Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Επικοινωνιων, Ηλεκτρονικης και Συστηματων Πληροφορικης

Προσομοίωση Διάδοσης Περιεχομένου σε Ασύρματα Οδικά Δίκτυα (VANETs) με τη χρήση μεθόδων μηχανικής μάθησης

Μελέτη και υλοποίηση

Διπλωματικη Εργασια

του

ΑΣΤΡΙΝΑΚΗ Ι. ΝΙΚΟΛΑΟΥ

Επιβλέπων: Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Συνεπιβλέπουσα: Ειρήνη Κοιλανιώτη
ΕΔΙΠ Ε.Μ.Π.

Αθήνα, Μάρτιος 2023

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Επικοινωνιων, Ηλεκτρονικης και Συστηματων Πληροφορικης

Προσομοίωση Διάδοσης Περιεχομένου σε Ασύρματα Οδικά Δίκτυα (VANETs) με τη χρήση μεθόδων μηχανικής μάθησης

Μελέτη και υλοποίηση

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΑΣΤΡΙΝΑΚΗ Ι. ΝΙΚΟΛΑΟΥ

Επιβλέπων: Συμεών Παπαβασιλείου
Καθηγητής Ε.Μ.Π.

Συνεπιβλέπουσα: Ειρήνη Κοιλανιώτη
ΕΔΙΠ Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 6η Μαρτίου 2023.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Συμεών Παπαβασιλείου	Ιωάννα Ρουσσάκη	Γεώργιος Ματσόπουλος
Καθηγητής Ε.Μ.Π.	Αν. Καθηγήτρια Ε.Μ.Π.	Καθηγητής Ε.Μ.Π

Αθήνα, Μάρτιος 2023

Εθνικο Μετσοβιο Πολυτεχνειο
Σχολη Ηλεκτρολογων Μηχανικων και Μηχανικων Υπολογιστων
Τομεας Επικοινωνιων, Ηλεκτρονικης και Συστηματων Πληροφορικης

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάσει επιστημονικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.

(Υπογραφή)

.....
Αστρινάκης Νικόλαος

6 Μαρτίου 2023

Περίληψη

Η παρούσα διπλωματική εργασία έχει ως στόχο τη δημιουργία προσομοιώσεων ενός ad-hoc οδικού δικτύου και την εφαρμογή διαφόρων μεθόδων διάδοσης περιεχομένου σε αυτές, ώστε να συγκρίνουμε την αποδοτικότητα τους. Η ανάλυση των ad-hoc οδικών δικτύων είναι πολύ σημαντική, καθώς μέσα από αυτή μπορούμε να βελτιώσουμε την οδική ασφάλεια και να κάνουμε την διαδικασία της οδήγησης πιό άνετη.

Ο τρόπος με τον οποίο έγιναν αυτές οι προσομοιώσεις ήταν με τη χρήση των εργαλείων OMNET++, Veins και SUMO. Αναλυτικότερα, γράφτηκε κώδικας ώστε να προστεθούν διάφορες λειτουργίες στις έτοιμες προσομοιώσεις του Veins. Οι βασικότερες εξ αυτών των λειτουργιών είναι η δυνατότητα υπολογισμού διαφόρων μετρικών του δικτύου, η δημιουργία πολιτικών διαγραφής και αποθήκευσης δεδομένων καθώς και ο ορισμός πολιτικών για την διάδοση περιεχομένου.

Ταυτόχρονα, υπάρχει και η δυνατότητα χρήσης αλγορίθμων μηχανικής μάθησης ώστε να καθοριστεί πιό εύκολα και γρήγορα ο βέλτιστος τρόπος διάδοσης περιεχομένου. Οι αλγόριθμοι αυτοί κάνουν χρήση προεκπαιδευμένων μοντέλων νευρωνικών δικτύων της βιβλιοθήκης scikit-learn.

Με αυτές τις λειτουργίες, μπορούμε να εκτελέσουμε την προσομοίωση, θέτοντας τις παραμέτρους που επιθυμούμε (όπως για παράδειγμα τι είδος μετρικών θα χρησιμοποιηθούν) και να μετρήσουμε τους χρόνους απόκρισης, ώστε να έχουμε μία εικόνα για το ποιές μετρικές και πολιτικές είναι αποδοτικότερες για την μετάδοση περιεχομένου.

Λέξεις Κλειδιά

Οδικά ad-hoc δίκτυα, Ανάλυση Κοινωνικών Δικτύων, Διάδοση Περιεχομένου, Νευρωνικά Δίκτυα, Μη επιβλεπόμενη μάθηση, OMNET++, Veins, SUMO

Abstract

The main goal of this diploma thesis is the creation of different simulation scenarios for Vehicular ad-hoc networks and the application of content distribution policies, in order to compare their effectiveness.

To achieve this goal, the Discrete Event Simulator OMNET++, alongside with Veins and SUMO were used. More specifically, the code written adds various functionalities to the pre-existing Veins simulations. The most noteworthy out of those functionalities include metrics calculations and the implementation of caching and content distribution policies.

At the same time, machine learning algorithms can be used to easily and quickly predict the best way to distribute content to the nodes of the network. These algorithms are pre-trained neural network models from the scikit-learn Python library.

Using these new functions, we can run many different simulations, specifying the parameters we want each simulation to have (such as the type of metric used) and measure the response times, in order to determine which metrics and policies are better for content distribution.

Keywords

Vehicular ad-hoc networks, Social Network Analysis, Content Distribution, Neural Networks, Unsupervised Learning, OMNET++, Veins, SUMO

στην οικογένειά μου

Ευχαριστίες

Η παρούσα διπλωματική εργασία σηματοδοτεί το τέλος της πενταετούς μου φοίτησης στη σχολή ΗΜΜΥ, μία περίοδο με πολλές χαρές, αλλά και αρκετές δυσκολίες. Σε αυτή την περίοδο απέκτησα πολλές γνώσεις και εμπειρίες που θα μείνουν αξέχαστες.

Θα ήθελα, καταρχάς, να ευχαριστήσω τον καθηγητή κ. Συμεών Παπαβασιλείου, ο οποίος μου έδωσε την ευκαιρία να εκπονήσω αυτή τη διπλωματική εργασία στο εργαστήριό του. Ακόμη, θέλω να ευχαριστήσω ιδιαίτερα και την κα. Ειρήνη Κοιλανιώτη για τον χρόνο που αφιέρωσε σε εμένα για να με καθοδηγήσει και να με βοηθήσει όσο πιό άμεσα μπορούσε σε ο,τιδήποτε χρειαζόμουν.

Τέλος, θέλω να ευχαριστήσω τους φίλους μου που έκαναν αυτά τα χρόνια υπέροχα, καθώς και τους γονείς μου, Ιωάννη και Ρίτσα, και τις αδερφές μου Μαίρη και Τίνα, που ήταν δίπλα μου και με στήριξαν σε κάθε απόφασή μου.

Αθήνα, Μάρτιος 2023

Αστρινάκης Νικόλαος

Περιεχόμενα

Κατάλογος Σχημάτων

Κεφάλαιο 1

Εισαγωγή

Ένας τομέας που βρίσκεται σε ανάπτυξη στη σημερινή εποχής είναι ο τομέας της Ανάλυσης Κοινωνικών Δικτύων. Μέσα από τη μελέτη κοινωνικών δικτύων, μπορούμε να λάβουμε πληροφορίες για διάφορες τομείς της καθημερινότητάς μας, και οι τεχνικές που εφαρμόζονται για τη μελέτη αυτή, μπορούν να εφαρμοστούν και για άλλα είδη δικτύων.

Ένα τέτοιο είδος είναι τα οδικά ad-hoc δίκτυα, τα οποία είναι δίκτυα από οχήματα, των οποίων η θέση μεταβάλλεται συνέχεια, που επικοινωνούν ασύρματα μεταξύ τους. Τα δίκτυα αυτά έχουν ιδιαίτερο ενδιαφέρον, καθώς μελετώντας τα, μπορούμε να βρούμε τρόπους να αυξήσουμε την οδική ασφάλεια. Ο τρόπος με τον οποίο αυτό μπορεί να επιτευγθεί είναι με την αποτελεσματική μετάδοση περιεχομένου, καθώς η ενημέρωση του οδηγού όσο πιο γρήγορα γίνεται για επικίνδυνες οδικές συνθήκες μπορεί να οδηγήσει στην αποφυγή κάποιου ατυχήματος.

1.1 Συνεισφορά της διπλωματικής

Η κύρια συνεισφορά της διπλωματικής αυτής, είναι η επέκταση ενός πλαισίου προσομοίωσης με διάφορες λειτουργίες, προκειμένου να μπορέσουμε να συγκρίνουμε την απόδοση διαφόρων μετρικών και πολιτικών για τη μετάδοση περιεχομένου. Πιο αναλυτικά, η επέκταση αυτή περιλαμβάνει την υλοποίηση πολιτικών διαμοιρασμού περιεχομένου και της προσθήκης μηχανισμών για την αποστολή και αποδοχή πολυμεσικού περιεχομένου. Ταυτόχρονα, προστέθηκε η δυνατότητα υπολογισμού μετρικών του δικτύου προκειμένου να μπορούν να εντοπιστούν οι πιο σημαντικοί κόμβοι. Παρέχεται, επίσης, και η δυνατότητα χρήσης αλγορίθμων μηχανικής μάθησης για την επιλογή του πιο αποτελεσματικού τρόπου κατανομής περιεχομένου. Με αυτές τις λειτουργίες, τρέξαμε προσομοιώσεις, αποθηκεύοντας τα αποτελέσματα, και συγκρίναμε τον χρόνο απόκρισης που έχει η κάθε μετρική και κάθε αλγόριθμος μηχανικής μάθησης για την κάθε πολιτική διαμοιρασμού που υλοποιήθηκε.

1.2 Οργάνωση του τόμου

Η συγκεκριμένη διπλωματική εργασία, είναι οργανωμένη σε επτά κεφάλαια: Στο Κεφάλαιο 2 παρέχεται το θεωρητικό υπόβαθρο που είναι απαραίτητο για την κατανόηση των εννοιών που αναφέρονται. Πιο συγκεκριμένα εξηγείται τι είναι θεωρία δικτύων, τα κοινωνικά δίκτυα και οι μετρικές που χρησιμοποιούνται στην ανάλυσή τους, ενώ ορίζονται τα ασύρματα

ad-hoc δίκτυα και γίνεται μία εισαγωγή στη μηχανική μάθηση. Στο Κεφάλαιο 3, αναφέρονται τα προαπαιτούμενα προγράμματα για τις προσομοιώσεις και περιγράφεται η χρήση του κάθε ενός. Στο Κεφάλαιο 4 περιγράφεται η δομή του κώδικα, καθώς και οι αλγόριθμοι που χρησιμοποιήθηκαν για την υλοποίηση των λειτουργιών της προσομοίωσης. Στο Κεφάλαιο 5 δίνονται οδηγίες για την εγκατάσταση των απαραίτητων προγραμμάτων και στο Κεφάλαιο 6 παρουσιάζονται οι μετρήσεις που έγιναν με κάποιο σχολιασμό. Τέλος, στο Κεφάλαιο 7, δίνεται ένα συμπέρασμα της συνολικής διπλωματικής εργασίας, καθώς και πιθανές μελλοντικές επεκτάσεις που θα μπορούσαν να γίνουν.

Μέρος Ι

Θεωρητικό Μέρος

Κεφάλαιο 2

Θεωρητικό υπόβαθρο

Στο κεφάλαιο αυτό, θα γίνει μία παρουσίαση των θεωρητικών εννοιών που είναι απαραίτητες για την κατανόηση της παρούσας διπλωματικής εργασίας.

2.1 Θεωρία Δικτύων

2.1.1 Τι είναι η Θεωρία Δικτύων

Αρχικά, η θεωρία δικτύων είναι ένας τομέας που ανήκει στους κλάδους των μαθηματικών, της επιστήμης υπολογιστών και της επιστήμης δικτύων και αποτελεί κομμάτι της θεωρίας γράφων. Ο λόγος που η θεωρία δικτύων είναι τμήμα της θεωρίας γράφων, είναι διότι πρακτικά ένας γράφος και ένα δίκτυο ταυτίζονται. Αν θεωρήσουμε ότι μια ακμή αντιστοιχεί σε ένα κόμβο και μία πλευρά αντιστοιχεί σε έναν σύνδεσμο, τότε μπορούμε να δούμε ότι δεν υπάρχει ουσιαστική διαφορά μεταξύ των δύο εννοιών. [13]

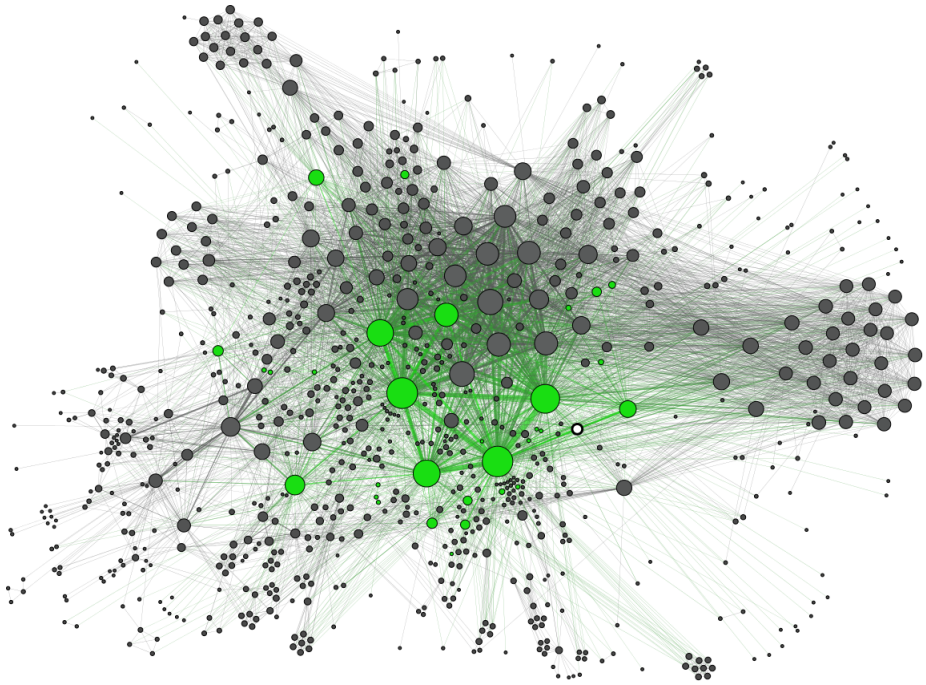
Η βασική έννοια του γράφου ξεκίνησε το 1736 από τον Leonhard Euler, ο οποίος σχεδίασε τη πρώτη οπτική απεικόνιση ενός γράφου για τη λύση του προβλήματος των γεφυρών του Koenigsberg. Ως εκ τούτου, ο Euler έγινε γνωστός και ως ο ¨Πατέρας της Θεωρίας Γράφων¨. [14]

2.1.2 Τι είναι ένα δίκτυο

Ένα δίκτυο ορίζεται ως μία συλλογή αντικειμένων στην οποία κάποια από αυτά τα αντικείμενα είναι συνδεδεμένα ανα δύο μεταξύ τους με σύνδεσμους. Οι σχέσεις αυτές μπορούν να είναι είτε μονόδρομες ή αμφίδρομες και έχουν διαφορετική σημασία ανάλογα με το είδος του δικτύου και το πλαίσιο το οποίο αναλύουμε.

Για παράδειγμα, σε ένα κοινωνικό δίκτυο οι σύνδεσμοι που προκύπτουν μπορούν αντιπροσωπεύουν κοινωνικές σχέσεις, φιλικές σχέσεις ή και οποιοδήποτε είδος διαπροσωπικής επαφής. Οι κόμβοι του δικτύου πάλι εξαρτώνται από το αντικείμενο που θέλουμε να μελετήσουμε. Δηλαδή ένας κόμβος μπορεί να αντιπροσωπεύει ένα άτομο, έναν οργανισμό ή ακόμα και ολόκληρα κοινωνικά σύνολα.

Όπως βλέπουμε, ο ορισμός ενός δικτύου είναι ιδιαίτερα γενικός, κάτι που έχει ως αποτέλεσμα να μπορούμε να βρούμε παραδείγματα δικτύων σε μία πληθώρα τομέων. Ως εκ τούτου, η ανάλυση δικτύων είναι ένα πολύ σημαντικό εργαλείο μπορεί να εφαρμοστεί σε όλα αυτά τα δίκτυα για να εξάγουμε πληροφορίες. [4]



Σχήμα 2.1: *Απεικόνιση ενός δικτύου από [1].*

2.2 Σύνθετα δίκτυα

2.2.1 Τι είναι ένα σύνθετο δίκτυο

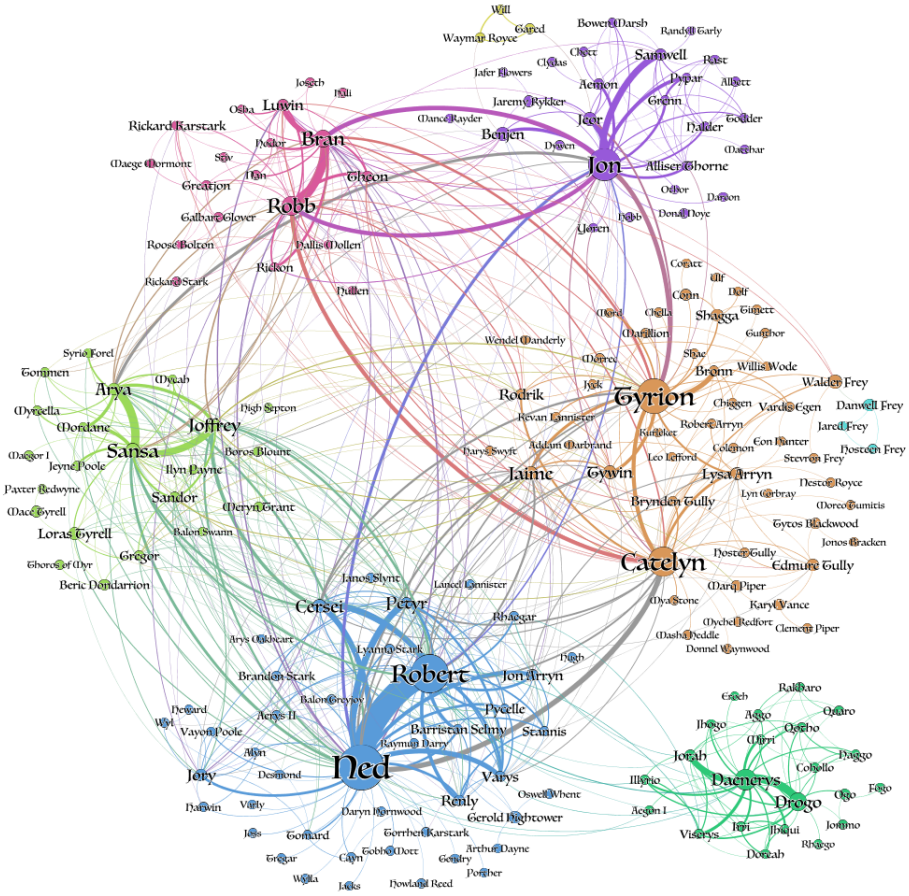
Κατ' αρχάς, μπορούμε να ορίσουμε ένα σύνθετο δίκτυο ως ένα σύνολο από συνδεδεμένα στοιχεία που μπορούν να αλληλεπιδράσουν μεταξύ τους. Η διαφορά του από τα κλασικά δίκτυα, έγκειται στο γεγονός ότι η συμπεριφορά του συστήματος ενός σύνθετου δικτύου, δεν μπορεί να κατανοηθεί πλήρως μέσω της μελέτης του κάθε μεμονομένου στοιχείου του [15]. Δηλαδή, αυτό που μας ενδιαφέρει όταν αναλύουμε ένα σύνθετο δίκτυο είναι οι αλληλεπιδράσεις μεταξύ των κόμβων του δικτύου και τον τρόπο με τον οποίο οι αλληλεπιδράσεις αυτές επηρεάζουν το δίκτυο σαν σύνολο. Έτσι, μπορούμε να πάρομε δεδομένα για το δίκτυο και πιθανώς να προβλέψουμε την εξέλιξη του.

Ένα ακόμη χαρακτηριστικό που διαφοροποιεί τα σύνθετα από τα κλασικά δίκτυα, είναι το γεγονός ότι τα σύνθετα δίκτυα έχουν χαρακτηριστικά που συνήθως προκύπτουν σε πραγματικά δίκτυα. Δηλαδή τα σύνθετα δίκτυα έχουν μη τετριμμένη τοπολογική δομή, σε αντίθεση με τα κλασσικά δίκτυα που μπορεί να έχουν τη μορφή δέντρου η πλέγματος [16].

2.2.2 Είδη Σύνθετων Δικτύων

Όπως αναφέρθηκε προηγουμένως, τα σύνθετα δίκτυα προσομοιάζουν δίκτυα που προκύπτουν στη φύση. Επομένως, είναι αναμενόμενο να υπάρχει μεγάλο πλήθος δικτύων που μπορούν να χαρακτηριστούν ως σύνθετα δίκτυα. Κάποια χαρακτηριστικά παραδείγματα αποτελούν τα εξής:

- **Κοινωνικά Δίκτυα (Social Networks)**. Ένα κοινωνικό δίκτυο ορίζεται ως μία κοινω-



Σχήμα 2.2: *Κοινωνικό δίκτυο που σχηματίστηκε μεταξύ χαρακτήρων του Game of Thrones, από [2].*

νική δομή που σχηματίζεται μεταξύ είτε ατόμων ή συνόλων και απεικονίζει τον τρόπο με τον οποίο οι κόμβοι αυτοί συνδέονται. Οι σύνδεσμοι αυτοί χαρακτηρίζονται από την ισχύ του δεσμού μεταξύ των κόμβων, από την ευστάθεια της σχέσης αυτής και από την αμοιβαιότητα της σχέσης. [17]

- **Οικονομικά Δίκτυα (Financial Networks)**. Οικονομικά δίκτυα είναι τα δίκτυα που αποτελούνται από κόμβους που αντιπροσωπεύουν οικονομικές οντότητες και οι ζεύξεις με τις οποίες είναι συνδεδεμένα συμβολίζουν τις συναλλαγές που γίνονται μεταξύ τους [18]. Οικονομικές οντότητες μπορεί να είναι εταιρίες, τράπεζες ή και άτομα που συμμετέχουν σε οικονομικές συναλλαγές.

- **Βιολογικά Δίκτυα (Biological Networks)**. Στην περίπτωση των βιολογικών δικτύων, οι ζεύξεις αναπαριστούν τις βιολογικές αλληλεπιδράσεις που υπάρχουν μεταξύ βιολογικών οντοτήτων. Οι οντότητες αυτοί μπορεί να είναι από πρωτεΐνες και γονίδια έως και νευρώνες.

2.3 Κοινωνικά Δίκτυα

Από τα είδη σύνθετων δικτύων που περιγράφηκαν στην προηγούμενη ενότητα, κρίνεται αναγκαία η εστίαση στα κοινωνικά δίκτυα. Ο λόγος που εστιάζουμε εκεί, είναι διότι οι ιδιότητες και τα χαρακτηριστικά'που διέπουν το συγκεκριμένο είδος δικτύων μπορούν να εφαρμοστούν σε διάφορους άλλους τομείς, και οι μετρικές που χρησιμοποιούνται στην ανάλυση κοινωνικών δικτύων έχουν τη δυνατότητα να προσαρμοστούν ώστε να μελετήσουμε και άλλα είδη δικτύων.

2.3.1 Παραδείγματα χρήσης της Ανάλυσης Κοινωνικών Δικτύων

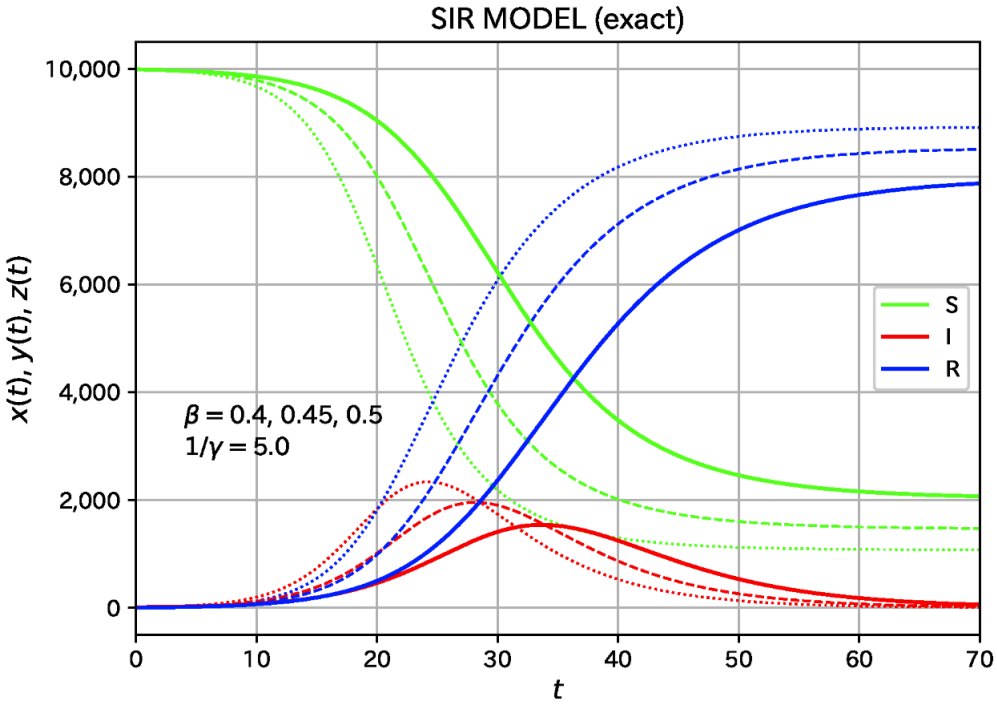
Αρχικά, η ανάλυση κοινωνικών δικτύων πρόκειται για ένα ιδιαίτερα σημαντικό αντικείμενο έρευνας, καθώς μέσω αυτής μπορούμε να λάβουμε δεδομένα για την καλύτερη κατανόηση ζητημάτων που αφορούν τομείς όπως η ανθρωπολογία, η κοινωνιολογία, η πολιτική επιστήμη και η ψυχολογία. Στην ψυχολογία, λόγου χάριν, γίνεται χρήση της ανάλυσης κοινωνικών δικτύων ώστε να μπορέσουμε να εξηγήσουμε τις πράξεις, σκέψεις και τα συναισθήματα ενός ανθρώπου λαμβάνοντας υπόψην το κοινωνικό πλαίσιο στο οποίο βρίσκεται. Ο τρόπος με τον οποίο εξάγονται οι πληροφορίες, είναι μέσα από την μελέτη των διαπροσωπικών σχέσεων που σχηματίζονται μεταξύ ατόμων και των χαρακτηριστικών που διέπουν αυτές τις σχέσεις. [19].

Ακόμη ένα παράδειγμα που υποδεικνύει τη σημασία που έγκειται στη μελέτη των κοινωνικών δικτύων είναι η συμβολή της ανάλυσης δικτύων σε τομείς όπως η ιατρική. Αναλυτικότερα, μέσα από τη μελέτη κοινωνικών δικτύων έχουν υπάρξει σημαντικές πρόοδοι στον τομέα της επιδημιολογίας. Μία επιδημία δεν εξαρτάται μόνο από τη μεταδοτικότητα της ασθένειας, αλλά και στις κοινωνικές δομές που σχηματίζονται στον πληθυσμό τον οποίο επηρεάζει. Ως εκ τούτου, μπορούμε να εφαρμόσουμε τεχνικές ανάλυσης κοινωνικών δικτύων, προκειμένου να μπορέσουμε να προβλέψουμε και να καταλάβουμε καλύτερα τον τρόπο με τον οποίο η ασθένεια μεταδίδεται και εξαπλώνεται. Αυτό γίνεται δημιουργώντας ένα ¨δίκτυο επαφών¨, στο οποίο οι κόμβοι αντιπροσωπούν άτομα και οι ζεύξεις μεταξύ κόμβων συμβολίζουν επαφή των δύο ατόμων με τρόπο που επιτρέπει την εξάπλωση της ασθένειας [4]. Με αυτές τις πληροφορίες, έχουμε τη δυνατότητα να πάρουμε τα απαραίτητα μέτρα ώστε να μπορέσουμε να επιβραδύνουμε τον ρυθμό εξάπλωσης και να ελαχιστοποιήσουμε τις απώλειες που προκύπτουν σε μία επιδημία.

Μπορούμε, λοιπόν, να δούμε ότι η μελέτη και ανάλυση κοινωνικών δικτύων συμβάλλει σε πολλούς διαφορετικούς τομείς, και συμβάλλει στην καταπολέμηση καίριων προβλημάτων.

2.4 Ανάλυση Κοινωνικών Δικτύων

Για να μπορέσουμε να κάνουμε ανάλυση ενός κοινωνικού δικτύου, είναι κατ' αρχάς αναγκαίο να γίνει ορισμός συγκεκριμένων ιδιοτήτων που διέπουν τα δίκτυα αυτά και των μετρικών που χρησιμοποιούνται. Οι μετρικές αυτές περιγράφουν τόσο την τοπολογία όσο και τα χαρακτηριστικά του δικτύου και αποσκοπούν στην εξαγωγή πληροφοριών από αυτά.



Σχήμα 2.3: *Διάγραμμα ενός επιδημιολογικού μοντέλου από [3].*

2.4.1 Χαρακτηριστικά Διασύνδεσης

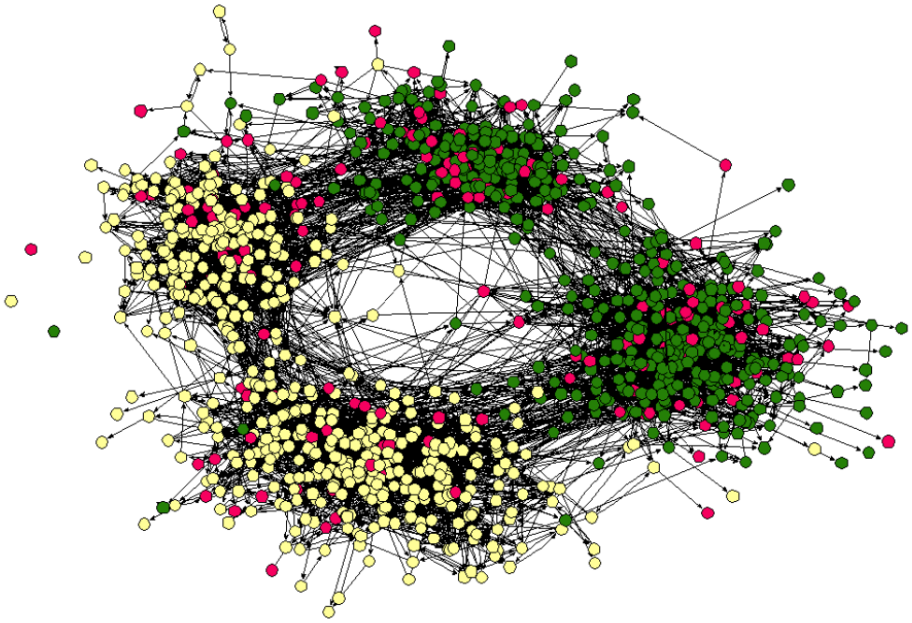
Για αρχή, θα παρουσιαστούν μερικές σημαντικές ιδιότητες που αφορούν τις συνδέσεις που εμφανίζονται μεταξύ κόμβων ενός δικτύου.

- **Ομοφιλία (Homophily)**. Η ομοφιλία είναι μία ιδιότητα των δικτύων, η οποία συμβολίζει την τάση των κόμβων σε ένα δίκτυο να δημιουργούν διασυνδέσεις κυρίως με κόμβους με παρόμοια χαρακτηριστικά, παρά με κόμβους με τους οποίους δεν υπάρχει μεγάλη ομοιότητα. Μπορούμε έτσι, μελετώντας τις ζεύξεις μεταξύ κόμβων να τις χαρακτηρίσουμε ως ¨ετερογενείς¨ στην περίπτωση που συνδέουν δύο κόμβους που, με βάση τα κριτήρια που έχουμε ορίσει, δεν είναι όμοιοι ή ¨ομογενείς¨ στην αντίθετη περίπτωση [4].

- **Αμοιβαιότητα (Reciprocity)**. Η αμοιβαιότητα σε ένα δίκτυο έχει νόημα να οριστεί μόνο στην περίπτωση οι διασυνδέσεις του δικτύου είναι κατευθυνόμενες. Με την προϋπόθεση αυτή, μία σχέση μεταξύ δύο κόμβων μπορεί να είναι είτε μονόδρομη ή αμφίδρομη (δηλαδή αμοιβαία). Αμοιβαιότητα, λοιπόν, είναι ο αριθμός των αμφίδρομων σχέσεων που υπάρχουν σε ένα δίκτυο ως πρός τον συνολικό αριθμό σχέσεων του δικτύου.

- **Τριαδική Κλειστότητα (Triadic Closure)**. Η τριαδική κλειστότητα είναι η ιδιότητα που περιγράφει την τάση των κόμβων ενός δικτύου να δημιουργούν κοινότητες (ή συστάδες). Για παράδειγμα, έστω ότι σε ένα κοινωνικό δίκτυο οι κόμβοι αντιπροσωπεύουν άτομα. Αν δύο άτομα Α και Β έχουν έναν κοινό φίλο Γ, τότε η πιθανότητα να σχηματιστεί στο μέλλον φιλία μεταξύ των Α και Β είναι αυξημένη. Ο λόγος που ονομάζεται,

αυτή η ιδιότητα, τριαδική κλειστότητα είναι διότι η δημιουργία της ζεύξης μεταξύ των Α και Β που αναφέρθηκαν στο παράδειγμα, ¨κλείνει¨ το τρίγωνο που σχηματίζεται από τους Α, Β και Γ [4].



Σχήμα 2.4: *Χωρισμός ενός δικτύου σε στενά διασυνδεδεμένες, ομογενοποιημένες κοινότητες λόγω της ιδιότητας της ομοφιλίας, από [4].*

2.4.2 Μετρικές Δικτύου

Όπως αναφέρθηκε και προηγουμένως, οι μετρικές ενός δικτύου είναι ο τρόπος που έχουμε προκειμένου να περιγράψουμε ένα δίκτυο και να εξάγουμε δεδομένα από αυτό, ώστε να μπορέσουμε να το αναλύσουμε. Στη συνέχεια θα αναφερθούν κάποιες από τις σημαντικότερες μετρικές, και σε επόμενη ενότητα θα γίνει πιο λεπτομερής ανάλυση συγκεκριμένων μετρικών.

Κατάρχάς, για την καλύτερη κατανόηση των μετρικών, πρέπει πρώτα να σημειωθεί ο ορισμός της έννοιας της απόστασης (distance). Ως απόσταση μεταξύ δύο κόμβων ενός δικτύου, ορίζουμε τον ελάχιστο αριθμό των ακμών (διασυνδέσεων) που χρειάζεται, ώστε να δημιουργηθεί ένα μονοπάτι από τον ένα κόμβο στον άλλο. Αυτό το ελάχιστο μονοπάτι, ονομάζεται επισήμως γεωδεσική απόσταση ή γεωδεσική του δικτύου.

Μερικές από τις πιο βασικές μετρικές που χρησιμοποιούνται, λοιπόν, είναι οι εξής:

- **Εκκεντρότητα (Eccentricity)**. Η εκκεντρότητα ενός κόμβου ορίζεται ως η μέγιστη απόσταση του κόμβου αυτού ως πρός οποιονδήποτε άλλο κόμβο στο δίκτυο. Η μετρική αυτή μπορεί να χρησιμοποιηθεί για να βρούμε το πόσο σημαντικός είναι ένας κόμβος στο δίκτυο [20], και μέσα από αυτή, μπορούμε να ορίσουμε τις έννοιες της ακτίνας και της διαμέτρου του δικτύου, που αντιπροσωπεύουν την ελάχιστη και τη μέγιστη εκκεντρότητα του δικτύου, αντίστοιχα.

- **Κεντρικότητα (Centrality)**. Η μετρική της κεντρικότητας, αποσκοπεί στην εύρεση των πιο σημαντικών κόμβων ή των κόμβων που έχουν τη μεγαλύτερη επιρροή σε ένα δίκτυο. Υπάρχουν διάφορα είδη κεντρικότητας και κάθε ένα από αυτά ορίζει με διαφορετικό τρόπο το τι θεωρείται ως ¨σημαντικός κόμβος¨. Οι κεντρικότητες και οι υποκατηγορίες τους θα μελετηθούν λεπτομερώς σε επόμενη ενότητα.

- **Αρθρωτότητα (Modularity)**. Η αρθρωτότητα είναι η μετρική η οποία μας δείχνει τον βαθμό με τον οποίο οι συστάδες (ή κοινότητες) ενός δικτύου, δηλαδή στενά συνδεδεμένα σύνολα κόμβων, είναι αραιά συνδεδεμένες μεταξύ τους. Αναλυτικότερα, σε ένα δίκτυο που έχει υψηλή αρθρωτότητα, υπάρχουν στενά συνδεδεμένες γειτονιές κόμβων που σχηματίζουν κοινότητες, αλλά υπάρχουν ελάχιστες συνδέσεις μεταξύ κόμβων που ανήκουν σε διαφορετικές συστάδες [21].

- **Συντελεστής Συσταδοποίησης (Clustering Coefficient)**. Αρχικά, ο συντελεστής συσταδοποίησης είναι η μετρική που μας δείχνει πόσο ισχυρή είναι η τάση να σχηματιστούν συστάδες σε ένα δίκτυο. Υπάρχουν δύο μορφές για τον συντελεστή συσταδοποίησης, μία τοπική που αναφέρεται σε έναν κόμβο και τη ¨γειτονιά¨ του, και μία ολική που περιγράφει το δίκτυο σαν σύνολο. Υψηλός συντελεστής συσταδοποίησης σημαίνει ότι ο γράφος μας χαρακτηρίζεται από υψηλή τριαδική κλειστότητα.

2.5 Κεντρικότητα

Η κεντρικότητα, είναι ένα από τα πιο γνωστά και ισχυρά εργαλεία που έχουμε στη διάθεσή μας για την ανάλυση ενός δικτύου. Η μετρική αυτή δημιουργήθηκε για την ανάλυση κοινωνικών δικτύων, επομένως αρκετοί όροι που αφορούν κεντρικότητες έχουν κοινωνιολογική προέλευση. Θα ορίσουμε τις κεντρικότητες με την παραδοχή ότι οι διασυνδέσεις που υπάρχουν στο δίκτυο είναι αμφίδρομες.

2.5.1 Κεντρικότητα Βαθμού (Degree)

Αρχικά, ξεκινάμε από την πιο απλοϊκή μορφή κεντρικότητας, που ονομάζεται κεντρικότητα βαθμού. Η κεντρικότητα βαθμού ορίζει τη σημασία ενός κόμβου, με βάση τον αριθμό των ζεύξεων που έχει ο κόμβος αυτός. Αναλυτικότερα, οι πληροφορίες που παίρουμε για έναν κόμβο Α, με τη χρήση της κεντρικότητας βαθμού, είναι ο αριθμός των γειτονικών κόμβων του στους οποίους μπορεί να φτάσει με ένα μόνο βήμα, δηλαδή έχει απόσταση ίση με 1 [22].

Η χρήση αυτού του είδους κεντρικότητας μας δείχνει πόσο ¨δημοφιλής¨ είναι ένας κόμβος. Πιο συγκεκριμένα, σε ένα κοινωνικό δίκτυο, όσο μεγαλύτερη κεντρικότητα βαθμού έχει ένας κόμβος, τόσο περισσότεροι κόμβοι είναι συνδεδεμένοι με αυτόν, κάτι που σημαίνει ότι μπορεί να περιέχει περισσότερες πληροφορίες.

Αν θεωρήσουμε ένα δίκτυο n κόμβων, έναν κόμβο k και τον πίνακα γειτνίασης a_{ik}, τότε η κεντρικότητα βαθμού υπολογίζεται ως

$$C_D(k) = \sum_{i=1}^{n} a_{ik}$$

και η κανονικοποιημένη κεντρικότητα βαθμού ως

$$C_D^{'}(k) = \sum_{i=1}^{n} \frac{a_{ik}}{n-1}$$

2.5.2 Κεντρικότητα Εγγύτητας (Closeness)

Η κεντρικότητα εγγύτητας χαρακτηρίζει έναν κόμβο ως ¨σημαντικό¨ με βάση την απόσταση του κόμβου από τους υπόλοιπους κόμβους του δικτύου (δηλαδή την εγγύτητα του σε αυτούς).

Ο λόγος για τον οποίο χρησιμοποιούμε αυτό το είδος κεντρικότητας είναι διότι μας παρέχει μία ένδειξη για το ποιοί κόμβοι μπορούν να επηρεάσουν ευκολότερα ολόκληρο το δίκτυο [22]. Όσο μεγαλύτερη είναι η κεντρικότητα εγγύτητας ενός κόμβου Α, τόσο πιο κοντά βρίσκεται κατά μέσο όρο στους κόμβους του δικτύου, οπότε η πληροφορίες που διαδίδει ο κόμβος Α φτάνουν σχετικά γρήγορα στους υπόλοιπους.

Για έναν κόμβο Α, λοιπόν, ο τρόπος που βρίσκεται η κεντρικότητα εγγύτητας του είναι με τον υπολογισμό των ελάχιστων μονοπατιών που υπάρχουν από τον Α πρός όλους τους άλλους κόμβους του δικτύου και στη συνέχεια αθροίζοντας τα μήκη των μονοπατιών αυτών. Διαιρώντας τον αριθμό των κόμβων του δικτύου που μελετάμε (χωρίς να λαμβάνουμε υπ'όψην τον κόμβο Α) με το άθροισμα των ελάχιστων μονοπατιών από τον Α στο υπόλοιπο δίκτυο, παίρνουμε την κεντρικότητα εγγύτητας για τον κόμβο Α. Ο λόγος που στον αριθμητή του κλάσματος βρίσκονται οι κόμβοι του δικτύου, είναι ώστε να πάρουμε μία κανονικοποιημένη έκφραση για την κεντρικότητα, κάτι που βοηθάει στη σύγκριση κεντρικότητας μεταξύ κόμβων δικτύων διαφορετικού μεγέθους.

Για παράδειγμα, θεωρούμε ένα δίκτυο n κόμβων, έναν κόμβο k και ορίζουμε την απόσταση του κόμβου k από οποιονδήποτε κόμβο i του δικτύου ως $d(i,k)$. Τότε, η κεντρικότητα εγγύτητας υπολογίζεται με τον ακόλουθο τύπο:

$$C_P(k) = \frac{n-1}{\sum_{i=1}^{n} d(i,k)}$$

2.5.3 Κεντρικότητα Διαμεσικότητας (Betweenness)

Ο τρόπος με τον οποίο ένας κόμβος ορίζεται ως σημαντικός, σύμφωνα με τη μετρική της κεντρικότητας διαμεσικότητας, εξαρτάται από τον αριθμό των ελάχιστον μονοπατιών, από ένα τυχαίο κόμβο του δικτύου σε έναν άλλο, στα οποία ανήκει ο κόμβος που μελετάμε.

Η μετρική αυτή, μας δίνει υποδεικνύει ποιοί κόμβοι του δικτύου λειτουργούν σαν ¨γέφυρες¨ μεταξύ κόμβων [22]. Άν ένας κόμβος Α έχει υψηλή διαμεσικότητα, αυτό σημαίνει πως ανήκει σε πολλά ελάχιστα μονοπάτια και κατά συνέπεια, συνδέει διαφορετικές κοινότητες του γράφου μεταξύ τους. Ως εκ τούτου ο κόμβος Α επηρεάζει τη ροή πληροφορίας στο δίκτυο και η αφαίρεση κόμβων με υψηλή διαμεσικότητα μπορεί να κάνει το γράφο μη συνεκτικό.

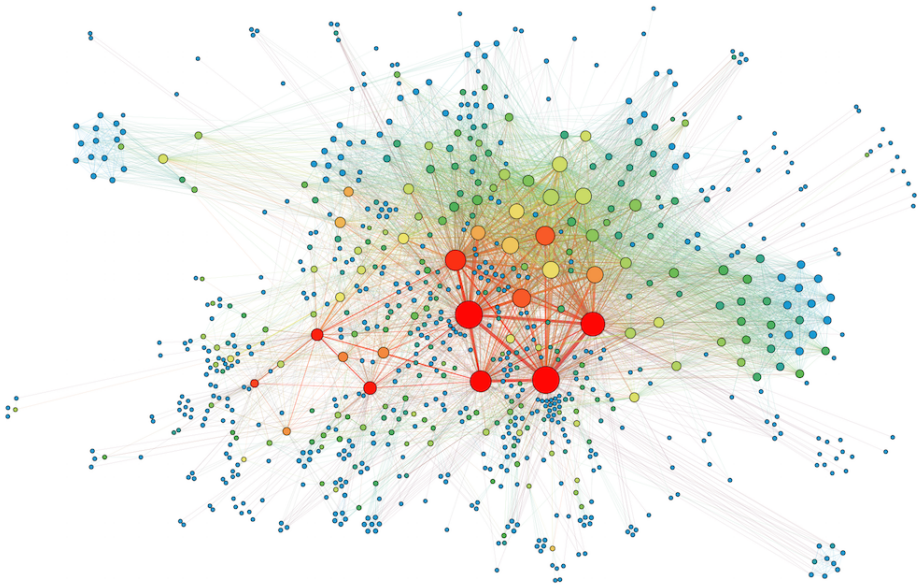
Ο τρόπος με τον οποίο γίνεται ο υπολογισμός της κεντρικότητας διαμεσικότητας, είναι αρχικά υπολογίζοντας όλα τα ελάχιστα μονοπάτια μεταξύ οποιωνδήποτε τυχαίων δύο κόμβων που υπάρχουν στο δίκτυο. Στη συνέχεια μετράμε σε πόσα από αυτά ανήκει ο κόμβος που μας ενδιαφέρει και τα διαιρούμε με το σύνολο των ελάχιστον μονοπατιών. Προφανώς, στην

περίπτωση που υπάρχει μόνο ένα ελάχιστο μονοπάτι μεταξύ των δύο τυχαίων κόμβων απλά ελέγχουμε αν ανήκει ο κόμβος που μας ενδιαφέρει σε αυτό. Επαναλαμβάνουμε αυτή τη διαδικασία για κάθε ζευγάρι κόμβων στο δίκτυο.

Έστω, για παράδειγμα, δίκτυο n κόμβων, έναν κόμβο k και δύο τυχαίους κόμβους i, j, με την προϋπόθεση ότι $i \neq j$. Αν συμβολίσουμε τα ελάχιστα μονοπάτια μεταξύ i και j στα οποία ανήκει ο k ως $s^k_{(i,j)}$ και τα ελάχιστα μονοπάτια στα οποία δεν ανήκει ως $s_{(i,j)}$, τότε η κεντρικότητα διαμεσικότητας υπολογίζεται ως εξής:

$$C_B(k) = \sum_{i \neq j} \frac{s^k_{(i,j)}}{(s_{(i,j)}}$$

Όπως μπορούμε να παρατηρήσουμε, ο υπολογισμός της κεντρικότητας διαμεσικότητας είναι εξαιρετικά χρονοβόρος. Η ανάγκη υπολογισμού όλων των ελάχιστων μονοπατιών μεταξύ κάθε δύο κόμβων του δικτύου προκειμένου να μπορέσουμε να υπολογίσουμε τη διαμεσικότητα ενός μόνο κόμβου καθιστά αυτή τη μετρική δύσχρηστη σε περιπτώσεις που θέλουμε πολύ γρήγορα αποτελέσματα. Ένδειξη αυτού είναι η χρονική πολυπλοκότητα, που έχει ο πιο αποδοτικός αλγόριθμος που χρησιμοποιείται για τον υπολογισμό της, η οποία είναι ίση με $O(nm + n^2 logn)$ [23].



Σχήμα 2.5: *Γράφημα δικτύου. Το μέγεθος των κόμβων είναι συνάρτηση της κεντρικότητας βαθμού τους, ενώ το χρώμα τους είναι συνάρτηση της κεντρικότητας διαμεσικότητάς τους, από [5].*

2.5.4 Κεντρικότητα Katz

Η κεντρικότητα Katz, σε αντίθεση με τις πιο τυπικές μετρικές κεντρικότητας, δεν χρησιμοποιεί μόνο τη γεωδεσική μεταξύ δύο κόμβων για να ορίσει την έννοια της σημαντικότητας. Αντί αυτού, υπολογίζει στον σχετικό βαθμό επιρροής ενός κόμβου στο δίκτυο το οποίο μελετάμε [24]. Αξίζει να σημειωθεί, πως η κεντρικότητα Katz είναι παρόμοια με την κεντρικότητα

ιδιοδιανύσματος (Eigenvector centrality) και με τον αλγόριθμο Pagerank που χρησιμοποιείται από την Google.

Πιο συγκεκριμένα, ο τρόπος με τον οποίο υπολογίζουμε την κεντρικότητα Katz ενός κόμβου Α είναι, αρχικά, υπολογίζοντας τον αριθμό των γειτόνων πρώτου βαθμού του (δηλαδή των κόμβων που απέχουν από τον Α απόσταση ίση με 1) και πολλαπλασιάζοντας αυτό τον αριθμό επί a. Το a είναι μία μεταβλητή που ονομάζεται ¨παράγοντας απόσβεσης¨. Στη συνέχεια, βρίσκουμε όλους τους κόμβους του δικτύου που συνδέονται με τον Α μέσω των γειτόνων πρώτου βαθμού του και τους πολλαπλασιάζουμε με a^d, όπου d είναι η απόσταση τους από τον Α [24].

2.6 Συντελεστής Συσταδοποίησης

Όπως αναφέρθηκε και σε προηγούμενη ενότητα, ο συντελεστής συσταδοποίησης συμβολίζει την τάση κόμβων σε ένα γράφο να σχηματίζουν συστάδες. Για την κατανόηση αυτής της μετρικής, χρειάζεται ο ορισμός της έννοιας της κλίκας. Μία κλίκα σε ένα δίκτυο είναι ο μέγιστος αριθμός κόμβων. οι οποίοι είναι συνδεδεμένοι μεταξύ τους με κάθε δυνατό τρόπο.

Ο συντελεστής συσταδοποίησης μπορεί να υπολογιστεί με δύο τρόπους - τοπικά και ολικά (δηλαδή σε όλο το γράφο) [25].

2.6.1 Τοπικός Συντελεστής Συσταδοποίησης (Local)

Ο τοπικός συντελεστής συσταδοποίησης αναφέρεται σε κάποιον κόμβο του γράφου και περιγράφει το πόσο κοντά βρίσκεται η γειτονιά του στον σχηματισμό μίας κλίκας. Αυτό σημαίνει ότι ορίζουμε τον τοπικό συντελεστή συσταδοποίησης για έναν κόμβο A που βρίσκεται σε μία γειτονιά N ως τον αριθμό των ζεύξεων που υπάρχουν στη γειτονιά ως προς τον αριθμό των ζεύξεων που θα μπορούσαν να υπάρχουν.

Για τον υπολογισμό αυτού του λόγου, πρέπει να σημειωθεί ότι αν k_i είναι ο αριθμός των κόμβων σε μία γειτονιά N_i, για έναν μή κατευθυνόμενο γράφο, δηλαδή έναν γράφο στον οποίο οι ζεύξεις μεταξύ κόμβων είναι αμφίδρομες, οι συνολικές ζεύξεις που μπορούν να υπάρξουν στη γειτονιά N_i είναι ίσες με $\frac{k_i(k_i-1)}{2}$ [25].

2.6.2 Ολικός συντελεστής συσταδοποίησης (Global)

Αρχικά, ορίζουμε την έννοια της τριπλέτας (triplet) ή τριάδας (triad), ως τρεις κόμβους οι οποίοι συνδέονται από 2 ακμές (ανοιχτή τριπλέτα) ή από 3 ακμές (κλειστή τριπλέτα).

Ο ολικός συντελεστής συσταδοποίησης, ή αλλιώς μεταβατικότητα (transitivity), είναι το μέτρο της τριαδικής κλειστότητας που υπάρχει σε ένα γράφο, δηλαδή μετράει τον αριθμό και το είδος των τριπλετών που υπάρχουν. Ένα τρίγωνο μεταξύ τριών ενωμένων κόμβων περιλαμβάνει τρείς κλειστές τριπλέτες, μία για κάθε κόμβο που ανήκει στο τρίγωνο. Ο ολικός συντελεστής συσταδοποίησης, λοιπόν, ορίζεται ως ο αριθμός κλειστών τριπλετών (ή $3\#triangles$ ως πρός τον συνολικό αριθμό τριπλέτων στον γράφο (είτε ανοιχτές ή κλειστές).

Ως εκ τούτου, η μαθηματική έκφραση είναι [26]:

$$T = 3\frac{\#triangles}{\#triplets} = \frac{\#closed\ triplets}{\#triplets}$$

2.7 Τοπολογίες Δικτύων

Η τοπολογία ενός δικτύου είναι ο τρόπος με τον οποίο είναι δομημένο το δίκτυο, δηλαδή η διάταξη των κόμβων και των ζεύξεων που υπάρχουν. Όμως, η μελέτη τοπολογιών σε πραγματικά δίκτυα πρόκειται για ένα πολύ πολύπλοκο ζήτημα, καθώς τα δίκτυα αυτά είναι τεράστια σε όγκο. Ως εκ τούτου, προέκυψε η ανάγκη δημιουργίας συνθετικών μοντέλων που προσομοιάζουν τα πραγματικά δίκτυα ώστε να μπορέσουμε να τα μελετήσουμε καλύτερα σε ελεγχόμενα περιβάλλοντα [27].

2.7.1 Χαρακτηριστικά πραγματικών δικτύων

Για αρχή, θα παρουσιαστούν κάποια χαρακτηριστικά που προκύπτουν σε πολλά, αλλά όχι όλα, πραγματικά δίκτυα. Τα χαρακτηριστικά αυτά είναι:

- **Ιδιότητα μικρού κόσμου (Small-World)**. Ένας γράφος έχει την ιδιότητα μικρού κόσμου, αν χαρακτηρίζεται από σχετικά υψηλό συντελεστή συσταδοποίησης και μικρό μέσο μήκος μονοπατιού [28].

- **Ιδιότητα Scale-Free**. Έχει παρατηρηθεί, ότι σε πολλά πραγματικά δίκτυα, η κατανομή βαθμού που παρουσιάζουν είναι power-law. Αυτό σημαίνει ότι το ποσοστό $P(k)$ των κόμβων του δικτύου που έχουν k διασυνδέσεις, δηλαδή βαθμό k, για μεγάλες τιμές του k ακολουθεί τη σχέση $P(k) = ck^{-\gamma}$. Το γ είναι μία παράμετρος του οποίου οι τιμές συνήθως είναι μεταξύ του 2 και του 3 [29].

- **Δυναμική Ανάπτυξη (Growth)**. Ένα πραγματικό δίκτυο συνήθως αναπτύσσεται, δηλαδή συνεχώς εισέρχονται νέοι κόμβοι στο δίκτυο, οι οποίοι σχηματίζουν καινούριες διασυνδέσεις με τους προϋπάρχοντες κόμβους του γράφου.

2.7.2 Συνθετικές τοπολογίες

Παραπάνω αναφέρθηκε ότι για την καλύτερη μελέτη των πραγματικών δικτύων κάνουμε χρήση συνθετικών τοπολογιών. Τα πιο βασικά μοντέλα παραγωγής σύνθετων τοπολογιών που χρησιμοποιούνται στην ανάλυση δικτύων είναι τα εξής:

Οι πιο βασικές συνθετικές τοπολογίες (και μοντέλα παραγωγής συνθετικών τοπολογιών) που χρησιμοποιούνται στην ανάλυση δικτύων είναι οι εξής:

- **Πλέγμα (Lattice)**. Ένα δίκτυο πλέγμα (ή αλλιώς κανονικός γράφος) είναι το δίκτυο στο οποίο κάθε κόμβος συνδέεται με τους k κοντινότερους κόμβους του, δηλαδή η κεντρικότητα βαθμού του κάθε κόμβου είναι ίση με k. Το μέσο μήκος μονοπατιού στους γράφους αυτούς είναι σταθερό και αρκετά μεγαλο. Υπάρχει, επίσης, σχετικά υψηλός συντελεστής συσταδοποίησης και η κατανομή βαθμού είναι μία συνάρτηση Dirac. Προφανώς, τα πραγματικά δίκτυα έχουν μη τετριμμένη δομή και χαρακτηριστικά, κάτι που δεν ισχύει για αυτό το είδος τοπολογίας.

- **Μοντέλο Erdos-Renyi**. Το μοντέλο Erdos-Renyi παράγει τυχαίους γράφους (RGER) που σχηματίζονται ορίζοντας τον αριθμό κόμβων n και ακμών e. Στη συνέχεια ο γράφος

επιλέγεται ομοιόμορφα από μία συλλογή γράφων που έχουν αυτά τα χαρακτηριστικά. Αυτή η τοπολογία λόγω της τυχαίας δομής της, έχει μικρό συντελεστή συσταδοποίησης και η κατανομή βαθμού της είναι κατανομή Poisson. Αυτά τα χαρακτηριστικά, σε συνδυασμό με το γεγονός ότι μία ακμή προκύπτει με τυχαίο τρόπο, σημαίνει επίσης ότι τέτοιες τοπολογίες δεν συναντάμε σε πραγματικά δίκτυα.

- **Τυχαίοι Γεωμετρικοί Γράφοι (RGG)**. Οι τυχαίοι γεωμετρικοί γράφοι είναι η πιο απλή περίπτωση χωρικού δικτύου (spatial network). Αυτό σημαίνει ότι οι κόμβοι του δικτύου υπάρχουν στο χώρο και μπορει κανείς να υπολογίσει τη χωρική απόσταση (συνήθως γεωγραφική απόσταση) μεταξύ δύο κόμβων [30]. Για τη δημιουργία ενός τυχαίου γεωμετρικού γράφου ορίζουμε τη μεταβλητή *radius*. Στη συνέχεια τοποθετούμε τους κόμβους του δικτύου σε τυχαία σημεία στο χώρο με ομοιόμορφο τρόπο και έπειτα τοποθετούμε ζεύξεις μεταξύ δύο κόμβων υπό την προϋπόθεση ότι η (ευκλείδεια) απόσταση μεταξύ τους είναι μικρότερη ή ίση από *radius* [31]. Κάποια χαρακτηριστικά των τυχαίων γεωμετρικών γράφων είναι ομοιόμορφη κατανομή βαθμού και το μεγάλο μήκος μονοπατιού, που προκύπτει λόγω της δημιουργίας ζεύξης ανάλογα με την απόσταση. Τα χαρακτηριστικά αυτά σημαίνουν ότι τα RGG δεν έχουν ιδιότητες ούτε μικρού κόσμου ούτε scale-free. Οι τυχαίοι γεωμετρικοί γράφοι έχουν εφαρμογές σε συστήματα που οι ζεύξεις τους εξαρτώνται από χωρικούς περιορισμούς, όπως για παράδειγμα τα ασύρματα ad-hoc δίκτυα, τα οποία θα αναλυθούν σε επόμενη ενότητα.

- **Μοντέλο Watts - Strogatz**. Το μοντέλο Watts - Strogatz χρησιμοποιείται για την παραγωγή τοπολογιών με ιδιότητες μικρού κόσμου. Αναλυτικότερα, για να κατασκευάσουμε ένα τέτοιο δίκτυο, ξεκινάμε από έναν κανονικό γράφο βαθμού k. Στη συνέχεια, ορίζουμε πιθανότητα p για επανασύνδεση (rewiring) οποιασδήποτε ακμής που συνδέει κάποιον κόμβο i με έναν κόμβο j, ώστε πλέον να συνδέει τον κόμβο i με έναν άλλο τυχαίο κόμβο z, με την προυπόθεση ότι $i \neq z$ και $j \neq z$. Με τον τρόπο αυτό, καταλήγουμε σε ένα δίκτυο το οποίο έχει αρκετά υψηλό συντελεστή συσταδοποίησης, κάτι που προκύπτει επειδή το δίκτυο αυτό ξεκινάει ως πλέγμα, και σχετικά μικρό μήκος μέσου μονοπατιού, εξαιτίας της πιθανότητας επανασύνδεσης που δημιουργεί ¨γέφυρες¨ μεταξύ μακρινών γειτονιών κόμβων. Παρόλα αυτά, η κατανομή βαθμού της τοπολογίας δεν είναι power-law και ο τρόπος δημιουργίας αυτού του γράφου γίνεται με συγκεκριμένο αριθμό κόμβων, κάτι που δεν ισχύει συνήθως στα πραγματικά δίκτυα στα οποία ο αριθμός των κόμβων αυξάνεται συνεχώς [27]. Παρόλα αυτά αποτελεί ένα μοντέλο που προσεγγίζει αρκετά πραγματικά δίκτυα.

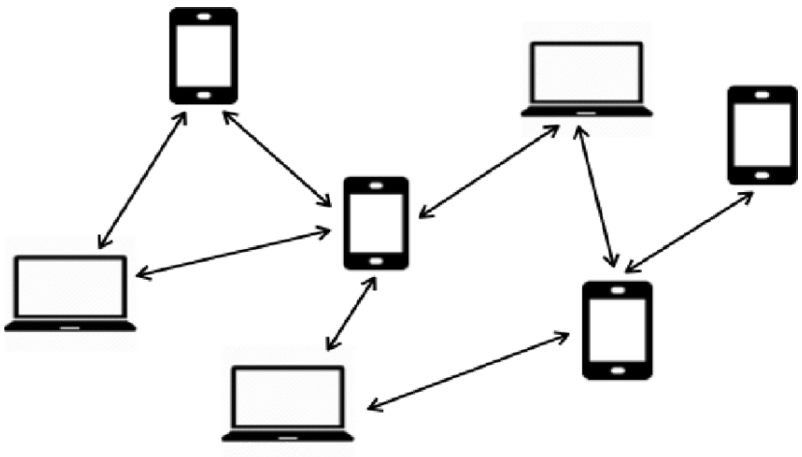
- **Μοντέλο Barabasi - Albert**. Το μοντέλο Barabasi - Albert παράγει συνθετικές τοπολογίες με ιδιότητες scale-free. Ο αλγόριθμος που ακολουθείται για την κατασκευή συνθετικής τοπολογίας, ξεκινάει ορίζοντας m_0 κόμβους και δημιουργώντας αυθαίρετα ζεύξεις μεταξύ τους, με την προϋπόθεση ότι το δίκτυο μας είναι συνεκτικό. Έπειτα, για κάθε επόμενη χρονική στιγμή εισάγουμε έναν νέο κόμβο και δημιουργούμε ζεύξεις με m κόμβους του δικτύου (όπου $m < m_0$). Η πιθανότητα να συνδεθεί ο νέος κόμβος με κάποιον τυχαίο κόμβο i είναι ανάλογη με τον βαθμό του i. Οι τοπολογίες που προκύπτουν από αυτό το μοντέλο, παρουσιάζουν power-law κατανομή βαθμού, έχουν

μικρό μέσο μήκος μονοπατιού, υψηλό συντελεστή συσταδοποίησης και επίσης προσομοιάζουν την δυναμική ανάπτυξη που υπάρχει στα πραγματικά δίκτυα. Ο τρόπος με τον οποίο η κατανομή βαθμού είναι power-law, είναι χρησιμοποιώντας μηχανισμό προτιμησιακής διασύνδεσης που υπάρχει σε πραγματικά δίκτυα, με κύριο παράδειγμα τα κοινωνικά. Η προτιμησιακή διασύνδεση σημαίνει ότι όσο μεγαλύτερο βαθμό έχει ένας κόμβος (είναι δηλαδή πιο ¨δημοφιλής¨), τόσοι περισσότεροι κόμβοι θέλουν να συνδεθούν με αυτόν. Λόγω, λοιπόν, της δυναμικής ανάπτυξης, της ιδιότητας scale-free και της ιδιότητας small-world που παρουσιάζει αυτή η συνθετική τοπολογία, αποτελεί μία πολύ καλή προσέγγιση για αρκετά πραγματικά δίκτυα.

2.8 Ασύρματα ad-hoc δίκτυα

Μία βασική έννοια που είναι απαραίτητη για την παρούσα διπλωματική εργασία, είναι τα ασύρματα ad-hoc δίκτυα. Με την εξέλιξη της τεχνολογίας, πλέον κάθε μηχάνημα έχει τη δυνατότητα να συνδέεται στο διαδίκτυο και να επικοινωνεί ασύρματα με άλλες συσκευές που βρίσκονται εντός του εύρους ζώνης μετάδοσης του. Ως εκ τούτου, έχει προκύψει μία νέα κατηγορία δικτύων, τα ασύρματα ad-hoc δίκτυα.

Τα δίκτυα αυτά είναι διαφορετικά από τα κλασικά σύνθετα δίκτυα που αναλύσαμε προηγουμένως, καθώς η τοπολογία τους είναι δυναμική και αλλάζει συνεχώς όσο οι κόμβοι του δικτύου κινούνται. Αποτέλεσμα αυτού είναι η συχνή δημιουργία και κατάργηση ζεύξεων μεταξύ κόμβων του δικτύου. Επίσης, σε αυτά τα δίκτυα δεν υπάρχει ανάγκη για κάποια σταθερή υποδομή, και για αυτό το λόγο χαρακτηρίζονται ως ad-hoc, δηλαδή αυτοοργανώμενα ή δίκτυα κατ' απαίτηση. Για να μπορεί να λειτουργήσει το δίκτυο χωρίς κάποια σταθερή υποδομή, οι κόμβοι του δικτύου λειτουργούν τόσο ως δρομολογητές πρός άλλους κόμβους του δικτύου, όσο και σαν hosts [32].



Σχήμα 2.6: *Απεικόνιση ενός κινητού ad-hoc δικτύου, από [6].*

Είναι σημαντικό να αναφερθεί, πως μία από τις βασικές κατηγορίες ad-hoc δικτύων, είναι τα κινητά ad-hoc δίκτυα ή αλλιώς MANETs. Στα δίκτυα αυτά, οι κόμβοι αντιπροσωπεύουν συσκευές με δυνατότητα ασύρματης επικοινωνίας, όπως για παράδειγμα κινητά και φορητοί υπολογιστές, και οι ζεύξεις μεταξύ κόμβων σχηματίζονται όταν ένας κόμβος εισέλθει εντός του

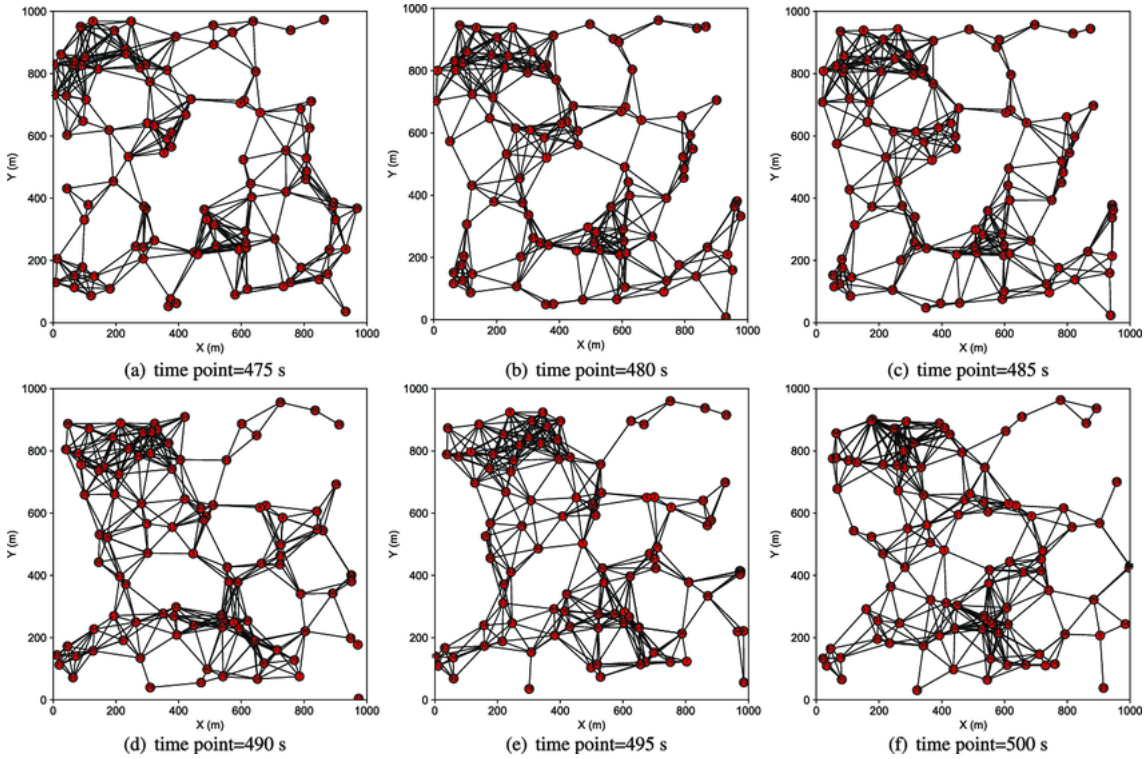
βελησνεκούς εκπομπής ασυρμάτου της συσκευής. Οι διασυνδέσεις αυτές μπορεί να είναι είτε αμφίδρομες ή μονόδρομες και μπορούν να καταργηθούν εάν κάποιος από τους δύο κόμβους μετακινηθεί εκτός της ζώνης μετάδοσης.

2.8.1 Ανάλυση ad-hoc δικτύων

Η δυναμική τοπολογία και η έλλειψη σταθερής υποδομής που αποτελεί βασικό χαρακτηριστικό των ad-hoc δικτύων, έχει ως αποτέλεσμα η ανάλυση των δικτύων αυτή να είναι ιδιαίτερα πολύπλοκη.

Αναφέρθηκε σε προηγούμενη ενότητα, ότι τα ad-hoc δίκτυα μπορούν να προσομοιαστούν με τη χρήση της συνθετικής τοπολογίας των τυχαίων γεωμετρικών γράφων. Ο λόγος για αυτή την αντιστοιχία, είναι το γεγονός ότι τα ασύρματα ad-hoc δίκτυα σχηματίζουν ζεύξεις με κριτήριο την γεωγραφική απόσταση μεταξύ κόμβων, καθώς η απόσταση αυτή αντιπροσωπεύει την εμβέλεια του σήματος. Με αυτή τη γνώση, μπορούμε να έχουμε μία εκτίμηση ως προς τις τιμές των μετρικών που θα λάβουμε αναλύοντας το δίκτυο.

Η διαδικασία που ακολουθείται προκειμένου να γίνει ανάλυση ενός ad-hoc δικτύου είναι με τη δημιουργία στιγμιοτύπων (snapshots) του δικτύου σε συγκεκριμένες χρονικές στιγμές, όπως φαίνεται στο Σχήμα 2.7. Τα στιγμιότυπα αυτά μας επιτρέπουν να αγνοήσουμε τη χρονομεταβλητότητα των ad-hoc δικτύων και να εφαρμόσουμε μεθόδους ανάλυσης κοινωνικών δικτύων στα στιγμιότυπα αυτά, αφού μπορούμε να δημιουργήσουμε από το στιγμιότυπο τον γράφο του δικτύου, δημιουργώντας ζεύξεις ανάλογα με την εμβέλεια του σήματος [33].



Σχήμα 2.7: *Στιγμιότυπα ενός κινητού ad-hoc δικτύου, από [7].*

2.8.2 Οδικά ad-hoc δίκτυα (VANETs)

Πλέον, σχεδόν κάθε σύγχρονο όχημα έρχεται εξοπλισμένο με διάφορους σένσορες και τεχνολογίες που του δίνουν τη δυνατότητα της ασύρματης επικοινωνίας με άλλους χρήστες του οδικού δικτύου.

Μπορούμε να δούμε, ότι το ασύρματο ad-hoc δίκτυο που προκύπτει εξαιτίας αυτής της δυνατότητας μπορεί να θεωρηθεί σαν μία εξειδίκευση των κινητών ad-hoc δικτύων. Τα νεα δίκτυα αυτά ονομάζονται οδικά ad-hoc δίκτυα ή αλλίως VANETs.

Οι κόμβοι που συναντάμε κυρίως στα VANETs είναι οι εξής [33]:

- **Οχήματα (Vehicles)**. Τα οχήματα μπορεί να είναι είτε ιδιωτικά, δηλαδή να ανήκουν σε κάποιο άτομο ή σε εταιρείες, ή οχήματα δημόσιας συγκοινωνίας, όπως για παράδειγμα λεωφορεία.

- **Διακομιστές (Servers / Origin)**. Οι κόμβοι αυτοί λειτουργούν ως αποθήκες δεδομένων για περιεχόμενο που υπάρχει στο δίκτυο. Δεν μπορούν να επικοινωνήσουν απευθείας με τα οχήματα, αλλά μόνο με τα RSUs.

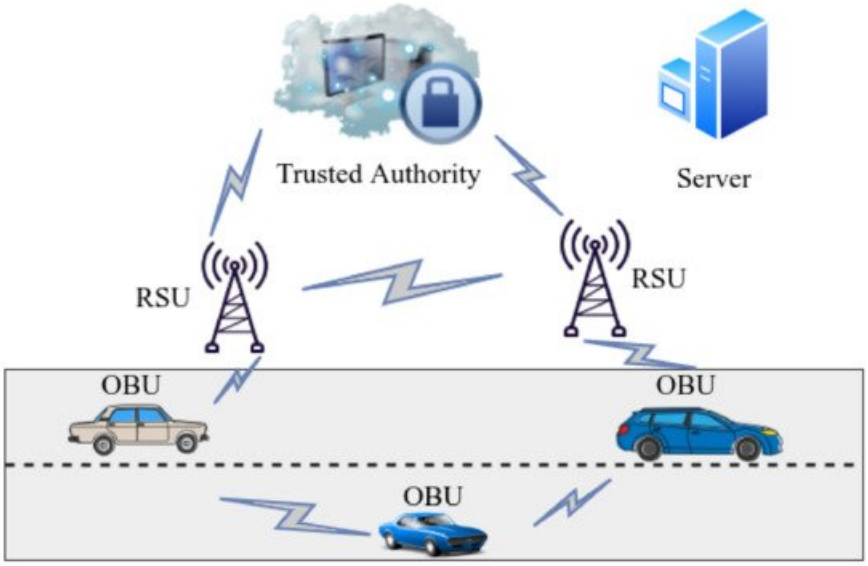
- **Roadside Units ή RSUs**. Οι κόμβοι αυτοί του δικτύου, λειτουργούν σαν γέφυρα επικοινωνίας των οχημάτων με τους διακομιστές. Στα RSUs μπορεί να βρίσκονται επίσης κάποιες αποθηκευμένες πληροφορίες για το δίκτυο που μπορούν να έχουν πρόσβαση τα οχήματα. Τα RSUs μπορεί να είναι είτε κινητά ή στατικά και ένα όχημα μπορεί, υπό συγκεκριμένες προϋποθέσεις να λειτουργήσει ως ένα RSU.

Αξίζει να σημειωθεί πως τα οδικά ad-hoc δίκτυα έχουν ορισμένες σημαντικές διαφορές από τα κινητά ad-hoc δίκτυα. Πιο συγκεκριμένα, στα VANETs, οι κόμβοι του δικτύου κινούνται με αρκετά πιο γρήγορες ταχύτητες σε σχέση με τα MANETs, κάτι που σημαίνει ότι αρκετοί αλγόριθμοι δρομολόγησης που χρησιμοποιούνται για τα κινητά ad-hoc δίκτυα δεν μπορούν να εφαρμοστούν και σε οδικά ad-hoc δίκτυα, καθώς η τοπολογία μεταβάλλεται πολύ πιο απότομα. Επίσης, υπάρχει διαφορά στον τρόπο κίνησης των κόμβων. Στα κινητά ad-hoc δίκτυα, οι κόμβοι κινούνται με ουσιαστικά τυχαίο τρόπο σε αντίθεση με τα οδικά ad-hoc δίκτυα, στα οποία η διαδρομή που ακολουθούν οι κόμβοι περιορίζονται από το οδικό δίκτυο [34].

2.8.3 Λόγοι Χρήσης και Έρευνας των VANETs

Η οδική ασφάλεια είναι ένα ζήτημα που μας αφορά όλους μας. Κάθε χρόνο, εκτιμάται ότι περίπου 1.35 εκατομμύρια άτομα πεθαίνουν σε αυτοκινητιστικό δυστύχημα [35]. Σκοπός της χρήσης των οδικών ad-hoc δικτύων είναι η αποφυγή συγκρούσεων και η δυναμική αλλαγή της διαδρομής που ακολουθεί το όχημα ώστε να εξασφαλιστεί η οδική ασφάλεια. Ο τρόπος που γίνεται αυτό είναι μέσα από την έγκαιρη και γρήγορη μετάδοση πληροφοριών για επικίνδυνες οδικές συνθήκες, όπως για παράδειγμα για πάγο στο οδόστρωμα, ή προειδοποιήσεις για ατυχήματα που έχουν συμβεί στο δρόμο [33].

Ακόμη, τα VANETs χρησιμοποιούνται για την παρακολούθηση της κυκλοφοριακής συμφόρησης και την ελαχιστοποίηση του χρόνου αναμονής ενός οχήματος. Για παράδειγμα, αν παρατηρηθεί σε κάποιο δρόμο έντονη κίνηση, μπορεί να σταλθεί σήμα στα οχήματα που



Σχήμα 2.8: *Διάγραμμα ενός οδικού ad-hoc δικτύου, από [8].*

οδεύουν προς εκεί να αλλάξουν τη διαδρομή τους και να ακολουθήσουν κάποια εναλλακτική διαδρομή. Προφανώς, σε αυτή την περίπτωση τα μηνύματα δεν χρειάζεται να είναι τόσο άμεσα όσο στην περίπτωση της αποφυγής ατυχημάτων [33].

Ένας επιπρόσθετος λόγος αξιοποίησης των οδικών ad-hoc δικτύων είναι η αύξηση της άνεσης κατά τη χρήση του οχήματος Αναλυτικότερα, με τη χρήση των VANETs, μπορεί ο οδηγός να ενημερώνεται για το πλησιέστερο πρατήριο καυσίμων σε περίπτωση που χρειάζεται ανεφοδιασμό, μπορεί να πληρώνει αυτόματα τα διόδια ή και να συνδέεται με μεγαλύτερη ευκολία στο διαδίκτυο, με τη βοήθεια των RSUs, που υπάρχουν εγκατεστημένα στον δρόμο και είναι συνδεδεμένα σε κάποιο πάροχο [9].

2.8.4 Αρνητικά των οδικών ad-hoc δικτύων

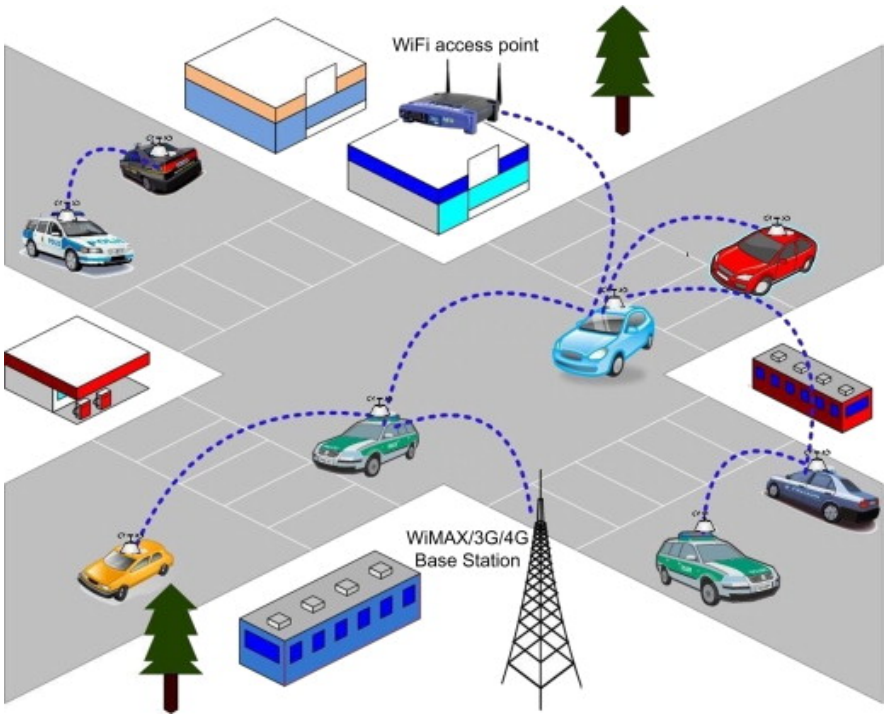
Παρά τις σημαντικές προσφορές που έγκεινται στη χρήση των VANETs, υπάρχουν ορισμένα μειονεκτήματα, τα οποία η μελέτη των δικτύων αυτών αποσκοπεί να εξαλείψει.

Αρχικά, υπάρχουν αρκετά θέματα ασφαλείας που προκύπτουν σε κάθε ασύρματο ad-hoc δίκτυο. Οι πληροφορίες που μεταδίδονται μπορεί να είναι επιστευτικές ή προσωπικές, και κάποιος κακόβουλος χρήστης, λόγω της έλλειψης υποδομής του δικτύου (όπως αναφέρθηκε, τα δίκτυα αυτά είναι αυτοοργανώμενα) μπορεί εύκολα να υποκλέψει τα μηνύματα. Κρίνεται, λοιπόν, αναγκαία η ύπαρξη κάποιου τρόπου ώστε να εξασφαλίζεται η εμπιστευτικότητα στο δίκτυο, δηλαδή να μην μπορεί να γίνει πρόσβαση ευαίσθητων δεδομένων μόνο από τους χρήστες που αφορά η πληροφορία που εμπεριέχεται στα μηνύματα [36].

Επίσης, τα δεδομένα που στέλνονται, ειδικά αυτά που αφορούν την οδική κατάσταση, πρέπει να προέρχονται από αυθεντικούς χρήστες του δικτύου. Ένας κακόβουλος χρήστης μπορεί να στέλνει μηνύματα χρησιμοποιώντας την ταυτότητα κάποιου RSU, κάτι που μπορεί να προκαλέσει σοβαρά προβλήματα στο δίκτυο και στη χειρότερη περίπτωση, ατυχήματα. Για το λόγο αυτό, είναι απαραίτητη η ταυτοποίηση των χρηστών του δικτύου ώστε να μην υπάρχει περίπτωση να σταλθεί μήνυμα από χρήστες εκτός του δικτύου [36].

Ένα επιπλέον πρόβλημα αφορά την αξιοπιστία των μηνυμάτων. Ένα μήνυμα για να φτάσει σε έναν κόμβο τις περισσότερες φορές περνάει από αρκετά οχήματα τα οποία το αναμεταδίδουν μέχρι να φτάσει στον προορισμό του. Συνέπεια αυτού είναι πως ένα όχημα μπορεί να αλλάξει πληροφορίες ενός μηνύματος που πρόκειται να αναμεταδόσει και έτσι να φτάσουν λανθασμένες πληροφορίες. Όπως αναφέρθηκε και προηγουμένως, αυτό είναι κάτι που μπορεί να έχει πολύ σημαντικά προβλήματα στο οδικό δίκτυο [36].

Τέλος, ένα πιο δομικό πρόβλημα που υπάρχει στα VANETs είναι η κακή κλιμακωσιμότητα τους. Αυτό σημαίνει ότι το VANET δεν μπορεί να χειριστεί την εισαγωγή νέων κόμβων στο δίκτυο χωρίς να μειωθεί η απόδοση του. Ο λόγος για τον οποίο προκύπτει αυτό το πρόβλημα, είναι διότι μία αύξηση στον αριθμό των κόμβων του δικτύου συνεπάγεται μεγαλύτερη συμφόρηση στην ασύρματη επικοινωνία. Ο τρόπος που οι πληροφορίες μεταδίδονται είναι με αναμετάδοση από τους κόμβους, οπότε περισσότεροι κόμβοι έχουν ως αποτέλεσμα την αναμετάδοση του ίδιου μηνύματος από μεγαλύτερο πλήθος οχημάτων και συμφόρηση του δικτύου [37].



Σχήμα 2.9: *Ένα οδικό ad-hoc δίκτυο, από [9].*

2.9 Μηχανική Μάθηση

Αρχικά, για να ορίσουμε τη μηχανική μάθηση, χρειάζεται να εξηγήσουμε τι είναι η τεχνητή νοημοσύνη artificial intelligence). Τεχνητή νοημοσύνη είναι ο τομέας της επιστήμης υπολογιστών που αποσκοπεί στη δημιουργία μηχανών, και ειδικά υπολογιστικών μηχανών, με ικανότητες που προσομοιάζουν την ανθρώπινη νοημοσύνη. Ο κλάδος αυτός ξεκίνησε από τον Alan Turing το 1950 με το επιστημονικό άρθρο του με τίτλο "Computing Machinery and Intelligence", στο οποίο τίθεται η ερώτηση αν οι μηχανές μπορούν να σκεφτούν [38].

Ένας, λοιπόν, πολύ δημοφιλής κλάδος της τεχνητής νοημοσύνης, είναι η μηχανική μάθηση (machine learning). Ο κλάδος αυτός ασχολείται με τη δημιουργία αλγορίθμων, οι οποίοι επιτρέπουν σε μηχανές να ¨μαθαίνουν¨ τόσο από καινούργια δεδομένα όσο και από παλαιότερες γνώσεις. Αναλυτικότερα, ο τρόπος με τον οποίο ένα μηχάνημα είναι ικανό να εκτελέσει αυτή τη λειτουργία είναι εντοπίζοντας και αναγνωρίζοντας μοτίβα που υπάρχουν σε δεδομένα που έχει επεξεργαστεί και στη συνέχεια κάνοντας προβλέψεις για τα νέα δεδομένα που του δίνονται, χωρίς να απαιτείται ιδιαίτερη ανθρώπινη παρέμβαση [39].

Υπάρχουν 3 βασικές κατηγορίες μηχανικής μάθησης, η επιβλεπόμενη μάθηση (supervised learning), η μή επιβλεπόμενη μάθηση (unsupervised learning) και η ενισχυτική μάθηση (reinforcement learning).

2.9.1 Επιβλεπόμενη μάθηση

Η επιβλεπόμενη μάθηση είναι η πρώτη κατηγορία που θα αναλύσουμε. Ο τρόπος που λειτουργεί είναι με τη χρήση συνόλων δεδομένων με ετικέτες (labeled datasets), ώστε να εκπαιδευτεί ο αλγόριθμος μηχανικής μάθησης για να μπορέσει να ταξινομέι δεδομένα ή να προβλέπει αποτελέσματα με ακρίβεια. Το σύνολο δεδομένων που δίνεται στο μοντέλο για εκπαίδευση περιλαμβάνει εισόδους και τις σωστές εξόδους που αντιστοιχούν σε αυτές (δηλαδή τα labels) και με αυτό τον τρόπο το μοντέλο σταδιακά μαθαίνει να αναγνωρίζει τις αντιστοιχίες που υπάρχουν και να δίνει σωστές απαντήσεις. Η ακρίβεια του αλγορίθμου μετριέται με τη χρήση συνάρτησης απώλειας (loss function) και αλλάζουμε τις παραμέτρους του μοντέλου μέχρι να είμαστε ικανοποιημένοι από το περιθώριο λάθους [40].

Με την επιβλεπόμενη μηχανική μάθηση μπορούμε να αντιμετωπίσουμε δύο κατηγορίες προβλημάτων:

- **Προβλήματα ταξινόμησης (Classification)**. Στα προβλήματα ταξινόμησης, ο στόχος είναι η τοποθέτηση των δεδομένων που έχουμε σε κατηγορίες. Το μοντέλο πρέπει να βρεί χαρακτηριστικά ανάμεσα στα δεδομένα που του δίνουμε και καλείται να τοποθετήσει τα δεδομένα με συγκεκριμένα κοινά χαρακτηριστικά στην αντίστοιχη ομάδα. Ένα παράδειγμα είναι η ταξινόμηση εικόνων από διάφορα ζώα στην αντίστοιχη κατηγορία ζώων που ανήκουν. Οι πιο συχνοί αλγόριθμοι ταξινόμησης είναι οι γραμμικοί ταξινομητές (linear classifiers), τα support vector machines ή SVM, και το k-nearest neighbor [40].

- **Προβλήματα Παλινδρόμησης (Regression)**. Στην ανάλυση παλινδρόμησης, ο στόχος είναι να κατανοήσουμε τη σχέση μεταξύ κάποιας εξαρτημένης και άλλων ανεξάρτητων μεταβλητών. Παράδειγμα αυτής της ανάλυσης, είναι η πρόβλεψη της τιμής ενός σπιτιού, με δεδομένα τα χαρακτηριστικά του σπιτιού όπως το μέγεθός του και η τοποθεσία του [41]. Οι πιο δημοφιλείς αλγόριθμοι που χρησιμοποιούνται είναι η γραμμική παλινδρόμηση (linear regression) και η λογιστική παλινδρόμηση (logistic regression).

2.9.2 Μη επιβλεπόμενη μάθηση

Η επόμενη κατηγορία μηχανικής μάθησης που θα εξηγήσουμε είναι η μη επιβλεπόμενη μάθηση. Σε αντίθεση με την επιβλεπόμενη μάθηση, οι αλγόριθμοι που χρησιμοποιούμε

έχουν ως στόχο να αναλύσουν και να συσταδοποιήσουν σύνολα δεδομένων στα οποία δεν υπάρχουν ετικέτες. Ως εκ τούτου η κατηγορία ονομάζεται μη επιβλεπόμενη μάθηση, διότι δεν χρειάζεται ανθρώπινη επέμβαση, δηλαδή υπόδειξη μέσω των ετικετών. Οι αλγόριθμοι αυτοί βρίσκουν κρυμμένα μοτίβα ή ομαδοποιήσεις δεδομένων που υπάρχουν και αυτή η ικανότητα κάνει αυτό το είδος μηχανικής μάθησης ιδανικό για τομείς όπως διερευνητική ανάλυση [42].

Για την μη επιβλεπόμενη μάθηση υπάρχουν 3 τεχνικές που χρησιμοποιούνται:

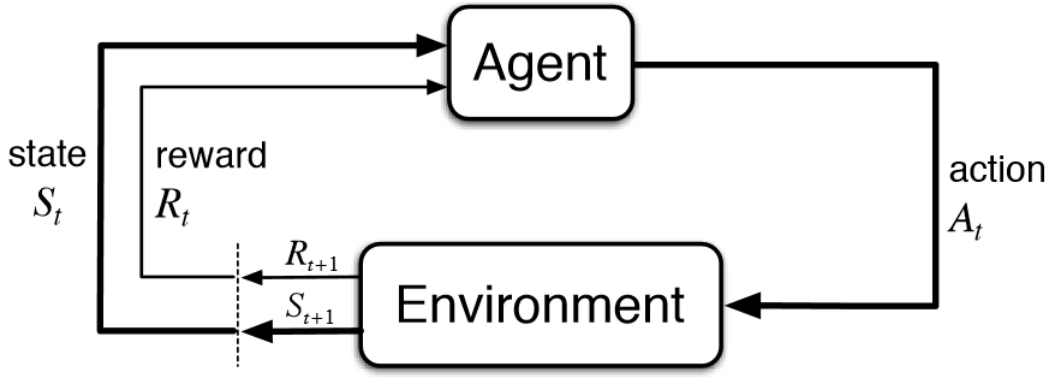
- **Συσταδοποίηση (Clustering)**. Η συσταδοποίηση λειτουργεί βρίσκοντας ομοιότητες σε δεδομένα χωρίς ετικέτες και τοποθετώντας τα στην ίδια συστάδα. Ο πιο βασικός αλγόριθμος συσταδοποίησης είναι το K-means clustering, ένας αλγόριθμος, ο οποίος αρχικά ορίζει k κέντρα συστάδων (centroids) και στη συνέχεια αντιστοιχίζει κάθε σημείο δεδομένων που έχουμε στον χώρο σε μία συστάδα ανάλογα με την απόσταση του σημείου από αυτή. Τα κέντρα των συστάδων αρχικά επιλέγονται τυχαία και με κάθε επανάληψη του αλγορίθμου βρίσκονται οι βέλτιστες θέσεις τους [43].

- **Κανόνες Συσχέτισης (Association Rules)**. Οι κανόνες συσχέτισης είναι μία τεχνική μη επιβλεπόμενης μηχανικής μάθησης, η οποία στοχεύει στην ανακάλυψη κανόνων που περιγράφουν μεγάλα τμήματα από το σύνολο δεδομένων που έχουμε. Για παράδειγμα ένας κανόνας συσχέτισης είναι ¨οι άνθρωποι που αγοράζουν το αντικείμενο X αγοράζουν και το αντικείμενο Y¨ [44].

- **Μείωση Διαστατικότητας (Dimensionality Reduction)**. Στη μηχανική μάθηση, υπάρχει το πρόβλημα της υπερπροσαρμογής (overfitting), που προκύπτει όταν εκπαιδεύουμε το μοντέλο μας πολλές φορές στο training set, και ως εκ τούτου το μοντέλο χάνει την ικανότητα του να γενικεύει. Πολύ σύνθετα μοντέλα, που προκύπτουν όταν τα δεδομένα μας έχουν πάρα πολλά χαρακτηριστικά (δηλαδή υψηλή διαστατικότητα) τείνουν να παρουσιάζουν υπερπροσαρμογή. Για το λόγο αυτό, χρησιμοποιείται η τεχνική της μείωσης διαστατικότητας που μειώνει τον αριθμό των χαρακτηριστικών στα δεδομένα αφαιρώντας αχρείαστα χαρακτηριστικά και αντιμετωπίζοντας κατά συνέπεια το πρόβλημα της υπερπροσαρμογής [45].

2.9.3 Ενισχυτική μάθηση

Η ενισχυτική μάθηση είναι μία κατηγορία μηχανικής μάθησης, η οποία ασχολείται με τη λήψη της κατάλληλης απόφασης ώστε να μεγιστοποιήσουμε το κέρδος σε μία συγκεκριμένη περίσταση. Πιο συγκεκριμένα, χρησιμοποιείται για την εύρεση του βέλτιστου μονοπατιού που πρέπει να ακολουθηθεί ώστε να φτάσουμε σε ένα επιθυμητό αποτέλεσμα. Σε αντίθεση με την επιβλεπόμενη μάθηση, στην οποία το σετ εκπαίδευσης εμπεριέχει τις ετικέτες (που είναι η σωστή απάντηση), στην ενισχυτική μάθηση δεν υπάρχει εξαρχής σωστή απάντηση, αλλά μόνο ένας στόχος, και ο τρόπος με τον οποίο θα επιτευγθεί ο στόχος αυτός βρίσκεται μέσα από τη διαδικασία της ανταμειβής για σωστή απόφαση και τιμωρία για λανθασμένη [46].



Σχήμα 2.10: *Βρόχος πράξης - ανταμειβής για ενισχυτική μάθηση, από [10].*

2.9.4 Γνωστές βιβλιοθήκες μηχανικής μάθησης

Μία βιβλιοθήκη μηχανικής μάθησης είναι ένα σύνολο από αλγορίθμους μηχανικής μάθησης και εξόρυξης δεδομένων, γραμμένες σε κάποια γλώσσα προγραμματισμού. Οι βιβλιοθήκες αυτές, παρέχουν έτοιμες συναρτήσεις και όχι μόνο μαθηματικούς, αλλά και στατιστικούς τυύπους, με σκόπο τη μείωση του κόπου που χρειάζεται για την συγγραφή κώδικα μηχανικής μάθησης. Ως εκ τούτου, μπορεί ο προγραματιστής να αφήσει τα χρονοβόρα και κουραστικά κομμάτια της μηχανικής μάθησης στη βιβλιοθήκη και να ασχοληθεί με άλλα σημαντικά ζητήματα. [47]

Οι πιο δημοφιλείς βιβλιοθήκες ειναι οι εξής [47]:

- **NumPy**. Η NumPy πρόκειται για μία πολύ δημοφιλή βιβλιοθήκη γραμμένη σε Python που επιτρέπει την εκτέλεση πολύπλοκης γραμμικής άλγεβρας, όπως για παράδειγμα πράξεις με πολυδιάστατους πίνακες. Ακόμα, έχει δυνατότητες υπολογισμού μετασχηματισμών Fourier και παραγωγής τυχαίων αριθμών.

- **SciPy**. Το SciPy παρέχει συναρτήσεις για αριθμητική ολοκλήρωση, παρεμβολή, βελτιστοποίηση, γραμμική άλγεβρα και στατιστική. Χρησιμοποιείται συχνά στην επεξεργασία εικόνας.

- **Scikit-learn**. Το Scikit-learn είναι μία βιβλιοθήκη που βασίζεται στο NumPy και στο SciPy και χρησιμοποιείται για εξόρυξη και ανάλυση δεδομένων. Παρέχει αλγορίθμους για συσταδοποίηση, παλινδρόμηση και κατηγοριοποίηση.

- **TensorFlow**. Μία βιβλιοθήκη της Google, βασισμένη σε Python που χρησιμποιείται για εφαρμογές βαθιάς μάθησης. Χρησιμοποιείται από την Google σε εφαρμογές που χρησιμοποιούν αναγνώριση φωνής και εικόνας.

Κεφάλαιο 3

Προαπαιτούμενα προγράμματα

Στο κεφάλαιο αυτό, θα παρουσιαστούν τα προγράμματα και τα πλαίσια (frameworks) που χρησιμοποιήθηκαν για την υλοποίηση της διπλωματικής εργασίας.

3.1 OMNET++

Το κύριο πρόγραμμα που χρησιμοποιήθηκε, πάνω στο οποίο χτίστηκε η προσομοίωση που θα μελετήσουμε στην εργασία αυτή είναι το OMNET++.

3.1.1 Τι είναι το OMNET++

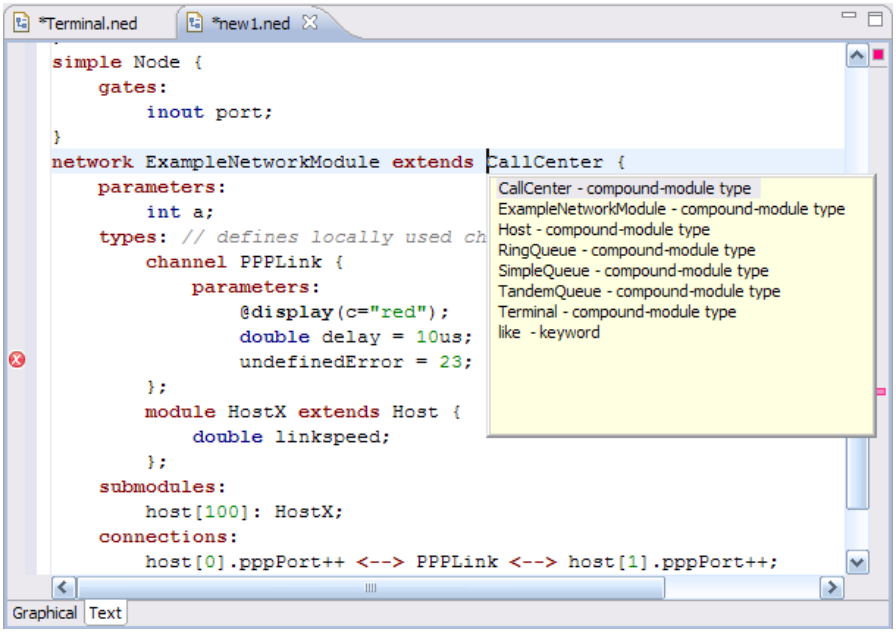
Το OMNET++ είναι ένα εργαλείο που χρησιμοποιείται για τη δημιουργία διαφόρων προσομοιωτών δικτύων. Αναλυτικότερα, είναι μία επεκτάσιμη, modular βιβλιοθήκη βασισμένη στη γλώσσα προγραμματισμού C++ που περιέχει και ενσωματωμένο γραφικό περιβάλλον για τη συγγραφή κώδικα (IDE) με βάση το Eclipse. Ακόμα, παρέχεται το διαδραστικό γραφικό περιβάλλον Qtenv (GUI) που χρησιμοποιείται για την διαχείριση προσομοιώσεων.

Στη σελίδα του OMNET++ παρέχονται και ανεξάρτητα projects, που μπορούν να χρησιμοποιηθούν για να προσθέσουν νέες λειτουργικότητες στην προσομοίωση. Αξίζει να τονιστεί όμως, ότι το OMNET++ δεν είναι είναι ένας προσομοιωτής δικτύου, αλλά χρησιμοποιείται σαν πλατφόρμα για τη δημιουργία προσομοιωτών δικτύου [48].

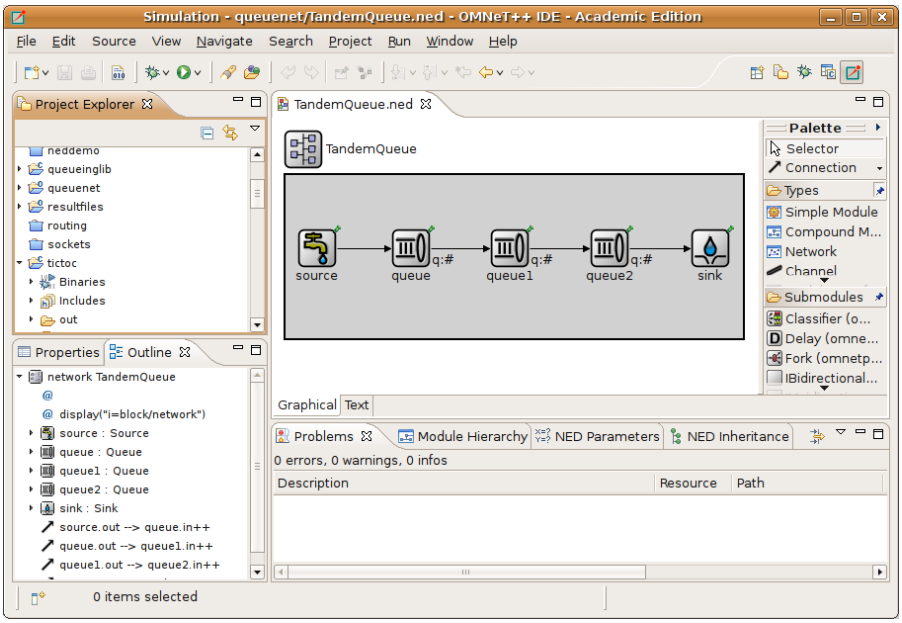
3.1.2 Πως φτιάχνουμε μια προσομοίωση

Αρχικά, το OMNET++ ακολουθεί αρχιτεκτονική που βασίζεται σε "μονάδες" (components ή modules) για τη δημιουργία μοντέλων που χρησιμοποιούνται για την προσομοίωση. Πιο συγκεκριμένα, τα modules προγραμματίζονται σε C++ και ενώνονται σε μοντέλα με τη χρήση της γλώσσας προγραμματισμού NED (Network Description) του OMNET++. Η γλώσσα NED είναι μία γλώσσα υψηλού επιπέδου, που περιγράφει την τοπολογία του μοντέλου που φτιάχνουμε [48].

Το γραφικό περιβάλλον του OMNET++, δίνει επίσης τη δυνατότητα επεξεργασίας της τοπολογίας του μοντέλου που περιγράφεται από τα NED αρχεία χωρίς τη χρήση κώδικα.



Σχήμα 3.1: *Το IDE του OMNET++, από [11].*



Σχήμα 3.2: *Επεξεργασία NED αρχείων μέσα από το γραφικό περιβάλλον του OMNET++, από [11].*

3.2 SUMO

Το επόμενο πρόγραμμα που θα περιγράψουμε είναι το SUMO (Simulation of Urban MObility).

3.2.1 Τι είναι το SUMO

Το SUMO είναι ένα open-source πρόγραμμα, το οποίο επιτρέπει την προσομοίωση κίνησης σε οδικά δίκτυα. Αυτό σημαίνει ότι παρέχει τη δυνατότητα προσομοίωσης δικτύων που

έχουν το μέγεθος μίας πόλης. Υπάρχει επίσης η επιλογή προσαρμογής του μεγέθους του δικτύου, υπό την προϋπόθεση ότι το μηχάνημα στο οποίο γίνεται η προσομοίωση έχει αρκετή υπολογιστική δύναμη [49].

3.2.2 Λειτουργίες του SUMO

Η χρήση του SUMO παρέχει μία πληθώρα λειτουργιών στον χρήστη. Οι πιο σημαντικές εξ' αυτών είναι οι εξής [50]:

- **Δυνατότητα εισαγωγής δικτύου**. Στο SUMO είναι δυνατή η εισαγωγή οδικών δικτύων από τις πιο συχνές μορφές αρχείων όπως για παράδειγμα OpenStreetMap, VISUM, VISSIM, NavTeq, MATsim και OpenDRIVE.
- **Μοντελοποίηση οδικής κίνησης**. Με τη χρήση του SUMO, έχουμε τη δυνατότητα να μοντελοποιήσουμε και να προσομοιώσουμε συστήματα οδικής κίνησης που περιέχουν διάφορα οχήματα και πεζούς. Τα οχήματα μπορεί να είναι αυτοκίνητα, λεωφορεία, τρένα, ποδήλατα και πολλά άλλα.
- **Έλεγχος οχημάτων**. Το Traffic Control Interface (TraCI) που παρέχεται στο SUMO, επιτρέπει τον έλεγχο της συμπεριφοράς οχημάτων που βρίσκονται στην προσομοίωση, όπως για παράδειγμα η αλλαγή της διαδρομής που ακολουθεί ένα όχημα.
- **Λειτουργικότητα φαναριών**. Στο SUMO υπάρχει η λειτουργία προμοίωσης φαναριών στο δρόμο και η δημιουργία αυτόματων προγραμμάτων που καθορίζουν την κατάσταση του φαναριού. Υπάρχει επίσης η επιλογή εισαγωγής κάποιου προγράμματος από αρχείο, αντί για την αυτόματη δημιουργία.
- **Εξαγωγή μετρικών οχημάτων**. Ακόμα, υπάρχουν λειτουργίες που επιτρέπουν την εξαγωγή μετρικών από τα οχήματα κάθε χρονική στιγμή, όπως για παράδειγμα η ταχύτητα και κατεύθυνση ή τις εκκρίσεις διοξειδίου του άνθρακα τους (CO_2), καθώς και την αλλαγή της πορείας που ακολουθεί κάποιο όχημα.

3.3 Veins

Το τελευταίο εργαλείο που χρησιμοποιήσαμε για να φτιάξουμε την προσομοίωση είναι το Veins.

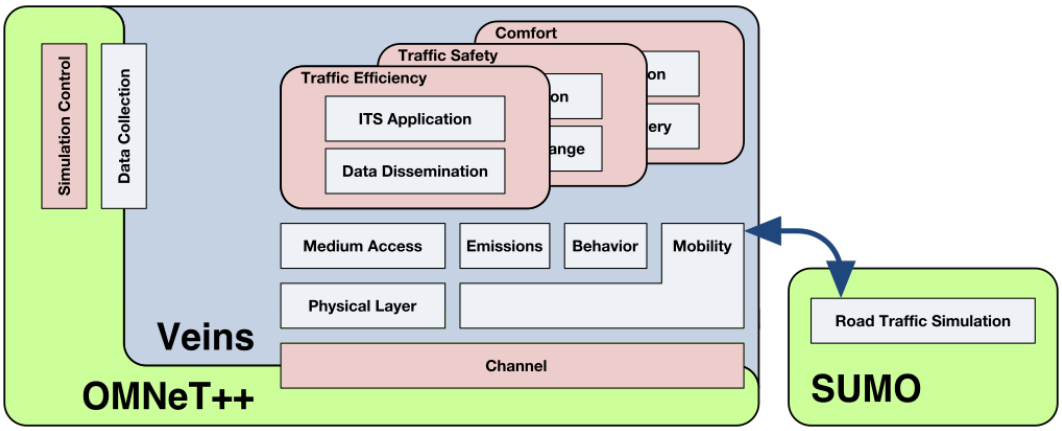
3.3.1 Τι είναι το Veins

Το Veins είναι ένα πλαίσιο ανοιχτού κώδικα που χρησιμοποιείται για τη δημιουργία προσομοιώσεων οδικών δικτύων. Πιο συγκεκριμένα, επεκτείνει τα δύο εργαλεία που περιγράψαμε προηγουμένως, δηλαδή το OMNET++ και το SUMO, ώστε να παρέχει δυνατότητα για επικοινωνία μεταξύ οχημάτων [12].

Στο πλαίσιο αυτό, παρέχεται μεγάλος αριθμός από μοντέλα προσομοίωσης, τα οποία μπορούν να χρησιμοποιηθούν σε προσομοιώσεις. Η επιλογή των μοντέλων που θα χρησιμοποιηθούν σε κάθε προσομοίωση καθορίζεται από τον χρήστη ανάλογα με τις ανάγκες που υπάρχουν [12].

3.3.2 Πως χρησιμοποιείται το Veins

Για τη σωστή χρήση του Veins, πρέπει αρχικά να τρέχει η προσομοιωτής δικτύου που χτίστηκε με το OMNET++ και ταυτόχρονα να τρέχει ο προσομοιωτής οδικής κίνησης SUMO [51]. Οι δύο αυτοί προσομοιωτές επικοινωνούν με μία υποδοχή TCP και χρησιμοποιείται το πρωτόκολλο TraCI. Με τον τρόπο αυτό, γίνεται εφικτή η σύνδεση του OMNET++ και του SUMO και η κίνηση των οχημάτων στον προσομοιωτή οδικής κίνησης μπορεί να αναπαρασταθεί με την κίνηση κόμβων στην προσομοίωση του δικτύου [12].



Σχήμα 3.3: *Σύνδεση του Veins με το OMNET++ και το SUMO, από [12].*

Μέρος II

Πρακτικό Μέρος

Κεφάλαιο 4

Υλοποίηση Κώδικα

Στο κεφάλαιο αυτό θα παρουσιαστεί ο κώδικας που γράφτηκε και οι αλγόριθμοι που υλοποιήθηκαν για την υλοποίηση των προσομοιώσεων που είναι το αντικείμενο της διπλωματικής εργασίας. Αξίζει να σημειωθεί πως ο κώδικας που γράφτηκε επεκτείνει το παράδειγμα που έρχεται με το Veins, χρησιμοποιώντας τον χάρτη του πανεπιστημίου Erlangen στη Νυρεμβέργη για την προσομοίωση οδικής κίνησης μέσω του SUMO.

Οι κόμβοι του δικτύου μας είναι οχήματα (Cars), στατικά RSU και ένα Origin. Η λειτουργία των κόμβων αυτών έχει επεξηγηθεί στο θεωρητικό μέρος.

4.1 Δομή Κώδικα

Αρχικά, θα αναφερθούμε στα αρχεία που δημιουργήθηκαν για τις προσομοιώσεις, καθώς και η χρήση που επιτελούν.

4.1.1 Αρχεία προσομοίωσης

Ξεκινώντας με τον κώδικα γραμμένο σε C++, υπάρχουν 4 βασικά αρχεία και οι αντίστοιχες επικεφαλίδες. Σημειώνεται ότι για κάθε ένα από αυτά, υπάρχει και ένα αντίστοιχο NED αρχείο που χρησιμοποιείται για να δηλώσει την κλάση στο OMNET++. Τα αρχεία αυτά είναι τα εξής:

- **UnitHandler**. Το αρχείο αυτό είναι η κλάση "γονέας" που χρησιμοποιείται από κάθε κόμβο του δικτύου. Περιέχει επί το πλείστον βασικές συναρτήσεις που χρησιμοποιούνται από όλες τις κλάσεις "παιδιά", όπως για παράδειγμα τη συνάρτηση που χειρίζεται τον τρόπο αποστολής μηνυμάτων. Στον ορισμό της κλάσης UnitHandler γίνεται κληρονόμηση από την κλάση DemoBaseApplLayer, η οποία υπάρχει στο παράδειγμα που έρχεται με το Veins και δεν θα αναλυθεί περεταίρω.

- **OriginHandler**. Το OriginHandler είναι η κλαση που προσδιορίζει τη συμπεριφορά των Origin κόμβων και προσθέτει μερικές συναρτήσεις αποκλειστικές σε αυτούς. Μία τέτοια συνάρτηση είναι η συνάρτηση που δίνει εντολή στα RSUs να υπολογίσουν την κεντρικότητα που τους ζητάει το Origin.

- **RsuHandler**. Η κλάση αυτή, αντίστοιχα, περιέχει συναρτήσεις που περιγράφουν τον τρόπο με τον οποίο τα RSUs χειρίζονται διάφορα σενάρια. Παράδειγμα αυτού είναι

η αντιμετώπιση αιτημάτων από τα οχήματα για κάποιο περιεχόμενο που είναι αποθηκευμένο σε κάποιο RSU ή Origin της προσομοίωσης.

- **CarHandler**. Τέλος, το CarHandler επίσης έχει συναρτήσεις που ασχολούνται με τη διαχείριση και δημιουργία αιτημάτων. Τα οχήματα στο δίκτυο είναι κινητά, οπότε υπάρχει και συνάρτηση που καλείται κάθε στιγμή της προσομοίωσης για σκοπούς όπως η ανίχνευση ατυχήματος.

Επίσης, δύο αρχεία που χρησιμοποιούνται στην προοσομοίωση είναι τα **KMeansML.py** και **AgglomML.py**. Τα δύο αυτά αρχεία είναι γραμμένα σε Python και μέσα σε αυτά υπάρχει κώδικας που εκτελεί κάποιον αλγόριθμο μηχανικής μάθησης και γράφει τα αποτελέσματα του αλγορίθμου σε ένα αρχείο .csv. Το KMeansML.py εκτελεί τον αλγόριθμο K-Means ενώ το AgglmoML.py εκτελεί τον Agglomerate Clustering.

4.1.2 Αρχείο μηνυμάτων

Ακόμη, ένα σημαντικό αρχείο που πρέπει να εξηγήσουμε, είναι το αρχείο Message.msg. Τα .msg αρχεία έχουν ιδιαίτερη σύνταξη και όταν γίνεται compile ο κώδικας, το OMNET++ αναλαμβάνει να δημιουργήσει τα αντίστοιχα αρχεία και την επικεφαλίδα τους σε C++. Δηλαδή, θα δημιουργηθούν τα αρχεία Message_m.cc και Message_m.h.

Το Message ορίζει έναν τύπο πακέτου που περιγράφει τις πληροφορίες που περιέχουν τα μηνύματα που στέλνονται από τους κόμβους του δικτύου κατά την προσομοίωση. Κάθε κόμβος για να στείλει ένα μήνυμα πρώτα προγραμματίζει τη χρονική στιγμή αποστολής και στη συνέχεια το μήνυμα αποστέλλεται στον εαυτό του (self-message). Έπειτα, ανάλογα με ορισμένες πληροφορίες του μηνύματος καθορίζεται αν θα σταλθεί και σε άλλους κόμβους ή θα χρησιμοποιηθεί για άλλους σκοπούς.

Θα χωρίσουμε τις πληροφορίες που περιέχει ένα μήνυμα ανάλογα με το είδος της πληροφορίας και θα αναφέρουμε τη λειτουργία τους. Για αρχή, οι πληροφορίες που έχουν να κάνουν με τα στοιχεία αποστολής του μηνύματος είναι οι εξής:

- **SenderAddress**. Η διεύθυνση του τελευταίου κόμβου που έστειλε το μήνυμα. Χρησιμοποιείται για να μπορούμε να πάρουμε μία ένδειξη της διαδρομής του μηνύματος.

- **Recipient**. Η διεύθυνση του κόμβου που θα παραλάβει το μήνυμα. Όποιος κόμβος, του οποίου η διεύθυνση είναι διαφορετική από αυτή που αναγράφεται σε αυτό το πεδίο, πάρει το μήνυμα το αγνοεί.

- **Source**. Η διεύθυνση του κόμβου που δημιούργησε το μήνυμα. Χρησιμοποιείται για διάφορους σκοπούς, κυρίως για να ξέρουμε σε ποιόν κόμβο πρέπει να απαντήσουμε όταν μας έρθει κάποιο αίτημα.

- **Dest**. Η διεύθυνση του κόμβου στον οποίο θέλουμε να φτάσει το μήνυμα. Στην περίπτωση που λάβει αυτό το μήνυμα κόμβος με διεύθυνση διαφορετική από αυτή που βρίσκεται στο Dest, προωθεί το μήνυμα ώστε να φτάσει στον προορισμό του.

- **SenderPosition**. Οι συντεταγμένες του τελευταίου κόμβου που έστειλε το μήνυμα. Χρησιμοποιείται για να δούμε αν ο αποστολέας του μηνύματος είναι εντός του εύρους ζώνης μετάδοσης.

Ακόμα, οι πληροφορίες που αφορούν χαρακτηριστικά του μηνύματος είναι:

- **MaxHops**. Είναι ο μέγιστος αριθμός αναπηδήσεων του μηνύματος που μπορεί να κάνει πριν διαγραφεί.
- **Hops**. Είναι ο αριθμός αναπηδήσεων που έχει κάνει το μήνυμα μέχρι τώρα. Αν ξεπεράσει το MaxHops το μήνυμα διαγράφεται.
- **Type**. Το πεδίο αυτό προσδιορίζει το είδος του μηνύματος που στέλνεται. Υπάρχουν 18 διαφορετικοί τύποι μηνυμάτων, ο καθένας για διαφορετική περίπτωση και λειτουργία.
- **State**. Όπως αναφέρθηκε στην αρχή αυτής της υποενότητας, κάθε μήνυμα πρώτα στέλνεται σαν ¨αυτο-μήνυμα¨ ή αλλιώς self-message. Το State καθορίζει το σε τι κατάσταση λειτουργίας αντιστοιχεί το μήνυμα αυτό. Για παράδειγμα, το SENDING, που είναι η προεπιλεγμένη κατάσταση, στέλνει το μήνυμα στους κόμβους που έχουν προσδιοριστεί. Αντίθετα, η κατάσταση CACHING δεν στέλνει μήνυμα σε άλλους κόμβους, αλλά ξεκινάει τη διαδικασία διαχείρισης αποθήκευσης περιεχομένου. Υπάρχουν σύνολο 7 διαφορετικές προθέσεις μηνύματος.
- **Centrality**. Το πεδίο αυτό χρησιμοποιείται μόνο για αιτήματα υπολογισμού κέντρικότητας και τις απαντήσεις τους. Προσδιορίζει τι είδος κεντρικότητας θέλουμε να υπολογίσουμε.

Επιπλέον, υπάρχουν πληροφορίες που χρησιμοποιούνται για την αποθήκευση αριθμητικών δεδομένων και δεδομένων χρόνου προσομίωσης:

- **MsgInfo**. Σε αυτό το πεδίο αποθηκεύουμε αριθμητικές πληροφορίες που θέλουμε να στείλουμε, όπως για παράδειγμα αποτελέσματα κεντρικότητας.
- **AckInfo**. Εδώ αποθηκεύονται πληροφορίες με τύπο simtime_t, δηλαδή χρόνου προσομοίωσης. Η χρήση του πεδίου είναι για την αποστολή επιβεβαιώσεων (Acknowledgements) και την εξασφάλιση ότι η επιβεβαίωση που λάβαμε είναι για το σωστό περιεχόμενο.

Ένα ακόμα είδος πεδίων του μηνύματος, είναι τα πεδία, τα οποία χρησιμοποιούνται για την εύρεση βέλτιστης διαδρομής ή για την διάσχιση μονοπατιών που δεν μεταβάλλονται με το χρόνο (για παράδειγμα μονοπάτια μεταξύ RSUs).

- **Route**. Ένας vector που αποθηκεύει τη διαδρομή που έχει ακολουθηθεί μέχρι τώρα. Χρησιμοποιείται κυρίως για μηνύματα εύρεσης βέλτιστης διαδρομής. Μία άλλη χρήση είναι για μηνύματα μεταξύ Origin και RSU. Αφού το Origin και τα RSU είναι στατικά, αν υπολογιστεί η βέλτιστη διαδρομή μεταξύ τους, μπορούμε να χρησιμοποιούμε αυτή τη διαδρομή για την ταχύτερη αποστολή μηνύματος.

- **PreviousNodes**. Ένας vector που αποθηκεύει τη διαδρομή που έχει ακολουθήσει η απάντηση ενός μηνύματος. Όσο διασχίζουμε ανάποδα ένα μονοπάτι, αφαιρούμε κόμβους από το Route και προσθέτουμε κόμβους στο PreviousNodes, η ακριβής λειτουργία θα αναλυθεί σε επόμενη ενότητα.

Σε ένα μήνυμα, υπάρχουν και ορισμένα πεδία τα οποία χρησιμοποιούνται για να αλλάξουν τον τρόπο αντιμετώπισης τους:

- **OriginMessage**. Αυτό το πεδίο, άν έχει τιμή true, σηματοδοτεί ότι πρόκειται για ένα μήνυμα μεταξύ RSUs και Origin. Ως εκ τούτου, όλα τα οχήματα που λαμβάνουν αυτό το μήνυμα το αγνοούν. Αντίστοιχα αν είναι false, τότε αγνοείται από το Origin.
- **UpdatePaths**. Θα δούμε και στη συνέχεια, όταν μελετήσουμε την δημιουργία πίνακα δρομολόγησης, ότι ένα μήνυμα που έχει Route, υπόκειται από έλεγχο δρομολόγησης. Αυτή η μεταβλητή αν είναι false, μπορεί να αγνοήσει αυτόν τον έλεγχο.

Η τελευταία κατηγορία πληροφοριών που υπάρχουν σε ένα μήνυμα, είναι οι πληροφορίες που χρησιμοποιούνται για τη μετάδοση περιεχομένου.

- **ContentId**. Χρησιμοποιείται για τον προσδιορισμό της ταυτότητας του περιεχομένου που ζητάμε ή που στέλνουμε.
- **Content**. Σε αυτό το πεδίο αποθηκεύουμε το περιεχόμενο που θέλουμε να στείλουμε.
- **Segments**. Εδώ ορίζουμε πόσα τμήματα έχει το περιεχόμενο που πρόκειται να στείλουμε. Αυτός που ζητάει περιεχόμενο δεν ξέρει πόσα τμήματα έχει εξάρχής.
- **SegmentNumber**. Στην περίπτωση αποστολής πολυτμηματικού περιεχομένου, αυτή η μεταβλητή χρησιμοποιείται για να ορίσουμε ποιό τμήμα στέλνουμε.
- **Multimedia**. Ξανά για τη μετάδοση περιεχομένου, αυτό το πεδίο καθορίζει εάν το περιεχόμενο που ζητάμε ή στέλνουμε είναι πολυμεσικό περιεχόμενο, δηλαδή κάποιο βίντεο ή εικόνα, ή περιεχόμενο που περιέχει κάποια πληροφορία σχετικά με τον δρόμο (όπως για παράδειγμα την ταυτότητα του δρόμου στον οποίο έχει γίνει μία σύγκρουση).

4.2 Περιγραφή δομών που χρησιμοποιήθηκαν

Για τις ανάγκες αυτής της προσομοίωσης, ήταν απαραίτητη η δημιουργία ορισμένων δομών δεδομένων, ώστε να μπορούμε να αποθηκεύουμε καλύτερα τα μηνύματα που λαμβάνει ο κάθε κόμβος.

Η πρώτη εξ αυτών ονομάζεται **MessageData** και έχει παρόμοιες πληροφορίες με αυτές που υπάρχουν σε ένα μήνυμα. Πιο αναλυτικά, περιλαμβάνει τα πεδία SenderAddress, Source, Dest, Type, ContentId, Content, Segments, SegmentNumber και Multimedia ενός μηνύματος. Επίσης περιλαμβάνει τη χρονική στιγμή δημιουργίας (timestamp) του μηνύματος, τη τελευταία φορά που χρησιμοποιήθηκε (lastUsed) το μήνυμα και το πότε έγινε λήψη του μηνύματος (receivedAt) από τον κόμβο που θέλει να το αποθηκεύσει. Τέλος, περιέχει τη

συχνότητα χρήσης του μηνύματος (usedFrequency) και τις προσπάθειες αναμετάδοσης που έχουν γίνει (attempts).

Σε αυτή τη δομή, έχει γίνει και υλοποίηση συναρτήσεων που συγκρίνουν δύο MessageData δεδομένα ανάλογα με διάφορες παραμέτρους.

Η δεύτερη δομή που δημιουργήθηκε, ονομάζεται **ContentWrapper**. Στη δομή αυτή γίνεται αποθήκευση των δεδομένων που υπάρχουν σε ένα μήνυμα. Αυτό σημαίνει ότι η δομή αυτή περιέχει τα πεδία Content, Segments και SegmentNumber ενός μηνύματος.

Η λειτουργία αυτών των δομών θα γίνει εμφανής στη συνέχεια.

4.3 Περιγραφή Αλγορίθμων

Σε αυτή την ενότητα θα γίνει μία αναλυτική περιγραφή των αλγόριθμων που έχουν γραφτεί για κάθε μία από τις λειτουργίες που υπάρχουν σε αυτή την προσομοίωση.

4.3.1 Αλγόριθμος Αποστολής Μηνυμάτων

Ο πιο βασικός αλγόριθμος, είναι αυτός της αποστολής μηνυμάτων. Όταν ένας κόμβος θέλει να στείλει ένα μήνυμα, πρέπει πρώτα να στείλει ένα μήνυμα στον εαυτό του τη χρονική στιγμή που καθορίσει. Άν το State του μηνύματος είναι SENDING, τότε ξεκινάει η διαδικασία αποστολής. Αρχικά, γίνεται έλεγχος των hops του μηνύματος, και αν είναι μικρότερα από το MaxHops, τότε αυξάνουμε την τιμή του πεδίου Hops κατά 1 και στέλνουμε το μήνυμα. Αλλιώς, το μήνυμα διαγράφεται.

4.3.2 Διαχείρηση Απλών Μηνυμάτων

Στις προσομοιώσεις, υπάρχουν δύο βασικές κατηγορίες μηνυμάτων. Τα μηνύματα περιεχομένου και τα απλά μηνύματα. Ως μηνύματα περιεχομένου, ορίζουμε τα μηνύματα, των οποίων το πεδίο Content δεν είναι άδειο. Η διαχείριση καθεμίας από τις δύο κατηγορίες διαφέρει σημαντικά.

Αρχικά, στην περίπτωση των απλά μηνυμάτων, για κάθε μήνυμα που έρχεται καλούνται κάποιες συναρτήσεις, που θα αναλυθούν σε επόμενη ενότητα, και ύπο ορισμένες προϋποθέσεις γίνεται αποθήκευση του μηνύματος σε μία λίστα. Τα μηνύματα δεν αποθηκεύονται αυτούσια, αλλά δημιουργείται ένα MessageData που περιέχει τις πληροφορίες του μηνύματος και αποθηκεύεται αυτό. Για λόγους συντομίας, οι λίστες παρόλα αυτά θα αναφέρονται σαν λίστες μηνυμάτων.

Οι λίστες αποθήκευσης απλών μηνυμάτων είναι δύο: Η storedMessages και η candidateMessages. Οι λίστες αυτές σαν σύνολο έχουν περιορισμένη χωρητικότητα, δηλαδή το σύνολο των μηνυμάτων που βρίσκονται και στις δύο λίστες δεν μπορεί να ξεπερνά ένα συγκεκριμένο νούμερο που ορίζουμε εμείς, ανάλογα με τις ανάγκες του δικτύου.

Στη storedMessages υπάρχουν τα πρόσφατα μηνύματα, ενώ στην candidateMessages μεταφέρονται μηνύματα από την storedMessages, όταν τα μηνύματα αυτά θεωρηθούν πλέον stale, δηλαδή σχετικά παλιά. Η διαδικασία μεταφοράς παλιών μηνυμάτων γίνεται περιοδικά, ανά χρονικά διαστήματα που καθορίζουμε εμείς. Τα δεδομένα που είναι αποθηκευμένα στη storedMessages είναι ταξινομημένα με βάση τον χρόνο δημιουργίας τους (από το πιο

καινούριο στο πιο παλιό). Επομένως η διαδικασία μεταφοράς στην candidateMessages είναι απλή: Ξεκινάμε από το τέλος της λίστας και αν ο χρόνος δημιουργίας ενός μηνύματος θεωρείται παλιός σε σχέση με τη τρέχουσα στιγμή της προσομοίωσης, τότε μεταφέρουμε το μήνυμα και πάμε στο επόμενο. Αν όμως δεν είναι σχετικά παλιός, τότε δεν έχει νόημα να ψάξουμε και την υπόλοιπη λίστα και σταματάμε εκεί.

Στην περίπτωση που προσπαθήσουμε να εισχωρήσουμε νέο μήνυμα όταν έχουμε φτάσει τη μέγιστη χωρητικότητα, πρέπει να εφαρμόσουμε κάποιο αλγόριθμο διαγραφής μηνυμάτων. Ο τρόπος με τον οποίο κρίνουμε ποιά μηνύματα θα διαγράψουμε (Caching Policy), καθώς και το πόσα μηνύματα θα διαγράψουμε, επιλέγονται στην αρχή της προσομοίωσης και παραμένουν σταθερά.

Οι τρόποι διαγραφής που έχουν υλοποιηθεί είνα οι εξής:

- **FIFO ή First In First Out**. Το policy αυτό σημαίνει ότι το μήνυμα που ήρθε πρώτο πρέπει να διαγραφεί πρώτο. Για να γίνει αυτό, ταξινομούμε τη λίστα στην οποία θα γίνει η διαγραφή, με βάση το πεδίο receivedAt του αποθηκευμένου MessageData. Το πεδίο αυτό αντιπροσωπεύει τη χρονική στιγμή λήψης του μηνύματος. Στη συνέχεια διαγράφονται όσα μηνύματα επιθυμούμε, ξεκινόντας από αυτό με τη πιο μικρή στιγμή λήψης.

- **LRU ή Least Recently Used**. Αυτή η πολιτική ασχολείται με το πόσο συχνά χρησιμοποιήθηκε ένα μήνυμα. Κάθε φορά που λαμβάνουμε ένα μήνυμα, ορίζουμε εκείνη τη στιγμή ως τη τελευταία στιγμή που χρησιμοποιήθηκε (πεδίο lastUsed της δομής MessageData) και αν μας ξαναζητηθεί το μήνυμα αυτό, ανανεώνεται η στιγμή χρησιμοποίησης. Για να διαγράψουμε με βάση αυτή τη πολιτική, ταξινουμούμε τη λίστα και διαγράφουμε τον ορισμένο αριθμό μηνυμάτων με το πιο μικρό lastUsed.

- **LFU ή Least Frequently Used**. Η συγκεκριμένη πολιτική καθορίζει ποιά μηνύματα θα διαγραφούν ανάλογα με το πόσο συχνά χρησιμοποιούνται. Η συχνότητα χρήσης μπορεί να βρεθεί στο πεδίο usedFrequency της δομής MessageData και για τη διαγραφή μηνυμάτων, και πάλι, ταξινομούμε τη λίστα από τη πιο μεγάλη στη πιο μικρή συχνότητα χρήσης. Στη συνέχεια διαγράφουμε τον αριθμό μηνυμάτων που θέλουμε να αφαιρέσουμε, που έχουν τη μικρότερη συχνότητα χρήσης.

Ο τρόπος με τον οποίο εφαρμόζεται αυτή η διαδικασία, έχει μερικές υποπεριπτώσεις.

Στην περίπτωση που έχουμε μηνύματα στην candidateMessages και φτάσουμε μέγιστη χωρητικότητα, η αντιμετώπιση είναι απλή. Αν το μέγεθος της candidateMessages είναι μικρότερο από τον επιλεγμένο αριθμό μηνυμάτων που θέλουμε να διαγράψουμε, απλά αδειάζουμε τη λίστα αυτή. Στη περίπτωση που δεν είναι μικρότερο, εφαρμόζουμε το ορισμένο caching policy.

Αν όμως δεν έχει μηνύματα το candidateMessages και δεν υπάρχει άλλος χώρος για αποθήκευση, πλέον οι αλγόριθμοι εφαρμόζονται στο storedMessages. Ξανά, λοιπόν, αν το μέγεθος του storedMessages είναι μικρότερο από τον αριθμό των μηνυμάτων που πρέπει να διαγραφούν, αδειάζουμε το storedMessages. Αξίζει να σημειωθεί ότι η περίπτωση αυτή είναι αρκετά σπάνια. Στην περίπτωση όμως που δεν είναι μικρότερο, εφαρμόζουμε το επιλεγμένο caching policy και στη συνέχεια, για να μη χαλάσουμε τη σειρά των μηνυμάτων, διότι τα

μηνύματα της storedMessages είναι ταξινομημένα, κάνουμε άλλη μία ταξινόμηση σε αυτή, σύμφωνα με το πεδίο timestamp των MessageData δεδομένων που είναι αποθηκευμένα στη λίστα. Έτσι την επαναφέρουμε στην αρχική της κατάσταση.

4.3.3 Διαχείριση Μηνυμάτων Περιεχομένου

Όπως αναφέρθηκε, η αντιμετώπιση μηνυμάτων περιεχομένου διαφέρει σημαντικά από αυτή των απλών μηνυμάτων.

Κάθε φορά που φτάνει σε ένα κόμβο ένα μήνυμα περιεχομένου πάλι καλούνται κάποιες συναρτήσεις που θα δούμε στη συνέχεια, όμως αυτή τη φορά, άν το μήνυμα γίνεται δεκτό και αποθηκευτεί, αποθηκεύεται στη λίστα segmentedMessages. Η λίστα αυτή δημιουργείται με τέτοιο τρόπο ώστε όλα τα τμήματα του ίδιου μηνύματος να βρίσκονται το ένα μετά το άλλο, από το μήνυμα με το πιο μικρό segment number μέχρι το πιο μεγάλο. Επίσης, η λίστα αυτή δεν έχει κάποια μέγιστη χωρητικότητα, οπότε δεν χρειάζεται να εφαρμόσουμε κάποιο caching policy.

Μετά την εισαγωγή του μηνύματος στη λίστα αυτή, καλείται μία συνάρτηση με σκοπό την ανασύσταση του αρχικού μηνύματος, αν είναι δυνατή, από τα επιμέρους τμήματα που υπάρχουν στη λίστα. Είναι σημαντικό να αναφερθεί εδώ, ότι για τον σκοπό αυτής της προσομοίωσης, το περιεχόμενο των μηνυμάτων είναι ένα string και στην περίπτωση πολυτμηματικού περιεχόμενου, κάθε τμήμα περιέχει ένα κομμάτι του string, κάτι που μας βοηθάει πολύ στην ανασύσταση του αρχικού μηνύματος. Πιο συγκεκριμένα, ο τρόπος με τον οποίο γίνεται η διαδικασία αυτή είναι ο εξής:

Από το μήνυμα που λάβαμε (το οποίο είναι μήνυμα περιεχομένου και περιέχει πληροφορίες στα πεδία Content ID, Segments, SegmentNumber και Multimedia), εξάγουμε το από πόσα τμήματα αποτελείται το συνολικό μήνυμα και αν πρόκειται για πολυμεσικό περιεχόμενο ή οχι. Στη συνέχεια ψάχνουμε την segmentedMessages για την πρώτη εμφάνιση μηνύματος με ίδιο Content ID, Segments και Multimedia με αυτό του μηνύματος μας. Η αναζήτηση αυτή γίνεται με τη χρήση ενός iterator, δηλαδή ενός δείκτη, και μόλις βρούμε την πρώτη εμφάνιση, δημιουργούμε άλλον έναν iterator που τον ονομάζουμε end, ο οποίος δείχνει στο ίδιο στοιχείο με τον άλλο. Έπειτα, συνεχίζουμε να διασχίζουμε τη λίστα, και λόγω της δομής της, όλα τα τμήματα του ίδιου μηνύματος πρέπει να είναι διαδοχικά. Για κάθε τμήμα που βρίσκουμε, αθροίζουμε το περιεχόμενο του σε ένα άλλο string, αυξάνουμε έναν μετρητή που είχε αρχικοποιηθεί στο 0, και προχωράμε τον iterator στην επόμενη θέση. Όταν το στοιχείο που δείξει ο iterator δεν έχει τα ίδια χαρακτηριστικά στοιχεία με αυτό που ψάχνουμε η αναζήτησή μας έχει τελειώσει.

Από εδώ και πέρα υπάρχουν δύο σενάρια. Το ένα σενάριο είναι ότι ο μετρητής που είχαμε έχει τιμή μικρότερη από το Segments του μηνύματός μας, κάτι που σημαίνει ότι δεν έχουμε πάρει ακόμα όλα τα τμήματα, και τότε ξανααρχικοποιούμε το string που είχαμε ορίσει και εκεί λήγει η συνάρτηση.

Η άλλη περίπτωση, είναι να έχουμε όλα τα τμήματα που χρειαζόμαστε. Τότε, αυτό που γίνεται είναι: Δημιουργούμε ένα ContentWrapper με τιμή στο πεδίο Segments ίδια με αυτή του αρχικού μηνύματος. Το πεδίο Content του ContentWrapper είναι το ανασυνταγμένο string που φτιάξαμε και εισάγουμε αυτό το μήνυμα σε ένα λεξικό, με κλειδί το ContentId του

αρχικού μηνύματος και τιμή το ContentWrapper. Υπάρχουν δύο λεξικά για την αποθήκευση περιεχομένου. Το ένα είναι το multimediaData, στο οποίο εισάγουμε multimedia δεδομένα και το άλλο είναι το roadData για τα υπόλοιπα δεδομένα. Τέλος, διαγράφουμε όλα τα επιμέρους τμήματα που υπάρχουν ακόμα στη λίστα segmentedContent, αφού πλέον έχουμε ξαναφτιάξει το αρχικό περιεχόμενα.

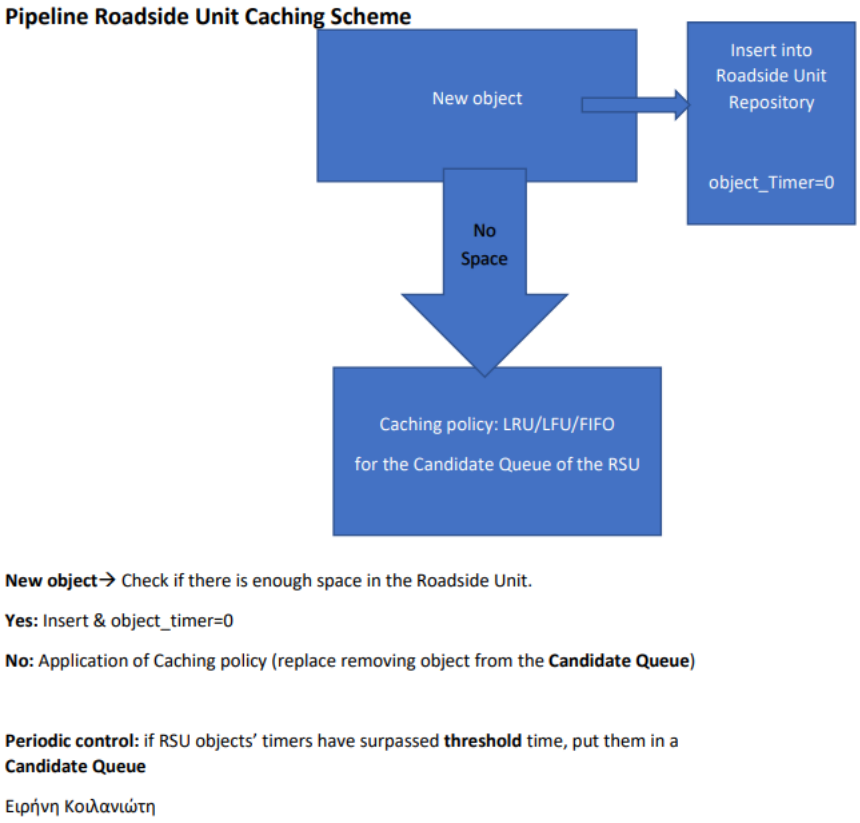
4.3.4 Αλγόριθμοι Αποθήκευσης Απλών Μηνυμάτων

Υπάρχουν δύο συναρτήσεις στον κώδικα που χρησιμοποιούνται για την αποθήκευση μηνυμάτων: μία για μηνύματα περιεχομένου και μία για τα υπόλοιπα. Ξεκινώντας με την εισαγωγή απλών μηνυμάτων, αναφέρθηκε προηγουμένως πως η λίστα storedMessages φτιάχνεται με τρόπο ώστε να είναι ταξινομημένη ανάλογα με το πόσο πρόσφατα είναι τα μηνύματα, δηλαδή με αυξοντα αριθμό δημιουργίας μηνύματος. Η μέθοδος με την οποία επιτυγχάνεται αυτό είναι:

Όταν λάβουμε ένα μήνυμα, αν γίνει αποδεκτό, τότε ξεκινάμε τη διαδικασία εισαγωγής. Ελέγχεται πρώτα αν είναι άδειος ο χώρος storedMessages. Στην περίπτωση που δεν είναι άδειος ξεκινάμε να ανατρέχουμε τα μηνύματα που βρίσκονται εκεί. Αν βρούμε ένα μήνυμα που έχει μεγαλύτερο χρόνο δημιουργίας, δηλαδή είναι πιο πρόσφατο, από το μήνυμα που θέλουμε να εισχωρήσουμε συνεχίζουμε στο επόμενο μήνυμα στη λίστα. Άν βρούμε μήνυμα που έχει ίδιο χρόνο δημιουργίας με το μήνυμα μας, γίνεται έλεγχος. Αν τα δύο μηνύματα ταυτίζονται, δηλαδή έχουν ίδιο χρόνο δημιουργίας, ίδιο δημιουργό μηνύματος και ίδιο τύπο μηνύματος, τότε δεν γίνεται εισχώρηση και τελειώνει η συνάρτηση. Αν δεν ταυτίζονται, προχωράμε στο επόμενο μήνυμα και ελέγχουμε τους χρόνους δημιουργία τους ξανά. Αν το μήνυμα που συγκρίνουμε έχει μικρότερο χρόνο δημιουργίας, είναι δηλαδή πιο παλιό, από το μήνυμά μας, τότε εισάγουμε το μήνυμα μας εκεί και επιστρέφουμε επιτυχή εισαγωγή. Σημειώνεται ότι όταν λέμε ¨εισάγουμε το μήνυμα εκεί¨ επειδή η εισαγωγή γίνεται με τη χρήση iterators, η εισαγωγή γίνεται ακριβώς πριν το πιο παλιό στοιχείο.

Αν γίνει σύγκριση με όλα τα στοιχεία της storedMessages και δεν υπάρξει εισαγωγή σε αυτή (ένα τέτοιο παράδειγμα είναι όλα τα μηνύματα της λίστας να είναι πιο καινούρια από το μήνυμα που θέλουμε να εισχωρήσουμε), τότε ξεκινάμε να ελέγχουμε την λίστα candidateMessages για να δούμε αν υπάρχει μήνυμα εκεί που να ταυτίζεται με το μήνυμα μας. Αν γίνει ταύτιση, η συνάρτηση τελειώνει και επιστρέφει ανεπιτυχή εισαγωγή.

Είτε στην περίπτωση που είναι άδεια η storedMessages ή στην περίπτωση που τελειώσουν οι παραπάνω έλεγχοι χωρίς να γίνει κάποια εισαγωγή, δηλαδή το μήνυμα μας δεν υπάρχει σε καμία από τις δύο λίστες και είναι πιο παλιό από κάθε μήνυμα της storedMessages, τότε απλά το βάζουμε στο τέλος της.



Σχήμα 4.1: *Διαδικασία μεταφοράς στο Candidate Queue*

4.3.5 Αλγόριθμοι Αποθήκευσης Μηνυμάτων Περιεχομένου

Ο αλγόριθμος αποθήκευσης μηνυμάτων περιεχομένου ακολουθεί παρόμοια λογική με τον αντίστοιχο αλγόριθμο για απλά μηνύματα, είναι όπως σε μερικά σημεία πιο απλός. Αυτή τη φορά, η λίστα segmentedContent φτιάχνεται με τρόπο ώστε όλα τα τμήματα του ίδιου μηνύματος να είναι μαζί και με αύξοντα αριθμό τμήματος (SegmentNumber). Αυτό επιτυγχάνεται έτσι:

Για κάθε τμήμα που παίρνουμε, διασχίζουμε τη λίστα. Αν βρούμε κάποιο τμήμα από το ίδιο αρχικό μήνυμα (δηλαδή έχει ίδιο ContentId, Segments και Multimedia), κοιτάμε το SegmentNumber του. Άν έχουμε μικρότερο αριθμό τμήματος, εισάγουμε το μήνυμα μας στη θέση εκείνη, και επιστρέφουμε επιτυχή εισαγωγή. Αν όμως έχουμε ίδιο αριθμό τμήματος, τελειώνει εκεί η συνάρτηση και η εισαγωγή είναι ανεπιτυχής. Αν το μήνυμα μας έχει μεγαλύτερο αριθμό τμήματος, προχωράμε στην επόμενη θέση και ξανακάνουμε σύγκριση. Μπορεί το επόμενο μήνυμα που θα συγκριθούμε να μην είναι από το ίδιο αρχικό μήνυμα με το μήνυμα μας, οπότε σταματάμε την αναζήτηση και το τοποθετούμε εκεί. Αν δεν βρεθεί κανένα άλλο τμήμα του αρχικου μηνύματος, τοποθετούμε το μήνυμά μας στο τέλος της segmentedMessages.

4.3.6 Αλγόριθμος Αποδοχής Μηνυμάτων

Μία σημαντική συνάρτηση, που καλείται κάθε φορά που ένας κόμβος λάβει μήνυμα, είναι η συνάρτηση που ελέγχει αν ένα μήνυμα θα γίνει αποδεκτό ή όχι. Σημειώνεται πως

η συνάρτηση αυτή εκτελεί και άλλες λειτουργίες, όμως κυριότερη είναι αυτή της αποδοχής μηνύματος. Στην περίπτωση που γίνει αποδεκτό το μήνυμα, ο κόμβος συνεχίζει, καλώντας τις αντίστοιχες συναρτήσεις που πρέπει να καλέσει ανάλογα με τον τύπο του μηνύματος (πεδίοType). Είναι σημαντικό να τονίσουμε ότι τα μηνύματα που στέλνει ένας κόμβος στον εαυτό τους δεν επηρεάζονται από αυτή τη συνάρτηση αποδοχής και δεν αποθηκεύονται κάπου.

Τα βήματα που ακολουθούνται για την εκτέλεση του αλγορίθμου αποδοχής είναι:

Αρχικά, κοιτάμε αν ο δημιουργός του μηνύματος που πήραμε είμαστε εμείς. Στην περίπτωση αυτή το μήνυμα απορρίπτεται κατευθείαν, διότι από τη στιγμή που δεν αποθηκεύουμε το μήνυμα μας στη λίστα μηνυμάτων, αν το δεχτούμε θα δημιουργηθούν επιπλοκές. Στη συνέχεια, υπάρχει έλεγχος για το πεδίο Recipient. Δηλαδή αν έχει προσδιοριστεί κάποιος άμεσος παραλήπτης και η διεύθυνση του δεν είναι ίδια με τη διεύθυνση του κόμβου που πήρε το μήνυμα, το μήνυμα αγνοείται. Επίσης, ορίζουμε δύο μεταβλητές, τις insertion και update με αρχική τιμή false.

Η διαδικασία που ακολουθεί μετά, έχει να κάνει με ελάχιστα μονοπάτια και θα αναλυθεί με περισσότερη λεπομέρεια σε επόμενη ενότητα. Σαν πρώτο βήμα, παίρνουμε την τιμή του πεδίου OriginMessage. Αν είναι false και η τελευταία ενημέρωση του πίνακα δρομολόγησης έγινε τουλάχιστον 20 δευτερόλεπτα πριν, γίνεται καθαρισμός του πίνακα. Ο λόγος για αυτή τη διαδικασία θα ξεκαθαριστεί σε αργότερο σημείο.

Έπειτα, ελέγχουμε την τιμή UpdatePaths του μηνύματος. Αν η τιμή της είναι true, εξάγουμε τα Route και PreviousNodes από το μήνυμα. Άν το Route δεν είναι άδειο και έχει μέσα τη διεύθυνση του κόμβου που πήρε το μήνυμα, σημαίνει ότι το μήνυμα έκανε κύκλο, κάτι που δεν θέλουμε σε συναρτήσεις εύρεσης ελάχιστου μονοπατιού, οπότε απορρίπουμε το μήνυμα. Μετά, καλούμε τη συνάρτηση που ενημερώνει τα ελάχιστα μονοπάτια, που θα περιγραφεί αργότερα. Πρώτα καλείται για το Route, αν δεν είναι άδειο, και αν επιστρέψει false, το μήνυμα απορρίπτεται. Αλλιώς, κάνουμε τη μεταβλητή update ίση με true. Η ίδια διαδικασία ακολουθείται και για το PreviousNodes με την προϋπόθεση ότι και αυτό δεν είναι άδειο.

Επόμενο βήμα είναι ο έλεγχος της χωρητικότητας απλών μηνυμάτων. Αν ο συνολικός αριθμός μηνυμάτων στις storedMessages και candidateMessages είναι ίσος με ή ξεπερνά τη μέγιστη χωρητικότητα, γίνεται κλήση της συνάρτησης διαγραφής μηνυμάτων.

Τέλος, ανάλογα με αν πρόκειται για μήνυμα περιεχομένου ή όχι, καλείται η αντίστοιχη συνάρτηση εισαγωγής, της οποίας το αποτέλεσμα (ψευδές ή αληθές) εισχωρείται στη μεταβλητή insertion. Ένα μήνυμα γίνεται, λοιπόν, αποδεκτό με την προϋπόθεση ότι μία από τις μεταβλητές update, insertion είναι αληθής.

Αυτό γίνεται διότι δύο μηνύματα με διαφορετικό Route μπορεί να έχουν ίδια άλλα στοιχεία, οπότε να μη γίνουν δεκτά από τη συνάρτηση εισαγωγής. Όμως μπορεί να έχουν καλύτερο μονοπάτι από αυτό που υπάρχει ήδη, οπότε η μεταβλητή update θα είναι αληθής και το μήνυμα θέλουμε να γίνει αποδεκτό. Αντίστοιχα, η μεταβλητή update ορίζεται από την αρχή σαν ψευδής, και στην περίπτωση που είτε τα Route, PreviousNodes είναι κενά, ή το UpdatePaths είναι ψευδές, θα παραμείνει έτσι μέχρι το τέλος της συνάρτησης, όμως μπορεί να είναι μήνυμα που να πρέπει να εισχωρηθεί στη λίστα μηνυμάτων.

4.3.7 Αλγόριθμοι και Διαδικασία Εύρεσης Ελαχίστων Μονοπατιών

Μία πολύ βασική διαδικασία που υλοποιήθηκε για το σκοπό της παρούσας διπλωματικής εργασίας είναι η διαδικασία εύρεσης ελάχιστων μονοπατιών.

Η διαδικασία ξεκινάει με τη δημιουργία ενός μηνύματος τύπου ¨αίτημα δρομολόγησης¨ (Route Request). Το μήνυμα αυτό, ιδανικά έχει MaxHops ίσο με τον συνολικό αριθμό κόμβων στο δίκτυο, ώστε να φτάσει όλους τους κόμβους. Ακόμα, το PreviousNodes του είναι άδειο, ενώ στο Route του, έχει μόνο τη δική του διεύθυνση. Όταν ένας κόμβος δέχεται ένα αίτημα δρομολόγησης, ο αλγόριθμος που ακολουθεί είναι ο εξής:

Αρχικά, παίρνει τις πληροφορίες Route και PreviousNodes από το αίτημα. Στη συνέχεια, στέλνει μία απάντηση δρομολόγησης (Route Reply) με recipient την τελευταία θέση του Route και προορισμό τον δημιουργό του αιτήματος. Το Route της απάντησης είναι ίδιο με του αιτήματος, αν αφαιρέσουμε την τελευταία θέση του (pop_back), ενώ στο PreviousNodes έχει προστεθεί η διεύθυνση του κόμβου που στέλνει την απάντηση δρομολόγησης. Ο λόγος για αυτή τη προσθήκη είναι για να μπορέσουμε να ανιχνεύσουμε το δρόμο που έχει ακολουθήσει η απάντηση μέχρι τώρα.

Την ίδια στιγμή, προωθείται το αίτημα δρομολόγησης, ώστε να το λάβουν και άλλοι κόμβοι του δικτύου, προσθέτοντας στο τέλος του Route και τη δική του διεύθυνση. Με τον τρόπο αυτό, δημιουργούνται όλα τα πιθανά μονοπάτια από τον κόμβο που έκανε το αίτημα δρομολόγησης πρός κάθε κόμβο του δικτύου.

Αντίστοιχα, πρέπει να προσδιορίσουμε τον βασικό αλγόριθμο που χρησιμοποιείται όταν ένας κόμβος λαμβάνει μία απάντηση δρομολόγησης. Αν ένας κόμβος λάβει απάντηση δρομολόγησης και είναι ο recipient που καθορίζεται από το μήνυμα, υπάρχουν δύο πιθανά σενάρια. Αν η διεύθυνση του παραλήπτη είναι ίδια με τον προορισμό που αναγράφεται στο πεδίο dest του μηνύματος, τότε το μήνυμα έφτασε στον προορισμό του και τελειώνει εκεί η διαδικασία. Αλλιώς, ο παραλήπτης αφαιρεί την τελευταία θέση του Route και τη βάζει στο πεδίο recipient, ώστε να συνεχίσουμε να διασχίζουμε τη διαδρομή μέχρι να φτάσουμε στον κόμβο στον οποίο προορίζεται η απάντηση, ενώ ταυτόχρονα προσθέτει τη διεύθυνση του στο PreviousNodes.

Αυτή η διαδικασία από μόνη της, όμως, δεν ενημερώνει κάπως τα ελάχιστα μονοπάτια που βρίσκονται. Ως εκ τούτου, υπάρχει μία άλλη συνάρτηση, που ενημερώνει τα ελάχιστα μονοπάτια και καλείται κάθε φορά που ένας κόμβος λάβει ένα μήνυμα που έχει UpdatePaths με τιμή true και που υπάρχει κάποιο μονοπάτι είτε στο Route ή στο PreviousNodes του μηνύματος. Αρχικά, ως πίνακα δρομολόγησης κάποιου κόμβου A, ορίζουμε ένα λεξικό, του οποίου τα κλειδιά είναι διευθύνσεις άλλων κόμβων και οι τιμές του είναι τα ελάχιστα μονοπάτια από τον κόμβο A πρός τον κόμβο που βρίσκεται στο κλειδί. Ο αλγόριθμος της συνάρτησης ενημέρωσης είναι ο εξής:

Παίρνουμε το μονοπάτι που μελετάμε, και παίρνουμε την διεύθυνση που υπάρχει στη τελευταία θέση του μονοπατιού, έστω κόμβος B. Στη συνέχεια ψάχνουμε αν αυτή η διεύθυνση υπάρχει στον πίνακα δρομολόγησης μας. Αν υπάρχει, συγκρίνουμε το μήκος της διαδρομής που είναι αποθηκευμένη, με το μήκος της διαδρομής που υπάρχει στο μονοπάτι που μελετάμε, μετρώντας από το τέλος του μονοπατιού μέχρι τη θέση στην οποία υπάρχει η διεύθυνση του κόμβου B. Στην περίπτωση που το μήκος της διαδρομής που έχουμε αποθη-

κεύσει στον πίνακα δρομολόγησης είναι μικρότερο από το μήκος της διαδρομής που υπάρχει στο μονοπάτι του μηνύματος, η συνάρτηση λήγει και επιστρέφει ανεπιτυχή ενημέρωση. Αν δεν υπάρχει ή αν η διαδρομή που έχουμε αποθηκευμένη στον πίνακα δρομολόγησης είναι μεγαλύτερη ή ίση από αυτή του μηνύματος, απλά ορίζουμε ως ελάχιστη απόσταση πρός τον κόμβο B, το τμήμα από την τελευταία θέση του μονοπατιού μέχρι το B. Επαναλαμβάνουμε αυτή τη διαδικασία, μέχρι να φτάσουμε στην αρχή του μονοπατιού. Για κάθε ενημέρωση του πίνακα δρομολόγησης, θέτουμε μία μεταβλητή που έχει κάθε κόμβος του δικτύου με τιμή ίση με τη χρονική στιγμή της ενημέρωσης.

Αξίζει εδώ να σημειώσουμε ότι οι διαδρομές αυτές, για τα οχήματα δεν είναι σταθερές. Λόγω της φύσης των VANETs, η τοπολογία του δικτύου μεταβάλλεται πολύ γρήγορα, και για αυτό το λόγο όταν έρχεται μήνυμα, αν έχουμε δεδομένα στον πίνακα που είναι παλιά, τον αδειάζουμε. Παρόλα αυτά, οι ελάχιστες διαδρομές μεταξύ RSUs και του Origin είναι σταθερές. Για το λόγο αυτό, υπάρχει ένας ξεχωριστός πίνακας δρομολόγησης για τις διαδρομές αυτές. Στην αρχή κάθε προσομοίωσης, το Origin στέλνει ένα ειδικό αίτημα δρομολόγησης στα RSU και αυτά με τη σειρά τους στέλνουν άλλα ειδικά αιτήματα δρομολόγησης. Στο τέλος αυτής της διαδικασίας, και τα RSUs και το Origin ξέρουν όλες τις βέλτιστες διαδρομές μεταξύ τους, και αυτές οι διαδρομές χρησιμοποιούνται για τη γρήγορη αποστολή μηνυμάτων από το ένα στο άλλο.

4.3.8 Διαδικασία Υπολογισμού Κεντρικοτήτων

Επίσης, είναι σημαντικό να περιγράψουμε τη διαδικασία με την οποία εκτελείται ο υπολογισμός κεντρικοτήτων. Αρχικά, ανά συγκεκριμένα χρονικά διαστήματα, ο κόμβος Origin, στέλνει στα RSUs ένα αίτημα εκκίνησης υπολογισμού. Με τη σειρά τους, τα RSUs προωθούν το αίτημα αυτό στα άλλα RSUs και στέλνουν ταυτόχρονα αίτημα υπολογισμού κεντρικότητας στα RSUs και οχήματα του δικτύου. Ακόμα, προγραμματίζουν την αποστολή ενός μηνύματος στον εαυτό τους σε 10 δευτερόλεπτα από τη στιγμή λήψης του αιτήματος εκκίνησης υπολογισμού. Εδώ σημειώνεται ότι η τιμή του πεδίου Dest είναι ίδια με τη διεύθυνση του κόμβου που κάνει τι αίτημα, η χρήση του οποίου θα εξηγηθεί αργότερα. Ακόμη, η τιμή του πεδίου State του μηνύματος είναι COLLECTING. Μηνύματα με τέτοιο State χρησιμοποιούνται για τον τελικό υπολογισμό των δεδομένων που πήραμε από άλλους κόμβους κατά το αίτημα και την αποστολή αυτού του υπολογισμού στο Origin. Επιπλέον, στο πεδίο Centrality του αιτήματος εκκίνησης περιέχεται το είδος της κεντρικότητας που θέλουμε υπολογιστεί. Οι κεντρικότητες που έχουν υλοποιηθεί είναι: Βαθμού, Εγγύτητας και Ενδιαμεσικότητας.

Στην περίπτωση αιτήματος υπολογισμού κεντρικότητας βαθμού, το RSU που κάνει το αίτημα, θέτει τον μέγιστο αριθμό hops του μηνύματος σε 1. Με τον τρόπο αυτό, εξασφαλίζουμε ότι το αίτημα θα φτάσει μόνο στους άμεσους γείτονες του κόμβου. Οι γειτονικοί κόμβοι με τη σειρά τους στέλνουν μία απάντηση κεντρικότητας βαθμού και για κάθε απάντηση που λάβει ο κόμβος που έκανε το αρχικό αίτημα, αυξάνει ένα μετρητή κατά 1. Όταν έρθει η ώρα για συλλογή, στέλνουμε την τιμή του μετρητή στο Origin.

Ένα αίτημα υπολογισμού κεντρικότητας βαθμού ταυτίζεται με ένα αίτημα δρομολόγησης και η διαδικασία έχει περιγραφεί παραπάνω. Τη στιγμή της συλλογής, το RSU που έκανε το αίτημα δρομολόγησης αθροίζει το μήκος όλων των μονοπατιών που υπάρχουν στον πίνακα

Αλγοριθμος 4.1: *Degree Centrality Calculation*

degree ← 0
RSU sends request with maxHops = 1
Other nodes receive request and answer
degree ← number of answers received
RSU sends that number to Origin

δρομολόγησης του. Στη συνέχεια, το αποτέλεσμα που στέλνει στο Origin είναι ίσο με το μήκος του πίνακα δρομολόγησης διαιρεμένο με το άθροισμα αυτό.

Αλγοριθμος 4.2: *Closeness Centrality Calculation*

closeness ← 0
RSU sends request with maxHops ≥ number of nodes in the network
Other nodes receive request and answer with the path traversed so far
Nodes that receive the request, also forward it to other nodes, updating path traversed
if path traversed is worse than the path in the routing table **then**
    discard message
else
    update routing table with new shortest path
end if
After a few seconds, *closeness* ← (sum of all shortest paths)/(size of routing table)
RSU sends that number to Origin

Για τα αιτήματα υπολογισμού κεντρικότητας ενδιαμεσικότητας η διαδικασία είναι λίγο πιο περίπλοκη. Αρχικά όποιος κόμβος λάβει αίτημα υπολογισμού κεντρικότητας ενδιαμεσικότητας, προωθεί το αίτημα, ώστε να φτάσει όλους τους κόμβους του δικτύου, και στη συνέχεια δημιουργεί ένα αίτημα δρομολόγησης. Με τον τρόπο αυτό, κάθε κόμβος του δικτύου υπολογίζει τον πίνακα δρομολόγησης του. Ταυτόχρονα, όποιος πάρει αίτημα υπολογισμού κεντρικότητας ενδιαμεσικότητας, καλεί τη συλλογή κάποια δευτερόλεπτα αργότερα. Αυτή τη φορά, όμως, το πεδίο Dest του μηνύματος για τη συλλογή δεν είναι η διεύθυνση του ίδιου του κόμβου που την καλεί, αλλά του κόμβου που έκανε το αίτημα για τον υπολογισμό της κεντρικότητας. Η λειτουργία αυτού είναι διπλή. Πρώτον, με τον τρόπο αυτό ο κόμβος ξέρει ποιό RSU έκανε το αίτημα για να του στείλει τα αποτελέσματα. Δεύτερον, τα RSUs είναι κόμβοι που μπορούν και να λάβουν αιτήματα υπολογισμού διαμεσικότητας και να κάνουν οι ίδιοι τέτοια αιτήματα, οπότε, με τον τρόπο αυτό, όταν καλείται η συλλογή, αν η διεύθυνση τους είναι διαφορετική από τη διεύθυνση dest του μηνύματος συλλογής, η συνάρτηση συμπεριφέρεται διαφορετικά. Ο τρόπος με τον οποίο γίνεται η συλλογή στα οχήματα και στα RSUs αν δεν συλλέγουν για αίτημα υπολογισμού που έκαναν οι ίδιοι είναι απλός. Διατρέχουμε τον πίνακα δρομολόγησης, και για κάθε ελάχιστο μονοπάτι που περιέχει τον κόμβο που αναγράφεται στο πεδίο Dest του μηνύματος συλλογής, προσθέτουμε 1 σε ένα μετρητή. Στο τέλος αυτής της διαδικασίας στέλνουμε μήνυμα με αυτό το αποτέλεσμα στον κόμβο του Dest. Ο κόμβος του Dest, κάθε φορά που παίρνει μία τέτοια απάντηση στο αίτημα του, αυξάνει τη τιμή ενός μετρητή κατά την τιμή που έχει το μήνυμα και στη συλλογή, που γίνεται μετά από μερικά δευτερόλεπτα, στέλνει τον μετρητή αυτό στο Origin, ενώ ταυτόχρονα καθαρίζει

τον πίνακα δρομολόγησης του.

Αλγοριθμος 4.3: *Betweenness Centrality Calculation*

betweenness ← 0
RSU sends request with maxHops ≥ number of nodes in the network
Other nodes receive request and begin calculating the shortest paths to other nodes
Nodes that receive the request, also forward it to other nodes
When a node has calculated their routing table:
result ← number of times requested RSU appears in a shortest path
Send result to the RSU
betweenness ← betweenness + result
After some time, to make sure all results have come in, RSU sends that number to Origin

Όταν τα αποτελέσματα των κεντρικοτήτων έρθουν στο Origin, το Origin βρίσκει το μεγαλύτερο εξ αυτών και στέλνει ένα μήνυμα στα RSUs, ενημερώνοντας τα για το πιο ¨σημαντικό¨ RSU.

4.3.9 Διαδικασία Μετάδοσης Περιεχομένου

Η μετάδοση περιεχομένου, είναι μία λειτουργία απαραίτητη σε κάθε VANET δίκτυο. Για το λόγο αυτό, και στον κώδικα υπάρχουν συναρτήσεις που υλοποιούν τη λειτουργία αυτή.

Ο αλγόριθμος που χρησιμοποιείται για τη μετάδοση περιεχομένου είναι σχετικά απλός. Ο κόμβος που στέλνει το περιεχόμενο έχει γνώση από πριν τον αριθμό των τμημάτων που θα χρειαστεί να το διασπάσει (στα πλαίσια της προσομοίωσης έχει γίνει η παραδοχή ότι αν ένα μήνυμα χρειάζεται διάσπαση, το μήκος του string που περιέχει το περιεχόμενό του, είναι ίσο με τον αριθμό των τμημάτων). Με τη γνώση αυτή, ξεκινάει δημιουργώντας ένα μήνυμα για κάθε τμήμα, ορίζοντας σε αυτό το κατάλληλο SegmentNumber και συμπληρώνοντας το ContentId και το Multimedia. Για το περιεχόμενο, το σπάει ομοιόμορφα σε όσα τμήματα χρειάζεται και βάζει ένα τμήμα σε κάθε μήνυμα που στέλνει. Για παράδειγμα, αν το περιεχόμενο είναι το "test" και έχουμε στο πεδίο Segments την τιμή 4, τότε στέλνουμε 4 μηνύματα, με content: "t", "e", "s" και "t" αντίστοιχα. Προφανώς, αν δεν χρειάζεται τεμάχιση το μήνυμα, απλά στέλνεται αυτούσιο.

Ένα όχημα μπορεί είτε να στείλει πληροφορίες ή να ζητήσει πληροφορίες από το δίκτυο. Η αποστολή πληροφοριών (για παράδειγμα ένα ατύχημα) στέλνεται στο δίκτυο όπως αναλύθηκε προηγουμένως.

Γνωρίζουμε ότι το Origin περιέχει αποθηκευμένες όλες τις πληροφορίες που υπάρχουν στην προσομοίωση. Ως εκ τούτου, οι πληροφορίες που λαμβάνουν τα RSUs από τα οχήματα πρέπει κάπως να μεταβιβάζονται στο Origin, κάτι που επιτυγχάνεται με τη χρήση μηνυμάτων προώθησης περιεχομένου (Content push) με τον τρόπο που περιγράφηκε παραπάνω. Ένα τέτοιο μήνυμα αν ληφθεί από άλλο RSU, το προωθεί, ακολουθώντας το βέλτιστο μονοπάτι πρός το Origin, μέχρι να φτάσει σε αυτό.

Στην περίπτωση, λοιπόν, που ένα όχημα ζητήσει πληροφορίες, τα οχήματα που παίρνουν το αίτημα, ελέγχουν αν έχουν τις πληροφορίες που ζητήθηκαν και αν ναι τις στέλνουν. Ένα RSU που λάβει τέτοιο αίτημα, αντίστοιχα αν έχει τις πληροφορίες τις στέλνει, αλλά στην

περίπτωση που δεν τις έχει τότε ξεκινάει μία διαδικασία απόκτησης (fetching) περιεχομένου είτε από τα άλλα RSUs ή από το Origin. Πιο συγκεκριμένα, πρώτα ζητάει το περιεχόμενο από το πιο ¨σημαντικό¨ RSU. Η σημαντικότητα ενός RSU μπορεί είτε να έχει μετρηθεί με τη χρήση μετρικών κεντρικότητας, ή με τη χρήση ενός αλγορίθμου μηχανικής μάθησης που θα αναλυθεί σε επόμενη υποενότητα. Αν αυτό το ¨σημαντικό¨ RSU έχει το περιεχόμενο που ψάχνουμε, το στέλνει σε αυτό που έκανε την αίτηση απόκτησης περιεχομένου. Αν όμως δεν το έχει, τότε το ¨σημαντικό¨ RSU κάνει αίτηση στο Origin. Η αίτηση αυτή, γίνεται με ένα μήνυμα του οποίου στο πεδίο Source περιέχεται η διεύθυνση του αρχικού RSU που έκανε την αίτηση απόκτησης και όχι του πιο σημαντικού RSU. Ο λόγος που γίνεται αυτό είναι, έτσι ώστε να στείλει το Origin το περιεχόμενο κατευθείαν στο RSU που ενδιαφέρεται χωρίς επιπλέον καθυστέρηση. Όταν το RSU που έκανε την αίτηση πάρει το περιεχόμενο, τότε αυτό με τη σειρά του στέλνει το περιεχόμενο στο όχημα που το ζήτησε.

Εδώ, πρέπει να αναλύσουμε τις πολιτικές κατανομής περιεχομένου που χρησιμοποιούνται. Οι πολιτικές κατανομής περιεχομένου είναι δύο: το Push και το Pull. Με την πολιτική Push, ανα διαστήματα, το Origin στέλνει ό,τι πληροφορία έχει στο πιο ¨σημαντικό¨ RSU. Στην πολιτική Pull, περιεχόμενο στέλνεται μόνο αν ζητηθεί από το Origin. Το Origin δεν χρειάζεται να στείλει όλες τις πληροφορίες που έχει αποθηκευμένες, παρά μόνο αυτές που του ζητήθηκαν. Μπορούμε να παρατηρήσουμε πως η πολιτική Push είναι ¨προληπτική¨ (proactive), ενώ η Pull είναι ¨αντιδραστική¨ (reactive). Αυτό σημαίνει ότι η πολιτική Pull επιβαρύνει περισσότερο το δίκτυο, αφού μεταφέρει περιεχόμενο που μπορεί να μην χρειαστεί καν, όμως έχει γενικά καλύτερο μέσο χρόνο απόκρισης για αιτήματα περιεχομένου από κόμβους του δικτύου.

Αλγοριθμος 4.4: *Finding most central RSU*

highestCentrality ← 0
centralRSUs is empty
Origin sends centrality calculation request
RSUs calculate their centrality and send results to Origin
if result > highestCentrality **then**
    highestCentrality ← result
    centralRSUs add RSU that send the result
else if result = highestCentrality **then**
    centralRSUs add RSU that send the result
else
    Discard the message
end if
After some time, Origin sends a message to all RSUs, specifying the most central RSUs

4.3.10 Αλγόριθμοι Επιβεβαίωσης Μετάδοσης

Για τα μηνύματα περιεχομένου, είναι σημαντικό να υπάρχει τρόπος ώστε να γνωρίζει ο κόμβος που στέλνει το περιεχόμενο ότι τα μηνύματά του έφτασαν με επιτυχία στον προορισμό τους. Έτσι, έπρεπε να γραφτεί ένας αλγόριθμος που να εκτελέι αυτή τη λειτουργία .

Όταν ένας κόμβος θέλει να στείλει ένα μήνυμα περιεχομένου, γίνεται η εξής διαδικασία:

Αλγοριθμος 4.5: *Push Policy*

When the above algorithm is executed:
if policy is Push **then**
    Origin sends every piece of content it has saved to the most central RSU(s)
end if
if RSU needs content **then**
    It asks the most central RSU closest to them
    The most central RSU has the content and replies
end if

Αλγοριθμος 4.6: *Pull Policy*

When the above algorithm is executed:
if policy is Pull **then**
    Origin does not send the content it has saved to the most central RSU(s)
end if
if RSU needs content **then**
    It asks the most central RSU closest to them
    if Most central RSU has content **then**
        The most central RSU replies
    else
        Most central RSU asks Origin for content for requesting RSU
        Origin sends content to requesting RSU
    end if
end if

Αρχικά, εισάγουμε σε ένα vector που θα ονομάσουμε pendingAck, το μήνυμα περιεχομένου, σε μορφή MessageData, που πρόκειται να στείλουμε. Στέλνουμε, έπειτα, ένα self-message μετά από 5 δευτερόλεπτα, το οποίο έχει στο πεδίο State την τιμή REPEATING και στα πεδία ContentId, Segments, SegmentId και Multimedia τις τιμές του μηνύματος περιεχομένου που πρόκειται να στείλουμε. Όταν παίρνει ένας κόμβος ένα αυτο-μήνυμα με αυτή τη τιμή στο πεδίο State, ψάχνει το pendingAck και αν βρεί εκεί μήνυμα που να έχει τα ίδια χαρακτηριστικά περιεχομένου με αυτά του αυτο-μηνύματος, το ξαναστέλνει και ξαναπρογραμματίζει το ίδιο self-message 5 δευτερόλεπτα αργότερα. Κάθε φορά που ξαναστέλνει κάποιο μήνυμα του pendingAck, αυξάνει το πεδίο attempts κατά 1. Όταν ξεπεράσει το όριο που έχουμε θέσει, διαγράφεται αυτόματα το μήνυμα αυτό.

Ταυτόχρονα, όταν ένας κόμβος λάβει κάποιο μήνυμα περιεχομένου, στέλνει μήνυμα επιβεβαίωσης με τα χαρακτηριστικά του μηνύματος αυτού. Όταν ο κόμβος που έστειλε το περιεχόμενο λάβει κάποιο μήνυμα επιβεβαίωσης, διαγράφει από τον πίνακα pendingAck το μήνυμα περιεχομένου που περιγράφει η επιβεβαίωση.

4.3.11 Αλγόριθμοι Μηχανικής Μάθησης

Βασικό αντικείμενο έρευνας της διπλωματικής αυτής, είναι η χρήση αλγορίθμων μηχανικής μάθησης για τον προσδιορισμό των πιο ¨σημαντικών¨ RSUs. Βασικό πλεονέκτημα αυτής της μεθόδου, είναι το γεγονός ότι σε αντίθεση με τον υπολογισμό κεντρικοτήτων, δεν επιβαρύνει το δίκτυο, καθώς δεν χρειάζεται η ανίχνευση διαδρομών, όπως στην περίπτωση του Closeness και Betweenness.

Οι δύο αλγόριθμοι μηχανικής μάθησης που χρησιμοποιούνται είναι αλγόριθμοι Clustering, κάτι που σημαίνει ότι κατατάσσουν τα οχήματα που υπάρχουν στο δίκτυο σε κάποια συστάδα. Τα δεδομένα των οχημάτων που δίνουμε στον αλγόριθμο μηχανικής μάθησης είναι 4: Η θέση στον άξονα x, η θέση στον άξονα y, η ταχύτητα και η κατεύθυνση. Ο κάθε αλγόριθμος παίρνει τα δεδομένα αυτά και δημιουργεί 3 συστάδες. Οι αλγόριθμοι που χρησιμοποιούνται στην παρούσα διπλωματική είναι ο K-Means και ο Agglomerative Clustering, και βρίσκονται στη βιβλιοθήκη scikit-learn της Python. Ο κάθε ένας από αυτούς έχει άλλα κριτήρια για να αποφασίσει σε ποιά συστάδα κατατάσσεται το κάθε όχημα. Για τον K-means, η διαδικασία συσταδοποίησης έχει αναλυθεί στο Θεωρητικό Μέρος. Όσον αφορά το Agglomerative Clustering, ο αλγόριθμος αυτός ανήκει στην κατηγορία αλγορίθμων ιεραρχικής συσταδοποίησης (Hierarchical Clustering). Αναλυτικότερα, ξεκινάμε με n συστάδες, μία για κάθε σημείο που υπάρχει στο σύνολο δεδομένων μας. Μετά, ανάλογα με τις παραμέτρους που έχουν οριστεί, ενώνουμε διαδοχικά συστάδες στοιχείων με τη μικρότερη απόσταση, σχηματίζοντας σταδιακά ένα δέντρο από κάτω πρός τα πάνω. Βασική διαφορά του από τον K-Means είναι ότι η απόσταση μεταξύ δύο σημείων που υπολογίζεται από τον K-Means είναι πάντα η ευκλείδεια απόστασή τους, κάτι που δεν ισχύει στον Agglomerative, όπου μπορούμε εμείς από πριν να επιλέξουμε τι μετρική θα χρησιμοποιηθεί.

Αρχικά, για να εφαρμόσουμε τον αλγόριθμο μηχανικής μάθησης που θέλουμε, πρέπει να εξάγουμε από την προσομοίωση τα χαρακτηριστικά των οχημάτων που μας ενδιαφέρουν. Για να γίνει αυτό, ένα όχημα που είναι συνδεδεμένο με το TraCIMobility του SUMO, αναλαμβάνει να γράψει σε ένα .csv αρχείο με όνομα vehicles.csv τα χαρακτηριστικά όλων των

οχημάτων που βρίσκονται στο δίκτυο, με τη μορφή (PosX, PosY, Speed, Direction) ανα ορισμένα χρονικά διαστήματα. Όταν γίνει η εγγραφή, σταματάμε την προσομοίωση, εκτελούμε τον αλγόριθμο μηχανικής μάθησης, τρέχοντας το αρχείο KMeansML.py ή το AgglomML.py, ανάλογα με τον αλγόριθμο που επιθυμούμε. Ο αλγόριθμος, με τη σειρά, γράφει τα αποτελέσματα του στο cluster_centers.csv, ενώ διαγράφει το vehicles.csv. Ο λόγος για τη διαγραφή αυτή, είναι έτσι ώστε στην επόμενη φορά που γίνεται η ίδια διαδικασία να μην υπάρχουν παλιά δεδομένα. Τα αποτελέσματα που γράφουν οι δύο αλγόριθμοι μηχανικής μάθησης έχουν τη μορφή μίας λίστας, της οποίας η κάθε θέση περιέχει ένα ζευγάρι συντεταγμένων x και y που αντιστοιχούν στο κέντρο κάθε συστάδας που δημιουργήθηκε. Μόλις τελειώσει ο αλγόριθμος μηχανικής μάθησης, ξανασυνεχίζουμε την προσομοίωση. Το Origin ελέγχει ανα διαστήματα αν υπάρχει το αρχείο cluster_centers.csv και αν υπάρχει εξάγει τα δεδομένα. Έπειτα, συγκρίνει τις συντεταγμένες του κάθε cluster με τις συντεταγμένες των RSU που υπάρχουν στο δίκτυο, και τα RSU με τη μικρότερη απόσταση από το κάθε cluster είναι τα πιο κεντρικά. Μόλις γίνει αυτός ο υπολογισμός των πιο σημαντικών RSUs, ενημερώνεται το δίκτυο με την ίδια διαδικασία που περιγράφηκε στο υποκεφάλαιο υπολογισμού κεντρικότητας.

Στη συνέχεια θα παρουσιαστούν οι ψευδοκώδικες για τους δύο αλγορίθμους μηχανικής μάθησης:

Αλγοριθμος 4.7: *K-Means*

The algorithm reads the vehicles.csv file
A dataframe is created with the columns Position X, Position Y, Speed, Direction from the csv file
Define the model variables
The data from the dataframe is fitted to the K-Means model
The model clusters the data based on the four-dimensional distance between each object
The centroids of the clusters are extracted
These centroids are put in the new dataframe *result*
The results are written on the new cluster_centers.csv file

Αλγοριθμος 4.8: *Agglomerative Clustering*

The algorithm reads the vehicles.csv file
A dataframe is created with the columns Position X, Position Y, Speed, Direction from the csv file
Define the model variables
metric ← cosine
linkage ← average
The data from the dataframe is fitted to the Agglomerate Clustering model
Each node from the original dataframe is now labeled
We calculate each centroid by finding the average coordinates of the vehicles in each cluster
The results are written on the new cluster_centers.csv file

Κεφάλαιο 5

Εγκατάσταση και Εκτέλεση Προσομοίωσης

Το συγκεκριμένο κεφάλαιο περιγράφει τα απαραίτητα βήματα που χρειάζεται να ακολουθήσει κανείς για να τρέξει την προσομοίωση.

5.1 Εγκατάσταση Προαπαιτούμενων

Ξεκινάμε κάνοντας εγκατάσταση των προαπαιτούμενων προγραμμάτων, που περιγράφηκαν στο Κεφάλαιο 3. Ο οδηγός εγκατάστασης που ακολουθήθηκε για αυτό το σκοπό βρίσκεται στη σελίδα του Veins [52]. Η εγκατάσταση έγινε σε περιβάλλον Windows 64-bit.

Αρχικά, κατεβάζουμε τα αρχεία του SUMO με την έκδοση 1.8 από τον σύνδεσμο που δίνεται [53] και τα κάνουμε extract στην τοποθεσία που θα χρησιμοποιήσουμε για το πρότζεκτ. Πηγαίνουμε στον φάκελο που προκύπτει, και ανοίγουμε τον υποφάκελο bin. Εκεί, ελέγχουμε αν περιέχεται το αρχείο sumo.exe που θα χρειαστούμε.

Συνεχίζοντας με την εγκατάσταση του OMNET++, κατεβάζουμε την έκδοση 5.6.2, για το λειτουργικό που χρησιμοποιήθηκε, από την επίσημη σελίδα του [54]. Κάνουμε ξανά αποσυμπίεση του αρχείου στον ίδιο φάκελο που κάναμε αποσυμπίεση και το SUMO και πηγαίνουμε στον φάκελο που δημιουργήθηκε. Εκεί, υπάρχει το αρχείο mingwenv.cmd, το οποίο εκτελούμε. Έπειτα, γράφουμε την εντολή ./configure και μετά make. Αν προκύψει κάποιο πρόβλημα κατά οποιαδήποτε από τις δύο αυτές εντολές επειδή λείπει κάποιο πακέτο, φροντίζουμε να εγκαταστήσουμε το πακέτο αυτό (θα αναγράφεται μαζί με το μήνυμα του error). Όταν τελειώσει η διαδικασία αυτή, γράφουμε την εντολή omnetpp και ανοίγει το IDE του.

Συνεχίζοντας την εγκατάσταση των προαπαιτούμενων, κατεβάζουμε τα αρχεία του Veins 5.2 από την επίσημη σελίδα [55] και τα κάνουμε unpack, ξανά, στο φάκελο που υπάρχουν και τα άλλα δύο προγράμματα. Φροντίζουμε να μην κατεβάσουμε την έκδοση Instant Veins. Την ίδια στιγμή, κατεβάζουμε τα αρχεία που γράφτηκαν στη διπλωματική, και τα τοποθετούμε στον φάκελο του Veins, πατώντας να γίνει αντικατάσταση των παλιών αρχείων με τα καινούρια αρχεία της διπλωματικής. Μετά, ανοίγουμε το IDE του OMNET++, αν δεν είναι ήδη ανοιχτό, και πατάμε File > Import > General: Existing Projects into Workspace και διαλέγοντας τον φάκελο που βρίσκεται το Veins.

5.2 Εκτέλεση προσομοίωσης

Τώρα που έχουν εγκατασταθεί όλα τα αρχεία, ανοίγουμε τη γραμμή εντολών MinGW εκτελώντας το αρχείο mingwenv.cmd. Σε αυτή τη γραμμή γράφουμε (dir)/sumo-1.8.0/bin/sumo.exe -c erlangen.sumo.cfg, όπου (dir) είναι η διεύθυνση του φακέλου, στον οποίο υπάρχει ο φάκελος του SUMO που κάναμε extract. Έπειτα, στο γραφικό περιβάλλον του OMNET++, πατάμε δεξί κλίκ στο πρότζεκτ που κάναμε import και πατάμε την επιλογή Build Project. Μόλις τελειώσει η διαδικασία αυτή, πατάμε πάλι δεξί κλίκ και αυτή τη φορά επιλέγουμε το Run As > OMNET++ Simulation.

Αν έχει γίνει σωστά η εγκατάσταση, θα ανοίξει το παράθυρο της προσομοίωσης. Εκεί, στο πρώτο pop-up παράθυρο που θα εμφανιστεί πατάμε "OK". Στο επόμενο παράθυρο που θα εμφανιστεί, βάζουμε έναν αριθμό από το 1-24. Κάθε αριθμός αντιστοιχεί σε διαφορετικό σενάριο προσομοίωσης και θα αναλυθεί στη συνέχεια. Μετά, εμφανίζεται άλλο ένα παράθυρο, στο οποίο η τιμή που θα βάλουμε καθορίζει το ID της πολυμεσικής πληροφορίας που θα ζητηθεί από τα RSU κατά τη διάρκεια της προσομοίωσης. Το πολυμεσικό περιεχόμενο που υπάρχει ήδη αποθηκευμένο στο Origin έχουν ID ίσο με 4 και 5. Το περιεχόμενο με ID 4 έχει ένα μόνο τμήμα, ενώ το περιεχόμενο με ID 5 έχει 3 τμήματα. Στο τελευταίο παράθυρο που θα εμφανιστεί, τοποθετούμε τον αριθμό των RSUs που θέλουμε να έχει η προσομοίωση. Υπάρχουν έτοιμες συντεταγμένες για μέχρι 20 RSUs, όμως αν θέλουμε παραπάνω, θα ζητηθούν σε pop-up οι συντεταγμένες των περίσσειων RSUs.

Τα έτοιμα σενάρια καθορίζουν τις εξής παραμέτρους: Στιγμή αιτήματος, Είδος μετρικής και Πολιτική κατανομής περιεχομένου. Με αυτή τη σειρά παραμέτρων, θα αντιστοιχίσουμε στη συνέχεια τον αριθμό σεναρίου με τις τιμές των παραμέτρων. Σημειώνεται εδώ ότι για τη μετρική ¨ML (Machine Learning)¨, πρέπει να τρέξει κάποιος από τους δύο αλγορίθμους μηχανικής μάθησης που βρίσκονται στα αρχεία KMeansML.py και AgglomML.py.

- **1**. 130, ML, Push.
- **2**. 130, ML, Pull.
- **3**. 130, Degree, Push.
- **4**. 130, Degree, Pull.
- **5**. 130, Closeness, Push.
- **6**. 130, Closeness, Pull.
- **7**. 130, Betwenness, Push.
- **8**. 130, Betweenness, Push.
- **9**. 210, ML, Push.
- **10**. 210, ML, Pull.
- **11**. 210, Degree, Push.

- **12**. 210, Degree, Pull.
- **13**. 210, Closeness, Push.
- **14**. 210, Closeness, Pull.
- **15**. 210, Betwenness, Push.
- **16**. 210, Betweenness, Push.
- **17**. 290, ML, Push.
- **18**. 290, ML, Pull.
- **19**. 290, Degree, Push.
- **20**. 290, Degree, Pull.
- **21**. 290, Closeness, Push.
- **22**. 290, Closeness, Pull.
- **23**. 290, Betwenness, Push.
- **24**. 290, Betweenness, Push.

Κεφάλαιο 6

Μετρήσεις

Στο κεφάλαιο αυτό θα παρουσιαστούν οι μετρήσεις που έγιναν για αυτή τη διπλωματική εργασία. Οι μετρήσεις αφορούν τον χρόνο απόκρισης (response time), δηλαδή τον χρόνο από τη δημιουργία του αιτήματος μέχρι τη λήψη της απάντησης. Επειδή έχουμε δύο είδη δεδομένων, μονοτμηματικά και πολυτμηματικά, έγιναν προσομοιώσεις και για τις δύο περιπτώσεις αυτές. Για τα πολυτμηματικά δεδομένα, πέρα από τον χρόνο απόκρισης, σημειώνεται και ο χρόνος που το όχημα που έκανε το αίτημα έλαβε το πρώτο τμήμα της απάντησης.

Γενικότερα, στις προσομοιώσεις μας, υπάρχουν έως 100 οχήματα που περιοδικά μπαίνουν στο δίκτυο και τη χρονική στιγμή 83 αρχίζουν να συμβαίνουν ατυχήματα στο δίκτυο. Τα ατυχήματα αυτά αλλάζουν τις διαδρομές που ακολουθούν τα οχήματα του δικτύου, αλλάζοντας την τοπολογία τους. Αξίζει να προστεθεί, ότι σε κάθε αποστολή μηνύματος υπάρχει μία τυχαία καθυστέρηση που ακολουθεί ομοιόμορφη κατανομή και βρίσκεται στο πεδίο (0.01, 0.5) δευτερόλεπτα, καθώς και άλλες πάγιες καθυστερήσεις. Ακόμα, υπάρχει περίπτωση απόρριψης ενός μηνύματος, η οποία επηρεάζεται από διάφορους παράγοντες υπάγονται στα πλαίσια της διπλωματικής.

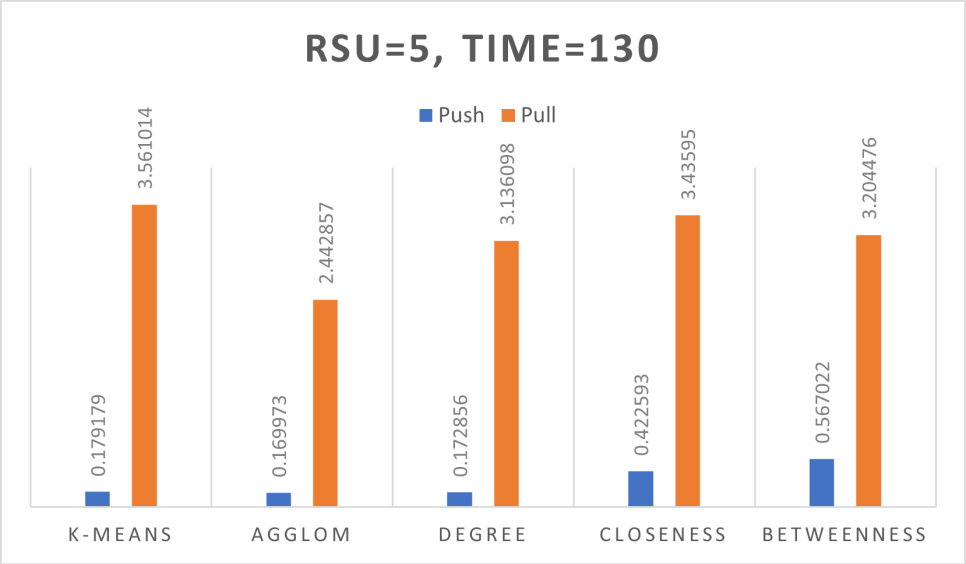
Για τους σκοπούς της συγκεκριμένης εργασίας, για κάθε μία από τις 2 κατηγορίες δεδομένων έγιναν 120 προσομοιώσεις (συνολικά 240). Αναλυτικότερα, στάλθηκαν αιτήματα σε 3 διαφορετικές χρονικές στιγμές από διαφορετικά οχήματα, έτσι ώστε να έχουμε διαφορετικές τοπολογίες και κατά συνέπεια διαφορετικά αποτελέσματα μετρικών. Η ακριβής τοποθεσία του κάθε οχήματος που έκανε το αίτημα είναι δύσκολο να προσδιοριστεί από πρίν λόγω των ατυχημάτων που συμβαίνουν και η ταυτότητα αυτών των 3 οχημάτων καθορίζεται μέσα στον κώδικα. Επίσης έγιναν προσομοιώσεις τόσο για Push όσο και για Pull πολιτικές διαμοιρασμού περιεχομένου και για κάθε μία από τις 3 κεντρικότητες και τους 2 αλγορίθμους μηχανικής μάθησης που υλοποιήθηκαν. Επιπλέον, η πολιτική caching που χρησιμοποιήθηκε ήταν η FIFO.

Οι μετρήσεις χωρίζονται ανά πλήθος RSU και χρονική στιγμή που έγινε η μέτρηση.

6.1 Μετρήσεις Μονοτμηματικού Περιεχομένου

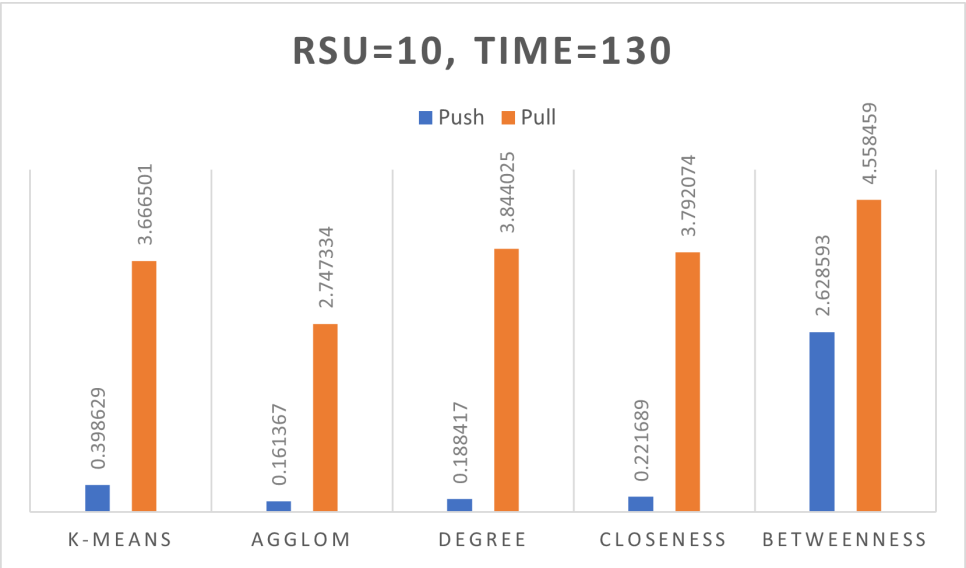
Σε αυτό το κεφάλαιο θα παρουσιαστούν οι χρόνοι απόκρισης για περιεχόμενο ενός τμήματος.

6.1.1 Μετρήσεις Χρονική Στιγμή 130



Σχήμα 6.1: *Αριθμός RSU = 5, Χρονική Στιγμή = 130*

Μπορούμε να δούμε ότι για αριθμό RSUs ίσο με 5, καλύτερη απόκριση και στις δύο πολιτικές έχει ο αλγόριθμος μηχανικής μάθησης Agglomerative Clustering.



Σχήμα 6.2: *Αριθμός RSU = 10, Χρονική Στιγμή = 130*

Εδώ, για RSU Count = 10, βλέπουμε πως ο Agglomerative Clustering έχει και πάλι την καλύτερη απόδοση τόσο στην πολιτική Push όσο και στην Pull.

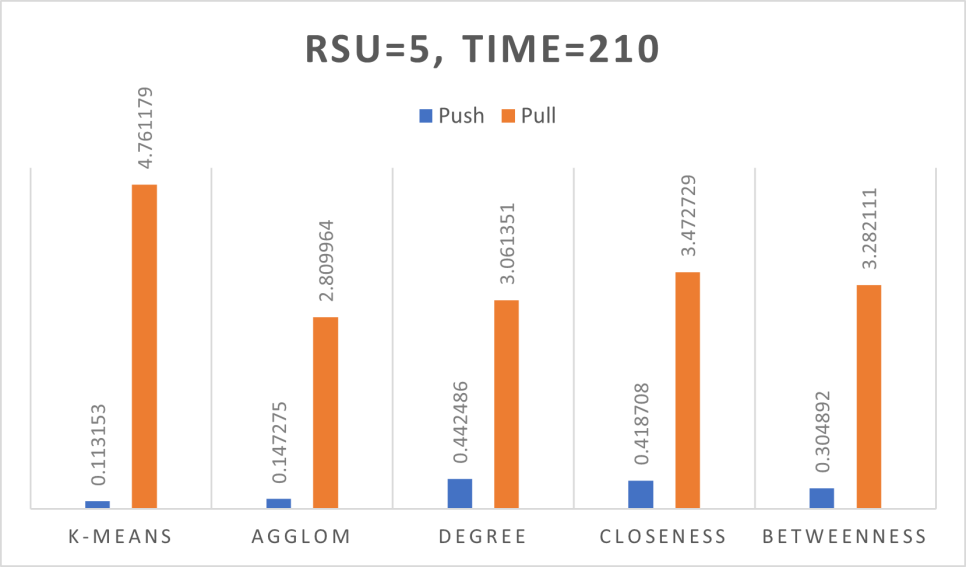
Σχήμα 6.3: *Αριθμός RSU = 15, Χρονική Στιγμή = 130*

Αυτή τη φορά, για αριθμό RSU ίσο με 15, καλύτερη απόδοση στην πολιτική Push έχει ο K-Means, ενώ στην Pull έχει ο Agglomerative Clustering.

Σχήμα 6.4: *Αριθμός RSU = 20, Χρονική Στιγμή = 130*

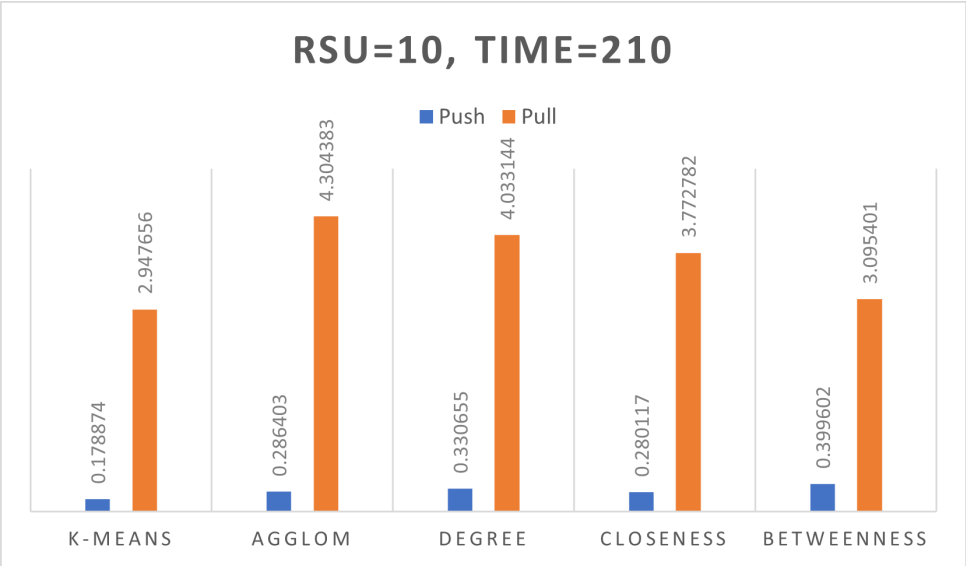
Στην περίπτωση που έχουμε 20 RSUs, καλύτερη συνολική απόδοση έχει η μετρική Closeness Centrality.

6.1.2 Μετρήσεις Χρονική Στιγμή 210



Σχήμα 6.5: *Αριθμός RSU = 5, Χρονική Στιγμή = 210*

Για τη χρονική στιγμή 210 και αριθμό RSU ίσο με 5, καλύτερη απόδοση στην πολιτική Push έχει ο αλγόριθμος μηχανικής μάθησης K-Means, ενώ στην πολιτική Pull έχει ο Agglomerative Clustering.



Σχήμα 6.6: *Αριθμός RSU = 10, Χρονική Στιγμή = 210*

Σε αυτή τη περίπτωση, που έχουμε 10 RSUs, καλύτερη απόδοση και για τις δύο πολιτικές έχει ο K-Means.

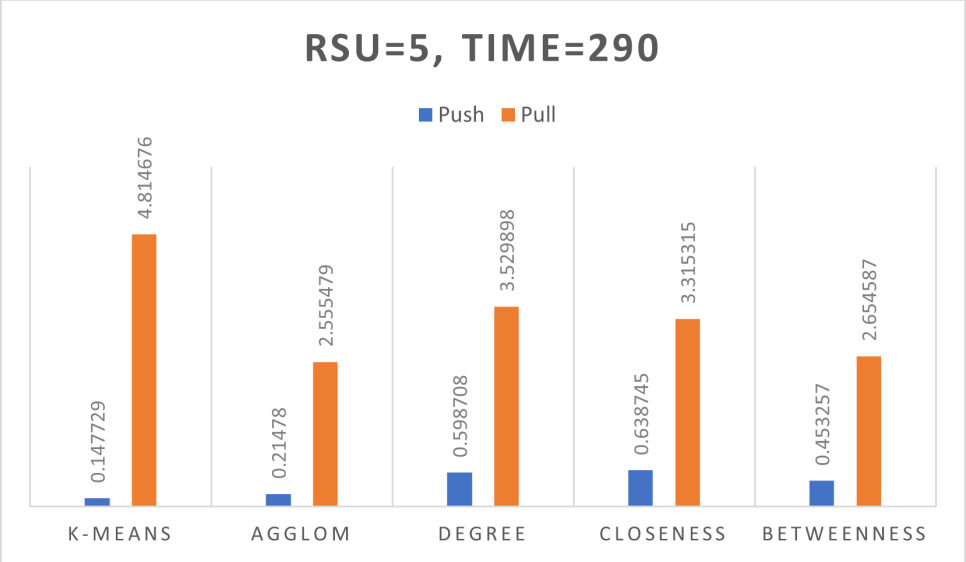
Σχήμα 6.7: *Αριθμός RSU = 15, Χρονική Στιγμή = 210*

Αν έχουμε 15 RSUs, καλύτερη απόδοση στο Push Policy έχει ο Agglomerative Clustering ενώ στο Pull έχει η μετρική Betweenness Centrality.

Σχήμα 6.8: *Αριθμός RSU = 20, Χρονική Στιγμή = 210*

Για 20 RSUs, καλύτερη απόδοση έχει η μετρική Closeness Centrality.

6.1.3 Μετρήσεις Χρονική Στιγμή 290

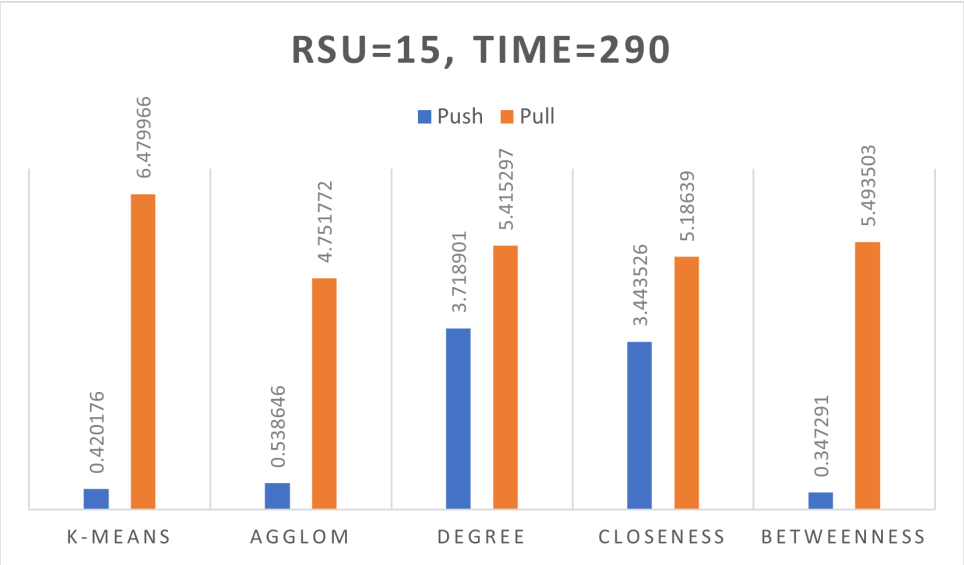


Σχήμα 6.9: *Αριθμός RSU = 5, Χρονική Στιγμή = 290*

Όσον αφορά την περίπτωση των 5 RSUs, βλέπουμε ότι καλύτερη απόδοση στην πολιτική Push έχει ο K-Means, ενώ για την Pull έχει ο Agglomerative Clustering.

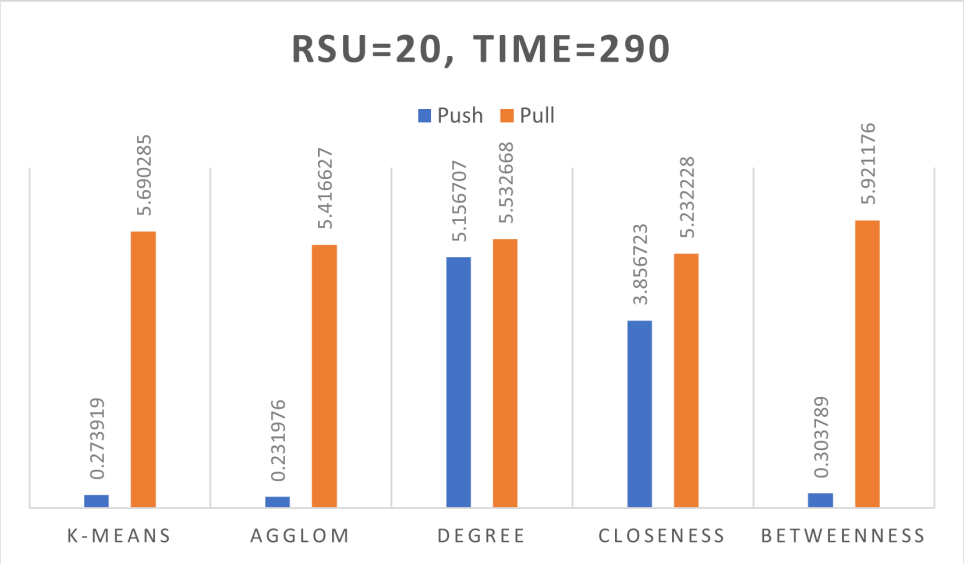
Σχήμα 6.10: *Αριθμός RSU = 10, Χρονική Στιγμή = 290*

Αν έχουμε 10 RSUs, βλέπουμε ότι καλύτερη απόδοση και για τις δύο πολιτικές έχει ο K-Means. Σημειώνεται η πολύ κακή απόδοση του Degree Centrality για την πολιτική Push.



Σχήμα 6.11: *Αριθμός RSU = 15, Χρονική Στιγμή = 290*

Για 15 RSUs, παρατηρούμε πως καλύτερη απόδοση στην πολιτική Push έχει το Betweenness Centrality, ενώ στην πολιτική Pull έχει ο Agglomerative Clustering.



Σχήμα 6.12: *Αριθμός RSU = 20, Χρονική Στιγμή = 290*

Στα 20 RSUs, σημειώνουμε πως καλύτερη απόδοση για την πολιτική Push έχει ο Agglomerative Clustering, ενώ στην πολιτική Pull έχει το Closeness Centrality. Ταυτόχρονα, πάλι βλέπουμε ότι το Degree Centrality έχει πολύ κακή απόδοση στην πολιτική Push.

6.2 Πίνακες Μετρήσεων Μονοτμηματικού Περιεχομένου

Σε αυτή την ενότητα θα τοποθετηθούν οι πίνακες των μετρήσεων. Η στήλη Timestamp 1 είναι η ακριβής χρονική στιγμή που γίνεται το αίτημα, η Timestamp 2 είναι πότε λήφθηκε το πρώτο τμήμα και η Completion πότε λήφθηκε το τελευταίο. Προφανώς, αφού είναι

μονοτμηματικό περιεχόμενο, οι δύο χρόνοι αυτοί ταυτίζονται.

RSU Count	ScenarioId	Algorithm	Timestamp1	Timestamp2	Completion	OriginPolicy	Timestamp
5	1	k-means	135.227848	135.407027	135.407027	Push	130
5	2	k-means	135.227848	138.788862	138.788862	Pull	130
5	1	Agglom	135.227848	134.497821	134.497821	Push	130
5	2	Agglom	135.227848	136.770705	136.770705	Pull	130
10	1	k-means	135.110682	135.509311	135.509311	Push	130
10	2	k-means	135.110682	138.777183	138.777183	Pull	130
10	1	Agglom	135.110682	134.372049	134.372049	Push	130
10	2	Agglom	135.110682	136.958016	136.958016	Pull	130
15	1	k-means	135.219651	135.549486	135.549486	Push	130
15	2	k-means	135.219651	138.728065	138.728065	Pull	130
15	1	Agglom	135.219651	135.055936	135.055936	Push	130
15	2	Agglom	135.219651	137.181097	137.181097	Pull	130
20	1	k-means	135.326525	135.777489	135.777489	Push	130
20	2	k-means	135.326525	139.371132	139.371132	Pull	130
20	1	Agglom	135.326525	134.846137	134.846137	Push	130
20	2	Agglom	135.326525	137.053126	137.053126	Pull	130

Σχήμα 6.13: *Χρονική Στιγμή = 130, Μηχανική Μάθηση*

RSU Count	ScenarioId	Timestamp1	Timestamp2	Completion	SimType	Centrality	OriginPolic	Timestamp
5	3	135.026122	135.198978	135.198978	Manual	Degree	Push	130
5	4	135.026122	138.16222	138.16222	Manual	Degree	Pull	130
5	5	135.026122	135.448715	135.448715	Manual	Closeness	Push	130
5	6	135.026122	138.462072	138.462072	Manual	Closeness	Pull	130
5	7	135.026122	135.593144	135.593144	Manual	Betweenness	Push	130
5	8	135.026122	138.230598	138.230598	Manual	Betweenness	Pull	130
10	3	135.110682	135.299099	135.299099	Manual	Degree	Push	130
10	4	135.110682	138.954707	138.954707	Manual	Degree	Pull	130
10	5	135.110682	135.332371	135.332371	Manual	Closeness	Push	130
10	6	135.110682	138.902756	138.902756	Manual	Closeness	Pull	130
10	7	135.110682	137.739275	137.739275	Manual	Betweenness	Push	130
10	8	135.110682	139.669141	139.669141	Manual	Betweenness	Pull	130
15	3	135.219651	135.969165	135.969165	Manual	Degree	Push	130
15	4	135.219651	139.498465	139.498465	Manual	Degree	Pull	130
15	5	135.219651	136.435184	136.435184	Manual	Closeness	Push	130
15	6	135.219651	139.025405	139.025405	Manual	Closeness	Pull	130
15	7	135.219651	135.672426	135.672426	Manual	Betweenness	Push	130
15	8	135.219651	139.009479	139.009479	Manual	Betweenness	Pull	130
20	3	135.326525	136.211879	136.211879	Manual	Degree	Push	130
20	4	135.326525	139.398211	139.398211	Manual	Degree	Pull	130
20	5	135.326525	135.51955	135.51955	Manual	Closeness	Push	130
20	6	135.326525	137.948453	137.948453	Manual	Closeness	Pull	130
20	7	135.326525	135.56435	135.56435	Manual	Betweenness	Push	130
20	8	135.326525	139.529335	139.529335	Manual	Betweenness	Pull	130

Σχήμα 6.14: *Χρονική Στιγμή = 130, Μετρικές Δικτύου*

RSU Count	ScenarioId	Algorithm	Timestamp1	Timestamp2	Completion	OriginPolicy	Timestamp
5	9	k-means	215.22266	215.335813	215.335813	Push	210
5	9	Agglom	215.388436	214.635711	214.635711	Push	210
5	10	k-means	215.218323	219.979502	219.979502	Pull	210
5	10	Agglom	215.16474	217.074704	217.074704	Pull	210
10	9	k-means	215.095455	215.274329	215.274329	Push	210
10	9	Agglom	215.204863	214.591266	214.591266	Push	210
10	10	k-means	215.380753	218.328409	218.328409	Pull	210
10	10	Agglom	215.380753	218.785136	218.785136	Pull	210
15	9	k-means	215.050778	215.576552	215.576552	Push	210
15	9	Agglom	215.13153	214.410709	214.410709	Push	210
15	10	k-means	215.133513	218.744147	218.744147	Pull	210
15	10	Agglom	215.133513	218.571459	218.571459	Pull	210
20	9	k-means	215.288252	215.621771	215.621771	Push	210
20	9	Agglom	215.021709	214.589758	214.589758	Push	210
20	10	k-means	215.290382	218.486158	218.486158	Pull	210
20	10	Agglom	215.290382	218.218563	218.218563	Pull	210

Σχήμα 6.15: *Χρονική Στιγμή = 210, Μηχανική Μάθηση*

RSU Count	ScenarioId	Timestamp1	Timestamp2	Completion	SimType	Centrality	OriginPolicy	Timestamp
5	11	215.281153	215.723639	215.723639	Manual	Degree	Push	210
5	12	215.424799	218.48615	218.48615	Manual	Degree	Pull	210
5	13	215.069806	215.488514	215.488514	Manual	Closeness	Push	210
5	14	215.171433	218.644162	218.644162	Manual	Closeness	Pull	210
5	15	215.355015	215.659907	215.659907	Manual	Betweenness	Push	210
5	16	215.264173	218.546284	218.546284	Manual	Betweenness	Pull	210
10	11	215.1067	215.437355	215.437355	Manual	Degree	Push	210
10	12	215.16908	219.202224	219.202224	Manual	Degree	Pull	210
10	13	215.114693	215.39481	215.39481	Manual	Closeness	Push	210
10	14	215.249485	219.022267	219.022267	Manual	Closeness	Pull	210
10	15	215.481721	215.881323	215.881323	Manual	Betweenness	Push	210
10	16	215.421745	218.517146	218.517146	Manual	Betweenness	Pull	210
15	11	215.281864	215.762587	215.762587	Manual	Degree	Push	210
15	12	215.067621	220.099817	220.099817	Manual	Degree	Pull	210
15	13	215.203115	215.568235	215.568235	Manual	Closeness	Push	210
15	14	215.272021	218.392081	218.392081	Manual	Closeness	Pull	210
15	15	215.416407	215.892004	215.892004	Manual	Betweenness	Push	210
15	16	215.076768	218.062789	218.062789	Manual	Betweenness	Pull	210
20	16	215.207822	217.915393	217.915393	Manual	Betweenness	Pull	210
20	15	215.337687	215.906833	215.906833	Manual	Betweenness	Push	210
20	14	215.352161	217.906123	217.906123	Manual	Closeness	Pull	210
20	13	215.352464	215.652753	215.652753	Manual	Closeness	Push	210
20	12	215.018828	220.644723	220.644723	Manual	Degree	Pull	210
20	11	215.388279	218.353557	218.353557	Manual	Degree	Push	210

Σχήμα 6.16: *Χρονική Στιγμή = 210, Μετρικές Δικτύων*

RSU Count	ScenarioId	Algorithm	Timestamp1	Timestamp2	Completion	OriginPolicy	Timestamp
5	18	k-means	295.360437	300.175113	300.175113	Pull	290
5	17	k-means	295.264073	295.411802	295.411802	Push	290
5	17	Agglom	295.405979	294.720759	294.720759	Push	290
5	18	Agglom	295.360437	297.015916	297.015916	Pull	290
10	17	k-means	295.266858	295.390611	295.390611	Push	290
10	18	k-means	295.34109	299.387622	299.387622	Pull	290
10	17	Agglom	295.124855	294.369012	294.369012	Push	290
10	18	Agglom	295.34109	298.75843	298.75843	Pull	290
15	18	k-means	295.06268	301.542646	301.542646	Pull	290
15	17	k-means	295.015546	295.435722	295.435722	Push	290
15	17	Agglom	295.06014	294.698786	294.698786	Push	290
15	18	Agglom	295.06268	298.914452	298.914452	Pull	290
20	18	k-means	295.087086	300.777371	300.777371	Pull	290
20	17	k-means	295.347957	295.621876	295.621876	Push	290
20	17	Agglom	295.333661	294.665637	294.665637	Push	290
20	18	Agglom	295.087086	299.603713	299.603713	Pull	290

Σχήμα 6.17: *Χρονική Στιγμή = 290, Μηχανική Μάθηση*

RSU Count	ScenarioId	Timestamp1	Timestamp2	Completion	SimType	Centrality	OriginPolic	Timestamp
5	19	295.493087	296.091795	296.091795	Manual	Degree	Push	290
5	20	295.317832	298.84773	298.84773	Manual	Degree	Pull	290
5	21	295.36667	296.005415	296.005415	Manual	Closeness	Push	290
5	22	295.487697	298.803012	298.803012	Manual	Closeness	Pull	290
5	23	295.041611	295.494868	295.494868	Manual	Betweenness	Push	290
5	24	295.240478	297.895065	297.895065	Manual	Betweenness	Pull	290
10	19	295.1067	299.783103	299.783103	Manual	Degree	Push	290
10	20	295.16908	300.356742	300.356742	Manual	Degree	Pull	290
10	21	295.114693	298.700645	298.700645	Manual	Closeness	Push	290
10	22	295.249485	300.305734	300.305734	Manual	Closeness	Pull	290
10	23	295.481721	295.863752	295.863752	Manual	Betweenness	Push	290
10	24	295.421745	300.58462	300.58462	Manual	Betweenness	Pull	290
15	19	295.281864	299.000765	299.000765	Manual	Degree	Push	290
15	20	295.067621	300.482918	300.482918	Manual	Degree	Pull	290
15	21	295.203115	298.646641	298.646641	Manual	Closeness	Push	290
15	22	295.272021	300.458411	300.458411	Manual	Closeness	Pull	290
15	23	295.416407	295.763698	295.763698	Manual	Betweenness	Push	290
15	24	295.076768	300.570271	300.570271	Manual	Betweenness	Pull	290
20	19	295.360088	300.516795	300.516795	Manual	Degree	Push	290
20	20	295.060772	300.59344	300.59344	Manual	Degree	Pull	290
20	21	295.021921	298.878644	298.878644	Manual	Closeness	Push	290
20	22	295.153548	300.385776	300.385776	Manual	Closeness	Pull	290
20	23	295.405576	295.709365	295.709365	Manual	Betweenness	Push	290
20	24	295.136929	301.058105	301.058105	Manual	Betweenness	Pull	290

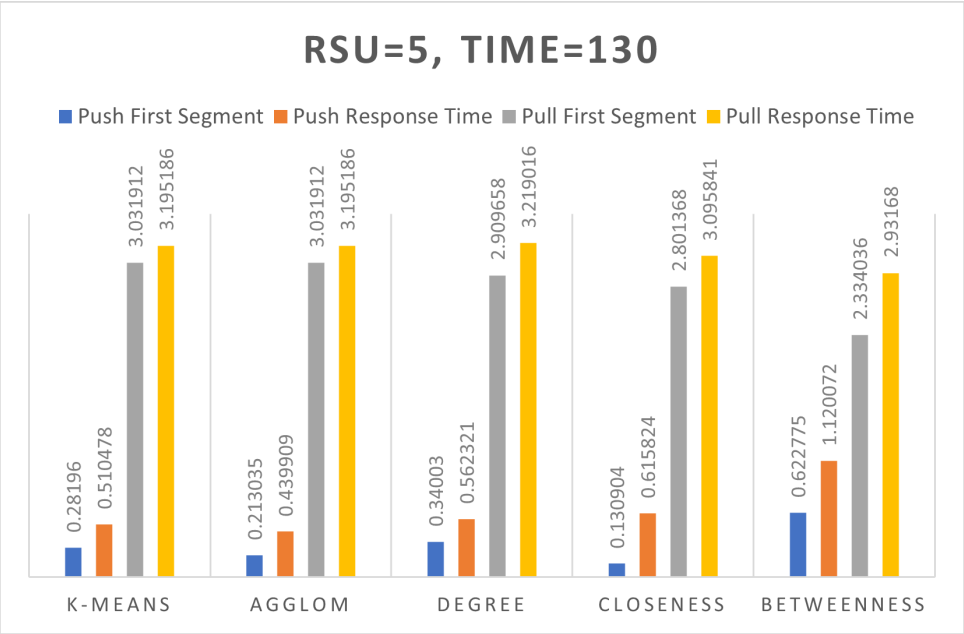
Σχήμα 6.18: *Χρονική Στιγμή = 290, Μετρικές Δικτύου*

6.3 Μετρήσεις Πολυτμηματικού Περιεχομένου

Σε αυτή την ενότητα θα σχολιάσουμε τους χρόνους απόκρισης πολυτμηματικού περιεχόμενου. Πρέπει να τονιστεί, ότι λόγω των τυχαίων καθυστερήσεων που υπάρχουν κατά την αποστολή των τμημάτων του περιεχομένου, ενδέχεται κάποια μετρική να έχει τον καλύτερο χρόνο απόκρισης για το πρώτο τμήμα, αλλά όχι τον καλύτερο χρόνο απόκρισης για ολόκληρο το μήνυμα. Επίσης, για λόγους ποικιλίας μετρήσεων, η πάγια καθυστέρηση που στέλνεται το αίτημα είναι διαφορετική σε σχέση με του μονοτμηματικού. Ως εκ τούτου, επειδή η χρονική

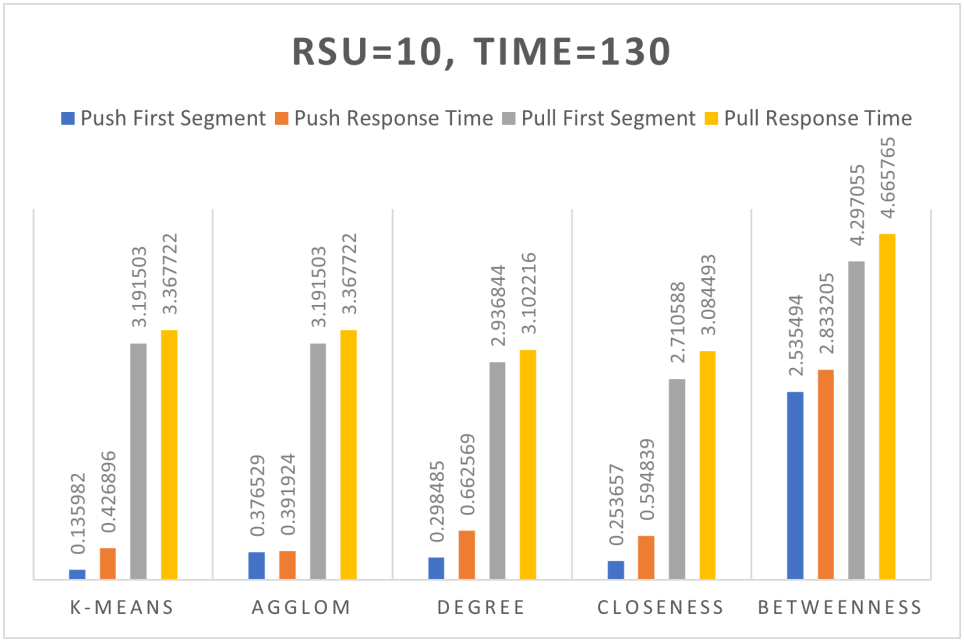
στιγμή θα διαφέρει λίγο, και εξαιτίας της τυχαιότητας, δεν περιμένουμε ίδια αποτελέσματα με τις προηγούμενες μετρήσεις.

6.3.1 Μετρήσεις Χρονική Στιγμή 130



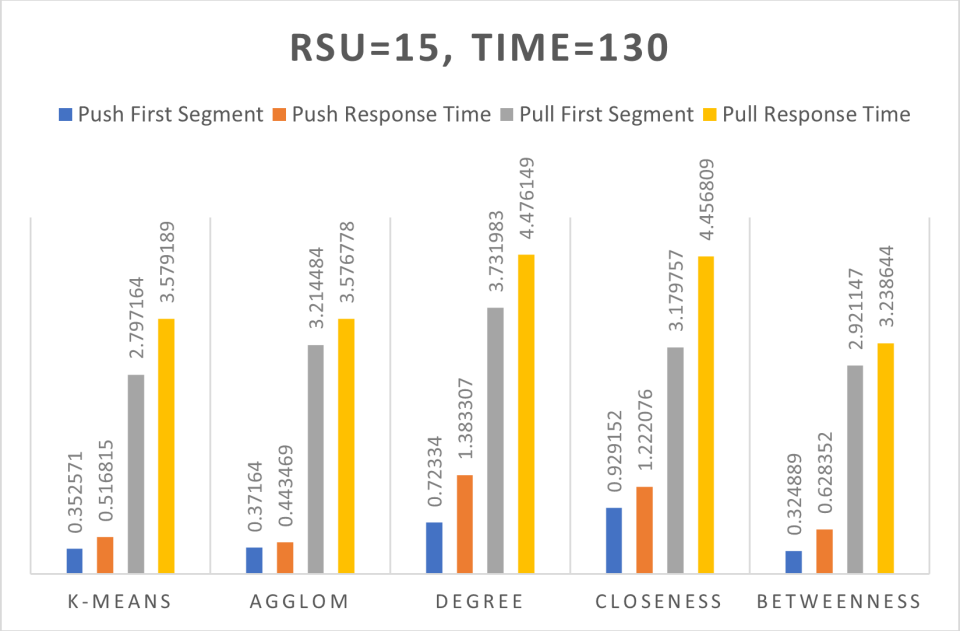
Σχήμα 6.19: *Αριθμός RSU = 5, Χρονική Στιγμή = 130*

Σε αυτή τη περίπτωση, βλέπουμε ότι για το πρώτο τμήμα, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει το Closeness Centrality, ενώ στην Pull έχει το Betweenness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει το Betweenness Centrality.



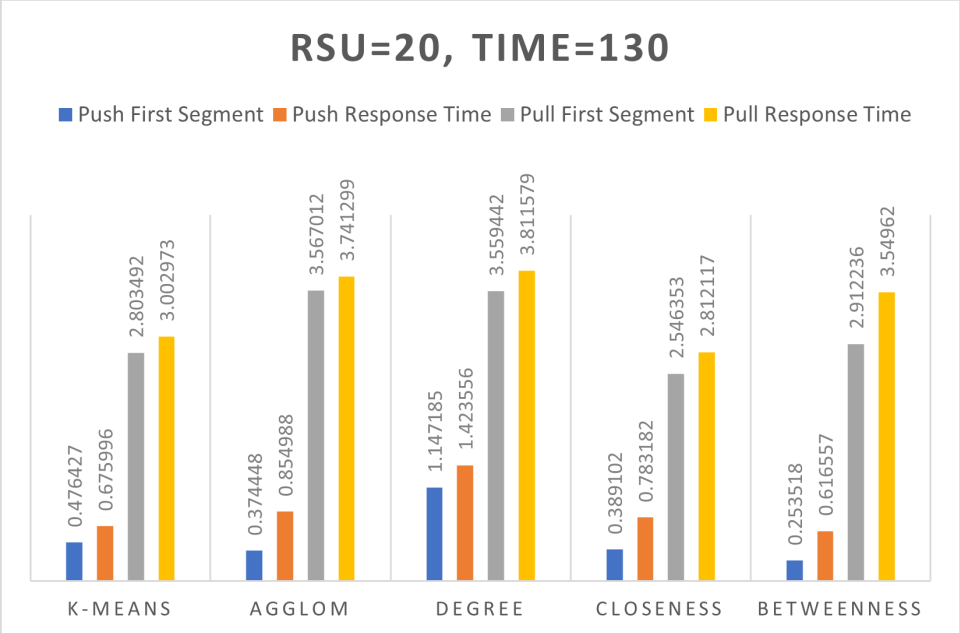
Σχήμα 6.20: *Αριθμός RSU = 10, Χρονική Στιγμή = 130*

Εδώ, παρατηρούμε ότι για το πρώτο τμήμα, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο K-Means, ενώ στην Pull έχει το Closeness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει πάλι το Closeness Centrality.



Σχήμα 6.21: *Αριθμός RSU = 15, Χρονική Στιγμή = 130*

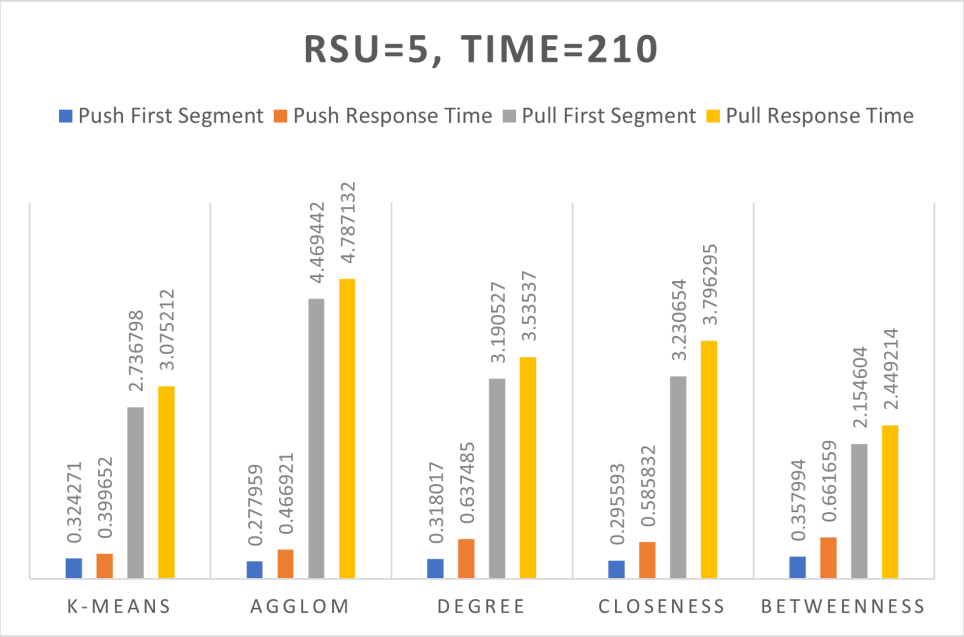
Στην περίπτωση των 15 RSUs, για το πρώτο τμήμα, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει το Betweenness Centrality, ενώ στην Pull έχει ο K-Means. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει το Betweenness Centrality.



Σχήμα 6.22: *Αριθμός RSU = 20, Χρονική Στιγμή = 130*

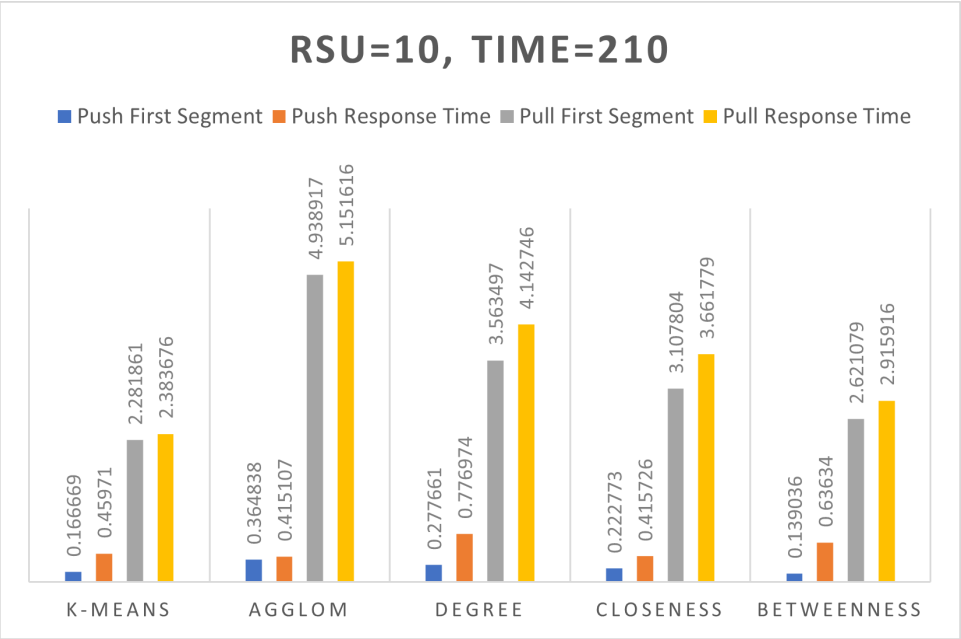
Εδώ, παρατηρούμε ότι για το πρώτο τμήμα, καλύτερος χρόνος απόδοσης στην πολιτική Push έχει ο K-Means, ενώ στην Pull έχει το Closeness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο K-Means, ενώ στην Pull έχει πάλι το Closeness Centrality.

6.3.2 Μετρήσεις Χρονική Στιγμή 210



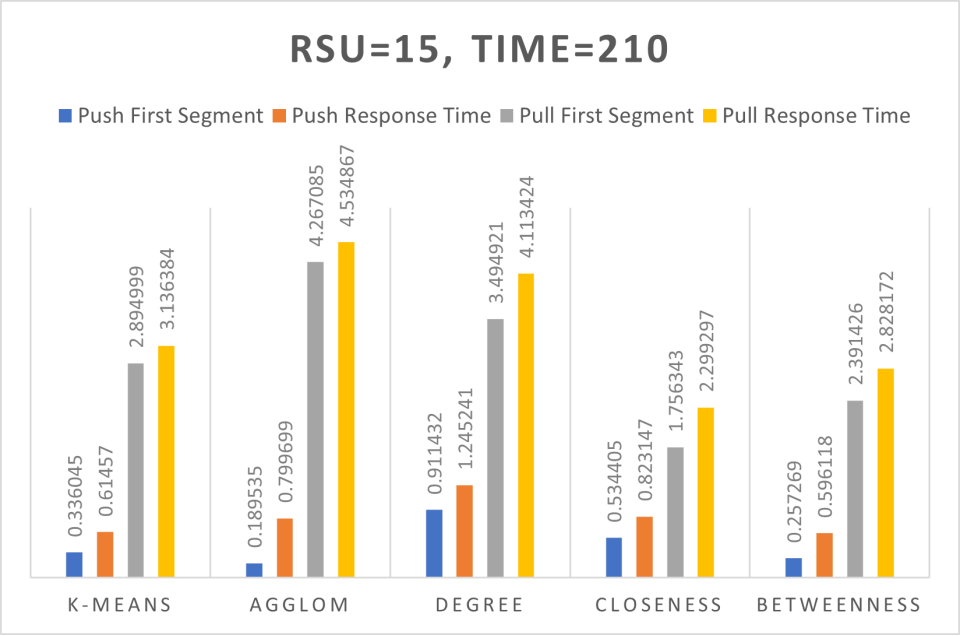
Σχήμα 6.23: *Αριθμός RSU = 5, Χρονική Στιγμή = 210*

Τη συγκεκριμένη χρονική στιγμή, για 5 RSUs, καλύτερη απόδοση για το πρώτο τμήμα έχει στο Push ο Agglomerative Clustering, ενώ στο Pull το Betweenness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο K-Means, ενώ για την Push έχει το Betweenness Centrality.



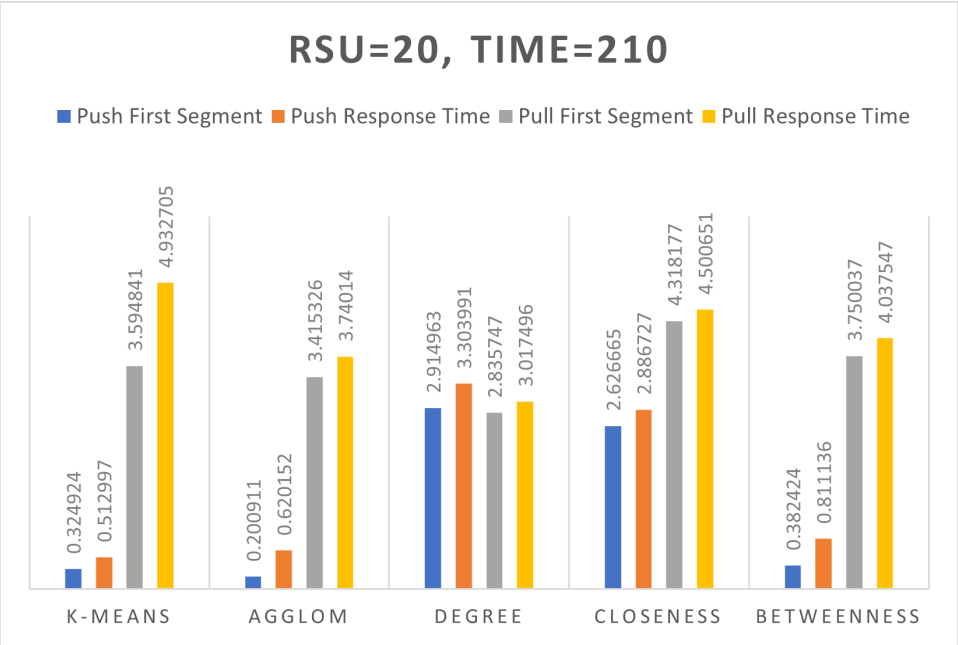
Σχήμα 6.24: *Αριθμός RSU = 10, Χρονική Στιγμή = 210*

Εδώ βλέπουμε ότι για το πρώτο τμήμα, καλύτερος χρόνος απόδοσης και στις δύο πολιτικές έχει το Betweenness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην πολιτική Pull έχει ο K-means.



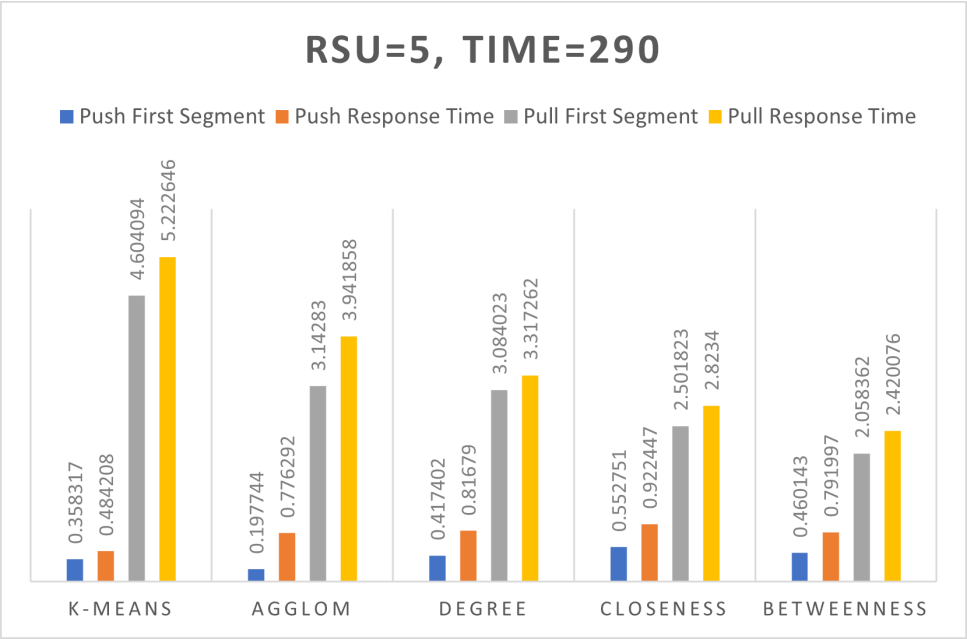
Σχήμα 6.25: *Αριθμός RSU = 15, Χρονική Στιγμή = 210*

Μπορούμε σε αυτή τη περίπτωση να παρατηρήσουμε ότι για το πρώτο τμήμα, καλύτερος χρόνος απόδοσης στην πολιτική Push έχει το Betweenness Centrality, ενώ στην Pull έχει το Closeness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο K-Means, ενώ στην Pull έχει πάλι το Closeness Centrality.



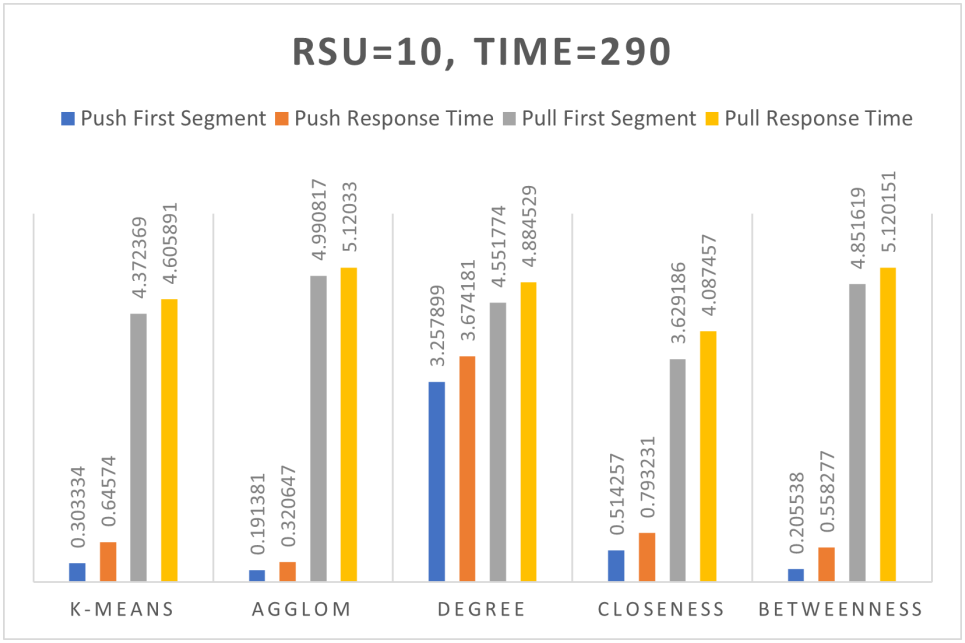
Σχήμα 6.26: *Αριθμός RSU = 20, Χρονική Στιγμή = 210*

Όσον αφορά την περίπτωση των 20 RSU τη χρονική στιγμή 210, για το πρώτο τμήμα, καλύτερος χρόνος απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει το Degree Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο K-Means, ενώ στην Pull έχει πάλι το Degree Centrality Αξίζει να σημειωθεί, ότι σε αυτή τη μέτρηση, μπορούμε να δούμε μία από τις μόνες περιπτώσεις πού οι χρόνοι απόκρισης του Push είναι μεγαλύτεροι από του Pull, στο Degree Centrality. Αυτό μπορεί να συμβαίνει για διάφορους λόγους, από τις τυχαίες καθυστερήσεις, τις απορρίψεις των μηνυμάτων ή την επιλογή κεντρικού RSU πολύ μακριά από τον κόμβο που ζήτησε το μήνυμα.



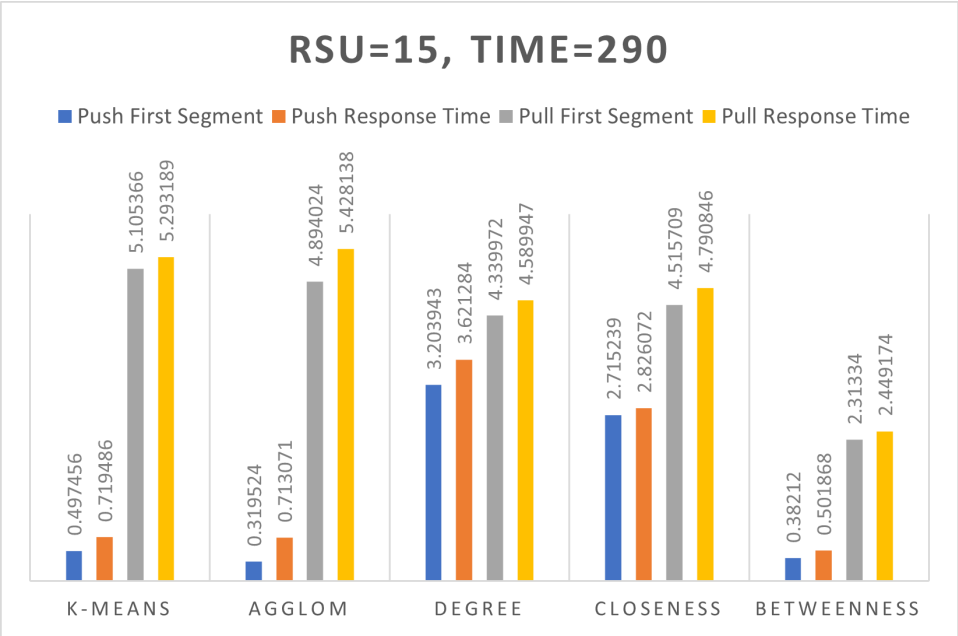
Σχήμα 6.27: *Αριθμός RSU = 5, Χρονική Στιγμή = 290*

Για 5 RSUs τη χρονική στιγμή 290, καλύτερος χρόνος απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει το Betweenness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης για τη πολιτική Push έχει ο K-Means και για την Pull έχει πάλι το Betweenness Centrality.



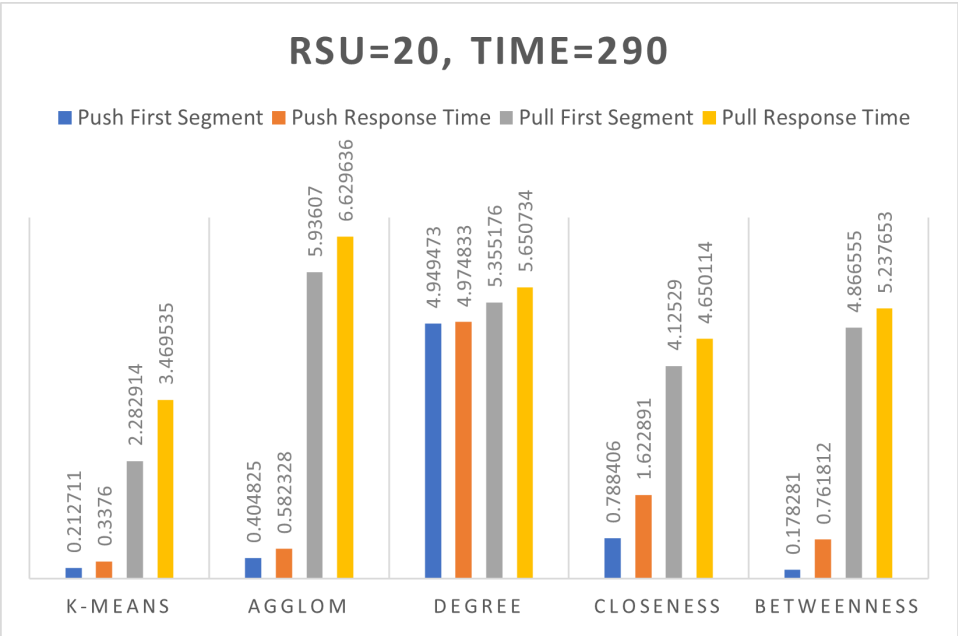
Σχήμα 6.28: *Αριθμός RSU = 10, Χρονική Στιγμή = 290*

Για 10 RSUs τη χρονική στιγμή 290, καλύτερος χρόνος απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει το Closeness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης στην πολιτική Push έχει ο Agglomerative Clustering, ενώ στην Pull έχει και πάλι το Closeness Centrality.



Σχήμα 6.29: *Αριθμός RSU = 15, Χρονική Στιγμή = 290*

Στην περίπτωση των 15 RSUs, καλύτερος χρόνος απόδοσης στη πολιτική Push έχει ο Agglomerative Clustering και στην Pull, το Betweenness Centrality. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης και στις δύο πολιτικές έχει το Betwenness Centrality.



Σχήμα 6.30: *Αριθμός RSU = 20, Χρονική Στιγμή = 290*

Στην τελευταία περίπτωση, καλύτερος χρόνος απόδοσης στην πολιτική Push έχει και εδώ το Betweenness Centrality, ενώ στην Pull έχει ο K-Means. Για την ολοκλήρωση του μηνύματος, καλύτερο χρόνο απόδοσης και στις δύο πολιτικές έχει ο K-Means.

6.4 Πίνακες Μετρήσεων Πολυτμηματικού Περιεχομένου

Σε αυτή την ενότητα θα τοποθετηθούν οι πίνακες των μετρήσεων πολυτμηματικού περιεχομένου. Όμοια με πρίν, η στήλη Timestamp 1 είναι η ακριβής χρονική στιγμή που γίνεται το αίτημα, η Timestamp 2 είναι πότε λήφθηκε το πρώτο τμήμα και η Completion πότε λήφθηκε το τελευταίο.

RSU Count	ScenarioId	Centrality	Timestamp1	Timestamp2	Completion	OriginPolicy	Timestamp
5	1	k-means	130.227848	130.509808	130.738326	Push	130
5	1	Agglom	130.227848	130.440883	130.667757	Push	130
5	2	k-means	130.227848	133.25976	133.423034	Pull	130
5	2	Agglom	130.227848	133.25976	133.423034	Pull	130
10	1	k-means	130.110682	130.246664	130.537578	Push	130
10	1	Agglom	130.110682	130.487211	130.502606	Push	130
10	2	k-means	130.110682	133.302185	133.478404	Pull	130
10	2	Agglom	130.110682	133.302185	133.478404	Pull	130
15	1	k-means	130.219651	130.572222	130.736466	Push	130
15	1	Agglom	130.219651	130.591291	130.66312	Push	130
15	2	k-means	130.219651	133.016815	133.79884	Pull	130
15	2	Agglom	130.219651	133.434135	133.796429	Pull	130
20	1	k-means	130.326525	130.802952	131.002521	Push	130
20	1	Agglom	130.326525	130.700973	131.181513	Push	130
20	2	k-means	130.326525	133.130017	133.329498	Pull	130
20	2	Agglom	130.326525	133.893537	134.067824	Pull	130

Σχήμα 6.31: *Χρονική Στιγμή = 130, Μηχανική Μάθηση*

RSU Count	ScenarioId	Timestamp1	Timestamp2	Completion	SimType	Centrality	OriginPolicy	Timestamp
5	3	130.31759	130.65762	130.879911	Manual	Degree	Push	130
5	4	130.026122	132.93578	133.245138	Manual	Degree	Pull	130
5	5	130.31759	130.448494	130.933414	Manual	Closeness	Push	130
5	6	130.31759	133.118958	133.413431	Manual	Closeness	Pull	130
5	7	130.026122	130.648897	131.146194	Manual	Betweenness	Push	130
5	8	130.31759	132.651626	133.24927	Manual	Betweenness	Pull	130
10	3	130.110682	130.409167	130.773251	Manual	Degree	Push	130
10	4	130.110682	133.047526	133.212898	Manual	Degree	Pull	130
10	5	130.110682	130.364339	130.705521	Manual	Closeness	Push	130
10	6	130.110682	132.82127	133.195175	Manual	Closeness	Pull	130
10	7	130.110682	132.646176	132.943887	Manual	Betweenness	Push	130
10	8	130.110682	134.407737	134.776447	Manual	Betweenness	Pull	130
15	3	130.219651	130.942991	131.602958	Manual	Degree	Push	130
15	4	130.219651	133.951634	134.6958	Manual	Degree	Pull	130
15	5	130.219651	131.148803	131.441727	Manual	Closeness	Push	130
15	6	130.219651	133.399408	134.67646	Manual	Closeness	Pull	130
15	7	130.219651	130.54454	130.848003	Manual	Betweenness	Push	130
15	8	130.219651	133.140798	133.458295	Manual	Betweenness	Pull	130
20	3	130.326525	131.47371	131.750081	Manual	Degree	Push	130
20	4	130.326525	133.885967	134.138104	Manual	Degree	Pull	130
20	5	130.326525	130.715627	131.109707	Manual	Closeness	Push	130
20	6	130.326525	132.872878	133.138642	Manual	Closeness	Pull	130
20	7	130.326525	130.580043	130.943082	Manual	Betweenness	Push	130
20	8	130.326525	133.238761	133.876145	Manual	Betweenness	Pull	130

Σχήμα 6.32: *Χρονική Στιγμή = 130, Μετρικές Δικτύων*

RSU Count	ScenarioId	Centrality	Timestamp1	Timestamp2	Completion	OriginPolicy	Timestamp
5	9	k-means	210.22266	210.546931	210.622312	Push	210
5	9	Agglom	210.388436	210.666395	210.855357	Push	210
5	10	k-means	210.218323	212.955121	213.293535	Pull	210
5	10	Agglom	210.16474	214.634182	214.951872	Pull	210
10	9	k-means	210.095455	210.262124	210.555165	Push	210
10	9	Agglom	210.204863	210.569701	210.61997	Push	210
10	10	k-means	210.380753	212.662614	212.764429	Pull	210
10	10	Agglom	210.380753	215.31967	215.532369	Pull	210
15	9	k-means	210.050778	210.386823	210.665348	Push	210
15	9	Agglom	210.13153	210.321065	210.931229	Push	210
15	10	k-means	210.133513	213.028512	213.269897	Pull	210
15	10	Agglom	210.133513	214.400598	214.66838	Pull	210
20	9	k-means	210.288252	210.613176	210.801249	Push	210
20	9	Agglom	210.021709	210.22262	210.641861	Push	210
20	10	k-means	210.290382	213.885223	215.223087	Pull	210
20	10	Agglom	210.290382	213.705708	214.030522	Pull	210

Σχήμα 6.33: *Χρονική Στιγμή = 210, Μηχανική Μάθηση*

RSU Count	ScenarioId	Timestamp1	Timestamp2	Completion	SimType	Centrality	OriginPolicy	Timestamp
5	11	210.362032	210.680049	210.999517	Manual	Degree	Push	210
5	12	210.424799	213.615326	213.960169	Manual	Degree	Pull	210
5	13	210.303829	210.599422	210.889661	Manual	Closeness	Push	210
5	14	210.367842	213.598496	214.164137	Manual	Closeness	Pull	210
5	15	210.355015	210.713009	211.016674	Manual	Betweenness	Push	210
5	16	210.03119	212.185794	212.480404	Manual	Betweenness	Pull	210
10	11	210.1067	210.384361	210.883674	Manual	Degree	Push	210
10	12	210.16908	213.732577	214.311826	Manual	Degree	Pull	210
10	13	210.418377	210.64115	210.834103	Manual	Closeness	Push	210
10	14	210.249485	213.357289	213.911264	Manual	Closeness	Pull	210
10	15	210.025524	210.16456	210.661864	Manual	Betweenness	Push	210
10	16	210.203507	212.824586	213.119423	Manual	Betweenness	Pull	210
15	11	210.281864	211.193296	211.527105	Manual	Degree	Push	210
15	12	210.067621	213.562542	214.181045	Manual	Degree	Pull	210
15	13	210.255492	210.789897	211.078639	Manual	Closeness	Push	210
15	14	210.342099	212.098442	212.641396	Manual	Closeness	Pull	210
15	15	210.416407	210.673676	211.012525	Manual	Betweenness	Push	210
15	16	210.076768	212.468194	212.90494	Manual	Betweenness	Pull	210
20	11	210.388279	213.303242	213.69227	Manual	Degree	Push	210
20	12	210.131565	212.967312	213.149061	Manual	Degree	Pull	210
20	13	210.115369	212.742034	213.002096	Manual	Closeness	Push	210
20	14	210.083572	214.401749	214.584223	Manual	Closeness	Pull	210
20	15	210.219221	210.601645	211.030357	Manual	Betweenness	Push	210
20	16	210.052704	213.802741	214.090251	Manual	Betweenness	Pull	210

Σχήμα 6.34: *Χρονική Στιγμή = 210, Μετρικές Δικτύων*

RSU Count	ScenarioId	Centrality	Timestamp1	Timestamp2	Completion	OriginPolicy	Timestamp
5	17	k-means	290.264073	290.62239	290.748281	Push	290
5	17	Agglom	290.405979	290.603723	291.182271	Push	290
5	18	k-means	290.360437	294.964531	295.583083	Pull	290
5	18	Agglom	290.360437	293.503267	294.302295	Pull	290
10	17	k-means	290.266858	290.570192	290.912598	Push	290
10	17	Agglom	290.124855	290.316236	290.445502	Push	290
10	18	k-means	290.34109	294.713459	294.946981	Pull	290
10	18	Agglom	290.34109	295.331907	295.46142	Pull	290
15	17	k-means	290.015546	290.513002	290.735032	Push	290
15	17	Agglom	290.06014	290.379664	290.773211	Push	290
15	18	k-means	290.06268	295.168046	295.355869	Pull	290
15	18	Agglom	290.06268	294.956704	295.490818	Pull	290
20	17	k-means	290.347957	290.560668	290.685557	Push	290
20	17	Agglom	290.333661	290.738486	290.915989	Push	290
20	18	k-means	290.087086	292.37	293.556621	Pull	290
20	18	Agglom	290.087086	296.023156	296.716722	Pull	290

Σχήμα 6.35: *Χρονική Στιγμή = 290, Μηχανική Μάθηση*

RSU Count	ScenarioId	Timestamp1	Timestamp2	Completion	SimType	Centrality	OriginPolicy	Timestamp
5	19	290.493087	290.910489	291.309877	Manual	Degree	Push	290
5	20	290.317832	293.401855	293.635094	Manual	Degree	Pull	290
5	21	290.36667	290.919421	291.289117	Manual	Closeness	Push	290
5	22	290.243847	292.74567	293.067247	Manual	Closeness	Pull	290
5	23	290.041611	290.501754	290.833608	Manual	Betweenness	Push	290
5	24	290.240478	292.29884	292.660554	Manual	Betweenness	Pull	290
10	19	290.1067	293.364599	293.780881	Manual	Degree	Push	290
10	20	290.16908	294.720854	295.053609	Manual	Degree	Pull	290
10	21	290.265816	290.780073	291.059047	Manual	Closeness	Push	290
10	22	290.249485	293.878671	294.336942	Manual	Closeness	Pull	290
10	23	290.481721	290.687259	291.039998	Manual	Betweenness	Push	290
10	24	290.421745	295.273364	295.541896	Manual	Betweenness	Pull	290
15	19	290.281864	293.485807	293.903148	Manual	Degree	Push	290
15	20	290.044817	294.384789	294.634764	Manual	Degree	Pull	290
15	21	290.391425	293.106664	293.217497	Manual	Closeness	Push	290
15	22	290.432674	294.948383	295.22352	Manual	Closeness	Pull	290
15	23	290.467708	290.849828	290.969576	Manual	Betweenness	Push	290
15	24	290.407534	292.720874	292.856708	Manual	Betweenness	Pull	290
20	19	290.221832	295.171305	295.196665	Manual	Degree	Push	290
20	20	290.410505	295.765681	296.061239	Manual	Degree	Pull	290
20	21	290.021921	290.810327	291.644812	Manual	Closeness	Push	290
20	22	290.153548	294.278838	294.803662	Manual	Closeness	Pull	290
20	23	290.3475	290.525781	291.109312	Manual	Betweenness	Push	290
20	24	290.136929	295.003484	295.374582	Manual	Betweenness	Pull	290

Σχήμα 6.36: *Χρονική Στιγμή = 290, Μετρικές Δικτύων*

Μέρος III

Επίλογος

Κεφάλαιο 7

Επίλογος

7.1 Συμπεράσματα

Εν κατακλείδι, οι μετρήσεις που παρατέθηκαν παραπάνω και η σύγκρισή τους ήταν ο κύριος σκοπός της πτυχιακής αυτής. Για να παρθούν οι μετρήσεις, έπρεπε να επεκταθούν οι λειτουργίες του προσομοιωτή για να μπορεί να γίνει δυνατή η αποστολή πολυμεσικού περιεχομένου και ο υπολογισμός διαφόρων μετρικών. Ταυτόχρονα χρησιμοποιήθηκαν και αλγόριθμοι μηχανικής μάθησης και συγκρίναμε την απόδοσή τους με τις μετρικές.

Από τα παραπάνω διαγράμματα, δεν μπορούμε να πάρουμε κάποιο γενικευμένο συμπέρασμα για την πιο αποτελεσματική μετρική, καθώς οι παράμετροι που αλλάζουν σε κάθε μέτρηση είναι πολλές. Κάποιοι από αυτούς του παράγοντες είναι οι τυχαίες καθυστερήσεις που προστίθενται κατά την αποστολή μηνύματος, η θέση του οχήματος που κάνει το αίτημα κάθε φορά και η πιθανότητα απόρριψης κάποιου μηνύματος. Παρόλα αυτά, μπορούμε να δούμε ότι με τη χρήση αλγορίθμων μηχανικής μάθησης, παίρνουμε χρόνους απόκρισης που είναι αρκετές φορές καλύτεροι από αυτούς που δίνουν οι κλασικές μετρικές ανάλυσης κοινωνικών δικτύων. Αυτό, σε συνδυασμό με τον ταχύτερο υπολογισμό που έχουμε με τους αλγορίθμους μηχανικής μάθησης, και την μειωμένη επιβάρυνση του δικτύου λόγω της μη ανάγκης εύρεσης ελάχιστων μονοπατιών (όπως στην περίπτωση των κεντρικοτήτων εγγύτητας και ενδιαμεσικότητας), καθιστά τους αλγορίθμους αυτούς σαν μία πολύ καλή λύση για την εύρεση κεντρικότερων RSU. Ταυτόχρονα, σημειώνεται ότι όσον αφορά τις πολιτικές μετάδοσης περιεχομένου Push και Pull, μπορούμε να παρατηρήσουμε ότι κατά κύριο λόγο, η πολιτική Push είναι αρκετά γρηγορότερη, όμως επιβαρύνει αρκετά το δίκτυο, αφού το περιεχόμενο στέλνεται προληπτικά στα κεντρικότερα RSUs ακόμα και αν δεν χρειαστεί ποτέ.

7.2 Μελλοντικές Επεκτάσεις

Η επέκταση του κώδικα του Veins που έγινε στην διπλωματική αυτή, επιτρέπει με την είσοδο των επιθυμητών παραμέτρων, την εύκολη εξαγωγή μίας προσομοίωσης. Παρόλα αυτά, υπάρχουν βελτιώσεις που θα μπορούσαν να γίνουν περαιτέρω, προκειμένου πρός να γίνει ακόμα πιο εύκολη η διαδικασία αυτή. Αναλυτικότερα, θα μπορούσε να αναπτυχθεί κάποια διεπαφή, έτσι ώστε να είναι ευκολότερη η μαζική εκτέλεση προσομοιώσεων με πιο αποδοτικό τρόπο. Ταυτόχρονα, θα μπορούσε σε αυτή τη διεπαφή να υπάρχουν επιλογές που να τροποποιούν ορισμένα χαρακτηριστικά της προσομοίωσης (όπως για παράδειγμα το

περιεχόμενο αποθηκευμένο στην αρχή της προσομοίωσης ή το ποιός κόμβος κάνει αίτημα), έτσι ώστε να μη χρειάζεται η άμεση επαφή του χρήστη με τον κώδικα.

Επιπλέον, μία ακόμη βελτίωση που θα μπορούσε να γίνει είναι η προσθήκη επιπλέων χαρακτηριστικών στα πολυμεσικά αρχεία. Πιο συγκεκριμένα, αυτή τη στιγμή τα πολυμεσικά αρχεία χαρακτηρίζονται από το ID, το περιεχόμενο και τα segments που χρειάζονται για να σταλθεί. Όμως, μπορούμε να δεδομένα όπως το αν πρόκειται για video ή εικόνα χάρτη, το μέγεθος του αρχείου και άλλα.

χαρακτηρισμό όπως φπς, ιδιαίτερα χαρακτηριστικά των πολυμεσικών αρχείων, π.χ. αν πρόκειται για ΐδεο/χάρτη, κλπ.), προσθήκη διεπαφής, αυτόματο τρέξιμο βατση προσομοιώσεων, κλπ.

Βιβλιογραφία

[1] *L'archive mise en réseau. La visualisation de données en sciences humaines est un moyen, pas une fin !* https://www.martingrandjean.ch/archive-reseau-visualisation-donnes-sciences-humaines/. Ημερομηνία πρόσβασης: 8-2-2023.

[2] *Volume 1: A Game of Thrones*. https://networkofthrones.wordpress.com/the-novels/a-game-of-thrones/. Ημερομηνία πρόσβασης: 8-2-2023.

[3] Y. Okabe και A. Shudo. *Microscopic Numerical Simulations of Epidemic Models on Networks*. *Mathematics*, 9:932, 2021.

[4] D. Easley και J. Kleinberg. *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*. Cambridge University Press, 2010.

[5] *Introduction à la visualisation de données : l'analyse de réseau en histoire*. https://www.martingrandjean.ch/introduction-visualisation-de-donnees-analyse-de-reseau-histoire/. Ημερομηνία πρόσβασης: 12-2-2023.

[6] A. Singh, A. Chawla και S. Surjeet. *Performance Issues and Behavioral Analysis of Routing Protocols in MANETs*. *International Journal of Communications, Network and System Sciences*, σελίδες 431–439, 2016.

[7] Z. Niu, Q. Li, C. Ma, H. Li, H. Shan και F. Yang. *Identification of Critical Nodes for Enhanced Network Defense in MANET-IoT Networks*. *IEEE Access*, 2020.

[8] *VANETs*. https://encyclopedia.pub/entry/8743. Ημερομηνία πρόσβασης: 16-2-2023.

[9] M. S. Obaidat, P. Nicopolitidis και F. Zarai. *Modeling and Simulation of Computer Networks and Systems: Methodologies and Applications*. Morgan Kaufman, 2015.

[10] *Reinforcement Learning 101*. https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292. Ημερομηνία πρόσβασης: 16-2-2023.

[11] *Screenshots*. https://omnetpp.org/intro/screenshots. Ημερομηνία πρόσβασης: 19-2-2023.

[12] *Documentation*. https://veins.car2x.org/documentation/. Ημερομηνία πρόσβασης: 19-2-2023.

[13] *What's the difference between a graph and a network?* https://bence.ferdinandy.com/2018/05/27/whats-the-difference-between-a-graph-and-a-network/. Ημερομηνία πρόσβασης: 8-2-2023.

[14] *History of Graph Theory*. https://prinsli.com/history-of-graph-theory/. Ημερομηνία πρόσβασης: 8-2-2023.

[15] A. Sousada Mata. *Complex Networks: a Mini-review*. *Brazilian Journal of Physics*, 50:658–672, 2020.

[16] *INTRODUCTION TO COMPLEX NETWORK ANALYSIS*. https://medium.com/analytics-vidhya/introduction-to-complex-network-analysis-15b50947a794/. Ημερομηνία πρόσβασης: 8-2-2023.

[17] M. W.H. Weenig. *Encyclopedia of Applied Psychology*. Elsevier, 2004.

[18] A. Nagurney και K. Ke. *Financial networks with intermediation*. *Quantitative Finance*, 1:441–451, 2010.

[19] *Social Network Analysis: Purpose Examples*. https://study.com/learn/lesson/social-network-analysis-purpose-examples.html. Ημερομηνία πρόσβασης: 8-2-2023.

[20] M. Krnc, J.S. Sereni, R. Skrekovski και Z.B. Yilma. *Eccentricity of Networks with Structural Constraints*. *Discussiones Mathematicae Graph Theory*, σελίδες 1141–1162, 2020.

[21] A. Kharrazi. *Resilience*. *Encyclopedia of Ecology (Second Edition)*, σελίδες 414–418, 2019.

[22] *Social network analysis 101: centrality measures explained*. https://cambridge-intelligence.com/keylines-faqs-social-network-analysis/. Ημερομηνία πρόσβασης: 8-2-2023.

[23] S. Bhardwaj, R. Niyogi και A. Milani. *Performance Analysis of an Algorithm for Computation of Betweenness Centrality*. *Computational Science and Its Applications*, σελίδα 537–546, 2011.

[24] *Katz Centrality (Centrality Measure)*. https://www.geeksforgeeks.org/katz-centrality-centrality-measure/. Ημερομηνία πρόσβασης: 12-2-2023.

[25] *Clustering Coefficient in Graph Theory*. https://www.geeksforgeeks.org/clustering-coefficient-graph-theory/. Ημερομηνία πρόσβασης: 12-2-2023.

[26] *transitivity*. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.cluster.transitivity.html#networkx.algorithms.cluster.transitivity. Ημερομηνία πρόσβασης: 13-2-2023.

[27] *What are Small-world Network Models?* https://towardsdatascience.com/what-are-small-world-network-models-87bbcfe0e038. Ημερομηνία πρόσβασης: 13-2-2023.

[28] H. Mehlhorn και F. Schreiber. *Small-World Property*. *Encyclopedia of Systems Biology*, σελίδα 957–1959, 2013.

[29] *Barabasi Albert Graph (for Scale Free Models)*. https://www.geeksforgeeks.org/barabasi-albert-graph-scale-free-models/. Ημερομηνία πρόσβασης: 14-2-2023.

[30] A. Noulas. *Social and Technological Network Analysis: Spatial Networks, Mobility and Applications*. Τεχνική Αναφορά με αριθμό, University of Cambridge, 2015.

[31] *random_geometric_graph*. https://networkx.org/documentation/stable/reference/generated/networkx.generators.geometric.random_geometric_graph.html. Ημερομηνία πρόσβασης: 13-2-2023.

[32] J. F. Buford, H. Yu και E. K. Lua. *P2P Networking and Applications*. Morgan Kaufman, 2009.

[33] Y. Do. *Centrality Analysis in Vehicular Ad Hoc Networks*. Τεχνική Αναφορά με αριθμό, EPFL, 2008.

[34] *Difference between MANET and VANET*. https://www.geeksforgeeks.org/difference-between-manet-and-vanet/?ref=rp. Ημερομηνία πρόσβασης: 16-2-2023.

[35] *Global Road Safety*. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-ml/. Ημερομηνία πρόσβασης: 16-2-2023.

[36] R. Mishra, A. Singh και R. Kumar. *VANET Security: Issues, Challenges and Solutions*. *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Gorakhpur, India, 2016.

[37] M. Ashraf, H. Bilal, I. A. Khan και F. Ahmad. *Vanet Challenges of Availability and Scalability*. *VFAST Transactions on Software Engineering*, 2016.

[38] *What is artificial intelligence (AI)?* https://www.ibm.com/topics/artificial-intelligence. Ημερομηνία πρόσβασης: 16-2-2023.

[39] *What Is Machine Learning? Definition, Types, Applications, and Trends for 2022*. https://www.cdc.gov/injury/features/global-road-safety/index.html. Ημερομηνία πρόσβασης: 16-2-2023.

[40] *What is supervised learning?* https://www.ibm.com/topics/supervised-learning. Ημερομηνία πρόσβασης: 16-2-2023.

[41] *A Beginner's Guide to Regression Analysis in Machine Learning*. https://towardsdatascience.com/a-beginners-guide-to-regression-analysis-in-machine-learning-8a828b491bbf. Ημερομηνία πρόσβασης: 16-2-2023.

[42] *What is unsupervised learning?* https://www.ibm.com/topics/unsupervised-learning. Ημερομηνία πρόσβασης: 16-2-2023.

[43] *Understanding K-means Clustering in Machine Learning*. https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1. Ημερομηνία πρόσβασης: 16-2-2023.

[44] *Supervised and Unsupervised learning*. https://www.geeksforgeeks.org/supervised-unsupervised-learning/. Ημερομηνία πρόσβασης: 16-2-2023.

[45] *How to Mitigate Overfitting with Dimensionality Reduction*. https://towardsdatascience.com/how-to-mitigate-overfitting-with-dimensionality-reduction-555b755b3d66. Ημερομηνία πρόσβασης: 16-2-2023.

[46] *Reinforcement learning*. https://www.geeksforgeeks.org/what-is-reinforcement-learning/. Ημερομηνία πρόσβασης: 16-2-2023.

[47] *How the Chosen ML Library and Algorithm Can Make or Mar Your Project?* https://bluewhaleapps.com/blog/how-the-chosen-ml-library-and-algorithm-can-make-or-mar-your-project. Ημερομηνία πρόσβασης: 19-2-2023.

[48] *What is OMNeT++?* https://omnetpp.org/intro/. Ημερομηνία πρόσβασης: 19-2-2023.

[49] *FAQ*. https://sumo.dlr.de/docs/FAQ.html. Ημερομηνία πρόσβασης: 19-2-2023.

[50] *Simulation of Urban MObility*. https://www.eclipse.org/sumo/. Ημερομηνία πρόσβασης: 19-2-2023.

[51] C. Sommer, R. German και F. Dressler. *Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis*. *IEEE Transactions on Mobile Computing*, 10, 2011.

[52] *Tutorial*. https://veins.car2x.org/tutorial/. Ημερομηνία πρόσβασης: 19-2-2023.

[53] *Simulation of Urban MObility Files*. https://sourceforge.net/projects/sumo/files/sumo/. Ημερομηνία πρόσβασης: 19-2-2023.

[54] *OMNeT++ Older Versions*. https://omnetpp.org/download/old. Ημερομηνία πρόσβασης: 19-2-2023.

[55] *Download*. https://veins.car2x.org/download/. Ημερομηνία πρόσβασης: 19-2-2023.