ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Ανίχνευση κυκλοφορικής συμφόρησης με τεχνικές πληθοπορισμού

Detecting traffic congestion with crowdsourcing techniques

Διπλωματική εργασία

Αλφόνσος Σωτήριος

Επιβλέποντες

Μποζάνης Παναγιώτης

Τσομπανοπούλου Παναγιώτα

Ευχαριστίες

Στο σημείο αυτό θα ήθελα να ευχαριστήσω θερμά τον καθηγητή του τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών και επιβλέποντα της διπλωματικής μου εργασίας κύριο Μποζάνη Παναγιώτη για την ανάθεση του θέματος αυτού και την βοήθεια που μου προσέφερε, όπως επίσης και την επίκουρο καθηγήτρια και συνεπιβλέπουσα της πτυχιακής μου κυρία Τσομπανοπούλου Παναγιώτα. Θα ήθελα επίσης να ευχαριστήσω θερμά τον κύριο Φεύγα Αθανάσιο για την πολύ καλή συνεργασία που είχαμε και για την βοήθεια και υπομονή που έδειξε προκειμένου να προχωρήσει ομαλά η εργασία και να ολοκληρωθεί η εφαρμογή.

Τέλος θα ήθελα να ευχαριστήσω την οικογένεια, του φίλους και την κοπέλα μου που με στήριξαν και μου συμπαραστάθηκαν σε όλη την διάρκεια της εκπόνησης αυτής της εργασίας.

Περίληψη

Σκοπός της εκπόνησης της παρούσης εφαρμογής αποτελεί η αντιμετώπιση του προβλήματος της κυκλοφοριακής συμφόρησης στο οδικό δίκτυο με τη μέθοδο του πληθοπορισμού (crowdsourcing) σε περιβάλλον android. Το φαινόμενο της κυκλοφοριακής συμφόρησης έχει ενταθεί τα τελευταία χρόνια, καθώς, το πλήθος των οχημάτων αυξάνεται ολοένα και περισσότερο με αποτέλεσμα την κατασπατάληση πολύτιμου χρόνου των οδηγών.

Στη μελέτη γίνεται εκτενής αναφορά σε παρόμοιες εφαρμογές, και κυρίως στο front-end αλλά και στο back-end κομμάτι της εφαρμογής που αναπτύχθηκε για την αντιμετώπιση του προβλήματος της κυκλοφοριακής συμφόρησης. Επίσης, παρουσιάζονται οι δυσκολίες που αντιμετώπισε αυτό το εγχείρημα. Τέλος επισημαίνεται η σημασία του ελέγχου και της αποδοτικότητα του αλγορίθμου, καθώς και τα θετικά και αρνητικά στοιχεία που παρουσιάζει.

Περιεχόμενα

Εισαγωγή

Στο πρώτο κεφάλαιο της εργασίας επισημαίνονται πληροφορίες που αφορούν στο προγραμματιστικό περιβάλλον και σε έννοιες που είναι σημαντικές για την καλύτερη κατανόηση του προβλήματος και της εφαρμογής. Θα παρουσιαστεί το πρόβλημα και θα αναλυθούν έννοιες όπως αυτή του πληθοπορισμού.

Στο δεύτερο κεφάλαιο παρουσιάζονται παρεμφερείς εφαρμογές και μέτρα που έχουν παρθεί για την αντιμετώπιση του προβλήματος αυτού τα τελευταία έτη. Επεξηγείται η μεθοδολογία πίσω από κάποιες από αυτές που καταφέρνουν να αντιμετωπίσουν με μεγάλη αποτελεσματικότητα το πρόβλημα.

Συνεχίζοντας παρουσιάζεται και επεξηγείται το interface της εφαρμογής δηλαδή το γραφικό περιβάλλον με το οποίο αλληλεπιδρά ο χρήστης. Επιπροσθέτως, παρουσιάζεται ο τρόπος συλλογής των απαραίτητων δεδομένων για τον υπολογισμό της κυκλοφοριακής συμφόρησης στο οδικό δίκτυο, όπως και τα προβλήματα που αντιμετωπίστηκαν στο στάδιο αυτό.

Το τέταρτο κεφάλαιο αφιερώνεται στην ανάλυση του back-end κομματιού της εφαρμογής. Ειδικότερα, παρουσιάζεται ο αλγόριθμος που χρησιμοποιείται για την εύρεση κυκλοφοριακής συμφόρησης καθώς επίσης και το γραφικό περιβάλλον (web service) όπου προβάλλονται τα αποτελέσματα για να ενημερωθούν οι χρήστες σχετικά με τις περιοχές που έρχονται αντιμέτωπες με το πρόβλημα συμφόρησης.

Στο πέμπτο κεφάλαιο αναφέρεται ο τρόπος επαλήθευσης του αλγορίθμου που αναπτύχθηκε και εξετάζεται η ορθότητα των αποτελεσμάτων

Τέλος, το έκτο κεφάλαιο τονίζει τα πλεονεκτήματα και τα μειονεκτήματα τις συγκεκριμένης προσέγγισης καθώς και τις προοπτικές της για μελλοντική ανάπτυξη.

1. Γενικές πληροφορίες - Βασικές έννοιες

Στο παρόν κεφαλαίο παρέχονται πληροφορίες σχετικά με το android και το πρόβλημα της κυκλοφοριακής συμφόρησης, οι οποίες κρίνεται αναγκαίο να επισημανθούν εκ των προτέρων.

1.1. Το android

Η Android Inc. ιδρύθηκε στο Palo Alto της California τον Οκτώβριο του 2003 από τους Andy Rubin (συνιδρυτή της Danger), Rich Miner (συνιδρυτή της Wildfire Communications, Inc.), Nick Sears (αντιπρόεδρο της T-Mobile), και Chris White (επικεφαλής ανάπτυξης του σχεδιασμού και του interface της WebTV). Η εταιρία λειτουργούσε κρυφά για 2 χρόνια και το 2005 αγοράστηκε από την Google έναντι περίπου 50 εκατομμυρίων διατηρώντας μάλιστα στην εταιρία τους Rubin, Miner και White.

Το Android που αναπτύχθηκε τελικά και κυκλοφόρησε από την Google είναι ένα λειτουργικό σύστημα για κινητά βασισμένο στον Linux kernel. Είναι κυρίως σχεδιασμένο για κινητά αφής όπως είναι τα smartphone και τα tablet που πλέον κατέχουν εξέχουσα θέση στην παγκόσμια αγορά. Η πρώτη του εμπορική έκδοση ήταν το Android 1.0 και έγινε το Σεπτέμβριο του 2008 ενώ μέχρι σήμερα έχουν ακολουθήσει πολλές αναβαθμίσει με ποιο πρόσφατη την έκδοση Android 5.0 "Lollipop" που έγινε διαθέσιμη στις 3 Νοεμβρίου του 2014 [15].

Σύμφωνα με ανακοίνωση της Google τον Σεπτέμβριο του 2013 το λειτουργικό σύστημα android είναι ενσωματωμένο σε περισσότερες από 1 δισεκατομμύριο συσκευές. Στις συσκευές αυτές συμπεριλαμβάνονται πλέον και τηλεοράσεις (Android TV), αυτοκίνητα (Android Auto) και ρολόγια χειρός (Android Wear). Επιπροσθέτως το Google play που λειτουργεί ως το επίσημο app store του android, από το 2013 φιλοξενεί περισσότερες από 1 εκατομμύριο εφαρμογές [16].

Σύμφωνα με στοιχεία της IDC [14] για το οικονομικό έτος 2014 το λειτουργικό android έχει κατακτήσει το μεγαλύτερο μερίδιο της αγοράς με τους κύριους ανταγωνιστές του iOS και windows phone να είναι αρκετά χαμηλότερα (Εικόνα 1). Το γεγονός αυτό, για πολλούς developers, σύμφωνα με στοιχεία του 2015, έχει θέσει την ανάπτυξη εφαρμογών σε android(40%) σημαντικότερη από ανάπτυξη εφαρμογών για λειτουργικά όπως iOS(37%) και άλλα.

Operating System	2014 Unit Volumes	2014 Market Share	2013 Unit Volumes	2013 Market Share	Year-Over-Year Change
Android	1,059.3	81.5%	802.2	78.7%	32.0%
iOS	192.7	14.8%	153.4	15.1%	25.6%
Windows Phone	34.9	2.7%	33.5	3.3%	4.2%
BlackBerry	5.8	0.4%	19.2	1.9%	-69.8%
Others	7.7	0.6%	2.3	0.2%	234.8%
Total	1,300.4	100.0%	1,018.7	100.0%	27.7%

(Εικόνα 1) πηγή: http://www.idc.com/getdoc.jsp?containerId=prUS25450615

1.2. Τα εργαλεία

Τα εργαλεία που χρησιμοποιήθηκαν διατίθενται όλα δωρεάν και χωρίζονται σε δυο κυρίως κατηγορίες:

- για την ανάπτυξη της εφαρμογής σε περιβάλλον android (Android Studio)
- για το server και τη βάση δεδομένων που δοκιμάστηκαν σε τοπικό δίκτυο και για την ιστοσελίδα στην οποία παρουσιάζονται τα αποτελέσματα (NetBeans και xampp)

Επιπροσθέτως τόσο στην εφαρμογή όσο και στον server έγινε χρήση δυο api της Google (directions, roads). Application programming interface (API) είναι ένα σύνολο από συναρτήσεις, πρωτόκολλα και εργαλεία που χρησιμοποιείται από software εφαρμογές. Ένα api προσφέρει δυνατότητες ανεξάρτητες από την εκάστοτε εφαρμογή, και συνήθως είτε καλείται απομακρυσμένα είτε ενσωματώνεται στην εφαρμογή σαν βιβλιοθήκη [26].

1.2.1. Android Studio

Το προγραμματιστικό περιβάλλον στο οποίο αναπτύχθηκε το front-end κομμάτι της εφαρμογής είναι το Android Studio. Το περιβάλλον αυτό είναι το πλέον διαδεδομένο για ανάπτυξη android εφαρμογών και έχει ξεπεράσει τα τελευταία χρόνια τον προκάτοχο του

eclipse. Παρέχει ένα πολύ εύχρηστο interface ενώ επίσης ένα .gradle αρχείο περιέχει όλα τα δεδομένα για να γίνει η εκτέλεση του κώδικα (βιβλιοθήκες κλπ.). Το γεγονός αυτό δίνει τη δυνατότητα μεταφοράς του κώδικα σε νέο μηχάνημα με απλό και εύκολο τρόπο σε αντίθεση με το eclipse.

Το Android Studio ανακοινώθηκε το Μάιο του 2013 από τη Google και η πρώτη του επίσημη έκδοση κυκλοφόρησε το Δεκέμβριο του 2014 με την έκδοση 1.0 στην οποία βρίσκεται μέχρι και σήμερα. Από την ημέρα που ξεκίνησε η κυκλοφορία του η χρήση του παραμένει δωρεάν μαζί με το SDK (software development kit) όπου παρέχει όλες τις βιβλιοθήκες που είναι απαραίτητο για να ξεκινήσει κάποιος τον προγραμματισμό μιας android εφαρμογής [27].

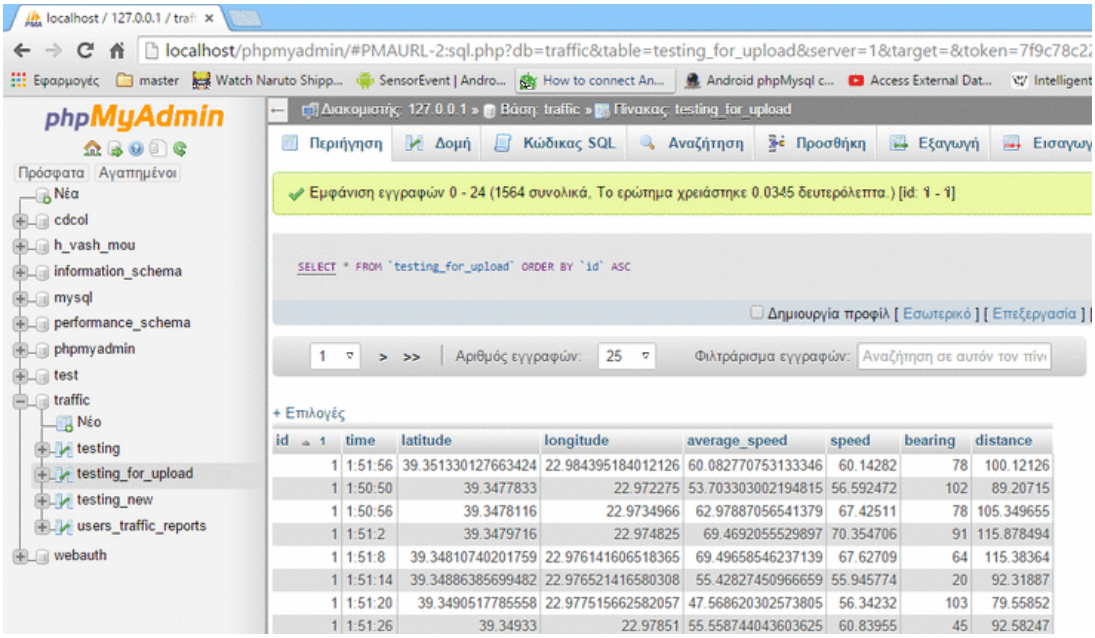
Επιπροσθέτως για την δοκιμή του κώδικα παρέχεται emulator από το Android Studio το οποίο προσομοιώνει στον υπολογιστή μας ένα κινητό τηλέφωνο με λειτουργικό android για να δούμε την λειτουργία του κώδικα μας σε αυτό. Ωστόσο σε εφαρμογές που χρησιμοποιούν το gps για την τοποθεσία του χρήστη όπως η δική μας, το emulator δεν βοηθάει και πρέπει να γίνουν δοκιμές σε πραγματικά κινητά τηλέφωνα.

1.2.2. NetBeans και xampp

Το NetBeans [18] είναι μια πλατφόρμα για ανάπτυξη software εφαρμογών. Ξεκίνησε το 1996 από έναν φοιτητή με την καθοδήγηση του τμήματος μαθηματικών και φυσικής στο Charles University της Πράγας. Σήμερα έχει αγοραστεί πλέων από την Oracle και βρίσκεται στην έκδοση 8.0. Χρησιμοποίησα την πλατφόρμα αυτή για την ανάπτυξη της ιστοσελίδας που παρουσιάζει τα αποτελέσματα του αλγορίθμου μας και δείχνει τα σημεία που έχει κίνηση στο χάρτη.

Η ιστοσελίδα για να λειτουργήσει και να δοκιμαστεί χρειάζεται έναν server και μια βάση δεδομένων. Η βάση δεδομένων αποθηκεύει τα δεδομένα που λαμβάνουμε από τους χρήστες και στη συνέχεια τα διαβάζει ο αλγόριθμος μας για τους υπολογισμούς που πρέπει να γίνουν. Τόσο ο server όσο και η database στήθηκαν τοπικά στον υπολογιστή μου με τη βοήθεια του xampp [28]. Το xampp δημιουργεί έναν τοπικό Apache HTTP server και μια τοπική βάση δεδομένων phpmyadmin.

Το site είναι προσβάσιμο μέσω του url (*http://localhost:8084/traffic_server_side*) ενώ η βάση μας με το σύνδεσμο (*http://localhost/phpmyadmin/*)



(Εικόνα 2) Βάση δεδομένων (http://localhost/phpmyadmin/)

1.3. Τι είναι πληθοπορισμός (crowdsourcing)

Πληθοπορισμός ή crowdsourcing είναι η διαδικασία απόκτησης πληροφοριών ή ιδεών με τη βοήθεια μιας μεγάλης ομάδας ατόμων αντί η συλλογή αυτή να γίνει από εργαζομένους η προμηθευτές. Σύμφωνα μα την έννοια αυτή, μια πάρα πολύ μεγάλη ομάδα ατόμων, συγκεντρώνει πληροφορίες ή δεδομένα (στην δική μας περίπτωση) που με παραδοσιακές τεχνικές δεν θα μπορούσαν να συγκεντρωθούν.

Η έννοια του πληθοπορισμού δημιουργήθηκε το 2005 και μπορεί να προσδιορίσει ένα μεγάλο σύνολο δραστηριοτήτων. Μπορεί να περιλαμβάνει διαμοιρασμό εργασίας για περιπτώσεις μεγάλου όγκου δεδομένων, έρευνα για εύρεση απάντησης σε κάποιο ερώτημα, ή και ακόμα για την εύρεση κάποιου ατόμου που αγνοείται [29]. Το crowdsourcing είναι η βασική έννοια πίσω από όλα τα κοινωνικά δίκτυα (Facebook, tweeter) και επίσης πίσω από αρκετά γνωστά site όπως είναι το YouTube.

Από το crowdsourcing πηγάζει η πολύ σημαντική έννοια του crowdfunding. Βάση της τεχνικής αυτής μια εργασία συγκεντρώνει μικρά χρηματικά ποσά από διάφορα άτομα προκειμένου να υλοποιηθεί. Σημαντικό παράδειγμα είναι το Kickstarter.

1.4. Το πρόβλημα της κυκλοφοριακής συμφόρησης

Σύμφωνα με στατιστικές από το Υπουργείο συγκοινωνιών της Αμερικής το 2007, περίπου 4,2 δισεκατομμύρια ώρες σπαταλήθηκαν από τους οδηγούς στην κίνηση μόνο στις εθνικές οδούς. Βάση έρευνας που έγινε από τους [7], αρκετές πόλεις ανά τον κόσμο παρουσίασαν μεγάλη αύξηση στην κυκλοφορία ιδιωτικών οχημάτων τα τελευταία χρόνια. Για το λόγο αυτό πολλές προσπάθειες έχουν γίνει για την εύρεση λύσεων στο πρόβλημα της κυκλοφοριακής συμφόρησης. Η κατάσταση παρουσιάζει σημαντική επιδείνωση κυρίως σε ανεπτυγμένες περιοχές για τους ακόλουθους λόγους:

Ανεπαρκές σχέδιο πόλεων ή μικροί και πρόχειρα κατασκευασμένοι δρόμοι. Καθώς οι πόλεις επεκτείνονται, σπάνια γίνεται πρόβλεψη για την βελτίωση του οδικού δικτιού. Αυτό έχει ως αποτέλεσμα σε αρκετές οδικές αρτηρίες να παρουσιάζεται συχνά κυκλοφοριακή συμφόρηση η οποία μάλιστα διαρκεί αρκετές ώρες. Επίσης, καθώς στις περισσότερες πόλεις όταν κατασκευαζόταν το οδικό δίκτυο ήταν αδύνατο να υπάρξει πρόβλεψη για τον αριθμό των οχημάτων που σήμερα κυκλοφορούν, οι δρόμοι είναι κατασκευασμένοι με σκοπό την εξυπηρέτηση περιορισμένου αριθμού οχημάτων.

Κανόνες οδικής συμπεριφοράς. Οι οδηγοί πολλές φορές πολλές φορές και για ποικίλους λόγους δεν ακολουθούν σωστά τις λωρίδες κυκλοφορίας. Η επίδραση του γεγονότος αυτού, ειδικά σε διασταυρώσεις με αυξημένη κίνηση, επιδεινώνει την κυκλοφορία προκαλώντας συμφόρηση. Επιπροσθέτως, οι οδηγοί συχνά παραβιάζουν κόκκινους σηματοδότες και μπλοκάρουν τις διασταυρώσεις επιδεινώνοντας ακόμα περισσότερο το πρόβλημα. Τα προβλήματα αυτά σε συνδυασμό με το γεγονός ότι η επιβολή του νόμου σε τροχαίες παραβάσεις είναι ανεπαρκής, δεν παρέχουν κίνητρο για τους οδηγούς για να ακολουθήσουν τους κανόνες.

Μέσα μαζικής μεταφοράς. Χώρες με γρήγορα αυξανόμενες οικονομίες έχουν παρουσιάζει σημαντική αύξηση οχημάτων στις κεντρικές τους πόλεις. Αρκετές από τις πόλεις αυτές έχουν ανεπαρκή ή μη αποδοτικά δίκτυα μέσων μαζικής μεταφοράς, οδηγώντας τους ανθρώπους να χρησιμοποιούν κατά κύριο λόγο τα οχήματα τους. Το πρόβλημα ενισχύεται από την πλευρά της κοινωνίας, όπου οι άνθρωποι αντιλαμβάνονται την χρήση ιδιωτικού οχήματος ως δείγμα ευημερίας, ενώ τα μέσα μαζικής μεταφοράς κρίνεται ότι χρησιμοποιούνται από τα κατώτερα κοινωνικά στρώματα.

Διοίκηση του οδικού δικτύου. Σε αρκετές πόλεις το πρόβλημα της κυκλοφοριακής συμφόρησης προκαλείται από τους φωτεινούς σηματοδότες όπου δεν έχουν αυτόματα ρυθμιζόμενου χρόνους, αλλά σταθερούς. Οι χρόνοι με τους οποίους αυτοί λειτουργούν εξυπηρετούν εύκολα συνθήκες ομαλής κυκλοφορίας αλλά σε ώρες αιχμής δεν είναι αποδοτικοί με αποτέλεσμα οι φωτεινοί σηματοδότες που σκοπός τους είναι η διευκόλυνση της κυκλοφορίας, να ευθύνονται για την δημιουργία κυκλοφοριακής συμφόρησης. Το πρόβλημα αυτό αντιμετωπίζεται μερικώς κάποιες φορές όταν η αστυνομία επεμβαίνει προσπαθώντας να εξομαλύνει την κυκλοφορία.

Κρατικοί προϋπολογισμοί. Ένας σημαντικός αριθμός επενδύσεων απαιτείται για να κατασκευαστεί υποδομή διαχείρισης της κίνησης που να μπορεί να ανταποκριθεί σε μικρή αλλά και αυξημένη συμφόρηση. Μια τέτοια υποδομή δεν αρκεί μόνο να περιλαμβάνει τη μέτρηση και ανάλυση της κίνησης σε πραγματικό χρόνο αλλά πρέπει και να εστιάζει στον εντοπισμό κυκλοφοριακής συμφόρησης, την επίλυση της, αλλά και την πρόβλεψη μελλοντικών συμβάντων.

Αντιλαμβανόμενοι λοιπόν το πρόβλημα και πόσο καίριο ρόλο παίζει στις ζωές των ανθρώπων διακρίνουμε και τη ανάγκη για ανάπτυξη μιας λύσεις που θα το αντιμετωπίσει.

2. Σχετικές Έρευνες

Το πρόβλημα του εντοπισμού, στις σύγχρονες κοινωνίες, της κυκλοφοριακής συμφόρησης, με αυτόματους και έξυπνους τρόπους, χωρίς χρονική καθυστέρηση έχει αποτελέσει αντικείμενο πολλών ερευνών και τη βάση ανάπτυξης εφαρμογών με διαφοροποιήσεις στον τρόπο προσέγγισης του φαινομένου. Έτσι λοιπόν, στο σημείο αυτό παρατίθενται κάποιες προσεγγίσεις για την αντιμετώπιση του συγκεκριμένου προβλήματος, οι οποίες χωρίζονται σε δυο κατηγορίες, βάση της μεθόδου που ακολουθήθηκαν για τον εντοπισμό της κυκλοφοριακής συμφόρησης.

2.1 Crowdsourcing τεχνικές για κινητά με gps εντοπισμό

Αρχικά, πάνω στο αντικείμενο με το οποίο ασχοληθήκαμε (δηλαδή εντοπισμό της κυκλοφοριακής συμφόρησης με τη βοήθεια των χρηστών μέσω gps εντοπισμού και crowdsourcing τεχνικών) έχουν γίνει έρευνες που μελετούν και τεκμηριώνουν διάφορες πτυχές του προβλήματος, όπως το τι θεωρούμε ως κυκλοφοριακή συμφόρηση, ποιος είναι ο καλύτερος τρόπος εντοπισμού της θέσης του χρήστη μελετώντας power consumption vs accuracy κλπ. Παρόλα αυτά, εκτενής αναφορά θα πραγματοποιηθεί μόνο σε ολοκληρωμένες εφαρμογές που προσφέρουν πληροφορίες για την προσέγγιση του προβλήματος, όπως το VTrack [1], το Nericell [2] , καθώς επίσης και σε μια μελέτη που έχει γίνει πρόσφατα για τον εντοπισμό κυκλοφοριακής συμφόρησης με Vehicular Ad-hoc Networks (VANET) [5].

Τέλος, θα παρουσιάσουμε εν συντομία τρεις από τις γνωστότερες εφαρμογές android. Το Waze [4], το Google Traffic [3] και το TomTom [9] που παράλληλα με πολλές άλλες λειτουργίες, εντοπίζουν και ενημερώνουν τους χρήστες σχετικά με τα σημεία που παρουσιάζουν συμφόρησης.

i. VTrack

Οι συγγραφείς όρισαν ως hotspot ένα κομμάτι δρόμου στο οποίο παρατηρείται ότι ο χρόνος ταξιδιού υπερβαίνει κατά ένα συγκεκριμένο ποσοστό το χρόνο που έχει προβλεφθεί βάση του ορίου ταχύτητας. Τα hotspots δεν αποτελούν άχρηστα δεδομένα στην κίνηση, για την ακρίβεια εμφανίζονται καθημερινά στην ώρα αιχμής, για παράδειγμα όταν οι οδηγοί έχουν κολλήσει στην κίνηση.

Στόχος της εφαρμογής είναι ο εντοπισμός και η εμφάνιση όλων των hotspot σε μια γεωγραφική περιοχή, όπου ο χρήστης θα μπορεί να τα δει μέσω της ιστοσελίδας. Ο εντοπισμός αυτός επιτυγχάνεται καθώς ο κάθε χρήστης στέλνει στο server περιοδικά τη διαδρομή που ακολουθεί και στη συνέχεια ο server υπολογίζει τον χρόνο που έκανε ο οδηγός για να τη διανύσει και βρίσκει τα hotspots (σημεία που η χρονική καθυστέρηση του χρήστη υπερβαίνει αυτής που υπολογίζει ο server). Οι συγγραφείς ακόμα όπως αναφέρουν επιδίωξαν και κατάφεραν να κρατήσουν δυο βασικές τιμές χαμηλά στον υπολογισμό των hotspots: το miss rate όπου είναι το σύνολο των hotspot που δεν καταγράφηκαν και το false positive rate που είναι το σύνολο των σημείων που καταγράφηκαν ως hotspot αλλά δεν ήταν. Έτσι, η εφαρμογή αυτή μπορεί να χρησιμοποιηθεί απευθείας από χρήστες όπου θα μπορούν να δουν τα hotspot στη περιοχή τους και τα αποφύγουν αλλάζοντας κατάλληλα τη διαδρομή που θα ακολουθήσουν.

Στο συγκεκριμένο project οι Arvind Thiagarajan, Lenin Ravindranath, Katrina LaCurts, Sivan Toledo, Jakob Eriksson, Samuel Madden, Hari Balakrishnan ασχολήθηκαν κυρίως με τον ακριβή εντοπισμό της θέσης του χρήστη και της διαδρομής που ακολούθησε για την εύρεση του χρόνου που έκανε για την ολοκλήρωση της.

Οι συγγραφείς εστίασαν την προσοχή τους στην υψηλή κατανάλωση ενέργειας που έχει η χρήση του gps στο κινητό και προσπάθησαν να τεκμηριώσουν μια βέλτιστη λύση δοκιμάζοντας διάφορες προσεγγίσεις. Αναφερόμενοι εκτενώς σε δειγματοληψίες ανά διαφορετικά χρονικά διαστήματα αλλά και με wifi μόνο ή gps, καταλήγουν να προσαρμόσουν τη δειγματοληψία στα 30 δευτερόλεπτα με gps, κάτι που δε θα επηρέαζε την ποιότητα των δεδομένων και θα είχε υψηλή απόδοση όσον αφορά στην κατανάλωση ενέργειας.

Στη συνέχεια, επιχείρησαν να ορίσουν τμήματα δρόμου ως hotspots βάση των ορίων ταχύτητας που πήραν από την NAVTEQ road database και μιας τιμής k που προσαρμόζει το όριο ταχύτητας. Επειδή όπως παρατήρησαν, τα οχήματα δεν κινούνται ακριβώς με βάση το όριο ταχύτητας, δίνοντας στο k τιμή περίπου ίση με 0,67, κατάφεραν να προσαρμόσουν τα όρια ταχύτητας σε ποσοστό το οποίο κάλυπτε περίπου το 67% των οδηγών. Με βάση λοιπόν αυτά τα πλέον προσαρμοσμένα όρια ταχύτητας, όρισαν hotspot σε σημεία όπου παρατηρούνταν μεγάλη χρονική καθυστέρηση από τους χρήστες συγκρίνοντας τη με την καθυστέρηση που θα έβρισκε ο server για την εκάστοτε διαδρομή.

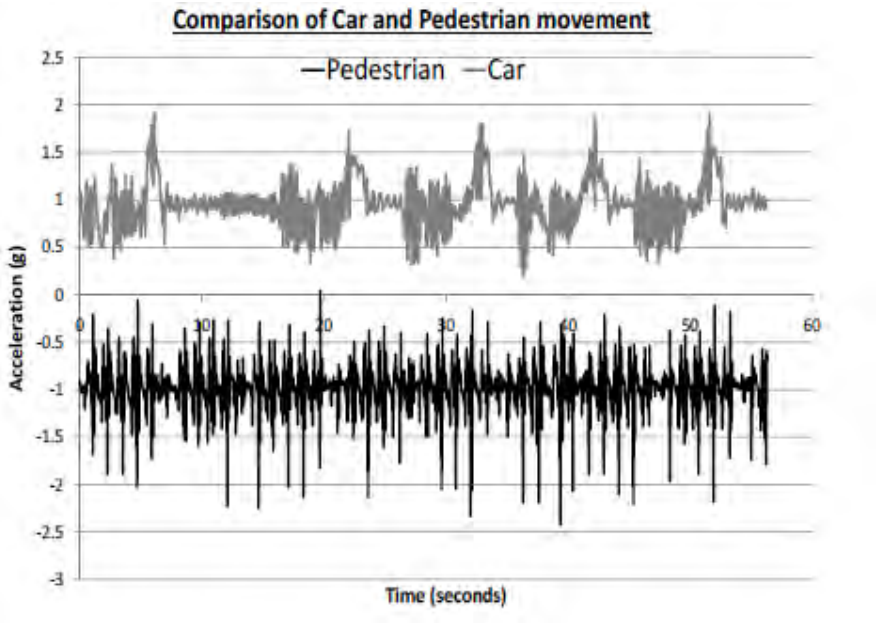
ii. Nericell

Η συγκεκριμένη εφαρμογή εστιάζει κυρίως στη χρήση του επιταχυνσιόμετρου, που πλέον σχεδόν όλα τα κινητά έχουν ενσωματωμένο, μικροφώνου, GSM ραδιοφώνου και gps με σκοπό να εντοπίζει τον θόρυβο της κίνησης, την κατάσταση του οδοστρώματος και πότε ο χρήστης επιταχύνει ή επιβραδύνει.

Μια από τις δυσκολίες της συγκεκριμένης εφαρμογής όπως αναφέρουν οι συγγραφείς είναι ο εικονικός επαναπροσανατολισμός του επιταχυνσιόμετρου για να αντισταθμίσει τον γενικό προσανατολισμό που αυτό έχει αρχικά. Επίσης, δίνοντας προτεραιότητα στην ενεργειακή αποδοτικότητα αλλά και το κόστος επικοινωνίας κινητού-server επέλεξαν να επεξεργάζονται να δεδομένα που λαμβάνουν οι αισθητήρες, τοπικά, και να στέλνουν τα αποτελέσματα στο Server.

Το επιταχυνσιόμετρο έχει τρις άξονες(X,Y,Z). Στην αρχή κάθε διαδρομής και όταν ξεκινάει η εφαρμογή, ανάλογα με τον τρόπο που είναι τοποθετημένο το κινητό τρέχει ένας αλγόριθμος όπου ρυθμίζει σωστά τους άξονες x, y, z με βάση την επιτάχυνση λόγο της βαρύτητας g. Όταν το επιταχυνσιόμετρο έχει αρχικοποιηθεί η εφαρμογή λαμβάνει μετρήσεις για την επιτάχυνση και την επιβράδυνση με βάση τις τιμές στον x άξονα. Γενικά, το φρενάρισμα θα προκαλούσε μια απότομη αύξηση στην τιμή του x επειδή το επιταχυνσιόμετρο θα δεχόταν μια δύναμη να το σπρώχνει προς τα μπροστά. Η απότομη αυτή αύξηση της τιμής μπορεί να σημειωθεί ακόμα και όταν το όχημα κινείται με χαμηλή ταχύτητα. Αν ένα όχημα κινείται με ταχύτητα 10 kmph και σταματήσει σε 1 δευτερόλεπτο, αυτό θα προκαλούσε μια μέση αύξηση της τιμής του x κατά 0,28g ή και ακόμα μεγαλύτερη άνοδο.

Ακόμα, με την χρήση του επιταχυνσιόμετρου κατάφεραν να διαχωρίσουν πότε ο χρήστης κινείται πεζός ή με αυτοκίνητο. Ο διαχωρισμός φαίνεται στην (Εικόνα 3).



(Εικόνα 3) πηγή: Nericell: rich monitoring of road and traffic conditions using mobile smartphones

Τέλος, σημαντικό κομμάτι της εφαρμογής και κάθε εφαρμογής τέτοιου τύπου είναι ο εντοπισμός της θέσης του χρήστη. Ο τρόπος που ακολουθήθηκε στο Nericell είναι ο εντοπισμός μέσω gsm, δηλαδή, βρίσκει τις συντεταγμένες του τηλεφώνου από πύργους ραδιοφωνίας. Η μέθοδος αυτή είναι αρκετά αποδοτική από άποψη ενεργειακού κόστους αλλά προσφέρει ακρίβεια περίπου 100 μέτρα (δηλαδή η θέση που θα δώσει να είναι 100 μέτρα μακριά από την πραγματική τοποθεσία του χρήστη).

iii. VANET

Τα Vehicular Ad-hoc Networks (VANET) είναι ασύρματα δίκτυα που υποστηρίζουν ανταλλαγή πληροφοριών μεταξύ κινητών οχημάτων χωρίς να χρειάζεται υποδομή για σταθερό ίντερνετ. Το VANET έχει αποτελέσει ακρογωνιαίο λίθο για το όραμα του έξυπνου δικτύου μεταφορών (Intelligent Transportation System (ITS)).

Οι συγγραφείς προτείνουν ένα καινοτόμο σύστημα επικοινωνίας οχημάτων για τον εντοπισμό της κυκλοφοριακής συμφόρησης από πληροφορίες που συλλέγονται μέσω επικοινωνίας μεταξύ των οχημάτων. Αυτή η πληροφορία επεξεργάζεται στιγμιαία από τα οχήματα, κάτι που τους επιτρέπει να εντοπίσουν τα σημεία κυκλοφοριακής συμφόρησης μέσα σε κατοικημένες περιοχές, και στη συνέχεια να υπολογίσουν μια εναλλακτική διαδρομή

με λιγότερη κίνηση. Στην προσέγγιση αυτή, τα οχήματα παίζουν το ρόλο του κινητού αισθητήρα και συνεχώς καταμετρούν την συμφόρηση του δρόμου υπολογίζοντας την ταχύτητα τους και τους χρόνους ταξιδιού. Αυτή η πληροφορία διανέμεται μεταξύ των υπολοίπων οχημάτων μέσω ενός απλού geocast μοντέλου[6].

Οι συγγραφείς, όπως αναφέρουν, ακολούθησαν την παρακάτω προσέγγιση για τον εντοπισμό κυκλοφοριακής συμφόρησης. Όταν ένα όχημα πλησιάζει μια συγκεκριμένη διασταύρωση, χρησιμοποιεί το geocast για να εκπέμψει ένα αίτημα προς τα άλλα οχήματα στην κοντινή του περιοχή (ZOR (Zone of Relevance)) σχετικά με την ύπαρξη ή όχι κυκλοφοριακής συμφόρησης. Τα οχήματα της γύρω περιοχής απαντούν με το βαθμό κίνησης που έχουν, το κάθε ένα στο σημείο που βρίσκεται. Στη συνέχεια, το όχημα που έκανε την αρχική αίτηση, λαμβάνει έναν πίνακα με δεδομένα σχετικά με το βαθμό συμφόρησης σε διάφορα σημεία της γύρω περιοχής. Ενημερώνει τη βάση δεδομένων του με τα τμήματα του δρόμου για τα οποία έλαβε δεδομένα και διατηρεί τις τιμές που είχε για σημεία για τα οποία δεν έλαβε ενημέρωση. Όμως, επειδή περισσότερα από ένα οχήματα μπορεί να στείλουν πληροφορίες συμφόρησης για το ίδιο οδικό τμήμα, λαμβάνεται υπόψη ο μέσος όρος των πληροφοριών που θα ληφθούν.

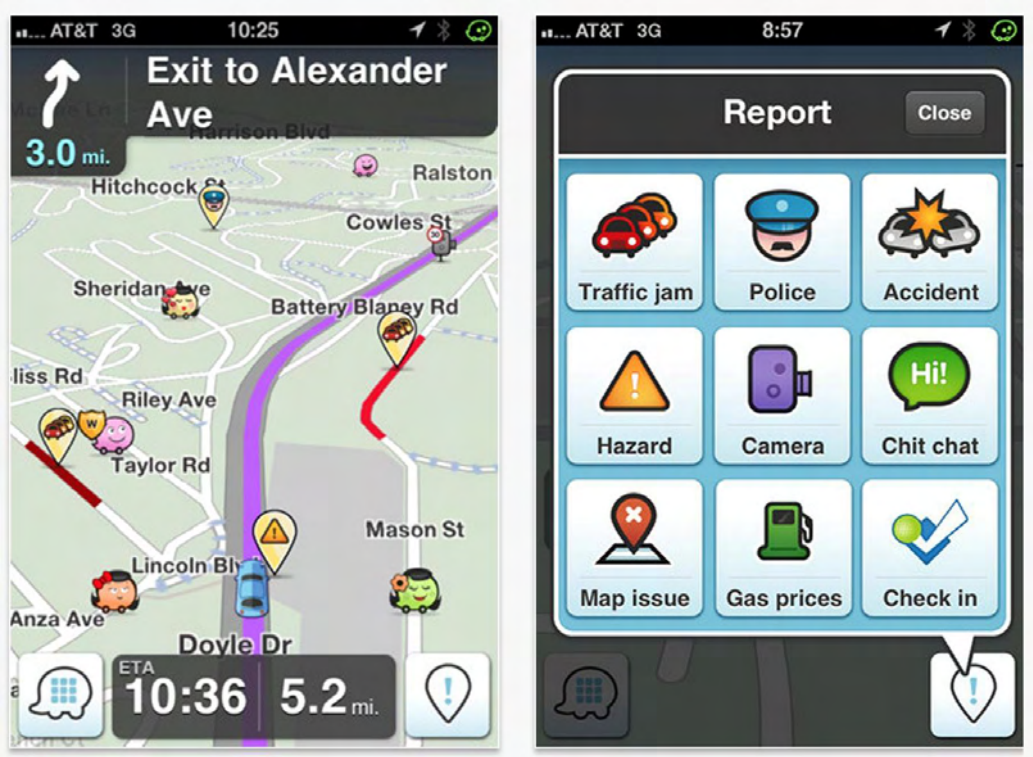
Αφού ενημερωθεί η βάση δεδομένων του οχήματος, δίνει τις τιμές της ως είσοδο για τον αλγόριθμο που θα υπολογίσει τη διαδρομή με τη λιγότερη κυκλοφοριακή συμφόρηση καταλήγοντας στον επαναπροσδιορισμό μιας νέας διαδρομής για το όχημα.

iv. Waze

Το Waze αποτελεί την πιο διαδεδομένη εφαρμογή σε android όσον αφορά στον εντοπισμό κυκλοφοριακής συμφόρησης και στην πλοήγηση με 2,5 περίπου εκατομμύρια download στο Play Store. Αποτελεί τη μεγαλύτερη, σχετιζόμενη με την κίνηση στο οδικό δίκτυο, κοινότητα. Όσο ο χρήστης είναι συνδεδεμένος και λειτουργεί την εφαρμογή, αυτή συλλέγει δεδομένα. Όταν εντοπιστεί συμφόρηση ο χρήστης ενημερώνει με το πάτημα ενός κουμπιού και στη συνέχεια ενημερώνονται και οι υπόλοιποι χρήστες της εφαρμογής [20].

Η δημοφιλής αυτή εφαρμογή δίνει ακόμα τη δυνατότητα στους χρήστες να επικοινωνήσουν με φίλους τους, να ενημερώσουν πόσο χρόνο χρειάζονται για να φτάσουν στον προορισμό τους, να ενημερώσουν τους άλλους χρήστες για μπλόκα της αστυνομίας ή για ατύχημα σε κάποιο σημείο του οδικού δικτύου κλπ (Εικόνα 4). Όλα αυτά σε συνδυασμό με το εύκολο

και φιλικό προς το χρήστη γραφικό περιβάλλον την καθιστούν ως τη νούμερο ένα εφαρμογή για πλοήγηση και εντοπισμό συμφόρησης σε δρόμους σε ολόκληρο τον κόσμο.



(Εικόνα 4) Η εφαρμογή Waze.

Πηγή: http://theapplegoogle.com/wp-content/uploads/2012/09/waze.jpg

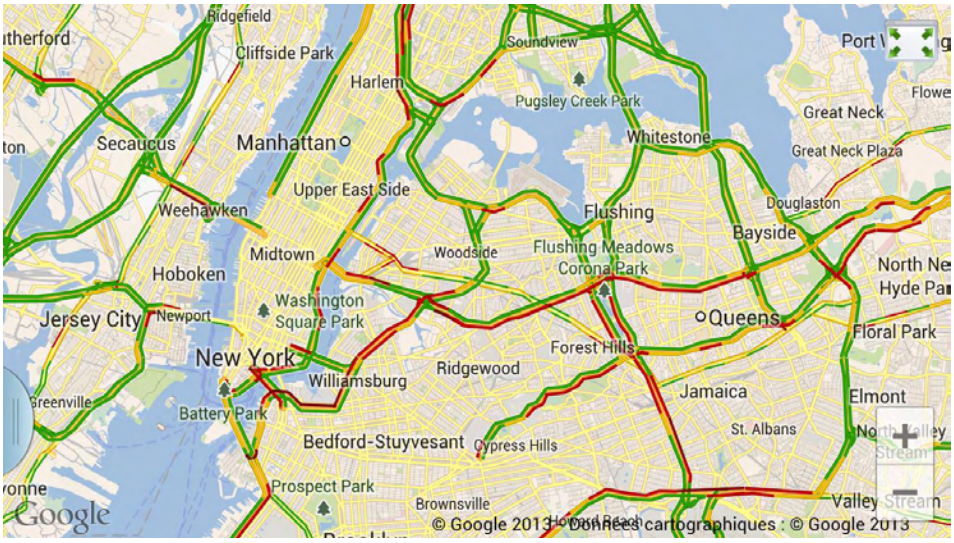
Τέλος, είναι σημαντικό να σημειωθεί πως σε αντίθεση με το google maps, που δεν καλύπτει πολλές χώρες όπως την Ελλάδα ή άλλες βαλκανικές χώρες, η συγκεκριμένη εφαρμογή καλύπτει τις περισσότερες πόλεις του κόσμου συμπεριλαμβανομένων και τις Αθήνας και της Θεσσαλονίκης.

v. Google Traffic

Το Google Traffic είναι μια λειτουργία του Google Maps που εμφανίζει τις συνθήκες κίνησης σε πραγματικό χρόνο σε κεντρικούς δρόμου και εθνικές οδούς σε πάνω από 50 χώρες. Μπορεί κανείς να κάνει χρήση της υπηρεσίας στο site του Google Maps, ή χρησιμοποιώντας την εφαρμογή του Google Maps για κινητό.

Το Google Traffic (Εικόνα 5) λειτουργεί αναλύοντας τις gps τοποθεσίες που του μεταδίδονται από ένα μεγάλο αριθμό χρηστών κινητών τηλεφώνων. Υπολογίζοντας την

ταχύτητα των χρηστών κατά μήκους ενός τμήματος δρόμου, η Google είναι σε θέση να παράγει έναν live χάρτη κίνησης. Για να το πετύχει αυτό, επεξεργάζεται τα εισερχόμενα ακατέργαστα δεδομένα για τις τοποθεσίες των κινητών συσκευών, και εν συνεχεία αποκλείει ανωμαλίες όπως το ταχυδρομικό όχημα που κάνει συχνές στάσεις. Όταν ένα όριο από χρήστες σε μια συγκεκριμένη περιοχή παρατηρηθεί, το περίγραμμα κατά μήκος των δρόμων και των εθνικών οδών στο χάρτη της Google αλλάζει χρώμα.



(Εικόνα 5) πηγή: http://yalantis.com/media/content/ckeditor/2014/11/unnamed.png

Το Google Traffic είναι διαθέσιμο επιλέγοντας "Traffic" από ένα drop-down μενού στο Google Maps. Τότε οι κεντρικοί δρόμοι και οι εθνικές οδοί αποκτούν χρώμα, το πράσινο αντιπροσωπεύει κανονικές ταχύτητες κίνησης, το κίτρινο αργές συνθήκες κυκλοφορίας και το κόκκινο τη μικρότερη ταχύτητα, δηλαδή κυκλοφοριακή συμφόρηση. Το γκρι υποδηλώνει ότι δεν έχουν συγκεντρωθεί δεδομένα για το συγκεκριμένο σημείο. Οι χρήστες μπορούν να χρησιμοποιήσουν το "search" για να δουν την κίνηση σε μια συγκεκριμένη περιοχή. Για παράδειγμα, ο χρήστης μπορεί να πληκτρολογήσει "traffic near Edmonton, Alberta" στην αναζήτηση, για να δει την κίνηση σε μια συγκεκριμένη περιοχή. Τέλος μια ακόμα δυνατότητα που παρέχεται είναι με τη βοήθεια ιστορικών δεδομένων να δείξει στου χρήστες τη "συνηθισμένη κίνηση" για μια συγκεκριμένη ώρα και μέρα της εβδομάδας.

vi. TomTom

Το TomTom είναι μια Ολλανδική εταιρία που παράγει συστήματα πλοήγησης. Η εταιρία ιδρύθηκε το 1991 και αρχικά κατασκεύαζε εφαρμογές για ανάγνωση bar-code. Ακολουθώντας την πορεία της αγοράς και της ζήτησης σύντομα εστίασε στον τομέα των

συστημάτων πλοήγησης. Το 2002 κυκλοφόρησε το πρώτο TomTom navigator το οποίο είχε ενσωματωμένο gps. Πλέον η εταιρία με έδρα το Άμστερνταμ απασχολεί περισσότερους από 4.000 υπαλλήλους σε όλο τον κόσμο και διανέμει προϊόντα σε περίπου 41 χώρες.

Η εταιρία προσφέρει τριών ειδών προϊόντα. Συσκευές πλοήγησης, συσκευές πλοήγησης ενσωματωμένες σε αυτοκίνητα και λογισμικό πλοήγησης για εγκατάσταση σε κινητά τηλέφωνα. Οι συσκευές αυτές παρέχουν δυνατότητα πλοήγησης του χρήστη σε συγκεκριμένο προορισμό με φωνητικές οδηγίες, ειδοποίηση ατυχημάτων και εμφάνιση στο χάρτη όλων των σημείων του οδικού δικτύου που παρουσιάζουν κυκλοφοριακή συμφόρηση. Επιπροσθέτως, κρατώντας ιστορικό υπολογίζει και προβλέπει ποια σημεία του οδικού δικτύου θα παρουσιάσουν συμφόρηση ανάλογα με την οδική συμπεριφορά των οδηγών που το χρησιμοποιούν.

Η εταιρία, μαζί με το μεγαλύτερο μέρος των προϊόντων που παράγει, μπορεί να υποστηριχθεί πως αποτελεί ένα πολύ σημαντικό παράδειγμα λειτουργίας του crowdsourcing καθώς το μεγαλύτερο κομμάτι των δεδομένων που επεξεργάζεται προέρχονται από τους χρήστες της.

2.2 Άλλες τεχνικές, σένσορες και κάμερες

Ένας τρόπος για τη μέτρηση της κυκλοφοριακής συμφόρησης είναι ο υπολογισμός της ποσότητας της κίνησης σε έναν συγκεκριμένο κόμβο κάθε χρονική στιγμή. Μια συνηθισμένη μέθοδος είναι η τοποθέτηση σένσορα στο δρόμο αυτό και η καταμέτρηση των φορών που ενεργοποιήθηκε από τις ρόδες των διερχόμενων οχημάτων. Αυτή η προσέγγιση όμως έχει τέσσερα βασικά μειονεκτήματα :

- Είναι ακριβή να εφαρμοστεί αφού οι σένσορες πρέπει να είναι μερικώς ενσωματωμένοι στην άσφαλτο.
- Οι σένσορες στο δρόμο είναι εύκολο να κλαπούν.
- Οι σένσορες πρέπει να τοποθετηθούν σε πολλαπλά σημεία εισόδου και εξόδου στο δρόμο για να έχουμε ακριβείς μετρήσεις.
- Ακόμα και σε ένα συγκεκριμένο κομμάτι δρόμου οι σένσορες πρέπει να τοποθετηθούν σε συχνά διαστήματα για να μπορέσουμε να μελετήσουμε την πυκνότητα της κίνησης σε διαφορετικά σημεία του δρόμου.

Παρά τα μειονεκτήματα που παρουσιάζει η προσέγγιση αυτή, εδώ και περίπου επτά χρόνια, έχει εφαρμοστεί σε διάφορες χώρες για την επίλυση του προβλήματος της κυκλοφοριακής συμφόρησης.

Ισπανία

Στην Ισπανία από το καλοκαίρι του 2007, το DGT (Dirección General de Trafico) παρέχει μια μεγάλη ποσότητα δεδομένων κίνησης πραγματικού χρόνου που είναι μάλιστα ενσωματωμένη στο Google Maps. Ο χρήστης μπορεί εύκολα να συλλέξει αυτά τα δεδομένα τα οποία προέρχονται από 4.000 σένσορες, οι οποίοι βρίσκονται κατά μήκος ολόκληρου του Ισπανικού οδικού δικτύου. Για παράδειγμα, είναι δυνατό να συλλεχθούν η ροή της κίνησης και η μέση ταχύτητα περιφερειακά της Μαδρίτης από σένσορες που βρίσκονται στους κόμβους της εθνικής οδού.

Φινλανδία

Η Φινλανδική οδική υπηρεσία παρέχει πληροφορίες σχετικά με την κίνηση σε πραγματικό χρόνο από 330 αυτόματους σταθμούς καταμέτρησης κατά μήκος του οδικού της δικτύου. Τα δεδομένα κίνησης σχετικά με την ροή και τη μέση ταχύτητα σε κεντρικούς δρόμους αφορούν πάνω από επτά κεντρικές της πόλεις.

Γαλλία

Στο περιοχή του Στρασβούργου της Γαλλίας η περίπτωση είναι κατά πολύ όμοια με την Ισπανία μόνο που εδώ καλύπτεται μόνο η συγκεκριμένη περιοχή. Ένας διαδραστικός χάρτης της περιοχής του Στρασβούργου παρέχει στο χρήστη μετρήσεις πραγματικού χρόνου που αφορούν την κυκλοφοριακή ροή, τη μέση ταχύτητα και το βαθμό κυκλοφοριακής συμφόρησης στις εθνικές οδούς γύρω από την πόλη. Τα δεδομένα αυτά συλλέγονται κάθε λεπτό από 42 σταθμούς καταμέτρησης κυκλοφορίας μέσω σενσόρων που είναι τοποθετημένοι στο δρόμο ανά χιλιόμετρο. Επιπροσθέτως, 50 κάμερες έχουν τοποθετηθεί στο οδικό δίκτυο με σκοπό να ολοκληρώσουν και να επιβεβαιώσουν τις μετρήσεις αυτές. Τέλος, αφού συλλεχθούν τα δεδομένα, ενημερώνεται ο διαδραστικός χάρτης που βρίσκεται στην ιστοσελίδα του υπουργείου μεταφορών.

3. Η εφαρμογή android (front-end)

Στο σημείο αυτό θα γίνει αναφορά στην εφαρμογή από την πλευρά του χρήστη, δηλαδή με λίγα λόγια σε αυτό που βλέπει στο κινητό του και τις λειτουργίες που τρέχουν σε αυτό. Πιο συγκεκριμένα, η εφαρμογή θα μπορούσε να διαχωριστεί σε 3 βασικά τμήματα:

- Στον τρόπο συλλογής των απαραίτητων για τον προσδιορισμό της κυκλοφοριακής συμφόρησης δεδομένων και το σύστημα πλοήγησης.
- Στο χρονικό διάστημα της συλλογής και αποστολής των δεδομένων.
- Στην αποστολή τους και την επικοινωνία της εφαρμογής με το server.

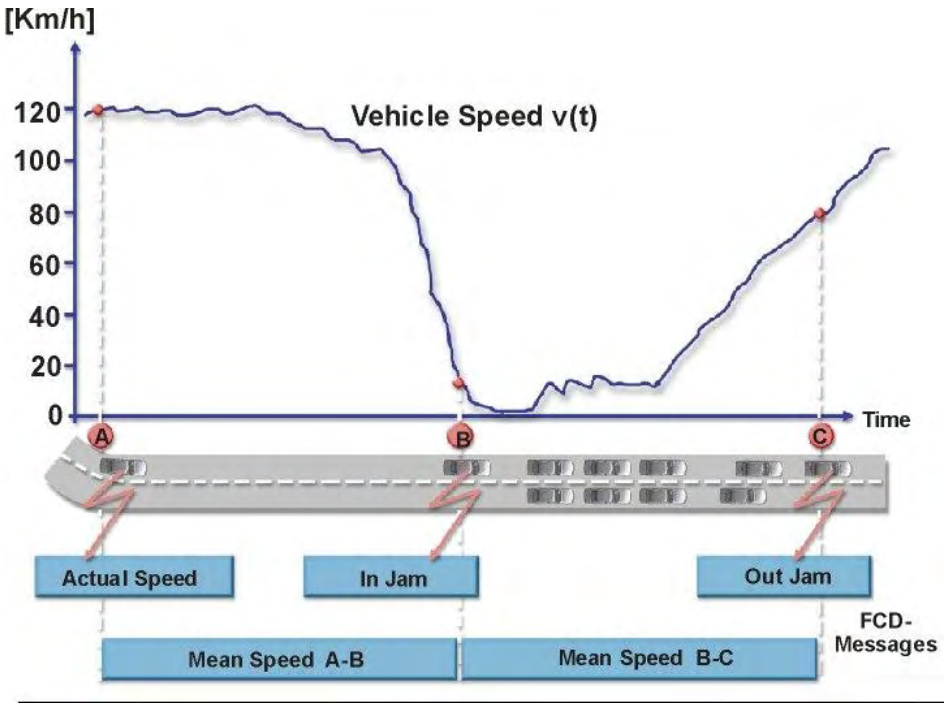
Όλα τα τμήματα αυτά είναι σημαντικά, καθώς μια εφαρμογή που βασίζεται στο crowdsourcing πρέπει να παρέχει στο χρήστη και ένα κίνητρο να τη χρησιμοποιεί καθημερινά (έτσι ώστε να μπορούμε να συλλέγουμε συνεχώς δεδομένα) όμως παράλληλα πρέπει να παρέχει και αξιόπιστα και ακριβή δεδομένα σχετικά με την κίνηση.

3.1 Συλλογή των δεδομένων και εντοπισμός της θέσης του χρήστη

Το βασικότερο κομμάτι της ανάπτυξης ενός αποδοτικού αλγορίθμου για τον εντοπισμό της κυκλοφοριακής συμφόρησης με τη μέθοδο του Floating Car Data (FCD) είναι τα δεδομένα που μπορούμε να συλλέξουμε από τις συσκευές κινητής τηλεφωνίας των χρηστών.

Τι είναι όμως Floating Car Data (FCD)?

FCD (*Floating Car Data*) [8], ή αλλιώς γνωστά ως *Floating Cellular Data* είναι μια μέθοδος για να προσδιοριστεί η ταχύτητα της κίνησης σε ένα οδικό δίκτυο. Βασίζεται στη συλλογή των δεδομένων τοποθεσίας, ταχύτητα, κατεύθυνσης ταξιδιού και χρόνου από κινητά τηλέφωνα σε οχήματα τα οποία οδηγούνται. Αυτά τα δεδομένα είναι η βασική πηγή για πληροφορίες κίνησης και για τα πιο ευφυή δίκτυα μεταφοράς. Αυτό σημαίνει ότι κάθε όχημα με ένα ενεργό κινητό τηλέφωνο λειτουργεί ως αισθητήρας για το οδικό δίκτυο. Βασιζόμενοι σε αυτά τα δεδομένα, μπορούμε να υπολογίσουμε την κυκλοφοριακή συμφόρηση, τους χρόνους ταξιδιού, και να δημιουργήσουμε δελτία κίνησης. Σε αντίθεση με τις κάμερες κυκλοφορίας, την αναγνώριση πινακίδων και τη μαγνητική επαγωγή επανάληψης (induction loop) εγκατεστημένη σε εθνικές οδούς, στην περίπτωση του FCD δεν χρειάζεται επιπρόσθετο hardware υλικό στο οδικό δίκτυο.



(Εικόνα 6) πηγή: EXTENDED FLOATING-CAR DATA FOR THE ACQUISITION OF TRAFFIC INFORMATION

Στην κατηγορία του Floating Car Data έχουμε τρεις σημαντικές μεθόδους για τον εντοπισμό της θέσης του χρήστη [30].

- **Μέθοδος τριγωνομέτρησης (Triangulation method).** Στις ανεπτυγμένες χώρες ένα πολύ μεγάλο ποσοστό αυτοκινήτων έχουν ένα ή περισσότερα κινητά τηλέφωνα. Τα τηλέφωνα αυτά στέλνουν περιοδικά την τοποθεσία τους στο δίκτυο κινητής τηλεφωνίας ακόμα και όταν οι χρήστες δεν τα χρησιμοποιούν. Από τα μέσα του 2000 ξεκίνησαν να γίνονται προσπάθειες για να χρησιμοποιηθούν τα κινητά τηλέφωνα με σκοπό την εύρεση κυκλοφοριακής συμφόρησης στις οδικές αρτηρίες. Καθώς το αυτοκίνητο κινείται, παράλληλα κινείται και το σήμα οποιασδήποτε κινητής συσκευής που είναι μέσα σε αυτό. Έτσι, με τη μέθοδο της τριγωνομέτρησης εντοπίζεται με ακρίβεια η θέση του χρήστη. Όσο περισσότερη κίνηση παρατηρείται σε κάποιο σημείο τόσο περισσότερα οχήματα υπάρχουν και άρα περισσότερα κινητά τηλέφωνα. Ένα πλεονέκτημα της μεθόδου αυτής είναι ότι δεν χρειάζονται επιπρόσθετες εγκαταστάσεις κατά μήκος του οδικού δικτύου, όμως στην πραγματικότητα η εφαρμογή της τριγωνομέτρησης μπορεί να είναι πολύ σύνθετη, ειδικά σε περιοχές όπου ένας πύργος τηλεπικοινωνιών εξυπηρετεί δύο ή περισσότερες παράλληλες διαδρομές. Στις αρχές του 2010 ξεκίνησε η εγκατάλειψη της μεθόδου αυτής και πήραν τη σκυτάλη άλλες, πιο σύγχρονες.

- **Αναγνώριση οχήματος (Vehicle re-identification).** Η μέθοδος της αναγνώρισης οχημάτων απαιτεί ένα σύνολο από ανιχνευτές κατά μήκος του οδικού δικτύου. Κατά την τεχνική αυτή, εντοπίζεται ένας μοναδικός σειριακός αριθμός για μια συσκευή μέσα στο όχημα σε κάποια τοποθεσία και έπειτα εντοπίζεται ξανά σε κάποιο άλλο σημείο του δρόμου. Με αυτόν τον τρόπο υπολογίζεται ο χρόνος ταξιδιού και η ταχύτητα του οχήματος συγκρίνοντας τους χρόνους κατά τους οποίους εντοπίστηκε η συγκεκριμένη συσκευή από ζευγάρια σενσόρων.
- **Μέθοδοι GPS εντοπισμού (GPS based methods).** Ένας συνεχώς αυξανόμενος αριθμός οχημάτων είναι εφοδιασμένος με GPS (satellite navigation) συστήματα που έχουν αμφίδρομη επικοινωνία με κάποιον πάροχο δεδομένων κίνησης. Η καταγραφή της θέσης σε αυτά τα οχήματα χρησιμεύει για τον υπολογισμό της ταχύτητας. Σύγχρονες μέθοδοι δεν χρησιμοποιούν εξειδικευμένο hardware αλλά αντ' αυτού χρησιμοποιούν λύσεις βασισμένες σε κινητά τηλέφωνα (Smartphones).

Για την παρούσα εφαρμογή χρησιμοποιήσαμε την μέθοδο εντοπισμού της θέσης του χρήστη με GPS και WiFi. Πρακτικά το κινητό του χρήστη στέλνει αίτηση για update της τοποθεσίας του από GPS ή από ίντερνετ (Mobile Data) εντοπιστεί η θέση του με την καλύτερη δυνατή ακρίβεια. Οι περισσότερες, αν όχι όλες οι κινητές συσκευές τα τελευταία χρόνια έχουν δυνατότητα σύνδεσης στο δίκτυο μέσω πύργων κινητής τηλεφωνίας και μάλιστα πλέον με αρκετά μεγάλες ταχύτητες (4G), γεγονός που δίνει τη δυνατότητα στους χρήστες να συνδέονται στο ίντερνετ σε οποιοδήποτε σημείο και αν βρίσκονται. Εκμεταλλευόμενοι τη δυνατότητα για σύνδεση στο ίντερνετ παντού και σε συνδυασμό με το gps βρίσκουμε με μεγάλη ακρίβεια τη θέση του χρήστη.

Ο λόγος που αναζητείται με μεγάλη ακρίβεια η θέση του χρήστη είναι γιατί πρέπει να είμαστε σε θέση να μπορούμε να σχεδιάσουμε την ακριβή διαδρομή που ακολούθησε, σε περίπτωση που βρεθεί ότι σε αυτή υπάρχει πρόβλημα κυκλοφοριακής συμφόρησης. Έχουμε λοιπόν τη συνάρτηση:

```
@Override
public void onLocationChanged(Location location) {
    Do stuff with the new location...
}
```

η οποία καλείται αυτόματα όταν υπάρξει κάποιο update σχετικά με την τοποθεσία του κινητού τηλεφώνου. Μέσα στη συνάρτηση αυτή συλλέγουμε όλα τα χρήσιμα δεδομένα που αργότερα θα στείλουμε στο server, δηλαδή:

- *Location.getSpeed():* για την ταχύτητα εκείνη τη χρονική στιγμή.
- *Location.getBearing():* για την κατεύθυνση του οχήματος.
- *Location.getAccuracy():* για την ακρίβεια της θέσης που βρήκαμε.
- *Location.getTime():* για την ώρα που λάβαμε την τοποθεσία
- *Location.getLatitude():* για το γεωγραφικό πλάτος
- *Location.getLongitude():* για το γεωγραφικό μήκος

Από τα δεδομένα αυτά υπολογίζεται η μέση ταχύτητα μεταξύ κάθε δυο διαδοχικών σημείων.

$$\text{average speed} = \frac{\text{distance between two points}}{\text{time difference}}$$

Κατά κύριο λόγο αυτή είναι η τιμή που χρησιμοποιείται αργότερα στο Server για τον υπολογισμό της κυκλοφοριακής συμφόρησης.

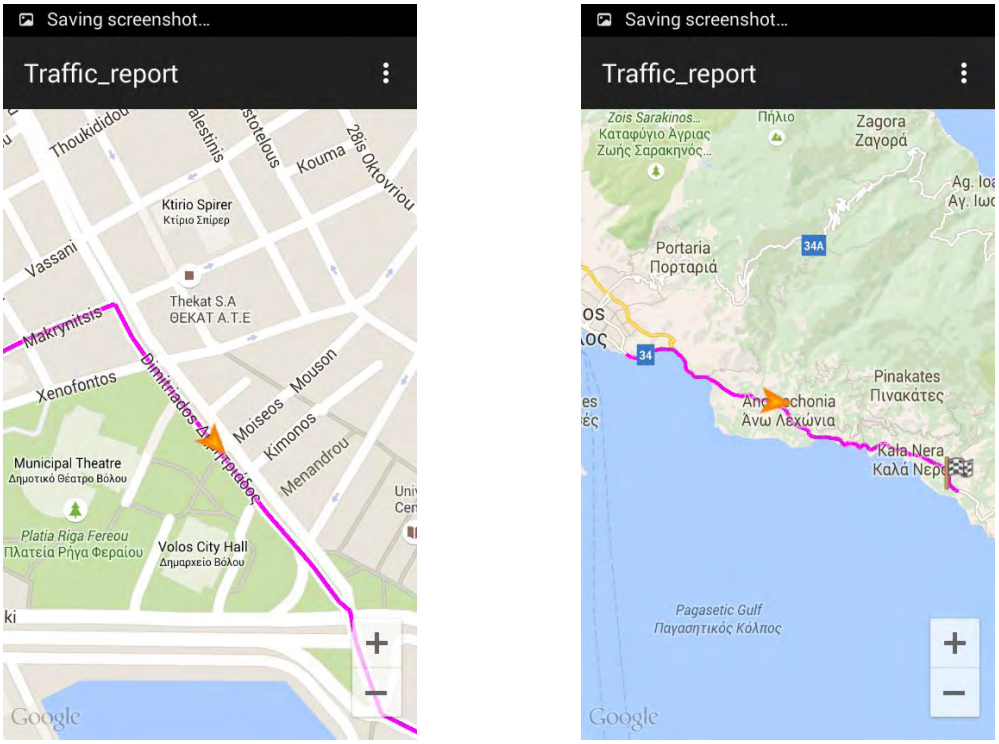
Το πρώτο βασικό πρόβλημα που συναντούμε σε αυτό το σημείο είναι ο ακριβής εντοπισμός της θέσης του χρήστη χωρίς αποκλίσεις. Θέλουμε η θέση να μπορεί να προσδιοριστεί με τέτοια ακρίβεια έτσι ώστε να μας είναι εύκολο να προβάλλουμε τη θέση του χρήστη ακριβώς στο δρόμο στον οποίο βρίσκεται. Η μέθοδος λοιπόν που χρησιμοποιήθηκε δίνει την τοποθεσία του κινητού τηλεφώνου με ακρίβεια περίπου 30 μέτρα σε εσωτερικούς χώρους όπου το σήμα του GPS δεν είναι αρκετά ισχυρό, ενώ στο δρόμο και γενικά σε εξωτερικούς χώρους κατά μέσο όρο η ακρίβεια με την οποία εντοπίζεται η τοποθεσία του χρήστη είναι 5-10 μέτρα ενώ σε πυκνοκατοικημένες περιοχές 5-15 μέτρα. Όταν, τέλος, ο χρήστης μιλάει στο τηλέφωνο ενώ η εφαρμογή είναι σε λειτουργία επηρεάζεται η απόδοση των mobile data και η ακρίβεια της θέσης του κυμαίνεται από 7-15 μέτρα.

Το καλύτερο όριο που βρέθηκε για να αποδεχόμαστε ένα location update είναι τα 7 μέτρα. Μετά από αρκετές προσπάθειες παρατηρήθηκε ότι για αποκλίσεις μεγαλύτερες των 7 μέτρων δεν μπορεί εύκολα να βρεθεί ακριβώς ο δρόμος στον οποίο βρίσκεται ο χρήστης καθώς ο αλγόριθμος λανθασμένα προβάλει τη θέση του σε άλλο, παράλληλο ή κάθετο δρόμο.

Τον προσδιορισμό της τοποθεσίας του χρήστη θα τον χωρίσουμε σε δύο κατηγορίες. Την περίπτωση που ο χρήστης έχει προσθέσει τον προορισμό του οπότε γνωρίζουμε τη διαδρομή που θεωρητικά θα ακολουθήσει, και την περίπτωση όπου δεν γνωρίζουμε τον τελικό του προορισμό.

3.1.1. Όταν γνωρίζουμε τον προορισμό του χρήστη

Η εφαρμογή δίνει τη δυνατότητα στο χρήστη να προσθέσει τον προορισμό της επιλογής του και στη συνέχεια σχεδιάζει στο χάρτη τη βέλτιστη διαδρομή που πρέπει να ακολουθήσει (Εικόνα 6). Η διαδρομή αυτή αποτελεί μια polyline.



(Εικόνα 6) Στιγμιότυπα από την εφαρμογή στο κινητό τηλέφωνο.

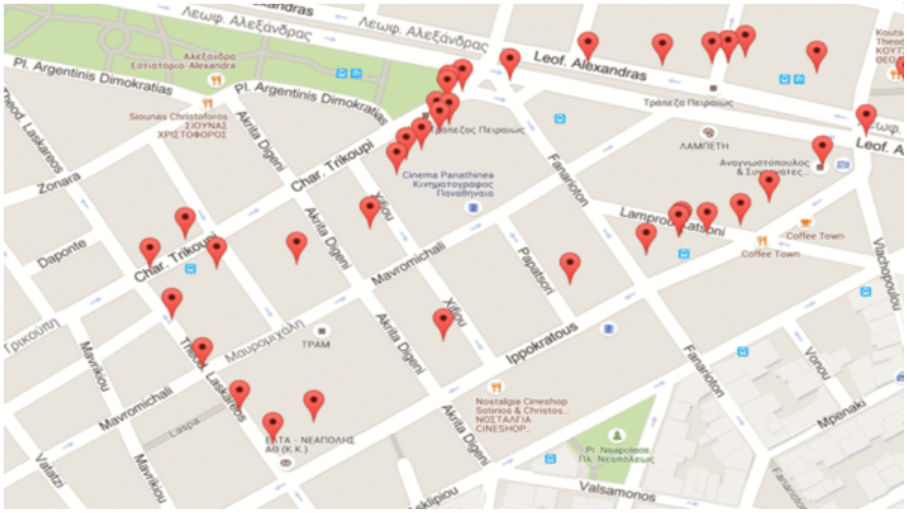
Πώς όμως σχεδιάζουμε αυτή την polyline; Σε κάθε διασταύρωση μεταξύ του χρήστη και του προορισμού του ορίζουμε και από ένα σημείο. Στη συνέχεια, ενώνουμε κάθε σημείο με το επόμενο του ανάλογα με τη μορφολογία του δρόμου που τα ενώνει (ευθεία, στροφές κτλ.) και σχεδιάζουμε την αντίστοιχη συνάρτηση. Έτσι, όταν όλα τα σημεία έχουν ενωθεί έχουμε την τελική polyline.

Έχοντας τη διαδρομή και ουσιαστικά γνωρίζοντας την πορεία που θα ακολουθήσει ο χρήστης, για κάθε νέα τοποθεσία που λαμβάνουμε μέσω του GPS-WiFi έστω x (latitude, longitude) ελέγχουμε αν απέχει λιγότερο από 25 μέτρα από την ήδη

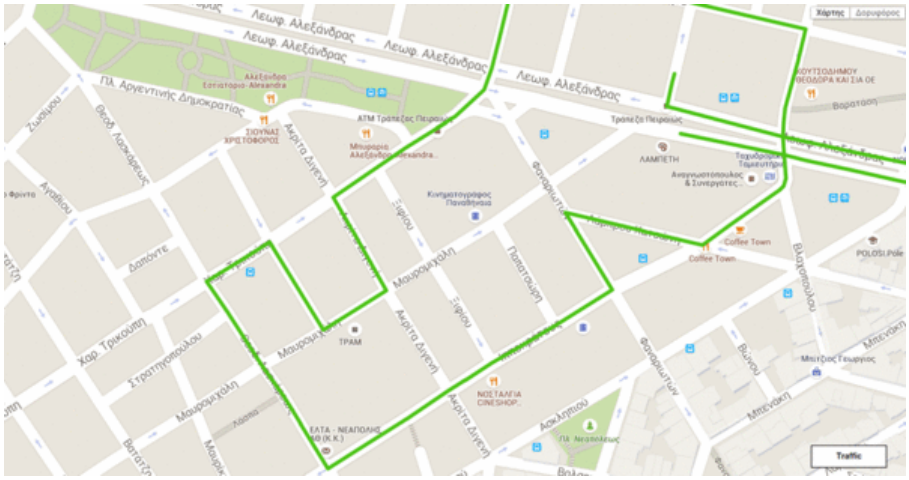
σχεδιασμένη διαδρομή. Αν ναι, τότε υπολογίζουμε το κοντινότερο σημείο του x πάνω στην polyline και ορίζουμε αυτό ως τη διορθωμένη θέση του χρήστη. Ο τρόπος αυτός και όταν ο χρήστης ακολουθεί την διαδρομή που έχει ορίσει μας δίνει ένα αρκετά μεγάλο ποσοστό ακρίβειας στον προσδιορισμό της τοποθεσίας του. Αν η τοποθεσία του χρήστη βρεθεί να αποκλίνει πάνω από 25 μέτρα από την polyline τότε ξανά-υπολογίζουμε τη διαδρομή που πρέπει να ακολουθήσει για να φτάσει στον προορισμό του και χρησιμοποιούμε το directions api (θα ακολουθήσει σχετική αναφορά) για τον εντοπισμό της θέσης του. Ακόμα και με αυτή τη μέθοδο διατηρούμε το accuracy σε τιμές μικρότερες του 7 για τα αποτελέσματα της τοποθεσίας που δεχόμαστε στη συνάρτηση *onLocationChanged*.

3.1.2. Όταν ΔΕΝ γνωρίζουμε τον προορισμό του χρήστη

Ο χρήστης δεν είναι υποχρεωμένος να προσθέσει τον προορισμό του στην εφαρμογή. Καθώς οι συντεταγμένες που θα πάρουμε είναι πάρα πολύ σπάνιο να βρίσκονται πάνω σε δρόμο, είναι πολύ σημαντικό το accuracy της θέσεις να είναι μικρότερο του 7. Ο παρακάτω έλεγχος πραγματοποιήθηκε για να ελεγχθούν τα αποτελέσματα της ακρίβειας του location σε τιμή μεγαλύτερη του 7. Όπως παρατηρείται από τις εικόνες 7 και 8, οι συντεταγμένες έχουν τόση απόκλιση ώστε είναι πάρα πολύ δύσκολο να προβλεφθεί η διαδρομή που ακολουθήθηκε από το χρήστη.



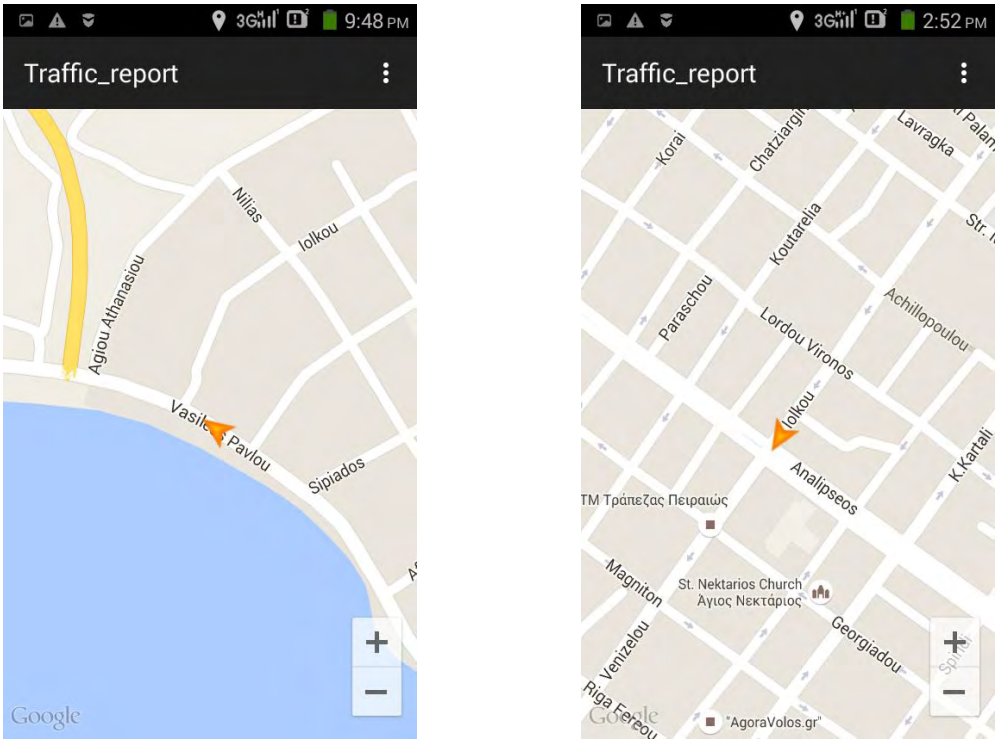
(Εικόνα 7) Μια διαδρομή χωρίς κανένα περιορισμό στην ακρίβεια του σήματος.



(Εικόνα 8) Αποτέλεσμα της προσπάθειας του αλγορίθμου να αντιστοιχίσει τα location σε κάποιο δρόμο.

Ακόμα όμως και για accuracy, στις συντεταγμένες που θα δεχόμαστε, μικρότερο του 7, η θέση περίπου στο 60% των περιπτώσεων δεν είναι πάνω σε δρόμο. Στην περίπτωση αυτή βασιζόμαστε κυρίως σε μια ασύγχρονη διαδικασία (AsyncTask) που δημιουργήθηκε για να μπορέσουμε να απεικονίσουμε την θέση του χρήστη πάνω στο δρόμο. Μια ασύγχρονη διαδικασία αποτελείται από τρία στάδια. Αρχικά, εκτελείται η onPreExecute(), στη συνέχεια η doInBackground() και τέλος η onPostExecute().

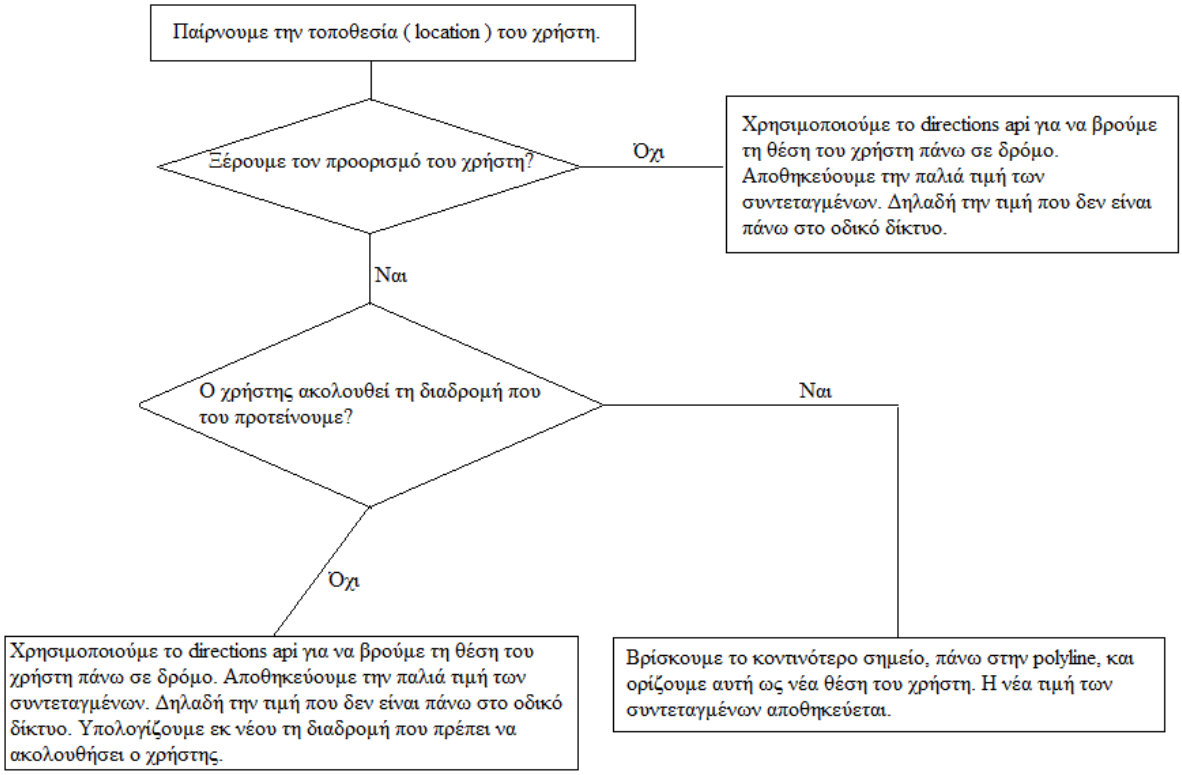
Στην doInBackground() στέλνεται μέσω url ένα αίτημα στο directions api της google. Το directions api δέχεται τις συντεταγμένες ενός σημείου (αρχής) και τις συντεταγμένες ενός σημείου (προορισμού) και σχεδιάζει τη διαδρομή. Εκμεταλλευόμενοι αυτό, στέλνουμε στο url σαν αρχή και προορισμό τις ίδιες συντεταγμένες, δηλαδή την ακριβή θέση του χρήστη, όπως την δεχτήκαμε από τη συνάρτηση onLocationChanged(). Επειδή το api στην απάντηση που στέλνει σχεδιάζει μόνο πάνω σε δρόμο, καταφέρνουμε να πάρουμε και τη θέση του χρήστη στο δρόμο. Στην onPostExecute() δεχόμαστε λοιπόν αυτή την απάντηση από το server της google που περιλαμβάνει τις κοντινότερες, σε σχέση με τη θέση του χρήστη, συντεταγμένες πάνω σε δρόμο (Εικόνα 9).



(Εικόνα 9) Στιγμιότυπα από την εφαρμογή στο κινητό τηλέφωνο.

Γενικά, όποτε χρειάζεται να χρησιμοποιήσουμε το directions api για να αντιστοιχίσουμε την θέση του χρήστη στο οδικό δίκτυο, στην ουσία αποθηκεύουμε τις παλιές συντεταγμένες, της συντεταγμένες δηλαδή χωρίς καμία επεξεργασία που μάλλον δεν είναι πάνω σε κάποιο δρόμο. Αυτό γίνεται διότι όπως αναφέραμε η συνάρτηση που χρησιμοποιεί το directions api είναι ασύγχρονη. Επειδή το αίτημα μας για να αντιστοιχιστούν οι συντεταγμένες στο οδικό δίκτυο είναι μέσο online request, η απάντηση κάποιες φορές τυχαίνει να καθυστερήσει περίπου 15-20 δευτερόλεπτα. Για να μπορέσουμε να μεταφράσουμε αργότερα τη διαδρομή που ακολουθεί ο χρήστης είναι σημαντικό οι συντεταγμένες να αποθηκεύονται με σωστή σειρά. Το γεγονός αυτό λοιπόν παρατηρήθηκε πως επηρεάζει σοβαρά την αποθήκευση των τιμών μας καθώς μέχρι να πάρουμε την απάντηση από το directions api τυχαίνει να έχουν αποθηκευτεί πρώτα, θέσης του χρήστη που χρονικά θα έπρεπε να αποθηκευτούν αργότερα. Για να αντιμετωπιστεί το πρόβλημα αυτό θα έπρεπε να παγώνουμε την εκτέλεση μέχρι να πάρουμε απάντηση από το api κάτι που για πολλούς λόγους δεν θα ήταν καθόλου αποδοτικό.

Έτσι το σκεπτικό και η μέθοδος που ακολουθήθηκε παρουσιάζονται συνοπτικά στο ακόλουθο διάγραμμα ροής.



Καταλήγοντας, ο χρήστης πάντα στην εφαρμογή του βλέπει την θέση του πάνω στο οδικό δίκτυο. Για τον λόγο που αναφέραμε προηγουμένως όμως, η θέση που εμείς αποθηκεύουμε και στέλνεται στο server δεν είναι πάντα αυτή. Παρατηρήθηκε ωστόσο ότι αυτό δεν έχει καμία επίπτωση καθώς το web service μας όταν προσπαθήσει να σχεδιάζει την διαδρομή που παρουσιάζει κυκλοφοριακή συμφόρηση, ακόμα και με συντεταγμένες εκτός οδικού δικτύου καταφέρνει αποτελεσματικά να σχεδιάσει τη σωστή διαδρομή με τρόπο που θα αναφέρουμε στο επόμενο κεφάλαιο.

3.2 Χρονικά διαστήματα συλλογής των δεδομένων.

Έχοντας αναλύσει τον τρόπο συλλογής των δεδομένων (συντεταγμένες, ταχύτητα κλπ.) από τα location update, τίθεται το ερώτημα κάθε πότε είναι καλύτερο να ζητάμε για ανανέωση της τοποθεσίας του χρήστη. Στο σημείο αυτό χρήζει μελέτης η εύρεση της καλύτερης αναλογίας μεταξύ ενεργειακού κόστους και ποιότητας δεδομένων. Αν οι αιτήσεις μας για εύρεση της νέας τοποθεσίας του χρήστη γίνονται πολύ αραιά μεταξύ τους (π.χ. Sampling Period > 30sec), όχι μόνο δεν θα μπορέσουμε με ακρίβεια να βρούμε στη συνέχεια

την πορεία του χρήστη αλλά και τα δεδομένα ταχύτητας θα είναι τόσο αραιά που δεν θα μπορούμε εύκολα να συμπεράνουμε αν υπάρχει ή όχι κυκλοφοριακή συμφόρηση.

Από άποψη κατανάλωσης ενέργειας τουλάχιστον, όπως βλέπουμε και από τον ακόλουθο πίνακα, η δειγματοληψία ανά 1 δευτερόλεπτο ή ανά 30 δευτερόλεπτα έχει πολύ μικρή διαφορά στην εξοικονόμηση ενέργειας.

Location Mechanism	Sampling Period	Lifetime
None	-	7 h
GPS	continuous (1/sec)	2 h 24 m
GPS	30 sec	2 h 27 m
GPS	2 min	2 h 44 m
WiFi	continuous (1/sec)	6 h 30 m

Πηγή: *VTrack: accurate, energy-aware road traffic delay estimation using mobile phones*

Βάση αυτού έγινε προσπάθεια να προσομοιώσουμε την περίοδο δειγματοληψίας σε μια τιμή μικρότερη του 30 (για να έχουμε συχνά update σχετικά με την θέση και την ταχύτητα) αλλά όχι και υπερβολικά μικρή σε σημείο που να έχουμε πάρα πολλά δεδομένα που δε θα χρειαζόμαστε. Ουσιαστικά αν εφαρμόζαμε δειγματοληψία κάθε ένα δευτερόλεπτο θα είχαμε τόσο συχνές ανανεώσεις της ταχύτητας και της θέσης του χρήστη που θα αποτελούσε πλεονασμό.

Πραγματοποιήθηκε λοιπόν έλεγχος για ανανέωση της τοποθεσίας του χρήστη και της ταχύτητας του κάθε 10 και κάθε 5 δευτερόλεπτα. Στην περίπτωση των 10 δευτερολέπτων παρατηρήθηκε ότι ενώ μπορούμε να βρούμε κατά προσέγγιση τη διαδρομή που ακολουθεί, είναι δύσκολο να εντοπίσουμε τις περιπτώσεις κυκλοφοριακής συμφόρησης με τον αλγόριθμο μας γιατί τα δείγματα είναι πολύ αραιά.

Για παράδειγμα, στον αλγόριθμο, στον οποίο θα αναφερθούμε στο επόμενο κεφάλαιο, χρησιμοποιείται και η στιγμιαία ταχύτητα του οχήματος για να ελέγξουμε αν είναι σε στάση ή εν κινήσει. Παρατηρήθηκε ότι σε περιπτώσεις μεγάλης κυκλοφοριακής συμφόρησης, το όχημα προλαβαίνει να ξεκινήσει από στάση και να ξανά σταματήσει μέσα σε χρόνο 10 δευτερολέπτων δίνοντας πολλές στιγμιαίες ταχύτητες ίσες με μηδέν. Το γεγονός αυτό καθιστά εξαιρετικά δύσκολη την κατανόηση του αν το όχημα απλά έχει σταθμεύσει κάπου ή αν είναι εν κινήσει. Απεναντίας, στην περίπτωση δειγματοληψίας ανά 5 δευτερόλεπτα παρατηρήθηκε μεγαλύτερη διαφοροποίηση στις στιγμιαίες ταχύτητες ακόμα και σε

περίπτωση μεγάλης κυκλοφοριακής συμφόρησης. Αυτό βοήθησε πολύ στην διαφοροποίηση μεταξύ της κίνησης και της περίπτωσης που απλά το όχημα είναι σταθμευμένο.

Συμπερασματικά, το GPS αναμφισβήτητα καταναλώνει πολύ ενέργεια και περιορίζει τη διάρκεια ζωής της μπαταρίας περίπου στο 1/3. Λαμβάνοντας update για την τοποθεσία του κινητού τηλεφώνου κάθε 5 δευτερόλεπτα δίνει για τον αλγόριθμο και την προσέγγιση μας, την καλύτερη αναλογία μεταξύ κατανάλωσης ενέργειας και ποιότητας δεδομένων.

3.3 Επικοινωνία της εφαρμογής με το server

Στα δύο παραπάνω στάδια αναφερθήκαμε στον τρόπο συλλογής των δεδομένων που χρησιμοποιούνται από τον αλγόριθμο μας για τον υπολογισμό της κυκλοφοριακής συμφόρησης. Πώς όμως στέλνουμε τα δεδομένα, κάθε πότε, και γιατί να μην γίνει ο υπολογισμός της κυκλοφοριακής συμφόρησης στην εφαρμογή;

Όπως ειπώθηκε παραπάνω, λαμβάνουμε μια μεταβλητή τύπου Location όταν αυτομάτως καλεστεί η onLocationChanged(). Από τη μεταβλητή αυτή αφού εξάγουμε τα δεδομένα (ταχύτητα, τοποθεσία κλπ.), υπολογίσουμε όσα δεν μας δίνονται (μέση ταχύτητα, απόσταση κλπ.) και προσδιορίσουμε τη θέση του χρήστη πάνω στο οδικό δίκτυο, τα αποθηκεύουμε σε μια λίστα. Κάθε θέση της λίστας περιέχει μια κλάση που δημιουργήσαμε και η οποία περιέχει όλες τις τιμές που χρειαζόμαστε και αργότερα θα στείλουμε στο server.

3.3.1. Πως γίνεται η επικοινωνία με το server

Για να επιτευχθεί η επικοινωνία με το server έχουμε ακόμα δυο ασύγχρονες διαδικασίες. Αυτές είναι η LoadAll() και AddNewData().

Η πρώτη χρησιμοποιείται για να διαβάσουμε από τη βάση δεδομένων τη στήλη id και εκτελείται μόνο μία φορά, όταν πρόκειται να γίνει η πρώτη εγγραφή της εφαρμογής. Κάθε χρήστης και πιο συγκεκριμένα κάθε διαδρομή έχει ένα συγκεκριμένο αναγνωριστικό (id). Όταν ο χρήστης ξεκινήσει την εφαρμογή και λίγο πριν κάνει εγγραφή των δεδομένων στη database διαβάζει για να δει ποιο είναι το αναγνωριστικό που θα του ανατεθεί. Τα αναγνωριστικά ξεκινούν από 1 και αυξάνονται κατά ένα για κάθε νέα διαδρομή. Όσο ο χρήστης έχει ανοιχτή την εφαρμογή, γράφει στη βάση δεδομένων με το αναγνωριστικό που βρήκε όταν την ξεκίνησε, άρα για παράδειγμα θα έχει μέχρι να κλείσει την εφαρμογή το id 5.

Η δεύτερη ασύγχρονη διαδικασία έχει ως σκοπό την εγγραφή των δεδομένων στη βάση. Με το αναγνωριστικό που έχει βρεθεί από την πρώτη AsyncTask καταχωρεί τα δεδομένα. Από τη βάση μας αργότερα διαβάζει ένα ένα τα id ο αλγόριθμος και υπολογίζει το βαθμό της κυκλοφοριακής συμφόρησης σε διάφορα σημεία του οδικού δικτύου.

Οι δύο αυτές ασύγχρονες διαδικασίες την πρώτη και μόνη φορά που θα εκτελεστούν μαζί, καλούνται και εκτελούνται παράλληλα, καθώς τις επόμενες φορές εκτελείται μόνο η δεύτερη. Είναι σημαντικό το γεγονός ότι θα εκτελεστούν παράλληλα καθώς όταν ανατεθεί ένα αναγνωριστικό σε ένα χρήστη, θέλουμε να γράψει κατευθείαν στη βάση για να το δεσμεύσει, έτσι ώστε αν κάποιο άλλος νέος χρήστης προσπαθήσει να βρει το δικό του αναγνωριστικό να μην πάρει το ίδιο με τον πρώτο.

Για να γίνει η σύνδεση της εφαρμογής με τον server, χρησιμοποιούμε τέσσερα php αρχεία που είναι αποθηκευμένα σε ένα φάκελο που δημιουργήθηκε (android_connect *)* σε συγκεκριμένο σημείο στο server. Τα php αρχεία είναι:

- **db_config.php**
- **db_connect.php**
- **get_all.php**
- **Insert.php**

Το πρώτο περιέχει το όνομα της βάσης δεδομένων μαζί με το username και password για να μπορεί ο χρήστης να έχει δικαιώματα να διαβάζει και να γράφει στη βάση. Το db_connect.php ανοίγει τη σύνδεση μεταξύ της εφαρμογής και της database και την κλείνει όταν ολοκληρωθεί η διεργασία που εκτελεί. Τέλος, από τα get_all.php και insert.php το πρώτο είναι αυτό που ελέγχει για να βρει το αναγνωριστικό που θα δώσει στο χρήστη και το επιστρέφει στην εφαρμογή και το δεύτερο είναι αυτό που κάνει τις εγγραφές των δεδομένων στη βάση.

Ουσιαστικά, η εφαρμογή και πιο συγκεκριμένα οι δυο ασύγχρονες διαδικασίες που αναφέραμε προηγουμένως καλούν μέσω url τα αρχεία get_all και insert.

http://192.168.1.2/android_connect/get_all.php

http://192.168.1.2/android_connect/insert.php

Τα δυο αρχεία αυτά καλούν τα db_config και db_connect προκειμένου να πραγματοποιήσουν τη σύνδεση.

3.3.2. Κάθε πότε στέλνουμε τα δεδομένα

Για λόγους που θα εξηγηθούν λεπτομερώς στο επόμενο κεφάλαιο, υπολογίζουμε την κυκλοφοριακή συμφόρηση σε διαστήματα των 5 λεπτών. Με λίγα λόγια, ελέγχουμε τα δεδομένα πέντε λεπτών και ανάλογα καταλήγουμε αν σε αυτό το διάστημα υπάρχει συμφόρηση ή όχι. Για το λόγο αυτό, κάθε πέντε λεπτά και εφόσον έχουν συλλεχθεί δεδομένα, η εφαρμογή προσπαθεί να επικοινωνήσει με το server και να γράψει στη database τα δεδομένα που έχει συγκεντρώσει. Αν η επικοινωνία με το server αποτύχει μια συγκεκριμένη χρονική στιγμή, λόγο έλλειψης σήματος, γίνεται ξανά προσπάθεια ανά πέντε περίπου δευτερόλεπτα. Τέλος, βρέθηκε πως το χρονικό αυτό διάστημα είναι βέλτιστο και για τον αλγόριθμο και για το κινητό τηλέφωνο του χρήστη, καθώς πετυχαίνουμε και γρήγορο υπολογισμό της κυκλοφοριακής συμφόρησης με άμεση ενημέρωση των χρηστών και δεν σπαταλούμε πόρους σε συνεχείς συνδέσεις.

3.3.3. Υπολογισμός κυκλοφοριακής συμφόρησης στο server ή στο κινητό

Η αρχική μας προσέγγιση ήταν να ενσωματώσουμε τον αλγόριθμο στην android εφαρμογή έτσι ώστε αμέσως όταν συλλέγονται τα δεδομένα να γίνεται ο υπολογισμός τοπικά για κυκλοφοριακή συμφόρηση και τα αποτελέσματα να στέλνονται στο server, ο οποίος θα ενημερώνει τους υπόλοιπους χρήστες. Το πρόβλημα που εντοπίστηκε σε αυτή την προσέγγιση ήταν η αύξηση στην κατανάλωση ενέργειας της εφαρμογής όπως είναι λογικό από τους συνεχείς υπολογισμούς. Λαμβάνοντας υπόψη ότι οι μπαταρίες των κινητών τηλεφώνων, μέχρι σήμερα τουλάχιστον, περιορίζουν συνεχώς την διάρκεια αυτονομίας τους, ακολουθήθηκε η προσέγγιση όπου θα εξοικονομούσε ενέργεια.

Επιπροσθέτως, ο βασικός λόγος που επιλέχθηκε η επεξεργασία των δεδομένων να πραγματοποιείται στο server, είναι ότι επιδιώκεται ο έλεγχος και η επαλήθευση των αποτελεσμάτων του αλγορίθμου, βλέποντας αν δυο χρήστες που πέρασαν από το ίδιο σημείο σημείωσαν παρόμοιες ή αρκετά διαφορετικές ταχύτητες. Είναι, λοιπόν, αρκετά πιο χρήσιμο να συγκεντρώνουμε ολόκληρες διαδρομές στο server αντί για μερικά σημεία συμφόρησης

όπως μας τα έχουν μεταδώσει οι χρήστες τα οποία μάλιστα δεν θα έχουμε κανέναν τρόπο να ελέγξουμε.

4. Η πλευρά του server (back-end)

Ο server όπως είδη έχουμε αναφέρει στήθηκε σε localhost με τη βοήθεια του xampp. Είναι σημαντικό στο σημείο αυτό να διαχωρίσουμε τις ενέργειες που εκτελούνται στο server. Αρχικά, θα αναφερθούμε στον αλγόριθμο που χρησιμοποιήθηκε και στη συνέχεια στον τρόπο που μεταφράστηκαν τα αποτελέσματα του αλγορίθμου με γραφική απεικόνιση στην ιστοσελίδα μας.

Πριν ξεκινήσουμε σημαντικό είναι να αναφέρουμε τι θεωρούμε ως κυκλοφοριακή συμφόρηση. Οι απόψεις πάνω στο συγκεκριμένο θέμα ποικίλουν ανάλογα με τους ανθρώπους και τις χώρες. Μιας και στις περισσότερες χώρες η κυκλοφοριακή συμφόρηση εντοπίζεται κυρίως στις εθνικές οδούς, ως συμφόρηση ορίζεται ακόμα και όταν τα οχήματα κινούνται με χαμηλή ταχύτητα (50-60 km/h) σε κεντρικές οδικές αρτηρίες.

Στη συγκεκριμένη μελέτη επικεντρωθήκαμε κυρίως στην κίνηση εντός των πόλεων οπότε ορίσαμε διαφορετικά την προσέγγιση μας. Ο μέσος χρόνος για να αλλάξει ένας φωτεινός σηματοδότης από κόκκινο σε πράσινο είναι περίπου δυο λεπτά. Βάση αυτού του δεδομένου ορίσαμε ως κυκλοφοριακή συμφόρηση όταν ο χρήστης καθυστερήσει να περάσει ένα σηματοδότη πάνω από πέντε λεπτά. Επιπροσθέτως, καθυστέρηση πάνω από πέντε λεπτά θεωρείται σημαντική και αξίζει να καταγράφεται.

4.1. Ο αλγόριθμος

Ένας αποδοτικός αλγόριθμος είναι πολύ σημαντικός για την αντιμετώπιση του προβλήματος. Οι περισσότερες εφαρμογές χρησιμοποιούν απλά την προσέγγιση του ορίου ταχύτητας. Μέσω ενός request στο server της Google, για κάθε σημείο στο χάρτη, γίνεται η αντιστοίχιση πάνω στο οδικό δίκτυο και επίσης δίνεται το όριο ταχύτητας. Μέσω του ορίου ταχύτητας μπορούμε εύκολα να εντοπίσουμε την κυκλοφοριακή συμφόρηση τόσο μέσα στις πόλεις όσο και σε εθνικές οδούς. Ωστόσο, η τιμή του ορίου ταχύτητας (speed limit) δεν διατίθεται δωρεάν παρά μόνο σε όσους διαθέτουν google maps api for work, γεγονός το οποίο λειτούργησε αποτρεπτικά στη χρήση της προσέγγισης αυτής.

Για την αντιμετώπιση του προβλήματος αναπτύχθηκε ένας αλγόριθμος που εντοπίζει την κυκλοφοριακή συμφόρηση μέσα σε κατοικημένες περιοχές ενώ απορρίπτει τις περιπτώσεις που ο χρήστης βρίσκεται σκοπίμως σε στάση χωρίς την παρουσία κίνησης. Επίσης,

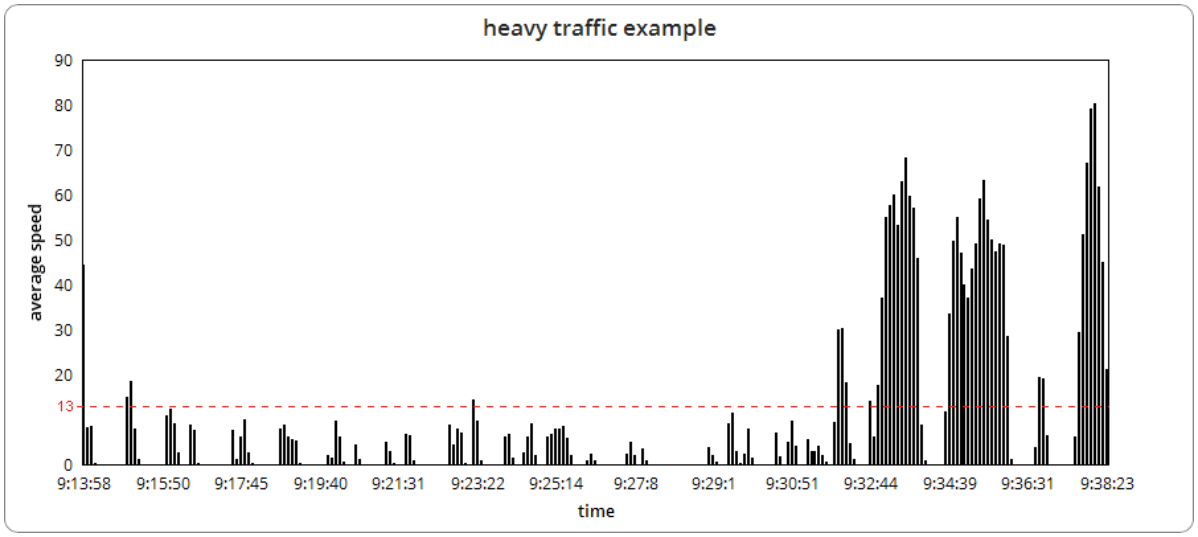
διακρίνεται η και απομονώνεται η περίπτωση όπου ο χρήστης κινείται μέσα στην πόλη, σε στενά, αλλάζοντας συνεχώς πορεία και κινούμενος με χαμηλή ταχύτητα χωρίς την παρουσία κίνησης.

Αρχικά, διαβάζουμε από τη βάση δεδομένων τα αναγνωριστικά (id) που αντιστοιχούν σε διαφορετικές διαδρομές. Για κάθε αναγνωριστικό συγκεντρώνουμε τα δεδομένα σε μια λίστα. Όταν αυτά συγκεντρωθούν καλούμε πρώτα μια συνάρτηση για τον υπολογισμό μεγάλης κυκλοφοριακής συμφόρησης και στη συνέχεια για τις ίδιες τιμές μια συνάρτηση για τον υπολογισμό μέτριας ή μικρής κυκλοφοριακής συμφόρησης.

Τμήμα του αλγορίθμου για τον υπολογισμό μεγάλης κυκλοφοριακής συμφόρησης (calc heavy traffic()).

```
For ( κάθε σημείο της συγκεκριμένης διαδρομής ) {
        If  ( average speed <= 13 ) {
                Moving = 0;
        }else if ( average speed > 13 ) {
                Moving++;
        }
        If ( moving < 3 ) {
                Συλλέγουμε τις συντεταγμένες των σημείων σε μια λίστα
        }else {
                If ( τα συνεχόμενα σημεία που έχουν συλλεγεί είναι περισσότερα από 60 ) {
                        If ( !Check if stopped() && Check if bearing changed() ) {
                                Στα σημεία που έχουμε στη λίστα υπάρχει συμφόρηση.
                                Εμφάνισε τα στο χάρτη.
                        }
                }
                Σβήνουμε όλες τις καταχωρήσεις της λίστας.
        }
}
```

Ο αλγόριθμος μας όσο βρίσκει συνεχόμενα μέσες ταχύτητες μικρότερες από 13 km/h τις αποθηκεύει σε μια λίστα μαζί με τις συντεταγμένες των σημείων. Όταν βρεθούν τρεις συνεχόμενες μέσες ταχύτητες μεγαλύτερες από 13 km/h τότε ελέγχει αν όσα στοιχεία έχουν συλλεχθεί είναι πάνω από 60. Το όριο των 13 χιλιομέτρων την ώρα τέθηκε έπειτα από συνεχείς ελέγχους και δοκιμές του αλγορίθμου στην πράξη. Αρχικά, το όριο δοκιμάστηκε στα 10 km/h αλλά βρέθηκε πως είναι πολύ χαμηλό και δεν ανταποκρίνεται σε πραγματικές συνθήκες κίνησης.



(Εικόνα 10) Δεδομένα που συλλέχθηκαν σε διαδρομή με μεγάλη κυκλοφοριακή συμφόρηση.

Όπως έχει προαναφερθεί, δεδομένα συλλέγονται κάθε πέντε δευτερόλεπτα, οπότε 60 σημεία συνεχόμενα αντιπροσωπεύουν δεδομένα πέντε λεπτών (12 φορές δειγματοληπτούμε μέσα σε ένα λεπτό, 12*5λεπτα = 60 σημεία σε πέντε λεπτά). Οπότε εάν τα δεδομένα που έχουν συλλεχθεί και άρα το μέγεθος της λίστας μας είναι μεγαλύτερο του 60 έχουμε αρκετά δεδομένα για να συνεχίσουμε τον έλεγχο για κυκλοφοριακή συμφόρηση. Αν τα δεδομένα, δηλαδή ο αριθμός των στοιχείων μέχρι να βρούμε τις τρεις συνεχόμενες ταχύτητες πάνω από το όριο (threshold), είναι λιγότερα από 60 απλά δεν το θεωρούμε ως κυκλοφοριακή συμφόρηση και διαγράφουμε τα στοιχεία αυτά από τη λίστα.

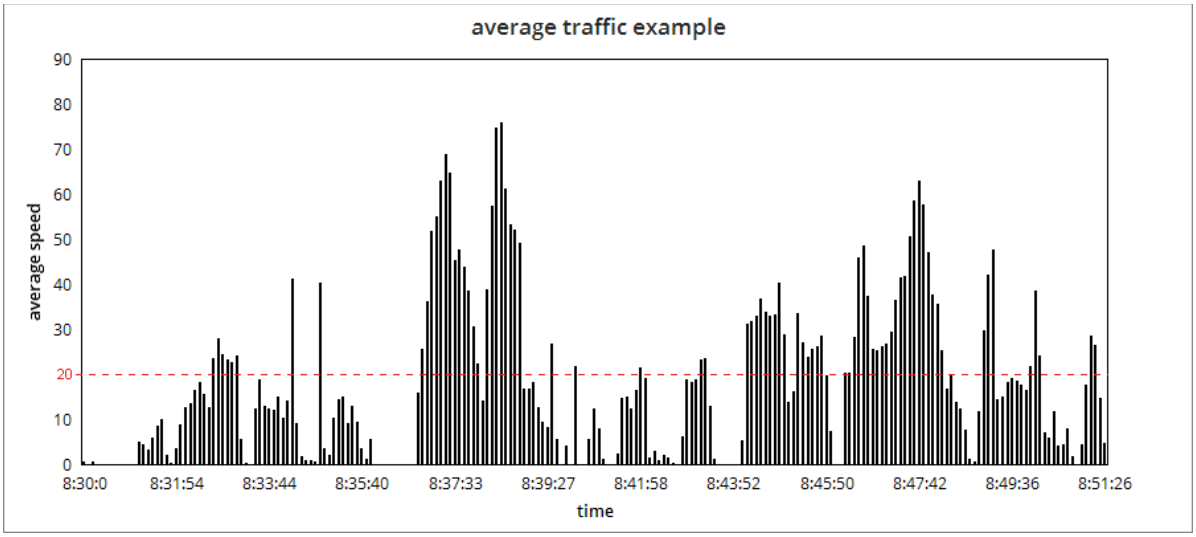
Ο αλγόριθμος για μικρή ή μέτρια συμφόρηση δεν διαφέρει πολύ από τον αλγόριθμο που μόλις εξηγήσαμε αλλά αξίζει να τον παρουσιάσουμε για να επισημανθούν οι λίγες διαφορές που έχει και γιατί αυτές έχουν γίνει.

Ο αλγόριθμος για τον υπολογισμό μικρής ή μέτριας κυκλοφοριακής συμφόρησης (calc average traffic()).

```
For ( κάθε σημείο της συγκεκριμένης διαδρομής ) {
        If  ( average speed <= 20 ) {
                Moving = 0;
        }else if ( average speed > 20 ) {
                if ( στιγμιαία ταχύτητα <= 1.0){
                        low speed++;
                }
                Moving++;
        }
        If ( moving < 3 ) {
                Συλλέγουμε τις συντεταγμένες των σημείων σε μια λίστα
        }else {
                If ( τα συνεχόμενα σημεία που έχουμε συλλέξει είναι περισσότερα από 60 )
        {
                        If ( !Check if stopped() && Check if bearing changed() ) {
                                If ( low speed >= (συνόλου στοιχείων λίστας) / 2 ) {
                                        Στα σημεία που έχουμε στη λίστα υπάρχει συμφόρηση
                                        Εμφάνισε τα στο χάρτη.
                                }
                        }
                }
                Σβήνουμε όλες τις καταχωρήσεις της λίστας.
        }
}
```

Ο συγκεκριμένος αλγόριθμος παρουσιάζει μεγάλη ομοιότητα με τον αλγόριθμο για εύρεση μεγάλης κυκλοφοριακής συμφόρησης που παρουσιάσαμε πρώτο. Στην προκειμένη περίπτωση έχουμε θέσει το όριο ταχύτητας στα 20km/h με σκοπό να δεχόμαστε

περισσότερες περιπτώσεις (Εικόνα 11). Το πρόβλημα με τη συγκεκριμένη προσέγγιση είναι ότι, ο χρήστης μπορεί να κινηθεί με τόσο μικρή ταχύτητα μέσα σε στενά τις πόλης ώστε ο αλγόριθμος εσφαλμένα να το παρουσιάσει ως συμφόρηση. Για την αντιμετώπιση του προβλήματος χρησιμοποιήθηκε η μεταβλητή (low speed) που μετράει πόσες φορές ο χρήστης σημείωσε ταχύτητα μικρότερη του ενός χιλιομέτρου την ώρα. Σε περιπτώσεις συμφόρησης, όπου οι ταχύτητες είναι χαμηλές και τα οχήματα σταματούν συχνά παρατηρήθηκε ότι η μεταβλητή low speed (και άρα οι τιμές της στιγμιαίας ταχύτητας που είναι μικρότερες του 1) ήταν λίγο περισσότερες από τις μισές, της συνολικής λίστας. Αντίθετα, σε περίπτωση κίνησης μέσα σε πυκνοκατοικημένες περιοχές, χωρίς την ύπαρξη κυκλοφοριακής συμφόρησης οι ταχύτητες από τη μια ήταν μικρότερες από 20km/h όμως από την άλλη η τιμή της μεταβλητής (low speed) ήταν πολύ χαμηλότερη.



(Εικόνα 11) Δεδομένα που συλλέχθηκαν σε διαδρομή που παρουσιάζει μικρή έως μέτρια κυκλοφοριακή συμφόρηση στη μέση μεταξύ περίπου (8:39:27 - 8:43:52).

Εξετάζοντας περαιτέρω και τους δυο αλγορίθμους και αν υπάρχουν αρκετά στοιχεία για να συνεχίσουμε, όπως διαπιστώνουμε και από τα παραπάνω, καλούνται οι συναρτήσεις *Check if stopped() και Check if bearing changes()* για να ελέγξουμε αν ο χρήστης έχει σκόπιμα κάνει στάση ή αν βρίσκεται μέσα σε στενά και ίσως ψάχνει για πάρκινγκ οπότε και για αυτό σημειώνει χαμηλές μέσες ταχύτητες. Οι δύο αυτές συναρτήσεις δέχονται ως είσοδο τα σημεία που έχουμε συγκεντρώσει, τα οποία περιέχονται στη λίστα και επιστρέφουν μια Boolean τιμή ανάλογα με το αποτέλεσμα.

Η Check if stopped() είναι η ακόλουθη

```
For ( κάθε στοιχείο της λίστας μας ) {

        If  ( ταχύτητα ( speed ) του στοιχείου i είναι 0 ) {

                If  ( ταχύτητα ( speed ) του στοιχείου i+1 > 5 km/h ) {

                        //Το όχημα είχε σταματήσει και ξεκίνησε
                        Stop move++;

                }
        }

        If  ( ταχύτητα ( speed ) του στοιχείου i είναι 0 ) {

                Stopped temp++;

        }else if  ( Stopped temp > stopped ) {

                stopped = Stopped temp;
                Stopped temp = 0;

        }else  {

                Stopped temp = 0;

        }
}
if  ( μέγεθος λίστας * ( 3 / 5 ) > stopped ) {

        return false;

} else {

        return !( Stop move >= 3);

}
```

Στην αρχή, ο αλγόριθμος εκτελεί μια επανάληψη για όλα τα στοιχεία της λίστας που αναφέρουμε πιο πάνω και υποπτευόμαστε ότι παρουσιάζουν κυκλοφοριακή συμφόρηση. Ελέγχει αρχικά την ταχύτητα του εκάστοτε στοιχείου της λίστας και του επόμενου του (εφόσον υπάρχει) για να διαπιστώσει αν το όχημα βρισκόταν σε ακινησία και την επόμενη

στιγμή ξεκίνησε. Για τον λόγο αυτό χρησιμοποιείται η μεταβλητή (Stop move) που μετράει πόσες φορές ο χρήστης ξεκίνησε από ακινησία.

Στη συνέχεια, ελέγχουμε όλα τα στοιχεία για να βρούμε το μεγαλύτερο σύνολο με συνεχόμενες στιγμιαίες ταχύτητες 0. Δηλαδή, επιθυμούμε να βρούμε το μεγαλύτερο χρονικό διάστημα κατά το οποίο το όχημα παρέμεινε σε συνεχόμενη ακινησία. Αν το διάστημα αυτό είναι ίσο ή μεγαλύτερο από τα 3/5 του συνολικού διαστήματος που υποπτευόμαστε (βάση των μέσων ταχυτήτων που ελέγξαμε) ότι παρουσιάζει κυκλοφοριακή συμφόρηση, τότε απορρίπτουμε τις τιμές καθώς το όχημα μάλλον ήταν απλά σε στάση. Η τιμή 3/5 θεωρείται ως κατάλληλη στην συγκεκριμένη περίπτωση, καθώς αν η στάση είναι μικρότερη από 3/5, τα υπόλοιπα δεδομένα, που θα είναι τουλάχιστον το 2/5 των συνολικών, είναι αρκετά για να εκτιμήσουμε αν πράγματι υπάρχει συμφόρηση ή όχι. Για παράδειγμα, έστω ότι έχουμε μόνο 60 συνεχόμενες τιμές με μέση ταχύτητα μικρότερη από το όριο (13 km/h) και στις 35 από αυτές παρατηρούμε ότι το όχημα βρισκόταν σε στάση. Για τις υπόλοιπες 25 που πάλι η μέση ταχύτητα θα είναι μικρότερη από τα 13km/h αν δεν υπάρχει συμφόρηση δεν θα καλύπτεται η συνθήκη (Stop move >= 3) ακόμα και αν χρήστης συναντήσει φανάρι αμέσως μόλις ξεκινήσει.

Κατά μέσο όρο παρατηρήθηκε ότι ακόμα και σε περίπτωση μεγάλης κυκλοφοριακής συμφόρησης το όχημα θα κινηθεί περισσότερες από τρεις φορές με το μέσο όρο (από τα δεδομένα που συλλέχθηκαν) να είναι στις 4-5 σε κάθε διάστημα πέντε λεπτών. Γενικά, αυτό που ελέγχουμε είναι, σε όλο το διάστημα που υποπτευόμαστε ότι μπορεί να παρουσιάζετε συμφόρηση, που ίσως για παράδειγμα να είναι ένα διάστημα 10 λεπτών, ο χρήστης να έχει ξεκινήσει και σταματήσει πάνω από 3 φορές. Αλλιώς μάλλον είναι απλά σταθμευμένος στην άκρη του δρόμου.

Στη συνάρτηση αυτή (*Check if stopped()*) του αλγορίθμου, εν αντιθέσει με προηγουμένως, χρησιμοποιούμε την στιγμιαία ταχύτητα και όχι τη μέση. Ο λόγος που ακολουθούμε αυτή την προσέγγιση είναι γιατί η μέση ταχύτητα υπολογίζεται βάση της απόστασης δυο διαδοχικών σημείων. Επειδή, η ακρίβεια του gps δεν είναι πάντοτε απόλυτη ακόμα και όταν το όχημα είναι σταματημένο δυο διαδοχικά σημεία μπορεί να απέχουν μεταξύ τους λίγα μέτρα. Ενώ, σε αυτή την περίπτωση θα σημειώσουμε στιγμιαία ταχύτητα και στα δυο αυτά σημεία 0, η μέση ταχύτητα μπορεί να δώσει τιμές από 0 μέχρι 2 περίπου. Ένα τέτοιο αποτέλεσμα θα αλλοίωνε αρκετά τα αποτελέσματα του κώδικα μας γι' αυτό και χρησιμοποιήθηκε η στιγμιαία ταχύτητα.

Η Check if bearing changed() είναι η ακόλουθη

```
Old bearing = 0;
For ( κάθε στοιχείο της λίστας ) {
        If ( στιγμιαία ταχύτητα >= 2 ) {
                Bearing = bearing ( κατεύθυνση ) του στοιχείου i
                If ( Old bearing != 0 ) {
                        If ( απόλυτη τιμή του ( Old bearing - Bearing ) >= 80 ) {
                                Change direction++;
                        }
                }
                Old bearing =  bearing ( κατεύθυνση ) του στοιχείου i
        }
}
If ( Change direction < ( σύνολο των στοιχείων της λίστας / 4 ) ) {
        return true;
} else {
        return false;
}
```

Επειδή στην Ελλάδα και πιθανώς και σε άλλες χώρες υπάρχει έλλειψη χώρων στάθμευσης συνηθίζεται το φαινόμενο οι οδηγοί να ψάχνουν για θέση παρκινγκ αρκετή ώρα. Σε αυτό το διάστημα και καθώς στρίβουν στα στενά η κατεύθυνση του οχήματος αλλάζει. Αυτό προσπαθούμε να μετρήσουμε στο συγκεκριμένο σημείο και να απορρίψουμε τις περιπτώσεις που μοιάζουν με συμφόρηση αλλά δεν είναι.

Αρχικά, όπως και προηγουμένως, εκτελούμε επανάληψη για κάθε στοιχείο από αυτά που περιέχονται στη λίστα μας και είναι υποψήφια για περίπτωση κυκλοφοριακής συμφόρησης.

Η τιμή της κατεύθυνσης, παρατηρήθηκε, ύστερα από δοκιμές, ότι όταν το όχημα κινείται δίνει στο 95% των περιπτώσεων σωστό αποτέλεσμα. Απεναντίας, όταν το όχημα βρίσκεται σε στάση, η τιμή της κατεύθυνσης (bearing) αλλάζει πολύ δίνοντας σωστή τιμή περίπου στο μισό των περιπτώσεων. Αυτό γίνεται διότι, το κινητό τηλέφωνο υπολογίζει την κατεύθυνση μεταξύ δυο διαδοχικών θέσεων. Αν το όχημα βρίσκεται σε στάση και οι δύο διαδοχικές θέσεις είναι το ίδιο σημείο δεν μπορεί να υπολογιστεί η κατεύθυνση (bearing).

Ξανά εδώ όπως και προηγουμένως χρησιμοποιούμε την στιγμιαία ταχύτητα. Θέλουμε να δούμε αν ακριβώς τη στιγμή της δειγματοληψίας ο χρήστης ήταν εν κινήσει γιατί είναι σημαντικό για τον υπολογισμό της σωστής κατεύθυνσης.

Έπειτα, ελέγχουμε ανά δύο την κατεύθυνση των περιπτώσεων με ταχύτητα μεγαλύτερη των 2km/h. Αν η διαφορά στο bearing (γενικά το bearing κυμαίνεται από 0.0 έως 360.0 μοίρες) είναι μεγαλύτερη των 80 μοιρών τότε ο χρήστης πιθανότατα έχει στρίψει σε κάποια διασταύρωση όποτε το σημειώνουμε.

Τέλος, αν το σύνολο των σημείων στα οποία παρατηρήσαμε αλλαγή κατεύθυνσης είναι μεγαλύτερο από το 1/4 της λίστας, τότε απορρίπτουμε την περίπτωση συμφόρησης. Γενικά, παρατηρήθηκε πως οι αλλαγές κατεύθυνσης που σημειώνουμε κυμαίνονται σε κανονικές διαδρομές από 0 μέχρι 8. Το 1/4 λοιπόν επιλέχθηκε ως όριο για να διαχειριστούμε και πιθανές λάθος τιμές τις κατεύθυνσης και για να απορρίψουμε περιπτώσεις που μοιάζουν με κυκλοφοριακή συμφόρηση αλλά δεν είναι.

4.2. Το web service

Το web service μας είναι μια html-servlet ιστοσελίδα. Το html είναι το κομμάτι που καλεί και λαμβάνει απαντήσεις από το servlet και σχεδιάζει εμφανισιακά την ιστοσελίδα. Στην αρχική μας σελίδα εμφανίζεται ένας χάρτης με ένα κουμπί " traffic ". Όταν ο χρήστης πατήσει αυτό το κουμπί καλείται το servlet για να υπολογίσει την κίνηση και να την εμφανίσει στο χρήστη. Το servlet είναι αυτό που αναλαμβάνει την επικοινωνία με τη βάση δεδομένων, διαβάζει τα δεδομένα και τα δίνει ως είσοδο στον αλγόριθμο.

Η βάση δεδομένων μας περιέχει τις τιμές:

- Αναγνωριστικό (id)
- Χρόνος (time)

- Γεωγραφικό πλάτος (latitude)
- Γεωγραφικό μήκος (longitude)
- Μέση ταχύτητα (average speed)
- Στιγμιαία ταχύτητα (speed)
- Κατεύθυνση (bearing)
- Απόσταση (distance)

Ο αλγόριθμος υπολογίζει τα σημεία στα οποία εμφανίζεται κυκλοφοριακή συμφόρηση και κάθε θέση του χρήση στην περιοχή συμφόρησης αποθηκεύεται σε μια λίστα. Η λίστα αυτή στη συνέχεια στέλνεται στο html. Όπως αναφέραμε και στο κεφάλαιο 3 οι συντεταγμένες που θα έχουμε μπορεί ή να είναι πάνω στο οδικό δίκτυο (αν ο χρήστης είχε δώσει τον προορισμό του στην εφαρμογή), ή να μην έχουν αντιστοιχιστεί σε δρόμο. Σε οποιαδήποτε από τις δυο περιπτώσεις αυτές καλούμε την roads api της Google. Μέσω url της μορφής

https://roads.googleapis.com/v1/snapToRoads?path=-35.27801,149.12958|-35.28032,149.12907|&interpolate=true&key=API_KEY

Δίνουμε ως path τις συντεταγμένες κάθε σημείου από αυτά που έχουμε συγκεντρώσει (εμφανίζουν συμφόρηση) και θέλουμε να εμφανίσουμε. Εφόσον κάθε αίτηση (request) δέχεται μέχρι και 100 σημεία, κάποιες φορές χρειάζεται να κάνουμε περισσότερα από ένα request αλλά αυτό δεν επηρεάζει το αποτέλεσμα. Ενδεικτικά το σύνολο των σημείων μας θα μπορούσε να είναι όπως αυτό στην (Εικόνα 12). Η κόκκινη γραμμή είναι η polyline που θα μας επιστρέψει η roads api.

(Εικόνα 12) Πηγή: http://googleforwork.blogspot.gr/2013/11/smarter-ways-to-plan-and-optimize.html

Η αρχική μας προσέγγιση ήταν να συνδέσουμε τα σημεία μεταξύ τους με τη βοήθεια της directions api. Όπως έχουμε αναφέρει το api αυτό δέχεται ως είσοδο τις συντεταγμένες δυο σημείων και επιστρέφει την κοντινότερη διαδρομή προκειμένου να ενωθούν τα δύο σημεία

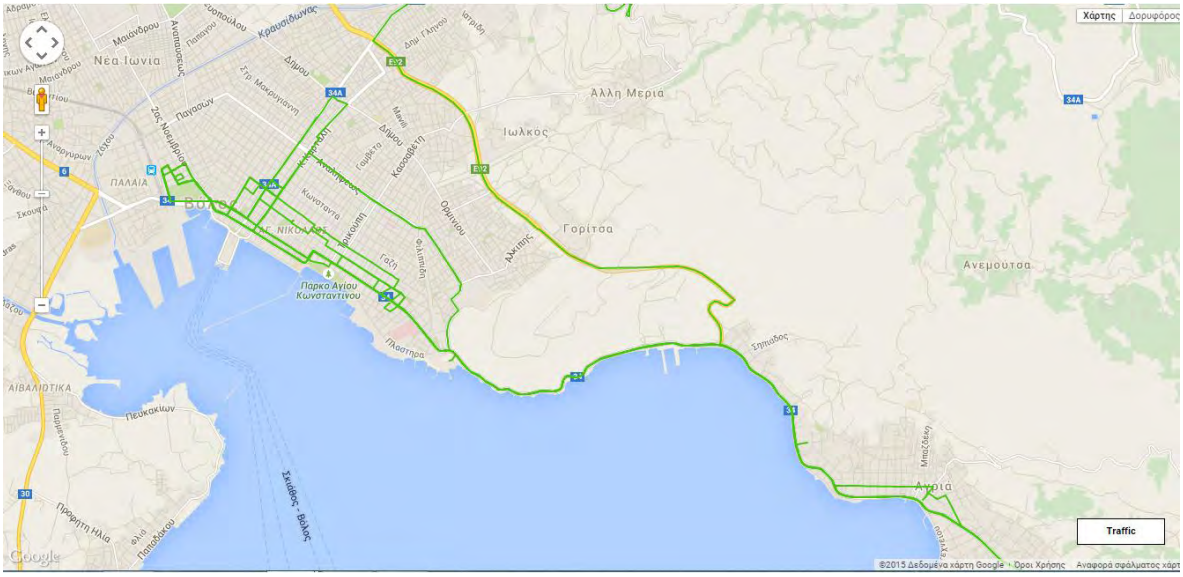
αυτά. Γρήγορα παρατηρήθηκε ότι επειδή οι συντεταγμένες των σημείων μας δεν είναι πάνω στο οδικό δίκτυο, όταν γίνεται η αντιστοίχιση τους στον κοντινότερο δρόμο μπορεί να τοποθετηθούν σε κάποιο παράλληλο ή κάθετο του πραγματικού δρόμου που βρισκόταν ο χρήστης. Όταν λοιπόν η directions api προσπαθεί να συνδέσει τα σημεία αυτά διαπιστώθηκε πως δίνει ένα αποτέλεσμα που στρίβει σε στενά που ο χρήστης δεν πέρασε, κάνει αναστροφή σε κεντρικούς δρόμους ή ακόμα και κύκλους σε ολόκληρα οικοδομικά τετράγωνα. Προφανώς τα περισσότερα από αυτά τα αποτελέσματα οδηγούσαν σε μια διαδρομή που απείχε από την πραγματική τουλάχιστον κατά το ήμισυ.

Το σημαντικό με τη roads api και ο λόγος που τη χρησιμοποιήσαμε είναι ότι, αντίθετα με τη directions api, λαμβάνει υπόψη το σύνολο των σημείων που δίνουμε και με μαθηματικές συναρτήσεις βρίσκει μια γραμμή που να περνάει από δρόμο και να είναι κοντά και στις συντεταγμένες μας.

Επιπροσθέτως, από τη στιγμή που έχουμε αποθηκεύσει και τη χρονική στιγμή που ο χρήστης βρέθηκε σε κάθε σημείο, δημιουργούμε ένα παράθυρο κάθε φορά που ο χρήστης επιλέγει μια polyline στο χάρτη και εμφανίζουμε την ώρα που παρατηρήθηκε συμφόρηση στο συγκεκριμένο σημείο.

5. Έλεγχος και επαλήθευση του αλγορίθμου (testing)

Η συσκευή που χρησιμοποιήθηκε είναι ένα κινητό τηλέφωνο Lenovo A536 με έκδοση λειτουργικού Android 4.4.2. Ο έλεγχος και η επαλήθευση του αλγορίθμου μας έγινε με πάνω από 100 διαδρομές στην πόλη του Βόλου και περίπου 10 στην Αθήνα. Ένα μέρος των διαδρομών στην πόλη του Βόλου απεικονίζεται στην (Εικόνα 13). Οι διαδρομές αυτές στην πόλη του Βόλου πραγματοποιήθηκαν καθημερινές αλλά και σαββατοκύριακα, σε πρωινές αλλά και βραδινές ώρες ξεκινώντας από τον μήνα Μάρτιο έως τον μήνα Ιούνιο. Καλύπτουν χρονικά διαστήματα από 10 λεπτά έως μια ώρα. Εκτός των διαδρομών μέσα στην πόλη του Βόλου υπάρχουν και οι διαδρομές από Μακρινίτσα προς Αγριά (πραγματοποιήθηκε 1η Μαΐου), από Βόλο προς κάτω Γατζέα (πραγματοποιήθηκε 7/5/2015) και από Βόλο προς Αγριά. Στην πόλη της Αθήνας πραγματοποιήθηκαν ελάχιστες διαδρομές στο χρονικό διάστημα 10 μέχρι 21 Απριλίου, κυρίως πρωινές ώρες. Η σημαντικότερη, από πλευράς παρουσίας κυκλοφοριακής συμφόρησης, πραγματοποιήθηκε στις 21 Απριλίου στο κέντρο της Αθήνας και είναι μια διαδρομή τριών ωρών. Τα αποτελέσματα μας για τη διαδρομή αυτή φαίνονται στην (Εικόνα 14)

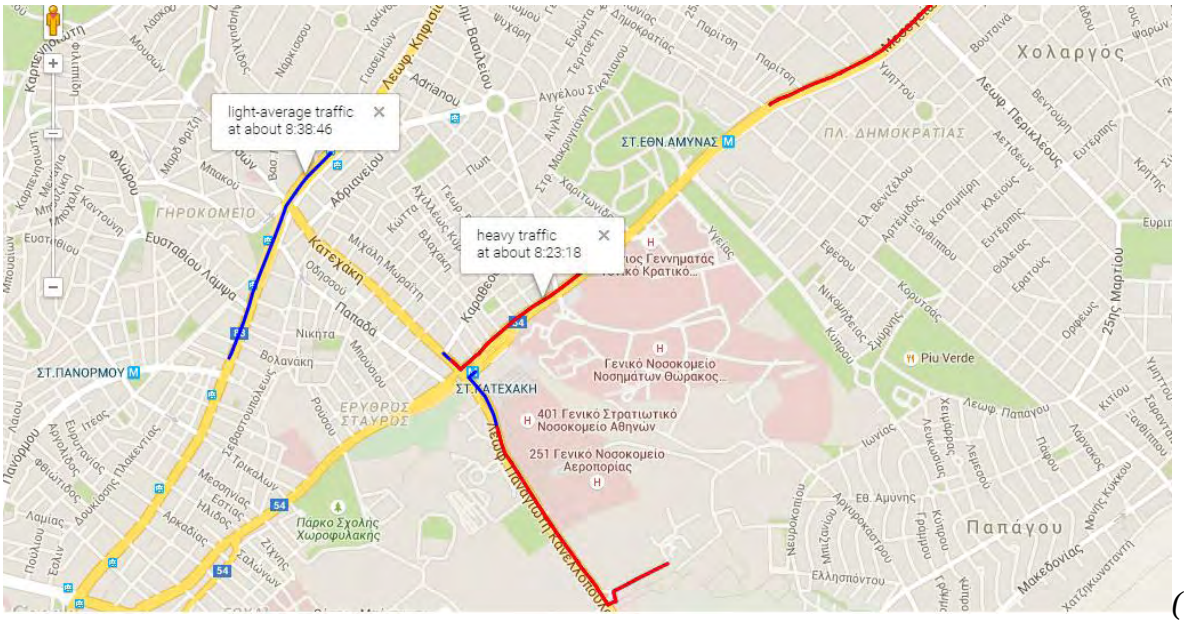


(Εικόνα 13) Διαδρομές στην πόλη του Βόλου

Δυστυχώς, στην πόλη του Βόλου δεν βρέθηκε κάποιο σημείο που να παρουσιάζει κυκλοφοριακή συμφόρηση και για το λόγο αυτό οι διαδρομές απεικονίζονται με πράσινο και όχι με κόκκινο χρώμα.

Για να συλλέξουμε τα δεδομένα κάθε διαδρομής και μιας και ο server είναι σε localhost (οπότε δεν μπορεί να επιτευχθεί επικοινωνία της συσκευής μαζί του αν είναι σε διαφορετικό δίκτυο) σε κάθε διαδρομή που εκτελούσαμε αποθηκεύαμε όλα τα δεδομένα σε ένα αρχείο. Στη συνέχεια, και σε τοπικό δίκτυο ανεβάζουμε μέσω του web service το αρχείο που έχουμε συλλέξει και μας εμφανίζει τη διαδρομή και τα σημεία συμφόρησης αν υπάρχουν.

Για παράδειγμα, στην Αθήνα παρατηρήθηκε συμφόρηση άρα τα δεδομένα που θα παρουσιάζονταν στους χρήστες στην ιστοσελίδα θα ήταν αυτά που φαίνονται στην ακόλουθη εικόνα.



Εικόνα 14) Με κόκκινο χρώμα εξεικονίζεται μεγάλη κυκλοφοριακή συμφόρηση και με μπλε μέτρια έως λίγη.

Από όσες διαδρομές εκτελέστηκαν μόνο σε μια εντοπίστηκε κυκλοφοριακή συμφόρηση και αυτή είναι που φαίνεται στην (Εικόνα 14). Ο αλγόριθμος εντόπισε με επιτυχία και τα τρία σημεία έντονης κυκλοφοριακής συμφόρησης αλλά και το ένα με μέτρια συμφόρηση. Με το δείγμα αυτό και λόγο απουσίας περισσοτέρων διαδρομών που να παρουσιάζουν συμφόρηση μπορούμε μόνο με επιφύλαξη να διαπιστώσουμε πως ο αλγόριθμος έχει πολύ καλά ποσοστά επιτυχίας.

6. Πλεονεκτήματα και μειονεκτήματα της εφαρμογής και μελλοντική εξέλιξη

Η εφαρμογή μας, όπως και κάθε άλλη εφαρμογή άλλωστε, παρουσιάζει κάποια πλεονεκτήματα και κάποια μειονεκτήματα.

Το σημαντικότερο μειονέκτημα είναι η κατανάλωση ενέργειας κατά την διάρκεια λειτουργίας της εφαρμογής. Καθώς τα smartphones τα τελευταία χρόνια καταναλώνουν ενέργεια για να προσφέρουν στο χρήστη καλύτερη απόδοση, η μέση διάρκεια λειτουργίας ενός κινητού τηλεφώνου πριν να χρειαστεί φόρτιση είναι περίπου μια μέρα. Με συνεχή χρήση της εφαρμογής μας και λόγο χρήσης του gps που καταναλώνει τεράστια ποσά ενέργειας η διάρκεια αυτονομίας της συσκευής περιορίζεται στις δυόμιση περίπου ώρες. Παρότι είναι μια εφαρμογή που στοχεύει κυρίως στον εντοπισμό της κυκλοφοριακής συμφόρησης μέσα σε μεγάλα αστικά κέντρα, στα οποία σπάνια θα χρειαστεί χρήση της εφαρμογής για περισσότερη από μια ώρα, το πρόβλημα αυτό παραμένει αρκετά σοβαρό.

Το δεύτερο σημαντικό μειονέκτημα της εφαρμογής, είναι ότι δεν μπορέσαμε να κάνουμε χρήση των ορίων ταχύτητας (speed limits) που παρέχει η Google για ολόκληρο το οδικό δίκτυο κάθε χώρας. Η δυνατότητα να γνωρίζουμε το όριο ταχύτητας σε κάθε τοποθεσία που λαμβάνουμε από το χρήστη θα βοηθούσε σημαντικά, όχι μόνο επικυρώνοντας τα αποτελέσματα του αλγορίθμου μας, αλλά και για να μπορέσουμε να βρούμε την συμφόρηση στις εθνικές οδούς. Σε πολλές χώρες του εξωτερικού η κίνηση με 50km/h σε μια κεντρική οδική αρτηρία με όριο 120km/h θεωρείται συμφόρηση. Χωρίς την γνώση του ορίου ταχύτητας είναι αδύνατο για τον αλγόριθμο μας να το εντοπίσει.

Παρόλα τα μειονεκτήματα που εμφανίζει η εφαρμογή μας, παρουσιάζει και αρκετά πλεονεκτήματα.

Παρόλο που ακόμα είναι σε πολύ αρχικό στάδιο για να μπορέσει να ανταγωνιστεί άλλες εφαρμογές του εμπορίου, το μεγάλο πλεονέκτημα είναι ότι χρησιμοποιεί έναν αρκετά αποδοτικό αλγόριθμο. Το σημαντικό δεν είναι μόνο ο εντοπισμός της κυκλοφοριακής συμφόρησης αλλά και η απομόνωση και απόρριψη περιπτώσεων που μοιάζουν με συμφόρηση αλλά δεν είναι (όπως στάση στην άκρη του δρόμου, κίνηση του οχήματος σε στενά κλπ.). Επιπροσθέτως, μεγάλο πλεονέκτημα είναι ότι σε αντίθεση με άλλες εφαρμογές όπως για παράδειγμα το Waze, ο χρήστης δεν χρειάζεται να αλληλεπιδρά με το κινητό ή να

πατά κάποιο κουμπί για να ενημερώσει ότι υπάρχει συμφόρηση καθώς αυτό πραματοποιείται αυτόματα μέσω του αλγορίθμου.

Επίσης, αρκετά σημαντικό για την εφαρμογή μας είναι ότι δεν παρέχει στον χρήστη μόνο δεδομένα σχετικά με την κυκλοφοριακή συμφόρηση αλλά λειτουργεί και ως σύστημα πλοήγησης. Συνοψίζοντας όλα τα στοιχεία της εφαρμογής είναι εμφανές ότι υπάρχουν πολύ μεγάλα περιθώρια για ανάπτυξη.

Όπως προαναφέρθηκε, ο server και η βάση δεδομένων μας βρίσκονται σε localhost οπότε ένα ακόμα βήμα θα ήταν η μεταφορά τους σε πραγματικό server που θα εξυπηρετεί τις συσκευές και απομακρυσμένα. Με την ολοκλήρωση του βήματος αυτού μένει να δώσουμε τη δυνατότητα στο server να στέλνει απάντηση στις συσκευές των χρηστών σχετικά με τα σημεία που παρουσιάζουν κυκλοφοριακή συμφόρηση.

Όταν πραγματοποιηθεί το παραπάνω βήμα η εφαρμογή θα είναι πλήρως λειτουργική και θα μπορούν να ακολουθήσουν εργασίες για περαιτέρω βελτίωσή της. Αν υπάρξει μια συνεννόηση με την Google μπορεί να αποκτηθεί και να χρησιμοποιηθεί το Google play for Work, το οποίο είναι ένα πακέτο που δίνεται κυρίως σε εταιρίες. Με τη χρήση του πακέτου αυτού θα μπορέσουμε να ενσωματώσουμε τα όρια ταχύτητας για ολόκληρο το οδικό δίκτυο στον αλγόριθμο μας και συνεπώς θα έχουμε τη δυνατότητα να εντοπίσουμε την κυκλοφοριακή συμφόρηση τόσο μέσα στα αστικά κέντρα όσο και στις εθνικές οδούς.

Επιπλέον, θα μπορούσε εύκολα να γίνεται και πρόβλεψη των σημείων που θα παρουσιάσουν μελλοντικά συμφόρηση βάση των δεδομένων που θα συλλέγουμε και αποθηκεύουμε. Αν για παράδειγμα παρατηρείται από τα δεδομένα που λαμβάνουμε από τους χρήστες, πως κάθε Δευτέρα σε κάποιο σημείο υπάρχει κυκλοφοριακή συμφόρηση, θα μπορούσε η εφαρμογή μετά από κάποιο χρονικό διάστημα να ενημερώνει τους χρήστες για συμφόρηση στο συγκεκριμένο σημείο χωρίς όμως αυτή να έχει παρατηρηθεί ή καταγραφεί από κάποιον εκείνη τη στιγμή.

Βιβλιογραφία

1. A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Toledo, J. Eriksson, S. Madden and H. Balakrishnan. *VTrack: accurate, energy-aware road traffic delay estimation using mobile phones,* 2009

2. P. Mohan , V. N. Padmanabhan and R. Ramjee. *Nericell: rich monitoring of road and traffic conditions using mobile smartphones*, 2008

3. *http://en.wikipedia.org/wiki/Google_Traffic*

4. *https://www.waze.com/*

5. A. Lakas and M. Cheqfah *Detection and Dissipation of Road Traffic Congestion Using Vehicular Communication,* 2014

6. Young-Bae Ko and Nitin H. Vaidya. *Flooding-Based Geocasting Protocols for Mobile Ad Hoc Networks*. December 2002

7. V. Jain, A. Sharma and L. Subramanian. *Road traffic congestion in the developing world,* 2012

8. *http://en.wikipedia.org/wiki/Floating_car_data*

9. *http://www.tomtom.com/en_gb/*

10. J. Yoon, B. Noble and M. Liu. *Surface street traffic estimation,* 2007

11. Kanhere, S.S. *Participatory Sensing: Crowdsourcing Data from Mobile Smartphones in Urban Spaces,* 2011

12. Mohamed Taher Alrefaie. *Demo: Road Speed Profile: From GPS Traces to Real-time Traffic Speed,* 2014

13. Ralf-Peter Schäfer, Kai-Uwe Thiessenhusen, Elmar Brockfeld and Peter Wagner. *A traffic information system by means of real-time floating-car data,* 2015

14. *http://www.idc.com/getdoc.jsp?containerId=prUS25450615*

15. *http://en.wikipedia.org/wiki/Android_version_history*

16. *http://en.wikipedia.org/wiki/Android_(operating_system)*

17. *http://www.forbes.com/sites/ewanspence/2015/02/25/android-ios-market-share/*

18. *https://netbeans.org/*

19. *https://developers.google.com/maps/documentation/roads/*

20. *http://appcrawlr.com/android/waze-social-gps-traffic-gas*

21. *https://en.wikipedia.org/wiki/TomTom*

22. B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan and S. Madden. *CarTel: A Distributed Mobile Sensor Computing System,* 2006

23. W. Huber, M. Lädke and R Ogger. EXTENDED FLOATING-CAR DATA

FOR THE ACQUISITION OF TRAFFIC INFORMATION, 1999

24. *https://developers.google.com/maps/documentation/directions/*

25. *http://www.androidhive.info/2012/05/how-to-connect-android-with-php-mysql/*

26. *https://en.wikipedia.org/wiki/Application_programming_interface*

27. *https://developer.android.com/develop/index.html*

28. https://www.apachefriends.org/index.html

29. *https://en.wikipedia.org/wiki/Crowdsourcing*

30. *https://en.wikipedia.org/wiki/Intelligent_transportation_system*