name** character varying(256),	Όνομα οργανισμού
auth_srid integer,	SRID συστήματος αναφοράς
srtext character varying(2048),	Ονομασία συστήματος και πληροφορίες
proj4text character varying(2048),	Πληροφορίες προβολής

- **spq_route**

ΧΡΗΣΗ: Λειτουργίες εμφάνισης αποτελεσμάτων SPQs στο QGIS.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση γεωμετρίας διαδρομής π που έχει οριστεί ως βασική διαδρομή SPQ σε wrapper επίδειξης SPQ ερωτημάτων. Οι συγκεκριμένες συναρτήσεις χρησιμοποιούνται για την εμφάνιση των αποτελεσμάτων των SPQ σε ένα χάρτη του QGIS.

ΚΑΤΑΣΚΕΥΗ: Ο πίνακας δημιουργείται από τις συναρτήσεις επίδειξης (περιέχουν το λεκτικό «_show» στην ονομασία τους π.χ. spq1_show(bigint, bigint)).

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL,	ID εγγραφής
routable_osm_id integer,	ID ακμής δρομολογήσιμου δικτύου
geom geometry	Γεωμετρία διαδρομής

- `truck_nodes`

ΧΡΗΣΗ: Φόρτωση θέσεων κινούμενων αντικειμένων από το σύνολο δεδομένων (dataset) στη Βάση Δεδομένων.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση θέσεων κινούμενων αντικειμένων συνόλου δεδομένων. Τα δεδομένα φορτώνονται σε αυτόν μέσω εντολής SQL COPY, απευθείας από το αρχείο που περιέχει τα δεδομένα (η δομή του αρχείου περιγράφεται στο παρόν Παράρτημα).

ΚΑΤΑΣΚΕΥΗ: Δημιουργείται κατά την κατασκευή της ΒΔ με χρήση SQL script.

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL,	ID εγγραφής
traj_id integer,	ID τροχιάς
trip_id integer,	ID υποτροχιάς
date_file character varying,	Ημερομηνία
time_file character varying,	Ώρα
time_stamp timestamp without time zone,	Ημερομηνία – ώρα
x double precision,	Γεωγραφικό μήκος (WGS 84)
y double precision,	Γεωγραφικό πλάτος (WGS 84)
x_egsa double precision,	Γεωγραφικό μήκος (ΕΓΣΑ 87)
y_egsa double precision,	Γεωγραφικό πλάτος (ΕΓΣΑ 87)

- `truck_segments`

ΧΡΗΣΗ: Σύνθεση τμημάτων τροχιών κινούμενων αντικειμένων από τις θέσεις που περιέχονται στο σύνολο δεδομένων.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση των τμημάτων τροχιών κινούμενων αντικειμένων του συνόλου δεδομένων. Τα δεδομένα του χρησιμοποιούνται για την οπτικοποίηση των τροχιών του συνόλου δεδομένων πριν την αντιστοίχισή τους στο χάρτη αλλά και για τον υπολογισμό χρονικών στοιχείων από αυτά από τη συνάρτηση add_timestamps().

ΚΑΤΑΣΚΕΥΗ: Δημιουργείται από τη συνάρτηση create_truck_segments().

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
traj_id integer	ID τροχιάς
trip_id integer	ID υποτροχιάς
segment_id integer	ID τμήματος τροχιάς
time_stamp_s timestamp without time zone	Χρονική στιγμή αρχικού σημείου τμήματος τροχιάς
time_stamp_e timestamp without time zone	Χρονική στιγμή τελικού σημείου τροχιάς
length double precision	Μήκος τμήματος τροχιάς
speed double precision	Ταχύτητα (δεν χρησιμοποιείται)
geom geometry	Γεωμετρία τμήματος τροχιάς

- **`unique_subpaths`**

ΧΡΗΣΗ: Εκτέλεση βελτιωμένης προσέγγισης ακριβούς SPQ.

ΛΕΙΤΟΥΡΓΙΑ: Αποθήκευση των αποτελεσμάτων εκτέλεσης του Unique-SubPaths αλγορίθμου. Η πληροφορία που αντλείται από αυτόν χρησιμεύει στην κατασκευή του ακριβούς SPQ που προκύπτει από αναζήτηση μοναδικών διαδρομών.

ΚΑΤΑΣΚΕΥΗ: Δημιουργείται από τη συνάρτηση call_unique_subpaths.

ΕΥΡΕΤΗΡΙΑ: -

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
start_routable_osm_id integer	Αρχική ακμή διαδρομής
end_routable_osm_id integer	Τελική ακμή διαδρομής
deltahash double precision	Βάρος διαδρομής (deltahash)

- **`visited_segments`**

ΧΡΗΣΗ: Εκτέλεση ερωτημάτων SPQ.

ΛΕΙΤΟΥΡΓΙΑ: Αποτελεί την υλοποίηση του ευρετηρίου NETTRA. Περιέχει την αντιστοίχιση των τροχιών κινούμενων αντικειμένων στις ακμές του δικτύου υποβάθρου, τη χρονική πληροφορία εισόδου και εξόδου του κινούμενου αντικειμένου σε κάθε ακμή καθώς και τα αθροιστικά βάρη (deltahash) μήκους και λογαρίθμου πρώτου αριθμού κάθε τροχιάς, υπολογιζόμενα για κάθε εγγραφή της.

ΚΑΤΑΣΚΕΥΗ: Κατασκευάζεται από την συνάρτηση create_visited_segments.

ΕΥΡΕΤΗΡΙΑ:

load_trajectories_idx	B-tree στα πεδία **routable_osm_id, time_enter, traj_id, trip_id, time_leave, s_hash**
visited_segments_id_index	B-tree στο πεδίο **id**
visited_segments_routable_index	B-tree στο πεδίο **routable_osm_id**

ΠΕΔΙΑ:

id serial NOT NULL	ID εγγραφής
traj_id integer	ID τροχιάς
traj_trip_id integer	ID υποτροχιάς πριν την κλήση της create_visited_segments()
traj_segment_id integer	ID τμήματος υποτροχιάς πριν την κλήση της create_visited_segments()
trip_id integer	ID υποτροχιάς μετά την κλήση της create_visited_segments()
segment_id integer	ID τμήματος υποτροχιάς μετά την κλήση της create_visited_segments()
routable_osm_id integer	ID δρομολογήσιμης ακμής δικτύου
time_enter timestamp without time zone	Χρονική στιγμή εισόδου του κινούμενου αντικειμένου στην ακμή
time_leave timestamp without time zone	Χρονική στιγμή εξόδου του κινούμενου αντικειμένου από την ακμή
s_hash double precision	Αθροιστικό βάρος εγγραφής από την αρχή της υποτροχιάς (μήκος)
s_length double precision	Αθροιστικό βάρος εγγραφής από την αρχή της υποτροχιάς (λογάριθμος πρώτου αριθμού)

7.2 Συναρτήσεις Βάσης Δεδομένων (pl/pgsql)

Στην παρούσα παράγραφο περιγράφονται συνοπτικά οι λειτουργίες που επιτελούνται από τις συναρτήσεις (stored procedures) που αναπτύχθηκαν κατά την υλοποίηση του συστήματος. Πέραν των λειτουργιών, παρουσιάζονται οι παράμετροι που απαιτούνται για την κλήση κάθε συνάρτησης και τα παραγόμενα αποτελέσματα, ενώ δίνονται παραδείγματα κλήσης κάθε συνάρτησης.

- **`add_timestamps()`**

ΕΙΣΟΔΟΣ: -

ΛΕΙΤΟΥΡΓΙΑ: Υπολογισμός χρονικών στιγμών εισόδου σε ακμή και εξόδου από ακμή για κάθε τμήμα τροχιάς του συνόλου δεδομένων. Η συνάρτηση χρησιμοποιείται για τον εμπλουτισμό του ευρετηρίου NETTRA (πίνακας

visited_segments) με χρονική πληροφορία, ώστε να είναι δυνατή η εκτέλεση χωροχρονικών SPQs με τη βοήθεια του αλγορίθμου ProcessSPQ. Σε περιπτώσεις αραιής δειγματοληψίας θέσεων, οι χρονικές στιγμές για τις ενδιάμεσες ακμές υπολογίζονται προσεγγιστικά, κάνοντας την παραδοχή ότι η μετάβαση από μία ακμή σε άλλη γίνεται σε ίσα μεταξύ τους χρονικά διαστήματα.

ΕΞΟΔΟΣ: Ανανέωση τιμών πίνακα visited_segments (ευρετήριο NETTRA) στα πεδία time_enter και time_leave.

ΚΛΗΣΗ: `select add_timestamps();`

➢ assign_weights_to_routable_segments()

ΕΙΣΟΔΟΣ: -

ΛΕΙΤΟΥΡΓΙΑ: Υπολογισμός βαρών μήκους ακμής και λογαρίθμου πρώτου αριθμού για κάθε ακμή του δρομολογήσιμου δικτύου. Εισαγωγή των τιμών σε κατάλληλα πεδία του πίνακα δρομολογήσιμου δικτύου και υπολογισμός αθροιστικών βαρών για κάθε τροχιά του συνόλου δεδομένων και αποθήκευση των τιμών σε κατάλληλα πεδία του πίνακα visited_segments (ευρετήριο NETTRA). Η κλήση της συνάρτησης είναι απαραίτητη για τη δημιουργία του ευρετηρίου NETTRA και την εκτέλεση των ερωτημάτων ακριβούς τροχιάς, αφού τα πεδία αθροιστικού βάρους χρησιμοποιούνται για την εξαγωγή του deltahash, απαραίτητης τιμής για το φιλτράρισμα των αποτελεσμάτων.

ΕΞΟΔΟΣ: Ανανέωση τιμών στους πίνακες osm_routable_network (δρομολογήσιμο δίκτυο) και visited_segments (ευρετήριο NETTRA).

ΚΛΗΣΗ: `select assign_weights_to_routable_segments();`

➢ call_unique_subpaths(bigint, bigint)

ΕΙΣΟΔΟΣ: 1. OSM_ID αρχικού κόμβου δικτύου (bigint)
2. OSM_ID τελικού κόμβου δικτύου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση κατασκευής και εκτέλεσης ακριβούς SPQ ερωτήματος. Σαν είσοδος δίνονται τα αναγνωριστικά δύο OSM κόμβων δικτύου, οι οποίοι αντιστοιχίζονται σε κόμβους δρομολογήσιμου δικτύου pgRouting. Μετέπειτα καλείται η συνάρτηση unique_shortest_paths2 η οποία υπολογίζει την συντομότερη διαδρομή που τους συνδέει και αναζητά αναδρομικά τις μοναδικές διαδρομές που την απαρτίζουν, αποθηκεύοντας στον πίνακα unique_subpaths πληροφορίες (αρχική ακμή, τελική ακμή, deltahash) για κάθε μία από αυτές. Με το πέρας της αναδρομικής διαδικασίας, καλείται η

συνάρτηση spq2_from_usp η οποία κατασκευάζει και εκτελεί το ερώτημα ακριβούς τροχιάς βάσει των αποτελεσμάτων που έχει εξάγει η unique_shortest_paths2.

ΕΞΟΔΟΣ: Ανανέωση τιμών πίνακα unique_subpaths με τις πληροφορίες για της μοναδικές υπο-διαδρομές. Επιστροφή σαν αποτέλεσμα των αναγνωριστικών τροχιών που ακολουθούν επακριβώς την υπολογισμένη διαδρομή.

ΚΛΗΣΗ: `select call_unique_subpaths(155586579, 911998377);`

- **`create_routable_linestrings()`**

ΕΙΣΟΔΟΣ: -

ΛΕΙΤΟΥΡΓΙΑ: Δημιουργία γεωμετρίας δρομολογήσιμου δικτύου από τα δεδομένα OSM δικτύου που βρίσκονται αποθηκευμένα στους πίνακες osm_nodes, osm_ways, osm_nodes_ways. Κατά την εκτέλεση της συνάρτησης δημιουργείται ο πίνακας osm_routable_network όπου και θα αποθηκευτούν οι απαραίτητες πληροφορίες για τις ακμές δρομολογήσιμου δικτύου, όπως υπολογίζονται από τη συνάρτηση.

ΕΞΟΔΟΣ: Δημιουργία και συμπλήρωση του πίνακα osm_routable_network.

ΚΛΗΣΗ: `select create_routable_linestrings();`

- **`create_truck_segments()`**

ΕΙΣΟΔΟΣ: -

ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση σύνθεσης τμημάτων τροχιών από τις θέσεις κινούμενων αντικειμένων που περιέχονται στο σύνολο δεδομένων (dataset). Για την σωστή λειτουργία του προϋποθέτει ο πίνακας truck_nodes να περιέχει τις θέσεις των κινούμενων αντικειμένων.

ΕΞΟΔΟΣ: Δημιουργία και ανανέωση πίνακα truck_segments με τις γεωμετρίες τροχιών του συνόλου δεδομένων. Οι γεωμετρίες μπορούν να χρησιμοποιηθούν για οπτικοποίηση του συνόλου δεδομένων.

ΚΛΗΣΗ: `select create_truck_segments();`

- **`create_visited_segments()`**

  ΕΙΣΟΔΟΣ: -

  ΛΕΙΤΟΥΡΓΙΑ: Η create_visited_segments() αποτελεί τη συνάρτηση που υλοποιεί το τελευταίο στάδιο της αντιστοίχισης των τροχιών στο δίκτυο, όπου απαλείφονται πιθανά σφάλματα αντιστοίχισης και δημιουργείται ο πίνακας visited_segments, που αποτελεί τη δομή του NETTRA. Ο visited_segments δέχεται τα αποτελέσματα εκτέλεσης της συνάρτησης.

  ΕΞΟΔΟΣ: Πίνακας visited_segments (ευρετήριο NETTRA).

  ΚΛΗΣΗ: `select create_visited_segments();`

- **`map_matching_strict()`**

  ΕΙΣΟΔΟΣ: -

  ΛΕΙΤΟΥΡΓΙΑ: Η συνάρτηση map_matching_strict() αναλαμβάνει την βελτιστοποίηση της αντιστοίχισης που έχει πραγματοποιηθεί από την μαζική εισαγωγή στη Βάση Δεδομένων των αποτελεσμάτων αντιστοίχισης από την χρήση της εφαρμογής MatchGPX2OSM (η αρχική αντιστοίχιση βρίσκεται στον πίνακα osm_visited_segments). Κατά την εκτέλεσή της, εντοπίζονται τα σημεία εντός των τροχιών όπου δεν υπάρχει αντιστοίχιση σε OSM κόμβο, και ελέγχεται η σύνδεσή τους. Η map_matching_strict συνδέει απευθείας τους κόμβους σε περίπτωση ύπαρξης ακμής δικτύου που τους συνδέει, ή εκτελεί δρομολόγηση εντός του δικτύου για να τους συνδέσει. Αν δεν βρεθεί σύνδεση, θεωρείται ότι ξεκινάει νέα υπο-τροχιά, γεγονός που σημειώνεται με κατάλληλη τιμή εντός της Βάσης Δεδομένων, για μετέπειτα επεξεργασία.

  ΕΞΟΔΟΣ: Δημιουργείται και ενημερώνεται ο πίνακας map_matching, ο οποίος περιέχει την βελτιωμένη αντιστοίχιση των τροχιών στις ακμές του δρομολογήσιμου δικτύου, για μετέπειτα επεξεργασία.

  ΚΛΗΣΗ: `select map_matching_strict();`

- **`osm_routable_segments_geometry()`**

  ΕΙΣΟΔΟΣ: -

  ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση δημιουργίας γεωμετρίας OSM δικτύου. Αποθήκευση αποτελεσμάτων στον πίνακα osm_ways. Χρησιμοποιείται για οπτικοποίηση του δικτύου υποβάθρου.

  ΕΞΟΔΟΣ: Πίνακας osm_ways, ενημερωμένος με τις γεωμετρίες των ακμών δικτύου.
  ΚΛΗΣΗ: `select osm_routable_segments_geometry();`

- **`pgr_set_vertices_osm_id()`**

  ΕΙΣΟΔΟΣ: -

  ΛΕΙΤΟΥΡΓΙΑ: Αντιστοίχιση κόμβων OSM στους κόμβους που δημιουργήθηκαν από το pgRouting. Η λειτουργία αυτή είναι απαραίτητη διότι σαν είσοδος στα ερωτήματα SPQ δίνονται αναγνωριστικά OSM κόμβων ενώ το pgrouting εκτελεί δρομολόγηση μεταξύ κόμβων με αρίθμηση που έχει οριστεί κατά τη δημιουργία τοπολογίας δικτύου από το ίδιο το pgrouting. Η συγκεκριμένη συνάρτηση αναζητά τον κόμβο του δικτύου OSM που απέχει μηδενική απόσταση από τον εκάστοτε κόμβο pgrouting και αναθέτει το OSM ID του OSM κόμβου στην εγγραφή του κόμβου pgrouting. Η συνάρτηση είναι ιδιαίτερα χρονοβόρα αλλά απαραίτητη για τη σωστή λειτουργία του αλγορίθμου. Η τροποποίηση του συστήματος ώστε να μην είναι απαραίτητη η συγκεκριμένη αντιστοίχιση μπορεί να βελτιώσει σημαντικά την απόδοση της διαδικασίας φόρτωσης χαρτογραφικού υποβάθρου.

  ΕΞΟΔΟΣ: Ενημέρωση πίνακα osm_routable_network_vertices_pgr με OSM IDs που αντιστοιχούν σε κάθε κόμβο του δικτύου pgrouting.

  ΚΛΗΣΗ: `select pgr_set_vertices_osm_id();`

- **`primes(integer, integer)`**

  ΕΙΣΟΔΟΣ: 1. Κάτω όριο διαστήματος (integer)
  2. Πάνω όριο διαστήματος (integer)

  ΛΕΙΤΟΥΡΓΙΑ: Υπολογισμός και επιστροφή ως αποτέλεσμα των πρώτων αριθμών εντός του δεδομένου αριθμητικού διαστήματος.

ΕΞΟΔΟΣ: Πρώτοι αριθμοί που βρίσκονται εντός του δεδομένου αριθμητικού διαστήματος.

ΚΛΗΣΗ: `select primes(1500, 3800);`

ΣΗΜΕΙΩΣΗ: Η συνάρτηση primes(integer, integer) χρησιμοποιήθηκε όπως είχε δημιουργηθεί από προηγούμενη υλοποίηση και δεν υλοποιήθηκε για τις ανάγκες της παρούσας εργασίας.

- **process_spq(bigint, bigint, boolean, timestamp, timestamp)**

ΕΙΣΟΔΟΣ: 1. Αρχικός OSM κόμβος (bigint)
2. Τελικός OSM κόμβος (bigint)
3. Αναζήτηση ακριβούς τροχιάς (boolean)
4. Κάτω όριο χρονικού διαστήματος (timestamp)
5. Πάνω όριο χρονικού διαστήματος (timestamp)

ΛΕΙΤΟΥΡΓΙΑ: Κατασκευή και εκτέλεση χωροχρονικού SPQ. Δίδεται δυνατότητα στο χρήστη να επιλέξει αν θα εκτελεστεί ακριβής η προσεγγιστική αναζήτηση τροχιών χωρίς παρεκκλίσεις από την αρχική διαδρομή. Στην περίπτωση επιλογής ακριβούς προσέγγισης, καλείται ο αλγόριθμος Unique-SubPaths ο οποίος πρώτα εξάγει τις μοναδικές υποδιαδρομές της διαδρομής για να ακολουθήσει κατασκευή και εκτέλεση του ερωτήματος. Στο εκτελούμενο SPQ προστίθεται επίσης και χρονικό κριτήριο αναζήτησης, οριζόμενο από τις τιμές χρονικού διαστήματος που ορίζει ο χρήστης.

ΕΞΟΔΟΣ: Τροχιές κινούμενων αντικειμένων που ακολουθούν τη δεδομένη διαδρομή ελέγχου χωρίς παρεκκλίσεις και εντός του δεδομένου χρονικού διαστήματος.
ΚΛΗΣΗ: `select * from process_spq(801467980, 985946206, true, '2000-12-31 00:00:00', '2015-12-31 00:00:00');`

- **spq1(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Κατασκευή και υλοποίηση προσεγγιστικού ερωτήματος SPQ με χρήση του μήκους ακμής ως μέγεθος βάρους για το φιλτράρισμα των αποτελεσμάτων. Η συνάρτηση δέχεται ως παραμέτρους δύο OSM κόμβους τους οποίους αντιστοιχεί με δύο κόμβους του δρομολογήσιμου δικτύου και υπολογίζει τη συντομότερη διαδρομή μεταξύ τους. Μετέπειτα ανακτώνται οι ακμές της διαδρομής και υπολογίζεται η διαφορά τιμής βάρους (deltahash)

μεταξύ αρχικής και τελικής ακμής της διαδρομής. Δημιουργείται το ερώτημα ακριβούς τροχιάς για τη συγκεκριμένη διαδρομή και εκτελείται, επιστρέφοντας στο χρήστη τα αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την συγκεκριμένη διαδρομή εντός του δικτύου.

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου.

ΚΛΗΣΗ: `select * from spq1(360221447,360198504);`

- **spq1_show(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Όμοια λειτουργικότητα με την spq1(bigint, bigint), με μόνη διαφορά τη δημιουργία και συμπλήρωση πινάκων με τη γεωμετρία της διαδρομής SPQ (πίνακας spq_route) και των τροχιών αποτελέσματος (πίνακας result_routes). Τα αποτελέσματα μπορούν να παρουσιαστούν οπτικά σε οποιαδήποτε εφαρμογή GIS (στην παρούσα εργασία χρησιμοποιήθηκε το QGIS).

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου. Δημιουργία και συμπλήρωση πινάκων spq_route, result_routes με τις γεωμετρίες διαδρομής SPQ και τροχιών που ακολουθούν τη διαδρομή SPQ χωρίς παρεκκλίσεις.

ΚΛΗΣΗ: `select * from spq1_show(360221447,360198504);`

- **spq1_show_primehash(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση όμοιας λειτουργικότητας με την spq1_show(bigint, bigint). Η διαφορά της έγκειται στη χρήση του λογαρίθμου πρώτου αριθμού ως τιμής βάρους για τον υπολογισμό του deltahash. Παρουσιάζει ίδια λειτουργικότητα με την spq3_show(bigint, bigint), συνεπώς η χρήση της δεν είναι απαραίτητη για την εκτέλεση «πρακτικής» προσέγγισης SPQ.

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου. Δημιουργία και συμπλήρωση πινάκων spq_route, result_routes με τις γεωμετρίες διαδρομής SPQ και τροχιών που ακολουθούν τη διαδρομή SPQ χωρίς παρεκκλίσεις.

ΚΛΗΣΗ: `select * from spq1_show_primehash(360221447,360198504);`

- **`spq2(bigint,bigint)`**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Υλοποίηση ακριβούς προσέγγισης SPQ. Η συνάρτηση δέχεται σαν παραμέτρους δύο OSM κόμβους, υπολογίζει τη συντομότερη διαδρομή μεταξύ τους και εξάγει ερώτημα ακριβούς προσέγγισης για αυτή τη διαδρομή, θεωρώντας την ένα σύνολο από υπο-διαδρομές με μήκος δύο ακμές. Το ερώτημα ακριβούς προσέγγισης εκτελείται, επιστρέφοντας στο χρήστη ως αποτέλεσμα τα αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς τη συντομότερη διαδρομή μεταξύ των δύο OSM κόμβων που έχουν δοθεί ως είσοδος στη συνάρτηση.

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου.

ΚΛΗΣΗ: `select * from spq2(360221447,360198504);`

- **`spq2_from_usp()`**

ΕΙΣΟΔΟΣ: -

ΛΕΙΤΟΥΡΓΙΑ: Ανάκτηση μοναδικών υπο-διαδρομών της διαδρομής SPQ που έχουν προκύψει από την κλήση του αλγορίθμου UniqueSubPaths και έχουν αποθηκευτεί στον πίνακα unique_subpaths. Βάσει των πληροφοριών αυτών κατασκευάζεται ερώτημα βελτιωμένης ακριβούς προσέγγισης SPQ, το οποίο εκτελείται, επιστρέφοντας στο χρήστη τα αποτελέσματά του.
ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου.

ΚΛΗΣΗ: `select * from spq2_from_usp();` (προϋποθέτει ο πίνακας unique_subpaths να περιέχει τις μοναδικές υποδιαδρομές της διαδρομής SPQ)
Η συνάρτηση καλείται από την call_unique_subpaths(bigint, bigint), που την καλεί με την εντολή `perform spq2_from_usp();`

- **spq2_show(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Όμοια λειτουργικότητα με την spq2(bigint, bigint), με μόνη διαφορά τη δημιουργία και συμπλήρωση πινάκων με τη γεωμετρία της διαδρομής SPQ (πίνακας spq_route) και των τροχιών αποτελέσματος (πίνακας result_routes). Τα αποτελέσματα μπορούν να παρουσιαστούν οπτικά σε οποιαδήποτε εφαρμογή GIS (στην παρούσα εργασία χρησιμοποιήθηκε το QGIS).

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου. Δημιουργία και συμπλήρωση πινάκων spq_route, result_routes με τις γεωμετρίες διαδρομής SPQ και τροχιών που ακολουθούν τη διαδρομή SPQ χωρίς παρεκκλίσεις.

ΚΛΗΣΗ: `select * from spq2_show(360221447,360198504);`

- **spq2_primehash(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση δημιουργίας και εκτέλεσης ερωτήματος ακριβούς προσέγγισης SPQ. Η συνάρτηση είναι ίδιας λειτουργικότητας με τη συνάρτηση spq2(bigint, bigint), με μόνη μεταξύ τους διαφορά τη χρήση λογαρίθμου πρώτου αριθμού για υπολογισμό του deltahash της διαδρομής και των εκάστοτε υπο-διαδρομών.

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου.

ΚΛΗΣΗ: `select * from spq2_primehash(bigint, bigint);`

- **spq2_show_primehash(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση όμοιας λειτουργικότητας με την spq2_show(bigint, bigint). Η διαφορά της έγκειται στη χρήση του λογαρίθμου πρώτου αριθμού ως τιμής βάρους για τον υπολογισμό του deltahash, μειώνοντας ακόμα περισσότερο την πιθανότητα ύπαρξης λανθασμένων αποτελεσμάτων.

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου. Δημιουργία και συμπλήρωση πινάκων spq_route, result_routes με τις γεωμετρίες διαδρομής SPQ και τροχιών που ακολουθούν τη διαδρομή SPQ χωρίς παρεκκλίσεις.
ΚΛΗΣΗ: `select * from spq2_show_primehash(bigint, bigint);`

- **spq3(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Κατασκευή και εκτέλεση SPQ ερωτήματος «πρακτικής» προσέγγισης με βάρος λογαρίθμου πρώτου αριθμού. Η συνάρτηση δέχεται σαν παραμέτρους δύο OSM κόμβους, εντοπίζει τη συντομότερη διαδρομή που τους συνδέει και υπολογίζει το deltahash για τη συγκεκριμένη διαδρομή. Μετέπειτα κατασκευάζεται κι εκτελείται το SPQ ερώτημα, επιστρέφοντας στο χρήστη τις τροχιές που κάποιο τμήμα τους ακολουθεί επακριβώς την διαδρομή που έχει οριστεί για το SPQ.

ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου.

ΚΛΗΣΗ: `select * from spq3(bigint, bigint);`

- **spq3_show(bigint, bigint)**

ΕΙΣΟΔΟΣ: 1. OSM ID αρχικού κόμβου (bigint)
2. OSM ID τελικού κόμβου (bigint)

ΛΕΙΤΟΥΡΓΙΑ: Όμοια λειτουργικότητα με την spq3(bigint, bigint), με μόνη διαφορά τη δημιουργία και συμπλήρωση πινάκων με τη γεωμετρία της διαδρομής SPQ (πίνακας spq_route) και των τροχιών αποτελέσματος (πίνακας result_routes). Τα αποτελέσματα μπορούν να παρουσιαστούν οπτικά σε οποιαδήποτε εφαρμογή GIS (στην παρούσα εργασία χρησιμοποιήθηκε το QGIS).
ΕΞΟΔΟΣ: Αναγνωριστικά τροχιών που ένα κομμάτι τους ακολουθεί επακριβώς την καθορισμένη διαδρομή εντός του δικτύου. Δημιουργία και συμπλήρωση πινάκων spq_route, result_routes με τις γεωμετρίες διαδρομής SPQ και τροχιών που ακολουθούν τη διαδρομή SPQ χωρίς παρεκκλίσεις.

ΚΛΗΣΗ: `select * from spq3_show(bigint, bigint);`

- **`unique_shortest_paths2(integer, integer)`**

ΕΙΣΟΔΟΣ: 1. pgrouting ID αρχικού κόμβου (integer)
2. pgrouting ID τελικού κόμβου (integer)

ΛΕΙΤΟΥΡΓΙΑ: Συνάρτηση υλοποίησης αλγορίθμου UniqueSubPaths. Λόγω της αναδρομικής εκτέλεσης του αλγορίθμου, σαν ορίσματα σε αυτόν δίνονται pgrouting IDs και όχι OSM IDs, για την αποφυγή μετατροπών που κοστίζουν σε χρόνο εκτέλεσης. Ο αλγόριθμος εκτελείται αναδρομικά, αναζητώντας σε κάθε επανάληψη μεταξύ δύο κόμβων της διαδρομής συντομότερη από την υπάρχουσα διαδρομή. Αν βρεθεί συντομότερη, η διαδρομή «σπάει» στον προηγούμενο κόμβο και οι πληροφορίες (αρχική ακμή, τελική ακμή, deltahash) για το πρώτο κομμάτι διαδρομής αποθηκεύονται στον πίνακα unique_subpaths. Στο δεύτερο κομμάτι της διαδρομής συνεχίζεται αναδρομικά ο έλεγχος για συντομότερες υπο-διαδρομές.

ΕΞΟΔΟΣ: Ανανέωση των τιμών του πίνακα unique_subpaths βάσει των μοναδικών υπο-διαδρομών που προέκυψαν από τον έλεγχο της διαδρομής που ορίστηκε για το SPQ.

ΚΛΗΣΗ: `select unique_shortest_paths2(4587,3521);`
Η συνάρτηση καλείται συνήθως από την συνάρτηση call_unique_subpaths με χρήση της εντολής perform unique_shortest_paths2(integer,integer);

7.3 PHP scripts προετοιμασίας δεδομένων

- **`xml_manip.php`**

  ΕΙΣΟΔΟΣ: 1. Παράμετρος **file**: Διαδρομή XML αρχείου δρομολογήσιμου δικτύου (εξάγεται από την εφαρμογή OSM2Routing.exe).

  ΛΕΙΤΟΥΡΓΙΑ: Ανάγνωση αρχείου XML δρομολογήσιμου δικτύου και εξαγωγή αρχείων TXT με τους κόμβους του δικτύου OSM, τους δρόμους του δικτύου OSM και τη συσχέτιση αυτών. Τα αρχεία έχουν την κατάλληλη μορφοποίηση για εύκολη εισαγωγή στη Βάση Δεδομένων.

  ΕΞΟΔΟΣ: Αρχεία TXT (osm_nodes.txt, osm_ways.txt, osm_nodes_ways.txt) σε delimited format με την πληροφορία κόμβων, δρόμων και συσχέτισης αυτών.

  ΚΛΗΣΗ: `http://localhost/essay/xml_manip.php?file=map-r.osm`
  (Από φυλλομετρητή που εκτελείται στον τοπικό υπολογιστή, όπου βρίσκεται η εγκατάσταση του PHP server και το αρχείο xml_manip.php)

- **`gpx_creator.php`**

  ΕΙΣΟΔΟΣ: 1. Παράμετρος **file**: Διαδρομή TXT αρχείου με τα δεδομένα θέσης των κινούμενων αντικειμένων του συνόλου δεδομένων.

  ΛΕΙΤΟΥΡΓΙΑ: Ανάγνωση TXT αρχείου δεδομένων θέσης και εξαγωγή τροχιών κινούμενων αντικειμένων σε πρότυπο GPX (XML). Το παραγόμενο αρχείο (gpx_dataset_trajectories.gpx) χρησιμοποιείται για την εισαγωγή των τροχιών στην εφαρμογή MatchGPX2OSM που πραγματοποιεί την αντιστοίχιση των τροχιών στις ακμές του δικτύου υποβάθρου.

  ΕΞΟΔΟΣ: Αρχείο gpx_dataset_trajectories.gpx. Περιλαμβάνει τις τροχιές αντικειμένων του συνόλου δεδομένων σε πρότυπο GPX.

  ΚΛΗΣΗ: `http://localhost/essay/gpx_creator.php?file=Trucks.txt`
  (Από φυλλομετρητή που εκτελείται στον τοπικό υπολογιστή, όπου βρίσκεται η εγκατάσταση του PHP server και το αρχείο xml_manip.php)

- **gpx_reader.php**

  ΕΙΣΟΔΟΣ: -

  ΛΕΙΤΟΥΡΓΙΑ: Ανάγνωση αρχείων που έχουν παραχθεί κατά την αντιστοίχιση των τροχιών στις ακμές του χαρτογραφικού υποβάθρου από την εφαρμογή MatchGPX2OSM. Το script διαβάζει ξεχωριστά κάθε αρχείο από την τοποθεσία που δίνεται εντός του κώδικα του script (μεταβλητή $path) και προσθέτει μια εγγραφή σε ένα delimited TXT αρχείο, το οποίο χρησιμοποιείται για τη φόρτωση της αντιστοίχισης στη Βάση Δεδομένων.

  ΕΞΟΔΟΣ: Αρχείο visited_nodes_segments.txt. Περιέχει σε κατάλληλη διαμόρφωση την πληροφορία αντιστοίχισης των τροχιών του συνόλου δεδομένων στις ακμές του δικτυακού υποβάθρου.

  ΚΛΗΣΗ: `http://localhost/essay/gpx_reader.php`
  (Από φυλλομετρητή που εκτελείται στον τοπικό υπολογιστή, όπου βρίσκεται η εγκατάσταση του PHP server και το αρχείο xml_manip.php)

7.4 Διαμόρφωση αρχείων TXT

Στην παρούσα παράγραφο περιγράφεται η δομή των αρχείων TXT που χρησιμοποιούνται για την μαζική είσοδο των δεδομένων στη Βάση Δεδομένων μέσω εντολών SQL COPY. Οι πίνακες περιγραφής πεδίων αποτελούνται από τρεις στήλες, με την πρώτη να περιέχει τον αύξοντα αριθμό του πεδίου, τη δεύτερη μια σύντομη περιγραφή της πληροφορίας που περιλαμβάνεται εντός του πεδίου και την τρίτη ένα μια τιμή που μπορεί να βρίσκεται εντός του πεδίου (παράδειγμα).

Σύνολο δεδομένων (dataset):

- **Trucks.txt**
  ΠΕΔΙΑ:

Πεδίο 1	Αναγνωριστικό τροχιάς	0862
Πεδίο 2	Αναγνωριστικό υπο-τροχιάς	1
Πεδίο 3	Ημερομηνία (ΗΗ/ΜΜ/ΕΕΕΕ)	10/09/2002
Πεδίο 4	Ώρα (ΩΩ:ΛΛ:ΔΔ)	09:15:59
Πεδίο 5	X (WGS 84)	23.845089
Πεδίο 6	Y (WGS 84)	38.018470
Πεδίο 7	X (ΕΓΣΑ 87)	486253.80
Πεδίο 8	Y (ΕΓΣΑ 87)	4207588.10

ΔΙΑΧΩΡΙΣΤΙΚΟ: Ελληνικό ερωτηματικό (;).

ΓΡΑΜΜΗ:
0862;1;10/09/2002;09:15:59;23.845089;38.018470;486253.80;4207588.10

ΦΟΡΤΩΣΗ ΣΤΗ ΒΔ:
```
copy truck_nodes(traj_id, trip_id, date_file,
    time_file, x, y, x_egsa, y_egsa)
from 'C:/Trucks.txt'
with delimiter ';' NULL as '';
```

Δεδομένα OSM υποβάθρου:

- **osm_nodes.txt**

ΠΕΔΙΑ:

Πεδίο 1	OSM ID κόμβου	78695
Πεδίο 2	X (WGS 84)	37.5964742
Πεδίο 3	Y (WGS 84)	23.0709818

ΔΙΑΧΩΡΙΣΤΙΚΟ: Ελληνικό ερωτηματικό (;).

ΓΡΑΜΜΗ: 78695;37.5964742;23.0709818

ΦΟΡΤΩΣΗ ΣΤΗ ΒΔ:
```
copy osm_nodes(osm_id, lat, lon)
from 'C:/osm_nodes.txt'
with delimiter ';'
```

- **osm_ways.txt**

ΠΕΔΙΑ:

Πεδίο 1	Αναγνωριστικό ακμής δρομολογήσιμου δικτύου	-1
Πεδίο 2	OSM ID δρόμου στον οποίο ανήκει η ακμή	4263049
Πεδίο 3	Ταχύτητα	30
Πεδίο 4	Πρόσβαση από οχήματα (OSM tag: accessible)	yes
Πεδίο 5	Δρόμος διπλής κατεύθυνσης (OSM tag: accessible_reverse)	no

ΔΙΑΧΩΡΙΣΤΙΚΟ: Ελληνικό ερωτηματικό (;).

ΓΡΑΜΜΗ: -1;4263049;30;yes;no

ΦΟΡΤΩΣΗ ΣΤΗ ΒΔ:
```
copy osm_ways(routable_osm_id, osm_id, speed,
    accessible, accessible_reverse)
    from 'C:/osm_ways.txt'
    with delimiter ';'
```

- **osm_nodes_ways.txt**

ΠΕΔΙΑ:

Πεδίο 1	Αναγνωριστικό ακμής δρομολογήσιμου δικτύου	-1
Πεδίο 2	OSM ID δρόμου	4263049
Πεδίο 3	OSM ID κόμβου	479796982
Πεδίο 4	Αύξων αριθμός κόμβου για την τρέχουσα ακμή δρομολογήσιμου δικτύου	1

ΔΙΑΧΩΡΙΣΤΙΚΟ: Ελληνικό ερωτηματικό (;).

ΓΡΑΜΜΗ: `-1;4263049;479796982;1`

ΦΟΡΤΩΣΗ ΣΤΗ ΒΔ:
```
copy osm_nodes_ways(routable_osm_id, osm_way_id,
                    osm_node_id, node_no)
                    from 'C:/osm_nodes_ways.txt'
                    with delimiter ';'
```

Αποτελέσματα αντιστοίχισης τροχιών στο χαρτογραφικό υπόβαθρο:

- **osm_visited_segments.txt**

ΠΕΔΙΑ:

Πεδίο 1	Αναγνωριστικό τροχιάς	0862
Πεδίο 2	Αναγνωριστικό υπο-τροχιάς	1
Πεδίο 3	Αναγνωριστικό τμήματος υπο-τροχιάς	110
Πεδίο 4	Αναγνωριστικό κόμβου OSM	597692580
Πεδίο 5	Αναγνωριστικό δρόμου OSM	44113452
Πεδίο 6	Αύξων αριθμός ακμής δικτύου για την αντιστοίχιση του δεδομένου τμήματος υπο-τροχιάς	2

ΔΙΑΧΩΡΙΣΤΙΚΟ: Ελληνικό ερωτηματικό (;).

ΓΡΑΜΜΗ: `0862;1;110;597692580;44113452;2`

ΦΟΡΤΩΣΗ ΣΤΗ ΒΔ:
```
copy osm_visited_segments(traj_id, trip_id,
                segment_id, osm_node_id, osm_way_id, order_id)
                from 'C:/visited_nodes_segments.txt'
                with delimiter ';' NULL as ''
```

7.5 Οδηγίες εγκατάστασης

Στην παρούσα παράγραφο περιγράφεται σε απλά βήματα η διαδικασία εγκατάστασης της εφαρμογής, η οποία διαχωρίζεται στις εξής φάσεις:

- Δημιουργία της Βάσης Δεδομένων (ΒΔ)
- Επεξεργασία δεδομένων χαρτογραφικού υποβάθρου
- Εισαγωγή δεδομένων χαρτογραφικού υποβάθρου στη ΒΔ
- Δημιουργία δρομολογήσιμου δικτύου
- Εισαγωγή δεδομένων θέσεων κινούμενων αντικειμένων
- Αντιστοίχιση θέσεων κινούμενων αντικειμένων στο χαρτογραφικό υπόβαθρο
- Κατασκευή ευρετηρίου NETTRA

Με την επιτυχή ολοκλήρωση των ανωτέρω διαδικασιών δημιουργείται το υπόβαθρο εκτέλεσης των ερωτημάτων SPQ, καθιστώντας δυνατή την εκτέλεσή τους από το χρήστη. Τα αρχεία και οι φάκελοι στους οποίους γίνεται αναφορά στην παρούσα παράγραφο συνοδεύουν την παρούσα εργασία. Οι χρόνοι περάτωσης είναι ενδεικτικοί και επηρεάζονται άμεσα από το μέγεθος των δεδομένων και τις επιδόσεις του υπολογιστικού συστήματος στο οποίο γίνεται η εγκατάσταση.

Δημιουργία Βάσης Δεδομένων

1. Μεταφόρτωση και εγκατάσταση της Postgres 9.3
2. Μεταφόρτωση και εγκατάσταση του PostGIS 2.1 από τον Stack Builder της Postgres 9.3 (το pgRouting είναι ενσωματωμένο στα PostGIS 2.x, αλλά απαιτείται η εντολή create extension pgrouting για τη φόρτωση των συναρτήσεών του)
3. Δημιουργία βάσης δεδομένων (μέσω του περιβάλλοντος pgAdmin III)
4. Εκτέλεση εντολών που περιέχονται στο αρχείο create_tables_functions.sql, που περιέχει τις εντολές για τη δημιουργία των πινάκων και των συναρτήσεων της Βάσης Δεδομένων *(περάτωση σε περίπου 4 sec)*.

Επεξεργασία δεδομένων χαρτογραφικού υποβάθρου

1. Για δημιουργία δρομολογήσιμου δικτύου OSM:

- Μετάβαση στο φάκελο create routable OSM
- Άνοιγμα cmd στην τοποθεσία του create routable OSM
- Εκτέλεση εντολής
  `OSM2Routing_.exe --osm=..\athens_greece.osm --config=sample.config --output=..\map-r.osm` *(περάτωση σε 67 sec)*

  Παράμετροι εντολής:
  - **`--osm:`** διαδρομή αρχείου δεδομένων χάρτη
  - **`--config:`** αρχείο επιλογής κατηγοριών δρόμων
  - **`--output:`** διαδρομή και όνομα αρχείου εξόδου (αρχείο δρομολογήσιμου δικτύου OSM)

2. Δημιουργία αρχείων για φόρτωση δρομολογήσιμου δικτύου στη ΒΔ (απαιτείται Apache με PHP 5+ εγκατεστημένη):

- Φάκελος PHP routable OSM to DB
- Κλήση του αρχείου **xml_manip.php** με παράμετρο **?file=...** το αρχείο δρομολογήσιμου δικτύου που εξήχθη από το προηγούμενο βήμα
  (περάτωση σε 120 sec)
  π.χ.: **localhost/test/xml_manip.php?file=map-r.osm** (αν το map-r.osm βρίσκεται στον ίδιο φάκελο με το xml_manip.php)
- Τα παραγόμενα αρχεία τοποθετούνται στον ίδιο φάκελο με το xml_manip.php και είναι τα εξής:

osm_nodes.txt	πληροφορίες κόμβων του OSM δικτύου
osm_ways.txt	πληροφορίες δρόμων του OSM δικτύου
osm_nodes_ways.txt	πληροφορίες για το δρόμο που ανήκει κάθε κόμβος του OSM δικτύου

Στον φάκελο PHP routable OSM to DB περιέχονται ενδεικτικά αρχεία εισόδου (map-r.osm) και εξόδου (osm_***.txt)

Εισαγωγή δεδομένων χαρτογραφικού υποβάθρου στη ΒΔ

1. Φόρτωση δεδομένων αρχείων .txt στη ΒΔ:

- Postgres PgAdmin: Εκτέλεση εντολών σε παράθυρο ερωτήματος SQL (το *folder_path* αντικαθίσταται με το μονοπάτι του φακέλου που περιέχει τα αρχεία osm_***.txt):
  - Εισαγωγή δεδομένων OSM οντοτήτων:

```
copy osm_nodes(osm_id, lat, lon)
from '*folder_path*/osm_nodes.txt'
with delimiter ';'
```
(περάτωση σε 35 sec)

```
copy osm_ways(routable_osm_id, osm_id, speed, accessible,
      accessible_reverse)
from '*folder_path*/osm_ways.txt'
with delimiter ';'
```
(περάτωση σε 8 sec)

```
copy osm_nodes_ways(routable_osm_id, osm_way_id, osm_node_id,
      node_no)
from '*folder_path*/osm_nodes_ways.txt'
with delimiter ';'
```
(περάτωση σε 118 sec)

  - Δημιουργία γεωμετριών χαρτογραφικού υποβάθρου:
    -Κόμβοι (osm_nodes):

```
update osm_nodes set
geom=ST_GeomFromText('POINT('||lon||' '||lat||')');
```
(περάτωση σε 44 sec)

    -Δρόμοι (osm_ways):

```
select osm_routable_segments_geometry();
```
(περάτωση σε 82 sec)

Δημιουργία δρομολογήσιμου δικτύου

1. Δημιουργία πίνακα δρομολογήσιμου δικτύου και προετοιμασία pgrouting:

- Κλήση της συνάρτησης δημιουργίας δρομολογήσιμου δικτύου με χρήση της εντολής **`select create_routable_linestrings();`**
  (περάτωση σε 95 sec)
- Εκτέλεση των εντολών:
  - Δημιουργία στηλών κι ευρετηρίων που θα χρησιμοποιηθούν από το pgrouting:

```
ALTER TABLE osm_routable_network
ADD COLUMN "source" integer;

ALTER TABLE osm_routable_network
ADD COLUMN "target" integer;

CREATE INDEX osm_routable_network_source_idx
ON osm_routable_network USING btree (source);

CREATE INDEX osm_routable_network_target_idx
ON osm_routable_network USING btree (target);
```

  - Δημιουργία τοπολογίας δικτύου από το pgrouting:

```
SELECT pgr_createTopology('osm_routable_network',
0.00001, 'geom', 'routable_osm_id');
```

    (περάτωση σε 561 sec)

  - Δημιουργία ευρετηρίου για το πεδίο αναγνωριστικού pgrouting:

```
CREATE INDEX pgr_vertices_id_idx
ON osm_routable_network_vertices_pgr USING btree (id);
```

  - Στήλη για αντιστοίχιση αναγνωριστικού κόμβου pgrouting με αναγνωριστικό κόμβου osm:

```
ALTER TABLE osm_routable_network_vertices_pgr
ADD COLUMN osm_id bigint;
```

  - Δημιουργία ευρετηρίου για το πεδίο αναγνωριστικού osm:

```
CREATE INDEX pgr_vertices_osm_id_idx
ON osm_routable_network_vertices_pgr USING btree (osm_id);
```

  - Δημιουργία ευρετηρίου για το πεδίο geom:

```
CREATE INDEX osm_routable_network_vertices_pgr_the_geom_idx
ON osm_routable_network_vertices_pgr USING gist (the_geom);
```

  - Συσχέτιση κόμβων του pgrouting με τους κόμβους OSM υποβάθρου:

```
select pgr_set_vertices_osm_id();
```

    (Το συγκεκριμένο βήμα συνήθως έχει μεγάλο χρόνο εκτέλεσης. Μπορεί να παραληφθεί, αν οι συναρτήσεις SPQ τροποποιηθούν ώστε να υπολογίζουν SPQ από τους κόμβους του δρομολογήσιμου δικτύου) *(περάτωση σε 27590 sec)*

Με την ολοκλήρωση των παραπάνω βημάτων, το δρομολογήσιμο χαρτογραφικό υπόβαθρο έχει φορτωθεί στη Βάση Δεδομένων και έχει εκτελεστεί προετοιμασία του για να χρησιμοποιηθεί στην εκτέλεση SPQs.

Εισαγωγή δεδομένων θέσεων κινούμενων αντικειμένων

1. Εισαγωγή θέσεων στη ΒΔ με copy στον πίνακα truck_nodes και εκτέλεση της create_truck_segments() για δημιουργία των τροχιών:

- Εισαγωγή θέσεων κινούμενων αντικειμένων στη ΒΔ:

```
copy truck_nodes(traj_id, trip_id, date_file, time_file, x, y,
x_egsa, y_egsa)
from 'C:/Users/STAMATIS/Desktop/Trucks.txt'
with delimiter ';' NULL as '';
```

- Διόρθωση πεδίου ημερομηνίας:

```
update truck_nodes
set time_stamp = (split_part(date_file, '/', 3)||'-
'||split_part(date_file, '/', 2)||'-'||split_part(date_file, '/',
1)||' '||time_file)::timestamp without time zone;
```

- Δημιουργία τμημάτων τροχιών κινούμενων αντικειμένων:

```
select create_truck_segments();
```

Αντιστοίχιση θέσεων κινούμενων αντικειμένων στο χαρτογραφικό υπόβαθρο

1. Μετατροπή τροχιών συνόλου δεδομένων σε αρχείο GPX (Φάκελος create dataset GPX). Παράγεται το αρχείο gpx_dataset_trajectories.gpx

- Κλήση php script **`gpx_creator.php`** με παράμετρο το μονοπάτι και το όνομα του αρχείου που περιέχει τις τροχιές του συνόλου δεδομένων
  π.χ. **`http://localhost/htdocs/gpx_creator.php?file=Trucks.txt`**

2. Χρήση της εφαρμογής MatchGPX2OSM για αντιστοίχιση τροχιών στο δρομολογήσιμο δίκτυο υποβάθρου. (Φάκελος create map matched OSM)

- Άνοιγμα παραθύρου εντολών (command prompt).
- Εκτέλεση εντολής

```
MatchGPX2OSM --osm=map-r.osm --gpx=gpx_dataset_trajectories.gpx --
output=res
```

  `-osm:` αρχείο δρομολογήσιμου δικτύου OSM
  `-gpx:` αρχείο τροχιών συνόλου δεδομένων σε πρότυπο GPX
  `-output:` τα αποτελέσματα εξάγονται στον φάκελο res (ένα αρχείο OSM για κάθε τμήμα τροχιάς)

3. Ανάγνωση αρχείων φακέλου res για εξαγωγή αρχείου TXT για φόρτωση της αντιστοίχισης στη Βάση Δεδομένων. Παράγεται το αρχείο visited_nodes_-segments.txt

- Κλήση php script **gpx_reader.php** με παράμετρο (μεταβλητή $path εντός του κώδικα) το μονοπάτι και το όνομα του φακέλου αποτελεσμάτων της εφαρμογής MatchGPX2OSM
  π.χ. **http://localhost/htdocs/gpx_reader.php**

4. Φόρτωση του visited_nodes_segments.txt στη Βάση Δεδομένων.
Εκτέλεση ερωτήματος SQL για φόρτωση στη ΒΔ:

```
copy osm_visited_segments(traj_id, trip_id, segment_id,
osm_node_id, osm_way_id, order_id)
from 'C:/Users/STAMATIS/Desktop/visited_nodes_segments.txt'
with delimiter ';' NULL as '';
```

Με την ολοκλήρωση των παραπάνω βημάτων ολοκληρώνεται η πρώτη φάση αντιστοίχισης των τροχιών του συνόλου δεδομένων στο χαρτογραφικό δικτυακό υπόβαθρο. Σε επόμενα βήματα ακολουθεί επεξεργασία εντός της Βάσης Δεδομένων για την κατασκευή του ευρετηρίου NETTRA.

Κατασκευή ευρετηρίου NETTRA

Για την κατασκευή του ευρετηρίου NETTRA, εκτελείται μια σειρά από pl/pgsql συναρτήσεις που επεξεργάζονται κατάλληλα τις τιμές αντιστοίχισης των δεδομένων στο χαρτογραφικό υπόβαθρο, που αρχικά βρίσκονται στον πίνακα osm_visited_segments, μετά από την εισαγωγή τους στη Βάση Δεδομένων.

1. Επεξεργασία τιμών αντιστοίχισης και διορθώσεις: Κλήση της map_matching-_strict().

```
select map_matching_strict();
```

2. Περαιτέρω διορθώσεις και δημιουργία πίνακα NETTRA: Κλήση της create-_visited_segments().

```
select create_visited_segments();
```

- Δημιουργία ευρετηρίων στον πίνακα visited_segments (NETTRA):

```
CREATE INDEX load_trajectories_idx
ON visited_segments
USING btree
(routable_osm_id, time_enter, traj_id, trip_id,
      time_leave, s_hash);

CREATE INDEX visited_segments_id_index
ON visited_segments
USING btree
(id);
```

```
CREATE INDEX visited_segments_routable_index
ON visited_segments
USING btree
(routable_osm_id);
```

3. Προσθήκη χρονικών στιγμών εισόδου και εξόδου κινουμένων αντικειμένων στις ακμές του δικτύου: Κλήση της add_timestamps().

```
select add_timestamps();
```

4. Προσθήκη βαρών στις ακμές του δρομολογήσιμου δικτύου, υπολογισμός αθροιστικών βαρών για κάθε τροχιά του συνόλου δεδομένων: Κλήση της assign_weights_to_routable_segments().

```
select assign_weights_to_routable_segments();
```

Με την επιτυχή ολοκλήρωση της εκτέλεσης των παραπάνω βημάτων, το ευρετήριο NETTRA έχει δημιουργηθεί επιτυχώς, ενώ έχουν εγκατασταθεί στη Βάση Δεδομένων του συστήματος όλες οι απαραίτητες συναρτήσεις για την κατασκευή και εκτέλεση των Ερωτημάτων Ακριβούς Τροχιάς (SPQs).

7.6 Παραδείγματα εκτέλεσης ερωτημάτων SPQ

1. Προσεγγιστικό SPQ με βάρος μήκους ακμής:

- Απλό SPQ
  ```
  select * from spq1(1753749971, 279591283);
  ```
- SPQ για μετέπειτα απεικόνιση αποτελεσμάτων
  ```
  select * from spq1_show(1753749971, 279591283);
  ```
- SPQ για μετέπειτα απεικόνιση αποτελεσμάτων (με βάρος λογαρίθμου πρώτου αριθμού)
  ```
  select * from spq1_show_primehash(1753749971, 279591283);
  ```

2. Ακριβής προσέγγιση SPQ:

- Απλό SPQ
  ```
  select * from spq2(1753749971, 279591283);
  ```
- SPQ για μετέπειτα απεικόνιση αποτελεσμάτων
  ```
  select * from spq2_show(1753749971, 279591283);
  ```
- Απλό SPQ (με βάρος λογαρίθμου πρώτου αριθμού)
  ```
  select * from spq2_primehash(bigint, bigint);
  ```

- SPQ για μετέπειτα απεικόνιση αποτελεσμάτων (με βάρος λογαρίθμου πρώτου αριθμού)

```
select * from spq2_show_primehash(bigint, bigint);
```

3. «Πρακτική» προσέγγιση SPQ με βάρος λογαρίθμου πρώτου αριθμού

- Απλό SPQ

```
select * from spq3(1753749971, 279591283);
```

- SPQ για μετέπειτα απεικόνιση αποτελεσμάτων

```
select * from spq3_show(1753749971, 279591283);
```

4. Βελτιωμένη ακριβής προσέγγιση SPQ

- Κλήση συνάρτησης δημιουργίας βελτιωμένου ακριβούς SPQ

```
select * from call_unique_subpaths(1753749971, 279591283);
```

- Αλγόριθμος UniqueSubPaths (καλείται από την call_unique_subpaths)

```
select unique_shortest_paths2(4587,3521);
```

- Κατασκευή ερωτήματος μετά από επεξεργασία από τον UniqueSubPaths (καλείται από την call_unique_subpaths)

```
select * from spq2_from_usp();
```

5. Χωροχρονική προσέγγιση SPQ:

```
select * from process_spq(801467980, 985946206, true, '2000-12-31
00:00:00', '2015-12-31 00:00:00');
```