

Τεχνητή Νοημοσύνη-1^ο Θέμα

Ιωάννης Αναγνωστίδης 03115094
Νικόλαος Μπούας 03115005

23 Δεκεμβρίου 2018

Περιεχόμενα

1 Εισαγωγή	1
2 Υλοποίηση	1
3 Εκτέλεση	3
4 Αναφορές	7

1 Εισαγωγή

Στα πλαίσια του πρώτου προγραμματιστικού θέματος, κληθήκαμε να υλοποιήσουμε μια εφαρμογή εύρεσης της συντομότερης διαδρομής μεταξύ ενός πελάτη και ορισμένων ταξί. Πιο συγκεκριμένα, το πρόγραμμά μας χρησιμοποιεί τα δεδομένα του αιθναϊκού οδικού δικτύου και εφαρμόζοντας τον αλγόριθμο A^* προσδιορίζει για το εκάστοτε ταξί το συντομότερο μονοπάτι. Έτσι, είμαστε σε θέση να διαχρίνουμε το ταξί που χαρακτηρίζεται από την μικρότερη απόσταση από τον πελάτη, μια υπηρεσία που όπως γίνεται αντιληπτό έχει άμεση πρακτική εφαρμογή.

2 Υλοποίηση

Για την υλοποίηση της συγκεκριμένης εφαρμογής χρησιμοποιήσαμε τη γλώσσα προγραμματισμού *Java*, ενώ τα δεδομένα εισόδου που επεξεργάζεται το πρόγραμμά μας αποτελούν αρχεία *.csv*. Επιπλέον, για να οπτικοποιήσουμε καλύτερα το τελικό αποτέλεσμα, χρησιμοποιήσαμε αρχεία της μορφής *.kml* που επεξεργάζονται κατάλληλα από την υπηρεσία *Google – Maps*. Για τη δομημένη και επεκτάσιμη υλοποίηση τμηματοποιήσαμε τον κώδικα στις ακόλουθες κλάσεις:

- *Point*: Αναπαριστά ένα σημείο στο χάρτη με *private* πεδία γεωγραφικές συντεταγμένες στην μορφή *latitude* και *longitude*, καθώς έτσι αναπαριστάται ένα σημείο στο χάρτη σε

αρχεία *.csv* και *.kml*. Επιπλέον, πέρα από κατάλληλες *public* μεθόδους για την προσπέλαση, την ανάθεση και την αρχικοποίηση των πεδίων ενός σημείου, έχει οριστεί και μία μέθοδο που υπολογίζει την ευκλίδεια απόσταση μεταξύ δύο σημείων, εφαρμόζοντας τον κατάλληλο γεωμετρικό μετασχηματισμό.

- *Junction*: Αποτελεί το σήμειο διασταύρωσης μεταξύ επιμέρους δρόμων. Χρησιμοποιείται για την απλοϊκότερη μοντελοποίηση του γραφήματος περιέχει για κάθε χόμβο την πληροφορία των γειτονικών σημείων, που φυσικά απαιτείται για την επέκταση ενός χόμβου κατά την εκτέλεση του A^* .
- *Node*: Αναπαριστά μια κορυφή, που αποτελεί ένα σημείο στο χάρτη ως στιγμιότυπο όμως της εκτέλεσης του αλγορίθμου. Στο πλαίσιο αυτό, περιέχει πληροφορίες σχετικά με την απόσταση που έχει διανυθεί καθώς και την εκτιμώμενη απόσταση από το στόχο. Αξίζει να αναφερθεί, ότι σαν ευριστική συνάρτηση έχει χρησιμοποιηθεί η ευκλίδεια απόσταση, καθώς αποτελεί μια πολύ διαισθητική και απλή εκτιμήτρια που πάντα υποεκτιμά την απόσταση από το στόχο. Κάθε αντικείμενο της κλάσης περιέχει επίσης ως πεδίο το τρέχων σημείο στο χάρτη που αντιστοιχίζεται ο κόμβος, καθώς επίσης και στατικές πληροφορίες, όπως είναι όλες οι πιθανές διασταύρωσεις των δρόμων μέσω της κλάσης *Junction* και το σημείο στόχος, δηλαδή η τοποθεσία του πελάτη. Στη συγκεκριμένη κλάση έχουν οριστεί επίσης κατάλληλες μέθοδοι που δημιουργούν και επιστρέφουν το σύνολο των γειτονικών κόμβων ενός γράφου, ενώ έχουμε κάνει *override* την μέθοδο *equals* και *hashcode* ώστε η ισότητα δύο *Nodes* να γίνεται με κριτήριο την ισότητα των αντίστοιχων συντεταγμένων.
- *AstarSolver*: Χρησιμοποιείται για την εφαρμογή του αλγορίθμου A^* . Δηλαδή, με δεδομένο ένα αρχικό σημείο και ένα σημείο στόχος προσδιορίζονται όλα τα πιθανά ελάχιστα μονοπάτια. Για να το πετύχουμε αυτό, επειδή οι αριθμοί που εμπλέκονται είναι αριθμοί μεγάλης ακρίβειας έχουμε θεωρήσει αυθαίρετα ότι οι διαδρομές που απέχουν λιγότερο από 1 μέτρο θα θεωρούνται ίσες. Για την αποδοτική εκτέλεση του αλγορίθμου έχουν χρησιμοποιηθεί σαν δομές δεδομένων ένα *PriorityQueue*, για την εύρεση του στοιχείου από το μέτωπο αναζήτησης που ελαχιστοποιεί την συνολική απόσταση από το στόχο, καθώς επίσης και ένα *HashTable* για την αποδοτική αναζήτηση στοιχείων στο κλειστό σύνολο. Έτσι, ένας κόμβος αποτελεί τερματικό όταν οι αντίστοιχες γεωγραφικές συντεταγμένες με τον κόμβο στόχο συμπτίπουν.
- *NodeComparator*: Στην συγκεκριμένη κλάση ορίζουμε τη σύγκριση μεταξύ των κορυφών, που γίνεται με κριτήριο το άνθροισμα της τρέχουσας απόστασης με την εκτιμώμενη απόσταση από το στόχο, όπως προβλέπει ο αλγόριθμος A^* .
- *Main*: Η βασική κλάσης που γίνεται ανάγνωση των δεδομένων εισόδου, αναπαριστώνται τα δεδομένα χρησιμοποιώντας τις παραπάνω κλάσεις, εκτελείται ο αλγόριθμος για κάθε ταξί και τελικά τυπώνεται η έξοδος, δηλαδή το ταξί που χαρακτηρίζεται από την μικρότερη απόσταση από τον πελάτη, καθώς και όλες οι ελάχιστες διαδρομές των ταξί από τον πελάτη. Στο σημείο αυτό, πρέπει να αναφέρουμε ότι επειδή ενδέχεται ο πελάτης να βρίσκεται σε σημείο που δεν είναι προσπελάσιμο από το οδικό δίκτυο, για παράδειγμα

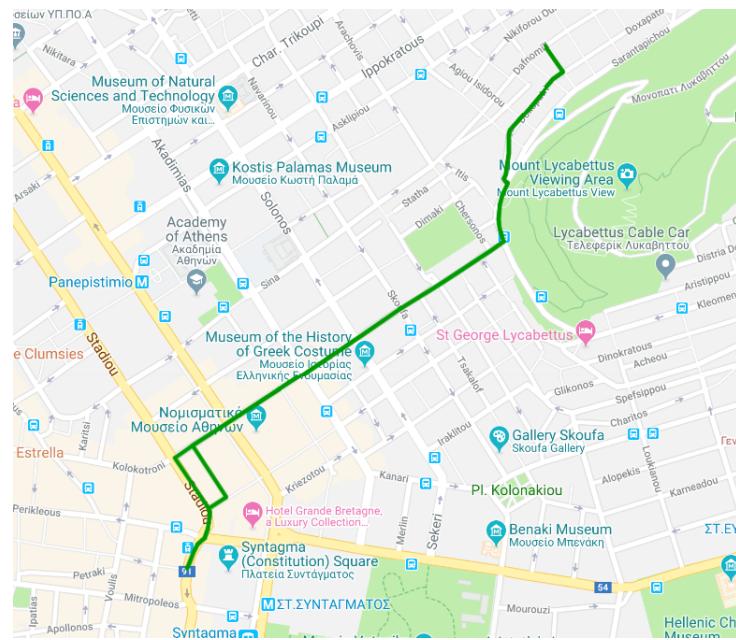
αν βρίσκεται σε ένα πάρκο ή σε μία πλατεία, αρχούμαστε με το να φτάσει το ταξί στον πλησιέστερό του κόμβο, όπως ακριβώς θα περιμέναμε και από μια ρεαλιστική εφαρμογή.

3 Εκτέλεση

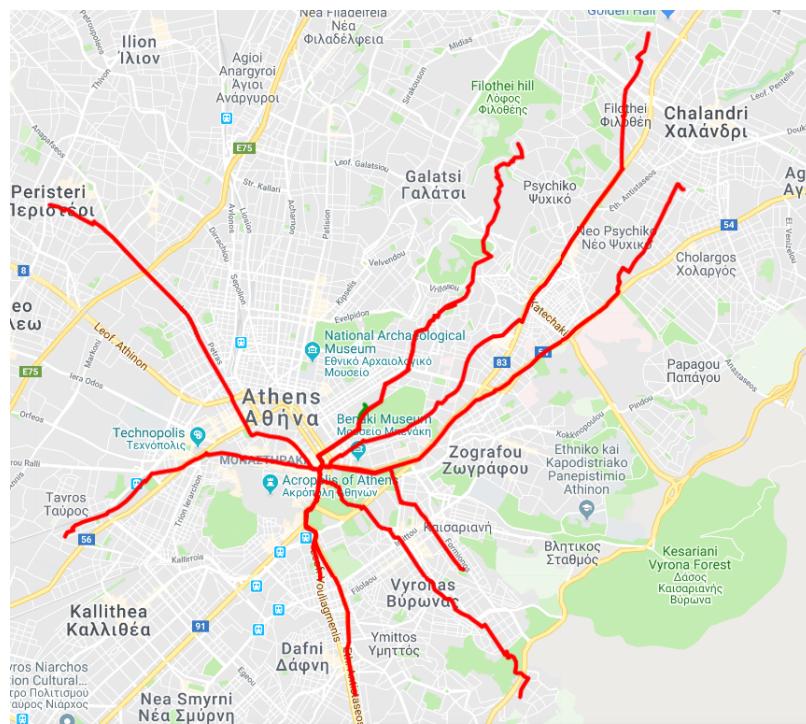
Για τις αρχικές θέσεις του πελάτη και των ταξί προκύπτουν με την εκτέλεση του αλγορίθμου οι ακόλουθες αποστάσεις:

- Απόσταση από το ταξί με $id = 100$: $1.503km$
- Απόσταση από το ταξί με $id = 110$: $5.150km$
- Απόσταση από το ταξί με $id = 120$: $3.395km$
- Απόσταση από το ταξί με $id = 130$: $9.481km$
- Απόσταση από το ταξί με $id = 140$: $7.101km$
- Απόσταση από το ταξί με $id = 150$: $1.805km$
- Απόσταση από το ταξί με $id = 160$: $3.515km$
- Απόσταση από το ταξί με $id = 170$: $3.818km$
- Απόσταση από το ταξί με $id = 180$: $5.717km$
- Απόσταση από το ταξί με $id = 190$: $6.535km$
- Απόσταση από το ταξί με $id = 200$: $8.496km$

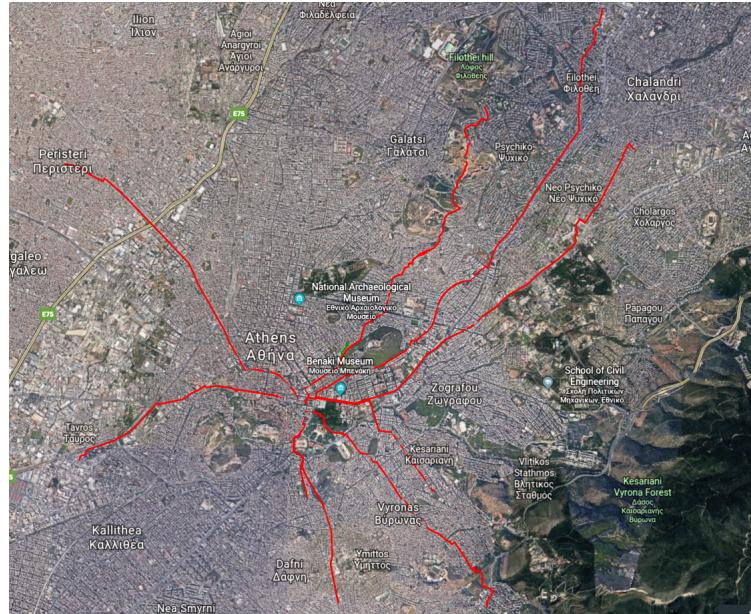
Επομένως, γίνεται φανερό ότι το ταξί με $id = 100$ χαρακτηρίζεται από την μικρότερη απόσταση από τον πελάτη σε σχέση με τα υπόλοιπα ταξί, με απόσταση $1.503km$ με ακρίβεια μέτρων. Για να οπτικοποιήσουμε τη διαδρομή που θα ακολουθήσει το συγκεκριμένο ταξί, θα προσθέσουμε τις συντεταγμένες των ενδιάμεσων σημείων που θα χρησιμοποιήσει το ταξί σε ένα αρχείο *.kml*. Έπειτα, θα εισάγουμε το συγκεκριμένο αρχείο στην υπηρεσία *Google – Maps* και θα πάρουμε μια γραφική απεικόνιση της διαδρομής που θα ακολουθήσει στο χάρτη. Έτσι, με πράσινο χρώμα φαίνονται οι δύο βέλτιστες από πλευράς χιλιομετρικής απόστασης που προέκυψαν με βάσει τις παραδοχές που συζητήθηκαν παραπάνω:



Στη συνέχεια, παρουσιάζουμε το σύνολο των βέλτιστων διαδρομών από κάθε ταξί, όπως αυτά προέκυψαν από την εφαρμογή του αλγορίθμου, όπου η βέλτιστη διαδρομή, που σε μεγάλο βαθμό έχει υπερκαλυφθεί από μία άλλη διαδρομή, φαίνεται με πράσινο χρώμα και οι υπόλοιπες με κόκκινο.



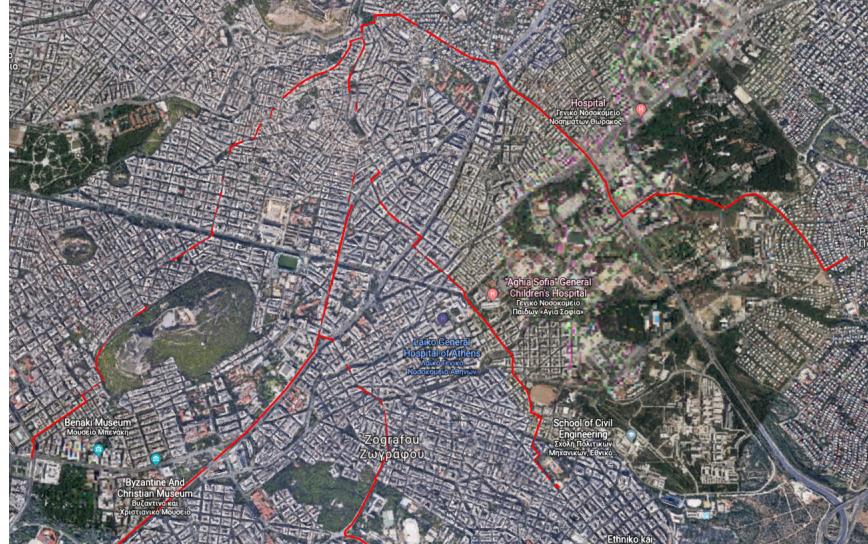
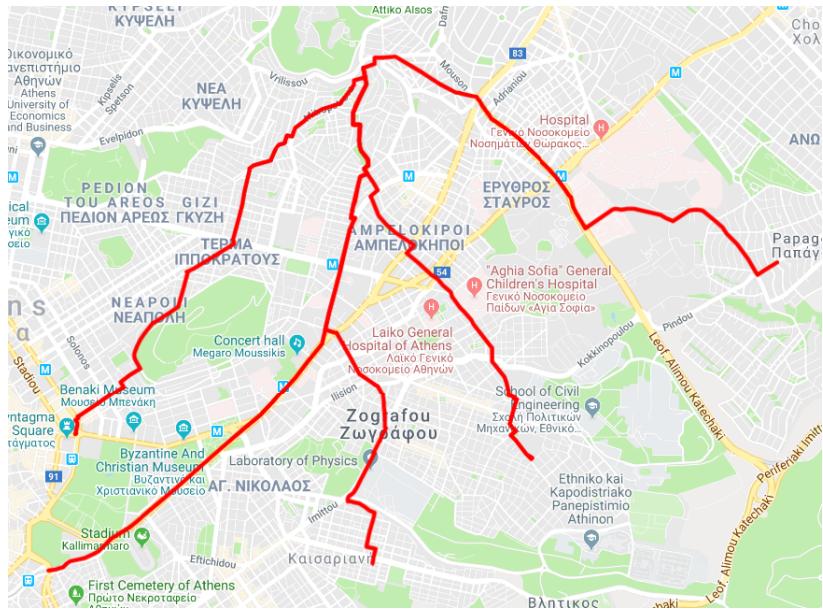
Επιπλέον, παρουσιάζουμε παρακάτω το σύνολο των ελάχιστων διαδρομών και μέσω της υπηρεσίας *Google – Earth*



Στη συνέχεια, δημιουργήσαμε ορισμένα δικά μας δεδομένα, που έχουν συμπλειληφθεί στο παραδοτέο στον φάκελο *csv*, για να αναδείξουμε καλύτερα τον τρόπο εκτέλεσης του αλγορίθμου. Συγκεκριμένα, παρακάτω φαίνεται ένα παράδειγμα όπου ο αλγόριθμος υπολογίζει δύο εναλλακτικές διαδρομές που χαρακτηρίζονται εικονικά από την ίδια απόσταση:



Συνολικά, οι ελάχιστες διαδρομές μεταξύ ενός πελάτη και πέντε διαφορετικών ταξί είναι οι ακόλουθες:



4 Αναφορές

Για την υλοποίηση της παραπάνω εφαρμογής αξίζει να αναφέρουμε τους παρακάτω ιστιοτόπους:

- *stackoverflow.com*: Για την αποσφαλμάτωση του προγράμματος και τον προσδιορισμό της ευκλίδειας απόστασης δύο σημείων με πεδία γεωμετρικές συντεταγμένες της μορφής *latitude* και *longitude*.
- *docs.oracle.com*: Για την αναζήτηση βιβλιοθηκών που υλοποιούν τις δομές δεδομένων που χρησιμοποιήσαμε.
- *www.google.com/maps* Για την οπτικοποίηση των διαδρομών που προκύπτουν από την εκτέλεση του αλγορίθμου.