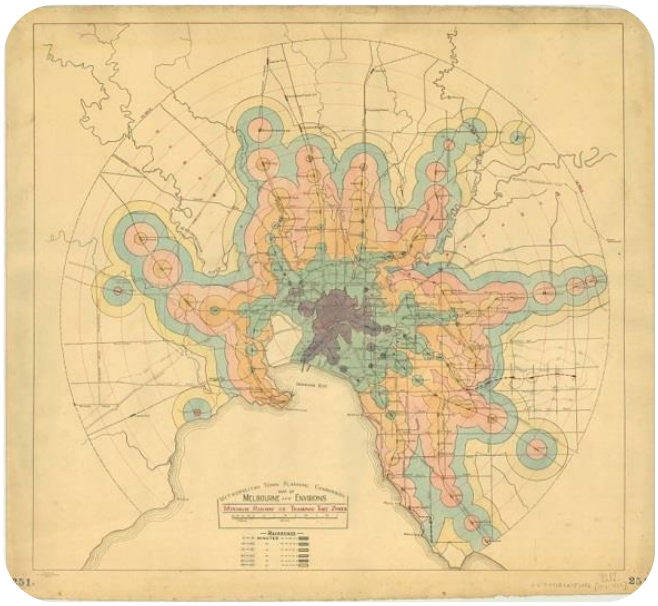
ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΑΓΡΟΝΟΜΩΝ ΚΑΙ ΤΟΠΟΓΡΑΦΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΟΜΕΑΣ ΤΟΠΟΓΡΑΦΙΑΣ

Δημιουργία αλγορίθμου και χαρτογραφική απόδοση του χρόνου μετακίνησης στο λεκανοπέδιο Αττικής με Δημόσια Μέσα Μεταφοράς



ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

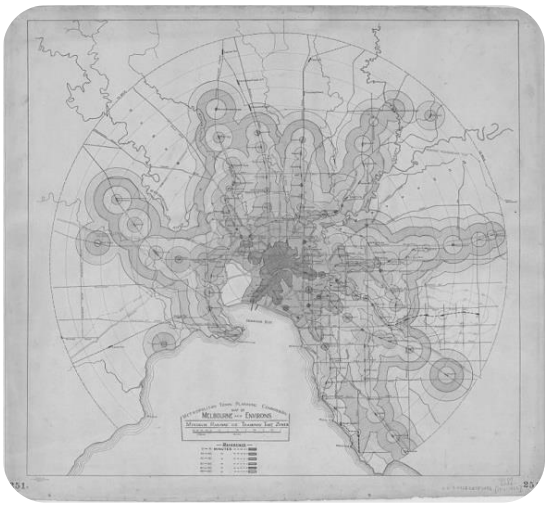
της

Στυλιανής Γκίρτσου

Επιβλέπων καθηγητής : Λύσανδρος Τσούλος

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΑΓΡΟΝΟΜΩΝ ΚΑΙ ΤΟΠΟΓΡΑΦΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΟΜΕΑΣ ΤΟΠΟΓΡΑΦΙΑΣ

Δημιουργία αλγορίθμου και χαρτογραφική απόδοση του χρόνου μετακίνησης στο λεκανοπέδιο Αττικής με Δημόσια Μέσα Μεταφοράς



ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

της

Στυλιανής Γκίρτσου

Επιβλέπων Καθηγητής: Λύσανδρος Τσούλος

.....	.....	.....
Λύσανδρος Τσούλος Καθηγητής ΕΜΠ	Κων/νος Κεπαπτζόγλου Επίκουρος Καθηγητής ΕΜΠ	Αντελίνα Σκοπελίτη Διδάκτορ ΕΜΠ – Ε.Ε.ΔΙ.Π.

Οκτώβρης, 2017

Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι να δημιουργηθούν χάρτες με καμπύλες χρονοαπόστασης στο δίκτυο Μέσων Μαζικής Μεταφοράς (ΜΜΜ) της Αθήνας με αφετηρία την πλατεία της Ομόνοιας. Στόχος είναι τα αποτελέσματα της ανάλυσης να είναι ρεαλιστικά και οι χρόνοι μετακίνησης λογικοί. Για το λόγο αυτό θα πρέπει να ληφθούν υπ' όψιν τα χρονοδιαγράμματα των ΜΜΜ και ο χρόνος που απαιτείται ώστε το επιβατικό κοινό να μεταβεί από τη μία στάση στην επόμενη. Ιδιαίτερου ενδιαφέροντος είναι ο ορισμός του μέγιστου χρόνου πεζής μετακίνησης μεταξύ των στάσεων όταν εκτελείται δρομολόγηση με μέσα πολλαπλής τροχιάς.

Ισοχρονικές καμπύλες χρονοαπόστασης καλούνται οι καμπύλες που ορίζουν ένα πολύγωνο πάνω στον χάρτη το οποίο περιλαμβάνει όλες τις περιοχές οι οποίες είναι προσβάσιμες μέσα σε ένα ορισμένο χρονικό διάστημα από ένα αρχικό σημείο. Ανάλογα με το δίκτυο στο οποίο γίνεται μία τέτοια ανάλυση αλλάζουν οι διαδικασίες και οι παράμετροι δημιουργίας των ισοχρονικών καμπυλών. Οι περισσότερες εργασίες που έχουν δημοσιευτεί μέχρι σήμερα αφορούν στα οδικά δίκτυα.

Στα πλαίσια της διπλωματικής αναπτύχθηκε πρωτότυπος κώδικας που επιλύει το πρόβλημα των ισοχρονικών καμπυλών για τα μέσα μαζικής μεταφοράς βασισμένος στον αλγόριθμο Dijkstra. Αντιμετωπίστηκαν οι ιδιαιτερότητες του συγκεκριμένου δικτύου όπως οι πολλαπλές ακμές ίδιου κόστους που μπορεί να περιέχονται ανάμεσα σε δύο κόμβους, το επιπρόσθετο κόστος που υπεισέρχεται σε περιπτώσεις αλλαγής κωδικού γραμμής (μετεπιβίβασης), το γεγονός ότι οι κόμβοι (στάσεις) δεν βρίσκονται επί του δικτύου αλλά παράλληλα σ' αυτό και άλλες που προέκυψαν στη διάρκεια της εργασίας.

Επιπλέον σχεδιάστηκε και υλοποιήθηκε μία διαδικτυακή χαρτογραφική εφαρμογή για την απεικόνιση των δημιουργημένων ισοχρονικών καμπυλών. Ο χρήστης έχει τη δυνατότητα να ορίσει το χρόνο μετακίνησης που επιθυμεί να απεικονίσει η ισοχρονική καμπύλη, καθώς και συμπληρωματικά θεματικά επίπεδα που συμπληρώνουν τον χάρτη όπως το χαρτογραφικό υπόβαθρο Open Street Map, το δίκτυο του Αττικό Μετρό και το δίκτυο των λεωφορειακών γραμμών με τις στάσεις τους. Μέσω λειτουργιών που έχουν εισαχθεί παρέχονται οι βασικές χαρακτηριστικές ιδιότητες των επιπέδων αυτών.

Η υποδομή δημιουργίας και υποστήριξης του αλγορίθμου και της εφαρμογή αναπτύχθηκε χρησιμοποιώντας αποκλειστικά τεχνολογίες και Ελεύθερο Λογισμικό / Λογισμικό Ανοιχτού Κώδικα (ΕΛ/ΛΑΚ) προσανατολισμένο στα γεωχωρικά δεδομένα όπως η βάση δεδομένων Postgres, η επέκτασή της PostGis, η βιβλιοθήκη pgRouting, το ΣΓΠ QGIS, η γλώσσα Python, η βιβλιοθήκη OpenLayers. Η εφαρμογή είναι συμβατή με τις τυποποιήσεις και προδιαγραφές του OGC, συνεπώς διανέμει γεωχωρικό περιεχόμενο στον Παγκόσμιο Ιστό μέσω των ευρέως αποδεκτών υπηρεσιών Web Map Service (WMS) και Web Feature Service (WFS).

Abstract

The main objective of this study is to create maps that show curves of equal travel times through the Public Transit Network of Athens , the Metro, Bus and Tram lines, starting from Omonoia Square. We aim to produce a realistic analysis of the results based on a reasonable estimate of travel times. For this reason, it is necessary to take into account the timetables provided for each line and mode of transport provided by the Transit Network of Athens as well as the time passengers will need to change from one line or mode of transport to another. The definition of maximum walking time between the stations is of great interest in multimodal routing as it depends on the characteristics of the network (density, territorial coverage etc.) and changes decisively the time cost of the routes .

An isochrone is defined as a polygon on the map that includes all the areas that can be accessed within a certain amount of time from an initial point. The processes and the parameters of isochrones' creation depend on the network based on which the analysis is conducted. Most published studies have applied this specific analysis on road networks.

As part of this study an original algorithm was developed for the calculation of isochrones in multimodal networks. This algorithm is based on Dijkstra' s routing algorithm. The specific characteristics of the studied network such as multiple edges (parts of bus routes) of the same time cost that may be located between two nodes (stations), the additional costs involved in cases of changing the type of transit line, the fact that the stations (nodes) do not lie on the network (edges) as it should be for the analysis through shortest path algoritmhs and other considerations that emerged during the procedure have been indexed and addressed.

Additionally, an online cartographic application was designed and implemented to illustrate the created isochrones. The user interface will allow the user to have the option of selecting the desired travel time. The user will also have the option to enable multiple layers on the map which will display the map of the corresponding mode of transport. The basemap used for the application is the OSM which available free of charge. The Metro and bus networks including all their stops were added to the map. Through functions added to the application, popup windows that contain the attributes of the layer are displayed on the mouse click on the layer.

The Algorithm and Application Creation and Support Infrastructure was developed using exclusive Free / Open Source Software packages oriented at geospatial data such as Postgres database, PostGis extension, pgRouting library, QGIS, Python programming language, OpenLayers' library. The application is compliant with OGC standards and specifications, and therefore distributes geospatial content to the Web via widely accepted Web Map Service (WMS) and Web Feature Service (WFS).

Ευχαριστίες

Καταρχήν θα ήθελα να ευχαριστήσω τον κ. Λύσανδρο Τσούλο, καθηγητή και επιβλέποντα της παρούσας εργασίας, για τη δυνατότητα που μου προσέφερε να ασχοληθώ με ένα ιδιαιτέρως ενδιαφέρον επιστημονικό αντικείμενο, για τη διαρκή υποστήριξή του και την αμέριστη κατανόηση που έδειξε κατά τη διάρκεια εκπόνησης της παρούσας εργασίας. Ιδιαίτερες ευχαριστίες οφείλω στην κυρία Αντελίνα Σκοπελίτη, Διδάκτωρ ΕΜΠ – ΙΔΑΧ για την πολύτιμη βοήθεια και καθοδήγησή της και για το χρόνο που μου αφιέρωσε κατά τη διάρκεια εκπόνησης της διπλωματικής. Επίσης ευχαριστώ τον κύριο Κεπτζόγλου για την πολύτιμη βοήθειά του στην ανεύρεση των δεδομένων της εργασίας.

Χρωστάω ένα μεγάλο ευχαριστώ στο Δημήτρη για τη βοήθεια και το χρόνο που μου προσέφερε και για το γεγονός ότι με εισήγαγε στον κόσμο της Python. Επίσης θα ήθελα να ευχαριστήσω το Νίκο, τη Μαρία, την Παναγιώτα, τον Αντρέα και τη Χριστίνα για τη στήριξη και την υπομονή τους.

Τέλος ευχαριστώ την οικογένειά μου για τη στήριξη και την εμπιστοσύνη τους.

Περιεχόμενα

Κατάλογος Εικόνων

Κατάλογος πινάκων

Κατάλογος Διαγραμμάτων ροής

Κατάλογος σχημάτων

1. ΕΙΣΑΓΩΓΗ

Οι τεχνολογικές εξελίξεις κατά τα τέλη του 20ου αιώνα έχουν μεταβάλλει άρδην τη φύση και τα προϊόντα της Χαρτογραφίας και των Συστημάτων Γεωγραφικών Πληροφοριών (GIS). Η ραγδαία ανάπτυξη και η ευρεία χρήση των ηλεκτρονικών υπολογιστών και του διαδικτύου, ιδιαίτερα του Παγκόσμιου Ιστού (Web), έχουν επηρεάσει σημαντικά και έχουν καθορίσει νέα πρότυπα και προδιαγραφές σε όλα τα στάδια της επεξεργασίας και ανάλυσης γεωχωρικών δεδομένων και της χαρτογραφικής παραγωγής.

Οι διαδικτυακές χαρτογραφικές υπηρεσίες, τα συστήματα πλοήγησης και άλλες εφαρμογές σχεδιασμού διαδρομών και εντοπισμού θέσης έχουν αποκτήσει ευρεία χρήση λόγω της σημαντικής προόδου των αλγορίθμων συντομότερων διαδρομών.

Το πρόβλημα εύρεσης της συντομότερης διαδρομής (shortest path problem) σε δίκτυα οδικά, σιδηροδρομικά, δίκτυα κοινής ωφέλειας, δίκτυα τηλεπικοινωνιών κ.λπ., συνιστά ένα από τα θεμελιώδη προβλήματα της σύγχρονης επιστήμης των υπολογιστών και αφορά ένα πλήθος εφαρμογών πλοήγησης και σχεδιασμού δικτύων.

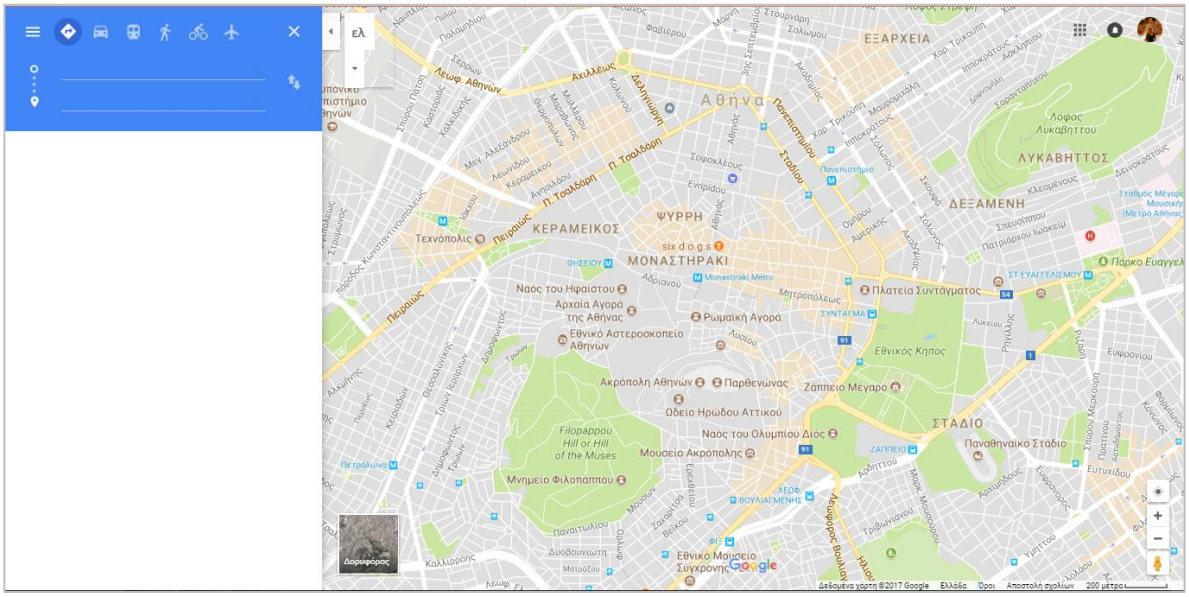
Η διαχείριση των χωρικών δεδομένων, η μοντελοποίησή τους, η χαρτογραφική αναπαράστασή τους και η επεξεργασία τους μέσα από την εφαρμογή αντίστοιχων αλγόριθμων δρομολόγησης συνιστούν μερικές από τις βασικότερες παραμέτρους που υπεισέρχονται στις διαδικασίες σχεδιασμού εφαρμογών δρομολόγησης. Η αποτελεσματική διαχείριση και ενσωμάτωση των παραπάνω παραμέτρων σε ολοκληρωμένες εφαρμογές δρομολόγησης απαιτεί συνήθως τη συνδυασμένη χρήση υπολογιστικών εργαλείων, όπως τα Συστήματα Διαχείρισης Χωρικών Βάσεων Δεδομένων, οι διαδικτυακές χαρτογραφικές υπηρεσίες κ.λπ., προκειμένου να καταστεί δυνατή η πολυδιάστατη διαχείριση των χωρικών δεδομένων που αφορούν τα στοιχεία και τα χαρακτηριστικά του εκάστοτε χωρικού δικτύου.

1.1. Αντικείμενο διπλωματικής

Τα προηγούμενα χρόνια, οι διαδραστικοί διαδικτυακοί χάρτες έγιναν ένα πολύ διαδεδομένα εργαλείο για τον σχεδιασμό διαδρομών όλων των ειδών. Σήμερα κάθε άνθρωπος με πρόσβαση στο διαδίκτυο μπορεί εύκολα να βρει τις διαθέσιμες διαδρομές όταν ταξιδεύει από ένα σημείο σε ένα άλλο. Οι ιστοσελίδες που προσφέρουν δρομολόγηση χρησιμοποιούν συνήθως αλγορίθμους συντομότερης διαδρομής (SP). Το πιο χαρακτηριστικό παράδειγμα τέτοιων εργαλείων είναι οι υπηρεσίες που προσφέρει η Google μέσω του Google Maps (*https://maps.google.com/)*. Το GoogleMaps είναι μία διαδικτυακή υπηρεσία χαρτογράφησης που αναπτύχθηκε από την Google. Προσφέρει

δορυφορικές εικόνες, οδικούς χάρτες, θέαση πανοραμικών χαρτών 360^{o} (Street View), συνθήκες κυκλοφορίας σε πραγματικό χρόνο (Google Traffic) και προγραμματισμό διαδρομών για μετακινήσεις μέσω αυτοκινήτου, ποδηλάτου (σε δοκιμαστική έκδοση), δημόσιων συγκοινωνιών ή και για πεζή μετακίνηση (Εικόνα 1). Ο χρήστης ορίζει το σημείο αφετηρίας και το σημείο προορισμού πληκτρολογώντας την οδό ή την περιοχή ή κάποιο σημείο ενδιαφέροντος. Ύστερα επιστρέφονται οι πληροφορίες της χρονικής διάρκειας που απαιτεί η διάσχιση της συγκεκριμένης διαδρομής, της απόστασης που υφίσταται μεταξύ των δύο σημείων του δικτύου, του τρόπου μετάβασης στον επιθυμητό προορισμο κ.ά.

Για αρκετό καιρό ήταν δυνατό να υπολογιστεί η συντομότερη διαδρομή χρησιμοποιώντας ένα μόνο μέσο μεταφοράς. Πρόσφατα όμως δόθηκε η δυνατότητα να υπολογιστεί συνδυάζοντας διαφορετικά μέσα μεταφοράς στην ίδια διαδρομή (multimodal routing), ενώ κάποια συστήματα συνδυάζουν και δεδομένα χρονοδιαγραμμάτων.



Εικόνα 1 : Η διεπαφή χρήστη που προσφέρει το Google Maps για δρομολόγηση

Λιγότερο γνωστό αλλά αλγοριθμικά πολύ ενδιαφέρον είναι να βρεθεί η απάντηση στο ως πού μπορεί κάποιος να ταξιδέψει σε δεδομένα χρόνο, ξεκινώντας από δεδομένο σημείο. Το αποτέλεσμα μιας τέτοιας ανάλυσης μπορεί να απεικονιστεί με ένα είδος ισαριθμικών καμπύλων που καλείται ισοχρονική καμπύλη.

Ένα από τα σημαντικότερα προτερήματα των ισοχρονικών καμπυλών είναι ότι μπορούν να χρησιμοποιηθούν για αναλύσεις προσβασιμότητας όλων των ειδών. Μπορούν να αποτελέσουν βοηθητικό εργαλείο σε ποικίλα πεδία, όπως στον αστικό σχεδιασμό και στη διαχείριση έκτακτων καταστάσεων.

Αντικείμενο της παρούσας εργασίας είναι η δημιουργία ενός αλγοριθμικού συστήματος για τον γρήγορο υπολογισμό ισοχρονικών καμπυλών στο δημόσιο δίκτυο μεταφορών της Αθήνας. Ιδιαίτερη έμφαση δίνεται στον υπολογισμό των χρονικά συντομότερων διαδρομών μεταξύ δύο σημείων του δικτύου, καθώς η διαδικασία αυτή έχει αρκετές ιδιαιτερότητες όταν αφορά δίκτυο μέσων μαζικής μεταφοράς. Επιπλέον ένα σημαντικό μέρος της εργασίας αποτέλεσε η διαδικασία οπτικοποίησης το εν λόγω καμπυλών και η παρουσίασή τους μέσω χαρτογραφικής διαδικτυακής εφαρμογής.

1.2. Οργάνωση κειμένου

Η εργασία οργανώνεται σε έξι ενότητες.

Η **πρώτη** ενότητα αποτελείται από τη εισαγωγή, στην οποία ο αναγνώστης ενημερώνεται για τη φύση της εργασίας, τα κίνητρα υλοποίησής της και τα βασικά στοιχεία που τη συντελούν. Επιπλέον παρέχεται το παρόν εδάφιο στο οποίο περιγράφεται ο δομή του συνόλου της εργασίας με τα βασικά χαρακτηριστικά κάθε κεφαλαίου.

Στην **δεύτερη** ενότητα περιγράφεται το εννοιολογικό πλαίσιο και οι βασικοί ορισμοί που αφορούν στη θεωρία των γράφων. Περιγράφονται οι τύποι γράφων καθώς και οι σημαντικότερες ιδιότητες που διακρίνουν τους γράφους διάσχισης ακμών. Επιπλέον παρατίθενται οι βασικότεροι αλγόριθμοι ελάχιστης διαδρομής με τις απαιτούμενες παραμέτρους και τα βασικά χαρακτηριστικά τους. Η επίλυση του προβλήματος της συντομότερης διαδρομής ανάγεται στην επίλυση προβλήματος σε γράφο. Κατόπιν παρουσιάζονται κάποιοι αλγόριθμοι που έχουν αναπτυχθεί πειραματικά για τη δημιουργία ισοχρονικών καμπυλών και διαδικτυακές εφαρμογές που έχουν αναπτυχθεί με σκοπό την παρουσίασή τους. Τέλος εισάγεται η περιοχή μελέτης της παρούσας εργασίας και αναλύονται οι ιδιότητες και τα ιδιαίτερα χαρακτηριστικά της.

Στο **τρίτο** κεφάλαιο εισάγεται το πρόβλημα της δρομολόγησης για μέσα πολλαπλής τροχιάς και της δημιουργίας ισοχρονικών καμπυλών σε τέτοια δίκτυα. Αρχικά αναλύονται τα χαρακτηριστικά των δεδομένων που διατέθηκαν από τον ΟΑΣΑ, τα προβλήματα που προέκυψαν και η διαδικασία επίλυσής τους. Ύστερα περιγράφονται τα εργαλεία που χρησιμοποιήθηκαν σε αυτή τη φάση και οι βασικές τους ιδιότητες και δυνατότητες. Ακολουθεί η διαδικασία δόμησης του γράφου του δικτύου μέσω των εργαλείων αυτών. Έπειτα αναλύονται τα προβλήματα που οδήγησαν στη δημιουργία του νέου αλγορίθμου, ο νέος αλγόριθμος και η μεθοδολογία που ακολουθείται με αυτόν. Τέλος, ακολουθεί η διαδικασία επιλογής της μεθόδου σχεδιασμού και οπτικοποίησης των χρόνων διαδρομής. Το κεφάλαιο αυτό αποτελεί τον πυρήνα της εργασίας.

Στο **τέταρτο** κεφάλαιο εξετάζονται τα πρότυπα και οι μορφές γεωχωρικών δεδομένων στον Παγκόσμιο Ιστό, ενώ αναλύονται οι τεχνολογίες, οι υπηρεσίες και τα συστήματα λογισμικού που εμπλέκονται στην ανάπτυξη διαδικτυακών γεωχωρικών εφαρμογών.

Η **πέμπτη** ενότητα αφορά στη σχεδίαση της διαδικτυακής εφαρμογής. Περιγράφονται αναλυτικά οι δυνατότητες που προσφέρει η βιβλιοθήκη OpenLayers και ο τρόπος με τον οποίο εντάχθηκαν στην εφαρμογή. Εξετάζεται η διαδικασία δημιουργίας και μορφοποίησης της ιστοσελίδας και τα εργαλεία που αξιοποιήθηκαν. Τέλος γίνεται εκτενής αναφορά στη γλώσσα JavaScript, καθώς μέσω αυτής ορίζονται οι λειτουργίες που εκτελούνται στο περιβάλλον της ιστοσελίδας.

Η **έκτη** ενότητα αποτελεί την αξιολόγηση του συνόλου της εργασίας και προτάσεις για μελλοντικές βελτιώσεις.

2. ΘΕΩΡΗΤΙΚΟ ΠΛΑΙΣΙΟ ΚΑΙ ΣΧΕΤΙΚΕΣ ΕΡΓΑΣΙΕΣ

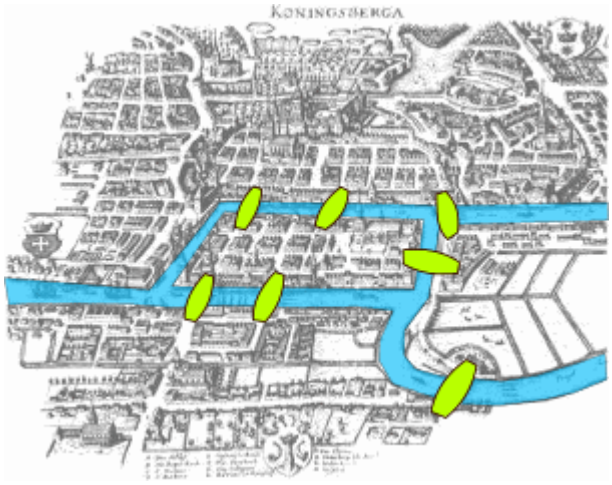
Ένας ισοχρονικός χάρτης στη Χαρτογραφία, είναι ένας χάρτης που απεικονίζει περιοχές που συνδέονται με μία ισοχρονική καμπύλη. Η ισοχρονική καμπύλη ορίζεται ως μια γραμμή σχεδιασμένη στο χάρτη με τρόπο ώστε να συνδέει σημεία στα οποία κάτι συμβαίνει ή φτάνει την ίδια στιγμή. Στο σχεδιασμό των μεταφορών, οι ισοχρονικοί χάρτες χρησιμοποιούνται για να απεικονίσουν περιοχές με ίση διάρκεια ταξιδιού από μία κοινή αφετηρία.

Έχει γίνει αρκετή έρευνα πάνω στον υπολογισμό τους και επίσης έχουν αναπτυχθεί κάποιες εφαρμογές που υπολογίζουν και παρουσιάζουν τις ισοχρονικές καμπύλες. Στο κεφάλαιο αυτό αναφέρουμε κάποιες θεωρητικές εργασίες σχετικές με τις ισοχρονικές καμπύλες χρονοαπόστασης καθώς και κάποιες από τις εφαρμογές υπολογισμού τους που έχουν υλοποιηθεί.

2.1. Θεωρία των γράφων

Οι βάσεις της θεωρίας των γράφων τέθηκαν από την επιστήμη των διακριτών μαθηματικών στο πρώτο μισό περίπου του 18^{ου} αιώνα. Η διαχρονική εξέλιξη και ανάπτυξη του συγκεκριμένου πεδίου των μαθηματικών κατέστησε τον γράφο μια δομή, η οποία υιοθετείται πλέον από διαφορετικούς επιστημονικούς κλάδους καθώς είναι κατάλληλη για την αναπαράσταση πάσης φύσεως δικτύων και συστημάτων τα στοιχεία των οποίων συνδέονται μεταξύ τους με σχέσεις αλληλεπίδρασης.

Για την ιστορία της θεωρίας γράφων θεωρείται σημαντική η μελέτη του Λέοναρντ Όιλερ για τις *Επτά Γέφυρες του Κένιγκσμπεργκ* το 1736 (Biggs N. et al.,1986). Η συγκεκριμένη δημοσίευση, όπως και εκείνη που γράφτηκε από τον Γάλλο χημικό Αλεξάντρ-Τεοφίλ Βαντερμόντ (Alexandre-Théophile Vandermonde) στο μαθηματικό πρόβλημα του Ίππου στη σκακιέρα που κατευθύνεται με την *ανάλυση θέσης* που εισήγαγε ο Γκότφριντ Βίλχελμ Λάιμπνιτς. Ο τύπος του Όιλερ, σχετικά με τον αριθμό των ακμών, των κορυφών και των εδρών ενός κυρτού πολυέδρου μελετήθηκε από τον Ωγκυστέν-Λουί Κωσύ και τον Σιμόν Αντουάν Ζαν Λ' Ουιγιέ (Simon Antoine Jean L'Huillier) και είναι αρχή της τοπολογίας (L'Huiller,1861).



Εικόνα 2 : Πρόβλημα της γέφυρας της Καίνιξμπεργκ

Το αντικείμενο της θεωρίας των γράφων είναι η μελέτη των γράφων, δηλαδή μαθηματικών δομών, που χρησιμοποιούνται για τη μοντελοποίηση και την αναπαράσταση των σχέσεων που υφίστανται μεταξύ ζευγών αντικειμένων. Σύμφωνα με τον ορισμό που προέρχεται από τα μαθηματικά, ένας γράφος $G = (V, E)$ συνιστά μια αναπαράσταση ενός συνόλου αντικειμένων που συνδέονται μεταξύ τους. Τα αντικείμενα καλούνται κόμβοι ή κορυφές (Vertices) του γράφου και συμβολίζονται με V και οι συνδέσεις που υφίστανται ανάμεσα σε ένα ζεύγος κόμβων ονομάζονται ακμές (Edges) του γράφου και συμβολίζονται με E. Οι Aldous and Wilson ορίζουν τον γράφο ως ένα διάγραμμα αποτελούμενο από σημεία που ονομάζονται κόμβοι, τα οποία συνδέονται μεταξύ τους με γραμμές που ονομάζονται ακμές, ενώ κάθε ακμή συνδέει ακριβώς δύο κόμβους (Aldus M. Joan & Wilson J Robin, 2000).

Κάθε ακμή του γράφου ορίζεται από ένα ζεύγος κόμβων στα άκρα της, οι οποίοι καλούνται άκρα της ακμής. Κάθε κόμβος του γράφου χαρακτηρίζεται από τον βαθμό του, ο οποίος ορίζεται ίσος με τον αριθμό των ακμών που συνδέονται με το συγκεκριμένο κόμβο. Δύο ακμές που συνδέονται στον ίδιο κόμβο καλούνται γειτονικές ακμές, ενώ δύο κόμβοι που συνδέονται μεταξύ τους μέσω μιας κοινής ακμής καλούνται γειτονικοί κόμβοι αντίστοιχα. Το σύνολο των ακμών ενός γράφου δύναται να είναι κενό ενώ το σύνολο των κόμβων είναι απαραιτήτως μη- κενό. Οι ακμές ενός γράφου μπορεί να είναι κυκλικές (loops), δηλαδή να καταλήγουν στον ίδιο κόμβο από τον οποίο ξεκινούν. Στην περίπτωση που μια ακμή συνδέεται με έναν κόμβο, ο οποίος με τη σειρά του δε συνδέεται μέσω ακμής με κάποιο άλλο κόμβο του γράφου, καλείται τυφλή ακμή, ενώ μια ακμή, η οποία δε συνδέεται με άλλη ακμή και τα άκρα της είναι βαθμού ένα, καλείται αιωρούμενη ακμή. Ακμές που συνδέουν ακριβώς το ίδιο ζεύγος κόμβων καλούνται πολλαπλές ακμές. Τέλος, θα πρέπει να αναφερθεί ότι ένας κόμβος δύναται να ανήκει σε ένα γράφο, χωρίς απαραίτητα να ανήκει ταυτόχρονα σε κάποια από τις ακμές του γράφου (Στεφανάκης Εμμανουήλ, 2003).

2.1.1. Τύποι Γράφων

Οι γράφοι εμφανίζονται διαφοροποιημένοι μεταξύ τους, ανάλογα με την ιδιαίτερη δομή και τα χαρακτηριστικά τους. Ως εκ τούτου, ταξινομούνται σε επιμέρους κατηγορίες στη βάση ενός αριθμού κριτηρίων που τίθενται κατά περίπτωση, όπως για παράδειγμα η ανάθεση ή μη φοράς στις ακμές του γράφου, η συνδεσιμότητα του γράφου, η ανάθεση βαρών στις ακμές του γράφου κ.λπ.

Ανάλογα με τη δομή και τα χαρακτηριστικά τους, οι γράφοι υιοθετούνται για τη μοντελοποίηση διαφορετικής φύσεως προβλημάτων. Ένας κατευθυνόμενος γράφος για παράδειγμα, συνιστά την καταλληλότερη δομή για τη μοντελοποίηση δικτύων ενώ ένας απλός μη κατευθυνόμενος γράφος, αποτελεί κατάλληλη δομή για την αναπαράσταση των χημικών δεσμών που υφίστανται μεταξύ των ατόμων ενός μορίου. Κάθε κατηγορία γράφων χαρακτηρίζεται από την ιδιαίτερη διαγραμματική αναπαράσταση των γράφων που ανήκουν σ' αυτή, τα ιδιαίτερα χαρακτηριστικά και τις ιδιότητες των δομικών στοιχείων των γράφων και τους κανόνες που τίθενται από τα μαθηματικά και χαρακτηρίζουν τους γράφους κάθε κατηγορίας.

Στη συνέχεια, παρουσιάζονται οι βασικότερες κατηγορίες στις οποίες ταξινομούνται οι γράφοι ανάλογα με τη δομή και τα χαρακτηριστικά τους.

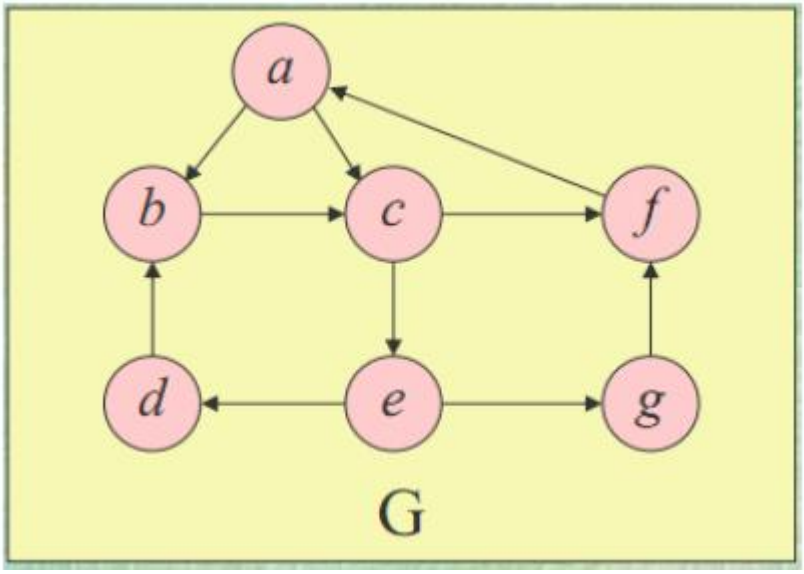
- **Απλοί Γράφοι (Simple Graphs)**: Ένας απλός γράφος *G = (V, E)* ορίζεται ως ένα διάγραμμα αποτελούμενο από κόμβους και ακμές, ενώ μεταξύ δύο κόμβων του γράφου υφίσταται μία και μόνο μία ακμή. Ένας απλός γράφος δεν περιλαμβάνει κυκλικές ακμές (Aldus M. Joan & Wilson J Robin, 2000).

- **Κατευθυνόμενοι Γράφοι (Directed Graphs)**: Ένας γράφος ορίζεται ως κατευθυνόμενος γράφος G = (V, A) όταν οι ακμές που συνδέουν τους κόμβους του είναι προσανατολισμένες προς μια κατεύθυνση (φορά), οπότε και αυτές με τη σειρά τους χαρακτηρίζονται ως κατευθυνόμενες ακμές (directed edges) ή τόξα (arcs). Ένα τόξο a = (x, y) που ανήκει σε έναν κατευθυνόμενο γράφο, έχει κατεύθυνση από τον κόμβο x προς τον κόμβο y. Ο κόμβος y καλείται άμεσος διάδοχος (direct successor) του x και ο κόμβος x άμεσος προκάτοχος (direct predecessor) του κόμβου y (Στεφανάκης, 2003).

- **Μεικτοί Γράφοι (Mixed Graphs)**: Μεικτός καλείται ένας γράφος G = (V, E, A) ο οποίος είναι δυνατό να περιλαμβάνει ταυτόχρονα κατευθυνόμενες και μη κατευθυνόμενες ακμές. Οι γράφοι αυτού του είδους συνιστούν ειδική περίπτωση γράφου (http://en.wikipedia.org/wiki/Graph).

- **Συνδεδεμένοι Γράφοι (Connected Graphs) και μη Συνδεδεμένοι Γράφοι**: Ως συνδεδεμένος χαρακτηρίζεται ένας γράφος, στον οποίο υπάρχουν ένα ή περισσότερα

μονοπάτια μέσω των οποίων συνδέονται δύο οποιοιδήποτε κόμβοι του γράφου. Ως μη συνδεδεμένος χαρακτηρίζεται ένας γράφος, στον οποίο δεν υφίσταται απαραίτητα σύνδεση μεταξύ του συνόλου των κόμβων που περιλαμβάνονται στο γράφο. Σε ένα μη συνδεδεμένο γράφο, δύο κόμβοι u και v ονομάζονται συνδεδεμένοι εάν υφίσταται στο γράφο μονοπάτι μέσω του οποίου συνδέονται οι δύο κόμβοι. Η αφαίρεση κόμβων ή ακμών δύναται να καταστήσει ένα συνδεδεμένο γράφο, μη συνδεδεμένο. Ένας γράφος είναι ισχυρά συνδεδεμένος (strongly connected graph) όταν, δύο οποιοιδήποτε κόμβοι του συνδέονται μέσω ενός μονοπατιού είτε αυτό κατευθύνεται από τον κόμβο u στον κόμβο v είτε έχει αντίστροφη κατεύθυνση από τον v στον u. Στην περίπτωση που οι κατευθυνόμενες ακμές ενός μη συνδεδεμένου γράφου αντικατασταθούν με μη κατευθυνόμενες ακμές, ένας γράφος που πριν την αντικατάσταση ήταν μη συνδεδεμένος δύναται να καταστεί συνδεδεμένος αλλά όχι ισχυρά συνδεδεμένος γράφος (Aldus M. Joan & Wilson J Robin, 2000).



Εικόνα 3 : Κατευθυνόμενος γράφος

- **Κανονικοί Γράφοι (Regular Graphs)**: Ένας γράφος ορίζεται ως κανονικός όταν όλοι οι κόμβοι του έχουν ίσο αριθμό γειτονικών κόμβων. Στην περίπτωση αυτή, όλοι οι κόμβοι του γράφου έχουν ακριβώς τον ίδιο βαθμό (Aldus M. Joan & Wilson J Robin, 2000).

- **Πλήρεις Γράφοι (Complete Graphs)**: Καλούνται οι γράφοι όπου μεταξύ κάθε ζεύγους κόμβων υφίσταται οπωσδήποτε μια σύνδεση. Οι γράφοι αυτής της κατηγορίας, περιλαμβάνουν το μέγιστο δυνατό αριθμό ακμών (http://mathworld.wolfram.com/ConnectedGraph.html)

- **Σταθμισμένοι Γράφοι (Weighted Graphs)**: Ένας σταθμισμένος γράφος, είναι ένας γράφος στις ακμές του οποίου ανατίθενται βάρη. Τα βάρη αυτά μπορεί να αναπαριστούν το κόστος μιας διαδρομής, το μήκος της, το χρόνο που απαιτείται για τη συνολική διάσχιση της συγκεκριμένης διαδρομής κ.λπ. (Aldus M. Joan & Wilson J Robin, 2000), (Στεφανάκης Εμμανουήλ 2003).

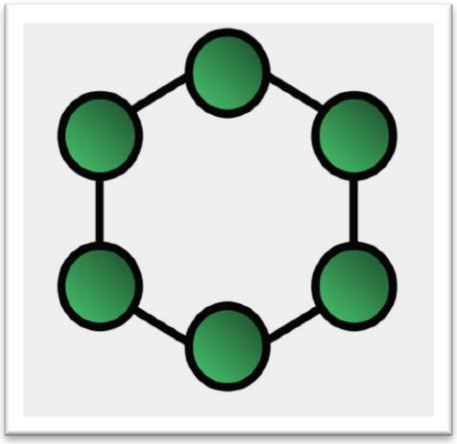
- **Επίπεδοι (Planar Graphs) και μη Επίπεδοι Γράφοι**. Ένας γράφος καλείται επίπεδος όταν μπορεί να σχεδιαστεί στο επίπεδο χωρίς να διασταυρώνονται οι πλευρές του. Δύο οποιεσδήποτε ακμές ενός επίπεδου γράφου συναντώνται μόνο σε προσκείμενους ή τερματικούς κόμβους. Στην περίπτωση που οι ακμές και οι κόμβοι του γράφου κείνται στο χώρο, ο γράφος καλείται μη επίπεδος. Στις τομές των ακμών ενός μη επίπεδου γράφου δεν παρεμβάλλεται κόμβος (Aldus M. Joan & Wilson J Robin, 2000).

- **Διμερείς Γράφοι (Bipartite Graphs)**: Ένας γράφος καλείται διμερής όταν οι κορυφές του μπορούν να διαιρεθούν σε δύο σύνολα έτσι ώστε κάθε στοιχείο του ενός να συνδέεται με κάποιο στοιχείο του άλλου. Δύο στοιχεία που ανήκουν στο ίδιο σύνολο δε συνδέονται μεταξύ τους.

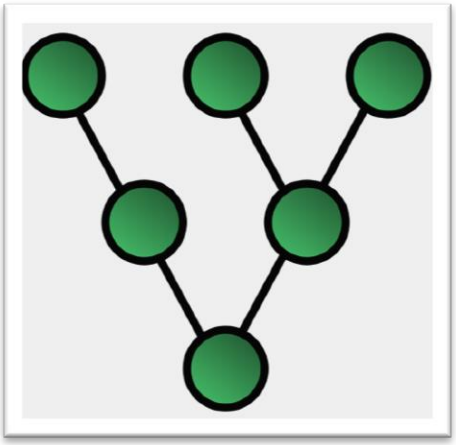
- **Κυκλικοί Γράφοι (Cycle Graphs)**: Ένας γράφος καλείται κυκλικός όταν αποτελείται από έναν κύκλο, δηλαδή έναν κόμβο και μια κυκλική ακμή όπου η αρχή και το πέρας της είναι ο μοναδικός κόμβος του γράφου (Εικόνα 4). Επίσης, ένας κυκλικός κόμβος ορίζεται και ως μια κλειστή «αλυσίδα» ακμών που συνδέονται μεταξύ τους, ενώ ο κόμβος αφετηρίας της πρώτης ακμής είναι ο ίδιος με τον κόμβο πέρατος της τελευταίας ακμής.

- **Άκυκλος γράφος** : Ένας γράφος καλείται άκυκλος όταν δεν περιέχει κύκλους (Εικόνα 5)

- **Δέντρο** : Καλούνται οι μη κατευθυνόμενοι Γράφοι, στους οποίους οποιεσδήποτε δύο κορυφές συνδέονται με ένα και μόνο απλό μονοπάτι. Με άλλα λόγια κάθε συνεκτικός γράφος χωρίς κύκλους είναι ένα δέντρο.



Εικόνα 4: Κυκλικός γράφος 6 κόμβων



Εικόνα 5: Άκυκλος γράφος 6 κόμβων

2.1.2. Ιδιότητες γράφων μετάβασης

Στο εδάφιο αυτό παρουσιάζονται οι βασικές ιδιότητες των γράφων οι οποίοι ενεργούν με προσπέλαση κόμβων ή ακμών. Πρόκειται για κοινούς γράφους με ιδιότητες διαφορετικές των υπολοίπων, αφού ανήκουν στην κατηγορία των ισχυρά συνεκτικών. Αυτό σημαίνει ότι υπάρχει δρόμος μεταξύ οποιωνδήποτε δύο κορυφών του ακολουθώντας τις κατευθύνσεις των ακμών.

Οι βασικές ιδιότητες που χαρακτηρίζουν κάθε γράφο είναι αυτές που σχετίζονται με το βαθμό του γράφου και των δομικών του στοιχείων, το μέγεθος του γράφου και τις σχέσεις γειτνίασης που αναπτύσσονται μεταξύ των κόμβων και των ακμών του.

Ο βαθμός (degree, valence) ενός κόμβου (μίας κορυφής) ορίζεται ως ο αριθμός των ακμών που προσπίπτουν στο κόμβο(v), και συμβολίζεται με deg(v) (Κολιός Ν., 2009). Το μέγεθος ενός γράφου ορίζεται ίσο με τον αριθμό των ακμών του και συμβολίζεται με $|E|$. βαθμός ενός κόμβου που είναι το πέρας μιας τυφλής ακμής είναι πάντα ίσος με ένα. Ίσος με ένα είναι επίσης και ο βαθμός των δύο κόμβων που ορίζουν μια αιωρούμενη ακμή. Σε κάθε γράφο, το άθροισμα των βαθμών των κόμβων του ισούται με το διπλάσιο του αριθμού των ακμών του (Aldus M. Joan & Wilson J Robin, 2000).

Οι ιδιότητες που αφορούν σχέσεις γειτνίασης σε γράφους, σχετίζονται με τις σχέσεις γειτνίασης που αναπτύσσονται μεταξύ των δομικών στοιχείων των γράφων. Έτσι, δύο ακμές που συνδέονται με τον ίδιο κόμβο καλούνται γειτονικές ακμές (adjacent edges), ενώ δύο κόμβοι που συνδέονται με μια κοινή ακμή καλούνται γειτονικοί κόμβοι (adjacent nodes). Μία ακολουθία γειτονικών ακμών σε ένα γράφο, κάθε μια από τις οποίες έχει έναν

κοινό κόμβο με την ακμή που προηγείται, ορίζει ένα μονοπάτι (path) που συνδέει δύο οποιουσδήποτε κόμβους του γράφου.

Κάθε ακμή που ανήκει σε ένα σταθμισμένο γράφο χαρακτηρίζεται από το βάρος που της αντιστοιχεί και το οποίο ανάλογα με την περίπτωση μπορεί να συμβολίζει το κόστος διάσχισής της, το μήκος της ή οποιοδήποτε άλλο μέγεθος απαιτεί η εκάστοτε εφαρμογή. Το συνολικό βάρος ενός σταθμισμένου γράφου ισούται με το άθροισμα των βαρών του συνόλου των ακμών του γράφου (Aldus M. Joan & Wilson J Robin, 2000).

Επιπρόσθετα, από έναν ή περισσότερους υπάρχοντες γράφους δύναται να προκύψει ένας νέος γράφος μέσα από την πρόσθεση ή την αφαίρεση κόμβων ή ακμών, τη συγχώνευση κόμβων ή τη σύνδεση κόμβων που ανήκουν σε διαφορετικούς γράφους. Οι διαδικασίες δημιουργίας ενός νέου γράφου από έναν ή περισσότερους υφιστάμενους γράφους συνιστούν στοιχειώδεις λειτουργίες οι βασικότερες των οποίων κατηγοριοποιούνται ως ακολούθως:

• Στοιχειώδεις (elementary) λειτουργίες: Αφορούν τη δημιουργία ενός νέου γράφου από έναν ήδη υπάρχοντα γράφο μέσω μιας απλής τοπικής αλλαγής όπως η πρόσθεση ή διαγραφή κόμβου ή ακμής, η συγχώνευση κόμβων κ.λπ.

• «Μοναδιαίες» (unary) λειτουργίες: Αφορούν τη δημιουργία ενός νέου γράφου, γραμμικού, συμπληρωματικού κ.α. από ήδη υπάρχοντα γράφο.

• Δυαδικές (binary) λειτουργίες: Αφορούν τη δημιουργία ενός νέου γράφου από δύο προϋπάρχοντες αρχικούς γράφους.

Τέλος, έχει αποδειχθεί ότι για έναν επίπεδο γράφο, το σύνολο των πιθανών αναπαραστάσεών του απαρτίζεται από το ίδιο πλήθος πολυγώνων. Συνεπώς, εάν ένας επίπεδος γράφος αποτελείται από κ κόμβους, α ακμές και π πολύγωνα ισχύει η σχέση:

$$\pi - \alpha + \kappa = 2$$

Η παραπάνω σχέση είναι γνωστή ως κριτήριο του Euler και αποτελεί έναν απλό έλεγχο για τη μη-επιπεδότητα ενός γράφου. Συνιστά ικανή αλλά όχι αναγκαία συνθήκη για την επιπεδότητα, δηλαδή κάποιος γράφος που δεν ικανοποιεί το κριτήριο είναι γράφος μη-επίπεδος. Ωστόσο, ένας γράφος που ικανοποιεί το κριτήριο του Euler δεν είναι απαραίτητα επίπεδος. Η γενικευμένη μορφή του κριτηρίου του Euler, προκειμένου να ισχύει και για μη-συνδεδεμένους γράφους είναι η ακόλουθη:

$$\pi - \alpha + \kappa - \mu = 1$$

όπου μ, ο αριθμός των συστατικών μερών που απαρτίζουν ένα μη-συνδεμένο γράφο.

2.1.3. Αναπαράσταση γράφων στην Μνήμη

Υπάρχουν δύο ενδεδειγμένοι τρόποι αναπαράστασης γράφων στη μνήμη του υπολογιστή. Ένας γράφος μπορεί να αναπαρασταθεί είτε με Πίνακες Γειτνίασης (Adjacency Matrix) είτε με λίστες γειτνίασης (Adjacency Lists) (Cormen H. Tomas et al., 2001) .

Σύμφωνα με τον πρώτο τρόπο που αναφέρθηκε ένας γράφος G=(V,E) με n κορυφές μπορεί να αναπαρασταθεί ως ένας n×n πίνακας που περιέχει τις τιμές 0 και 1, και όπου αν αν η (i,j) είναι ακμή τότε A[(i,j)]=1, διαφορετικά A[(i,j)]=0 (Πίνακας 1).

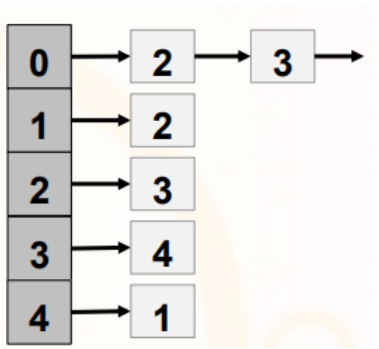
Αν ο γράφος είναι γράφος με βάρη, και το βάρος κάθε ακμής είναι τύπου t, τότε για την αναπαράσταση του γράφου μπορεί να χρησιμοποιηθεί πίνακας τύπου t με A[(i,j)] = βάρος(i,j), αν υπάρχει ακμή (i,j) και A[(i,j)] = ∞, αν δεν υπάρχει ακμή (i,j). Αυτή η αναπαράσταση απαιτεί χώρο $\Theta(n^2)$, όπου n=|V|. Αυτό οδηγεί σε σπατάλη χώρου και επιβράδυνση των υπολογισμών αν ο γράφος είναι αραιός.

Κατά τη μέθοδο αναπαράστασης με λίστες γειτνίασης, ένας γράφος G=(V,E) αναπαρίσταται ως ένας μονοδιάστατος πίνακας A. Για κάθε κορυφή v, A[v] είναι ένας δείκτης σε μια συνδεδεμένη λίστα στην οποία αποθηκεύονται οι κορυφές που γειτνιάζουν με την v (Πίνακας 2). Η μέθοδος αυτή απαιτεί σαφώς λιγότερη μνήμη αφού ο αριθμός των θέσεων που καταλαμβάνει είναι Θ(|V|+ |E|). Στην περίπτωση γράφων με βάρη στη λίστα γειτνίασης αποθηκεύεται επίσης το βάρος κάθε ακμής.

Πίνακας 1: Πίνακας Γειτνίασης

	0	1	2	3	4
0			1	1	
1			1		
2				1	
3					1
4		1			

Πίνακας 2: Λίστα Γειτνίασης



2.1.4. Αλγόριθμοι διάσχισης γράφων

Αν πρέπει να επισκεφθούν όλοι οι κόμβοι ενός γράφου μπορούν να χρησιμοποιηθούν ποικίλοι τρόποι, οι οποίοι διαφέρουν στη σειρά με την οποία εξετάζουν τους κόμβους. Διαδικασίες διάσχισης χρησιμοποιούνται και για τη διακρίβωση ύπαρξης μονοπατιού μεταξύ δύο κόμβων κ.α. Στη συνέχει παρουσιάζονται οι δύο κυριότεροι αλγόριθμοι

διάσχισης γράφων τα βασικά χαρακτηριστικά τους, η δομή και η λειτουργία τους. Ανάλογα με τις ιδιαιτερότητες και τις απαιτήσεις του εκάστοτε προβλήματος, εφαρμόζεται κατά περίπτωση ο αλγόριθμος που εξασφαλίζει την αποδοτικότερη διαχείρισή του. Αυτός είναι άλλωστε και ο λόγος που οι αλγόριθμοι διάσχισης γράφων εμφανίζονται διαφοροποιημένοι ως προς τα δεδομένα που κάθε φορά επεξεργάζονται και ως προς τις διαδικασίες επεξεργασίας των δεδομένων αυτών.

Οι αλγόριθμοι συνιστούν τη βάση για το σχεδιασμό αλγόριθμων που εφαρμόζονται για την επίλυση προβλημάτων, τα οποία με τη σειρά τους αφορούν ειδικές περιπτώσεις διάσχισης γράφου. Οι αλγόριθμοι αυτοί αναφέρονται ως «Κατά Πλάτος Αναζήτηση» γράφου (Breadth- First Search) και «Κατά Βάθος Αναζήτηση» γράφου (Depth- First Search).

Πρόκειται για δύο αλγόριθμους οι οποίοι δομήθηκαν υπό διαφορετική λογική, διαφοροποιώντας ουσιαστικά τη διαδικασία αναζήτησης σε γράφους. Κάθε ένας από αυτούς ενσωματώνει τα ιδιαίτερα πλεονεκτήματα και χαρακτηριστικά του, τα οποία καθορίζουν και την επιλογή εφαρμογής του αντίστοιχου αλγόριθμου ανάλογα με τις απαιτήσεις της εκάστοτε εφαρμογής.

Κατά Πλάτος Διάσχιση Γράφου (BFS)

Η κατά πλάτος αναζήτηση γράφου συνιστά έναν από τους απλούστερους αλγόριθμους που εφαρμόζονται κατά τη διαδικασία διάσχισης ενός γράφου, ενώ παράλληλα αποτελεί τη βάση για το σχεδιασμό εξειδικευμένων αλγόριθμων διάσχισης γράφων όπως για παράδειγμα ο αλγόριθμος Dijkstra, ο σχεδιασμός του οποίου στηρίζεται στην ιδέα της κατά πλάτος αναζήτησης γράφου.

Η ονομασία του αλγόριθμου της κατά πλάτος αναζήτησης (Breadth- First Search, BFS) γράφου, υποδηλώνει και τον τρόπο με τον οποίο εφαρμόζεται ο συγκεκριμένος αλγόριθμος αναζήτησης στους γράφους. Ο συγκεκριμένος αλγόριθμος «επεκτείνει» τα νοητά όρια μεταξύ των κόμβων που έχουν ανακαλυφθεί κατά τη διαδικασία της αναζήτησης και εκείνων που δεν έχουν ανακαλυφθεί ακόμη. Ο αλγόριθμος αναζητά τους κόμβους που πρόκειται να επισκεφθεί, εφαρμόζοντας μια κατά πλάτος τεχνική αναζήτησης καθώς δίνει προτεραιότητα στους κόμβους που απέχουν απόσταση k από τον αρχικό κόμβο έναντι εκείνων που απέχουν απόσταση $k + 1$. Με άλλα λόγια, ο αλγόριθμος επισκέπτεται πρώτα τους άμεσους κόμβους- γείτονες του αρχικού και στη συνέχεια με την ίδια λογική συνεχίζει την αναζήτηση στους υπόλοιπους κόμβους του γράφου (Cormen H. Tomas et al., 2001).

Η κατά πλάτος διάσχιση γράφου έχει ως αποτέλεσμα τη δημιουργία ενός κατά πλάτος δένδρου (breadth- first tree), το οποίο αρχικά αποτελείται μόνο από τη ρίζα του που είναι ο αρχικός κόμβος s και επεκτείνεται κάθε φορά που ανακαλύπτεται ένας νέος κόμβος, ο οποίος προστίθεται στο δένδρο μαζί με την ακμή που τον συνδέει με τον πρόγονο κόμβο

του (Cormen H. Tomas et al., 2001). Στη συνέχεια, παρουσιάζεται ο ψευδοκώδικας του αλγόριθμου BFS.

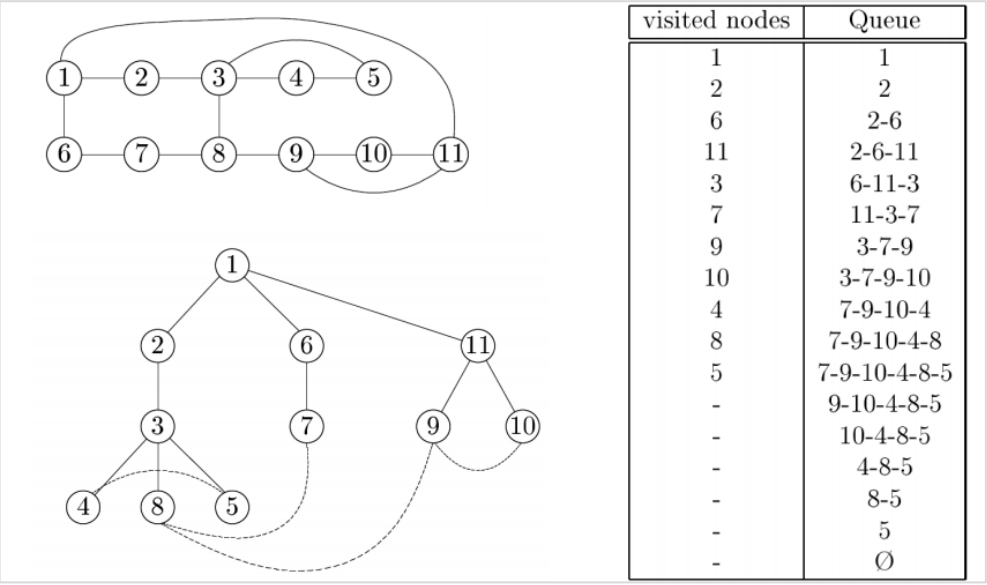
```
BFS (G , s)
1   for each vertex u ∈ V [G] - {s}
2        do color[u] ← WHITE
3           d[u] ← ∞
4           π[u] ← NIL
5   color[s] ← GRAY
6   d[s] ← 0
7   π[s] ← NIL
8   Q ← Ø
9   ENQUEUE(Q, s)
10  while Q ≠ Ø
11      do u ← DEQUEUE(Q)
12         for each v ∈ Adj[u]
13             do if color[v] = WHITE
14                   then color[v] ← GRAY
15                        d[v] ← d[u] + 1
16                        π[v] ← u
17                        ENQUEUE(Q, v)
18         color[u] ← BLACK
```

Σχήμα 1 : Ο ψευδοκώδικας του BFS Αλγορίθμου

Στις τέσσερις πρώτες γραμμές του ψευδοκώδικα, γίνεται η αρχικοποίηση των μεταβλητών και όλοι οι κόμβοι λαμβάνουν χρώμα λευκό το οποίο δηλώνεται μέσω της μεταβλητής *"color"* και σημαίνει ότι κανένας από τους κόμβους δεν έχει ακόμη προσπελαθεί από τον αλγόριθμο. Στη μεταβλητή *d[u]* που αντιστοιχεί στην απόσταση που υφίσταται μεταξύ του αρχικού κόμβου και των υπόλοιπων κόμβων του γράφου, δίνεται αρχικά τιμή ίση με το άπειρο για το σύνολο των κόμβων *u* του γράφου, ενώ στη μεταβλητή *π[u]* που αναπαριστά τον προκάτοχο ενός ανακαλυφθέντος κόμβου δίνεται αρχική τιμή ίση με το μηδέν. Στην πέμπτη γραμμή, ορίζεται ο αρχικός κόμβος αναφοράς ο οποίος λαμβάνει χρώμα γκρι, η απόσταση λαμβάνει την τιμή μηδέν καθώς δεν έχει ανακαλυφθεί κανένας άλλος κόμβος εκτός του αρχικού, ενώ η τιμή που αφορά στον προκάτοχο του ανακαλυφθέντος κόμβου είναι ίση με *NIL* διότι ο αρχικός κόμβος δεν έχει προκάτοχο κόμβο. Στις γραμμές 8 και 9 αρχικοποιείται η λίστα στην οποία καταχωρούνται οι κόμβοι που έχουν ανακαλυφθεί. Τέλος οι σειρές 10-18 περιλαμβάνουν έναν επαναληπτικό βρόχο (loop) ο οποίος ξεκινά με την εντολή "while". Οι εντολές που περιλαμβάνονται σε αυτό το τμήμα του κώδικα επαναλαμβάνονται όσο υπάρχουν γκρι κόμβοι στο γράφο, κόμβοι δηλαδή των οποίων οι λίστες γειτνίασης δεν έχουν ακόμη εξετασθεί πλήρως. Οι λίστες γειτνίασης «κρατούν» τους κόμβους οι οποίοι ανακαλύπτονται από τον αλγόριθμο σε σειρά προτεραιότητας. Ο πρώτος κόμβος που εισέρχεται στη λίστα είναι ο αρχικός κόμβος (source node), από τον οποίο ξεκινά η αναζήτηση στο γράφο. Κάθε κόμβος που ανακαλύπτεται από τον αλγόριθμο

βγαίνει από τη λίστα όταν λάβει μαύρο χρώμα, γεγονός που σημαίνει ότι το σύνολο των γειτονικών του κόμβων έχει ανακαλυφθεί από τον αλγόριθμο αναζήτησης. Παράλληλα με τη διαδικασία ανεύρεσης κόμβων, ενημερώνονται οι μεταβλητές d και π που αφορούν στην απόσταση ενός ανακαλυφθέντος κόμβου από τον αρχικό και τους προκατόχους των ανακαλυφθέντων κόμβων αντίστοιχα. Η πολυπλοκότητα του συγκεκριμένου αλγόριθμου είναι ίση με $O (V + E)$, γεγονός που σημαίνει ότι ο χρόνος που απαιτείται για την ολοκλήρωση του αλγόριθμου μεταβάλλεται γραμμικά ως προς το μέγεθος της λίστας γειτνίασης που χρησιμοποιείται για την αναπαράσταση του γράφου. Από τον ψευδοκώδικα, καθίσταται πλήρως κατανοητό ότι η κατά πλάτος αναζήτηση γράφου αναζητά συντομότερες διαδρομές (shortest paths) μετάβασης από τον αρχικό κόμβο του γράφου προς το σύνολο των κόμβων οι οποίοι είναι προσβάσιμοι από τον αρχικό.

Στο Σχήμα 2παρουσιάζεται ένα παράδειγμα διάσχισης γράφου με τη μέθοδο BFS.



visited nodes	Queue
1	1
2	2
6	2-6
11	2-6-11
3	6-11-3
7	11-3-7
9	3-7-9
10	3-7-9-10
4	7-9-10-4
8	7-9-10-4-8
5	7-9-10-4-8-5
-	9-10-4-8-5
-	10-4-8-5
-	4-8-5
-	8-5
-	5
-	Ø

Σχήμα 2 : Αναπαράσταση του BFS αλγορίθμου σε γράφο

Κατά Βάθος Αναζήτηση γράφου (DFS)

Σε αντίθεση με τον αλγόριθμο που αφορά την κατά πλάτος διάσχιση γράφου, η κατά βάθος διάσχιση γράφου εστιάζει στη διαδοχική επέκταση του γράφου αναζητώντας κόμβους προσβάσιμους από τον αρχικό, σε διάφορα «βάθη». Έτσι, ο αλγόριθμος επισκέπτεται έναν άμεσο γείτονα του αρχικού κόμβου και στη συνέχεια προχωρά με τη διαδοχική εξερεύνηση του άμεσου γείτονα του κόμβου που μόλις ανακαλύφθηκε, ανεξάρτητα από το εάν έχει ανακαλυφθεί το σύνολο των γειτονικών κόμβων του αρχικού κόμβου, μέχρις ότου εξαντληθεί ένα ορισμένο βάθος. Στη συνέχεια, ο αλγόριθμος επιστρέφει στον κόμβο από τον οποίο ξεκίνησε την κατά βάθος αναζήτηση προκειμένου να εξερευνήσει άλλα βάθη,

ξεκινώντας την αναζήτηση από έναν άλλο άμεσο γείτονα κόμβο του αρχικού. Η διαδικασία ολοκληρώνεται όταν ανακαλυφθούν όλοι οι κόμβοι που είναι προσβάσιμοι από τον αρχικό. Στην περίπτωση που μετά το πέρας της διαδικασίας αναζήτησης έχουν απομείνει κόμβοι που δεν έχουν ανακαλυφθεί, ορίζεται εκ νέου ένας δεύτερος αρχικός κόμβος και η διαδικασία αναζήτησης επαναλαμβάνεται μέχρις ότου ανακαλυφθούν όλοι οι κόμβοι του γράφου (Shekhar Shashi & Chawla Sanjay, 2003).

Όμοια με τον αλγόριθμο BFS, στην περίπτωση εφαρμογής του αλγορίθμου DFS, όταν ένας κόμβος ανακαλύπτεται, ο κόμβος που προηγείται του ανακαλυφθέντος και συνδέεται με αυτόν μέσω κοινής ακμής, ορίζεται ως ο προκάτοχος του ανακαλυφθέντος κόμβου και καταχωρείται στη μεταβλητή *π[u]*. Αρχικά, όλοι οι κόμβοι του γράφου λαμβάνουν χρώμα λευκό, όποιος κόμβος ανακαλύπτεται λαμβάνει χρώμα γκρι, ενώ όταν έχει ανακαλυφθεί το σύνολο των κόμβων που ανήκουν στη λίστα γειτνίασης του αρχικού κόμβου, τότε ο κόμβος αυτός λαμβάνει μαύρο χρώμα. Η κατά βάθος διάσχιση γράφου έχει ως αποτέλεσμα τη δημιουργία πολλαπλών κατά βάθος δένδρων (depth- first trees), καθώς η αναζήτηση δύναται να ξεκινά από περισσότερους του ενός αρχικών κόμβων. Το σύνολο των δένδρων του γράφου που προκύπτουν μετά την ολοκλήρωση του αλγόριθμου, συνιστούν ένα κατά βάθος δάσος (depth- first forest). Επιπλέον ο αλγόριθμος που εφαρμόζεται στην κατά βάθος αναζήτηση σημαίνει χρονικά κάθε κόμβο του γράφου. Κάθε κόμβος έχει δύο χρονικές σημάνσεις. Η πρώτη από αυτές αφορά την πρώτη φορά όπου ανακαλύφθηκε ο κόμβος και έλαβε γκρι χρώμα, ενώ η δεύτερη αφορά τη χρονική στιγμή όπου ο αλγόριθμος έχει ολοκληρώσει την αναζήτηση στο σύνολο των κόμβων που περιλαμβάνονται στη λίστα γειτνίασης του αρχικού, ο οποίος τότε λαμβάνει μαύρο χρώμα (Shekhar Shashi & Chawla Sanjay, 2003). Στο Σχήμα 3 παρουσιάζεται ο ψευδοκώδικας της κατά βάθος αναζήτησης γράφου.

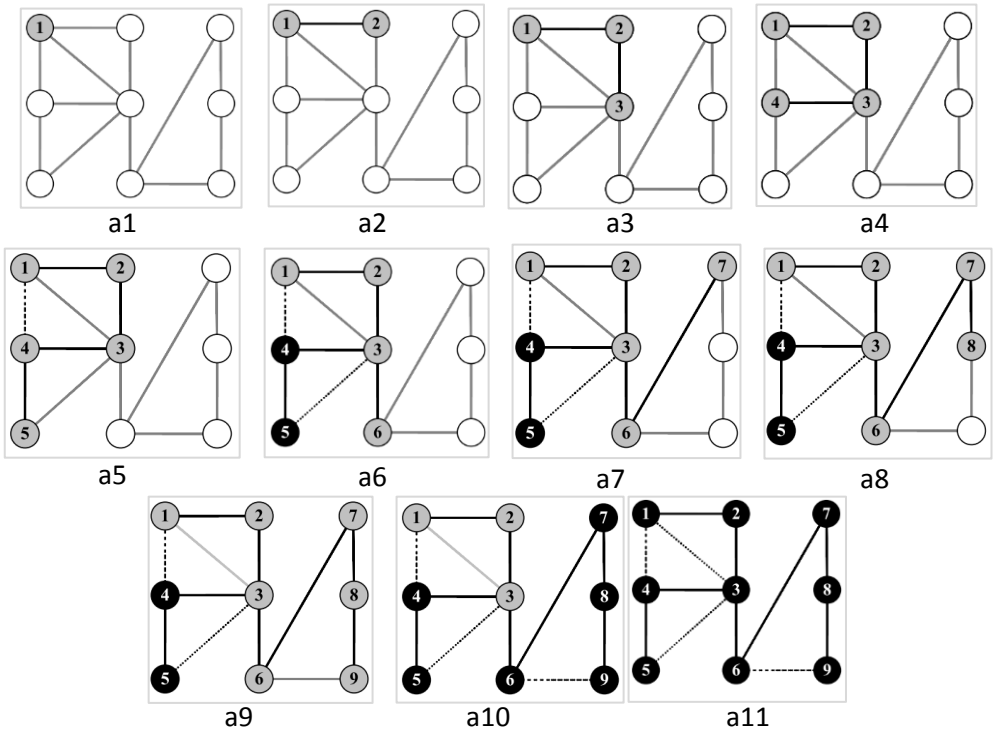
```
DFS (G)
1   for each vertex u ∈ V [G]
2        do color[u] ← WHITE
3           π[u] ← NIL
4   time ← 0
5   for each vertex u ∈ V [G]
6         do if color[u] = WHITE
7               then DFS-VISIT (u)
DFS-VISIT (u)
1  color[u] ← GRAY  ▶ White vertex u has just
been   discovered
2  time ← time + 1
3  d[u]  time
4  for each v ∈ Adj[u]  ▶ Explore edge (u,v)
5       do if color[v] = WHITE
6             then π[v] ← u
7                        DFS-VISIT(v)
8  color[u] ← BLACK    ▶ Blacken u; it is
finished
9  f[u] ▶ time ← time + 1
```

Σχήμα 3: Ο ψευδοκώδικας του DFS Αλγορίθμου

Οι τέσσερις πρώτες γραμμές του ψευδοκώδικα, περιλαμβάνουν την αρχικοποίηση των μεταβλητών. Όλοι οι κόμβοι του γράφου λαμβάνουν χρώμα λευκό και η μεταβλητή στην οποία καταχωρούνται οι προκάτοχοι κόμβοι *π[u]* των ανακαλυφθέντων κόμβων λαμβάνει αρχική τιμή *NIL*, δηλαδή κενή. Στην τέταρτη γραμμή, καταχωρείται η τιμή μηδέν στη μεταβλητή του χρόνου, ενώ στις γραμμές 5 έως 7 ελέγχεται το χρώμα του κόμβου και όταν αυτό είναι λευκό τότε ξεκινά μια διαδικασία κατά βάθος αναζήτησης από έναν αρχικό κόμβο. Κάθε φορά που καλείται αυτό το τμήμα του κώδικα, ο εκάστοτε κόμβος *u* του γράφου γίνεται η ρίζα ενός νέου δένδρου του κατά βάθος δάσους που δημιουργείται.

Όταν ο αλγόριθμος επιστρέφει στο σημείο αυτό κάθε κόμβος *u* έχει λάβει δύο χρονικές σημάνσεις, το χρόνο ανακάλυψης (discovery time) *d[u]* και το χρόνο περάτωσης της συγκεκριμένης αναζήτησης (finishing time) *f[u]*. Σε κάθε νέα κλήση του κώδικα, ο κόμβος *u* είναι αρχικά λευκός, τη στιγμή που ανακαλύπτεται λαμβάνει γκρι χρώμα, αυξάνεται η τιμή της μεταβλητής time και στη μεταβλητή *d[u]* καταχωρείται μια νέα χρονική στιγμή της ανακάλυψης του κόμβου. Ο κώδικας που περιλαμβάνεται στις γραμμές 4-7 του δεύτερου τμήματος του ψευδοκώδικα, ελέγχει κάθε γειτονικό κόμβο *v* του αρχικού κόμβου *u* και τον επισκέπτεται εάν αυτός είναι λευκός και έτσι ανακαλύπτεται και μια νέα ακμή του γράφου. Όταν ανακαλύπτεται το σύνολο των ακμών που συνδέονται με τον κόμβο *u*, τότε ο κόμβος *u* λαμβάνει χρώμα μαύρο και καταχωρείται η τιμή του χρόνου περάτωσης της συγκεκριμένης αναζήτησης στη μεταβλητή *f[u]* (γραμμές 8-9 του ψευδικώδικα).

Τα αποτελέσματα μιας κατά βάθος αναζήτησης, εξαρτώνται από τη σειρά που ελέγχονται οι κόμβοι στη γραμμή 5 του DFS και από τη σειρά επισκεψιμότητας των γειτονικών κόμβων του εκάστοτε αρχικού κόμβου- σειρά 4 του DFS-VISIT. Τέλος, η πολυπλοκότητα του αλγόριθμου και ο χρόνος τρεξίματός του προκύπτει ως εξής: Η ολοκλήρωση των επαναλήψεων στις γραμμές 1-3 και 5-7 του DFS απαιτούν χρόνο τρεξίματος ίσο με $\Theta(V)$. Η διαδικασία που περιλαμβάνεται στο τμήμα του DFS-VISIT, καλείται ακριβώς μια φορά για κάθε κόμβο *v* ενώ ο βρόχος που περιλαμβάνεται στις γραμμές 4-7 εκτελείται |Adj[v]| φορές. Ως εκ τούτου, το κόστος εκτέλεσης του βρόχου είναι ίσο με $\Theta(E)$. Συνεπώς, το συνολικό κόστος εκτέλεσης μιας κατά βάθος αναζήτησης γράφου είναι ίση με $\Theta(V + E)$.



Σχήμα 4: Αναπαράσταση του BFS αλγορίθμου σε γράφο

2.2. Αλγόριθμοι ελάχιστης διαδρομής

Οι αλγόριθμοι που έχουν αναπτυχθεί για τη δημιουργία καμπυλών ίσου χρόνου είναι βασισμένοι στον αλγόριθμο ελάχιστης διαδρομής του Dijkstra. Σαφώς έχουν αναπτυχθεί και άλλες προσεγγίσεις της εύρεσης της συντομότερης διαδρομής συνεπώς είναι απαραίτητη η μελέτη των βασικότερων από αυτές, ώστε να αποφασιστεί αν είναι χρήσιμες για την παρούσα εργασία.

2.2.1. Dijkstra

Πρόκειται για έναν αλγόριθμο εύρεσης των συντομότερων διαδρομών (single-source shortest path problem), από κοινή αφετηρία σε έναν (κατευθυνόμενο ή μη) γράφο με μη αρνητικά βάρη στις ακμές.

Συνεπώς δεδομένου ενός γράφου $G=(V,E)$, όπου V είναι το σύνολο των κόμβων και E το σύνολο των ακμών, και ενός αρχικού κόμβου s , ο αλγόριθμος Dijkstra υπολογίζει τις συντομότερες διαδρομές από το σημείο αυτό προς όλους τους κόμβους του γράφου (Langeman Matt, 2005).

Ο αλγόριθμος διατηρεί ένα πίνακα D στον οποίο αποθηκεύει την τρέχουσα υπολογισμένη ελάχιστη απόσταση κάθε κόμβου από τον αρχικό s. Κατά την αρχικοποίηση $D[s]=0$ και $D[u]=+\infty$, όπου u είναι κάθε κόμβος διάφορος του s. Επίσης, διατηρεί ένα σύνολο S που αποτελείται από τους κόμβους για τους οποίους έχει ήδη προσδιοριστεί η ελάχιστη διαδρομή και το οποίο είναι αρχικά κενό. Τέλος, ο αλγόριθμος χρησιμοποιεί ένα ακόμη διάνυσμα, το prev[], μεγέθους n, στο οποίο για κάθε κόμβο u αποθηκεύεται ο αμέσως προηγούμενος κόμβος στο ελάχιστο μονοπάτι προς τον u (Παπαθεοδώρου Σ. Θεόδωρος, 2007).

Η επαναληπτική διαδικασία είναι η εξής :

- Σημειώνεται για κάθε κόμβο μια ετικέτα απόστασης ($d[*]$) με τιμή 0 στον αρχικό κόμβο και ∞ σε όλους τους υπόλοιπους. Επίσης σημειώνεται μια ετικέτα ($prev[*]$) και εισάγεται κενή τιμή για όλους τους κόμβους. Η ετικέτα αυτή χρειάζεται για τον υπολογισμό της ζητούμενης διαδρομής στο τέλος.

- Θεωρούνται όλοι οι κόμβοι ως μη επεξεργασμένοι.

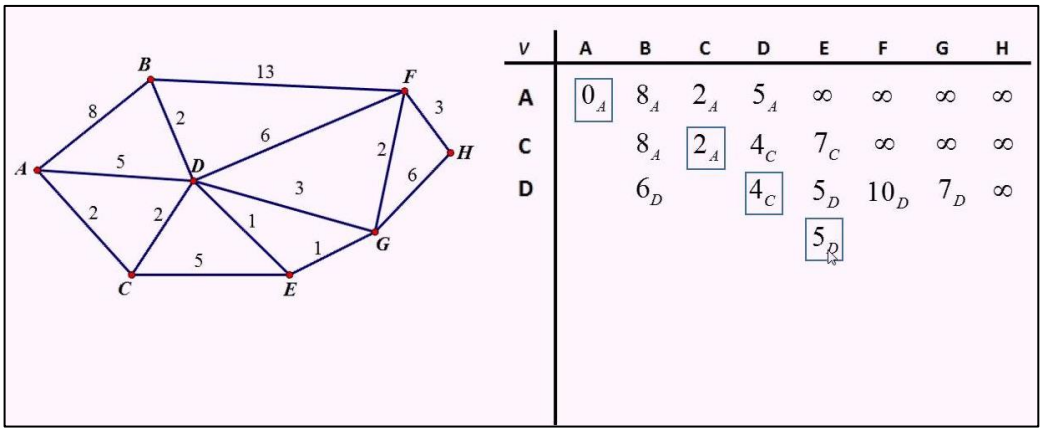
- Για τον τρέχον κόμβο εξετάζονται όλοι οι μη –επεξεργασμένοι γείτονές του και υπολογίζονται το συνολικό άθροισμα απόστασής τους από τον αρχικό κόμβο. Αν αυτή η απόσταση είναι μικρότερη από την ετικέτα απόστασης που είχε σημειωθεί στο πρώτο βήμα τότε αντικαθιστάται με τη νέα υπολογισμένη τιμή και ο κόμβος εισάγεται στην ετικέτα προηγούμενου κόμβου ($prev[*]$).

- Όταν εξετασθούν όλοι οι γειτονικοί κόμβοι του τρέχοντος κόμβου u, ο u θεωρείται επεξεργασμένος και προστίθεται στο σύνολο S. Ένας

επεξεργασμένος κόμβος δεν εξετάζεται ποτέ ξανά από τον αλγόριθμο, καθώς η ετικέτα απόστασής της είναι η ελάχιστη και θα παραμείνει σταθερή. Παράλληλα "χαλαρώνουν" όλες οι ακμές που ξεκινούν από αυτόν. Με τον όρο "χαλάρωμα" μιας ακμής εννοούμε την ανανέωση της εκτιμώμενης απόστασης για τον τελικό κόμβο v της ακμής δηλαδή γίνεται $D[v] = D[u] + w(u,v)$.

- Ο επόμενος τρέχων κόμβος θα είναι ο μη-επεξεργασμένος κόμβος με τη μικρότερη ετικέτα απόστασης.
- Ο αλγόριθμος τερματίζει όταν στο S προστεθεί και ο τελευταίος κόμβος του γράφου.

Στην εικόνα 6 φαίνεται η διαδικασία εκτέλεσης του αλγορίθμου για ένα μικρό γράφο.



V	A	B	C	D	E	F	G	H
A	0_A	8_A	2_A	5_A	∞	∞	∞	∞
C		8_A	2_A	4_C	7_C	∞	∞	∞
D		6_D		4_C	5_D	10_D	7_D	∞
					5_D			

Εικόνα 6 : Παράδειγμα εκτέλεσης του αλγορίθμου Dijkstra σε έναν μικρό γράφο

Για τον υπολογισμό των ισοχρονικών καμπυλών μπορεί και πάλι να χρησιμοποιηθεί ο αλγόριθμος Dijkstra χρησιμοποιώντας ως κόστος το χρόνο διαδρομής αντί για την απόσταση όπως στο παράδειγμα.

2.2.2. Ο αλγόριθμος A*

Ένας ακόμη αλγόριθμος ελάχιστης διαδρομής βασισμένος στον αντίστοιχο του Dijkstra ειναι ο A*. Χρησιμοποιείται σε περιπτώσεις μεγάλων γράφων όπου η πολυπλοκότητα του αλγόριθμου του Dijkstra καθίσταται απαγορευτική. Αξιοποιεί τεχνικές της τεχνητής

νοημοσύνης, προκειμένου να αυξηθεί η ταχύτητα αναζήτησης στους γράφους μέσα από την εφαρμογή ευρετικών κανόνων (heuristics).

Αρχικά, εκτιμάται το κόστος μετάβασης από τον κόμβο αφετηρίας στον κόμβο προορισμού. Πρόκειται για μια ενδεικτική και όχι για μια ακριβή εκτίμηση. Στη συνέχεια, υπολογίζονται αναδρομικά τα κόστη μετάβασης στους γειτονικούς κόμβους του αρχικού κόμβου και αποκλείονται εξ αρχής μονοπάτια που έχουν μεγαλύτερο κόστος σε σχέση με αυτό που είχε αρχικά εκτιμηθεί (Στεφανάκης Εμμανουήλ, 2003).

Αναλυτικότερα, ο αλγόριθμος χρησιμοποιεί μια συνάρτηση *f (u, d)* εκτίμησης του κόστους μετάβασης από τον κόμβο αφετηρίας στον κόμβο προορισμού προκειμένου να εκτιμηθεί ένα αρχικό κόστος, στη βάση του οποίου πρόκειται να ελεγχθούν τα κόστη των μονοπατιών που ανακαλύπτονται στη συνέχεια. Ο ψευδοκώδικας του αλγόριθμου A* που παρουσιάζεται στο σχήμα που ακολουθεί, προκύπτει από τον ψευδοκώδικα του αλγόριθμου του Dijkstra και εμφανίζεται διαφοροποιημένος αφενός ως προς τη συνάρτηση *f (u, d)* που χρησιμοποιείται για την αρχική εκτίμηση του κόστους και αφετέρου ως προς τους ευρετικούς κανόνες που υπεισέρχονται κατά τη διαδικασία εύρεσης του συντομότερου μονοπατιού (Shekhar Shashi & Chawla Sanjay, 2003)

```
1 procedure Dijkstra (G (V, E), v, d, f);
2 {
3   var: integer;
4     foreach u in V do {C (v, u) = inf; C(v, v) = 0; path
(v, u) :=null}
5    frontierSet := [v]; exploredSet :=emptySet;
6    while not_empty(frontierSet) do
7    {    select w from frontierSet with minimum (C (v, w)
+ f(w, d));
8         frontierSet := frontierSet - [w]; exploredSet :=
exploredSet + [w];
9         if (u = d) then terminate
10        else { fetch( w.adjacencyList);
11             foreach <  u, C(w, u) > in w.adjacencyList
12             if C(v, u) > C(v, w) + C(w, u) then
13             {
14                  C(v, u) := C(v, w) + C(w, u);
15                  path (v, u) := path (v, w) + (w, u);
16                  if u ∋ frontierSet U exploredSet then
17                  frontierSet := frontierSet + [u];
18             }
19        }
20   }
21 }
```

Σχήμα 5 : Ψευδοκώδικας του αλγορίθμου A*

Η διαδικασία αναζήτησης του αλγόριθμου A* ολοκληρώνεται μετά την επανάληψη εκείνη του βρόχου *while* όπου προσπελαύνεται ο κόμβος *u* ως ο κόμβος προορισμού της διαδρομής με το μικρότερο κόστος και εισάγεται στο σύνολο των επισκεφθέντων κόμβων *frontierSet* με το μικρότερο συσσωρευμένο κόστος. Όσο μικρότερο είναι το πλήθος των ακμών που συνθέτουν το συντομότερο μονοπάτι από τον κόμβο εκκίνησης στον κόμβο προορισμού, τόσο μικρότερος είναι ο χρόνος που απαιτείται για το τρέξιμο του αλγόριθμου. Ο αλγόριθμος δεν είναι απαραίτητο να επισκεφθεί το σύνολο των κόμβων του γράφου προκειμένου να ανακαλύψει τη συντομότερη διαδρομή μεταξύ των κόμβων αφετηρίας και προορισμού, καθώς πολλοί από αυτούς ανήκουν σε μονοπάτια τα οποία έχουν ήδη αποκλειστεί, πριν αυτοί οι κόμβοι προσπελαθούν λόγω υψηλού κόστους (www.w3.org/html/).

Η πολυπλοκότητα του αλγόριθμου A* εξαρτάται από τους ευρετικούς κανόνες που χρησιμοποιούνται κατά περίπτωση. Έτσι, εάν η συνάρτηση εκτίμησης που χρησιμοποιεί ο αλγόριθμος είναι η *f(x)* τότε η πολυπλοκότητα του αλγόριθμου ορίζεται ως εξής:

$$\left| f(x) - f^*(x) \right| = O(\log f^*(x))$$

Όπου f^* είναι ο βέλτιστος ευρετικός κανόνας που χρησιμοποιεί ο αλγόριθμος για την εύρεση της συντομότερης διαδρομής σε ένα γράφο. Ο αλγόριθμος A* είναι κατά κανόνα ταχύτερος από τον αλγόριθμο του Dijkstra, αφού όμως για τον υπολογισμό των ισοχρονικών καμπυλών δεν υπάρχει καθορισμένος προορισμός, δεν μπορεί να χρησιμοποιηθεί η συγκεκριμένη προσέγγιση.

2.2.3. Ο αλγόριθμος Bellman –Ford

Ο αλγόριθμος Bellman-Ford ο οποίος εφαρμόζεται σε ένα γράφο με συσχετισμένους συντελεστές βάρους. Αρχικά εντοπίζει το ελάχιστο μονοπάτι από το κόμβο εκκίνησης s αποτελούμενο από μία ζεύξη, στη συνέχεια το ελάχιστο μονοπάτι από τον s αποτελούμενο από δύο ζεύξεις και συνεχίζει κατ' αυτόν τον τρόπο για $n-1$ φορές όπου n το πλήθος των κόμβων στο γράφο. Αρχικά όλοι κόμβοι θεωρούνται ότι έχουν άπειρη απόσταση από τον s. Τελικά υπολογίζει τη συντομότερη διαδρομή από ένα σημείο εκκίνησης προς όλους τους υπόλοιπους κόμβους του γράφου, είναι όμως πιο αργός από τον Dijkstra στην επίλυση του ίδιου προβλήματος. Από την άλλη αποτελεί πιο ευέλικτη λύση αφού παρέχεται η δυνατότητα να χειρίζεται γραφήματα στα οποία υπάρχουν ακμές αρνητικού κόστους (https://en.wikipedia.org/wiki/bellman). Ο ψευδοκώδικας του αλγόριθμου είναι ο εξής :

```
BELLMAN-FORD (G, W, S)
1   INITIALIZE-SINGLE-SOURCE(G,S)
2   for i ← 1 to |V[G]| - 1
3       do for each edge (u,v) ∈ E[G]
4             do RELAX(u, v, w)
5   for each edge (u,v) ∈ E[G]
6         do if d[v] > d[u] + w(u,v)
7               then return FALSE
8   return TRUE
```

Σχήμα 6 : Ψευδοκώδικας του αλγορίθμου Bellman - Ford

Στην πρώτη γραμμή του ψευδοκώδικα, γίνεται η αρχικοποίηση των μεταβλητών *d* και *π* στις οποίες καταχωρούνται η απόσταση που διανύεται κατά τη διάρκεια τρεξίματος του αλγόριθμου και οι προκάτοχοι κόμβοι αντίστοιχα. Ο αλγόριθμος, υλοποιεί $|V|-1$ περάσματα από τις ακμές του γράφου, κάθε ένα από τα οποία αντιστοιχεί σε μια επανάληψη του βρόχου που περιλαμβάνεται στην εντολή *for* στις γραμμές 2-4. Αφού ολοκληρωθούν τα $|V|-1$ περάσματα από τις ακμές του γράφου, ο αλγόριθμος ελέγχει την ύπαρξη κύκλων με αρνητικά βάρη και επιστρέφει την αντίστοιχη boolean τιμή. Εάν υπάρχει κύκλος με αρνητικό βάρος το τρέξιμο του αλγόριθμου σταματά, σε διαφορετική περίπτωση ο αλγόριθμος δίνει ως αποτέλεσμα τα συντομότερα μονοπάτια και το κόστος τους.

2.3. Αλγόριθμοι για τον υπολογισμό ισοχρονικών καμπυλών

Τα τελευταία χρόνια έχουν αναπτυχθεί κάποιοι αλγόριθμοι για τον υπολογισμό ισοχρονικών καμπυλών. Σε αυτό το κεφάλαιο θα παρουσιαστούν κάποια γενικά στοιχεία γι' αυτούς.

2.3.1. MINE

Ο συγκεκριμένος αλγόριθμος παρουσιάζεται στην εργασία των Johann Gamper et al., το 2011. Αναπτύχθηκε για να υπολογίζει ισοχρονικές καμπύλες σε χωρικά δίκτυα πολλαπλών μέσων, όπου κάποιες γραμμές είναι συνεχόμενες και κάποιες διακριτές. Βασίζεται στον αλγόριθμο του Dijkstra και υπολογίζει τις συντομότερες διαδρομές από μια συγκεκριμένη αφετηρία (source) προς όλους τους υπόλοιπους κόμβους του δικτύου. Λειτουργεί

αυξάνοντας διαδοχικά την προκύπτουσα ισόχρονη καμπύλη από τις ακμές εντός του δικτύου. Όταν ένας κόμβος προσπελαστεί, αν το χρονικό κόστος είναι υψηλότερο από το κατώφλι που έχει τεθεί, η επέκταση της καμπύλης τερματίζεται εκεί.

Ένας παρόμοιος αλγόριθμος περιγράφεται στην εργασία των Veronica Bawer et al του 2008, ο οποίος μάλιστα εστιάζει στις δημόσιες μεταφορές με λεωφορείο. Τα δεδομένα έχουν δομηθεί έτσι ώστε οι στάσεις των λεωφορείων (nodes) να «πέφτουν» πάνω στις γραμμές του οδικού δικτύου (edges). Έτσι δηλαδή κάθε γραμμή συνδέει δύο σημεία, που αναπαριστούν δύο διαδοχικές στάσεις. Επιπλέον έχουν συμπεριληφθεί στην ανάλυση τα χρονοδιαγράμματα των λεωφορείων και οι μετεπιβιβάσεις που είναι δυνατόν να πραγματοποιηθούν. Οι παράμετροι αρχικοποίησης του αλγορίθμου είναι : το δίκτυο των γραμμών (*N*), ο κόμβος τερματισμού (*POI*), οι υπόλοιποι κόμβοι του δικτύου (n_d), η διάρκεια διαδρομής (*dur*), η επιθυμητή ώρα άφιξης (t_d) και η ταχύτητα βαδίσματος σε m/s (s_w). Ο αλγόριθμος παράγει ένα υποσύνολο του δικτύου των γραμμών (*Nstreets*) που σχηματίζουν τη ζητούμενη ισοχρονική καμπύλη. Παρακάτω παρουσιάζεται ο ψευδοκώδικας του τροποποιημένου αλγορίθμου MINE.

Algorithm: ISOCHRONE(N, n_d, t_d, dur, s_w)
Add $(n_d, t_d, N_{streets})$ to empty queue Q;
while $Q \neq \emptyset$ **do**
  $(n, t, N) \leftarrow$ next node from Q;
  foreach *(incoming) link* $(n', n) \in N$ **do**
    Compute latest starting time, t', for n';
    if $t' > t_d - dur$ **then**
      Add (n', t', N) to Q;
      if $N = N_{streets}$ **then**
        Add (n', n) to result set;
        Check for bus stops on (n', n) and add them to Q with appropriate starting time;
      else if N *is a bus network* **then**
        Check for stops of other buses at n' and add them to Q with appropriate starting time;

Σχήμα 7 : Τροποποιημένος αλγόριθμος MINE

2.3.2. MINEX

Ο αλγόριθμος αυτός είναι βασισμένος στον προαναφερθέντα αλγόριθμο MINE. Η βελτίωση έγγειται στο γεγονός ότι μετά την πρώτη επανάληψη απορρίπτονται τα στοιχεία που δεν χρειάζονται πλέον για την περαιτέρω διερεύνηση της ισόχρονης. Αυτό οδηγεί στη μείωση των απαιτήσεων μνήμης του συστήματος και άρα επιτάχυνση της διαδικασίας.

Ο αλγόριθμος MINE, ακριβώς όπως διευρυμένος Dijkstra, χρησιμοποιεί ένα σύνολο κλειστών κόμβων, το οποίο χρησιμοποιείται για να αποφευχθεί η κυκλική προσπέλαση

κόμβων που έχουν ήδη προσπελαστεί. Έτσι όσο προχωράει η δημιουργία της ισοχρονικής καμπύλης οι περισσότεροι κόμβοι του συνόλου δεν χρειάζονται πλέον.

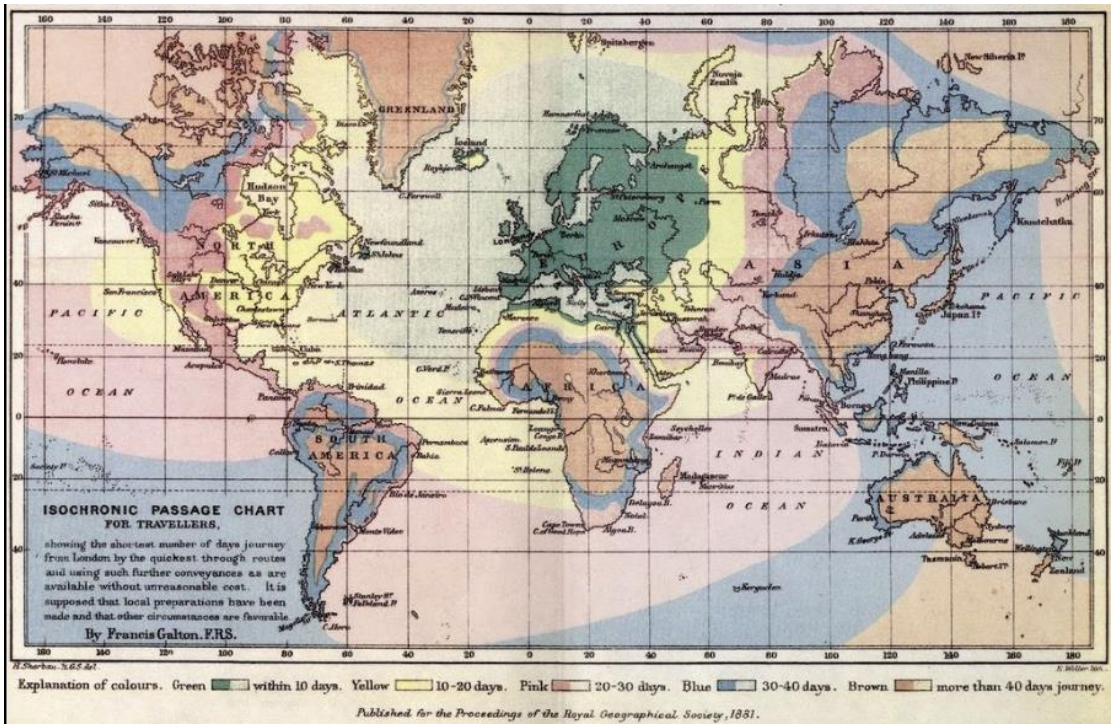
Για να περιοριστεί το μέγεθος του κλειστού συνόλου, ο αλγόριθμος MINEX εξάγει απ' αυτό τους κόμβους που έχουν «λήξει». Θεωρείτε ότι ένα κόμβος πρέπει να αφαιρεθεί από το κλειστό σύνολο όταν όλοι οι γειτονικοί του κόμβοι είτε έχουν ήδη αφαιρεθεί είτε είναι εντός. Ο αλγόριθμος χρησιμοποιεί μετρητή για τον κάθε κόμβο, ώστε να ορίζεται εγκαίρως πότε ένας κόμβος πρέπει να θεωρηθεί ότι έχει λήξει. Επίσης έχει αποδειχθεί ότι οι κόμβοι που έχουν αφαιρεθεί δεν χρειάζονται πλέον στο κλειστό σύνολο για να είναι σωστός ο αλγόριθμος.

2.4. Θεωρητικές εργασίες υπολογισμού ισοχρονικών καμπυλών

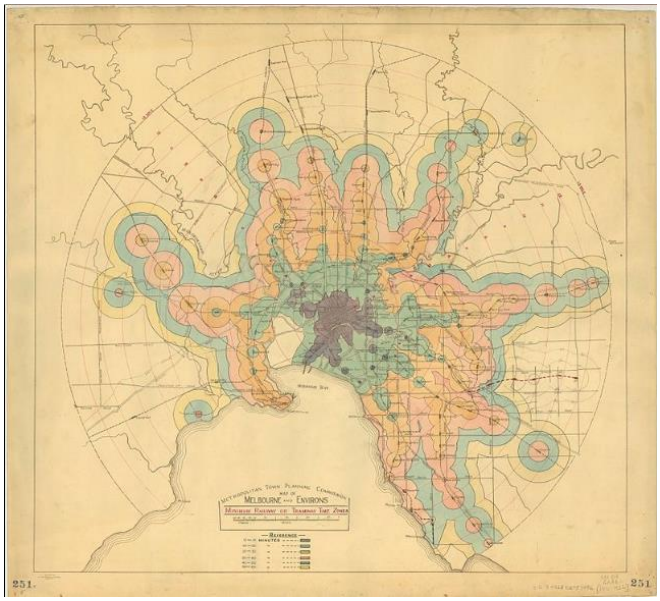
Ο πρώτος γνωστός ισοχρονικός χάρτης ανήκει στον Francis Galton και δημοσιεύθηκε για τα Πρακτικά της Βασιλικής Γεωγραφικής Εταιρείας, το 1881. Εμφανίζει τους χρόνους ταξιδιού από το Λονδίνο του Ηνωμένου Βασιλείου προς διάφορα μέρη του κόσμου σε ημέρες. Προϋποθέτεται ότι υπάρχουν προτιμώμενες συνθήκες ταξιδίου και ότι έχουν πραγματοποιηθεί εκ των πρότερων ρυθμίσεις για τις επίγειες μετακινήσεις. Το κόστος μετακινήσεων στη διάρκεια μίας μέρας κρατείται σε λογικά πλαίσια (Εικόνα 7).

Ένας πρώιμος ισοχρονικός χάρτης είχε δημιουργηθεί για να απεικονίσει το χρόνο διαδρομής με τρένο στην πόλη της Μελβούρνης. Χρονολογείται από το 1910 ως το 1922 (Εικόνα 8).

Η πρώτη θεωρητική εργασία στην οποία εισάγεται ο όρος isochrones είναι των V. Bauer et al. τον Νοέμβρη του 2008. Σύμφωνα με αυτό, isochrones είναι τα σύνολα των σημείων από τα οποία ένα συγκεκριμένο σημείο ενδιαφέροντος είναι προσβάσιμο μέσα σε ένα δεδομένο χρονικό διάστημα. Ασχολείται με ισοχρονικές καμπύλες σε οδικά δίκτυα και υπολογίζει το σύνολο των ακμών που είναι προσβάσιμες από ένα σημείο ενδιαφέροντος σε ένα συγκεκριμένο χρονικό διάστημα χρησιμοποιώντας δημόσια μέσα μεταφοράς και το περπάτημα δεδομένων των στάσεων των λεωφορείων και των δρομολογίων τους. Είναι δηλαδή αρκετά συναφής με την παρούσα εργασία με τη διαφορά ότι εδώ θα υπολογιστούν οι κόμβοι που είναι προσβάσιμοι από ένα σημείο ενδιαφέροντος σε συγκεκριμένο χρόνο.



Εικόνα 7 : Ο πρώτος ισοχρονικός χάρτης που απεικονίζει το χρόνο διαδρομής από το Λονδίνο προς άλλα μέρη του κόσμου



Εικόνα 8 : Ισοχρονικός χάρτης του χρόνου μετακίνησης στη Μελβούρνη μέσω τρένου

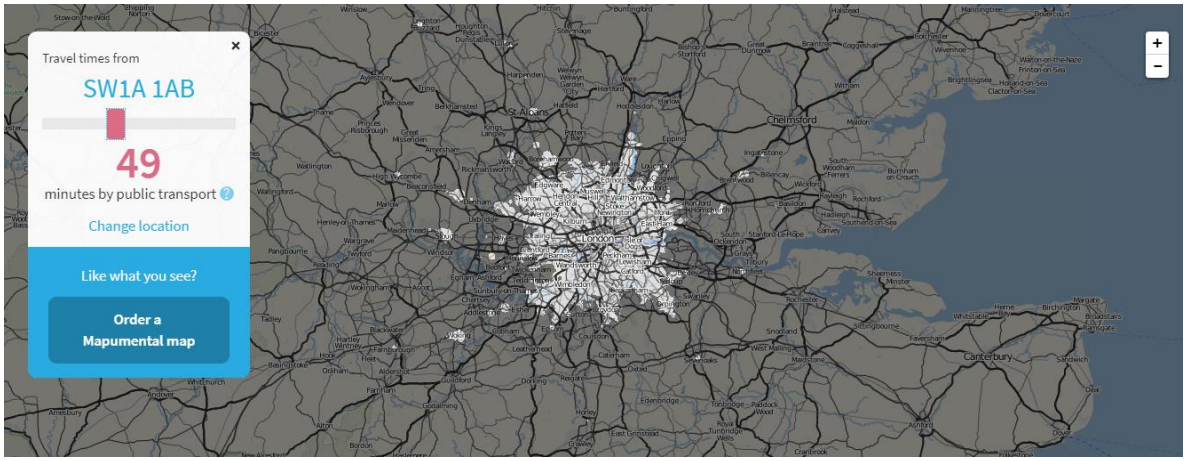
Μία ακόμα εργασία στην οποία πραγματοποιείται υπολογισμός ισοχρονικών καμπυλών δημιουργήθηκε από τους Moritz Baum et al. Στην εργασία αυτή αφενός προτείνεται ένας

πιο σύντομος ορισμός του όρου «ισοχρονικές καμπύλες», αφετέρου προτείνονται κάποιες εύχρηστες κλιμακούμενες αλγοριθμικές προσεγγίσεις για ταχύτερους υπολογισμούς. Παρ' όλο που παρατίθενται πολλαπλές μεθοδολογίες επίλυσης του συγκεκριμένου προβλήματος εστιάζει κυρίως στην ταχύτητα και την αποδοτικότητα της εφαρμογής, κάτι ιδιαίτερα σημαντικό για δυναμικές εφαρμογές και όχι τόσο για στατικές όπως είναι η παρούσα. Επιπλέον η ανάλυση περιορίζεται σε αυτοκινούμενη μετακίνηση, που διαφέρει αρκετά από τη μετακίνηση με μέσα πολλαπλής τροχιάς.

Ο υπολογισμός των ισοχρονικών καμπύλων σήμερα υλοποιείται με τα εργαλεία που διαθέτουν τα Συστήματα Γεωγραφικών Πληροφοριών (ΣΓΠ) π.χ. ArcGIS Routing Services. Στην παρούσα εργασία χρησιμοποιείται Ελεύθερο Λογισμικό / Λογισμικό Ανοιχτού Κώδικα (ΕΛ/ΛΑΚ) και συγκεκριμένα το ΣΓΠ QGIS και τη βάση χωρικών δεδομένων Postgres και το εργαλείο PgRouting (.https://anitagraser.com).

2.5. Διαδικτυακές εφαρμογές

Σήμερα έχουν αναπτυχθεί διαδικτυακές εφαρμογές υπολογισμού ισοχρονικών καμπυλών. Η πιο σχετική με την παρούσα εργασία είναι η εφαρμογή Mapumental στην οποία παρουσιάζονται οι χρόνοι ταξιδιού με χρήση των δημόσιων μέσων μαζικής μεταφοράς στο Λονδίνο. Συγκεκριμένα συνδυάζονται ταχυδρομικά στοιχεία από την Εθνική Υπηρεσία Χαρτογράφησης της Μεγάλης Βρετανίας (Ordance Survey), με τα χρονοδιαγράμματα των δημόσιων μεταφορών και παρουσιάζονται τα αποτελέσματα με χρήση της βιβλιοθήκης Leaflet στο υπόβαθρο του OpenStreetMap (Εικόνα 9).



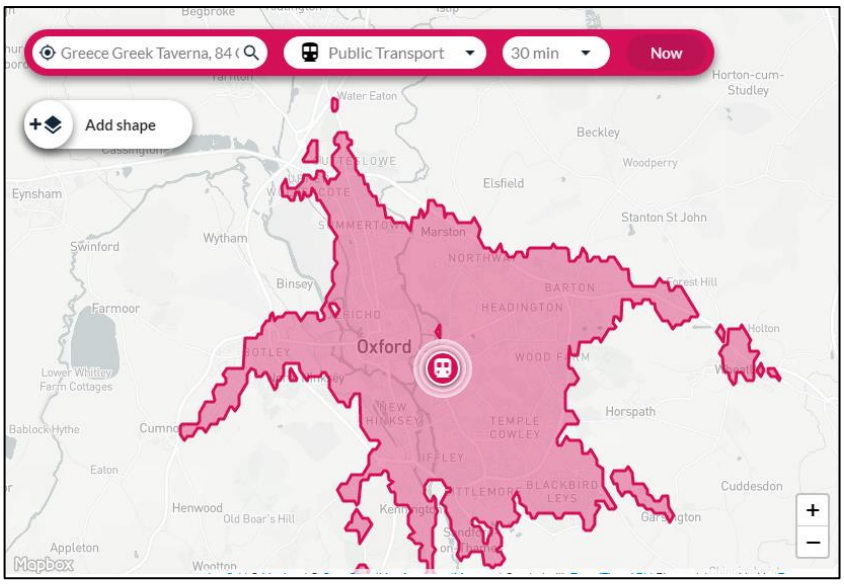
Εικόνα 9 : Παράδειγμα υπολογισμού ισοχρονικών καμπυλών από την εφαρμογή Mapumental

Στη συγκεκριμένη εφαρμογή θεωρείται ότι η ταχύτητα βαδίσματος είναι περίπου 4,8 χλμ/ώρα. Ο συνολικός επιτρεπόμενος χρόνος βαδίσματος του χρήστη είναι τα τριάντα λεπτά ανά διαδρομή, ενώ ο αντίστοιχος χρόνος μεταξύ δυο στάσεων ή από την αρχή και το τέλος τις διαδρομής είναι τα δέκα λεπτά. Επίσης υποθέτεται ότι η μετεπιβίβαση κοστίζει χρονικά ένα λεπτό αν πρόκειται για λεωφορεία και πέντε λεπτά για τρένο, μετρό και ferryboat αν πραγματοποιείται στην ίδια στάση. Σημειώνεται ότι εδώ, όπως και στη συγκεκριμένη εργασία, η αποστάσεις που διανύονται με βάδισμα υπολογίζονται με ευκλείδεια απόσταση και όχι πραγματική.

Στη συνέχεια παρατίθενται και άλλες εφαρμογές που υπολογίζουν καμπύλες χρονοαπόστασης.

Η εφαρμογή Pedcatch, η οποία προσφέρεται μόνο για πεζή και αυτοκινούμενη μετακίνηση και στην οποία ο χρήστης ορίζει το σημείο εκκίνησης και τον επιθυμητό χρόνο μεταφοράς και κατόπιν υπολογίζονται οι ισοχρονικές καμπύλες. Υπάρχει η δυνατότητα να εισαχθούν περισσότερα από ένα σημεία εκκίνησης και μέχρι τρία αλλά είναι ακόμα σε πειραματικό στάδιο. Επίσης ο χρήστης ορίζει την ταχύτητα των πεζών και παρέχεται ένα εύρος από 0,5m/s έως 10m/s. Τέλος η παρουσίαση γίνεται είτε με τον αλγόριθμο Convex Hull είτε με δημιουργία buffer, κάτι που και πάλι επιλέγει ο χρήστης.

Στον ιστότοπο TravelTime Maps μπορούν να υπολογιστούν περιοχές ίσου χρόνου με χρήση ενός από τα εξής μέσα : ποδήλατο, αυτοκίνητο, δημόσιες συγκοινωνίες και βάδισμα. Ο χρήστης ορίζει και εδώ το χρόνο μετακίνησης που επιθυμεί, ενώ παρέχεται και η δυνατότητα να εισάγει περισσότερα από ένα σημεία ενδιαφέροντος. Όσον αφορά στις δημόσιες συγκοινωνίες επιτρέπεται συνολικός χρόνος πεζής μετακίνησης 40 λεπτών, 20 λεπτά στην αρχή της διαδρομής και 20 στο τέλος της. Στην ανάλυση για μετακίνηση με αυτοκίνητο το σύστημα χρησιμοποιεί διαφορετικούς αλγορίθμους για τις ώρες αραιής κυκλοφορίας και τις ώρες αιχμής. Τέλος είναι η μόνη από τις εφαρμογές που αναφέρονται εδώ που για την αμιγώς πεζή μετακίνηση υπολογίζει τις πραγματικά περπατήσιμες διαδρομές και όχι την ευκλείδεια απόσταση. Η υπηρεσία παρέχεται για αρκετές χώρες της Αυστραλίας και της Ευρώπης, αλλά όχι για την Ελλάδα.



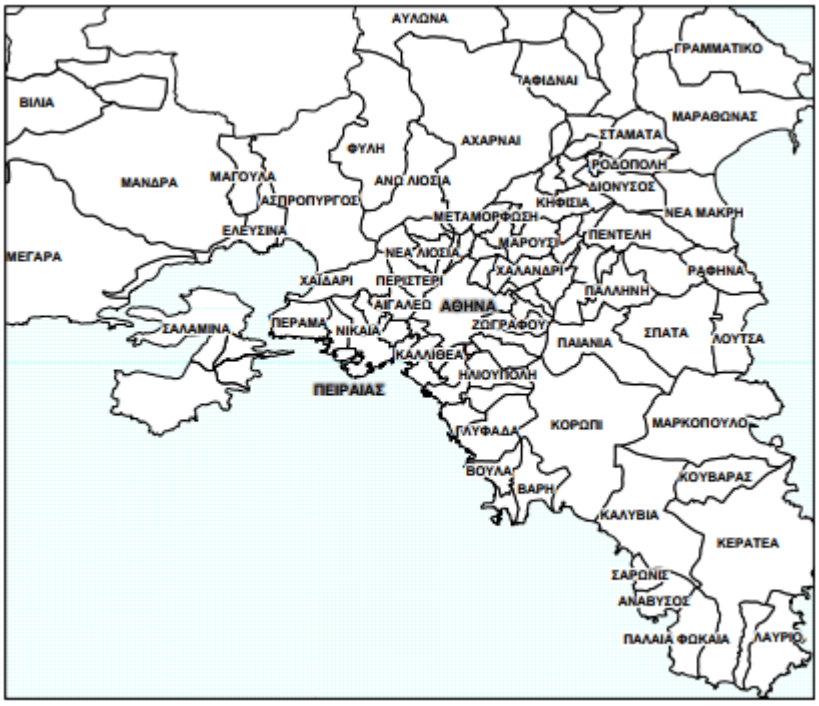
Εικόνα 10 : Παράδειγμα υπολογισμού ισοχρονικών καμπυλών από την εφαρμογή TravelTime Maps

2.6. Περιοχή μελέτης

Περιοχή μελέτης της εργασίας αποτελεί ο βασικός αστικός ιστός της χώρας γύρω από την πρωτεύουσα. Η περιοχή εξυπηρέτησης των Δημόσιων Μέσων Μεταφοράς εκτείνεται δυτικά από την περιοχή της Ελευσίνας και το Θριάσιο Πεδίο μέχρι τους πρόποδες της Πάρνηθας, βόρεια μέχρι το Κρυονέρι και το φράγμα της Λίμνης του Μαραθώνα, περιλαμβάνοντας τις περιοχές Δραπετώσας και Διονύσου, ενώ συμπεριλαμβάνει τη Νέα και την Παλαιά Πεντέλη.

Προς τα νότιο-ανατολικά, καλύπτει την περιοχή μέχρι και τη Σαρωνίδα. Στα ανατολικά (περιοχή των Μεσογείων) καλύπτει την περιοχή μέχρι τα Σπάτα και την Αρτέμιδα, νοτιοδυτικά την Κάντζα, τα Γλυκά Νερά, την Παιανία, το Κορωπί και την Παλλήνη.

Η έκταση της περιοχής αυτής υπερβαίνει τα 1.450 τετρ. χλμ. και αποτελεί περίπου το 35% της έκτασης του νομού Αττικής. Στο κέντρο της περιοχής αυτής βρίσκεται το λεκανοπέδιο της Αθήνας, που αποτελεί και το κύριο τμήμα της περιοχής ευθύνης, με το συντριπτικά μεγαλύτερο μέρος του πληθυσμού της. Η κεντρική περιοχή της Αθήνας, έχει έκταση 11,76 τετρ. χλμ. Ένα πιο περιορισμένο σε έκταση και συμπαγές εμπορικό κέντρο (CBD) περιλαμβάνει το Εμπορικό Τρίγωνο και τις κεντρικές συνοικίες της Πλάκας, Εξαρχείων-Κολωνακίου και Ψυρρή, με συνολική έκταση 3,86 τετρ. χλμ.



Εικόνα 11 : Περιοχή έκτασης των ΜΜΜ

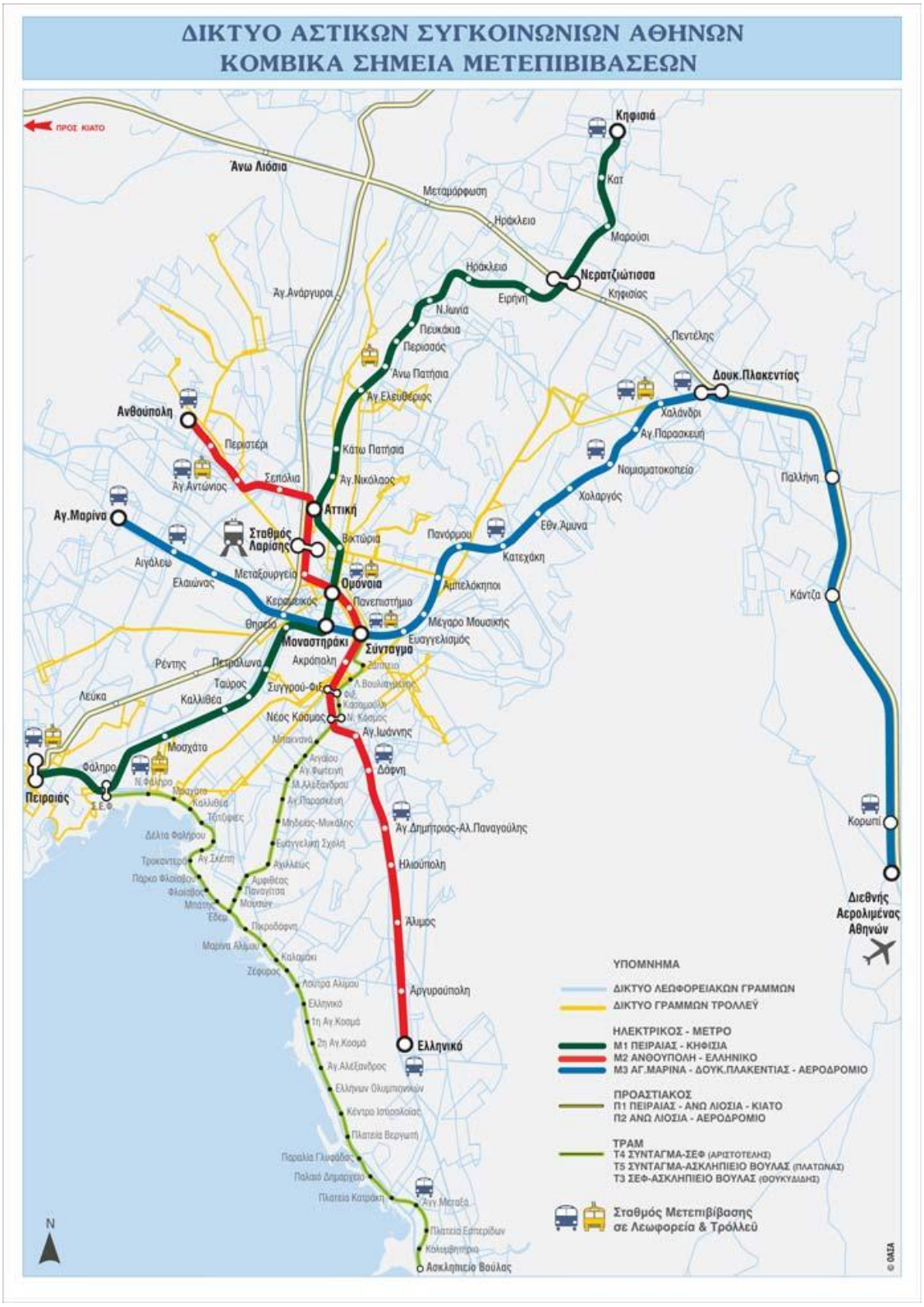
Το δίκτυο των Αστικών Συγκοινωνιών αποτελείται από δύο συνιστώσες : το δίκτυο των λεωφορειακών γραμμών και γραμμών τρόλεϊ και το δίκτυο μέσων σταθερής τροχιάς (ηλεκτρικός/μετρό, τραμ, προαστιακός). Συνολικά το δίκτυο Α.Σ. αποτελείται από 268 γραμμές με ανάπτυγμα διαδρομής σχεδόν 6.281 χλμ. Τα επιφανειακά μέσα χρησιμοποιούν σχεδόν 2.300 χλμ. δρόμων μονής κατεύθυνσης, με μέση πυκνότητα 13 χιλιόμετρα γραμμής ανά τετρ.χλμ στο Λεκανοπέδιο. Το δίκτυο μέσων σταθερής τροχιάς αποτελείται από τρεις γραμμές μετρό, τρεις γραμμές τραμ και μία γραμμή προαστιακού σιδηροδρόμου (www.oasa.gr).

2.6.1. Δίκτυο λεωφορείων και τρόλεϊ

Η πρωτεύουσα εξυπηρετείται από ένα ευρύτατο δίκτυο λεωφορείων και τρόλεϊ, που είναι μορφολογικά διατεταγμένο ιεραρχικά με σύστημα (κύριων) «γραμμών-κορμών», περιλαμβανομένων κάποιων γραμμών express (που ακολουθούν διαδρομές κορμών) και «τοπικών» τροφοδοτικών τους γραμμών.

Οι γραμμές – κορμοί ενώνουν τα κέντρα της Αθήνας (κυρίως) και του Πειραιά, με τοπικά περιφερειακά κέντρα, όπως η Γλυφάδα, η Βούλα, η Αγία Παρασκευή, η Κηφισιά, το Μενίδι, το Χαλάνδρι, η Πετρούπολη, το Περιστέρι, το Αιγάλεω, η Νίκαια κ.λπ. Οι τοπικές γραμμές διακλαδίζονται μέσα στους δήμους και καταλήγουν στο περιφερειακό κέντρο απ’

όπου ξεκινάει η αντίστοιχη γραμμή κορμός, την οποία τροφοδοτούν. Από την επέκταση του μετρό και ύστερα οι τοπικές γραμμές τροφοδοτούν τους αντίστοιχους σταθμούς μετρό, όπου υπάρχει αυτή η επιλογή. Οι σημαντικότεροι κόμβοι τροφοδότησης επιβατικού κοινού στους σταθμούς του μετρό φαίνονται στην Εικόνα 12.



Εικόνα 12 : Κυριότεροι κόμβοι μετεπιβιβάσεων στο Δίκτυο Συγκοινωνιών της Αθήνας

Αυτό το σύστημα συμπληρώνεται από ένα σημαντικό αριθμό «κεντρικών» γραμμών, που συνδέουν περιοχές κοντά στο κέντρο της Αθήνας με αυτό. Τυπικά, οι περιοχές (δήμοι, κοινότητες, συνοικίες), που απέχουν έως 5-6 χλμ. περίπου από το κέντρο της Αθήνας ή του Πειραιά έχουν απευθείας σύνδεση με τα αντίστοιχα κέντρα. Μέρος του κεντρικού υποδικτύου είναι και οι περισσότερες γραμμές τρόλεϊ, που εξυπηρετούν το κέντρο της Αθήνας. Τέλος, υπάρχει και ένας μικρός αριθμός εγκάρσιων περιφερειακών και διαδημοτικών γραμμών, όπως : Πολύγωνο –Γλυφάδα (140), Άγιοι Ανάργυροι -Αγία Παρασκευή (421), Πειραιάς -Στάση Δάφνης (218), Πειραιάς –Στάση Δάφνης (217), Σταθμός Καλλιθέας –Άγιος Δημήτριος (219).

Συμπληρωματικά, ένας μικρός αριθμός ειδικών γραμμών περιλαμβάνει τις γραμμές εξπρές. Από το 2014, οι γραμμές εξπρές μειώθηκαν στις μόλις τρεις, με τις δύο από αυτές να είναι σχολικές, που συνδέουν τον Σταθμό Λαρίσης με το ΤΕΙ Αθηνών και τον Πειραιά με την Πανεπιστημιούπολη. Η τρίτη εκτελεί το δρομολόγιο Σύνταγμα –ΟΑΚΑ – Αεροδρόμιο.

2.6.2. Δίκτυο μέσων σταθερής τροχιάς

Επιπλέον του δικτύου των λεωφορειακών γραμμών και γραμμών τρόλεϊ, οι αστικές συγκοινωνίες της Αθήνας περιλαμβάνουν τρεις γραμμές μετρό, τρεις γραμμές τραμ και μία γραμμή προαστιακού σιδηρόδρομου.

Σήμερα, οι δύο Γραμμές του Μετρό Αθήνας (2 και 3) έχουν συνολικό μήκος 59,7 χλμ. (συμπεριλαμβανομένων των 20,7 χλμ. γραμμής του προαστιακού από τον Σταθμό Δουκίσσης Πλακεντίας προς το Αεροδρόμιο) και διαθέτουν 40 Σταθμούς (περιλαμβάνοντας 4 σταθμούς σε κοινή χρήση με τον Προαστιακό). Καθημερινά περίπου 938.000 επιβάτες εξυπηρετούνται από τις Γραμμές 2 και 3 του Μετρό, ενώ η Γραμμή 1 των ΗΣΑΠ (μήκους 25,6 χλμ. με 24 Σταθμούς) εξυπηρετεί αντίστοιχα 460.000 επιβάτες. Οι Αθηναίοι έχουν τη δυνατότητα να πραγματοποιούν "συνδυασμένες διαδρομές" εξοικονομώντας πολύτιμο χρόνο στις καθημερινές τους μετακινήσεις. Ενδεικτικά αναφέρεται ότι με το Μετρό χρειάζονται μόλις 14 λεπτά για να καλυφθεί η απόσταση Σύνταγμα-Χαλάνδρι, ενώ με το αυτοκίνητο η ίδια απόσταση καλύπτεται σε 45 λεπτά σε ώρες αιχμής (www.googlemaps.gr).

Το ΤΡΑΜ διαθέτει δύο Γραμμές συνολικού μήκους 26,1 χλμ. που συγκλίνουν στη λεωφόρο Ποσειδώνος στο ύψος του Παλαιού Φαλήρου. Με τη λειτουργία του ΤΡΑΜ επιτυγχάνεται η σύνδεση του κέντρου της Αθήνας με την παραλιακή ζώνη έως το Ελληνικό (Γραμμή 1) και του Νέο Φαλήρου με τη Γλυφάδα (Γραμμή 2) (www.googlemaps.gr).

Τέλος, το δίκτυο του Προαστιακού Σιδηροδρόμου εξασφαλίζει πρόσβαση στο Διεθνές Αεροδρόμιο Αθηνών "Ελευθέριος Βενιζέλος" σε 40 περίπου λεπτά από το κέντρο της πόλης.

2.6.3. Αποκλειστικές Λωρίδες Λεωφορείων

Το έντονο κυκλοφοριακό πρόβλημα που επικρατεί στην πρωτεύουσα οδήγησε στην εφαρμογή τεχνικών που συμβάλλουν στην αποτελεσματική διαχείριση της κυκλοφορίας και στη λήψη μέτρων με στόχο την αναβάθμιση των υπηρεσιών των Μέσων Μαζικής Μεταφοράς (ΜΜΜ) και κατά συνέπεια τη βελτίωση της ποιότητας των μετακινήσεων. Ένα τέτοιο μέτρο είναι η προνομιακή μεταχείριση των ΜΜΜ κατά μήκος των οδικών αρτηριών, με χωροθέτηση Αποκλειστικών Λωρίδων Λεωφορείων (ΑΛΛ) Μέσων Μαζικής Μεταφοράς. Τα τελευταία χρόνια ο ΟΑΣΑ σε συνεργασία με το Υπουργείο Υποδομών Μεταφορών και Δικτύων, ανέπτυξε ένα συνεκτικό δίκτυο λεωφορειολωρίδων μήκους 50 περίπου χλμ. Τα σημαντικότερα οφέλη από την λειτουργία των ΕΛΛ συνοψίζονται στα εξής :

- Αύξηση της ταχύτητας των ΜΜΜ που κινούνται σε ΕΛΛ σε 23 χλμ /ώρα
- Βελτίωση της ποιότητας μεταφοράς των ΜΜΜ με αποτέλεσμα να είναι περισσότερο ελκυστικά για το επιβατικό κοινό
- Αύξηση της επιβατικής κίνησης των ΜΜΜ και μείωση της χρήσης των ΙΧ οχημάτων
- Μείωση της κατανάλωσης καυσίμων για τα ΜΜΜ
- Βελτίωση των περιβαλλοντικών συνθηκών

3. ΙΣΟΧΡΟΝΕΣ ΚΑΜΠΥΛΕΣ ΓΙΑ ΜΕΣΑ ΜΑΖΙΚΗΣ ΜΕΤΑΦΟΡΑΣ

Σε αυτό το κεφάλαιο θα εισαχθεί η προσέγγιση που ακολουθήθηκε στο πρόβλημα υπολογισμού ισοχρονικών καμπυλών με αφετηρία το κέντρο της Αθήνας. Πρώτα θα περιγραφούν τα δεδομένα που διατίθενται και την απαραίτητη επεξεργασία τους και ύστερα το μοντέλο που δημιουργήθηκε.

3.2. Χαρακτηριστικά δεδομένων

Το πρώτο μέλημα ήταν να συλλεχθούν δεδομένα για το δίκτυο των δημόσιων μέσων μαζικής μεταφοράς της Αθήνας.

Είναι απαραίτητο να δημιουργηθεί ένας κατευθυνόμενος γράφος με βάρη στις ακμές του που θα περιγράφει το δίκτυο των δημόσιων συγκοινωνιών της Αθήνας. Ο γράφος θα πρέπει να αποτελείται από τα εξής μέρη :

- Κόμβοι οι οποίοι αντιπροσωπεύουν τα σημεία όπου υπάρχουν στάσεις. Οι κόμβοι θα πρέπει να περιγράφονται από τις γεωγραφικές συντεταγμένες τους.

- Ακμές οι οποίες αντιπροσωπεύουν τους άξονες πάνω στους οποίους κινούνται τα δημόσια μέσα μεταφοράς.

- Τα βάρη των ακμών τα οποία αντιπροσωπεύουν το χρόνο προσπέλασης κάθε ακμής.

Εδώ πρέπει να σημειωθεί ότι η μοντελοποίηση των απλών οδικών δικτύων είναι αρκετά απλή. Έτσι τα μέρη που περιγράφηκαν θα ήταν αρκετά και η βέλτιστη διαδρομή από ένα σημείο Α σε ένα σημείο Β δηλαδή η διαδρομή με τον ελάχιστο συνολικό χρόνο θα μπορούσε να υπολογιστεί με την εφαρμογή ενός αλγορίθμου εύρεσης βέλτιστων μονοπατιών στο γράφο όπως ο Dijkstra.

Από την άλλη πλευρά τα δίκτυα των Μέσων Μαζικής Μεταφοράς, μπορούν να μοντελοποιηθούν με παρόμοιο τρόπο, με τη διαφορά ότι εκτός από τις γεωγραφικές πληροφορίες πρέπει να ληφθούν υπόψη και τα χρονικά διαγράμματα των δρομολογίων. Συνεπώς προστίθεται μία ακόμα παράμετρος για να ολοκληρωθεί η δημιουργία του γράφου του δικτύου, το οποίο είναι:

- Η συχνότητα κάθε δρομολογίου

Θα μπορούσαμε επίσης να προσθέσουμε συγκεκριμένο πρόγραμμα δρομολογίων αν είχαμε την ακριβή ώρα ή ώρες που το κάθε λεωφορείο επισκέπτεται την κάθε στάση. Τότε το κόστος της ακμής θα ήταν μια γραμμική συνάρτηση που αντιστοιχίζει κάθε πιθανή ώρα άφιξης στον κόμβο της ακμής σε ένα διαφορετικό κόστος διαδρομής κατά μήκος της ακμής αυτής.

Παρακάτω περιγράφεται η πηγή των δεδομένων και η αρχική τους μορφή, πριν την επεξεργασία τους ώστε να αποτελέσουν το γράφο πάνω στον οποίο θα εργαστούμε.

Οι θέσεις των στάσεων, υπάρχουν στο διαδύκτιο στην ιστοσελίδα http://geodata.gov.gr , σε μορφή txt. Οι πληροφορίες που παρέχονται είναι ο κωδικός κάθε στάσης, το όνομά της, το όνομα της οδού πάνω στη οποία βρίσκεται, η προηγούμενη και η επόμενη στάση και τέλος τα λεωφορεία που την επισκέπτονται. Στον Πίνακα 1 φαίνεται η δομή των δεδομένων μετά την εισαγωγή τους στο QGis.

code	name	street	previous	next	bus
2802	ΔΙΚΗΓΟΡΙΚΑ	ΛΕΩΦ.ΠΟΣΕΙΔΩΝΟΣ	ΕΘΝ.ΑΝΤΙΣΤΑΣΕΩΣ	ΒΕΛΗ	140 Α1 Α2 Χ96
2788	ΔΙΚΗΓΟΡΙΚΑ	ΛΕΩΦ.ΠΟΣΕΙΔΩΝΟΣ	ΕΥΡΥΑΛΗΣ	ΕΘΝ.ΑΝΤΙΣΤΑΣΕΩΣ	140 Α1 Α2 Χ96
2801	2η ΓΛΥΦΑΔΑΣ	ΛΕΩΦ.ΠΟΣΕΙΔΩΝΟΣ	ΑΡΤΕΜΙΔΟΣ	ΕΘΝ.ΑΝΤΙΣΤΑΣΕΩΣ	140 Α1 Α2 Χ96

Πίνακας 3 : Ο πίνακας ιδιοτήτων των στάσεων του δικτύου ΜΜΜ της Αθήνας

Οι άξονες πάνω στους οποίους κινούνται τα ΜΜΜ δόθηκαν από τον ΟΑΣΑ προς χρήση στην παρούσα εργασία. Για την κάθε ακμή του δικτύου παρέχεται η πληροφορία του ποιο λεωφορείο, ή τραμ ή μετρό τη διασχίζει, η συχνότητα του εν λόγω μέσου, η πρώτη και η τελευταία διαδρομή της ημέρας, ο αριθμός των κλάδων γραμμής, ενώ υπάρχει και ένας κωδικός που φανερώνει αν το δρομολόγιο εκτελεί τη διαδρομή «Αφετηρία – Τέρμα» ή «Τέρμα – Αφετηρία». Στον Πίνακα 4 φαίνεται η μορφή των δεδομένων μας όπως μας δόθηκαν, σε φύλλο Excell.

Πίνακας 4 : Χαρακτηριστικά βάση της ονομασίας γραμμής κάθε μέσου

ΓΡΑΜΜΗ (LINE)	ΟΝΟΜΑΣΙΑ	ΑΡΙΘΜΟΣ ΚΛΑΔΩΝ (ΚΑΤΕΥΘΥΝΣΕΙΣ)	ΚΥΚΛΙΚΗ ΔΡΟΜΟΛΟΓΗΣΗ ***	ΗΜΕΡΗΣΙΑ ΔΡΟΜΟΛΟΓΙΑ (ΚΑΘΗΜΕΡΙΝΗ)	ΤΥΠΙΚΗ ΣΥΧΝΟΤΗΤΑ (ΚΑΘΗΜΕΡΙΝΗ)	ΤΥΠΙΚΟΣ ΧΡΟΝΟΣ ΔΙΑΔΡΟΜΗΣ	ΦΟΡΕΑΣ ΠΑΡΟΧΗΣ ΣΥΓΚΟΙΝ	ΜΗΚΟΣ (μέτρα)	ΠΕΡΙΟΧΗ ΑΦΕΤΗΡΙΑΣ	ΠΕΡΙΟΧΗ ΤΕΡΜΑΤΟΣ
1	ΠΛ.ΑΤΤΙΚΗΣ-ΚΑΛΛΙΘΕΑ-ΜΟΣΧΑΤΟ	2		66	12	144	ΟΣΥ	23621	ΠΛΑΤ. ΑΤΤΙΚΗΣ	ΜΟΣΧΑΤΟ
2	ΑΝΩ ΚΥΨΕΛΗ-ΠΑΓΚΡΑΤΙ-ΚΑΙΣΑΡΙΑΝΗ	2		69	12	110	ΟΣΥ	16790	ΑΝΩ ΚΥΨΕΛΗ	ΚΑΙΣΑΡΙΑΝΗ

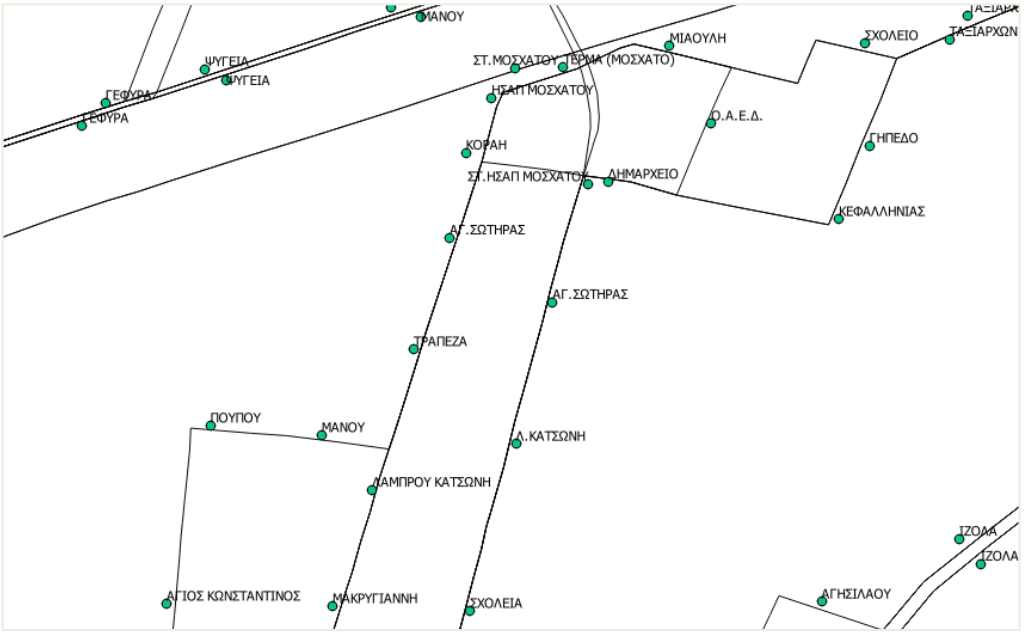
Για κάθε γραμμή υπάρχουν αναλυτικές πληροφορίες ανά κλάδο, οπως φαίνεται στον Πίνακα 5.

Πίνακας 5 : Χαρακτηριστικά βάση του κλάδου κάθε γραμμής

ΚΩΔΙΚΟΣ ΓΡΑΜΜΗΣ (LINE)	ΚΩΔ. ΚΛΑΔΟΥ (ROUTE)	ΚΑΤΕΥΘΥΝΣΗ ΚΛΑΔΟΥ	ΟΝΟΜΑΣΙΑ ΚΑΤΕΥΘΥΝΣΗΣ	ΜΗΚΟΣ (μέτρα)	ΗΜΕΡΗΣΙΑ ΔΡΟΜΟΛΟΓΙΑ			ΧΡΟΝΟΣ ΔΙΑΔΡΟΜΗΣ		ΚΑΘΗΜΕΡΙΝΗ				
					ΚΑΘΗΜΕΡΙΝΗ	ΣΑΒΒΑΤΟ	ΚΥΡΙΑΚΗ	ΤΥΠΙΚΟΣ	ΠΕΡΙΟΔΟΣ ΑΙΧΜΗΣ	ΠΡΩΤΟ ΔΡΟΜ.	ΤΕΛΕΥΤΑΙΟ ΔΡΟΜ.	ΣΥΧΝ. ΑΙΧΜΗΣ	ΣΥΧΝ. ΕΚΤΟΣ ΑΙΧΜΗΣ	ΣΥΧΝ. ΑΚΡΑ ΗΜΕΡΑΣ
1	0530	Από Αφετηρία	ΠΛ.ΑΤΤΙΚΗΣ-ΚΑΛΛΙΘΕΑ-ΜΟΣΧΑ	11805	65	48	36	72	73	05:00	23:35	12	15	20
1	0531	Από Τέρμα	ΜΟΣΧΑΤΟ-ΚΑΛΛΙΘΕΑ-ΠΛ.ΑΤΤΙΚ	11816	68	47	36	72	73	05:00	23:30	12	15	20
2	0536	Από Αφετηρία	ΑΝΩ ΚΥΨΕΛΗ-ΠΑΓΚΡΑΤΙ-ΚΑΙΣΑΙ	9113	68	56	39	55	55	06:25	23:25	12	18	25
2	0537	Από Τέρμα	ΚΑΙΣΑΡΙΑΝΗ-ΠΑΓΚΡΑΤΙ-ΑΝΩ ΚΥ	7677	71	58	40	55	55	06:30	23:35	12	18	25

Τα αντίστοιχα γεωγραφικά δεδομένα των ακμών που διατέθηκαν περιείχαν τον κωδικό γραμμής (LINE) και τον κωδικό κλάδου (ROUTE). Συνεπώς έγινε μια απλή ένωση με επιλεγμένα στοιχεία από τους πίνακες Excel για να προκύψουν πίνακες με ολοκληρωμένη γεωγραφική και μη πληροφορία.

Για τη συγκεκριμένη εργασία είναι πολύ σημαντική η πληροφορία της κατεύθυνσης των ακμών, η οποία δεν παρέχεται. Είναι αυτονόητο ότι κάθε γραμμή λεωφορείου/μετρό/τραμ μπορεί να κινηθεί μόνο προς μία κατεύθυνση, γεγονός που επηρεάζει καθοριστικά τη διαδικασία δρομολόγησης αφού εισάγει επιπλέον περιορισμούς. Επίσης οι στάσεις των λεωφορείων (κόμβοι) κείνται εκατέρωθεν των γραμμικών αξόνων, όπως δηλαδή στην πραγματικότητα (Εικόνα 13), όμως για να πραγματοποιηθεί η ανάλυση συντομότερης διαδρομής θα πρέπει να «πέφτουν» πάνω στο οδικό δίκτυο.



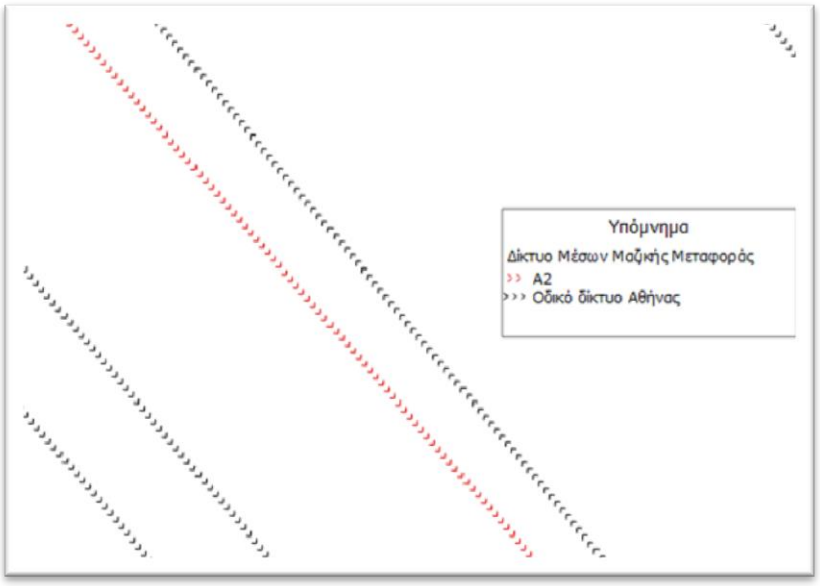
Εικόνα 13: Εικόνα στην οποία φαίνεται η θέση και το όνομα των στάσεων σε σχέση με τις ακμές του δικτύου

Στα επόμενα κεφάλαια αναλύεται η διαδικασία αντιμετώπισης αυτών και άλλων διορθώσεων που κρίθηκε απαραίτητο να γίνουν.

3.2.1. Θέματα προς επίλυση και αναπαράσταση οδικού δικτύου

Όπως αναφέρθηκε στο προηγούμενο εδάφιο είναι ιδιαίτερα σημαντικό να διορθωθεί η κατεύθυνση των ακμών. Οι αλγόριθμοι δρομολόγησης λειτουργούν με κατευθυνόμενους γράφους και είναι αυτονόητο ότι τα αποτελέσματα δεν θα είχαν καμία ουσιαστική σημασία εάν θεωρείτο ότι η κίνηση μπορεί να πραγματοποιηθεί προς κάθε κατεύθυνση.

Η διόρθωση αυτή υπήρξε μια αρκετά επίπονη και χρονοβόρα διαδικασία, αφού είναι απαραίτητο να γίνει χειροκίνητα και για κάθε κλάδο ξεχωριστά. Έτσι αφού απεικονίστηκαν οι ακμές του δικτύου των ΜΜΜ στο λογισμικό QGIS, οπτικοποιήθηκαν με βάση την κατεύθυνση. Στη συνέχεια εισαγάγαμε το οδικό δίκτυο της Αθήνας (στο οποίο είναι διαθέσιμη η πληροφορία κατεύθυνσης) και οπτικοποιήθηκε και αυτό με τον ίδιο τρόπο. Τέλος επιλεγόταν κάθε γραμμή (LINE) ξεχωριστά και ο έλεγχος έγινε κυρίως με βάση το οδικό δίκτυο. Επειδή όμως στην Αθήνα υπάρχουν κάποιες περιπτώσεις που τα δημόσια μέσα κινούνται αντίθετα από την προκαθορισμένη φορά του δρόμου, όπως στην Εικόνα 14, κρίθηκε απαραίτητος και ο έλεγχος με βάση την ιστοσελίδα του ΟΑΣΑ (www.telematics.oasa.gr).



Εικόνα 14 : Η κίνηση του λεωφορείο Α2 αντίθετης φοράς από την οδό Πανεπιστημίου

Κατάτμηση των γραμμών στις στάσεις

Αφού ολοκληρώθηκε η πρώτη διαδικασία διόρθωσης ήταν απαραίτητο να δομηθεί το γραμμικό δίκτυο έτσι ώστε κάθε ακμή να αντιπροσωπεύει τη διαδρομή ενός μέσου από μία στάση ως την επόμενή της. Για τον σκοπό αυτό δημιουργήθηκε ένα μοντέλο μέσω του εργαλείο Model Builder του λογισμικού ArcGIS (Εικόνα 15). Η διαδικασία που ακολουθεί το μοντέλο έχει ως εξής :

- Αρχικά επιλέγεται από το πηγαίο αρχείο των αξόνων του δικτύου μας ο πρώτος κλάδος (ROUTE) της πρώτης γραμμής (LINE).

- Κατόπιν γίνεται ενοποίηση των γραμμών που αποτελούν τον συγκεκριμένο κλάδο ώστε να αποτελεί μία ενιαία οντότητα.

- Ύστερα δημιουργείται μία ζώνη (buffer) πλάτους 50 μέτρων δεξιά από την κατεύθυνση του κλάδου, αφού οι στάσεις βρίσκονται πάντα δεξιά από την κίνηση των Μέσω Μαζικής Μεταφοράς.

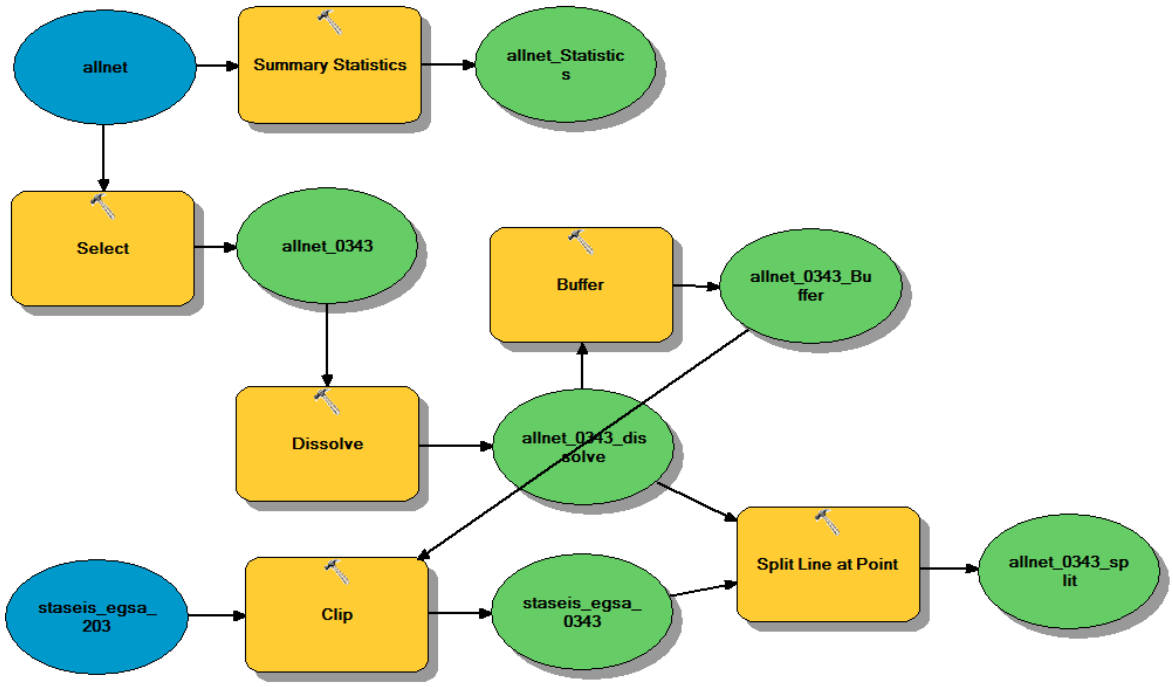
- Από το αρχείο των στάσεων επιλέγονται μόνο εκείνες τις οποίες επισκέπτεται η συγκεκριμένη γραμμή. Αυτό σημαίνει ότι θα επιλεχθούν οι στάσεις και των δύο κλάδων (ROUTE) και όχι μόνο του κλάδου που επεξεργαζόμαστε.

- Για τον λόγο αυτόν γίνεται επιλογή των στάσεων που βρίσκονται εντός του δημιουργημένου Buffer, με το εργαλείο Clip.

- Τέλος, η γραμμή του κλάδου «κόβεται» με βάση τα τελευταία επιλεγμένα σημεία, με το εργαλείο Split line at points.

- Η διαδικασία επαναλαμβάνεται για τον επόμενο κλάδο.

Τα εξαγόμενα αρχεία ενώνονται σε ένα συνολικό αρχείο που πλέον θα αποτελέσει τη βάση για την κατασκευή του γράφου από τον οποίο θα υπολογιστούν τελικά οι ισοχρονικές καμπύλες.



Εικόνα 15 : Απόδοση του μοντέλου που περιγράφηκε

3.3. Εργαλεία που χρησιμοποιήθηκαν

3.3.1. Συστήματα Βάσεων Χωρικών Δεδομένων

Οι βάσεις δεδομένων αποτελούν ένα σημαντικό και αναπόσπαστο κομμάτι στην ανάπτυξη διαδικτυακών και μη εφαρμογών. Ιδιαίτερα για τις χωρικές εφαρμογές, η χρήση βάσεων χωρικών δεδομένων (ΒΧΔ) είναι περισσότερο επιβεβλημένη λόγω της ιδιομορφίας των γεωγραφικών δεδομένων που συνίσταται στη χωρική και θεματική τους ταυτότητα καθώς και στις αυξημένες απαιτήσεις αποθηκευτικού χώρου. Η χωρική διάσταση (γεωμετρία) υποστηρίζεται από τις ΒΧΔ με την υλοποίηση ενός ειδικού τύπου δεδομένων geometry ο οποίος χρησιμοποιείται για την αποθήκευση των συντεταγμένων που ορίζουν τη γεωμετρία σε ένα καθορισμένο σύστημα αναφοράς.

Η επιλογή του Συστήματος Διαχείρισης Βάσεων Χωρικών Δεδομένων (Spatial Database Management System - SDBMS) είναι μείζονος σημασίας για την αποτελεσματική αποθήκευση και διαχείριση των δεδομένων και ακόμα περισσότερο η απόδοση του έχει αντίκτυπο στη διαδικτυακή εφαρμογή η οποία αντλεί τα δεδομένα από αυτό (Adnan et al., 2010). Τα δημοφιλέστερα SDBMS είναι τα Oracle Spatial, IBM DB2 και Microsoft SQL Server Spatial που αποτελούν εμπορικά, κλειστά συστήματα και η PostgreSQL με τη

χωρική επέκταση PostGIS που είναι ανοιχτού κώδικα ελεύθερο σύστημα. Στη παρούσα εργασία, ως SDBMS επιλέχθηκε η PostgreSQL / PostGIS η οποία περιγράφεται στη συνέχεια.

3.3.2. Σχεσιακή βάση δεδομένων PostgreSQL

Η PostgreSQL είναι μια σχεσιακή βάση δεδομένων ανοικτού κώδικα με πολλές δυνατότητες. Η ανάπτυξη της διαρκεί ήδη πάνω από δύο δεκαετίες και βασίζεται σε μια αποδεδειγμένα καλή αρχιτεκτονική η οποία έχει δημιουργήσει μια ισχυρή αντίληψη των χρηστών της γύρω από την αξιοπιστία, την ακεραιότητα δεδομένων και την ορθή λειτουργία. Είναι συμβατό με όλα τα κύρια λειτουργικά συστήματα όπως Linux, Unix (συμπεριλαμβανομένου του Mac OS X) και Windows. Είναι πλήρως συμβατή με το μοντέλο ACID[1] και υποστηρίζει πλήρως foreign keys, joins, views, triggers. Συμπεριλαμβάνει τους περισσότερους τύπους δεδομένων του SQL:2008 όπως INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL και TIMESTAMP. Επιπλέον, υποστηρίζει την αποθήκευση μεγάλων binary αντικειμένων όπως εικόνες, ήχους και βίντεο. Διαθέτει εγγενείς προγραμματιστικές διεπαφές για πολλές διαδεδομένες γλώσσες προγραμματισμού όπως C/C++, Java, .Net, Perl, Ruby, Python. Τέλος, έχει πολύ ισχυρό documentation.

Εκτός από τα παραπάνω που αποδεικνύουν ότι η PostgreSQL είναι μία πολύ αξιόπιστη και ευέλικτη επιλογή για σύστημα βάσεων δεδομένων και μάλιστα ανοικτού κώδικα, ένας ακόμα και μάλλον ο σημαντικότερος λόγος που οδήγησε στην επιλογή της είναι η χωρική επέκταση PostGIS. Το PostGIS είναι ένα ανοικτό project το οποίο προσθέτει στην PostgreSQL υποστήριξη για γεωγραφικά αντικείμενα. Στην πραγματικότητα, η PostGIS προσθέτει τη δυνατότητα στον PostgreSQL server να χρησιμοποιηθεί σαν εργαλείο διαχείρισης χωρικής βάσης δεδομένων σε γεωγραφικά συστήματα πληροφοριών (GIS) (https://www.postgresql.org/).

[1] Το ACID (atomicity, consistency, isolation, durability) είναι ένα σύνολο από ιδιότητες οι οποίες εγγυώνται ότι οι δοσοληψίες σε μία βάση δεδομένων εκτελούνται αξιόπιστα:

• Ατομικότητα. Εξασφαλίζει ότι οι πράξεις μίας δοσοληψίας είτε θα πετύχουν είτε θα αποτύχουν όλες. Δηλαδή κάθε δοσοληψία είτε ολοκληρώνεται επιτυχώς είτε εμφανίζεται σα να μην έχει ξεκινήσει καν.

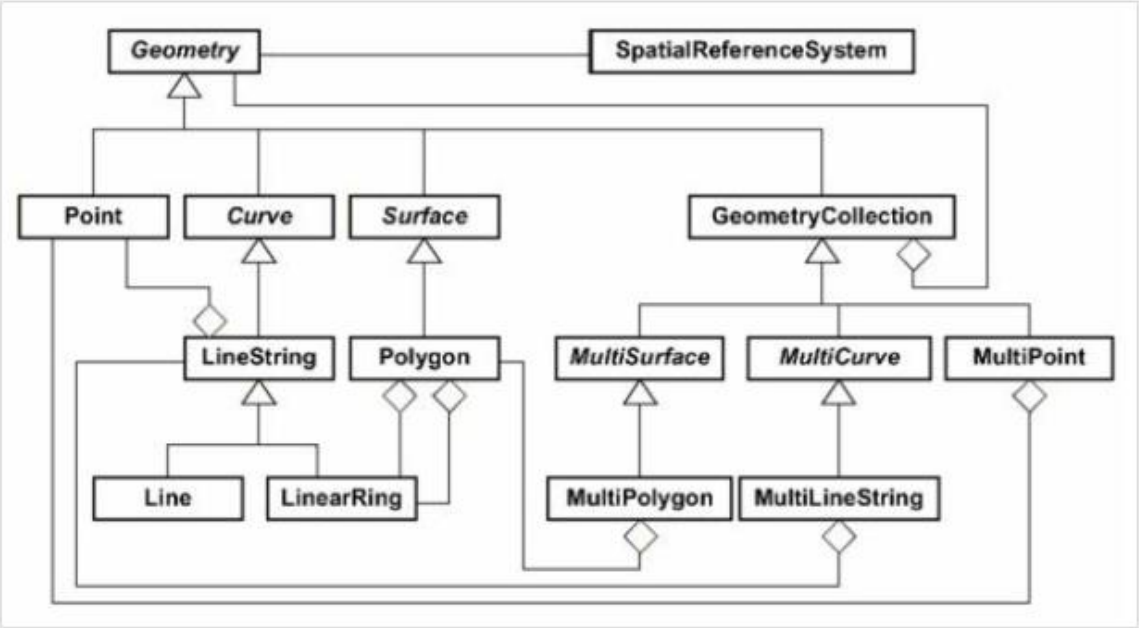
• Συνέπεια. Εξασφαλίζει ότι μετά από οποιαδήποτε δοσοληψία που πραγματοποιείται η βάση παραμένει σε συνεπή κατάσταση.

• Απομόνωση. Εξασφαλίζει ότι καμία δοσοληψία δε μπλέκει με την εκτέλεση μιας άλλης. Κατά συνέπεια, κάθε δοσοληψία συμπεριφέρεται σα να είναι η μόνη που τρέχει στο σύστημα.

• Μονιμότητα. Εξασφαλίζει ότι εφόσον μία δοσοληψία ολοκληρωθεί και καταχωρηθεί, δε υπάρχει περίπτωση αναίρεσης της.

Η PostGIS υποστηρίζει ένα υπερσύνολο των γεωγραφικών αντικειμένων που ορίζονται στη προδιαγραφή *Simple Features for SQL* του OGC. Στη προδιαγραφή αυτή ορίζονται όλοι οι γεωμετρικοί τύποι δεδομένων, απλοί και σύνθετοι (MultiPoint, MultiLineString, MultiPolygon) καθώς και άλλες περιπτώσεις, βάσει καθορισμένου συστήματος αναφοράς (Σχήμα 6). Το γεωμετρικό μοντέλο καλύπτει γεωγραφικά αντικείμενα τεσσάρων διαστάσεων (x, y, z, m). Επιπρόσθετα, η PostGIS περιλαμβάνει μεθόδους που προσδιορίζουν βασικές ιδιότητες των γεωμετρικών σχημάτων και αναλυτικές λειτουργίες GIS για γεωγραφική ανάλυση και εξαγωγή χρήσιμων πληροφοριών από τα δεδομένα (Πατρούμπας Κ., 2008).

Η χρήση της PostgreSQL με τη χωρικής επέκταση PostGIS ως εξυπηρετητή ΒΧΔ (Spatial Database Server) επιτρέπει τη σύνδεση της με χωρικούς εξυπηρετητές όπως ο GeoServer και τη διάχυση των δεδομένων που φιλοξενούνται στη βάση μέσω των υπηρεσιών του OGC. Επιπλέον, η χρήση γλωσσών σεναρίων στη πλευρά του Server όπως η PHP και η Python, επιτρέπει τη σύνδεση με τη PostgreSQL / PostGIS και τη σύνταξη SQL ερωτημάτων για διαχείριση και ανάλυση των δεδομένων. Τα αποτελέσματα των ερωτημάτων μπορούν να εμφανισθούν στην εφαρμογή του πελάτη μέσω της βιβλιοθήκης OpenLayers, προσδίδοντας στην εφαρμογή δυνατότητες ενός ισχυρού λογισμικού ΣΓΠ (Πατρούμπας Κ., 2008).



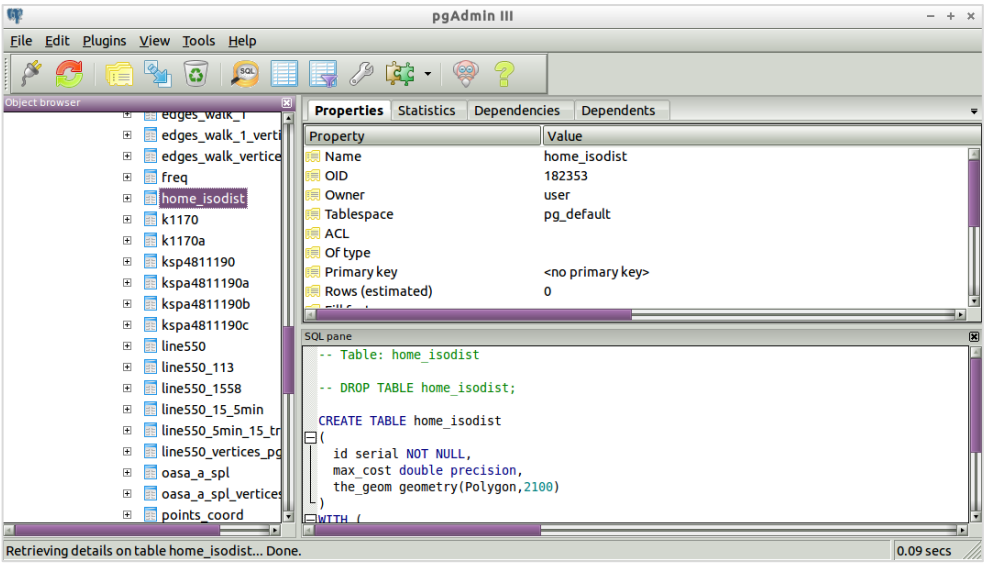
Σχήμα 6: Γεωμετρικοί τύποι κατά OGC – Simple Feature for SQL που υποστηρίζει η PostGIS.

Πίνακας 6: Βασικές μέθοδοι και λειτουργίες χωρικής ανάλυσης της PostGIS.

Μέθοδοι Προσδιορισμού Ιδιοτήτων	Λειτουργίες Χωρικής Ανάλυσης
Dimension (g)	Distance (a,b)
GeometryType (g)	Length (g)
SRID (g)	Area (g)
Envelope (g)	Centroid (g)
AsText (g)	Intersection (a,b)
AsBinary (g)	Union (a,b)
IsEmpty (g)	Difference (a,b)
IsSimple (g)	Buffer (g,d)

Η διαχείρηση μίας Postgres βάσης δεδομένων μπορεί να γίνει είτε από τη γραμμή εντολών είτε από το pgAdmin. Το pgAdmin είναι ένα ολοκληρωμένο σύστημα σχεδιασμού και διαχείρισης βάσεων δεδομένων PostgreSQL για συστήματα Unix και Windows και διατίθεται δωρεάν.

Στο κύριο παράθυρο εμφανίζεται η δομή των βάσεων δεδομένων. Υπάρχουν επιλογές δημιουργίας νέων αντικειμένων και διαγραφής ή επεξεργασίας ήδη υπάρχοντων. Επίσης με την επιλογή ενός αντικειμένου εμφανίζονται τα στοιχεία του αλλά και κάποια στατιστικά. Επίσης παρέχεται ένα παράθυρο αντίστροφης μηχανής SQL script. Αν δηλαδή επιλέξουμε να τροποποιήσουμε ένα πίνακα από τις επιλογές που έχουμε στο pgAdmin, δημιουργείται και αυτόματα το SQL script το οποίο μπορούμε είτε να αντιγράψουμε είτε να αποθηκεύσουμε για μελλοντική χρήση. Είναι λοιπόν ένα πολύ χρήσιμο εργαλείο, ιδιαίτερα για χρήστες με περιορισμένη εμπειρία στις βάσεις δεδομένων και την SQL.



Εικόνα 16 : Μορφή του pgAdmin

3.3.3. Η βιβλιοθήκη PgRouting

Το PgRouting είναι μια επέκταση ανοικτού κώδικα της PostGreSQL για την ανάπτυξη εφαρμογών δρομολόγησης δικτύου και ανάλυσης γραφημάτων. Στηρίζεται στην PostGIS, την γεωχωρική επέκταση της PostgreSQL.

Η PostGIS υπερέχει στην αποθήκευση και επεξεργασία χωρικών δεδομένων. Οι λειτουργίες μέτρησης απόστασης στην PostGIS απαντούν σε ερωτήσεις εγγύτητας δύο σημείων. Ωστόσο, υπολείπεται όταν η προσπάθεια εστιάζεται στην εύρεση απόστασης σε περιορισμένα μονοπάτια όπως οι δρόμοι. Επίσης είναι αδύνατο να εφαρμοστούν κόστοι και περιορισμοί πόρων σε αυτά τα ταξίδια. Για τους λόγους αυτούς δημιουργήθηκε το PgRouting (http://pgrouting.org/).

3.3.4. Χαρακτηριστικά που παρέχονται από το PgRouting

Το pgRouting περιλαμβάνει πολλές συναρτήσεις για την υλοποίηση αλγορίθμων βελτιστοποίησης, αλλά και για την προετοιμασία των δεδομένων. Εφόσον οι συναρτήσεις αυτές συμπεριλαμβάνονται στο PgRouting, έχουν το πρόθεμα pgr για να διαχωρίζονται από τις υπόλοιπες (pgRouting Guide, 2016). Μπορούν να κατηγοριοποιηθούν σε τέσσερις κύριες κατηγορίες :

- Συναρτήσεις builder για την δόμηση γράφων.
  Για παράδειγμα, εάν εισαχθούν όλες οι ακμές ενός δικτύου, θα δημιουργηθεί ένας

πίνακας κόμβων και θα συμπληρωθούν αυτόματα οι στήλες προέλευσης (source) και προορισμού (target) του πίνακα των ακμών.

- Συναρτήσεις ελέγχου και διόρθωσης.
  Συναρτήσεις που υποδεικνύουν τα προβληματικά σημεία του δικτύου μας και άλλες που διορθώνουν αυτόματα τα σφάλματα αυτά.

- Βοηθητικές συναρτήσεις που δεν σχετίζονται άμεσα με δρομολόγηση.
  Χρησιμοποιούνται για να δημιουργηθούν μήτρες, ισόχρονες, αρχεία διαφορετικής μορφοποίησης κ.λπ.

- Συναρτήσεις που υλοποιούν αλγόριθμους δρομολόγησης.

Συναρτήσεις Builder

- pgr_createTopology - Ενημερώνει τα πεδία προέλευσης (source) και στόχου (target) στο καθορισμένο πίνακα ακμών <edge_table>. Δημιουργεί επίσης έναν πίνακα κορυφών που ονομάζεται <edge_table> _vertices_pgr.

- pgr_createVerticesTable – Εάν τα πεδία προέλευσης (source) και στόχου (target) είναι ήδη συμπληρωμένα για έναν πίνακα ακμών <edge_table>, αυτή η εντολή δημιουργεί τον πίνακα κορυφών <edge_table> _vertices_pgr, από την πληροφορία αυτή.

Συναρτήσεις ελέγχου και διόρθωσης

Σε ένα ιδανικό δίκτυο, οι ακμές ενώνονται στο ίδιο σημείο και δεν υπάρχουν αδιέξοδα, όμως αυτό είναι σπάνιο όταν δουλεύουμε με πραγματικά δεδομένα. Γι' αυτό το PgRouting προσφέρει λειτουργίες επαλήθευσης που εντοπίζουν τα προβλήματα αυτά καθώς και λειτουργίες επιδιόρθωσης, ώστε να διορθώνονται αυτόματα (pgRouting Guide, 2016). Οι πιο σημαντικές από αυτές είναι οι παρακάτω:

- pgr_analyzeGraph – Αφού έχει δημιουργηθεί ο γράφος, αυτή η συνάρτηση ελέγχει το δίκτυο για αδιέξοδα και κενά στις συνδέσεις των ακμών.

- pgr_analyzeOneway – Ελέγχει την κατεύθυνση των ακμών και επισημαίνει εκείνες που παραβιάζουν κάποιον τέτοιο κανόνα.

- Pgr_nodeNetwork – Εξασφαλίζει ότι όλες οι ακμές συνδέονται σε κόμβους. Αντιμετωπίζει τα προβλήματα που προκύπτουν με την εισαγωγή επιπλέον ακμών ανάλογα με τις ανάγκες του δικτύου.

Συναρτήσεις δρομολόγησης

Οι συναρτήσεις δρομολόγησης είναι το μεγαλύτερο πλεονέκτημα του PgRouting. Μέσω αυτών εφαρμόζονται οι πιο γνωστοί αλγόριθμοι δρομολόγησης. Χωρίζονται σε αλγορίθμους συντομότερης διαδρομής με μοναδική αφετηρία, αλγορίθμους συντομότερης διαδρομής ανάμεσα σε ζεύγη σημείων και αλγορίθμους πολλαπλών διαδρομών. Πιο αναλυτικά:

Οι συναρτήσεις συντομότερης διαδρομής με μοναδική αφετηρία, παίρνουν σαν είσοδο έναν αρχικό κόμβο (source) και έναν τελικό (target), δημιουργούν όλες τις πιθανές διαδρομές και επιστρέφουν τη συντομότερη, δηλαδή εκείνη με το μικρότερο κόστος.Οι πιο σημαντικές είναι :

- pgr_dijkstra – Επιστέφει τη συντομότερη διαδρομή χρησιμοποιώντας τον αλγόριθμο Dijkstra.
- pgr_bdDijkstra – Επιστέφει τη συντομότερη διαδρομή χρησιμοποιώντας τον αλγόριθμο Dijksta για αμφίδρομους γράφους.
- pgr_astar - Επιστέφει τη συντομότερη διαδρομή χρησιμοποιώντας τον αλγόριθμο A*.
- pgr_bdAstar – Επιστρέφει τη συντομότερη διαδρομή χρησιμοποιώντας τον αλγόριθμο A* για αμφίδρομους γράφους.
- pgr_trsp – Επιστρέφει τη συντομότερη διαδρομή σε γράφους με περιορισμούς στις στροφές.
- pgr_ksp – Επιστρέφει τις πρώτες κ συντομότερες διαδρομές χρησιμοποιώντας τον αλγόριθμο Dijkstra.

Οι αλγόριθμοι συντομότερης διαδρομής ανάμεσα σε ζεύγη σημείων, διατρέχουν όλους τους κόμβους είτε συνδέονται με μία ακμή είτε με πολλαπλές. Παίρνουν σαν είσοδο έναν σταθμισμένο, κατευθυνόμενο γράφο και επιστρέφουν μια τετράγωνη μήτρα της συντομότερης διαδρομής μεταξύ του συνόλου των ζευγών κόμβων που περιλαμβάνει ο γράφος (pgRouting Guide, 2017).

- pgr_apspJohnson -Ζεύγος συντομότερης διαδρομής χρησιμοποιώντας τον αλγόριθμο του Johnson
- pgr_apspWarsall – Ζεύγος συντομότερης διαδρομής χρησιμοποιώντας τον αλγόριθμο Floyd-Warsall.

Οι αλγόριθμοι πολλαπλών διαδρομών παίρνουν σαν είσοδο έναν κόμβο αρχής και έναν πίνακα με τους κόμβους προορισμού. Ενδεικτικά:

- pgr_dijkstra+ - Αλγόριθμος πολλαπλών διαδρομών που εφαρμόζεται χρησιμοποιώντας τον Dijkstra.

Τέλος οι αλγόριθμοι βελτιστοποίησης προσπαθούν να βρουν την καλύτερη ακολουθία μετακίνησης δεδομένου ενός συνόλου κόμβων που πρέπει να διατρεχθούν. Θα αναφερθεί ένας από αυτούς.

- pgr_tsp – Το γνωστό πρόβλημα του πλανώδιου πωλητή λυμένο με ευρετικό κανόνα και προσεγγιστικούς αλγορίθμους.

3.3.5. Εισαγωγή ESRI shapefiles στο PgRouting

Μέχρι σήμερα ο πιο διαδεδομένος μορφότυπος γεωγραφικών δεδομένων είναι το Shapefile της ESRI**.** Το PostGIS εκδίδεται με δύο ενσωματωμένα εργαλεία για την εισαγωγή αυτών των αρχείων στη βάση.Το βοηθητικό πρόγραμμα shp2pgsql που καλείτε από τη γραμμή εντολών πληκτρολογώντας shp2pgsql-gui και ουσιαστικά το ίδιο πρόγραμμα μέσω του PgAdmin το οποίο όμως είναι διαθέσιμο μόνο σε κάποιες εκδόσεις του PostGIS.

Υπάρχουν επίσης πολλαπλά εργαλεία για εισαγωγή διάφορων τύπων δεδομένων, όπως text files, kml, osm κ.λπ. Δεν θα αναφερθούμε περαιτέρω σε αυτά καθώς δεν τα χρησιμοποιήσαμε στην παρούσα εργασία.

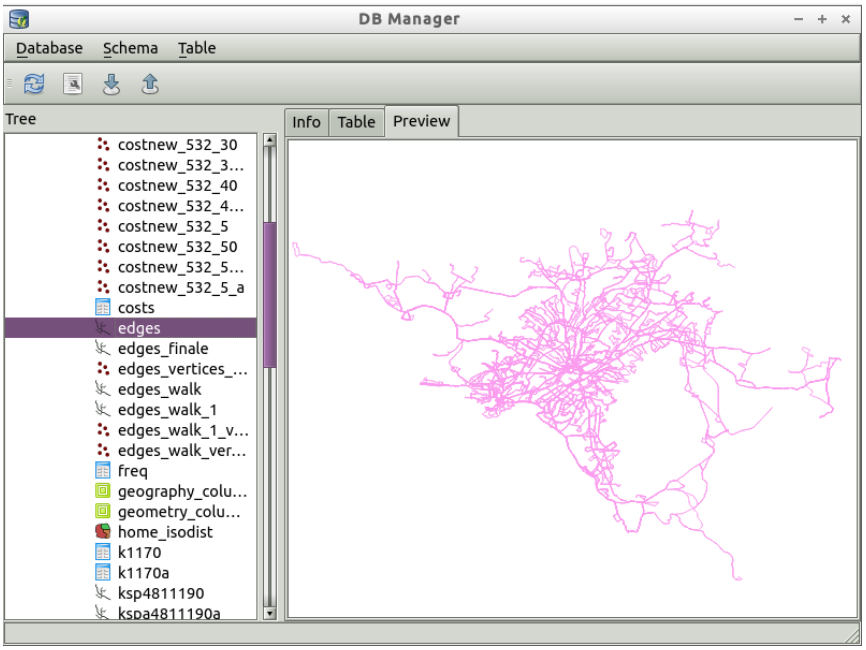
3.3.6. Χρήση του QGIS σε συνδιασμό με το PgRouting

Σε αυτό το σημείο θα διερευνήσουμε το ΣΓΠ QGIS και τα σχετικά plugins που συχνά χρησιμοποιούνται σε συνδιασμό με το PgRouting.

Το πρώτο βήμα για να χρησιμοποιηθεί το QGIS σε συνδιασμό με το PostGIS είναι να δημιουργηθεί σύνδεση με τη βάση δεδομένων. Μόλις συμβεί αυτό, η σύνδεση αποθηκεύεται για μελλοντική εργασία στο QGIS και μπορεί να χρησιμοποιηθεί και από διαφορετικά plugins.

Για να δημιουργηθεί σύνδεση με το PostGIS αρκεί η επιλογή ' *Add PostGIS Layer'* από το μενού του QGIS και κατόπιν συμπληρώνονται τα στοιχεία της βάσης δεδομένων και της θύρας που χρησιμοποιείται. Επίσης υπάρχουν οι επιλογές τροποποίησης ή και διαγραφής κάποιας σύνδεσης.

Το εργαλείο Db Manager είναι απαραίτητο εργαλείο για οποιαδήποτε είδος εργασίας βάσης δεδομένων και την προβολή χωρικών ερωτημάτων μέσω του PgRouting. Χρησιμοποιεί τις ίδιες συνδέσεις που ορίζονται όταν προστίθεται ένα PostGIS layer, ενώ παρέχει και τη δυνατότητα προεπισκόπησης δεδομένων που περιέχουν γεωγραφική πληροφορία.



Εικόνα 17 : Παράθυρο προεπισκόπισης θεματικού επιπέδου του DB Manager

Γενικά είναι ένα πολύτιμο εργαλείο που παρέχει τις παρακάτω δυνατότητες :

- Ορισμός χωρικών ερωτημάτων και μεταφορά τους για θέαση στο χάρτη. Από τα χωρικά ερωτήματα μπορούν να εξαχθούν είτε γεωμετρικά δεδομένα είτε δεδομένα ράστερ.
- Εισαγωγή νέων χωρικών πινάκων. Υποστηρίζει πολλές μορφές δεδομένων αφού για την εισαγωγή χρησιμοποιείται η βιβλιοθήκη GDAL(Geospatial Data Abstraction Library).

- Εξαγωγή πινάκων από τη βάση δεδομένων και εισαγωγή τους στο QGIS για προβολή στον χάρτη (εφόσον υπάρχει μία στήλη γεωμετρίας ή είναι ράστερ)

Το πιο χρήσιμο χαρακτηριστικό του DB Manager, όσον αφορά το PgRouting, είναι το παράθυρο SQL, που παρέχει, για τη σύνταξη ερωτημάτων . Μέσω αυτού του εργαλείου μπορούν να δομούνται τα ερωτήματα, να εκτελούνται αλλά και να οπτικοποιούνται στον χάρτη. Για όλες αυτές τις λειτουργίες πρέπει να δηλώνεται κάθε φορά η στήλη που περιέχει τη γεωμετρική πληροφορία και η αναγνωριστική στήλη (id).

Εικόνα 18 : Παράθυρο σύνταξης SQL ερωτημάτων του DB Manager

3.4. Κατασκευή γράφου

Στο κεφάλαιο αυτό θα αναλυθούν οι διαδικασίες δόμησης του γράφου του δικτύου των δημόσιων μέσων μεταφοράς. Υπάρχουν δύο βασικά χαρακτηριστικά για κάθε σωστή μηχανή δρομολόγησης. Απαιτούνται σωστά δομημένα δεδομένα ακμών και κόμβων και εφαρμογή του κόστους με ουσιαστικό τρόπο.

Οι σωστά δομημένες ακμές συνδέονται και στα δύο άκρα με κόμβους, έτσι ώστε όπου είναι δυνατόν να υπάρχει στάση, το σημείο αυτό να βρίσκεται είτε στην αρχή είτε στο τέλος της ακμής. Επίσης η κατεύθυνση των δεδομένων πρέπει να είναι ορισμένη ώστε οι ακμές που συνδέονται μέσω των κόμβων να έχουν κοινή φορά, όπως θα συνεχιζόταν και η πορεία ενός λεωφορείου μετά από τη στάση. Κατά τη διαδικασία διόρθωσης του δικτύου διορθώθηκε η φορά των ακμών και στη συνέχεια θα δηλωθεί και στη βάση δεδομένων.

Υπάρχουν και άλλοι παράγοντες που είναι πιο δύσκολο να καθοριστούν. Κάποιοι από αυτούς είναι τα δεδομένα που σχετίζονται με την κυκλοφορία των λεωφορείου, όπως για παράδειγμα η ταχύτητα σε κάθε ακμή του δικτύου, ο χρόνος αναμονής στις στάσεις μετεπιβίβασης και άλλα στοιχεία που μπορεί να διαφέρουν ανάλογα με την ώρα, την ημέρα και άλλες συνθήκες. Στις παραμέτρους αυτές έχουν γίνει κάποιες λογικές παραδοχές ώστε τα αποτελέσματα του αλγορίθμου να συγκλίνουν στη πραγματικότητα.

Ο υπολογισμός της συντομότερης διαδρομής έχει να κάνει με την ελαχιστοποίηση του κόστους διαδρομής ανάμεσα σε δύο σημεία. Ως κόστος μπορεί να θεωρηθεί οποιαδήποτε παράμετρος ορίσει ο σχεδιαστής, όπως για παράδειγμα η ελαχιστοποίηση του βαδίσματος, η ελαχιστοποίηση του χρόνου οδήγησης, η ελαχιστοποίηση κατανάλωσης καυσίμου ή η μεγιστοποίηση καύσης θερμίδων. Στο λογισμικό PgRouting, όπως και στον αλγόριθμο του Dijkstra στόχος είναι το ελάχιστο δυνατό κόστος, συνεπώς δεν θα πρέπει να ορίζονται κόστη που επιθυμείται να μεγιστοποιηθούν. Στην περίπτωση της εργασίας αυτής στόχος είναι να ελαχιστοποιηθεί το χρονικό κόστος μετακίνησης μέσω των μέσων μαζικής μεταφοράς.

3.4.1. Κατασκευή γράφου

Σε αυτό το στάδιο τα δεδομένα είναι έτοιμα ώστε να δημιουργηθεί ο γράφος. Για τον σκοπό αυτό, χρησιμοποιήθηκε η βιβλιοθήκη pgRouting της Postgres. Μία από τις συναρτήσεις που παρέχει η εν λόγω βιβλιοθήκη είναι το pgr_createTopology. Με τη συνάρτηση αυτή ουσιαστικά χτίζεται η τοπολογία του γράφου μας βασισμένος στη γεωμετρία του δικτύου.

Οι παράμετροι που πρέπει να δηλωθούν είναι οι εξής :

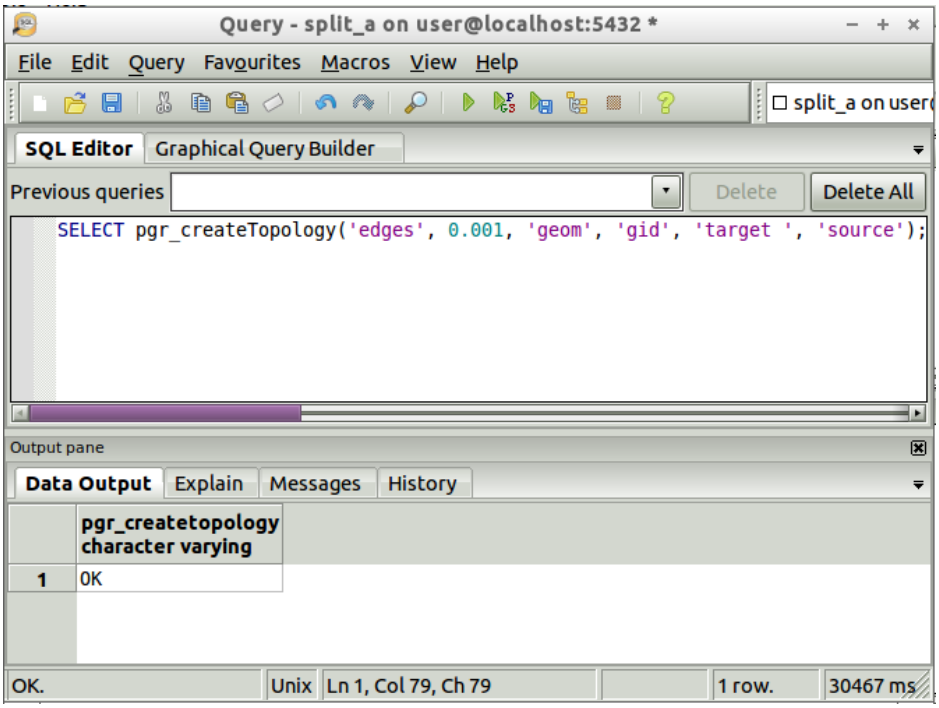
- Ο πίνακας ακμών
- Η ανοχή ταύτισης θέσης σε μονάδες του προβολικού συστήματος που χρησιμοποιήθηκε (snapping tolerance)
- Η στήλη που περιέχει την πληροφορία της γεωμετρίας (geom)
- Η στήλη του πρωτεύοντος κλειδιού του πίνακα ακμών (id)
- Το όνομα της στήλης που θα αποθηκευτούν οι κωδικοί των κόμβων αφετηρίας κάθε ακμής (SOURCE)
- Το όνομα της στήλης που θα αποθηκευτούν οι κωδικοί των κόμβων προορισμού κάθε ακμής (TARGET)

Συνεπώς, είναι απαραίτητο να δημιουργηθούν οι στήλες SOURCE και TARGET καθώς θα χρειαστούν στο επόμενα βήματα.

Ο κώδικας σε SQL είναι ο εξής :

SELECT pgr_createTopology('edges', 0.001, 'geom', 'gid', 'target ', 'source');

Η συνάρτηση επιστρέφει OK αν έχει κτιστεί η τοπολογία και FAIL αν υπήρξε κάποιο σφάλμα κατά τη διαδικασία. Επίσης ενημερώνει το χρήστη για τον αριθμό των κόμβων που δημιουργήθηκαν και τον αριθμό των εγγραφών που επηρεάστηκαν στον πίνακα των ακμών.



Εικόνα 19: Παράδειγμα εφαρμογής της συνάρτησης pgr_createTopology

Όταν ολοκληρωθεί η ανάλυση αφενός δημιουργείται ένα νέο επίπεδο κόμβων με μοναδικό κωδικό (id) και γεωμετρία (geom) στον καθένα, αφετέρου επηρεάζεται ο αρχικός πίνακας των ακμών και συμπληρώνονται τα στοιχεία SOURCE και TARGET από το id των κόμβων. Ουσιαστικά δηλαδή έχουν δημιουργηθεί κόμβοι, που αντιπροσωπεύουν τις θέσεις των στάσεων, στην αρχή και στο τέλος κάθε ακμής. Το νέο επίπεδο που δημιουργείται παίρνει το όνομα του από το αρχικό με την κατάληξη _vertices_pgr.

Στην περίπτωση της εργασίας ο αρχικός πίνακας ονομάζεται edges και ο πίνακας των κόμβων edges_vertices_pgr. Επίσης δημιουργήθηκαν 7080 κόμβοι.

Επειδή στις μετακινήσεις με τα Δημόσια Μέσα Μεταφοράς, ένα σημαντικό στοιχείο είναι η σύνδεση των διαφορετικών μέσων και η μετεπιβίβαση, έπρεπε να δημιουργηθούν μονοπάτια περπατήματος για τους επιβάτες. Όπως και σε κάποιες διαδικτυακές εφαρμογές

που αναφέρθηκαν στο πρώτο κεφάλαιο έτσι και εδώ συνδέθηκαν οι κόμβοι του δικτύου με ευθείες γραμμές και υπολογίστηκε η ευκλείδεια απόστασή μεταξύ τους.

Το πρόβλημα που έπρεπε να αντιμετωπιστεί εδώ είναι ότι έπρεπε από τον κάθε κόμβο του δικτύου να δημιουργηθούν συνδέσεις προς όλους τους υπόλοιπους που απέχουν έως 500 μέτρα από αυτόν. Εφόσον δεν υπάρχει κάποια τέτοια λειτουργία στην Postgres, ούτε στο QGis ή το ArcGis, δημιουργήθηκε στο ArcGis ένας απλός αλγόριθμος σε Python μετά από εγκατάσταση της βιβλιοθήκης ArcPy. Μετά την εκτέλεση του αλγορίθμου έχουν δημιουργηθεί συνδέσεις από και προς όλους τους κόμβους του δικτύου. Αυτό σαφώς δεν είναι σωστό, συνεπώς έπρεπε να ορισθεί μια μέγιστη επιτρεπόμενη απόσταση βαδίσματος μεταξύ των κόμβων. Επιλέχθηκε λοιπόν η απόσταση των 500 μέτρων με μέση ταχύτητα βαδίσματος τα 100 m/min. Συνεπώς αφαιρέθηκαν οι ακμές που είχαν μήκος μεγαλύτερο από τα 500 μέτρα.

Κατόπιν οι πεζές διαδρομές ενώθηκαν με το δίκτυο μας και τους αποδόθηκαν η τιμή 0100 στο πεδίο ROUTE και η τιμή W στο πεδίο LINE. Το νέο επίπεδο που δημιουργήθηκε ονομάστηκε edges_walk. Τέλος έπρεπε να εκτελεσθεί πάλι η εντολή pgr_createtopology, ώστε να συνδεθούν και οι νέες ακμές με τους κόμβους του δικτύου.

3.4.2. Έλεγχος και διορθώσεις γράφου

Η βιβλιοθήκη PgRouting παρέχει κάποιες συναρτήσεις για εφαρμογή ελέγχων και διορθώσεων στο δίκτυο όπως αναφέρεται στο κεφάλαιο 4.3.3. Χρησιμοποιήθηκαν κάποιες από αυτές για να αξιολογηθεί ο γράφος που δημιουργήθηκε αλλά και για να εξαχθούν τα απαραίτητα στατιστικά στοιχεία.

Πιο συγκεκριμένα, αρχικά εφαρμόστηκε η συνάρτηση pgr_analyzegraph, που ελέγχει την τοπολογία του γράφου. Εντοπίζει τα αδιέξοδα (dead ends), τα απομονωμένα τμήματα δικτύου (isolated segments), τμήματα δηλαδή που και τα δύο άκρα είναι αδιέξοδα, τα τμήματα που έχουν γεωμετρία δακτυλίου (ring geometries) , ενδεχόμενα κενά κοντά σε αδιέξοδα και τις διασταυρώσεις ακμών.

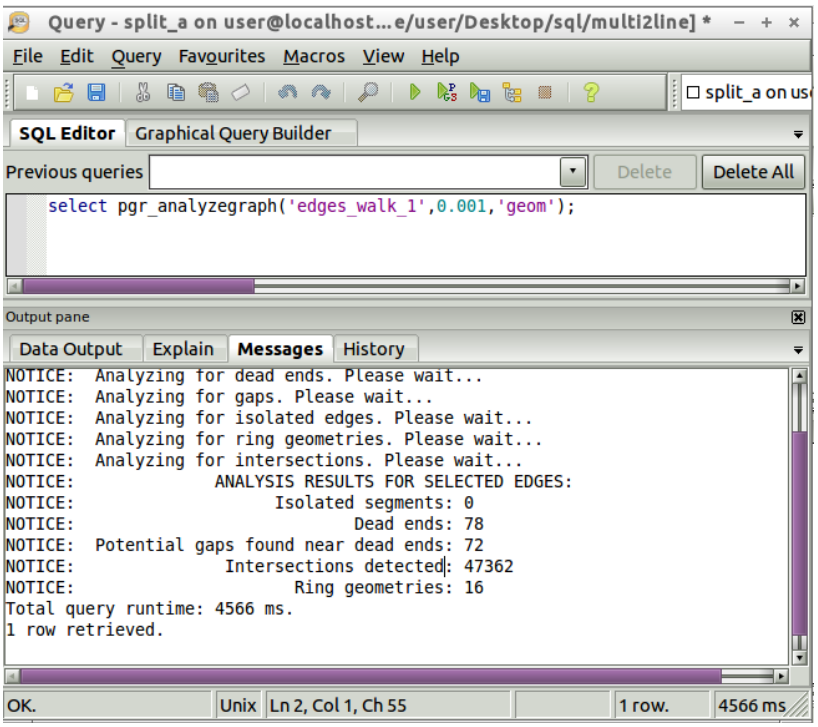
Για να εφαρμοστεί η συνάρτηση αυτή θα πρέπει ο πίνακας των ακμών να περιέχει τις στήλες SOURCE και TARGET συμπληρωμένες με το id των κόμβων. Για το λόγο αυτό θα πρέπει να έχει προηγηθεί η συνάρτηση pgr_createtopology. Οι παράμετροι εισόδου είναι οι εξής :

- Ο πίνακας ακμών που δηλώνεται με το όνομά του
- Η ανοχή ανοίγματος στις μη συνδεδεμένες ακμές σε μονάδες του προβολικού συστήματος που χρησιμοποιήθηκε (snapping tolerance)

- Η στήλη που περιέχει την πληροφορία της γεωμετρίας (geom)
- Η στήλη του πρωτεύοντος κλειδιού του πίνακα ακμών (id)
- Η στήλη SOURCE
- Η στήλη TARGET

Η συνάρτηση επιστρέφει το μήνυμα OK αφού ολοκληρωθεί η ανάλυση. Τότε θα έχουν συμπληρωθεί πλήρως οι στήλες cnt και chk του πίνακα των κόμβων, που αναφέρονται στον αριθμό των ακμών που αναφέρονται στον κάθε κόμβο και σε μία ένδειξη ότι η κορυφή μπορεί να έχει πρόβλημα αντίστοιχα.

Τα αποτελέσματα από την εφαρμογή του pgr_analyzegraph στον γράφο φαίνονται στην εικόνα 5.



Εικόνα 20 : Εφαρμογή συνάρτησης ελέγχου pgr_analyzegraph στο pgAdmin

Για να διορθωθούν τα σφάλματα αυτά, το PgRouting παρέχει τη συνάρτηση pgr_nodeNetwork η οποία έχει σχεδιαστεί για να διορθώνει την έλλειψη κόμβων στις διασταυρώσεις. Παρ' όλο που ο σκοπός του είναι η εισαγωγή κόμβων στις διασταυρώσεις τελικά διορθώνονται και άλλα προβλήματα όπως τα αδιέξοδα και τα κενά κοντά στο τέρμα των ακμών. Επειδή στην περίπτωση της εργασίας οι ακμές ορίζουν τη διαδρομή ενός μέσου από στάση σε στάση δεν μπορεί να αξιοποιηθεί η συνάρτηση αυτή, καθώς δεν πρέπει να εισαχθούν νέοι κόμβοι. Οι διορθώσεις στις προβληματικές περιοχές έγιναν εποπτικά στο QGIS.

Μία ακόμη συνάρτηση ελέγχου που εντοπίζει ζητήματα κατεύθυνσης είναι η pgr_analyzeOneway. Αναλύει τις ακμές ενός γράφου σύμφωνα με τους κανόνες κατεύθυνσης που έχουν τεθεί και επισημαίνει εκείνες που τους παραβιάζουν. Στα δεδομένα τύπου OSM υπάρχει μία στήλη, που συνήθως ονομάζεται oneway, που υποδεικνύει τη φορά της γραμμής ανάλογα με τον κωδικό της :

-1 : Μονής κατεύθυνσης, αντίθετης φοράς από τη φορά σχεδιασμού

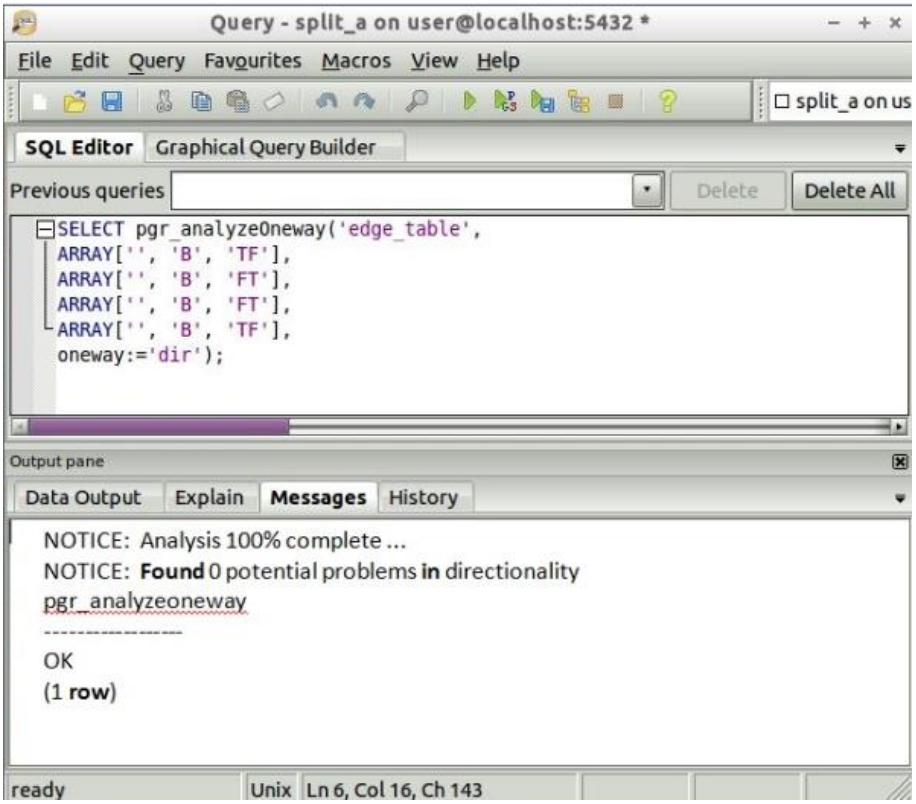
0 : Άγνωστης κατεύθυνσης

1 : Μονής κατεύθυνσης, ίδιας φοράς με τη φορά σχεδιασμού

2 : Διπλής κατεύθυνσης

3 : Εναλλασσόμενης κατεύθυνσης (φορά ανάλογα με την ώρα της ημέρας)

Στα δεδομένα που χρησιμοποιήθηκαν δημιουργήθηκε η στήλη oneway και όλοι οι κωδικοί πήραν την τιμή 1, αφού όλες οι ακμές είναι μονής κατεύθυνσης και η φορά ίδια με τη φορά σχεδιασμού. Αφού έτρεξε η συνάρτηση pgr_analyzeOneway επισημάνθηκαν πιθανά σφάλματα, εξήχθησαν οι προβληματικές ακμές στο QGIS και αφού έγιναν οι απαιτούμενες διορθώσεις εισήχθησαν και πάλι στο γράφο. Έπειτα εφαρμόσθηκε και πάλι η συνάρτηση, όπου δεν εντοπίστηκε κανένα σφάλμα (Εικόνα 21).

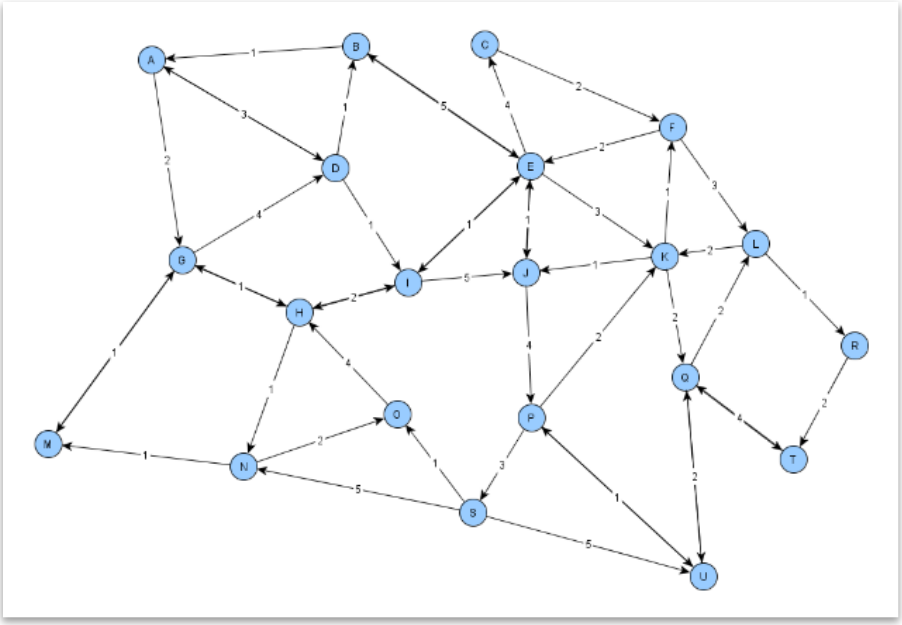


Εικόνα 21 : Εφαρμογή συνάρτησης ελέγχου pgr_analyzeoneway στο pgAdmin

3.4.3. Απόδοση κόστους στις ακμές

Ο γράφος ενός οδικού δικτύου ορίζεται ως το σύνολο των σημείων και των ακμών που τα συνδέουν, καθώς και το κόστος που αποδίδεται στις ακμές και αντιπροσωπεύει το κόστος (χιλιομετρικό, χρονικό, οικονομικό κ.λπ.) που απαιτείται για να μεταβεί κάποιος από κόμβο σε κόμβο (Εικόνα 22).

Στο pgRouting υπάρχουν δύο είδη κόστους. Το απλό κόστος που αναφέρεται στο κόστος διαδρομής από τον κόμβο αφετηρίας στον κόμβο προορισμού και το αντίστροφο κόστος το οποίο υποδηλώνει το κόστος πορείας αντίθετης φοράς από τη φορά σχεδίασης. Επειδή κάθε ακμή του γράφου είναι μονής κατεύθυνσης μπορεί να αγνοηθεί το αντίστροφο κόστος και στους υπολογισμούς ο γράφος να θεωρηθεί κατευθυνόμενος.



Εικόνα 22 : Παράδειγμα γράφου

Σε αυτό το στάδιο πρέπει να υπολογιστεί το κόστος κάθε ακμής. Αφού το ενδιαφέρον εστιάζεται στον ελάχιστο χρόνο διαδρομής, το κόστος αρχικά θα είναι ο χρόνος προσπέλασης κάθε ακμής. Η διαδικασία αυτή έγινε στο περιβάλλον PgAdmin με χρήση της Postgres.

Για να αποδοθεί χρονικό κόστος σε κάθε γραμμή θα πρέπει να είναι γνωστή η ταχύτητα κάθε μέσου. Από τον πίνακα 3 εξάγεται το συνολικό μήκος που καλύπτει κάθε ένα και τη

διάρκεια κάθε διαδρομής από την αφετηρία μέχρι το τέρμα και με την εξίσωση $u_\mu = \frac{x}{t}$, υπολογίζεται η μέση ταχύτητα σε μέτρα/λεπτό. Τα αποτελέσματα είναι τα εξής :

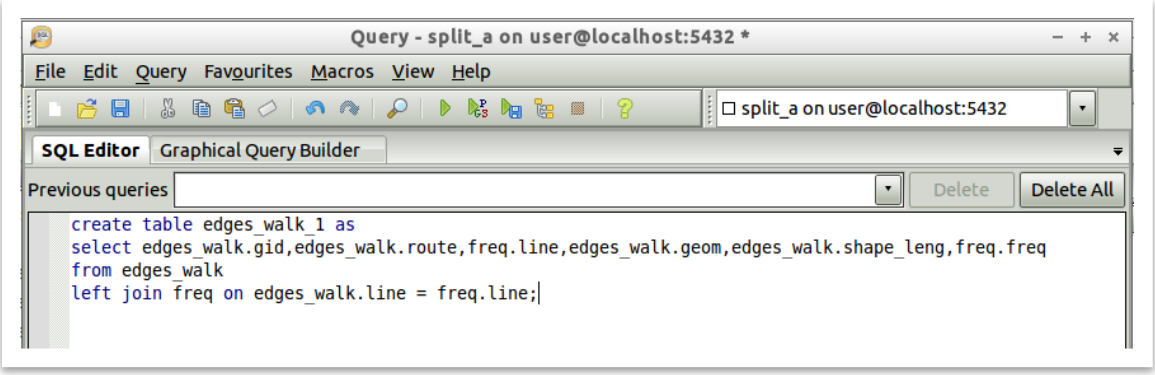
- Λεωφορεία –τρόλεϊ : 257 m/min
- Μετρό : 594 m/ min
- Τραμ : 325 m/min

Αφού υπολογιστεί και το μήκος κάθε ακμής με την αντίστοιχη συνάρτη ST_length που προσφέρει η επέκταση PostGis της Postgres, εφαρμόζεται και πάλι η εξίσωση $t = \frac{x}{u_\mu}$ και υπολογίζεται ο χρόνος που χρειάζεται για να διασχιστεί κάθε ακμή.

Το γεγονός ότι πρόκειται για δρομολόγηση σε δημόσια μέσα μεταφοράς, εισάγει ένα επιπλέον χρονικό κόστος στα δεδομένα μας, το κόστος μετεπιβίβασης. Το κόστος αυτό, δηλαδή ο χρόνος αναμονής για μετάβαση σε επόμενο μέσο στην ίδια διαδρομή, εξαρτάται από τη συχνότητα του εν λόγω μέσου. Συνεπώς δημιουργήθηκε μία νέα στήλη με το όνομα freq η οποία θα συμπληρωθεί με τη συχνότητα κάθε γραμμής.

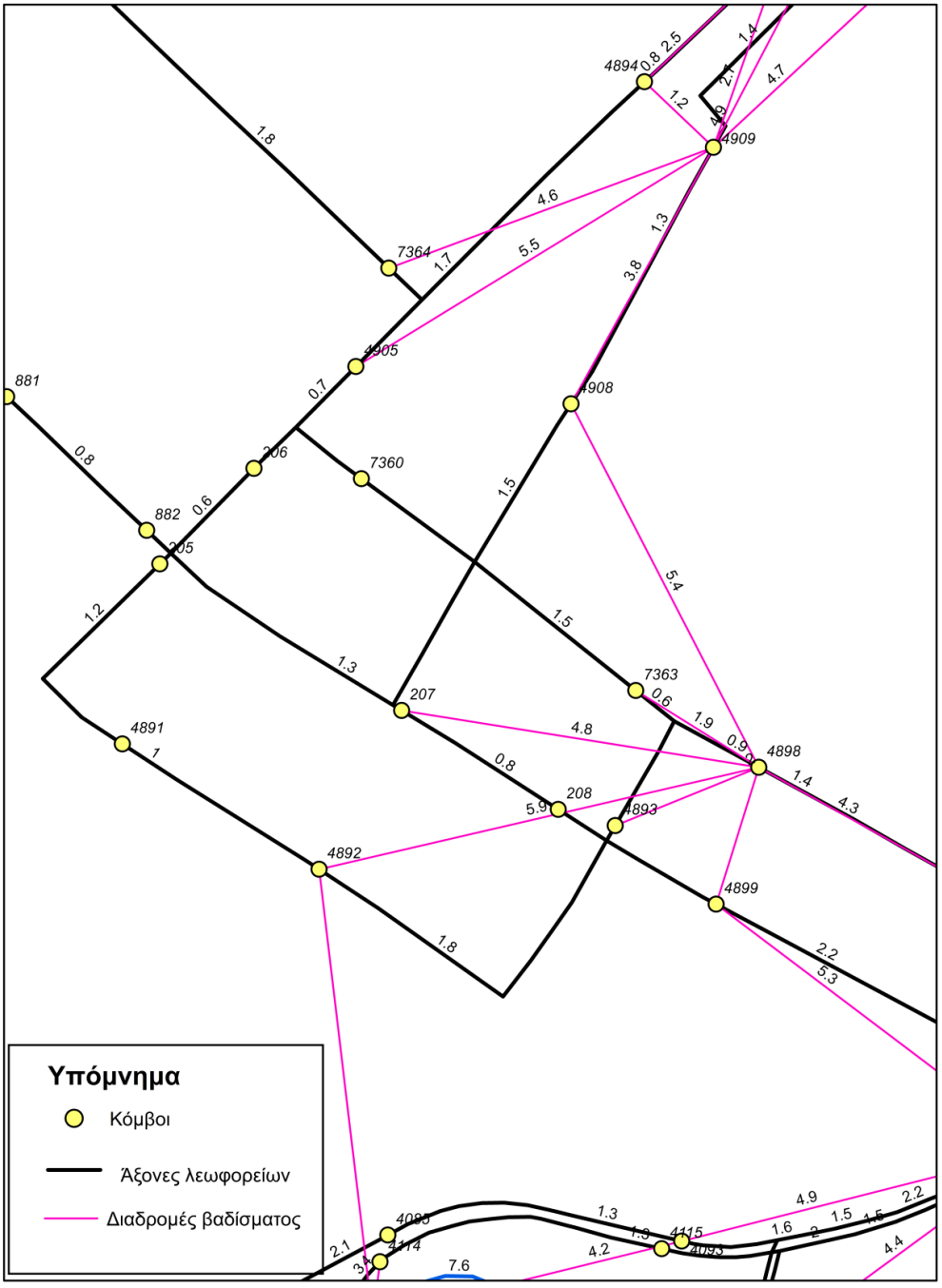
Όπως είναι φυσικό και όπως φαίνεται και στους Πίνακες 2 και 3, οι συχνότητες αλλάζουν ανάλογα με την ώρα αλλά και την ημέρα. Για παράδειγμα στην έναρξη αλλά και προς τη λήξη του ωραρίου αραιώνουν τα δρομολόγια, ενώ τις καθημερινές πραγματοποιούνται περισσότερα δρομολόγια απ' ότι τα Σαββατοκύριακα. Για την παρούσα εργασία χρησιμοποιήθηκε η τυπική (καθημερινή) συχνότητα.

Εισήχθη ο πίνακας συχνοτήτων στη βάση δεδομένων και με βάση τον αριθμό της γραμμής έγινε ένωση (join) με τον πίνακα των ακμών. Ο κώδικας φαίνεται στην Εικόνα 23.



Εικόνα 23 : Κώδικας για πραγματοποίηση ένωσης πινάκων με βάση μία στήλη

Στην Εικόνα 24 φαίνεται ένα απόσπασμα του γράφου που δημιουργήθηκε, στο οποίο φαίνεται ο κωδικός (id) κάθε στάσης και το χρονικό κόστος διάσχισης κάθε ακμής σε λεπτά.



Εικόνα 24 : Απόσπασμα του γράφου

3.5. Ανάπτυξη πρωτότυπου αλγορίθμου

Ο κορμός της εφαρμογής δεν είναι άλλος από τον υπολογισμό των σημείων του οδικού δικτύου στα οποία μπορεί να φτάσει ένας επιβάτης μέσα στο ζητούμενο χρονικό διάστημα ξεκινώντας από το αρχικό σημείο. Για τον υπολογισμό αυτό καταλληλότερος είναι ο αλγόριθμος συντομότερων μονοπατιών Dijkstra. Βέβαια, για τις ανάγκες της εφαρμογής χρειάστηκαν κάποιες βελτιώσεις μετά την εφαρμογή του αλγορίθμου. Σε αυτήν την παράγραφο, περιγράφεται η εφαρμογή του Dijksta και αυτές οι απαιτούμενες βελτιώσεις.

3.5.1. Θέματα προς επίλυση

Όπως έχει αναφερθεί και στο κεφάλαιο 2.2.1. ο αλγόριθμος Dijkstra υπολογίζει τα συντομότερα μονοπάτια. Για τον υπολογισμό ισοχρονικών καμπυλών όμως, το ενδιαφέρον εστιάζεται στους συντομότερους χρόνους διαδρομής, δηλαδή ο πίνακας D μετά την εκτέλεση του αλγορίθμου. Το πρόβλημα αυτό ξεπερνιέται εύκολα αν ως κόστος οριστεί ο χρόνος διαδρομής και όχι η χιλιομετρική απόσταση.

Επιπλέον στην εύρεση της χρονικά συντομότερης διαδρομής με μέσα μαζικής μεταφοράς, πρέπει να ληφθεί υπ’ όψιν ότι η μετεπιβίβαση επιφέρει ένα επιπλέον χρονικό κόστος. Το κόστος αυτό θα πρέπει να υπολογιστεί στην ανάλυση. Επίσης είναι αναγκαίο, το σύστημα να εισάγει μετεπιβίβαση μόνον όταν μειώνεται το συνολικό κόστος διαδρομής ή όταν δεν υπάρχει εναλλακτική.

Εφαρμόζοντας τον Dijkstra ανάμεσα σε ζεύγη κόμβων του γράφου, έγινε φανερό ότι υπολογίζονται μεν τα συντομότερα μονοπάτια, εισάγονται δε άσκοπες μετεπιβιβάσεις, γεγονός που καθιστά ανέφικτο τον σωστό υπολογισμό του χρονικού κόστους κάθε κόμβου. Αυτό οφείλεται στο γεγονός ότι στο γράφο του δικτύου ανάμεσα σε δύο κόμβους μπορεί να περιλαμβάνονται περισσότερες από μία ακμές με το ίδιο κόστος. Δηλαδή περισσότερες από μία γραμμές συνδέουν τις δύο στάσεις μέσω της ίδιας διαδρομής, κάτι που είναι αρκετά συνηθισμένο σε λεωφορειακές γραμμές. Το συγκεκριμένο πρόβλημα λοιπόν προκύπτει διότι στα λεωφορεία και τα τρόλεϊ έχει δοθεί μία ενιαία μέση ταχύτητα, συνεπώς ο αλγόριθμος αντιλαμβάνεται τις ακμές ως ίσου κόστους και δεν λαμβάνει υπ’ όψιν το κόστος μετεπιβίβασης. Αλλά και κατόπιν της εφαρμογής του αλγορίθμου δεν μπορεί να γίνει ο σωστός υπολογισμός αφού δεν είναι γνωστό ποιες είναι ορθές και ποιές άσκοπες ή τυχαίες μετεπιβιβάσεις. Στον Πίνακα 7 παρουσιάζεται το αποτέλεσμα της εφαρμογής του αλγορίθμου Dijkstra ανάμεσα στους κόμβους με κωδικούς 1161 και 565. Στην τελευταία στήλη με όνομα line_no φαίνεται η αλληλουχία των λεωφορείων που χρησιμοποιήθηκε για

την προσπέλαση κάθε ακμής. Είναι δεδομένο ότι στη δωδέκατη στάση γίνεται μία άσκοπη μετεπιβίβαση από τη γραμμή Α3 στην Β3 αφού στη δέκατη τέταρτη επιστρέφει στην αρχική γραμμή Α3.

Συνεπώς πρέπει να δομηθεί ένα σύστημα διόρθωσης των αποτελεσμάτων του Dijkstra και υπολογισμού του κόστους κάθε κόμβου.

Πίνακας 7 : Αποτέλεσμα δρομολόγησης ανάμεσα σε δύο κόμβους με τον αλγόριθμο Dijkstra

	seq integer	edge integer	edge_geor geometry(	source integer	target integer	coast double pre	line_no character v
5	5	1286	0105000020	1165	1166	20	B3
6	6	1287	0105000020	1166	1167	20	B3
7	7	1289	0105000020	1167	1147	20	B3
8	8	1288	0105000020	1147	1148	20	B3
9	9	1290	0105000020	1148	1168	20	B3
10	10	1291	0105000020	1168	620	20	B3
11	11	1292	0105000020	620	621	20	B3
12	12	504	0105000020	621	135	20	A3
13	13	1294	0105000020	135	136	20	B3
14	14	506	0105000020	136	622	20	A3
15	15	507	0105000020	622	623	20	A3
16	16	508	0105000020	623	626	20	A3
17	17	509	0105000020	626	627	20	A3
18	18	510	0105000020	627	628	20	A3
19	19	511	0105000020	628	629	20	A3
20	20	512	0105000020	629	630	20	A3
21	21	1302	0105000020	630	599	20	B3

3.5.2. Λογισμικό και Προγράμματα που χρησιμοποιήθηκαν

Όπως αναφέρθηκε στο προηγούμενο εδάφιο πρέπει να δομηθεί ένας αλγόριθμος ο οποίος θα δέχεται τα αποτελέσματα από την εκτέλεση του Dijkstra, θα τα ελέγχει και θα τα διορθώνει όπου χρειάζεται. Επιπλέον για την εξοικονόμηση χρόνου, θα πρέπει ο Dijkstra να καλείται αυτόματα για κάθε ζεύγος κόμβων του δικτύου, αφού οριστεί ο κόμβος αφετηρίας.

Με δεδομένο ότι ο όγκος των πληροφοριών είναι αρκετά μεγάλος και η επεξεργασία τους σημαντικά πολύπλοκη επιλέχθηκε να δομηθεί ο αλγόριθμος στη γλώσσα προγραμματισμού Python.

Η Python είναι μια υψηλού επιπέδου γλώσσα προγραμματισμού η οποία δημιουργήθηκε από τον Ολλανδό Γκβίντο βαν Ρόσσουμ (Guido van Rossum) το 1990 (Ο' Reilly Radar, 2006). Ο κύριος στόχος της είναι η ομοιότητα του κώδικά της με την καθημερινή γλώσσα και η ευκολία χρήσης της και το συντακτικό της επιτρέπει στους προγραμματιστές να εκφράσουν έννοιες σε λιγότερες γραμμές κώδικα απ' ότι θα ήταν δυνατόν σε γλώσσες όπως η C++ ή η Java. Διακρίνεται λόγω του

ότι έχει πολλές βιβλιοθήκες που διευκολύνουν ιδιαίτερα αρκετές συνηθισμένες εργασίες όπως επεξεργασία strings (συνήθησεις εκφράσεις, Unicode, υπολογισμός διαφορών μεταξύ αρχείων), διαδικτυακά πρωτόκολλα, (HTTP, FTP, SMTP, XML-RPC, POP, IMAP, (έλεγχος μονάδων, καταγραφή, δημιουργία προφίλ, ανάλυση κώδικα Python) και διεπαφές λειτουργικού συστήματος (κλήσεις συστήματος, συστήματα αρχείων, υποδοχές TCP / IP) (www.python.com).

Η Python αναπτύσσεται ως ανοιχτό λογισμικό (open source) και η διαχείρισή της γίνεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation. Ο κώδικας διανέμεται με την άδεια Python Software Foundation License η οποία είναι συμβατή με την GPL.

Η γλώσσα αυτή χρησιμοποιεί μεταγλωττιστή (compiler) για την δημιουργία του εκτελέσιμου κώδικα και σχετίζεται με τις γλώσσες προγραμματισμού Tcl, Perl, Scheme, Java και Ruby, καθώς και με την ABC η οποία υπήρξε η αρχική πηγή έμπνευσης για τη δημιουργία της. Ένα ιδιαίτερο χαρακτηριστικό της είναι η χρήση κενών διαστημάτων (whitespace) για τον διαχωρισμό των συντακτικών δομών που προγράμματος, σε αντίθεση με την πρακτική σε άλλες γλώσσες όπου για τον ίδιο σκοπό χρησιμοποιούνται ειδικά σύμβολα (πχ αγκύλες)[URL2]. Αυτό, σε συνδυασμό με το ότι χρησιμοποιεί πλήρεις αγγλικές λέξεις στη θέση συμβόλων, καθιστούν τον κώδικα της Python ευανάγνωστο από όσους έχουν βασική γνώση των αγγλικών (Εικόνα 25).

```
age = 21
if age >= 18:
    print('You vote')
else:
    print('You dont vote')
```

Εικόνα 25 : Παράδειγμα κώδικα Python

Ένα πρόγραμμα Python μπορεί να αναπτυχθεί σε οποιονδήποτε text editor και το αρχείο να σωθεί με την κατάληξη .py. Ύστερα καλείται το αρχείο από τη γραμμή εντολών, όπου και αρχικοποιούνται οι τιμές που έχουν δηλωθεί στο πρόγραμμα (Εικόνα 26). Εννοείται πως αρχικά πρέπει να έχουν γίνει οι συνδέσεις της γραμμής εντολών με το φάκελο που περιέχει το πρόγραμμα και να γνωστοποιηθεί ότι πρόκειται για python εντολές. Υπάρχει πληθώρα επιλογών που παρέχει η γραμμή εντολών, τα οποία θα πρέπει κανείς να μελετήσει για να δημιουργήσει ένα λειτουργικό πρόγραμμα χωρίς σφάλματα ανάγνωσης ή αναγνώρισης αρχείων και μεταβλητών.

```
$ python test.py arg1 arg2 arg3
```

Εικόνα 26 : Κάλεσμα προγράμματος Python και αρχικοποίηση μεταβλητών από τη γραμμή εντολών

Όπως προαναφέρθηκε η συγγραφή ενός Python αλγορίθμου είναι δυνατόν να γίνει σε οποιοδήποτε πρόγραμμα επεξεργασίας κειμένου (πχ Notepad). Εάν επιλεχθεί ένα απλό

πρόγραμμα επεξεργασίας κειμένου δεν θα υπάρχει η δυνατότητα για υποστήριξη κατά τη συγγραφή του κώδικα και πρόληψη σφαλμάτων. Παρέχονται πολλοί editors ιδικά για την συγγραφή προγραμμάτων Python κάποιοι από τους οποίους είναι το Eclipse, το GEdit, το Geany, το Atom κα.

Στην παρούσα εργασία χρησιμοποιήθηκε το Atom, κυρίως λόγω των δυνατοτήτων που προσφέρει. Πρόκειται για έναν επεξεργαστή κειμένου με ελεύθερο και ανοικτό λογισμικό για macOS, Linux και Microsoft Windows με επεκτάσεις γραμμένες Node.js και ενσωματωμένο Git Control, που αναπτύχθηκε από το GitHub. Μπορεί να χρησιμοποιηθεί ως ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) και θεωρείται από τους προγραμματιστές «ο επεξεργαστής κειμένου του 21^{ου} αιώνα».

Τα σημαντικότερα πλεονεκτήματά του είναι τα εξής :

- Είναι απλός και λιτός επεξεργαστής με πλήρεις συνδέσεις κλειδιών (key-bindings)
- Αρίθμηση των γραμμών
- Αυτόματη εισαγωγή των σωστών κενών διαστημάτων που απαιτούνται από την Python
- Παρέχει επισήμανση των συντακτικών λέξεων - κλειδιά και αυτόματη συμπλήρωση λέξεων ή φράσεων (για μεταβλητές, συναρτήσεις κλπ).
- Ταυτόχρονη αναζήτηση σε πολλαπλά αρχεία και γραμμές κώδικα με δυνατότητα επιλογής και αντικατάστασης της επιθυμητής λέξης ή λέξεων.
- Προβολή κλάσεων και συναρτήσεων για γρήγορη περιήγηση σε αρχεία μεγάλου μεγέθους.
- Υψηλές λειτουργικές αποδόσεις και αξιοπιστία
- Τέλος παρέχεται μια ιδιαίτερα απλή διεπαφή για ανεύρεση, εγκατάσταση και ενημέρωση επεκτάσεων (plugins), ενώ υπάρχουν περίπου 3000 πακέτα έτοιμα για την επέκταση του Atom.

```
35          if self.path_cost[target] == Dijkstra.INFINITY:
36              return (None, None) # koszt = None, ścieżka = None
37
38          path = []
39
40          last_node = None
41          prev_edge = self.predecessor_edge[target]
42          while prev_edge:
43              if prev_edge.source() == target or prev_edge.source() == last_node:
44                  break
45              path.insert(0,prev_edge) # push front prev_edge
46              last_node = prev_edge.source()
47              prev_edge = self.predecessor_edge[prev_edge.source()] # iterowanie wstecz ścieżki
48
49          return (self.path_cost[target], path) # koszt, ścieżka
50
```

Εικόνα 27 : Απόσπασμα κώδικα Python γραμμένο στο Atom. Φαίνεται η υπογράμμιση των λέξεων – κλειδιών, τα κενά διαστήματα μεταξύ των δομών και η αρίθμηση των γραμμών.

3.5.3. Νέος αλγόριθμος προσδιορισμού χρόνου βέλτιστης διαδρομής

Ο νέος αλγόριθμος που δομήθηκε ακολουθεί την εξής μεθοδολογία. Αρχικά εφαρμόζεται ο αλγόριθμος Dijkstra από τον κόμβο αφετηρίας προς όλους τους υπόλοιπους κόμβους του δικτύου. Αυτό καθίσταται δυνατό με μία δομή επανάληψης, που διαβάζει τους κόμβους του γράφου έναν έναν, ορίζει κάθε φορά τον τρέχοντα κόμβο ως κόμβο προορισμού και καλεί τον Dijkstra. Ύστερα προχωράει στον επόμενο κόμβο και επαναλαμβάνεται η διαδικασία.

Οι διαδρομές που δημιουργούνται εξάγονται σε ένα συνολικό αρχείο. Η κάθε διαδρομή αποτελεί μία λίστα που περιλαμβάνει την αλληλουχία των ακμών που διασχίσθηκαν, το κωδικό της γραμμής που χρησιμοποιήθηκε, τον κόμβο από τον οποίο ξεκινάει η ακμή (source) και τον κόμβο στον οποίο καταλήγει (target).

Σε αυτό το σημείο εισέρχεται η κύρια παρέμβαση του προγράμματος που δομήθηκε. Για κάθε δημιουργημένη διαδρομή εκτελείται ο έλεγχος των μετεπιβιβάσεων. Ουσιαστικά ελέγχεται η λίστα της διαδρομής σε σχέση με τη λίστα του γράφου. Δηλαδή όπου το σύστημα εντοπίσει αλλαγή στον κωδικό της γραμμής ελέγχει αν στη λίστα του γράφου υπάρχει η συγκεκριμένη γεωμετρία ακμής με κωδικό γραμμής ίδιον με της προηγούμενης. Αν ο έλεγχος επιστρέψει TRUE ο κωδικός γραμμής αλλάζει και ακυρώνεται η μετεπιβίβαση που είχε εισαχθεί τυχαία από τον Dijktra. Είναι ευνόητο ότι αν ο έλεγχος επιστρέψει FALSE το πρόγραμμα δεν προβαίνει σε καμία αλλαγή και προχωρά στον έλεγχο της επόμενης ακμής. Προκειμένου να αυξηθεί η αποτελεσματικότητα του νέου αλγορίθμου, δομήθηκε αναδρομικά. Δηλαδή όταν τελειώνει ο έλεγχος της λίστας, ο αλγόριθμος καλεί τον εαυτό του και ξανατρέχει για την επόμενη λίστα. Αυτό είναι κάτι ιδιαίτερα σημαντικό καθώς δεν θα ήταν δυνατό, αλλά και ούτε αποδοτικό να καλεστεί ο αλγόριθμος χειροκίνητα χιλιάδες φορές, όσες είναι οι παραχθείσες διαδρομές.

Στο προηγούμενο στάδιο παραλείφθηκε ένα καίριο βήμα του αλγορίθμου. Όταν τερματιστεί ο έλεγχος και η διόρθωση μίας διαδρομής και πριν το σύστημα προχωρήσει στην επόμενη, υπολογίζεται το συνολικό κόστος της και αποδίδεται στον κόμβο προορισμού. Η διαδικασία αυτή εκτελείται με μία ακόμη δομή επανάληψης. Διαβάζονται οι κωδικοί των γραμμών και προστίθενται διαδοχικά τα κόστη διάσχισης των ακμών. Επιπλέον κάθε φορά που εντοπίζεται μετεπιβίβαση προστίθεται και αυτό το επιπρόσθετο κόστος στο συνολικό. Ο χρόνος μετεπιβίβασης έχει ορισθεί ως ο μισός της συχνότητας κάθε γραμμής. Τελικά σε ένα νέο αρχείο γράφεται ο κωδικός του τελευταίου κόμβου της διαδρομής (κόμβος προορισμού) και δίπλα το συνολικό του κόστος. Το αρχείο αυτό εξάγεται σε txt και είναι έτοιμο προς περαιτέρω επεξεργασία στη βάση δεδομένων και τα προγράμματα γεωγραφικών συστημάτων πληροφοριών.

Η διαδικασία και τα βήματα υλοποίησης του εν λόγω αλγορίθμου επεξηγούνται παρακάτω.

Ο αλγόριθμος Dijksta διατίθεται σε Python στο Github (https://gist.github.com/econchick). Συνεπώς το μόνο που έπρεπε να υλοποιηθεί είναι να δηλώσουμε δεδομένα της εφαρμογής και να αρχικοποιηθούν οι τιμές τους.

Στη συνέχεια δημιουργήθηκαν κάποιες κλάσεις και συναρτήσεις για την αυτόματη εφαρμογή του Dijkstra από το κόμβο αφετηρίας προς όλους τους υπόλοιπους κόμβους του δικτύου αλλά και τη διόρθωση και την εξαγωγή των αποτελεσμάτων. Στη συνέχεια περιγράφεται πιο αναλυτικά η δομή του προγράμματος.

- Connector_class()

  Είναι η κλάση που δημιουργεί το μονοπάτι σύνδεσης με τη βάση δεδομένων μας. Ύστερα επιλέγονται τα αρχεία που δομούν το γράφο και με τη βοήθεια ενός κέρσορα (cursor) γράφονται σε λίστες που είναι η βασική δομή δεδομένων της Python. Έτσι μπορούμε να δηλώσουμε και να επεξεργαστούμε τα δεδομένα μας.

- Class_edge()

  Είναι η κλάση που δημιουργείται και χαρακτηρίζει τις ακμές του δικτύου. Περιλαμβάνει τις εξής συναρτήσεις:
  - Def__init__(self, start, end, weight, freq, line, geom)
    Ουσιαστικά ορίζονται οι παράμετροι αρχικοποίησης, που δεν είναι άλλες από την ίδια τη γραμμή, τον κωδικό του κόμβου έναρξης, τον κωδικό του κόμβου λήξης, το κόστος, τη συχνότητα, τον κωδικό της γραμμής και την πληροφορία της γεωμετρίας (Εικόνα 28).
  - Def target, def source, def weight, def freq, def line, def weight
    Με αυτές τις συναρτήσεις ορίζονται οι θέσεις και οι τιμές των παραμέτρων αρχικοποίησης από τα δεδομένα που εισάγονται από τη βάση δεδομένων.

- Class_VertexNotExistsError(Exception)

  Με αυτή την κλάση ο αλγόριθμος ελέγχει αν ο κόμβος που ορίζει ο χρήστης είναι μέσα στη λίστα των κόμβων. Εάν δεν είναι εμφανίζεται το σχετικό σφάλμα (error).

- Class_Graph

  Σε αυτή την κλάση δημιουργείται ο γράφος με τρόπο ώστε να τον αναγνωρίζει η python. Περιέχει αρκετές συναρτήσεις, οι κυριότερες από τις οποίες είναι:
  - def number_of_vertices(self)
    Επιστρέφει τον αριθμό των αντικειμένων που περιέχει η λίστα των κόμβων.
  - def number_of_edges(self)

Επιστρέφει τον αριθμό των αντικειμένων που περιέχει η λίστα των ακμών.

- def add_vertex(self, node)
  Η συνάρτηση προσθέτει έναν έναν του κόμβους στον γράφο

- def add_edge(self, edge)
  Η συνάρτηση προσθέτει μία μία τις ακμές στον γράφο

- def add_undirected_edge(self, v1, v2, weight)
  Η συνάρτηση εισάγει μία μη κατευθυνόμενη ακμή ανάμεσα στους κόμβους v1, v2 με βάρος weight. Καλείται εντός μίας δομής επανάληψης που σαρώνει τον πίνακα των ακμών και διαβάζει τα κελιά source και target.

- def direct_edge(self, v1, v2)
  Η μέθοδος αυτή αναζητά την ακμή από τον κόμβο v1 στον κόμβο v2 και της αποδίδει την πληροφορία της κατεύθυνσης. Η ακμή αναζητείται πλέον στον γράφο. Αν δεν βρεθεί η ακμή v1 – v2 προχωράει στον επόμενο συνδυασμό, μέχρι να εξαντλήσει κάθε πιθανή ακμή του γράφου (Εικόνα 28).

```
if v1 in self.graph:
    for edge in self.graph[v1]:
        if edge.target() == v2:
            return edge
```

Εικόνα 28 : Έλεγχος ταύτισης κωδικού ακμής σε Python

- def get_neighbours(self, v)
  Βρίσκει τους γειτονικούς κόμβους του γράφου, δηλαδή εκείνους που συνδέονται με μία ακμή. Για κάθε κόμβο v του γράφου που αποτελεί κόμβο source σε κάποια ακμή επιστέφει όλους τους αντίστοιχους που αποτελούν κόμβους target στην ίδια ακμή.

- def weight_between(self, v1, v2)
  Η μέθοδος αυτή επιστρέφει το ελάχιστος κόστος μεταξύ δύο κορυφών του γράφου ή το ορίζει ως άπειρο.

- def find_min_path(self, v1, v2)

Η συγκεκριμένη συνάρτηση βρίσκει την ελάχιστη διαδρομή μεταξύ δύο κορυφών v1 – v2 του γράφου. Ουσιαστικά καλεί τον αλγόριθμο του Dijkstra και εισάγει τις παραμέτρους του γράφου και των δύο κόμβων. Εάν δεν βρεθεί μονοπάτι μεταξύ των δύο κόμβων επιστρέφει το μήνυμα «Δεν υπάρχει σύνδεση μεταξύ των κορυφών». Εάν υπολογισθεί το συντομότερο μονοπάτι επιστέφει το κόστος και τη διαδρομή. Σε αυτό το σημείο το κόστος δεν θα είναι σωστό καθώς δεν έχουν εφαρμοστεί οι απαραίτητες διορθώσεις. Το μονοπάτι είναι σωστό γεωμετρικά όχι όμως οι γραμμές των μέσων μαζικής μεταφοράς.

- def get_subgraph(self, nodes)
  Η μέθοδος αυτή επιστέφει έναν υπογράφο με τους κόμβους που προσπελάστηκαν. Επιστρέφει ένα νέο αντικείμενο με τους συγκεκριμένους κόμβους και τις αντίστοιχες ακμές. Το νέο αντικείμενο αυτό θα υποστεί στη συνέχεια τις απαραίτητες διορθώσεις για τον ορθό υπολογισμό του κόστους.

```
10    def __init__(self, start, end, weight, freq, line, geom):
11        self.start = start
12        self.end = end
13        self._weight = weight
14        self._line = line
15        self._freq = freq
16        self._geom = geom
17
```

Εικόνα 29 : Αρχικοποίηση των στοιχείων του γράφου

- Isochroner_class()
  Είναι ο βασικός κορμός του προγράμματος και περιλαμβάνει τις εξής συναρτήσεις :

  - Def_assign_geoms_to_lines
    Είναι μια συνάρτηση που καλεί τις στήλες line, source και target κάθε διαδρομής και επιστρέφει το αρχείο geom_to_lines.

  - Def_get_isochrone_paths(self, starting_vertex)

Η συνάρτηση αυτή έχει ως είσοδο τον κόμβο αφετηρίας και περιλαμβάνει μία δομή επανάληψης η οποία διατρέχει όλους τους κόμβους του γράφου. Στον κάθε κόμβο ελέγχει αρχικά αν είναι διάφορος του κόμβου αφετηρίας. Αν ο έλεγχος επιστρέψει TRUE, εκτελείται η συνάρτηση find_min_path και καλείται η κλάση διόρθωσης RouteCorrector.

- class RouteCorrector()

  Η κλάση αυτή επεξεργάζεται τα αποτελέσματα των συναρτήσεων find_min_path και get_subgraph και επιστρέφει τις διορθωμένες διαδρομές. Οι συναρτήσεις που τη συντελούν είναι οι εξής :

  - Def_correct_lines (self, route, boarded_line, remaining_route, counter)

    Η συνάρτηση αυτή δέχεται ως παραμέτρους τη διαδρομή που έχει εξαχθεί από το get_subgraph, τη γραμμή επιβίβασης, την επόμενη γραμμή και έναν μετρητή που αρχικά έχει τιμή 0. Με τη βοήθεια του μετρητή συγκρίνει κάθε γραμμή με την προηγούμενή της. Αν δεν είναι ίδιες, αν δηλαδή υπάρχει μετεπιβίβαση, καλεί τη συνάρτηση is_line_in_edge που ελέγχει αν η είναι άσκοπη. Αν συμβαίνει αυτό, διορθώνει την τρέχουσα γραμμή και την κάνει ίδια με την προηγούμενή της. Στη συνέχεια ο μετρητής αυξάνεται κατά ένα και έτσι τρέχουσα γίνεται η επόμενη και επαναλαμβάνεται ο έλεγχος.

  - Def_is_line_in_edge (self, line, source, target)

    Όπως φαίνεται, η συνάρτηση αυτή καλείται μόνο όταν υπάρχει διαφοροποίηση μιας γραμμής από την προηγούμενή της. Έχει ως τιμές εισόδου την προηγούμενη γραμμή και τους κόμβους αφετηρίας και προορισμού της τρέχουσας. Ανατρέχει στον αρχικό πίνακα των ακμών και ψάχνει αν υπάρχει ακμή με τα στοιχεία αυτά. Δηλαδή τον αριθμό γραμμής της προηγούμενης και τη διαδρομή της τρέχουσας. Αν συμβαίνει αυτό, η μετεπιβίβαση κρίνεται άσκοπη και διορθώνεται από τη συνάρτηση Def_correct_lines.

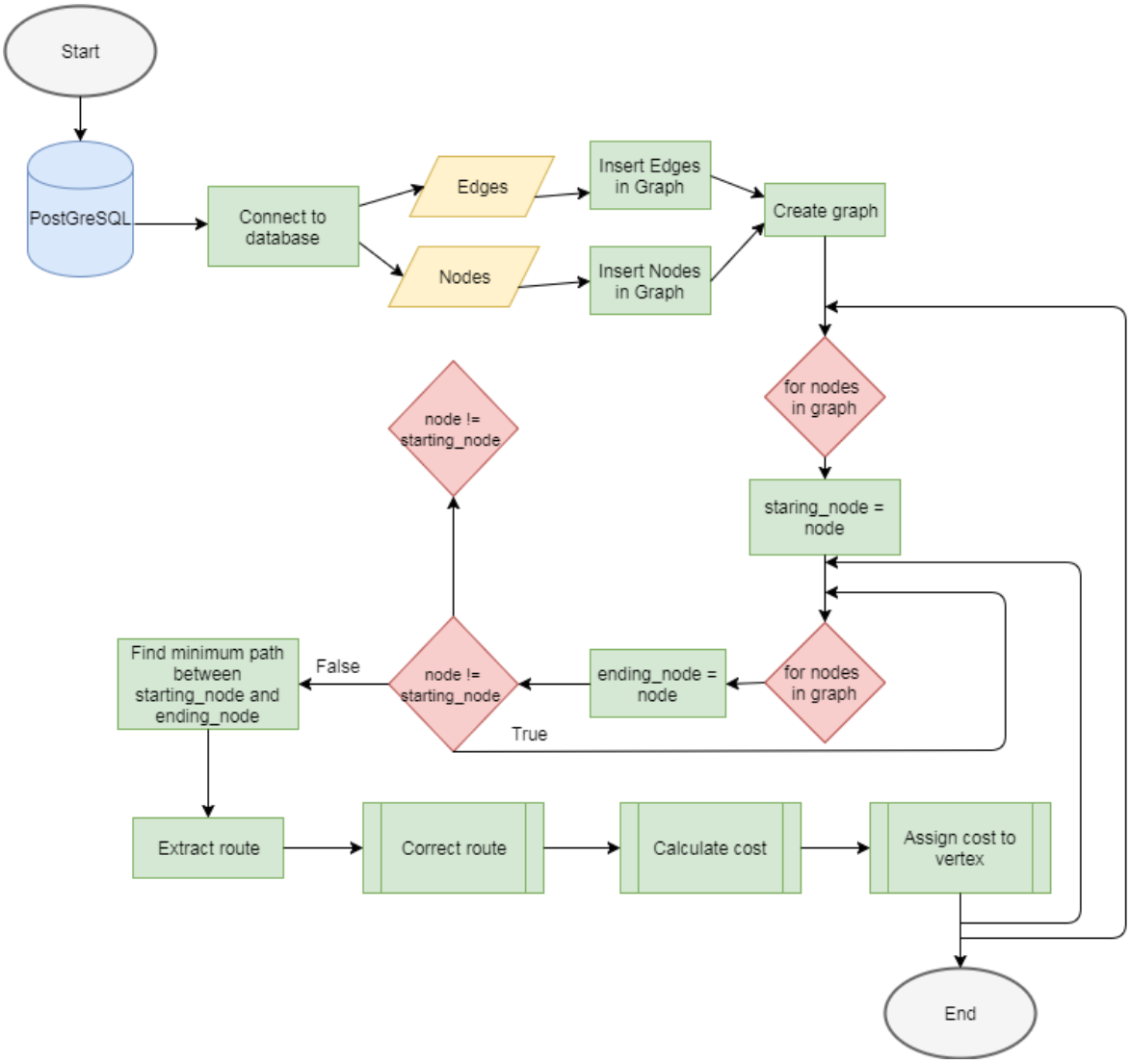
3.5.4. Υπολογισμός χρονικού κόστους κόμβων

Αφού έχουν διορθωθεί οι διαδρομές και δεν υπάρχουν άσκοπες μετεπιβιβάσεις, πρέπει να υπολογιστεί το τελικό κόστος κάθε κόμβου. Για το σκοπό αυτό δημιουργήθηκε η συνάρτηση def_calculate_path_weight στην κλάση Isochroner. Όταν καλείται η εν λόγω συνάρτηση σαρώνονται όλες οι εγγραφές της λίστας της διαδρομής και προστίθενται διαδοχικά τα κόστη διάσχισης των ακμών. Παράλληλα διαβάζει τον κωδικό κάθε γραμμής και όταν υπάρχει αλλαγή προστίθεται στο κόστος ο χρόνος μετεπιβίβασης. Συμβατικά έχει θεωρηθεί ότι ο χρόνος αυτός είναι ο μισός από την τυπική συχνότητα της γραμμής.

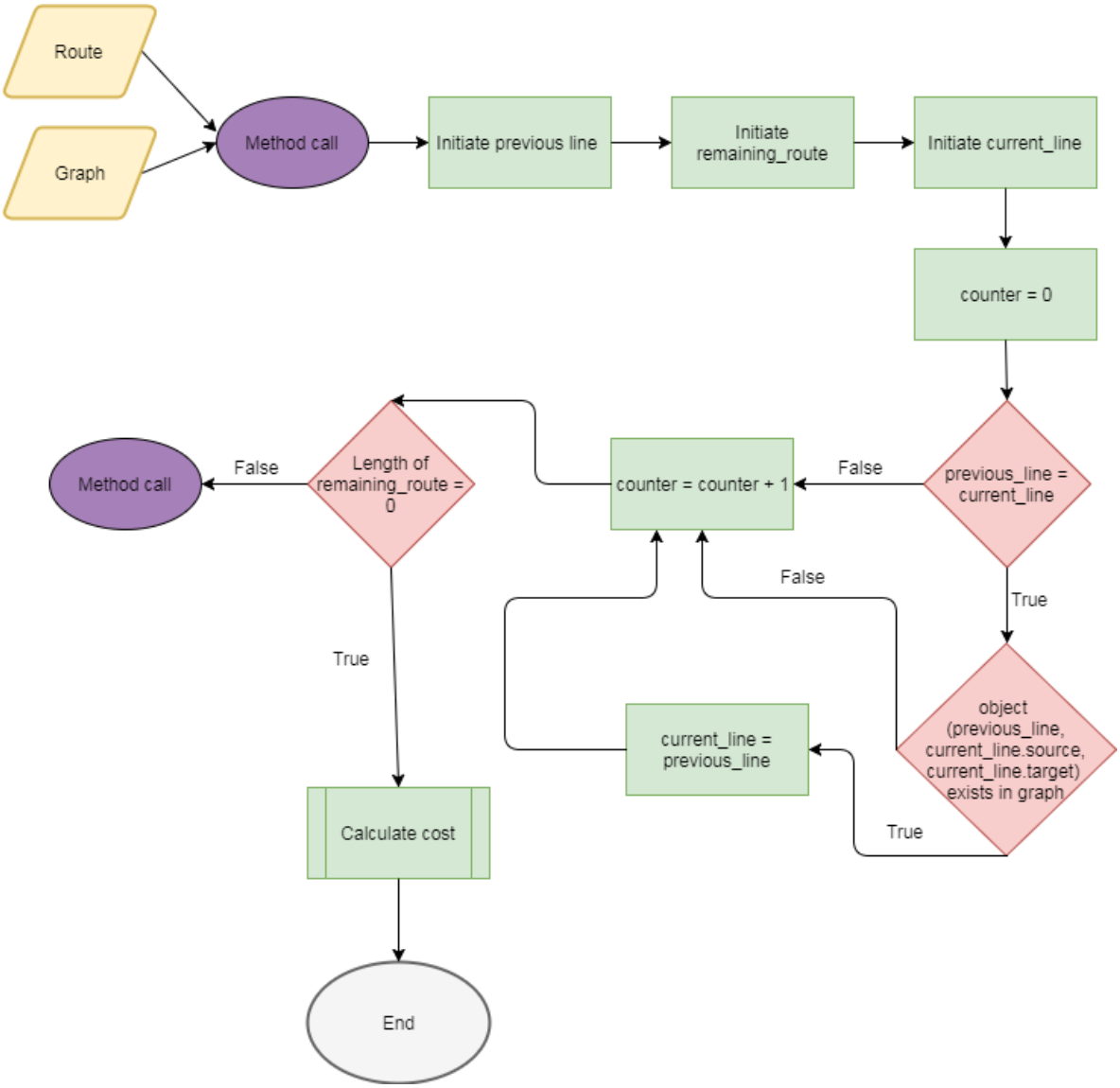
Ο κωδικός του κόμβου και το τελικό του κόστος καταχωρούνται σε μία καινούργια λίστα, η οποία όταν ολοκληρωθεί η διαδικασία θα περιέχει δύο στήλες μία με τους κωδικούς των κόμβων και μία με το χρονικό κόστος του καθενός. Επίσης δημιουργήθηκε μία ακόμα συνάρτηση προκειμένου να εξάγεται η λίστα αυτή σε αρχείο csv.

3.5.5. Διαγράμματα ροής του προγράμματος

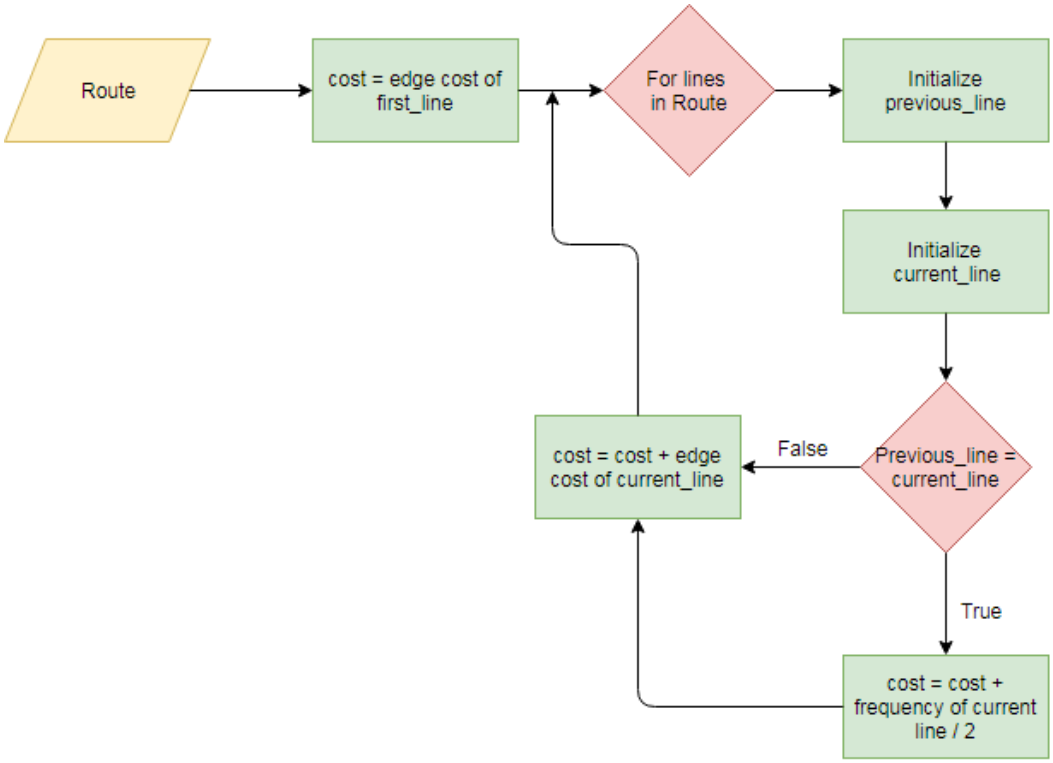
Παρουσιάζονται παρακάτω το διάγραμμα ροής του κυρίως κορμού του προγράμματος και τα αντίστοιχα των σημαντικότερων συναρτήσεων που καλούνται στο πρόγραμμα.



Διάγραμμα ροής 1 : Διαδικασία εξαγωγής χρονικού κόστους κόμβος



Διάγραμμα ροής 2 : Διαδικασία ελέγχου μετεπιβιβάσεων σε μία διαδρομή



Διάγραμμα ροής 3 : Διαδικασία υπολογισμού συνολικού κόστους διαδρομής

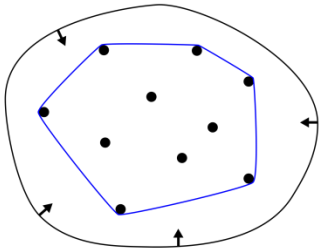
3.6. Σχεδιασμός Ισοχρονικών καμπυλών

Το πιο σημαντικό βήμα στην εκτέλεση της εφαρμογής είναι, με δεδομένο το χρονικό κόστος των κόμβων, ο σχεδιασμός των ισοχρονικών καμπυλών που είναι και ο σκοπός της εργασίας. Για το σκοπό αυτό δοκιμάστηκαν κάποιοι αλγόριθμοι και μέθοδοι ώστε να επιλεχθεί εκείνος που μπορεί να δώσει πιο ακριβή και ευανάγνωστα αποτελέσματα. Σε αυτήν την παράγραφο περιγράφεται η διαδικασία επιλογής της μεθοδολογίας που επιλέχθηκε για την απεικόνιση του χρόνου διαδρομής.

3.6.1. Convex hull

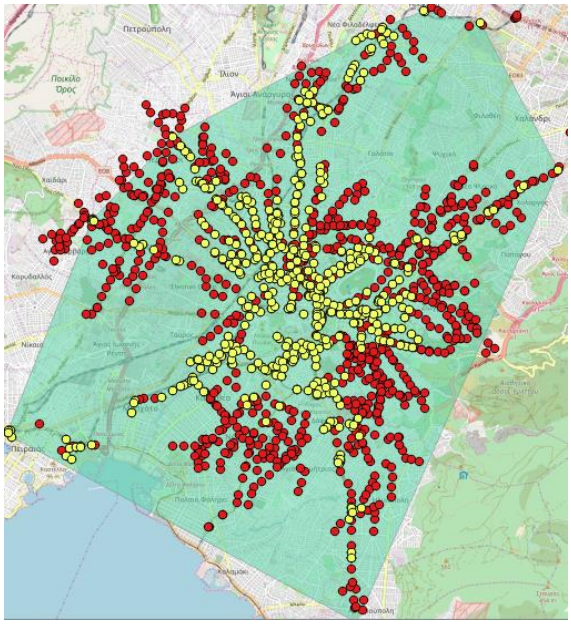
Η πρώτη προσέγγιση που έγινε είναι το Convex Hull. Το Convex Hull ενός συνόλου σημείων σε ένα Ευκλείδειο επίπεδο είναι το ελάχιστο κυρτό πολύγωνο που περιέχει όλα τα σημεία. Πιο περιγραφικά, θα λέγαμε ότι είναι το πολύγωνο που θα σχηματιζόταν αν

τεντώνονταν μία ελαστική κορδέλα γύρω από τα σημεία. Στην Εικόνα 30 απεικονίζεται ένα παράδειγμα του Convex Hull ενός μικρού συνόλου σημείων.



Εικόνα 30 : Το Convex Hull ενός μικρού συνόλου σημείων

Για την υλοποίηση του Convex Hull χρησιμοποιήθηκε ο αντίστοιχος αλγόριθμος από το λογισμικό QGIS. Τα αποτελέσματα δεν ήταν ικανοποιητικά όπως φαίνεται στην Εικόνα 31. σημεία μεγαλύτερου χρονικού κόστους είναι μέσα στο Convex Hull των κόμβων με χρονικό κόστος από 0 έως 20 λεπτά.



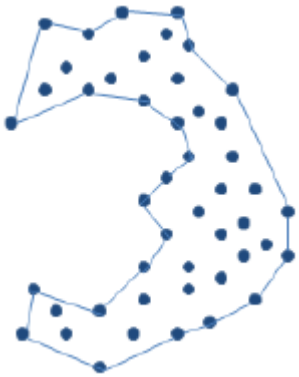
Εικόνα 31 : Convex Hull των κίτρινων κόμβων. Οι κίτρινοι είναι οι κόμβοι με χρονικό κόστος 0 – 20 λεπτά και οι κόκκινοι 20 – 30 λεπτά

3.6.2. Concave hull

Η επόμενη προσέγγιση που έγινε είναι η δημιουργία των ισοχρονικών καμπυλών με χρήση του αλγορίθμου Concave Hull (Jin-Seo Park & Se-Jong Oh, 2012) . Για το Concave Hull

δεν υπάρχει κάποιος συγκεκριμένος ορισμός αλλά αποτελεί τον επεξεργασμένο Convex Hull έτσι ώστε οι μεγάλες περιοχές χωρίς σημεία που περικλείει να αφαιρούνται. Στην Εικόνα 32 φαίνεται ένα Concave Hull ενός συνόλου σημείων.

Το κίνητρο για τη χρήση του Concave Hull είναι ότι θα μπορούσε να εξαφανίσει το κύριο μειονέκτημα του Convex Hull που είναι η συμπερίληψη μη ισοχρονικών κόμβων στο παραχθέν πολύγωνο. Δηλαδή, κόβοντας μεγάλες περιοχές του Convex Hull που δεν περιέχουν ισοχρονικούς κόμβους, πιθανότατα θα αποκλείει και πολλούς μη ισοχρονικούς κόμβους που πριν ήταν μέσα στο πολύγωνο.



Εικόνα 32 : Το Concave Hull ενός μικρού συνόλου σημείων

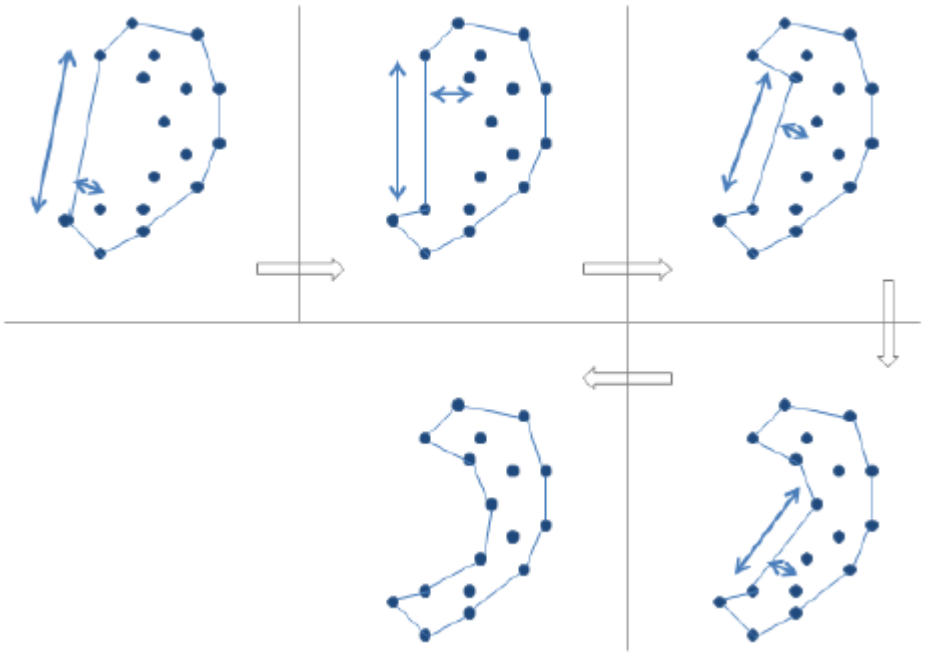
Εφόσον δεν υπάρχει καθολικός ορισμός, δεν υπάρχει και καθολικός αλγόριθμος που να υπολογίζει το Concave Hull. Υπάρχουν διάφορες προσεγγίσεις και αλγόριθμοι που προσπαθούν να δώσουν ένα ικανοποιητικό αποτέλεσμα. Οι αλγόριθμοι αυτοί είναι παραμετρικοί. Το Concave Hull στο σχήμα 3.10 θα μπορούσε να είναι αρκετά διαφορετικό αν η παραμετροποίηση του αλγορίθμου που έδωσε το αποτέλεσμα ήταν διαφορετική. Αυτό είναι σίγουρα ένα μειονέκτημα καθώς ακόμα κι αν υλοποιήσουμε έναν αλγόριθμο, σε κάποιες εκτελέσεις το αποτέλεσμα μπορεί να είναι ως κάποιο βαθμό απρόβλεπτο.

Για να δοκιμαστεί κατά πόσο το Concave Hull θα ήταν μια καλή λύση για την εφαρμογή χρησιμοποιήθηκε μία έτοιμη υλοποίηση ενός αλγορίθμου υπολογισμού του, αυτή που παρέχει το λογισμικό QGIS ως επέκταση (plugin). Ακολουθεί μια συνοπτική περιγραφή της λειτουργίας του.

- Αρχικά, υπολογίζεται το Convex Hull όπως περιγράφηκε στην προηγούμενη παράγραφο.

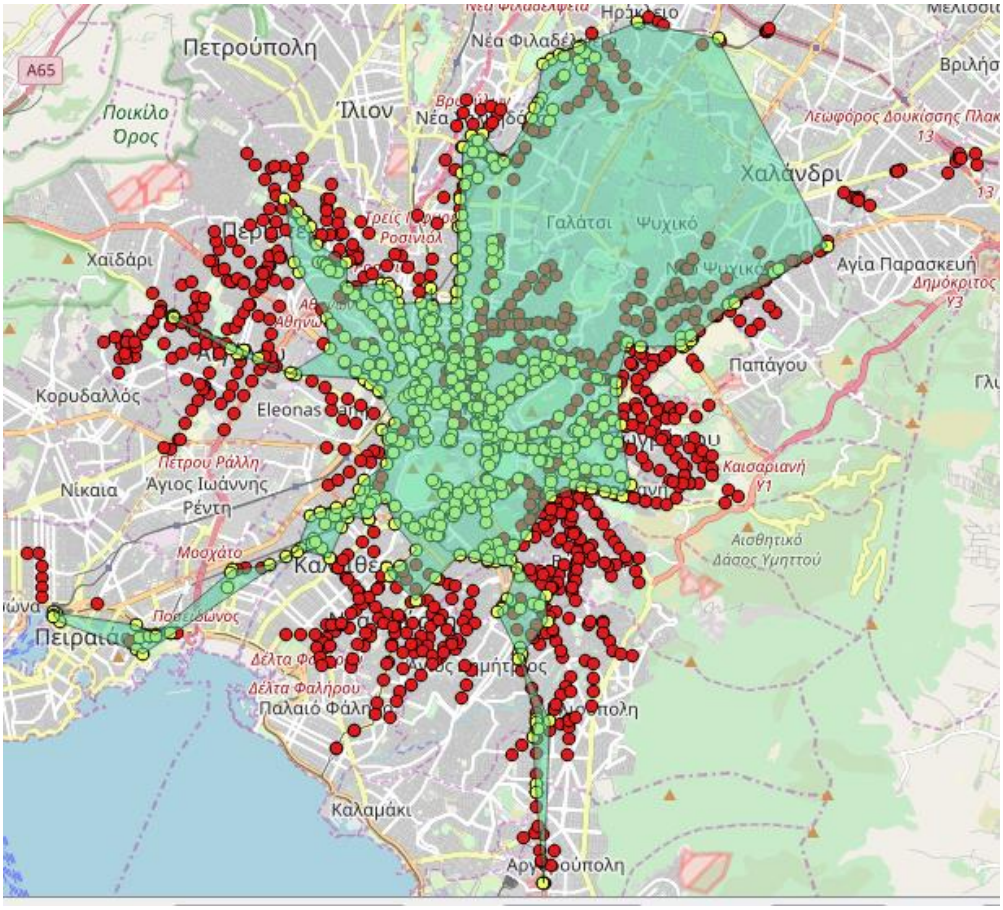
- Για κάθε ακμή του Convex Hull γίνονται τα εξής:
    - Υπολογίζεται το μήκος της ακμής.
    - Αναζητείται το κοντινότερο στην ακμή σημείο εντός του πολυγώνου.
    - Υπολογίζεται η κάθετη απόσταση του κοντινότερου σημείου και της ακμής.
- Αν ο λόγος του μήκους της ακμής προς την απόσταση του κοντινότερου μέσα σημείου είναι μεγαλύτερος από μία δεδομένη παράμετρο N τότε η αρχική ακμή παύει να υπάρχει και δημιουργούνται δύο ακμές που η κάθε μια έχει άκρα ένα σημείο της αρχικής ακμής και το κοντινότερο μέσα σημείο.
- Η διαδικασία συνεχίζεται για όλες τις ακμές του Convex Hull αλλά και για τις νέες ακμές που δημιουργούνται κατά την εκτέλεση του αλγορίθμου.

Στην Εικόνα 33 δίνεται ένα παράδειγμα εκτέλεσης μιας επανάληψης του αλγορίθμου για ένα μικρό σύνολο σημείων.



Εικόνα 33 : Ένα παράδειγμα εκτέλεσης του αλγορίθμου για το Concave Hull για ένα μικρό σύνολο σημείων

Στην Εικόνα 34 δίνεται το αποτέλεσμα που έδωσε η εκτέλεση του αλγορίθμου στο οδικό δίκτυο της Αθήνας για το ίδιο αρχικό σημείο και χρονικό διάστημα με το αντίστοιχο σχήμα του Convex Hull.



Εικόνα 34: Οι κίτρινοι είναι οι ισοχρονικοί κόμβοι 20 λεπτών ενώ οι κόκκινοι οι μη ισοχρονικοί. Το Concave Hull διακρίνεται με ανοικτό πράσινο πολύγωνο.

Είναι φανερό ότι το Concave Hull αποκλείει πολλούς μη ισοχρονικούς κόμβους οι οποίοι περιέχονται στο Convex Hull. Επιπλέον εποπτικά μπορεί κανείς να διακρίνει ότι το τελικό σχήμα του πολυγώνου είναι σημαντικά πιο ακριβές και πλησιάζει στο ιδανικό «με το μάτι». Παρ' όλα αυτά στο πάνω μέρος διακρίνονται κάποιες ανωμαλίες που οφείλονται στην προαναφερθείσα εξάρτηση από παραμέτρους. Έτσι ένας σημαντικός αριθμός μη ισοχρονικών κόμβων περικλείεται και σε αυτήν την καμπύλη, γεγονός που καθιστά και αυτή τη μέθοδο μη επαρκής.

3.6.3. Δημιουργία ζωνών (Buffers)

Λόγω των σημαντικών μειονεκτημάτων των δύο προηγούμενων προσεγγίσεων, κρίθηκε αναγκαίο να βρεθεί μία άλλη προσέγγιση η οποία θα δίνει πιο αξιόπιστα και ευανάγνωστα αποτελέσματα. Ένας ακόμα σημαντικός παράγοντας που συντέλεσε στην αναζήτηση

αποδοτικότερης λύσης είναι ότι σε καμία από τις δύο προηγούμενες προσεγγίσεις δεν είναι δυνατό να εισαχθεί ο εναπομείναντας χρόνος ως χρόνος περπατήματος. Για παράδειγμα αν ένας κόμβος έχει χρονικό κόστος 4 λεπτά και θέλουμε να απεικονίσουμε την ισοχρονική καμπύλη των 5 λεπτών θα πρέπει να δίνεται η δυνατότητα στον επιβάτη να χρησιμοποιήσει τον υπολειπόμενο χρόνο του ενός λεπτού βαδίζοντας, όπως θα συνέβαινε στην πραγματικότητα. Με αυτόν τον τρόπο θα φτάσει πιο μακριά και η ισόχρονη των 5 λεπτών θα καλύπτει μεγαλύτερη επιφάνεια.

Το πρόβλημα του χρόνου της πεζής μετακίνησης στο τέρμα μιας διαδρομής αντιμετωπίζεται με την εισαγωγή μίας ζώνης (buffer) γύρω από τους ισοχρονικούς κόμβους. Ταυτόχρονα έτσι δημιουργείται και η επιθυμητή ισοχρονική καμπύλη.

Με δεδομένο ότι στην εφαρμογή το χρονικό εύρος της μετακίνησης με μέσα μαζικής μεταφοράς είναι από πέντε(5) ως ενενήντα(90) λεπτά και οι ισόχρονες θα απεικονίζονται με βήμα πέντε(5) λεπτών, οι κόμβοι ομαδοποιήθηκαν με βάση το χρονικό τους κόστος ως εξής :

- Κόμβοι με χρονικό κόστος έως 5 λεπτά
- Κόμβοι με χρονικό κόστος έως 10 λεπτά
- Κόμβοι με χρονικό κόστος έως 15 λεπτά
- Κόμβοι με χρονικό κόστος έως 20 λεπτά
- Κόμβοι με χρονικό κόστος έως 25 λεπτά
- Κόμβοι με χρονικό κόστος έως 30 λεπτά
- Κόμβοι με χρονικό κόστος έως 35 λεπτά
- Κόμβοι με χρονικό κόστος έως 40 λεπτά
- Κόμβοι με χρονικό κόστος έως 45 λεπτά κοκ

Επιπλέον είναι απαραίτητο να προσδιοριστεί ο μέγιστος επιτρεπόμενος χρόνος βαδίσματος από το τέρμα μίας διαδρομής.

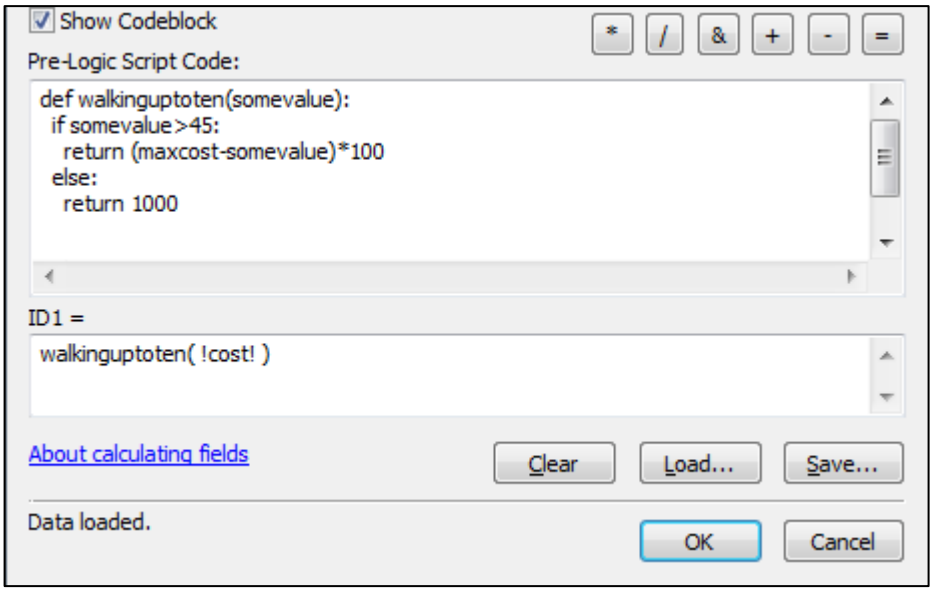
Η απόσταση που μπορεί να διανύσει ένας άνθρωπος διαφέρει εξαιτίας πολλών παραγόντων, όπως το περιβάλλον περπατήματος προς το σημείο διέλευσης της πρόσβασης (Agrawal, 2008), τη δομή της ηλικιακής ομάδας στις περιοχές ζήτησης, και την αξιοπιστία των δημόσιων υπηρεσιών διέλευσης. Ενδεικτικά αναφέρεται ότι για τους ηλικιωμένους ανθρώπους και τα παιδιά προσχολικής ηλικίας, η μέγιστη διανυόμενη απόσταση είναι τα 190 μέτρα, για τα παιδιά που πηγαίνουν σχολείο (5+) είναι από 191 μέχρι 380 μέτρα ενώ για τους έφηβους και τους ενήλικες διακυμαίνεται από 381 μέχρι 600 μέτρα (Azmi, 2012; Mavoa, 2012). Στη βιβλιογραφία επίσης, αναφέρεται ότι στα πλαίσια του σχεδιασμού η μέτρηση της διανυόμενης απόστασης με τα πόδια συμβολίζεται με ακτίνα μήκους 400 μέτρων η οποία αντιστοιχεί στην ελάχιστη απόσταση που διατίθεται να διανύσει ο μέσος άνθρωπος για να πάει σε τοπικές εγκαταστάσεις (στη δουλειά του, σε υπηρεσίες),

αντιπροσωπεύοντας μια διαδρομή διάρκειας περίπου 5 λεπτών ενώ τα 1000 μέτρα είναι μια μεγάλη απόσταση, προτεινόμενη για το περπάτημα σε ένα κέντρο πόλης (Azmi, 2012).

Με βάση τα δεδομένα αυτά και προσπαθώντας για όσο το δυνατό ρεαλιστικά αποτελέσματα, επιλέχθηκε ο μέγιστος χρόνος πεζής μετακίνησης από το τέρμα μιας διαδρομής τα δέκα λεπτά με ταχύτητα βαδίσματος 100 μέτρα/λεπτό. Αυτό μεταφράζεται σε 1000 μέτρα, δηλαδή το μέγιστο που προτείνεται από την βιβλιογραφία και είναι αρκετά ρεαλιστικό για ένα δίκτυο συγκοινωνιών όπως αυτό της Αττικής.

Συνεπώς για κάθε κατηγορία κόμβων δημιουργήθηκε ένα ξεχωριστό python script από το οποίο θα υπολογίζεται η ακτίνα του buffer. Η λογική πίσω από τα εν λόγω script είναι η ίδια, αλλάζουν όμως οι παράμετροι, καθώς αλλάζει το εύρος του κόστους.

Συγκεκριμένα για κάθε κόμβο της κατηγορίας ελέγχεται αν η διαφορά του κόστους του (cost) με το μέγιστο κόστος της κατηγορίας (maxcost) είναι περισσότερο από δέκα λεπτά. Σε αυτήν την περίπτωση η ακτίνα του buffer ορίζεται στα 1000 μέτρα, δηλαδή τη μέγιστη δυνατή. Στην περίπτωση που η διαφορά είναι μικρότερη από δέκα, η ακτίνα ορίζεται σύμφωνα με τον τύπο (maxcost – cost) *100, που αποτελεί εφαρμογή του τύπου της ευθύγραμμης ομαλής κίνησης x=u*t. Στην Εικόνα 35 φαίνεται ο κώδικας σε python για τον υπολογισμό της ακτίνας του buffer των κόμβων κόστους μέχρι 55 λεπτά. Η εργασία αυτή έγινε στο Field Calculator του λογισμικού ArcGIS.

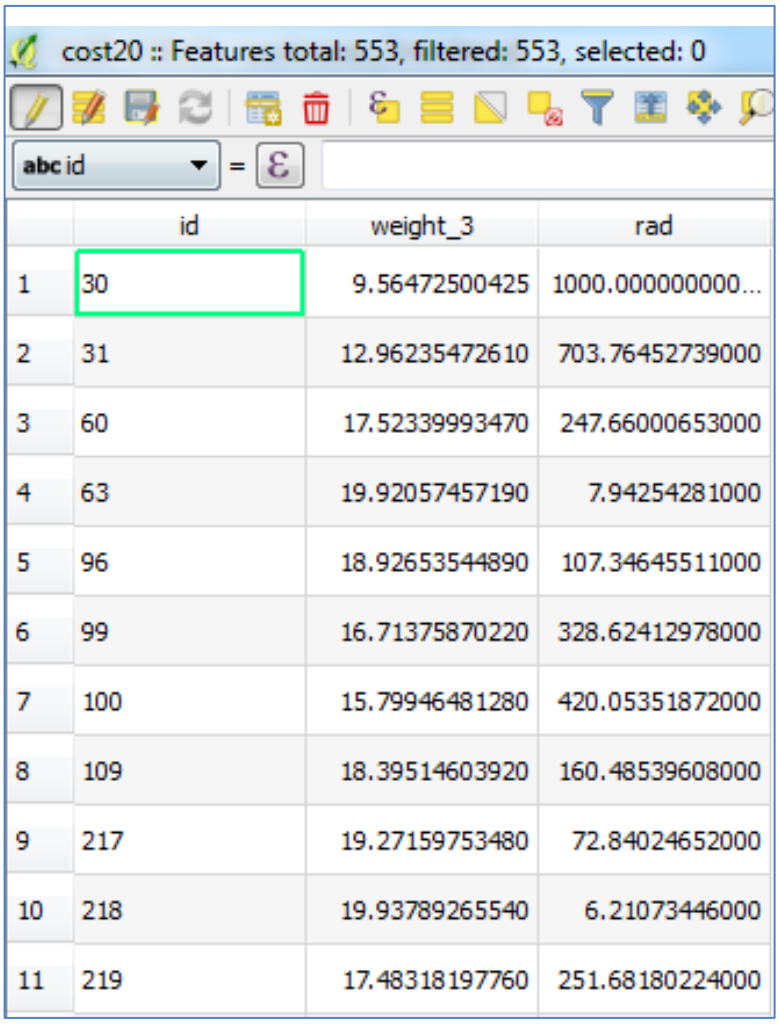


Εικόνα 35 : Κώδικας για τον υπολογισμό της ακτίνας των ζωνών (Buffers)

Στην πρώτη σειρά ορίζεται το όνομα της συνάρτηση και το πλήθος των μεταβλητών που θα έχει ως είσοδο. Στην δεύτερη ξεκινάει η δομή ελέγχου if, που ελέγχει τη διαφορά του μέγιστου κόστους από το τρέχον κόστος και αποδίδει την τιμή του κελιού κατά περίπτωση.

Στο δεύτερο παράθυρο καλείτε η συνάρτηση και αρχικοποιείται η τιμή somevalue με τη στήλη που πρέπει να εισαχθεί για τους υπολογισμούς, που δεν είναι άλλη από τη στήλη του κόστους (cost). Έτσι τελικά υπολογίζεται η ακτίνα του buffer κάθε σημείου και αποθηκεύεται σε ένα νέο κελί με όνομα rad. Στον Πίνακα 8 φαίνεται ο πίνακας ιδιοτήτων των κόμβων με κόστος έως 20 λεπτά με υπολογισμένη την ακτίνα του buffer. Η στήλη weight_3 αντιπροσωπεύει τη στήλη του κόστους.

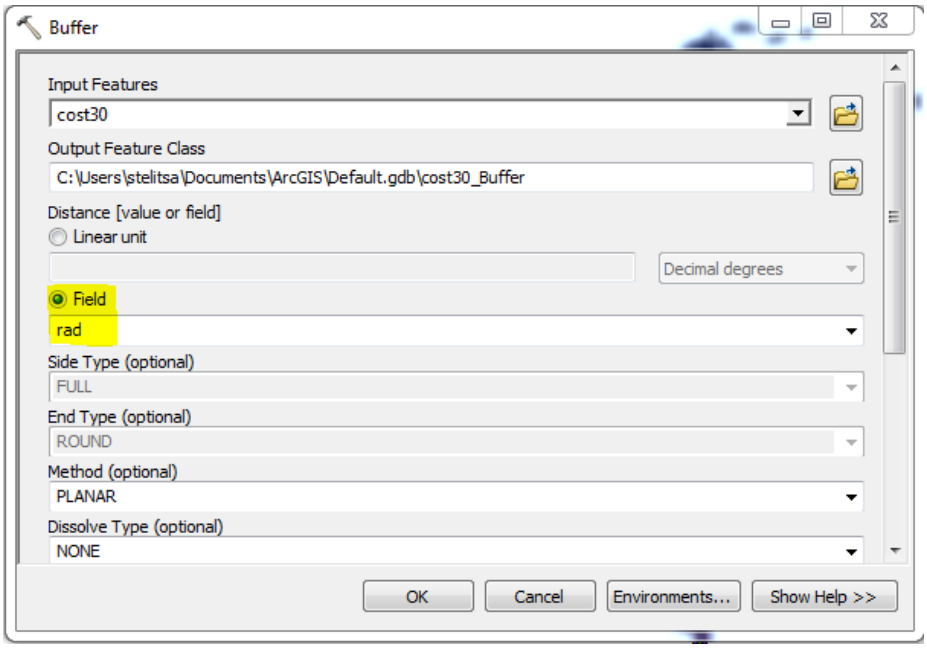
Πίνακας 8 : Υπολογισμένη ακτίνα της ζώνης σχεδίασης



	id	weight_3	rad
1	30	9.56472500425	1000.000000000...
2	31	12.96235472610	703.76452739000
3	60	17.52339993470	247.66000653000
4	63	19.92057457190	7.94254281000
5	96	18.92653544890	107.34645511000
6	99	16.71375870220	328.62412978000
7	100	15.79946481280	420.05351872000
8	109	18.39514603920	160.48539608000
9	217	19.27159753480	72.84024652000
10	218	19.93789265540	6.21073446000
11	219	17.48318197760	251.68180224000

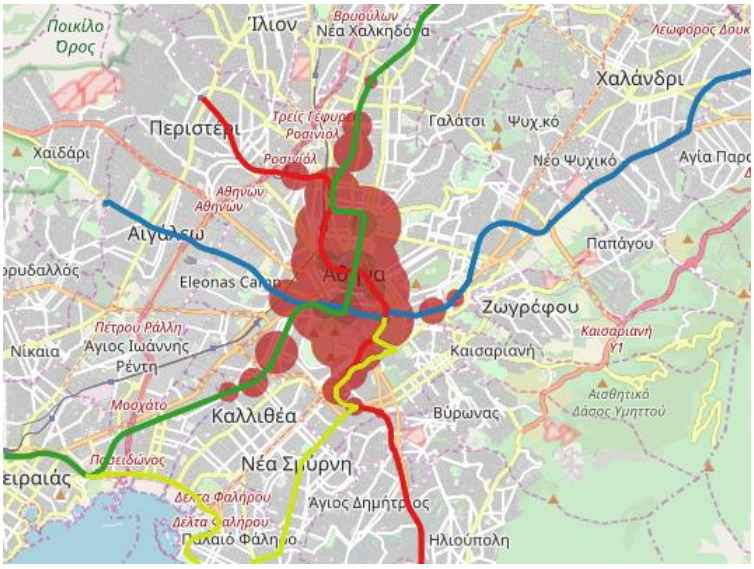
Κατόπιν έπρεπε σε κάθε κατηγορία να δημιουργηθούν buffers με ακτίνα σύμφωνα με τη στήλη rad. Αυτό είναι πολύ εύκολο καθώς από την επιλογή Buffer του ArcGis, παρέχεται η

δυνατότητα ορισμού ακτίνας από κάποια στήλη του πίνακα ιδιοτήτων του επιπέδου (Εικόνα 36).



Εικόνα 36 : Δημιουργία Buffer με βάση το δημιουργημένο πεδίο rad

Το αποτελέσματα της μεθόδου αυτής είναι ιδιαίτερα ικανοποιητικά. Στην εικόνα φαίνεται η ισοχρονική καμπύλη των 10 λεπτών. Οι χρωματιστές γραμμές είναι οι γραμμές του μετρό. Όπως φαίνεται οι ισόχρονες «ανοίγουν» κατά μήκος των αξόνων του μετρό γεγονός που είναι λογικό, αφού το μετρό κινείται με μεγαλύτερη ταχύτητα και έχει μεγαλύτερη συχνότητα από τα λεωφορεία.



Εικόνα 37 : Παρουσίαση ισοχρονικών καμπυλών μέσω δημιουργίας ζωνών

4. ΤΕΧΝΟΛΟΓΙΕΣ ΚΑΙ ΥΠΗΡΕΣΙΕΣ ΔΙΑΧΥΣΗΣ ΓΕΩΧΩΡΙΚΩΝ ΔΕΔΟΜΕΝΩΝ ΣΤΟΝ ΠΑΓΚΟΣΜΙΟ ΙΣΤΟ

Οι υπηρεσίες στον Παγκόσμιο Ιστό (Web Services) αποτελούν αυτόνομες εφαρμογές που συνήθως έχουν δημιουργηθεί με γλώσσες προγραμματισμού, όπως Java, C κ.α., είναι συμβατές με τη γλώσσα και τις τεχνολογίες XML και είναι ανεξάρτητες λειτουργικών συστημάτων και γλωσσών προγραμματισμού. Οι Web Services αναπτύσσονται βάσει συγκεκριμένων προδιαγραφών όπως οι SOAP, WSDL, UDDI, RDF κ.α., οι οποίες ορίζουν τους τρόπους επικοινωνίας μεταξύ αυτών και των διαδικτυακών εφαρμογών και εξασφαλίζουν τη συμβατότητα και κατανόηση των λειτουργιών μεταξύ των επιμέρους εφαρμογών που αναπτύσσονται στο διαδίκτυο. Συνεπώς, τα πρότυπα αυτά στοχεύουν στην επίτευξη της διαλειτουργικότητας ώστε η επικοινωνία και η ανταλλαγή πληροφοριών να είναι ενιαία, μειώνοντας τους απαιτούμενους πόρους για μετάφραση και χρήση των πληροφοριών αυτών. Για τη μεταξύ τους επικοινωνία οι Web Services χρησιμοποιούν το πρωτόκολλο HTTP και τη γλώσσα XML για τη μεταφορά δεδομένων. Τα τελευταία χρόνια, μια νέα μορφή δεδομένων, το πρότυπο JSON (JavaScript Object Notation), χρησιμοποιείται ευρέως για την ανταλλαγή πληροφοριών μεταξύ εφαρμογών στον Παγκόσμιο Ιστό και προσφέρει συγκριτικά πλεονεκτήματα έναντι της XML.

Σε αντιστοιχία με τις Web Services, οι υπηρεσίες γεωχωρικού περιεχομένου (Geospatial Web Services) έχουν αναπτυχθεί ώστε να παρέχουν τυποποιημένες διαδικασίες για τη διάχυση γεωχωρικών δεδομένων στον Παγκόσμιο Ιστό (Di et al., 2005; Kralidis, 2007). Η συλλογή των υπηρεσιών αυτών αναφέρεται συχνά με τον όρο Γεωχωρικός Παγκόσμιος Ιστός (Geospatial Web).

Στο παρόν κεφάλαιο δίνεται μια συνολική περιγραφή και ανάλυση των κεντρικών τεχνολογιών και υπηρεσιών που χρησιμοποιούνται σήμερα για την ανάπτυξη γεωχωρικών εφαρμογών στον Παγκόσμιο Ιστό.

4.2. Προδιαγραφές του OGC

Οι υπηρεσίες γεωχωρικού περιεχομένου – Geospatial Web Services που αναφέρθηκαν στην εισαγωγική ενότητα αναπτύσσονται από τον οργανισμό OGC ο οποίος ειδικεύεται στην ανάπτυξη ανοιχτών προτύπων και υπηρεσιών για τη διάχυση γεωχωρικών και χαρτογραφικών δεδομένων στο Web. Οι προδιαγραφές αυτές περιγράφουν τις μορφές ανταλλαγής των δεδομένων καθώς και τη διεπαφή μέσω της οποίας διατυπώνονται τα αιτήματα για τη χρήση των γεωχωρικών υπηρεσιών. Οι χρήστες λειτουργούν είτε ως καταναλωτές (στη πλευρά του πελάτη) ή ως προμηθευτές (στη πλευρά του εξυπηρετητή) ενώ δεν απαιτούνται ιδιαίτερες γνώσεις και πολλές τεχνικές λεπτομέρειες για την

αξιοποίηση των υπηρεσιών (Στεφανάκης, 2009). Οι υπηρεσίες αυτές εφαρμόζονται κατόπιν αιτήματος του πελάτη μέσω του φυλλομετρητή εντός του URL το οποίος εκτός από τις βασικές παραμέτρους του πρωτοκόλλου HTTP περιλαμβάνει και καθορισμένες παραμέτρους που έχουν οριστεί από τον OGC για κάθε υπηρεσία ξεχωριστά. Ο OGC επίσης έχει προδιαγράψει τα πρότυπα (μορφές) ανταλλαγής δεδομένων όπως τα GML και KML. Ιδιαίτερα δημοφιλής είναι το πρότυπο GeoJSON που χρησιμοποιεί το συντακτικό του JSON για την περιγραφή γεωχωρικών δεδομένων. Στην ενότητα 4.3 δίνεται μια συνολική περιγραφή των προτύπων αυτών.

Στο σύνολό τους οι υπηρεσίες έχουν ορισμένες κοινές έννοιες και λειτουργίες όπως περιγράφονται στην έκθεση OGC Web Services Commons Specification (OGC & Whiteside, 2007) οι οποίες εξασφαλίζουν τη μεταξύ τους συνέργεια και διαλειτουργικότητα και μειώνουν τους πόρους για τους προγραμματιστές και οργανισμούς στην υιοθέτηση νέων προδιαγραφών. Για παράδειγμα, η κοινή λειτουργία GetCapabilities επιστρέφει στην εφαρμογή – πελάτη ένα XML έγγραφο που ενημερώνει για τα διαθέσιμα θεματικά επίπεδα και μεταδεδομένα αυτών που φιλοξενεί ο εξυπηρετητής.

4.2.1. Υπηρεσία Web Map Service (WMS)

Η WMS αποτελεί τη θεμελιώδη υπηρεσία του OGC και προσφέρεται για τη διάχυση γεωαναφερμένων χαρτών στον Παγκόσμιο Ιστό. Ο πελάτης, μέσω μιας απλοϊκής HTTP διεπαφής αιτείται έναν χάρτη από έναν (χωρικό) εξυπηρετητή συμβατό με την υπηρεσία WMS, ο οποίος αποκρίνεται επιστρέφοντας το χάρτη σε μορφή εικόνας (GIF, PNG, κ.λπ.) που μπορεί να απεικονιστεί εύκολα στο φυλλομετρητή του πελάτη. Το αίτημα συνοδεύεται από μια σειρά παραμέτρων που ορίζουν το περιεχόμενο του χάρτη, τη χωρική του έκταση (extent), το προβολικό σύστημα, τη μορφή της εικόνας κ.α. Εκτός από την αίτηση μιας εικόνας – χάρτη (GetMa*p*), η υπηρεσία προσφέρει τη διατύπωση περαιτέρω αιτημάτων για πληροφόρηση σχετικά με τις τιμές γνωρισμάτων μιας γεωγραφικής οντότητας (GetFeatureInfo) ή το συμβολισμό (GetLegendGraphic) (Στεφανάκης, 2009).

4.2.2. Υπηρεσία Web Feature Service (WFS)

Η υπηρεσία WFS αποτελεί μια μέθοδο διάχυσης γεωγραφικών οντοτήτων στο Web. Η WFS προσφέρει λειτουργίες που επιτρέπουν την περιγραφή των διαθέσιμων οντοτήτων (DescribeFeatureType) και την ανάκτηση αυτών με χρήση του αιτήματος GetFeature. Τα δεδομένα αποστέλλονται σε διανυσματική μορφή (vector data), συνήθως βάσει των

προτύπων GML και GeoJSON, ωστόσο, ορισμένες υλοποιήσεις WFS παρέχουν περισσότερες κωδικοποιήσεις, όπως GeoRSS και Shapefile (Στεφανάκης, 2009).

Η επέκταση WFS-T (Web Feature Service – Transactional) επιτρέπει στο χρήστη να προβαίνει σε ενέργειες εισαγωγής (insert), ενημέρωσης (update) και διαγραφής (delete) γεωγραφικών οντοτήτων με την προϋπόθεση ότι ο χωρικός εξυπηρετητής αντλεί τα δεδομένα από έναν εξυπηρετητή Βάσεων Δεδομένων. Η χρήση της WFS-T αποτελεί μια χρηστική λειτουργία απαλλάσσοντας τις εφαρμογές – πελάτες να επικοινωνούν με εξυπηρετητές Χωρικών Βάσεων Δεδομένων (Spatial Databases) μέσω γλωσσών συγγραφής σεναρίων στον εξυπηρετητή (server-side scripting) όπως η PHP, Python κ.α., μειώνοντας τους πόρους και το χρόνο για την επεξεργασία των δεδομένων. Συνήθως οι χρήστες αλληλοεπιδρούν με τις υπηρεσίες WFS μέσω διαδικτυακών εφαρμογών ή κλασικών GIS λογισμικών (π.χ. QGIS), επιτρέποντας τους να ανακαλύπτουν και να ανακτούν δεδομένα από διάφορες πηγές στον Παγκόσμιο Ιστό δημιουργώντας χαρτογραφικά mashups.

4.2.3. Υπηρεσία Web Processing Service (WPS)

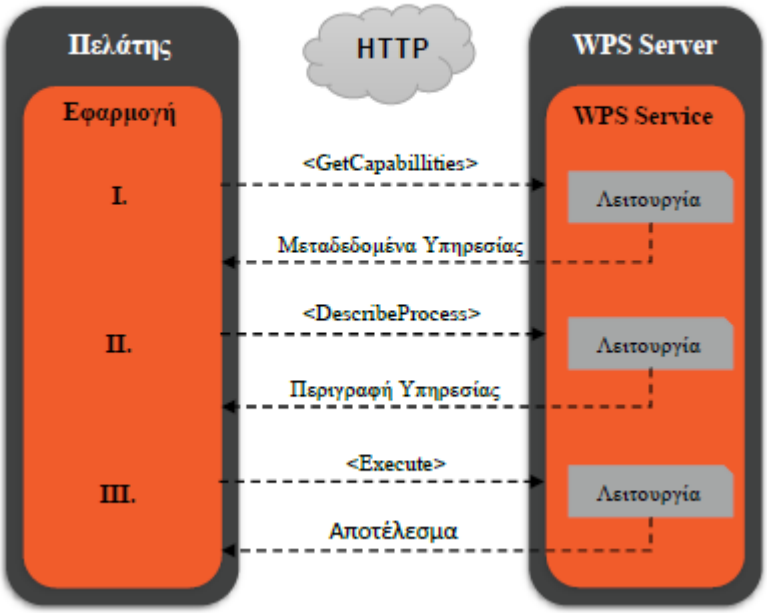
Η υπηρεσία Web Processing Service (WPS) αναπτύχθηκε το 2005 από την OGC (Schut, 2007) με σκοπό τη δημιουργία ενός τυποποιημένου προτύπου και διεπαφής βασισμένη στο πρωτόκολλο HTTP, προσφέροντας λειτουργίες GIS για την επεξεργασία γεωχωρικών δεδομένων διαμέσου του διαδικτύου. Υποστηρίζει και τα δυο βασικά μοντέλα αναπαράστασης γεωγραφικών οντοτήτων, το διανυσματικό (vector) και ψηφιδωτό (raster), επεκτείνοντας το πεδίο εφαρμογών της υπηρεσίας. Η αρχιτεκτονική του προτύπου είναι δομημένη ώστε να υποστηρίζει ακόμη και πολύπλοκες αλγοριθμικές αλληλουχίες επεξεργασίας γεωχωρικών δεδομένων, αποτελώντας ένα πολύτιμο εργαλείο για τους χρήστες, και ιδιαίτερα για τους υπεύθυνους στην επίλυση σύνθετων χωρικών προβλημάτων και στη λήψη αποφάσεων (Castronova et al., 2013)

Όπως και στις υπηρεσίες WMS, WFS, οι παράμετροι της WPS δηλώνονται στο URL. Η επεξεργασία των δεδομένων διεξάγεται με τη χρήση τριών βασικών λειτουργιών (αιτημάτων) που περιγράφονται παραστατικά στη Εικόνα 38:

- Με το αίτημα GetCapabilities ο WPS εξυπηρετητής αποκρίνεται αποστέλλοντας ένα XML έγγραφο με τα μεταδεδομένα, το οποίο ενημερώνει τον πελάτη για τις δυνατότητες της WPS υπηρεσίας που επιθυμεί να καταναλώσει
- Ο χρήστης μπορεί να λάβει περισσότερες πληροφορίες σχετικά με την επιλεγμένη λειτουργία που προσφέρει η WPS υπηρεσία δημιουργώντας το αίτημα DescribeProcess. Οι πληροφορίες που δίνονται αφορούν τις παραμέτρους για τα

δεδομένα εισόδου που επιθυμεί να αποστείλει ο πελάτης και τα δεδομένα εξόδου που επιστρέφει ο WPS εξυπηρετητής.

- Τέλος, βάσει της ενημέρωσης που έλαβε, ο πελάτης μπορεί να δημιουργήσει το αίτημα Execute για την εκτέλεση της λειτουργίας. Ο διακομιστής αποκρίνεται, εκτελώντας την λειτουργία ανάλυσης γεωχωρικών δεδομένων με τις παραμέτρους που έχουν επιλεχθεί, και επιστρέφει το αποτέλεσμα της επεξεργασίας των χωρικών δεδομένων εισόδου του πελάτη.



Εικόνα 38 : Επικοινωνία μιας εφαρμογής – πελάτη με έναν εξυπηρετητή που φιλοξενεί υπηρεσίες WPS.

4.3. Μορφότυποι γεωχωρικών δεδομένων στον Παγκόσμιο Ιστό

Όπως έχει ήδη αναφερθεί, η διάχυση γεωχωρικών δεδομένων στο Web επιτυγχάνεται με τη χρήση ανοιχτών τυποποιημένων προτύπων που αναπτύσσονται κύρια από τον OGC. Η χρήση τους απαλλάσσει τους χρήστες και προγραμματιστές εφαρμογών στη συγγραφή σεναρίων (scripts), για την κωδικοποίηση (encode – write) και μετάφραση (parse) των δεδομένων, καθώς τόσο στο επίπεδο του πελάτη όσο και στου εξυπηρετητή, υπάρχουν έτοιμες λύσεις Διεπαφής Προγραμματισμού Εφαρμογών (API) που παρέχουν μεθόδους γρήγορης ανάγνωσης και χρήσης των γεωχωρικών δεδομένων. Τα σημαντικότερα και πιο δημοφιλή πρότυπα είναι τα GML, KML και GeoJSON που περιγράφονται στη συνέχεια.

4.3.1. GML

Η γλώσσα GML (Geography Markup Language) αποτελεί επέκταση της XML και δημιουργήθηκε για τη κωδικοποίηση γεωγραφικών οντοτήτων με χωρικά και θεματικά γνωρίσματα (Peng και Zhang, 2004). Η νεότερη έκδοση GML 3.11 υποστηρίζει πέραν των απλών γεωμετριών (Σημειακή, Γραμμική, Πολυγωνική), σύνθετες γεωμετρίες όπως κλειστές και αθροιστικές συλλογές απλών γεωμετριών (MultiPoint, MultiLineString, MultiPolygon), κανονικοποιημένες δομές (raster), τοπολογικές σχέσεις, χωρο-χρονικά δεδομένα, συστήματα αναφοράς συντεταγμένων, σύμβολα οπτικοποίησης κ.α. Όλες οι παραπάνω πληροφορίες προδιαγράφονται στο έγγραφο GML σε τρία βασικά σχήματα: α) το Geometry Schema, β) το Feature Schema και γ) το Xlink Schema. Το έγγραφο μπορεί να επεκταθεί με το Application Schema το οποίο περιλαμβάνει πληροφορίες εξαρτώμενες από την εκάστοτε εφαρμογή.

Η GML είναι η δημοφιλέστερη μορφή διάχυσης χωρικών δεδομένων στον Ιστό με την υπηρεσία WFS και υποστηρίζεται από τους μεγαλύτερους εξυπηρετητές χαρτών όπως ο GeoServer και ο MapServer.

4.3.2. KML

Όπως η GML, και η KML (Keyhole Markup Language) υιοθετεί ένα προσαρμοσμένο XML Application Schema για την περιγραφή γεωγραφικών δεδομένων. Η βασική τους διαφορά συνιστάται στη δυνατότητα τρισδιάστατης (3D) αναπαράστασης, και περιήγηση στα δεδομένα, οι οποίες επιτυγχάνονται με τη κωδικοποίηση των παραμέτρων αυτών στο έγγραφο KML με τη χρήση XML ετικετών (tags) όπως <camera>, <gx:FlyTo>, κ.α. Τα γραφικά αυτά μπορούν να απεικονιστούν από λογισμικά οπτικοποίησης και αναπαράστασης της γης όπως το Google Earth και Google Maps. Ωστόσο, το πεδίο εφαρμογής της KML είναι αρκετά περιορισμένο λόγω της ανεπαρκής υποστήριξης σύνθετων γεωμετρικών δομών και σχέσεων μεταξύ των γεωγραφικών οντοτήτων. Οι OGC και οι Google έχουν συμφωνήσει πρόσφατα στην εναρμόνιση της KML με τη GML στο μέλλον, ώστε να χρησιμοποιούν ένα κοινό μοντέλο αναπαράστασης γεωμετριών.

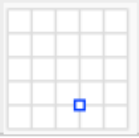
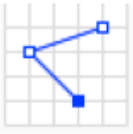
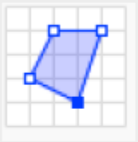
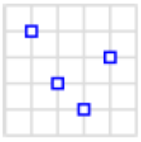
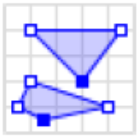
4.3.3. GeoJSON

Το πρότυπο GeoJSON αναπτύχθηκε το 2008 από μια ομάδα προγραμματιστών (Butler, et al., 2008) και έκτοτε χρησιμοποιείται ευρέως στην ανταλλαγή και διάχυση γεωχωρικών δεδομένων. Αποτελεί μια ελαφριά μορφή που μεταφέρεται γρήγορα και μπορεί να

μεταφραστεί εύκολα τόσο στη πλευρά του πελάτη όσο και στον εξυπηρετητή με χρήση της JavaScript και της PHP αντίστοιχα. Σε αντίθεση με τις GML και KML που βασίζονται στο αυστηρά δομημένο συντακτικό της XML με τη χρήση ετικετών, το GeoJSON βασίζεται στο JSON το οποίο ακολουθεί τη σύνταξη της JavaScript. Η δομή του GeoJSON είναι αρκετά απλοϊκή και συντίθεται από ζεύγη χαρακτηριστικών και τιμών τα οποία συνθέτουν ένα GeoJSON αντικείμενο (object). Ένα GeoJSON αντικείμενο μπορεί να κωδικοποιεί μια γεωμετρία, ένα γεωγραφικό αντικείμενο (feature) το οποίο περιγράφεται από τη γεωμετρία και τις συντεταγμένες της σε ένα καθορισμένο σύστημα αναφοράς, ή μια συλλογή γεωγραφικών αντικειμένων (FeatureCollection). Όπως η GML, το GeoJSON υποστηρίζει όλα τα είδη απλών και σύνθετων γεωμετρικών δομών (MultiPoint, MultiLineString, MultiPolygon) αλλά και συλλογές γεωμετρικών αντικειμένων (GeometryCollection) (Πίνακας 9).

Το GeoJSON προσφέρει ορισμένα πλεονεκτήματα έναντι των άλλων μορφών που στηρίζονται στην XML. Η μεταφορά δεδομένων υπό τη μορφή XML είναι χρονοβόρα και απαιτεί μεγάλους αποθηκευτικούς χώρους στον εξυπηρετητή ακόμα και αν τα δεδομένα υπόκεινται σε συμπίεση (Kovanen et al., 2013). Αντιθέτως, το JSON λόγω της απλούστερης συντακτικής δομής και το γεγονός ότι είναι υποσύνολο της JavaScript, το καθιστά εύκολα μεταφράσιμο από τις διαδικτυακές εφαρμογές. Επιπλέον, πέραν της κωδικοποίησης γεωγραφικών οντοτήτων, το πρότυπο JSON χρησιμοποείται για τη δημιουργία κανόνων συμβολισμού και τη γραφική σχεδίαση χαρτών, διαδικασία που μέχρι πρόσφατα στηριζόταν αποκλειστικά σε πρότυπα XML όπως το SLD (Styled Layer Descriptor). Σήμερα, το GeoJSON υποστηρίζεται από τις πιο δημοφιλείς υπηρεσίες χαρτογραφικού περιεχομένου στο Web όπως τον GeoServer, MapServer, OpenLayers και Leaflet.

Πίνακας 9 : Τύποι γεωμετρίας σύμφωνα με το πρότυπο GeoJSON.

Γεωμετρία	GeoJSON Αντικείμενο
Point 	{ "type": "Point", "coordinates": [30, 10] }
LineString 	{ "type": "LineString", "coordinates": [[30, 10], [10, 30], [40, 40]] }
Polygon 	{ "type": "Polygon", "coordinates": [[[30, 10], [40, 40], [20, 40], [10, 20], [30, 10]]] }
MultiPoint 	{ "type": "MultiPoint", "coordinates": [[10, 40], [40, 30], [20, 20], [30, 10]] }
MultiLineString 	{ "type": "MultiLineString", "coordinates": [[[10, 10], [20, 20], [10, 40]], [[40, 40], [30, 30], [40, 20], [30, 10]]] }
MultiPolygon 	{ "type": "MultiPolygon", "coordinates": [[[[30, 20], [45, 40], [10, 40], [30, 20]]], [[[15, 5], [40, 10], [10, 20], [5, 10], [15, 5]]]] }

4.4. Εργαλεία και Συστήματα Λογισμικού Ανοιχτού Κώδικα για διαδικτυακές χαρτογραφικές εφαρμογές

4.4.1. Εξυπηρετητές Γεωχωρικών Δεδομένων και Χαρτογραφικά APIs

Όπως έχει γίνει κατανοητό από τις προηγούμενες ενότητες, η διάχυση γεωχωρικών δεδομένων στον Παγκόσμιο Ιστό επιτυγχάνεται με τη χρήση των γεωχωρικών υπηρεσιών (Geospatial Web Services) του OGC. Ένας εξυπηρετητής γεωχωρικών δεδομένων (Geospatial server) υιοθετεί και υλοποιεί τις υπηρεσίες του OGC προκειμένου να παρέχει ένα τυποποιημένο τρόπο επικοινωνίας και ανταλλαγής πληροφοριών με τις εφαρμογές στον Παγκόσμιο Ιστό. Οι τυπικές λειτουργίες που παρέχουν τέτοιου είδους εξυπηρετητές αφορούν αφενός τη δημοσιοποίηση γεωχωρικού περιεχομένου αλλά υποστηρίζουν και μια σειρά επιπρόσθετων λειτουργιών όπως οπτικοποίηση και επεξεργασία δεδομένων, σύνδεση και άντληση δεδομένων από μια ή περισσότερες κατανεμημένες χωρικές βάσεις δεδομένων κ.α. Σήμερα, στον Παγκόσμιο Ιστό υπάρχει μια πληθώρα γεωχωρικών εξυπηρετητών όπως ο GeoServer, MapServer, Mapnik, ArcGIS for Server κ.α.

4.4.2. OpenLayers

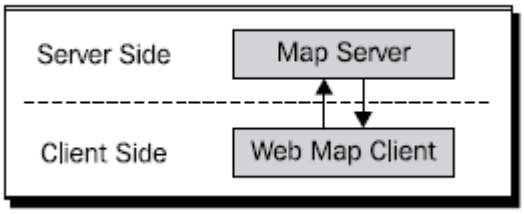
Η OpenLayers είναι μια ανοιχτού κώδικα βιβλιοθήκη JavaScript για την οπτικοποίηση γεωγραφικών δεδομένων στους σύγχρονους φυλλομετρητές (web browsers) και τη δημιουργία του περιβάλλοντος της χαρτογραφικής εφαρμογής. Η βιβλιοθήκη αναπτύχθηκε το 2005 από τη MetaCarta ως απάντηση στην αντίστοιχη κλειστού κώδικα βιβλιοθήκη Google Maps API της Google, ενώ το 2007 υιοθετήθηκε από τον οργανισμό OSGeo. Έκτοτε αποτελεί το πιο δημοφιλές JavaScript Framework για χαρτογραφική αλληλεπίδραση στον Παγκόσμιο Ιστό.

Λειτουργεί ως ένα JavaScript API στον πελάτη ανεξάρτητα από τον εξυπηρετητή και προσφέρει πλούσια διαδραστικά εργαλεία για την ανάπτυξη σύνθετων χαρτογραφικών εφαρμογών. Είναι συμβατή με τις τυποποιήσεις του OGC, συνεπώς χρησιμοποιείται ευρέως για την άντληση και οπτικοποίηση γεωχωρικών δεδομένων που φιλοξενούνται σε χωρικούς εξυπηρετητές όπως ο GeoServer. Για την αμφίδρομη επικοινωνία με τους χωρικούς εξυπηρετητές η OpenLayers χρησιμοποιεί την τεχνολογία AJAX.

Με τη νεότερη έκδοση OpenLayers 3.0 η βιβλιοθήκη ανανεώθηκε ριζικά για να ενσωματώσει νεότερες τεχνολογίες που εισήγαγε η HTML 5 όπως το στοιχείο Canvas που χρησιμοποιείται για τη σχεδίαση γραφικών στο Web, και τη WebGL JavaScript API που χρησιμοποιείται πλέον από τους σύγχρονους φυλλομετρητές για την αναπαράσταση 2D και

3D γραφικών. Η OpenLayers χρησιμοποιεί τις τεχνολογίες αυτές για την αναπαράσταση των διανυσματικών δεδομένων (layers) που εμπεριέχονται στο κεντρικό αντικείμενο *Map* της βιβλιοθήκης.

Η βιβλιοθήκη OpenLayers χρησιμοποιείται για παραγωγή χαρτών από την πλευρά του πελάτη (client side) (Εικόνα 39).



Εικόνα 39 : Παραγωγή χαρτών από την πλευρά του πελάτη

Αυτό ονομάζεται μοντέλο πελάτη/ διακομιστή και είναι ουσιαστικά ο πυρήνας του τρόπου με τον οποίο όλες οι διαδικτυακές εφαρμογές λειτουργούν. Στην περίπτωση μιας εφαρμογής διαδικτυακού χάρτη, κάποιος χάρτης - πελάτης (π.χ. OpenLayers) επικοινωνεί με κάποιον διακομιστή διαδικτυακού χάρτη (π.χ. διακομιστής WMS ή σύστημα υποστήριξης Google Maps). Ουσιαστικά ο χάρτης – πελάτης στέλνει νέα αιτήματα στον διακομιστή κάθε φορά που ο χρήστης περιηγείται στον χάρτη ή εστιάζει σε κάποιο σημείο, καθώς ζητείται η θέαση διαφορετικού αντικειμένου. Αυτό σημαίνει ότι κάθε φορά που ο χρήστης αλληλεπιδρά με τον χάρτη το OpenLayers στέλνει νέα αιτήματα στον διακομιστή και ύστερα συνδέει όλες τις επιστρεφόμενες εικόνες σε μία ώστε το αποτέλεσμα να μοιάζει με έναν μεγάλο, αδιάλειπτο χάρτη.

Ένας διακομιστής χάρτη (map server) παρέχει τον ίδιο το χάρτη. Με το OpenLayers μπορούν να επιλεχθούν πολλαπλοί διακομιστές υποστήριξης, οι οποίοι προστίθενται στον χάρτη ως αντικείμενα layers. Ο χάρτης που παρέχεται από τον διακομιστή ονομάζεται Base Layer, ενώ όλα τα υπόλοιπα αντικείμενα που «κάθονται πάνω» στο στρώμα αυτό ονομάζονται επίπεδα επικάλυψης (Overlay Layers). Όπως αναφέρθηκε και προηγουμένως είναι δυνατό να επιλεχθούν πολλαπλά base layers, μπορεί όμως μόνο ένα να είναι ενεργό κάθε φορά. Τα επίπεδα επικάλυψης (Overlay Layers) έχουν διαφορετική συμπεριφορά. Είναι δυνατό να είναι ενεργά απεριόριστα επίπεδα επικάλυψης χωρίς περιορισμούς, εκτός από προβλήματα ταχύτητας που μπορούν να προκληθούν σε χάρτες με πολύ μεγάλο αριθμό τέτοιων. Ένας ακόμα σημαντικός παράγοντας είναι η σειρά απεικόνισης των επιπέδων αυτών καθώς κάθε νέο επίπεδο που προστίθεται στον χάρτη τοποθετείται στην κορυφή και απεικονίζεται πρώτο (Open Layer’s 2.10 Beginner’s Guide).

5. ΣΧΕΔΙΑΣΗ ΔΙΑΔΙΚΤΥΑΚΗΣ ΕΦΑΡΜΟΓΗΣ

Η δημοσιοποίηση χαρτογραφικού περιεχομένου στον Παγκόσμιο Ιστό αποτελεί σήμερα μια κοινή πρακτική και επιτυγχάνεται με πολλαπλούς τρόπους, εφαρμόζοντας διαφορετικές πρακτικές κι εργαλεία. Στην απλούστερη περίπτωση, η δημοσιοποίηση αφορά σε στατικούς χάρτες με περιορισμένη διαδραστικότητα, ενώ πιο σύνθετες αρχιτεκτονικές μπορούν να οδηγήσουν στην δημοσιοποίηση χαρτών με υψηλή διαδραστικότητα και μεγάλη ευελιξία.

Αποτέλεσμα των εργασιών που περιγράφηκαν στα προηγούμενα κεφάλαια είναι η δημιουργία μιας διαδικτυακής εφαρμογής η οποία επιτρέπει στον χρήστη να προβαίνει στην επιλογή του χρόνου μετακίνησης από την αφετηρία του γράφου, ώστε να εμφανίζεται η αντίστοιχη ισοχρονική καμπύλη. Η εφαρμογή αναπτύσσεται πλήρως στον εξυπηρετητή πράγμα που σημαίνει ότι δεν απαιτείται κανένα ειδικό λογισμικό από τη μεριά του χρήστη πέρα από έναν κοινό φυλλομετρητή. Ο χρήστης δεν απαιτείται να έχει καμιά ειδικευμένη γνώση για το σύνολο των εργασιών που η εφαρμογή εκτελεί.

Στο παρόν κεφάλαιο περιγράφονται κάποιες προσεγγίσεις δημοσιοποίησης χαρτογραφικού περιεχομένου στον Παγκόσμιο Ιστό καθώς και τα εργαλεία και οι διαδικασίες σχεδίασης της διαδικτυακής εφαρμογής.

5.2. Επιλογή βιβλιοθήκης OpenLayers

Στο προηγούμενο κεφάλαιο αναφέρθηκε εκτενώς η βιβλιοθήκη OpenLayers. Επιλέχθηκε λοιπόν για την απεικόνιση των χαρτογραφικών δεδομένων της παρούσας εργασίας και την ανάπτυξη της εφαρμογής που οι χρήστες θα μπορούν να αλληλεπιδρούν με τον χάρτη. Η επιλογή έγινε καθώς αποτελεί τον πιο διαδεδομένο τρόπο απεικόνισης χαρτογραφικού περιεχομένου, με άπλετα παραδείγματα και επαρκείς οδηγίες. Επιπλέον δίνονται αρκετές δυνατότητες σε επίπεδο εργαλείων που μπορούν να εισαχθούν στον χάρτη και διαδραστικότητας με το χρήστη, ενώ παρέχονται πολλαπλές ιδιότητες στα επίπεδα πληροφοριών (layers). Κατόπιν αναλύονται οι ιδιότητες αυτές και περιγράφεται η διαδικασία δόμησης της εφαρμογής.

5.2.1. Ιδιότητες των Layers

Πριν αναλυθούν οι ιδιότητες των layers πρέπει να αναφερθούν οι τύποι δεδομένων που χρησιμοποιούνται. Οι αγκύλες {} υποδεικνύουν τον τύπο δεδομένων JavaScript που αναμένει η παράμετρος. Οι τύποι δεδομένων είναι οι εξής :

- {Array}: Μία σειρά στοιχείων που χωρίζονται με κόμματα και περικλείονται σε παρένθεση. Για παράδειγμα [1,2,3]
- {Boolean} : Οι πιθανές τιμές είναι True ή False
- {Float} : Οι πιθανές τιμές είναι αριθμοί με ένα δεκαδικό στοιχείο
- {Integer} : Πιθανές τιμές είναι ακέραιοι αριθμοί
- {Object} : Ένα αντικείμενο με ζεύγη τιμών διαχωρισμένα με κόμματα, που περικλείονται σε αγκύλες {}. Για παράδειγμα {'απάντηση': 42, 'ερώτηση': null}
- {OpenLayers._____} : Ένα αντικείμενο που προέκυψε από μία κλάση OpenLayers. Στη θέση του κενού θα μπορούσα να είναι οποιαδήποτε κλάση του OpenLayer, όπως για παράδειγμα OpenLayers.Control.LayerSwitcher ({});
- {String}: Πιθανή τιμή είναι οποιαδήποτε συμβολοσειρά, που υποδεικνύεται με μονά ή διπλά εισαγωγικά.

Οι ιδιότητες των επιπέδων απεικόνισης είναι οι εξής :

- Events {OpenLayers.Events}: Ένα αντικείμενο συμβάντος καλεί μία συνάρτηση όταν συμβαίνει μία ενέργεια, όπως το zoom.
- Map {OpenLayers.Map} : Το αντικείμενο map στο οποίο ανήκει το συγκεκριμένο layer. Αυτό ορίζεται αυτόματα όταν ένα layer προστίθεται στον χάρτη μέσω της συνάρτησης setMap().
- IsBaseLayer {Boolean} : Προσδιορίζει εάν ένα layer πρέπει να ενεργεί ως base layer. Η προεπιλεγμένη τιμή είναι False.
- DisplayInLayerSwitcher {Boolean} : Προσδιορίζει εάν το layer πρέπει να εμφανίζεται στο layer switcher.
- Visibility {Boolean} : Προσδιορίζει εάν το layer είναι ορατό στον χάρτη και ενεργοποιημένο ή απενεργοποιημένο στον διακόπτη layer switcher.
- Attribution {string}: Κείμενο που εμφανίζεται όταν ο έλεγχος Attribution προστίθεται στον χάρτη. Η προεπιλογή είναι το κείμενο να εμφανίζεται κάτω δεξιά και τα στοιχεία κάθε layer χωρίζονται με κόμμα.
- InRange {Boolean} : Οι τιμές που παίρνει είναι είτε True είτε False ανάλογα με το αν η ανάλυση του τρέχοντα χάρτη είναι εντός της μέγιστης και ελάχιστης εμβέλειας του layer. Καλείται όταν αλλάζει η τιμή του zoom.
- Options {Object} : Προαιρετικό αντικείμενο του οποίου οι ιδιότητες θα ορισθούν στο layer. Οποιαδήποτε από τις ιδιότητες του layer μπορεί να οριστεί από το Options {Object}.
- EventListeners {Object} : Η ιδιότητα αυτή θα ενεργοποιηθεί εάν αυτό έχει ορισθεί κατά τη διάρκεια δημιουργίας του layer.
- Projection {OpenLayers.Projection}or{String}: Με την ιδιότητα αυτή αντικαθιστάται η προεπιλεγμένη προβολή του χάρτη. Επίσης μπορεί να χρειαστεί να οριστεί το μέγιστο εύρος της γεωγραφικής περιοχής (maxextend) και η μέγιστη ανάλυση σχεδίασης (maxresolution) του χάρτη, όπως επίσης και μονάδες μέτρησης.

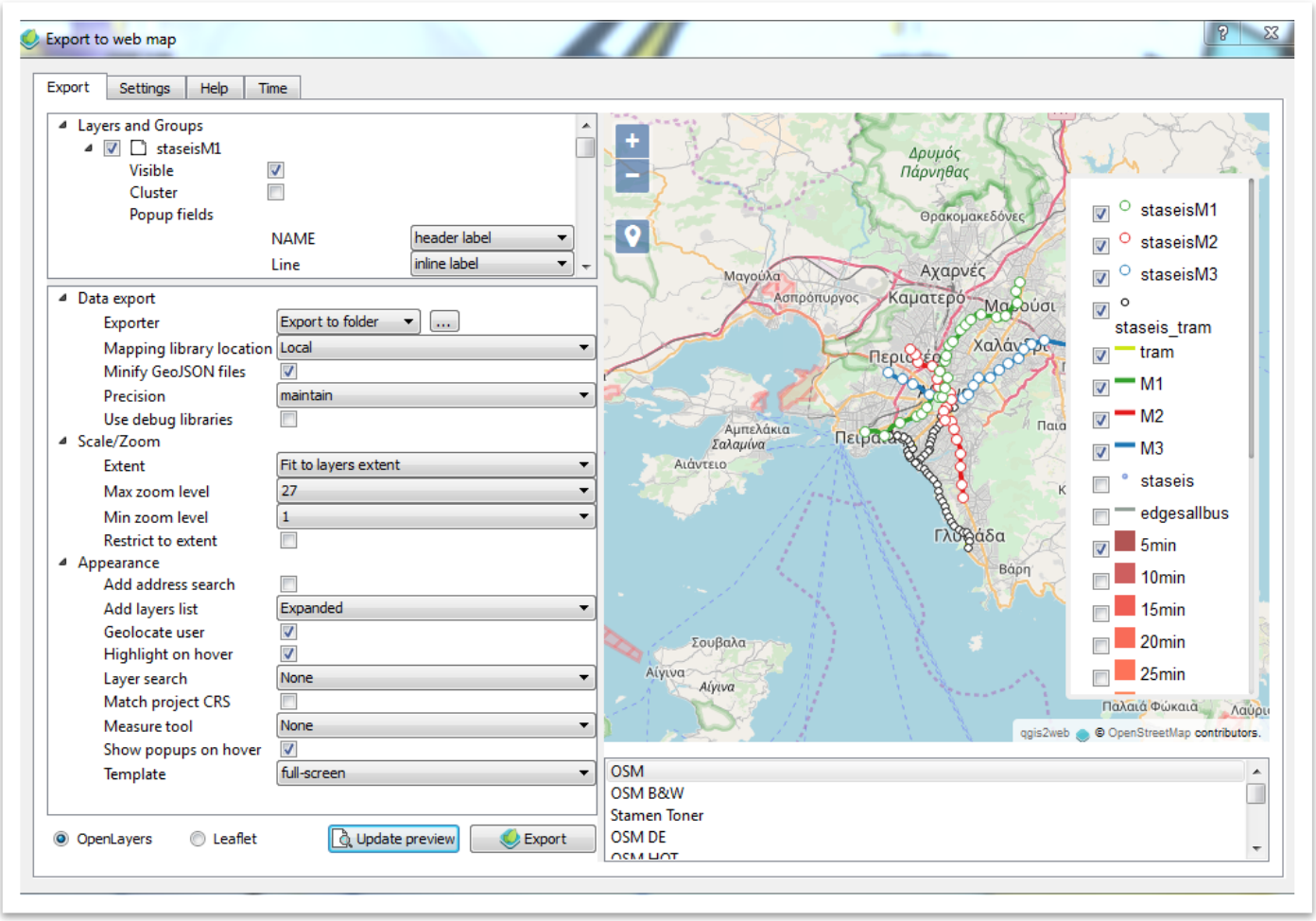
- Units {String}: Οι μονάδες μέτρησης του χάρτη μέσα στον οποίο βρίσκεται το layer. Πιθανές τιμές είναι ΄degrees', 'm', 'ft', 'km', 'mi' ή 'inches'.
- Scales {Array} : Περιέχει μια σειρά από τις κλίμακες του χάρτη, από την υψηλότερη προς τη χαμηλότερη τιμή. Οι μονάδες μέτρησης πρέπει να είναι ορισμένες όταν χρησιμοποιείται η συγκεκριμένη ιδιότητα. Συνήθως χρησιμοποιείται η ιδιότητα Resolution αντί για την Scales.
- Resolutions {Array} : Περιέχει μια σειρά των αναλύσεων του χάρτη (μονάδες μέτρησης χάρτη ανά εικονοστοιχείο) από την υψηλότερη προς τη χαμηλότερη τιμή. Εάν δεν ορισθεί, θα υπολογισθεί αυτόματα με βάση άλλες ιδιότητες όπως το maxExtent, maxResolution κλπ.
- maxExtent {OpenLayers.Bounds} : Ένα αντικείμενο OpenLayers.Bounds αποτελείται από ελάχιστο X, Y και μέγιστο X,Y που καθορίζουν τη γεωγραφική έκταση του layer. Οποιεσδήποτε συντεταγμένες εκτός αυτού του πλαισίου οριοθέτησης δεν θα εμφανιστούν.
- maxResolution {Float} : Ορίζει τη μέγιστη ανάλυση (το πλάτος ή ύψος στις μονάδες μέτρησης του χάρτη ανά εικονοστοιχείο). Το προεπικεγμένο μέγιστο είναι 360 μοίρες / 256 εικονοστοιχεία. Εάν δεν χρησιμοποιείται γεωγραφική προβολή μπορεί να ορισθεί διαφορετική τιμή.
- numZoomLevels {Float} : Καθορίζει το επίπεδο του zoom που έχει ένα layer. Εάν η τιμή του zoom είναι μεγαλύτερη από την τιμή αυτή, το layer δεν θα εμφανίζεται στον χάρτη.
- minScale {Float} : Καθορίζει την ελάχιστη τιμή της κλίμακας στην οποία το layer θα είναι ορατό.
- MaxScale {Float} : Καθορίζει τη μέγιστη τιμή της κλίμακας στην οποία το layer θα είναι ορατό.
- displayOutsideMaxExtent {Boolean} : Ορίζει εάν ο χάρτης πρέπει να στείλει αίτημα που είναι ολοκληρωτικά εκτός της ιδιότητας του layer maxExtent.
- wrapDateLine {Boolean} : Η ιδιότητα αυτή προκαλεί την περιτύλιξη του χάρτη γύρω από τη γραμμή ημερομηνίας. Αυτό επιτρέπει το συνεχές σύρσιμο του χάρτη αριστερά/ δεξιά, αφού ο χάρτης είναι τυλιγμένος γύρω από τον εαυτό του.
- transitionEffect {String} : Καθορίζει το εφέ μετάβασης όταν ο χάρτης μετακινείται ή μεγεθύνεται . Πιθανές τιμές είναι null και 'resize'.
- SUPPORTED_TRANSITIONS {Array}: Η ιδιότητα αυτή είναι σταθερή και δεν επιδέχεται αλλαγές. Περιέχει μια λίστα με υποστηρικτικά εφέ μετάβασης για χρήση με την ιδιότητα transitionEffect.
- Metadata {Object} : Επιτρέπει την αποθήκευση επιπλέον πληροφοριών για το layer. Δεν επηρεάζει τη συμπεριφορά του layer.

5.2.2. QGIS2Web

Η διαδικτυακή χαρτογραφία είναι ένα εξαιρετικό μέσο για τη δημοσιοποίηση δεδομένων GIS στον ιστό και τη δυνατότητα πρόσβασης τρίτων στους δημοσιοποιημένους χάρτες. Όπως φάνηκε η δημιουργία ενός διαδικτυακού χάρτη είναι πολύ διαφορετική διαδικασία από τη δημιουργία ενός σε GIS περιβάλλον. Για το λόγο αυτό έχουν αναπτυχθεί εργαλεία που μεταφράζουν χάρτες QGIS σε διαδικτυακούς χάρτες. Συγκεκριμένα το πρόσθετο εργαλείο (plugin) QGIS2Web έχει αναπτυχθεί ώστε οι χρήστες του QGIS να μπορούν να δημιουργήσουν εύκολα έναν διαδικτυακό χάρτη χρησιμοποιώντας το OpenLayers ή το Leaflet.

Οι δυνατότητες που δίνονται ακολουθώντας τη συγκεκριμένη διαδικασία, είναι πολύ βασικές και απαιτείται η περαιτέρω επεξεργασία του χάρτη για την παραγωγή ολοκληρωμένων εφαρμογών.

Συγκεκριμένα, όπως φαίνεται στην Εικόνα 40 αρχικά ορίζονται τα θεματικά επίπεδα layers που εντάσσονται στον χάρτη, η σειρά απεικόνισής τους και η μορφή των πληροφοριών που θα εμφανίζονται ως κύριος τίτλος ή ως ετικέτα γραμμής. Ύστερα επιλέγεται το εύρος της περιοχής (map extend) και το μέγιστο και ελάχιστο επίπεδο μεγέθυνσης (zoom). Επίσης υπάρχει η δυνατότητα δημιουργίας λίστας των layers, ώστε ο χρήστης να μπορεί να επιλέγει ποια επίπεδα θέλει να απεικονίζονται στον χάρτη, και η δυνατότητα γεωγραφικού εντοπισμού του χρήστη. Επιπλέον εισάγεται προαιρετικά η έμφαση (highlight) και η εμφάνιση των ιδιοτήτων (popup window) των επιλεγμένων αντικειμένων του χάρτη. Τέλος επιλέγεται ο χάρτης υποβάθρου (base layer) που επιθυμείται να εμφανίζεται στην εφαρμογή.



Εικόνα 40 : Επιλογές που παρέχονται από το εργαλείο QGIS2Web

5.2.3. Χρήση QGIS2Web στην παρούσα εφαρμογή

Για την παρούσα εφαρμογή χρησιμοποιήθηκε αρχικά η επέκταση QGIS2Web. Μία πολύ χρήσιμη ιδιότητα είναι ότι αποθηκεύεται η πηγή των θεματικών επιπέδων. Επίσης διατηρείται η σειρά απεικόνισης των επιπέδων σύμφωνα με τη σειρά απεικόνισής τους στο QGIS.

Τα θεματικά επίπεδα (layers) αποθηκεύτηκαν πρώτα σε μορφή GeoJSON, μέσω του QGIS. Συγκεκριμένα στην επιλογή Save as των θεματικών επιπέδων δίνεται η δυνατότητα αποθήκευσης αρχείων τύπου GeoJSON.

Ένα παράδειγμα κωδικοποίησης GeoJSON για σημειακά δεδομένα φαίνεται στην εικόνα. Επίσης δημιουργείται αυτόματα ένα αρχείο css με τη μορφοποίηση (style) του επιπέδου,

ώστε να παραμείνει έτσι και στον διαδικτυακό χάρτη. Περισσότερες πληροφορίες για την css παρέχονται στο κεφάλαιο 4.3.1.

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Εικόνα 41 : Παράδειγμα μορφής αρχείων GeoJSON

Για τα σημειακά δεδομένα που απεικονίζουν τις στάσεις των Μέσων Μαζικής Μεταφοράς της Αττικής, επιλέχθηκε να εμφανίζονται το όνομα της στάσης και τα μέσα που κάνουν στάση εκεί, όταν ο κέρσορας περνάει από πάνω.

Για τα γραμμικά δεδομένα, τα οποία απεικονίζουν τους άξονες κίνησης των Μέσων Μαζικής Μεταφοράς της Αττικής, επιλέχθηκε να εμφανίζονται οι γραμμές που την εξυπηρετούν καθώς και το συνολικό δρομολόγιο (Αφετηρία – Τέρμα) κάθε γραμμής.

Τέλος για τα επιφανειακά δεδομένα που δεν είναι άλλα από τις ισόχρονες καμπύλες, εμφανίζεται ο χρόνος που αντιπροσωπεύει κάθε μία, π.χ. 20 λεπτά.

Επίσης αξιοποιήθηκε η επιλογή δημιουργίας λίστας των layers ώστε η πληροφορία αυτή να χρησιμοποιηθεί αργότερα στην τελική μορφοποίηση της εφαρμογής.

Η μορφή της εφαρμογής σε αρχικό στάδιο είναι αυτή που φαίνεται στην Εικόνα 40. Για την περαιτέρω επεξεργασία της χρησιμοποιήθηκαν επιπλέον προγραμματιστικά εργαλεία που αναφέρονται κατόπιν.

Ως υπόβαθρο χρησιμοποιήθηκε το OpenStreetMap το οποίο διατίθεται δωρεάν υπό άδεια ελεύθερης χρήσης. Το OpenStreetMap [OSM] είναι ένα project που αποσκοπεί στη δημιουργία ενός δωρεάν επεξεργάσιμου χάρτη για ολόκληρο τον κόσμο. Οι χάρτες δημιουργούνται χρησιμοποιώντας δεδομένα από κινητές συσκευές GPS, αεροφωτογραφίες και άλλες δωρεάν πηγές. Κάθε χρήστης μπορεί να βοηθήσει στη βελτίωση των χαρτών με προσθήκες ή διορθώσεις. Το σημαντικό στοιχείο του project και αυτό που κάνει τη διαφορά είναι ότι όλα τα δεδομένα των χαρτών παρέχονται δωρεάν για ανάκτηση.

5.3. Διαμόρφωση Ιστοσελίδας της Εφαρμογής

Τα περιεχόμενα της ιστοσελίδας καθορίστηκαν σύμφωνα με τις ανάγκες της σχεδιαζόμενης εφαρμογής. Η δόμηση και η μορφοποίηση των στοιχείων της υλοποιήθηκε σε κώδικα HTML.

Η *HTML* (Hypertext Markup Language) είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες, και τα στοιχεία της είναι τα βασικά δομικά στοιχεία αυτών. Πρόκειται για μία ειδική γλώσσα που αναγνωρίζεται από τους δημοφιλέστερους φυλλομετρητές (web browsers), όπως το Ms Internet Explorer, Google Chrome, Netscape, Firefox, κα., για την παρουσίαση κειμένων και εικόνων. Αποτελεί υποσύνολο ενός προτύπου ISO, της γλώσσας *SGML* (Standard Generalized Markup Language) (www.w3org/html/).

Τα κείμενα HTML συχνά καλούνται και ιστοσελίδες (web pages) και περιέχουν μια σειρά από στοιχεία ή ετικέτες (tags), που φωλιάζουν και μορφοποιούν το περιεχόμενο. Για παράδειγμα η ετικέτα <*p*> αντιστοιχεί στην εκκίνηση μιας παραγράφου, ενώ η </*p*> στο τέλος αυτής (www.w3org/html/).

Ένα κείμενο HTML φωλιάζεται σε μία ετικέτα <*html*>. Συνήθως έχει δύο τμήματα, την κεφαλίδα του κειμένου, που φωλιάζεται στην ετικέτα <*head*> και το κυρίως σώμα του κειμένου που φωλιάζεται στην ετικέτα <*body*>. Η ετικέτα <*head*> συνήθως περιλαμβάνει τον τίτλο του κειμένου, που φωλιάζεται σε μία ετικέτα <*title*>. Ο τίτλος του κειμένου παρουσιάζεται στην κεφαλίδα (πλαίσιο) του φυλλομετρητή. Το κυρίως σώμα <body> περιλαμβάνει το περιεχόμενο του κειμένου που θα μορφοποιηθεί στον φυλλομετρητή.

Κατ' αντιστοιχία με τους επεξεργαστές κειμένου ένα κείμενο HTML έχει έξι επίπεδα κεφαλίδων. Το πρώτο (και πλέον σημαντικό) φωλιάζεται στην ετικέτα <h1>, το δεύτερο στην ετικέτα <h2> και το έκτο στην ετικέτα <h6>. Αν πρέπει να το κείμενο να οργανωθεί σε παραγράφους, διαχωρίζεται με την ετικέτα <p>, ενώ η ετικέτα φωλιάζει το κείμενο που εμφανίζεται τονισμένο (emphasized) κατά τη μορφοποίησή του στον φυλλομετρητή.

Στην Εικόνα 42 φαίνεται ένα απόσπασμα του κειμένου HTML, με κάποιες από τις ετικέτες που αναφέρθηκαν.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Example<title>
5          <link rel="stylesheet" href="styl
6      </head>
7      <body>
8          <h1>
9              <a href="/">Header</a>
10         </h1>
11         <nav>
12             <a href="one/">One</a>
13             <a href="two/">Two</a>
14             <a href="three/">Three</a>
15         </nav>
```

Εικόνα 42 : Παράδειγμα HTML κειμένου όπου φαίνεται η δομή της

Η δημιουργία ενός κειμένου HTML μπορεί να γίνει σε έναν απλό κειμενογράφο (π.χ. MS NodePad) ή χρησιμοποιώντας έναν κειμενογράφο HTML (HTML editor). Οι ετικέτες ενός κειμενογράφου αναγνωρίζονται από τον φυλλομετρητή, ο οποίος αναλαμβάνει και τη μορφοποίηση του κειμένου στην αντίστοιχη διεπαφή (παράθυρο) στην οθόνη του Η/Υ. Ένα κείμενο HTML έχει όνομα με επέκταση html.

Η συγκεκριμένη γλώσσα δίνει τη δυνατότητα ορισμού περιοχών στις εικόνες και την ανάθεση σε αυτές υπερσυνδέσεων σε άλλες σελίδες ή αρχεία στον Παγκόσμιο Ιστό. Οι περιοχές αυτές καλούνται ενεργές περιοχές και αντιστοιχούν σε απλούς γεωμετρικούς τύπους αντικειμένων δύο διαστάσεων, που σχεδιάζονται επί των εικόνων.

Οι γεωμετρικοί τύποι που υποστηρίζονται είναι το ορθογώνιο παραλληλόγραμμο (*rect*), ο κύκλος (*circle*) και το πολύγωνο (*poly*). Τα γεωμετρικά αντικείμενα έχουν συντεταγμένες εικόνας (εκφρασμένες σε εικονοστοιχεία – pixels) και οργανώνονται σε ένα επίθεμα (map), το οποίο έχει ένα μοναδικό όνομα (name) και επιτίθεται στην εικόνα. Οι συντεταγμένες των γεωμετρικών αντικειμένων οργανώνονται στην παράμετρο *coords*, η οποία έχει συγκεκριμένη δομή, ανάλογα με τον γεωμετρικό τύπο.

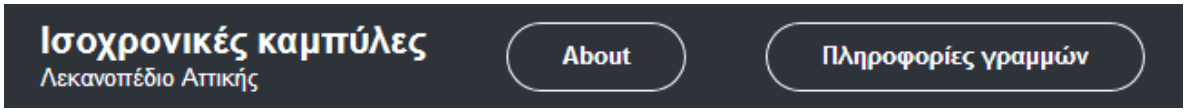
Αν δύο οι περισσότερα αντικείμενα επικαλύπτονται χωρικά στο επίθεμα, προτεραιότητα στην επικαλυπτόμενη περιοχή έχει το αντικείμενο (κι ο αντίστοιχος υπερσύνδεσμος), που εμφανίζεται πρώτο στο κείμενο HTML (Musciano C. & Kennedy B, 2006).

Στα πλαίσια υλοποίησης της παρούσας εφαρμογής και με τη βοήθεια HTML κώδικα, ορίστηκαν η θέση και τα χαρακτηριστικά του τίτλου της ιστοσελίδας, η θέση και το

μέγεθος του διαδικτυακού χάρτη και η εισαγωγή των διαφόρων πλήκτρων και κουμπιών μέσω των οποίων ο χρήστης αλληλεπιδρά με την ιστοσελίδα.

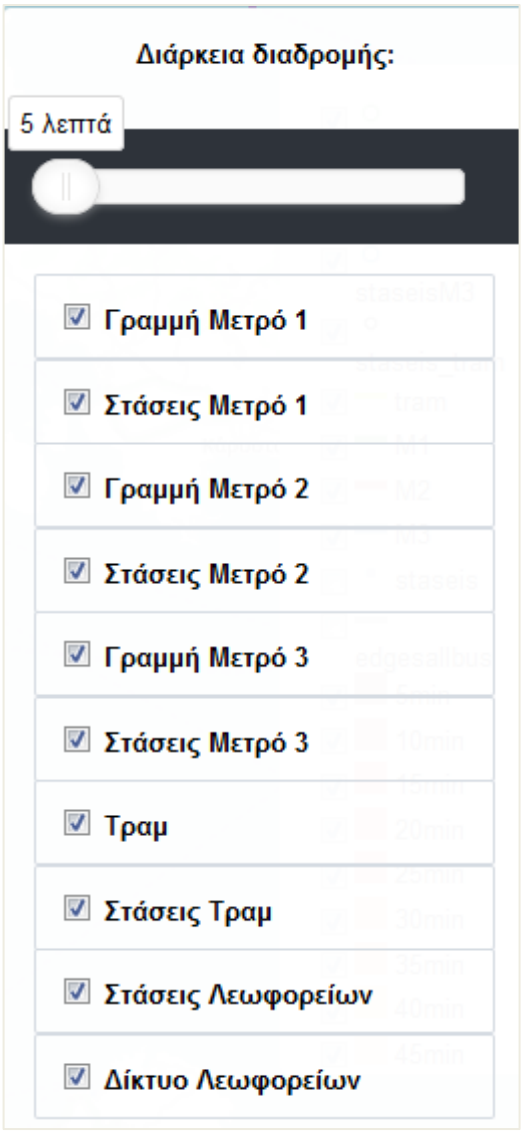
Αρχικά ορίσθηκε η επικεφαλίδα της εφαρμογής που περιλαμβάνει τον τίτλο και τον υπότιτλο.

Επιπλέον στην επικεφαλίδα εισήχθησαν και κάποιες επιλογές ώστε να παρέχονται στο χρήστη κάποιες επιπλέον πληροφορίες. Συγκεκριμένα τοποθετήθηκε μία επιλογή <About> μέσω της οποίας εμφανίζονται τα πλαίσια υλοποίησης της εργασίας, οι δημιουργοί κλπ. Δίπλα από την επιλογή αυτή εμφανίζεται και η εξής : <Πληροφορίες γραμμών>. Όταν ο χρήστης την επιλέξει μεταβαίνει αυτόματα στην ιστοσελίδα του ΟΑΣΑ (www.oasa.gr) όπου μπορεί να αναζητήσει πληροφορίες για όλες τις γραμμές αλλά και λειτουργίες δρομολόγησης (Εικόνα 43).



Εικόνα 43 : Επικεφαλίδα διαδικτυακής εφαρμογής

Το βασικό εργαλείο που περιλαμβάνει η ιστοσελίδα, είναι μία ροδέλα στην οποία ο χρήστης μπορεί να ορίζει τον χρόνο μετακίνησης, με αφετηρία την Ομόνοια (Εικόνα 44). Η δυνατότητα που παρέχεται είναι από πέντε έως ενενήντα λεπτά με βήμα πέντε λεπτών. Επίσης υπάρχουν πλαίσια ελέγχου (checkboxes) για επιλογή των θεματικών επιπέδων, όπως οι γραμμές του μετρό, το δίκτυο των λεωφορείων, οι στάσεις των Μέσων Μαζικής Μεταφοράς της Αττικής κ.λπ. (Εικόνα 44). Επιπλέον εισήχθησαν εικονίδια μεγέθυνσης και σμίκρυνσης του χάρτη και εντοπισμού της γεωγραφικής θέσης χρήστη. Εδώ πρέπει να σημειωθεί ότι μέσω της HTML οι οντότητες αυτές είναι απλά εικόνες χωρίς καμία ιδιότητα. Οι ιδιότητές τους ορίζονται μέσω JavaScript και αναλύονται σε επόμενο εδάφιο.



Εικόνα 44 : Ροδέλα ορισμού χρόνου μετακίνησης και πλαίσια ελέγχου θεματικών επιπέδων

Ο κώδικας HTML που δομήθηκε στα πλαίσια της παρούσας εφαρμογής βρίσκεται στο Παράρτημα, μαζί με ορισμένες επεξηγήσεις. Εντός του HTML κώδικα, περιλαμβάνονται τμήματα κώδικα css και Javascript τα οποία καθορίζουν τη μορφοποίηση της ιστοσελίδας και τις λειτουργίες που υλοποιούνται σ' αυτή και τον χάρτη, οι βασικότερες των οποίων περιγράφονται στα εδάφια που ακολουθούν.

5.4. Μορφοποίηση και λειτουργίες ιστοσελίδας

Στο εδάφιο που προηγήθηκε, έγινε η περιγραφή των διαδικασιών δόμησης των στοιχείων που συνθέτουν την εμφάνιση της σχεδιαζόμενης ιστοσελίδας και υλοποιούνται μέσω κώδικα HTML. Παράλληλα με τη δόμηση της ιστοσελίδας, έλαβαν χώρα και ορισμένες διαδικασίες οι οποίες σχετίζονται με τη μορφοποίηση και τις λειτουργίες που πρόκειται να υλοποιούνται στο διαδικτυακό τόπο που δημιουργήθηκε.

5.4.1. Μορφοποίηση μέσω css

Η CSS (Cascading Style Sheets-Διαδοχικά Φύλλα Στυλ) είναι μια γλώσσα υπολογιστή που χρησιμοποιείται για τον έλεγχο της εμφάνισης ενός εγγράφου που έχει γραφτεί με μια γλώσσα σήμανσης, όπως η HTML. Για μια όμορφη και καλοσχεδιασμένη ιστοσελίδα η χρήση της CSS κρίνεται ως απαραίτητη. Γράφοντας ιστοσελίδες μόνο με HTML κώδικα, μπορούν να ορισθούν το χρώμα και το μέγεθος του κειμένου αλλά και άλλων στοιχείων της σελίδας (όπως πίνακες, links, λίστες κτλ). Για να αλλάξει το χρώμα κάποιου κειμένου ή κάποιου πίνακα, θα πρέπει να βρεθεί το χρώμα αυτό μέσα στον κώδικα και να γίνει η μετατροπή. Η διαδικασία αυτή μπορεί να είναι εύκολη κατά τη διαχείριση μιας μόνο σελίδας, αλλά μία ιστοσελίδα μπορεί να αποτελείται από δεκάδες σελίδες οι οποίες χρήζουν εύκολης και γρήγορης διαχείρισης.

Με την χρήση CSS είναι δυνατό να ορισθούν χρώματα και μεγέθη οργανωμένα σε στυλ και έπειτα να εφαρμοσθούν τα στυλ αυτά στα στοιχεία των σελίδων μίας ιστοσελίδας. Με αυτόν τον τρόπο, κάθε φορά που αλλάζει το χρώμα ενός στυλ, αλλάζει το χρώμα όλων των στοιχείων που έχουν αναφορά στο στυλ αυτό.

Εκτός όμως από την ευκολία στην διαχείριση μίας ιστοσελίδας, ένα άλλο σημαντικό πλεονέκτημα της χρήσης CSS στις σελίδες είναι ο "καθαρότερος" κώδικας, χωρίς πολλές ιδιότητες στις ετικέτες οι οποίες τον κάνουν δυσανάγνωστο. Επιπλέον κάνει γρηγορότερη την πλοήγηση καθώς το αρχείο, μέσα στο οποίο ορίζονται τα στυλ, "διαβάζεται" από τον browser μόνο μια φορά και έπειτα αποθηκεύεται στην cache memory, μειώνοντας έτσι το μέγεθος της πληροφορίας που κατεβάζεται από τους φυλλομετρητές (web browsers).[22]

Ο πιο ενδεδειγμένος τρόπος να χρησιμοποιηθεί η css σε ένα HTML κείμενο, είναι να συμπεριληφθεί ένα εξωτερικό αρχείο css, το οποίο καλείται στο κυρίως κείμενο HTML και η ιστοσελίδα το διαβάζει και το εκτελεί.

Για της ανάγκες της παρούσας εργασίας και της τελικής ιστοσελίδας δομήθηκε ένα αρχείο css, με ονομασία custom.css, στο οποίο ορίζεται το χρώμα του φόντο, η μορφή των κουμπιών και εργαλείων που χρησιμοποιεί ο χρήστης, αλλά και των περιγραφικών

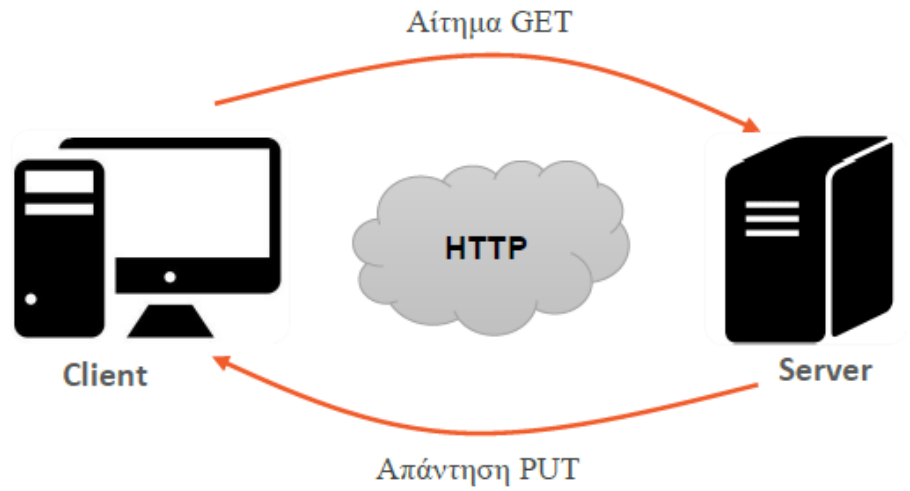
πληροφοριών που παρέχονται. Τέλος ορίστηκε το μέγεθος του χάρτη υποβάθρου και η θέση του στην ιστοσελίδα. Στην Εικόνα 45 φαίνεται ένα απόσπασμα κώδικα css.

```
.c-settings-panel {
  width: 260px;
  max-width: 100%;
  border-radius: 3px;
  background-color: rgba(255,255,255,.98);
  box-shadow: 0 0 0 1px rgba(0,0,0,.04), 0 1px 5px rgba(0,0,0,.1);

  position: absolute;
  right: 35px;
  top: 80px;
  z-index: 100;
}
```

Εικόνα 45 : Απόσπασμα του κώδικα css που δημιουργήθηκε για τη μορφοποίηση της Ιστοσελίδας

5.4.2. Ορισμός λειτουργιών μέσω JavaScript

Η ανάγκη παράκαμψης της στατικότητας του περιεχομένου της HTML, οδήγησε στην ανάπτυξη της φιλοσοφίας του προγραμματισμού από την πλευρά του πελάτη (Εικόνα 46). Κατά τον προγραμματισμό από την πλευρά του πελάτη, η σελίδα HTML εμπλουτίζεται με κατάλληλο κώδικα γραμμένο σε γλώσσα προγραμματισμού, ο οποίος εκτελείται από τον Η/Υ του πελάτη, επιτρέποντας τόσο το δυναμικό χειρισμό συμβάντων κατά τη διάρκεια της εμφάνισης της σελίδας, όσο και την αύξηση της διαδραστικότητας μεταξύ του χρήστη και της εφαρμογής. Οι γλώσσες προγραμματισμού που χρησιμοποιούνται στον προγραμματισμό από την πλευρά του πελάτη, ονομάζονται γλώσσες συγγραφής σεναρίων (scripting languages) και δίνουν τη δυνατότητα επεξεργασίας των δεδομένων της ιστοσελίδας, καθώς και τη διεξαγωγή υπολογισμών και μετασχηματισμών σε αυτά τα δεδομένα, με τη βοήθεια εντολών, συναρτήσεων, αντικειμένων και συμβάντων. Σημαντικότερος εκπρόσωπος των γλωσσών συγγραφής σεναρίων είναι η γλώσσα Javascript (Παναγιωτόπουλος Π. Ιωάννης – Χρήστος, 2003).



Εικόνα 46: Μοντέλο πελάτη – εξυπηρετητή

Η JavaScript (JS) είναι διερμηνευόμενη (interpreted) γλώσσα προγραμματισμού για διαδικτυακές εφαρμογές. Αρχικά αποτέλεσε μέρος της υλοποίησης των φυλλομετρητών Ιστού, ώστε τα σενάρια από την πλευρά του πελάτη (client-side scripts) να μπορούν να επικοινωνούν με τον χρήστη, να ανταλλάσσουν δεδομένα ασύγχρονα και να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται. Πρόκειται για γλώσσα σεναρίων που βασίζεται στα πρωτότυπα (prototype-based), είναι δυναμική, με ασθενείς τύπους και έχει συναρτήσεις ως αντικείμενα πρώτης τάξης. Η σύνταξή της είναι επηρεασμένη από τη C. Η JavaScript αντιγράφει πολλά ονόματα και συμβάσεις ονοματοδοσίας από τη Java, αλλά γενικά οι δύο αυτές γλώσσες δε σχετίζονται και έχουν πολύ διαφορετική σημασιολογία.. Η JavaScript χρησιμοποιείται και σε εφαρμογές εκτός ιστοσελίδων. Τέτοια παραδείγματα είναι τα έγγραφα PDF, οι εξειδικευμένοι browsers (site-specific browsers) και οι μικρές εφαρμογές της επιφάνειας εργασίας (desktop widgets).

Οι εφαρμογές της Javascript στο δυναμικό προγραμματισμό παγκόσμιου ιστού είναι πάρα πολλές. Η Javascript ακολουθεί τη φιλοσοφία των γλωσσών δομημένου προγραμματισμού, και βασίζεται στην υλοποίηση αντικειμένων (objects), γεγονότων (events) και μεθόδων (methods), η χρήση των οποίων συνιστά τον κύριο μηχανισμό αλληλεπίδρασης της Javascript με τις ιστοσελίδες HTML.

Ο κώδικας της JavaScript γράφεται σε καθαρό κείμενο (ASCII μορφή) και ενσωματώνεται μέσα στον κώδικα της HTML, μπορεί δε να εκτελεσθεί αμέσως με το φόρτωμα της σελίδας ή όταν λαμβάνει χώρα ένα συμβάν (event), όπως η πίεση ενός πλήκτρου του ποντικιού ή η τοποθέτηση του ποντικιού πάνω σε ένα αντικείμενο (Marijin Haverbeke, 2011).

Το σενάριο Javascript ενσωματώνεται στην ιστοσελίδα, ανάμεσα στις ετικέτες <SCRIPT> και </SCRIPT>. Μέσα σε ένα αρχείο HTML μπορούν να υλοποιηθούν πολλά σενάρια

JavaScript, χρησιμοποιώντας πολλαπλές ετικέτες SCRIPT. Τα σενάρια JavaScript δεν είναι υποχρεωτικά ενσωματωμένα στη σελίδα HTML. Αντίθετα, μπορούν να γραφούν ως ξεχωριστά αρχεία με κατάληξη .js, τα οποία καλούνται και εκτελούνται από τη σελίδα HTML με τη βοήθεια σχετικής αναφοράς. Σε αυτή την περίπτωση, η ετικέτα SCRIPT συνοδεύεται από την παράμετρο SRC, η οποία χρησιμοποιείται για τη δήλωση του εξωτερικού αρχείου στον κώδικα HTML (Marijin Haverbeke, 2011).

Η γλώσσα JavaScript βασίζεται στην έννοια του αντικειμένου. Το αντικείμενο είναι μια οντότητα η οποία κατέχει ένα σύνολο από ιδιότητες, οι οποίες μπορούν να τροποποιηθούν με κατάλληλες μεθόδους. Η JavaScript χρησιμοποιεί την εξής σύνταξη για να αναφερθεί σε κάποια ιδιότητα:

< όνομα αντικειμένου >.< ιδιότητα >

Η έννοια του αντικειμένου είναι στενά συνδεδεμένη και με την έννοια της μεθόδου. Οι μέθοδοι είναι συναρτήσεις ή διαδικασίες που χρησιμοποιούνται για να εκτελούν μια λειτουργία σ' ένα αντικείμενο. Η κλήση και εκτέλεση των μεθόδων των αντικειμένων γίνεται με τη δήλωση αντικείμενο.μέθοδος().

Τα χειριστήρια συμβάντος (event handlers) είναι παράμετροι που επιτρέπουν τη συσχέτιση του κώδικα JavaScript με συγκεκριμένα γεγονότα που προκύπτουν από τις κινήσεις του φυλλομετρητή ή του χρήστη. Παρακολουθώντας τις ενέργειες του χρήστη, η JavaScript μπορεί να εκτελεί σενάρια στο Η/Υ του πελάτη, χωρίς να χρειασθεί να μεσολαβήσει ο εξυπηρετητής. Οι ενέργειες του χρήστη σε μια ιστοσελίδα οι οποίες καλύπτονται από τα χειριστήρια συμβάντων, σχετίζονται με την κίνηση του ποντικιού και τις καταχωρήσεις στα στοιχεία (πεδία) των φορμών. Τα κυριότερα χειριστήρια συμβάντων της JavaScript συνοψίζονται στον Πίνακα 10.

Πίνακας 10 : JavaScript Event Handlers

Χειριστήριο Συμβάντος	Κριτήριο ενεργοποίησης
onBlur	Λαμβάνει χώρα όταν χάνεται η εστίαση (lost focus), δηλαδή όταν απομακρυνόμαστε είτε με κλικ με το ποντίκι είτε πατώντας το πλήκτρο tab σ' ένα πεδίο κειμένου ή σε μια περιοχή κειμένου ή και σ' ένα πλαίσιο λίστας (select) μιας φόρμας.
onChange	Λαμβάνει χώρα όταν τροποποιείται το περιεχόμενο ενός πεδίου κειμένου ή μιας περιοχής κειμένου ή και ενός πλαισίου λίστας μιας φόρμας, πριν ακόμα απομακρυνθούμε από το στοιχείο αυτό.
onClick	Λαμβάνει χώρα όταν κάνουμε κλικ με το ποντίκι σ' ένα αντικείμενο, όπως είναι για παράδειγμα ένα πλήκτρο εντολής (button) ή ένα πλαίσιο ελέγχου (check box)
onFocus	Λαμβάνει χώρα όταν κάνουμε εστίαση (focus) μέσα σ' ένα πεδίο μιας φόρμας.
onLoad	Λαμβάνει χώρα όταν τελειώσει το φόρτωμα μιας ιστοσελίδας σ' ένα παράθυρο ή όταν τελειώσει το φόρτωμα όλων των πλαισίων (frames) που βρίσκονται μέσα σε μια σελίδα.
onMouseOver	Λαμβάνει χώρα όταν ο δείκτης του ποντικιού τοποθετείται πάνω από ένα αντικείμενο ή από ένα σύνδεσμο.
onSelect	Λαμβάνει χώρα όταν επιλέγουμε ένα μέρος ή όλο το κείμενο ενός πεδίου κειμένου (text field) ή μιας περιοχής κειμένου (textarea).
onSubmit	Λαμβάνει χώρα όταν κάνουμε κλικ με το ποντίκι σ' ένα πλήκτρο υποβολής (submit button) για να υποβάλλουμε μια φόρμα.
onUnload	Λαμβάνει χώρα όταν φεύγουμε από ένα έγγραφο (ιστοσελίδα).

Όσον αφορά την παρούσα εφαρμογή τα αντικείμενα που εισάγονται επιτρέπουν στο χρήστη να αλληλεπιδρά με τον χάρτη, μέσω πλαισίων ελέγχου (checkboxes). Επίσης δίνεται η δυνατότητα προβολής επιπλέον πληροφοριών, όπως η κλίμακα ή τα περιγραφικά χαρακτηριστικά οντοτήτων του χάρτη.

Οι βασική λειτουργία της ιστοσελίδα αφορά στην απεικόνιση των επιλεγμένων ισοχρονικών καμπυλών με αφετηρία την πλατεία της Ομόνοιας, στο χρόνο που ορίζει ο χρήστης. Στην εφαρμογή έχουν εισαχθεί και το δίκτυο μέσων μαζικής μεταφορά της περιοχής μελέτης με τις αντίστοιχες στάσεις. Συνεπώς μέσω γεγονότων JavaScript δίνεται η δυνατότητα της εύρεσης των μέσων που εξυπηρετεί κάθε περιοχή και των στάσεων που υπάρχουν εκεί.

Στη συνέχεια, παρουσιάζονται οι συγκεκριμένες λειτουργίες που υλοποιούνται στην ιστοσελίδα ενώ ο κώδικας της εφαρμογής βρίσκεται στο Παράρτημα.

Αρχικά πρέπει να αναφερθεί ότι οι καμπύλες χρονοαπόστασης είναι ήδη υπολογισμένες και αποθηκευμένες στη βάση δεδομένων της ιστοσελίδας και καλούνται κάθε φορά που ο χρήστης επιλέγει κάποια ή κάποιες από αυτές. Το ίδιο συμβαίνει και με τα υπόλοιπα επίπεδα που εμφανίζονται στον χάρτη της ιστοσελίδας εκτός του υποβάθρου. Για το λόγο αυτό ήταν απαραίτητο τα δεδομένα να δομηθούν σε μορφή GeoJSON. Ο τύπος δεδομένων GeoJSON είναι σχεδιασμένος για να αντιπροσωπεύει απλά γεωγραφικά αρχεία, μαζί με τα μη χωρικά χαρακτηριστικά τους και βασίζεται στην JavaScript Notation Object.

Όταν δημιουργείτε ένας χάρτης μέσω του OpenLayers δημιουργούνται αυτόματα τέσσερις οντότητες, οι οποίες είναι οι εξής:

- OpenLayers.Control.Navigation: Ορίζει την αλληλεπίδραση του χάρτη από τα συμβάντα του ποντικιού, όπως κύλιση του ποντικιού (pan), μεγέθυνση του χάρτη μέσω της γραμμής κύλισης (scroll bar), διπλό κλικ κλπ.

- OpenLayers.Control.PanZoom: Εισάγει μία μπάρα Κύλισης/Μεγέθυνσης (Pan/Zoom) στο πάνω αριστερό άκρο του χάρτη ή δύο πλήκτρα για Μεγέθυνση (Zoom In) και σμίκρυνση (Zoom Out) αντίστοιχα στο ίδιο σημείο.

- OpenLayers.Control.ArgParser : Επιτρέπει στον χάρτη να κάνει μεγέθυνση σε μία συγκεκριμένη τοποθεσία και ενεργοποιεί η απενεργοποιεί τα επίπεδα πληροφορίας (layers) βάσει της διεύθυνσης URL με την οποία καλείται ο χάρτης.

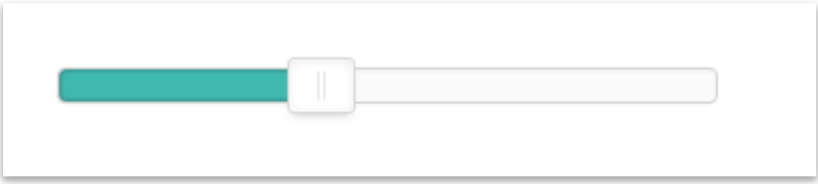
- OpenLayers.Control.Attribution: Η οντότητα αυτή προσθέτει πιθανές εξαιρέσεις από τα επίπεδα πληροφοριών του χάρτη, οι οποίες στη συνέχεια μεταβιβάζονται σε αυτά.

Στην παρούσα εργασία διατηρήθηκαν μόνο οι τρεις πρώτες οντότητες και προστέθηκαν κάποιες επιπλέον που αναλύονται κατόπιν.

- noUiSlider : Είναι μια JavaScript οντότητα μέσω της οποίας ο χρήστης μπορεί να ρυθμίσει το εύρος τιμών που επιθυμεί. Πρόκειται για μια ελαφριά προσθήκη που είναι συμβατή με όλους τους browsers ενώ λειτουργεί χωρίς πρόβλημα και σε κινητά με λειτουργία αφής. Ο προγραμματιστής ορίζει τη μέγιστη και την ελάχιστη τιμή εύρους. Επίσης μπορούν να ορισθούν συνεχείς ή διακριτές τιμές ανάλογα με τις ανάγκες της εκάστοτε εφαρμογής[23]. Για τις ανάγκες την παρούσας εργασίας ορίστηκε το κατώφλι (min) στα πέντε λεπτά, το ταβάνι (max) στα ενενήντα και το βήμα στα πέντε λεπτά. Αυτό πρακτικά σημαίνει ότι όσο ο χρήστης μετακινεί τον κέρσορα προς τα δεξιά, η τιμή του χρόνου αυξάνεται ανά πέντε λεπτά και ταυτόχρονα εμφανίζονται οι αντίστοιχες ισόχρονες στον χάρτη. Επίσης ορίσθηκε η θέση του κέρσορα κατά την έναρξη (φόρτωση) της ιστοσελίδας να είναι στα πέντε λεπτά. Στην Εικόνα 47 φαίνεται ένα παράδειγμα του κώδικα σε JavaScript που ορίζεται ένας ρυθμιστής εύρους (range slider) και στην Εικόνα 48 η μορφή του σε περιβάλλον ιστοσελίδας.

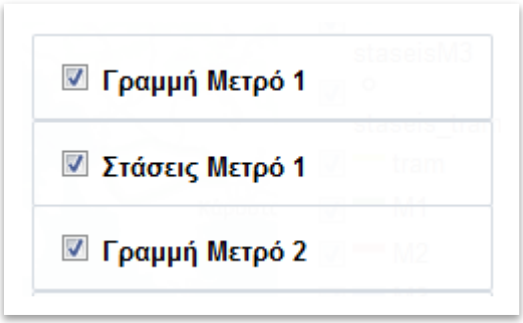
```
var connectSlider = document.getElementById('slider-connect');

noUiSlider.create(connectSlider, {
    start: 40,
    connect: [true, false],
    range: {
      'min': 0,
      'max': 100
    }
});
```

Εικόνα 47



Εικόνα 48

- Checkboxes : Δημιουργήθηκαν κάποιες επιλογής θέασης τον επιπέδων που έχουν εισαχθεί στον χάρτη μέσω πλαισίων ελέγχου όπως αναφέρθηκε στο κεφάλαιο 5.3. Ουσιαστικά κάθε «κουτί» αντιστοιχεί στο επίπεδο πληροφορίας που αναφέρεται στον τίτλο του και με το κλικ του ποντικιού (on click) εμφανίζεται ή εξαφανίζεται από τον χάρτη αντίστοιχα.



Εικόνα 49 : Πλάισια ελέγχου θεματικών επιπέδων

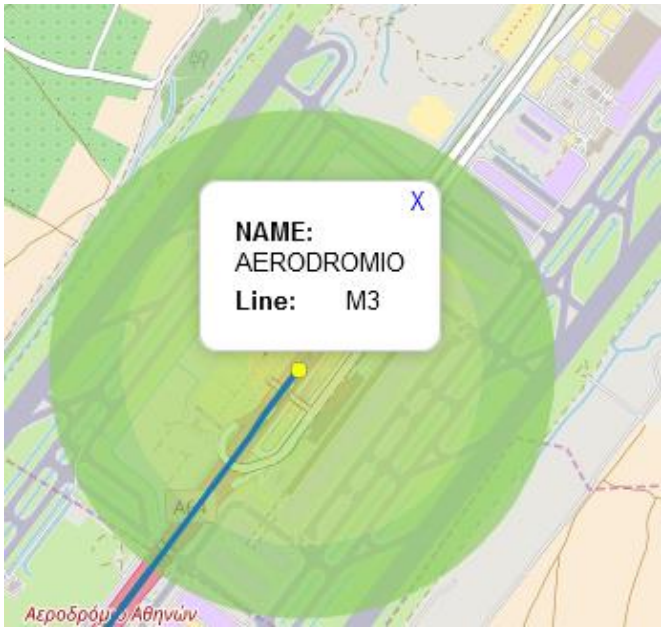
- Scale-line : Εισήχθη στον χάρτη μία γραφική κλίμακα η οποία αλληλεπιδρά με τις αλλαγές στο επίπεδο μεγέθυνσης (zoom) και πληροφορεί για την τρέχουσα κλίμακα. Δίνεται ελεύθερα από τα παραδείγματα που προσφέρει το OpenLayers και ενσωματώνεται στην εκάστοτε εφαρμογή σύμφωνα με τις οδηγίες (Εικόνα 50).

Εικόνα 50

Επιπλέον χρησιμοποιήθηκαν κάποιες λειτουργίες ώστε ο χρήστης να μπορεί να εκμεταλλευτεί όλες τις πληροφορίες που παρέχονται από τα layers.

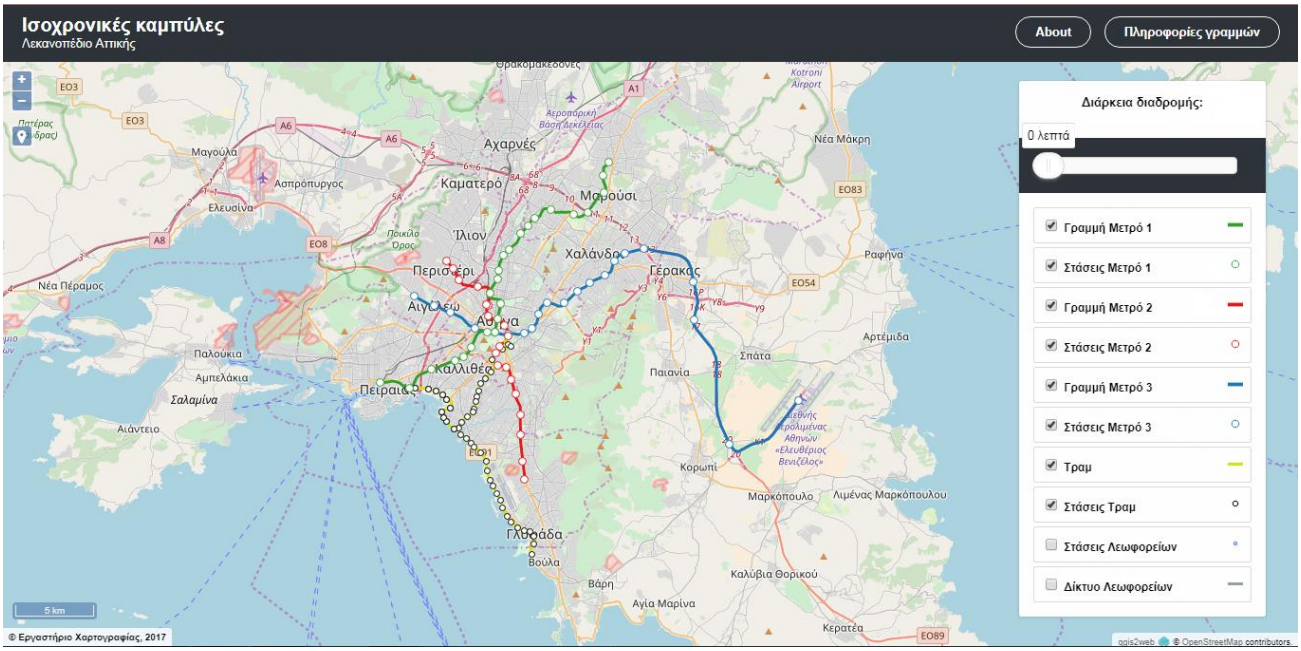
- Var FeatureOverlay : Πρόκειται για ένα μηχανισμό προσωρινής αλλαγής της απεικόνισης ενός στοιχείου του χάρτη. Εισάγεται ώστε να αναγνωρίζει πότε ο κέρσορας βρίσκεται πάνω από κάποιο επίπεδο.
- Var Highlight : Χρησιμοποιήθηκε ώστε όταν ο κέρσορας περνάει πάνω από κάποιο επίπεδο πληροφορίας (onMouseOver) αυτό να αλλάζει προσωρινά χρώμα (να υπογραμμίζεται). Το χρώμα που επιλέχθηκε είναι το κίτρινο.

- Var Hover : Με αυτή την ιδιότητα όταν ο κέρσορας περνάει πάνω από ένα γραμμικό σύμβολο εμφανίζεται ένα παράθυρο (pop up window) που αναγράφει τα χαρακτηριστικά του. Για παράδειγμα στην Εικόνα 51 ο κέρσορας βρίσκεται πάνω από το σημείο που απεικονίζει τη στάση του Μετρό Γραμμής 3 με όνομα Αεροδρόμιο.



Εικόνα 51 : Αλλαγή χρωματισμού και εμφάνιση παραθύρου ιδιοτήτων με την κίνηση του κέρσορα πάνω από το σημείο

Η τελική μορφή της εφαρμογής όταν φορτώνει για πρώτη φορά η ιστοσελίδα φαίνεται στην Εικόνα 52.

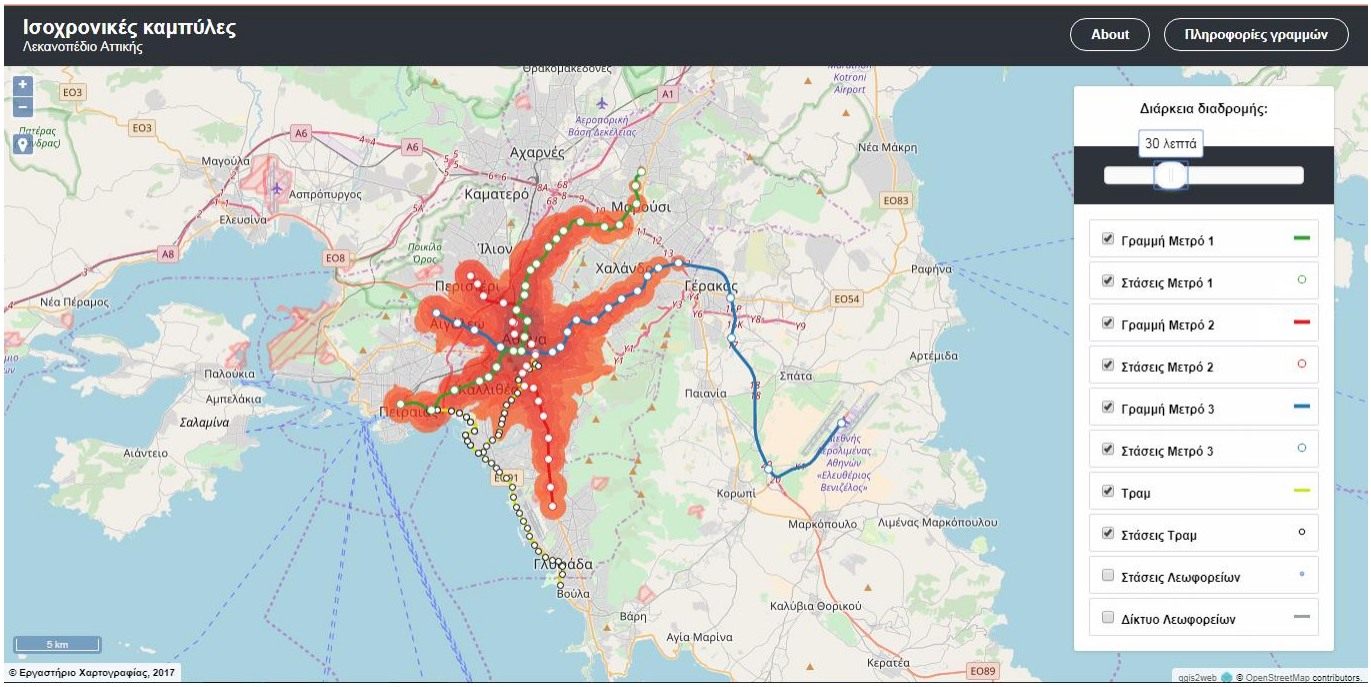


Εικόνα 52 : Μορφή ιστοσελίδας

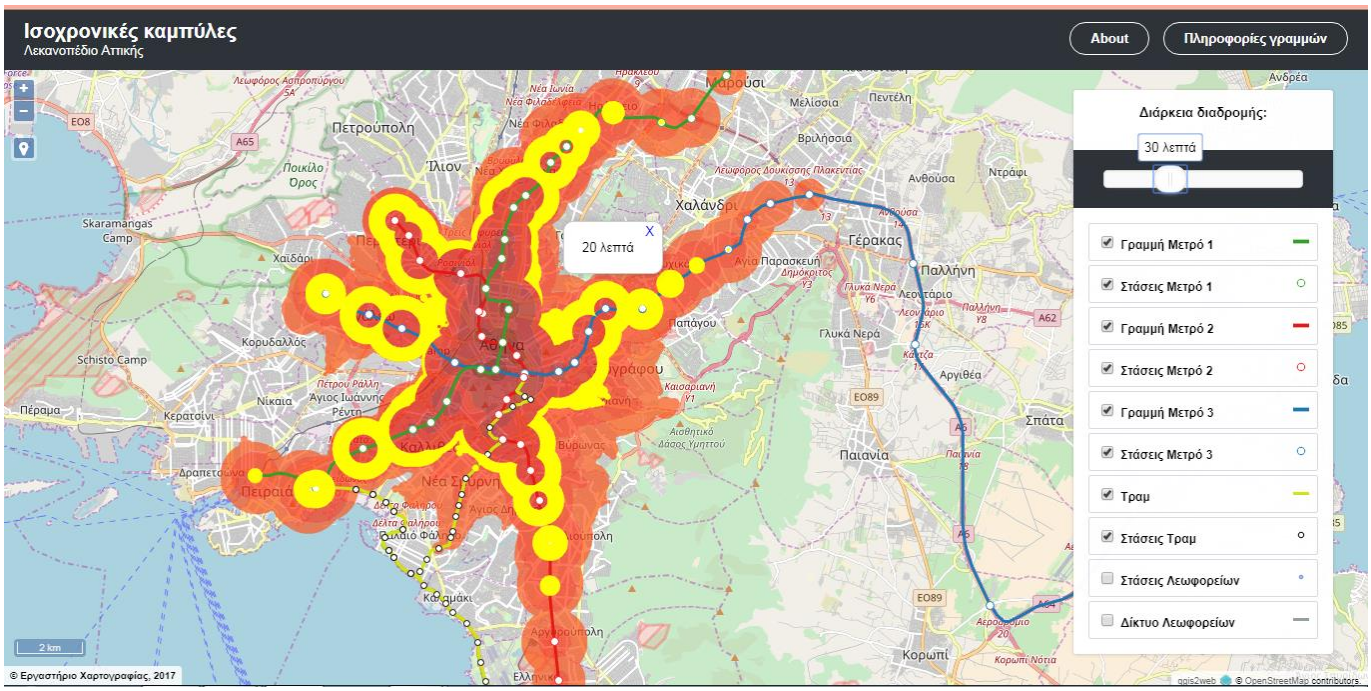
Για να αποφευχθεί ο θόρυβος στον χάρτη, που προκαλείται από μεγάλο όγκο πληροφοριών, ορίστηκαν ως ορατά μόνο τα θεματικά επίπεδα των μέσων σταθερής τροχιάς με τις στάσεις τους. Για να εμφανιστούν το σύνολο των στάσεων και των λεωφορειακών γραμμών ο χρήστης θα πρέπει να κάνει κλικ στο αντίστοιχο πεδίο ελέγχου (checkbox). Η διάρκεια διαδρομής είναι αρχικά ορισμένη στα 0 λεπτά, γι' αυτό δεν εμφανίζεται και καμία ισοχρονική καμπύλη στον χάρτη.

Ο χρήστης έχει τη δυνατότητα να κυλίσει τη μπάρα που ορίζεται ο χρόνος και θα εμφανιστεί η αντίστοιχη ισοχρονική καμπύλη (Εικόνα 53). Για την ακρίβεια θα εμφανιστούν όλες οι καμπύλες που είναι μικρότερου ή ίσου χρόνου με την επιλεγμένη.

Εάν ο κέρσορας τοποθετηθεί πάνω από κάποια καμπύλη, τότε θα αλλάξει προσωρινά το χρώμα της σε κίτρινο και θα εμφανιστεί ένα παράθυρο που εξηγεί το χρόνο διαδρομής εντός της καμπύλης αυτής (Εικόνα 54). Το ίδιο θα συνέβαινε και με τα υπόλοιπα θεματικά επίπεδα που βρίσκονται στον χάρτη.



Εικόνα 53 : Ορισμός του χρόνου μετακίνησης στα 30 λεπτά και εμφάνιση των αντίστοιχων ισοχρονικών καμπυλών



Εικόνα 54: Αλλαγή χρώματος της ισοχρονικής καμπύλης και εμφάνιση παραθύρου ιδιοτήτων με τη μετακίνηση του κέρσορα πάνω από την καμπύλη των 20 λεπτών

6. ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΕΣ ΠΡΟΕΚΤΑΣΕΙΣ

Ολοκληρώνοντας την παρούσα εργασία και έχοντας μελετήσει διεξοδικά τα ζητήματα που πραγματεύτηκε, μπορούν να εξαχθούν κάποια συμπεράσματα. Επίσης με βάση τα συμπεράσματα αυτά γίνονται μερικές προτάσεις για μελλοντική έρευνα και βελτιώσεις.

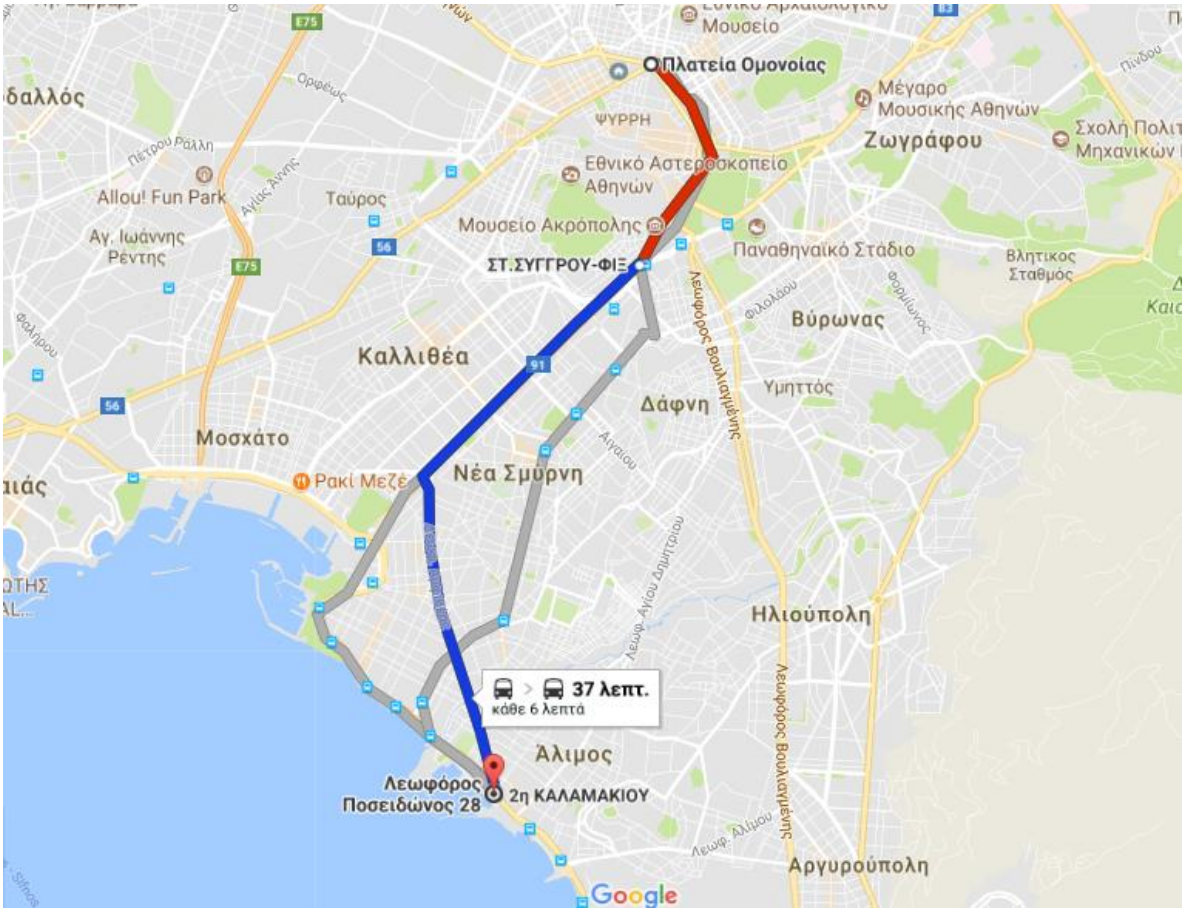
6.2. Συμπεράσματα

Όπως έχει ήδη αναφερθεί, σκοπός της παρούσας διπλωματικής εργασίας ήταν να δημιουργηθούν χάρτες με καμπύλες χρονοαπόστασης στο δίκτυο Μέσων Μαζικής Μεταφοράς (ΜΜΜ) της Αθήνας με αφετηρία την πλατεία της Ομόνοιας. Για το σκοπό αυτό ήταν αναγκαίο να δημιουργηθεί ένας πρωτότυπος αλγόριθμος ο οποίος θα εκτελεί τους απαιτούμενους υπολογισμούς γρήγορα και αποδοτικά. Σημαντικό μέρος αποτελεί και η χαρτογραφική παρουσίαση των αποτελεσμάτων μίας τέτοιας ανάλυσης, καθώς ο όγκος πληροφοριών είναι μεγάλος και πυκνός.

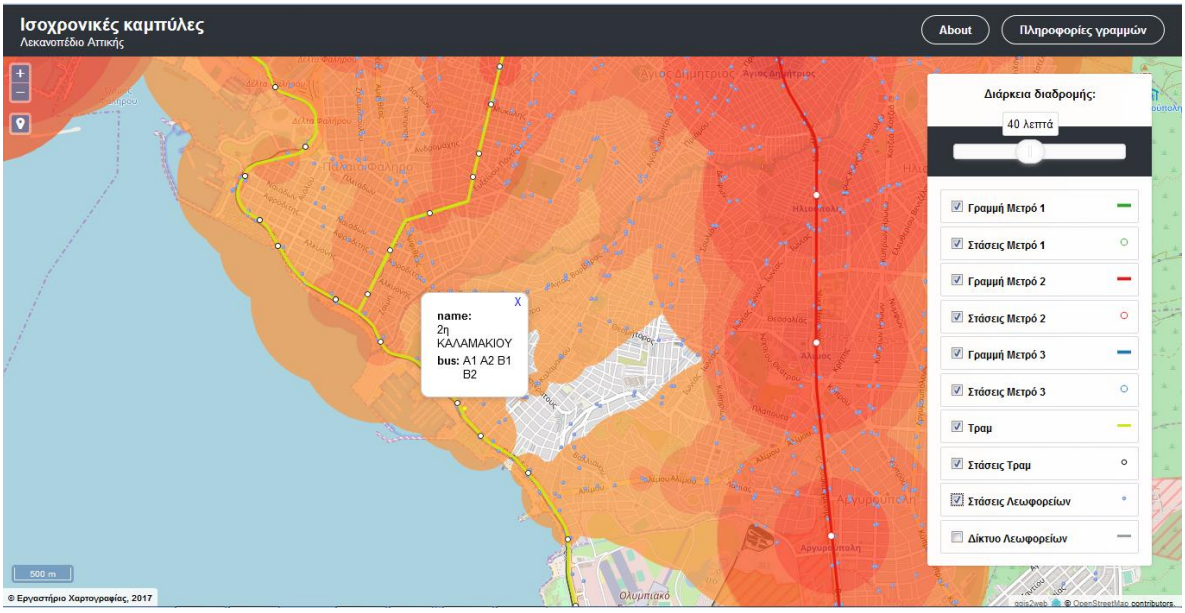
Μέσω της βελτιστοποίησης του αλγορίθμου επιτεύχθηκε ελαχιστοποίηση του χρόνου υπολογισμού και απλοποίηση της διαδικασίας, μέσω μίας μόνο εντολής. Με τη δημιουργία ζωνών (buffers) οι ισοχρονικές καμπύλες υπολογίζονται με ακρίβεια και με ιδιαίτερα ικανοποιητικά οπτικά αποτελέσματα.

Η δημιουργία της διαδικτυακής εφαρμογής συντελεί στην παρουσίαση των ισοχρονικών καμπυλών μέσω ενός φιλικού περιβάλλοντος προς το χρήστη. Το γεγονός ότι τέτοιου είδους μελέτες είναι δυνατό να χρησιμοποιηθούν και για εμπορικές εφαρμογές, καθιστά τη δημιουργία μίας εύχρηστης εφαρμογής αναγκαία.

Όσον αφορά τη λειτουργία του αλγορίθμου, τα αποτελέσματα ελέχθησαν μέσω συγκριτικών κοστών διαδρομής υπολογισμένων μέσω του Google Maps. Για παράδειγμα στις Εικόνες 55, 56 φαίνεται πως η συντομότερη διαδρομή Πλατεία Ομόνοιας –Στάση 2^{η} Καλαμακίου έχει προβλεπόμενο κόστος 37 λεπτών στην εφαρμογή του Google Maps, ενώ σύμφωνα με την παρούσα μελέτη η περιοχή βρίσκεται στην ισοχρονική των 35 – 40 λεπτών.



Εικόνα 55 : Χρόνος διαδρομής υπολογισμένος στο Google Maps



Εικόνα 56 : Χρόνος διαδρομής υπολογισμένος μέσω του αλγορίθμου του προγράμματος

6.3. Μελλοντικές προεκτάσεις

Ο νέος αλγόριθμος που δημιουργήθηκε μπορεί να αποτελέσει βάση για τον υπολογισμό ισοχρονικών καμπυλών σε δημόσια δίκτυα μεταφοράς. Επιδέχεται βέβαια περεταίρω βελτιώσεις και προεκτάσεις με σκοπό πιο ρεαλιστικά αποτελέσματα και μεγαλύτερη διαδραστικότητα της εφαρμογής. Στο εδάφιο αυτό παρουσιάζονται κάποιες προτεινόμενες προεκτάσεις και οι διαδικασίες με τις οποίες θα μπορούσαν να υλοποιηθούν.

Η πρώτη πρόταση αφορά στην ενσωμάτωση των χρονοδιαγραμμάτων των ΜΜΜ στην εφαρμογή. Η βάση δεδομένων θα μπορούσε συνδεθεί σε πραγματικό χρόνο με τα δεδομένα τηλεματικής του ΟΑΣΑ και με δεδομένη την ώρα εκκίνησης από την αφετηρία να υπολογίζονται οι ακριβείς χρόνοι διαδρομής και όχι η εκτίμηση αυτών. Για παράδειγμα αν ο χρήστης βρίσκεται σε μία στάση λεωφορείου και γνωρίζουμε ότι το επόμενο λεωφορείο έρχεται σε 3 λεπτά, θα χρησιμοποιηθεί αυτός ο χρόνος αναμονής και όχι κάποιος προσεγγιστικός.

Το πρόγραμμα θα μπορούσε να εμπλουτιστεί ώστε να δέχεται δεδομένα κίνησης της πόλης σε πραγματικό χρόνο και σύμφωνα με αυτά να πραγματοποιείται ο υπολογισμός στα βάρη των ακμών. Στη συγκεκριμένη προσθήκη δεν θα πρέπει να παραληφθούν οι αποκλειστικές λωρίδες λεωφορείων καθώς θεωρητικά η ταχύτητα των λεωφορείων που κινούνται κατά μήκος τους δεν επηρεάζεται από τις συνθήκες κίνησης. Με αυτό τον τρόπο ο χρήστης θα μπορεί να βλέπει τις ισοχρονικές καμπύλες όπως αυτές διαμορφώνονται αν ξεκινήσει εκείνη τη στιγμή από το αρχικό σημείο καθώς και πώς αυτές μεταβάλλονται κατά τη διάρκεια της ημέρας.

Σημαντικό χαρακτηριστικό του προγράμματος δρομολόγησης είναι οι διαδρομές βαδίσματος. Στην παρούσα εφαρμογή οι εν λόγω διαδρομές είναι ευθείες γραμμές των οποίων το μήκος είναι η ευκλείδεια απόσταση μεταξύ των κόμβων τους οποίους συνδέουν. Πρόκειται για μία σύμβαση που θα μπορούσε να αντικατασταθεί από πραγματικά περπατήσιμες διαδρομές. Για το σκοπό αυτό θα πρέπει να εισαχθεί στο γράφο, το δίκτυο των πεζοδρόμων , των διαβάσεων και των τεχνητών ή φυσικών φραγμών τους.

Μία επιπλέον προσθήκη που αφορά στην εισαγωγή διαδραστικότητας στην εφαρμογή είναι η σύνδεσή της με τη βάση δεδομένων και την Python, ώστε να δίνεται στον χρήστη η δυνατότητα να ορίζει το σημείο εκκίνησης και το χρόνο μετακίνησης και να εμφανίζονται οι αντίστοιχες ισοχρονικές καμπύλες. Για να υλοποιηθεί κάτι τέτοιο θα πρέπει να ακολουθηθεί η εξής ακολουθία : με το κλικ του ποντικιού του χρήστη πάνω στο χάρτη θα πρέπει να βρίσκεται ο εγγύτερος κόμβος, το αίτημα να στέλνεται στη βάση δεδομένων, να ξεκινάν επιτόπου οι υπολογισμοί με κόμβο αφετηρίας τον εγγύτερο κόμβο και τέλος να εμφανίζονται τα αποτελέσματα στην οθόνη του χρήστη.

ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΑΝΑΦΟΡΑ

V. Bauer, J. Gamper, R. Loperfido, S. Profanter, S. Putzer, I. Timko, Computing isochrones in multi- modal, schedule-based transport networks, GIS '08, Nov. 2008

Valentin Buchhold (B), Julian Dibbelt, and Dorothea Wagner Karlsruhe Fast Exact Computation of Isochrones in Road Networks Moritz Baum, Institute of Technology, Karlsruhe, Germany
{moritz.baum,valentin.buchhold,julian.dibbelt,dorothea.wagner}@kit.edu

Aldous M. Joan, Wilson J. Robin (2000), Graphs and Applications: An Introductory Approach, Springer Publications, London.

Jin-Seo Park and Se-Jong Oh, A New Concave Hull Algorithm and Concaveness Measure for n-dimensional Datasets, Journal of Information Science and Engineering, Vol. 28 No. 3, pp. 587-600, May 2012

O'Reilly Radar , Programming Language Trends, Radar.oreilly.com. 2 August 2006

Biggs, N.; Lloyd, E. and Wilson, R. (1986). Graph Theory, 1736-1936. Oxford University Press.

Cormen H. Thomas, Leiserson E. Charles, Rivest L. Ronald and Stein Clifford (2001), Introduction to Algorithms, The MIT Press, Massachussets Institute of Technology.
Reinhard Diestel, Graph Theory, Springer GTM 173, 5th edition 2016

L'Huillier, S.-A.-J. (1861). «Mémoire sur la polyèdrométrie». Annales de Mathématiques 3: 169–189.

Johann Gamper et al., De_ning isochrones in multimodal spatial networks, Proceedings of the 20th ACM international conference on Information and knowledge management, CIKM 2011, Glasgow, Scotland

Veronika Bauer et al., Computing isochrones in multi-modal, schedule-based transport networks, Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems. GIS '08. Irvine,California: ACM, 2008,

Johann Gamper, Michael Bohlen, and Markus Innerebner, Scalable Computation of Isochrones with Network Expiration, Proceedings of the 24th International Conference on Scientific and Statistical Database Management. SSDBM' 12. Chania, Crete, Greece: Springer-Verlag, 2012,.

Σχεδιασμός αλγορίθμων, John Kleinberg, Eva Tardos, 2008

Στεφανάκης Εμμανουήλ, Βάσεις Γεωγραφικών Δεδομένων και Συστήματα Γεωγραφικών Πληροφοριών, Εκδόσεις Παπασωτηρίου, Αθήνα, 2013.

Κολιός Ν. (2009), Χωρική Βάση Δεδομένων PostgreSQL/ PostGIS και Σύστημα Γεωγραφικών Πληροφοριών Quantum GIS, Οδηγός Χρήσης, Εταιρεία Ελεύθερου Λογισμικού/ Λογισμικού Ανοικτού Κώδικα- Creative Commons Attribution- Non Commercial- ShareAlike 3.0- Ελλάδα

Στεφανάκης Εμμανουήλ, Τεχνολογίες Δημοσιοποίησης Χαρτογραφικού Περιεχομένου στον Παγκόσμιο Ιστό, Εκδόσεις Νέων Τεχνολογιών, 2009

Shekhar Shashi, Chawla Sanjay (2003), Spatial Databases: A Tour, Pearson Education Inc., Upper Saddle River, New Jersey 07458.

openlayer's 2.10 Beginner's Guide

pgRouting, Practical Guide, 2016

Musciano, C., and Kennedy, B., 2006. HTML & XHTML: The Definitive Guide. O'Reilly Media, Inc.; 6 edition

Marijin Haverbeke, Eloquent JavaScriptsecond edition, 2011

Ιστοσελίδες

Algorithms and Data Structures:
http://www.algolist.net/Data_structures/Graph

Google Maps:
http://www.googlemaps.com

Graph Theory Springer GTM 173, 5th edition 2016
http://diestel-graph-theory.com/basic.html?

Mapumental :
http://www.mapumental.com

Wikipedia:
https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm
https://el.wikipedia.org/wiki/PostgreSQL
https://en.wikipedia.org/wiki/Bellman%E2%80%93Ford_algorithm

Oasa official site :
http://www.ametro.gr/?page_id=10

Oasa telematics site :
http://telematics.oasa.gr/

pgRouting Official Site :
http://pgrouting.org/

PostgreSQL official site :
https://www.postgresql.org/

PostGIS Manual 1.3.6:
http://postgis.refractions.net/documentation/manual-1.3/

MySQL official site :
https://www.mysql.com/

pgAdmin official site :
https://www.pgadmin.org/

Documentation for QGIS 2.18 :
http://docs.qgis.org/2.18/en/docs/

Python for Beginners :
https://www.python.org/about/gettingstarted/

Python Tutorials :
https://www.tutorialspoint.com/python/python_basic_syntax.htm

w3schools :
https://www.w3schools.com/

HTML Language Specification:
http://www.w3.org/html/

Css Specification:
http://www.wlearn.gr/index.php/home-css-83

JavaScript official site:
https://www.javascript.com/

Regular expressions of JavaScript :
https://regexr.com/

OpenLayers official site :
https://openlayers.org/

JavaScript Range Slider :
https://refreshless.com/nouislider/

GitHub, The world's leading software development platform :
https://gist.github.com/econchick/4666413
https://github.com/iotacss/iotacss