# 1ο ΘΕΜΑ ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

ΟΔΗΓΙΕΣ ΤΡΕΞΙΜΑΤΟΣ ΠΡΟΓΡΑΜΜΑΤΟΣ:
1) Μέσα από τον φάκελο code:  java Taxi path/to/csvs/

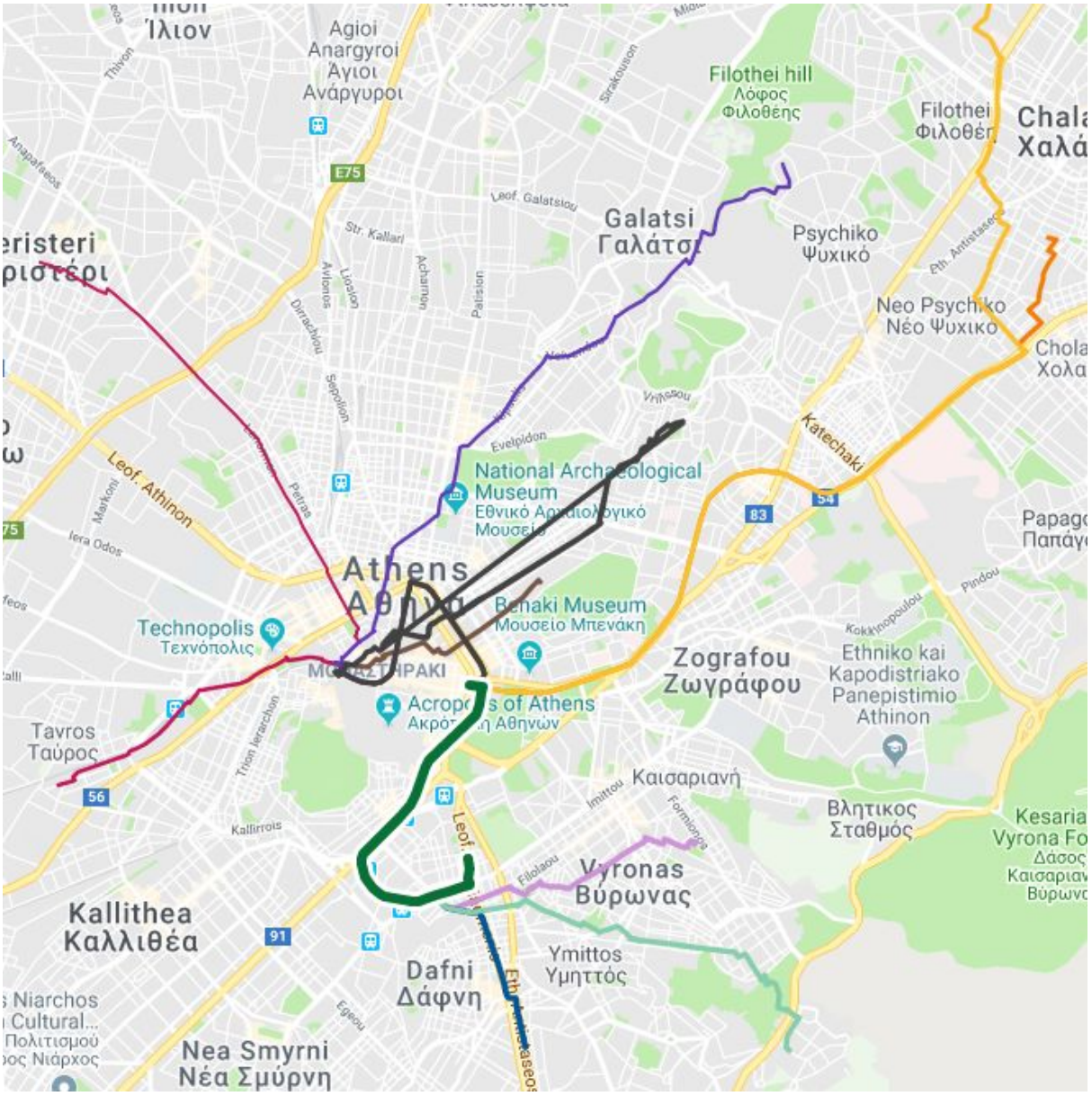
2) Μετατροπή των taxiI.out σε kml αρχεία (από τον φάκελο code)
./make_kml.sh code/folder_with_out_files result.kml

Κατ'αρχάς δεν χρειάστηκε να γίνει κάποια **προεπεξεργασία** στα δεδομένα που μας δόθηκαν.
Ο αλγόριθμος A* χρησιμοποιεί αντικείμενα κλάσης Node τα οποία έχουν συντεταγμένες και id δρόμου , και να μπορούν να μας επιστρέφουν τους γείτονές τους που είναι επίσης αντικείμενα κλάσης **Node**.
Με χρήση της **δομής δεδομένων Map** της java , κάθε σημείο που διαβάστηκε από το αρχείο nodes αντιστοιχίζεται σε μιά λίστα που περιέχει τους γείτονές του .
Έπειτα σε αυτόν τον χάρτη προστέθηκε το αρχικό σημειο (η θέση του κάθε taxi , διαφορετική για κάθε ένα) και το τελικό σημείο, δηλαδή η θέση του client.
Αυτό επιτυγχάνεται ως εξής : αν ήδη υπάρχουν αυτά τα σημεία στο Map με τους δοθέντες κόμβους παραμένει αμετάβλητο, αλλιώς αν κάποιο βρίσκεται μεταξύ δύο σημείων τότε το προσθέτουμε στο υπάρχον Map ανανεώνοντας τις λίστες γειτόνων για τα τρία σημεία. Τέλος στην περίπτωση που κάποιο-α σημείο έναρξης/ προορισμού δε βρίσκεται σε κάποια οδό του χάρτη τότε το ενώνουμε με το πιο κοντινό σημείο και ανανεώνουμε το Map.
Η **εκτίμηση της απόστασης** γίνεται προσθέτοντας τον αριθμό βημάτων χρειάστηκε να κάνουμε από την αρχική , με την εκτιμητέα απόσταση από την τελική κατάσταση (που γίνεται με τον γνωστό τύπο απόστασης σημείων). Στο τέλος για τη συντομότερη διαδρομή του κάθε taxi επιστρέφεται σε ένα αρχείο καθώς και ο αριθμός των βημάτων που πραγματοποιήθηκαν , που είναι φυσικά ίδιος για κάθε πιθανή εναλλακτική διαδρομή.

Ο αρχικός κόμβος, δηλαδή αυτός που αντιστοιχεί στο taxi έχει πρόσβαση στον χάρτη , και με αυτόν ως αρχή εφαρμόζουμε τον **παραλλαγμένο A* αλγόριθμο**.
Για την υλοποίηση του βασικού αλγορίθμου βασιστήκαμε στον ψευδοκώδικα της Wikipedia (https://en.wikipedia.org/wiki/A*_search_algorithm#Pseudocode). Ειδικότερα, το closedSet το υλοποιήσαμε με HashSet ενώ για το openSet δημιουργήσαμε δικιά μας δομή η οποία βασίζεται σε PriorityQueue .

Ο **αλγόριθμος A*** όπως τον υλοποιήσαμε , αν φτάσει δεύτερη (ή και τριτη , τεταρτη κ.ο.κ.) φορά σε ένα σημείο το οποίο βρίσκεται ήδη στο μέτωπο αναζήτησης από κάποια άλλη διαδρομή , έχοντας διανύσει ως εκεί ο αλγόριθμος/ταξί την ίδια απόσταση , προστίθεται στη λίστα των προηγούμενων κόμβων του ένας ακόμα εναλλακτικός.
Η παραλλαγή αυτή δεν αλλάζει την ιδιότητα του απλού A* να βρίσκει πάντοτε την βέλτιστη λύση. Όπως είναι γνωστό , όταν η τιμή της εκτιμώμενης απόστασης είναι μικρότερη ή ίση της πραγματικής μεταξύ ενός σημείου και του τελικού , τότε ο A* βρίσκει βέλτιστη λύση .Εδώ ισχύει αυτή η ιδιότητα αφού η εκτίμηση απόστασης σημείου από την τελική κατάσταση γίνεται σε ευθεία , υπολογίζοντας απόσταση των 2 σημείων γεωμετρικά , άρα επειδή μπορεί να μην βρίσκονται στον ίδιο δρόμο (ίδια ευθεία) , η πραγματική θα είναι πάντα μεγαλύτερη ή ίση. Η ιδιότητα αυτή λοιπόν δεν χάνεται γιατί ο αλγόριθμος απλά προτείνει εναλλακτικές διαδρομές από την αρχική ως κάποιον κόμβο όταν τα μήκη τους είναι ίσα και δεν επηρεάζει την ροή του απλού A* αλγορίθμου.

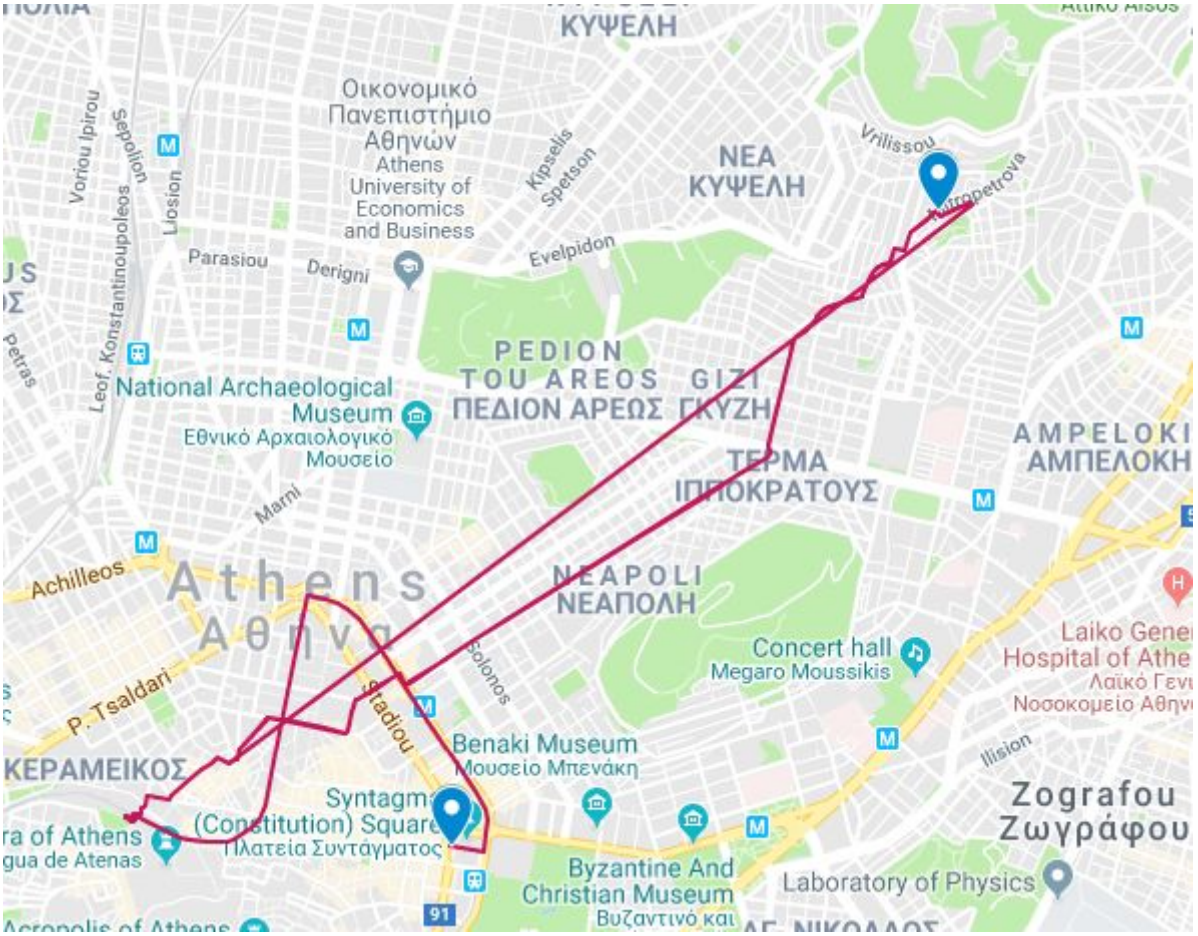
Παρακάτω φαίνεται ο χάρτης διαδρομών για τα **default input**, με βέλτιστη διαδρομή αυτή του taxi 6 που είναι με πράσινο χρώμα. Κάποια σημεία διαδρομών επικαλύπτονται για αυτό και φαίνονται με ένα χρώμα. Στην διαδρομή του taxi 8 ( με το γκρι χρώμα) φαίνεται πως υπάρχουν 2 πιθανές διαδρομές με ίσα βήματα.



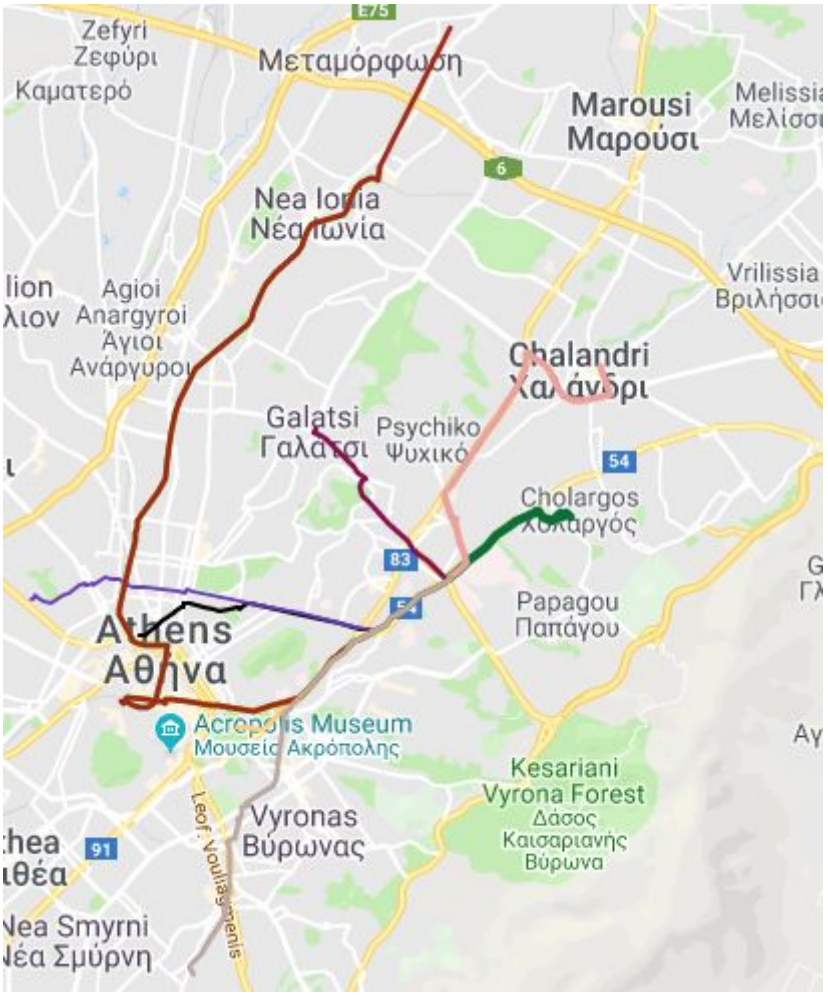

| TAXIS | STEPS | taxi6 | 10431 |
|---|---|---|---|
| taxi1 | 20707 | taxi7 | 10906 |

| | | | |
|---|---|---|---|
| taxi2 | 18348 | taxi8 | 30281 |
| taxi3 | 20884 | taxi9 | 20719 |
| taxi4 | 33772 | taxi10 | 23814 |
| taxi5 | 41100 | taxi11 | 14923 |

Στην παρακάτω εικόνα θα δούμε ξεχωριστά τις διαδρομές του taxi 8. Φαίνεται πως υπάρχουν αρκετές διακλαδώσεις/πιθανές διαδρομές που μπορεί να ακολουθήσει το ταξί από το σημείο που είναι μέχρι τον πελάτη. Χάρη στην παραλλαγή του A* φαίνονται και ολες αυτές.



Παρακάτω φαίνεται ο χάρτης για το **our input,** με τις διαδρομές για δικό μας σημείο του πελάτη και δικά μας σημεία για τα ταξί. Η βέλτιστη διαδρομή φαίνεται πως είναι για το δικό μας taxi 5.




| TAXIS | STEPS | taxi5 | 251 |
|---|---|---|---|
| taxi1 | 661 | taxi6 | 965 |
| taxi2 | 3876 | taxi7 | 6937 |
| taxi3 | 1105 | taxi8 | 6938 |
| taxi4 | 6320 | taxi9 | 6395 |