Τεχνητη Νοημοσυνη

$2^ο$ Θέμα

Ευσταθίου Δανάη, 03115122

Σπηλιόπουλος Διονύσιος, 03115195

Γενική Περιγραφή

Στο θέμα αυτό τροποποιήθηκε ο αλγόριθμος εύρεσης του κοντινότερου σημείου που είχε υλοποιηθεί στο $1^ο$ θέμα. Ποιο συγκεκριμένα:

- Χρησιμοποιήθηκε γλώσσα Prolog για την αποθήκευση της βάσης γνώσης και κατηγορημάτων, μέσω των οποίων η Java και ο αλγόριθμος A^* μπορεί να λάβει στοιχεία της γνώσης και να τα χρησιμοποιήσει.
- Επεκτάθηκε ο τρόπος δημιουργίας του γράφου που αναπαριστά τους δρόμους, τα ταξί και τους πελάτες, έτσι ώστε να μπορεί να κωδικοποιήσει επιπλέον μονόδρομους, διπλούς δρόμους, ώρες κίνησης, επιτρεπόμενα όρια ταχύτητας, πλήθος επιβατών ταξί, γλώσσα ομιλίας οδηγού, δυνατότητα εκτέλεσης μακρινών δρομολογίων, αξιολόγηση, καθώς και επιθυμητό σημείο άφιξης πελάτη, γλώσσα ομιλίας αυτού και πλήθος συνεπιβατών του.
- Ως κόστος ακμών στην διαδικασία εύρεσης κοντινότερης απόστασης, καθώς και ως ευρεστική συνάρτηση, χρησιμοποιήθηκαν αυτήν την φορά ο χρόνος που χρειάζεται ένα όχημα να μεταβεί από έναν κόμβο σε έναν άλλον, δεδομένου του ότι πάει με ορισμένη μέση ταχύτητα και ο εκτιμώμενος χρόνος για την άφιξη στο τελικό σημείο αντίστοιχα.
- Οι μέσες ταχύτητες ορίστηκαν με βάση τα χαρακτηριστικά των οδών που δίνονται και επηρεάζονται από την κίνηση καθιστώντας τον αλγόριθμο ικανό να αποφεύγει οδούς με μεγάλη κίνηση.
- Επετράπη επίσης με αυτόν τον τρόπο η δυνατότητα να έχουμε εκτιμώμενους χρόνους άφιξης από το ταξί στον πελάτη, καθώς και από τον πελάτη στον επιθυμητό προορισμό του.

Περιγραφή αλγορίθμου

Γενικά

Από την πλευρά της υλοποίησης του αλγορίθμου A^* δεν άλλαξε κάτι. Αυτό που άλλαξε ήταν όπως προειπώθηκε οι ορισμοί της απόστασης και της ευρεστικής συνάρτησης. Συγκεκριμένα, για την απόσταση δύο κόμβων που συνδέονται με ακμή δημιουργήθηκε κατηγόρημα $distance(node_id_1, node_id_2, time, D)$ στην Prolog το οποίο βρίσκει την απόσταση των δύο κόμβων, με βάση τις ταυτότητές τους, βρίσκει την εκτιμώμενη μέση ταχύτητα της οδού στην οποία βρίσκονται οι κόμβοι, την πολλαπλασιάζει με τον συντελεστή κίνησης (αν υπάρχει) δεδομένης της ώρας και επιστρέφει τον εκτιμώμενο χρόνο μετάβασης από τον έναν στον άλλον. Όσο για την ευρεστική συνάρτηση χρησιμοποιήθηκε μία παγκόσμια εκτίμηση της μέσης ταχύτητας ώστε να είναι της ιδίας τάξης μεγέθους με την απόσταση και να μην γίνεται ο αλγόριθμος πολύ επιθετικός, χάνοντας διαδρομές.

Preprocessing

Το preprocessing έγινε με γλώσσα Java. Αρχικά δημιούργησε κατηγορήματα:

- node(node_id, line_id, x, y) το οποίο αποθηκεύει τις πληροφορίες ενός κόμβου όπως περιγράφονται στο αρχείο nodes.csv.

- line(id, limit, toll) το οποίο αποθηκεύει τις πληροφορίες μίας γραμμής όπως περιγράφονται στο αρχείο lines.csv.
- client(x, y, x_dest, y_dest, time, people, language) το οποίο αποθηκεύει τις πληροφορίες του πελάτη όπως περιγράφονται στο αρχείο client.csv.
- taxi(x, y, id, capacity, language, rating, long_distance) το οποίο αποθηκεύει τις πληροφορίες του πελάτη όπως περιγράφονται στο αρχείο taxis.csv.
- child(id1, id2) το οποίο αποθηκεύει την ύπαρξη δρόμου από τον κόμβο αναγνωριστικού id1 προς τον κόμβο αναγνωριστικού id2. Για τη δημιουργία των κατηγορημάτων αυτών, αξιοποιήθηκαν οι πληροφορίες που δόθηκαν από τα αρχεία lines.csv και nodes.csv.
- traffic(line_id, start_time, end_time, status το οποίο αποθηκεύει την υπάρξη κίνησης μεταξύ των ωρών start_time και end_time, ενώ το status αποτελεί ένα συντελεστή, με τον οποίο πολλαπλασιάζουμε την ταχύτητα της συγκεκριμένης οδού στις συγκεκριμένες ώρες, προκειμένου να αποδοθεί η κίνηση στην οδό.

Να σημειωθεί ότι όσες οδοί δεν είχαν περιγραφή αφαιρέθηκαν από το σύνολο γραμμών μαζί με τους κόμβους τους, ώστε να μην μειώνουν την ποιότητα των δεδομένων. Από αυτές που έμειναν αφαιρέθηκαν όσες είχαν περιγραφές proposed, pedestrian, elevator, service, cycleway, track, construction, bridleway, services, path, footway, unclassified και steps, διότι στο documentation του OpenStreetMaps φάνηκε ότι δεν μπορούν να υποστηρίξουν μηχανοκίνητα οχήματα. Αρκετές από όσες παρέμειναν είχαν το πρόβλημα ότι δεν είχαν συμπληρωμένο το πεδίο oneway. Η αφαίρεσή τους όμως θα μείωνε σημαντικά το πλήθος των γραμμών άρα κρατήθηκαν και θεωρήθηκαν ότι είναι διπλής κατευθύνσεως προς ελαχιστοποίηση των λαθών. Ύστερα δημιουργήθηκαν κατηγορήματα child(node_id1, node_id2) που στην ουσία δημιουργούν τον γράφο καθώς απεικονίζουν τις ακμές από τον κόμβο με node_id1 προς node_id2.
Όλα τα παραπάνω κατηγορήματα αποθηκεύτηκαν σε αντίστοιχα αρχεία, τα οποία θα φορτώσει στην Prolog το κύριο πρόγραμμα.
Το πρόγραμμα φορτώνει τον πελάτη και τα ταξί από τα αρχεία data/client.csv και data/taxis.csv αντίστοιχα.

Prolog

Τα αρχεία Prolog αποτελούν τη βάση γνώσης του προγράμματος. Τα αρχεία Prolog είναι τα nodes.pl, lines.pl, client.pl, taxis.pl, graph.pl, traffic.pl, τα οποία περιέχουν τα προαναφερθέντα κατηγορήματα που δημιουργήθηκαν κατά τη διαδικασία του preprocessing, καθώς και το αρχείο data.pl, το οποίο περιέχει κατηγορήματα χρήσιμα για τη διαχείριση των δεδομένων.

- **distance(Id1, Id2, Time, Dist)**, το οποίο βρίσκει το χρόνο που χρειάζεται ένα ταξί στον κόμβο με αναγνωριστικό Id1 προκειμένου να φτάσει στον κόμβο με αναγνωριστικό Id2, λαμβάνοντας υπόψιν την κίνηση κατά την ώρα Time.
- can_**ride(client(X, Y, Xd, Yd, Time, People, Language), Id)**, το οποίο βρίσκει αν ένα ταξί αναγνωριστικού Id είναι κατάλληλο να εξυπηρετήσει τον δοσμένο πελάτη με People συνεπιβάτες και ομιλούμενη γλώσσα Language. Ο έλεγχος αυτός γίνεται βάσει των People και Language.

- **closest_node(X, Y, Id_ret)**, το οποίο βρίσκει για ένα ταξί ή πελάτη ή τον προορισμό αυτού με συντεταγμένες (X,Υ) το αναγνωριστικό του κοντινότερου σε αυτό/αυτόν κόμβο από τους δοσμένους, προκειμένου να μπορέσει να γίνει η προσομοίωση.

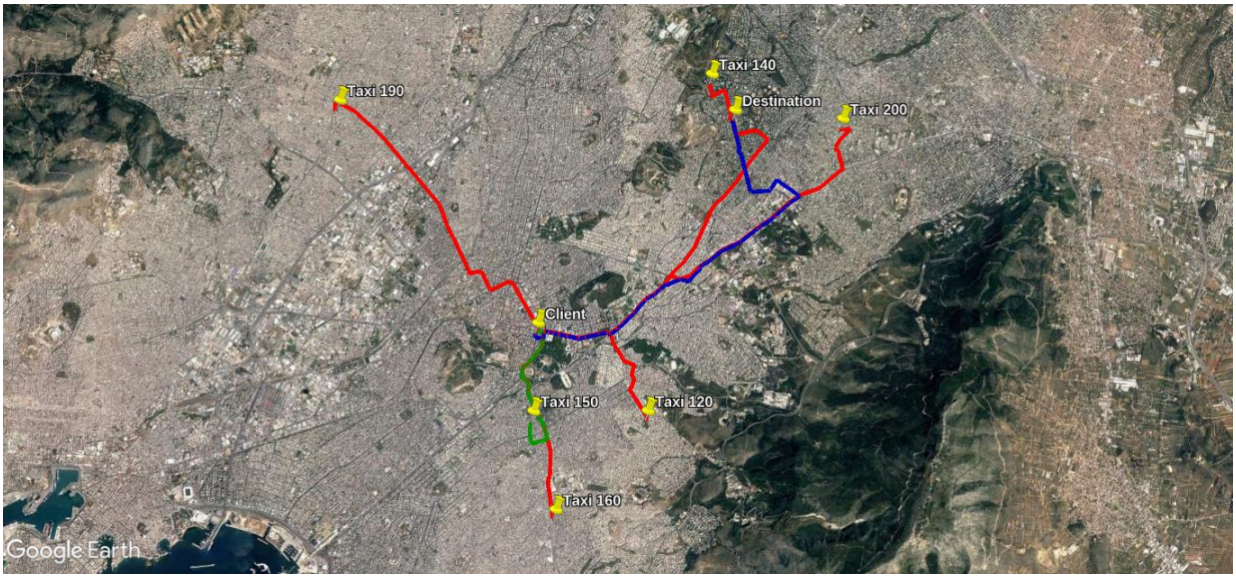
Java

Η σύνδεση Java-Prolog έγινε μέσω των βιβλιοθηκών της JIPProlog. Η λογική του αλγορίθμου σε Java παραμένει σχεδόν ίδια με εκείνη του 1ου θέματος, με διαφορά την εξαγωγή όλων των απαραίτητων πληροφοριών από τα αρχεία Prolog μέσω Queries. Πέρα από τον αλγόριθμο A* και την ευριστική συνάρτηση αυτού, δημιουργήθηκαν συναρτήσεις που εμφανίζουν τα 5 καλύτερα ταξί κατάλληλα για τον πελάτη, που προκύπτουν μετά από ταξινόμηση βάσει του χρόνου άφιξης του ταξί και μετά από ταξινόμηση βάσει της αξιολόγησης.

Αποτελέσματα

Πρώτη δοκιμή

Το KML αρχείο που προέκυψε για το δεδομένο dataset φαίνεται στην παρακάτω απεικόνιση:



Σε αυτήν φαίνεται με πράσινη γραμμή το ταξί που θα φτάσει πιο γρήγορα από όλα στον πελάτη, με κόκκινες τα υπόλοιπα ταξί και με μπλέ η προτεινόμενη διαδρομή από τον πελάτη προς τον προορισμό.

Παρακάτω παρατίθεται το output του προγράμματος που επιστρέφει τα 5 καλύτερα ταξί με βάση τον χρόνο στον οποίο θα φτάσουν όπως και με βάση το rating.

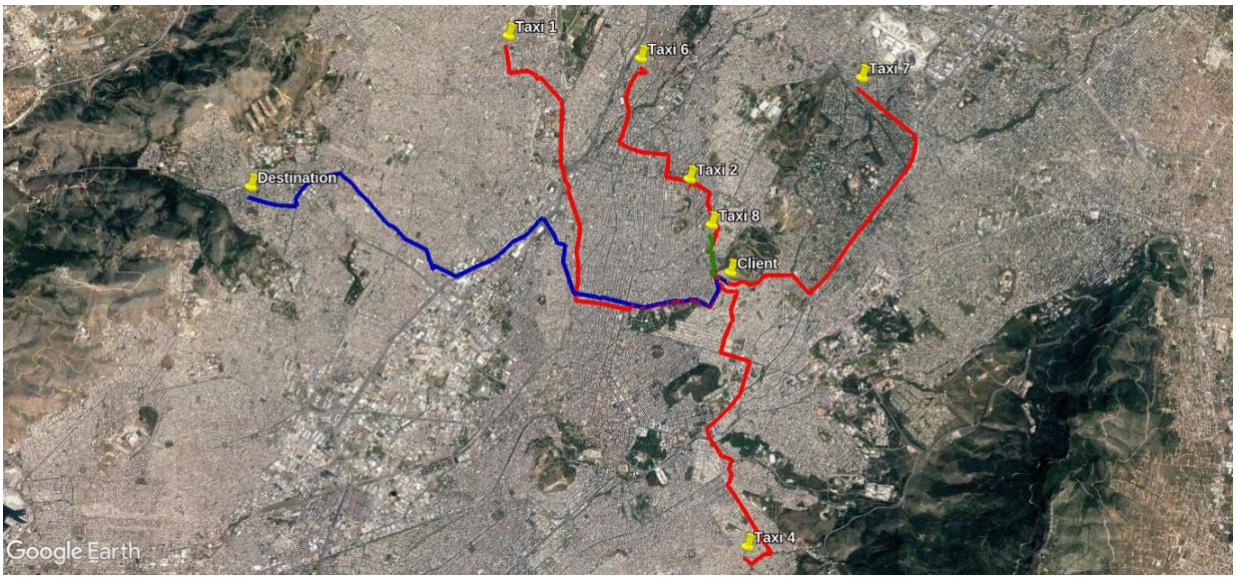
```
Suitable Taxis (sorted by Arrival Time):
#1 Id: 150  Arrival Time: 20:05  Rating: 7.3
#2 Id: 160  Arrival Time: 20:05  Rating: 6.1
#3 Id: 120  Arrival Time: 20:15  Rating: 9.2
```

```
#4 Id: 200  Arrival Time: 20:21  Rating: 5.2
#5 Id: 140  Arrival Time: 20:23  Rating: 7.1

Suitable Taxis (sorted by Rating):
#1 Id: 120  Arrival Time: 20:15  Rating: 9.2
#2 Id: 190  Arrival Time: 20:24  Rating: 7.4
#3 Id: 150  Arrival Time: 20:05  Rating: 7.3
#4 Id: 140  Arrival Time: 20:23  Rating: 7.1
#5 Id: 160  Arrival Time: 20:05  Rating: 6.1
```

Δεύτερη δοκιμή

Το KML αρχείο που προέκυψε για το dataset που δημιουργήσαμε τυχαία φαίνεται στην παρακάτω απεικόνιση:



Σε αυτήν φαίνεται με πράσινη γραμμή το ταξί που θα φτάσει πιο γρήγορα από όλα στον πελάτη, με κόκκινες τα υπόλοιπα ταξί και με μπλέ η προτεινόμενη διαδρομή από τον πελάτη προς τον προορισμό.

Παρακάτω παρατίθεται το output του προγράμματος που επιστρέφει τα 5 καλύτερα ταξί με βάση τον χρόνο στον οποίο θα φτάσουν όπως και με βάση το rating.

```
Suitable Taxis (sorted by Arrival Time):
#1 Id: 8  Arrival Time: 12:08  Rating: 8.0
#2 Id: 2  Arrival Time: 12:14  Rating: 2.0
#3 Id: 7  Arrival Time: 12:20  Rating: 7.0
#4 Id: 6  Arrival Time: 12:29  Rating: 6.0
#5 Id: 4  Arrival Time: 12:31  Rating: 4.0

Suitable Taxis (sorted by Rating):
#1 Id: 8  Arrival Time: 12:08  Rating: 8.0
#2 Id: 7  Arrival Time: 12:20  Rating: 7.0
```

```
#3 Id: 6  Arrival Time: 12:29  Rating: 6.0
#4 Id: 4  Arrival Time: 12:31  Rating: 4.0
#5 Id: 2  Arrival Time: 12:14  Rating: 2.0
```

Τα αρχεία που δημιουργήθηκαν για την δεύτερη δοκιμή έχουν αποθηκευτεί στο directory data με τίτλους client2.csv, taxis2.csv. Να σημειωθεί ότι πρέπει να μετονομαστούν στα αντίστοιχα default για να τα αναγνωρίσει το πρόγραμμα που κάνει το preprocessing.
Τα αρχεία KML που αντιστοιχούν στις παραπάνω εικόνες είναι αποθηκευμένα με ονόματα outfile.kml, outfile2.kml αντίστοιχα.